UNIVERSITY OF CATANIA

Department of Electrical, Electronic and Computer Engineering
Ph.D. in “Systems, Energy, Computer and Telecommunications Engineering”

CONTROL OF MULTICONSENSUS AND SYNCHRONIZATION IN MULTI-AGENT SYSTEMS: FROM THEORY TO ROBOTIC IMPLEMENTATION

Candidate:
Cinzia Tomaselli

Supervisors:
Prof. Mattia Frasca
Prof. Lucia V. Gambuzza

Cycle XXXVII

Acknowledgements

For this thesis, I would like to acknowledge several people. First, I would like to express my deepest gratitude to Professor Mattia Frasca and Professor Lucia Valentina Gambuzza for believing in me and allowing me to broaden my knowledge in the field of complex systems. They provided me with the opportunity not only to design theoretical models but also to conduct experiments with a team of Elisa-3 robots.

I am also thankful to Professor Sorrentino for his contributions to the development of the mathematical models for multiconsensus. I would like to extend a special thanks to Giuseppe Avon and Dario Calogero Guastella who assisted me with programming the Elisa-3 robots at the beginning of this project.

Part of this thesis was conducted at the University of Namur in collaboration with Professor Timoteo Carletti, to whom I am sincerely grateful. In this context, I also thank Professor Elio Tuci for the opportunity to work with the e-puck2 robots, and Gonzalo Marcelo Ramirez Avila for the three months spent together doing research.

Furthermore, I wish to acknowledge my coauthors Professor Stefano Boccaletti, Professor Ludovico Minati and Professor Giovanni Muscato, for their valuable contributions to works discussed here.

I would like to express a special thanks to my partner, Giuseppe, whose belief in me has been a constant source of strength. I hope that one day he will find fulfillment and will recognize his worth. I am also deeply grateful to my friends Bruno, Marco, Francesco, Roberto, and Lorenzo for their support along these years (and not only).

I extend my acknowledgement to my Lab-1 colleagues, including Minoo, Claudio, the Etnamatica company, the Robsys group, Luca, and Federico.

A special mention goes to Alessandra and Chiara, with whom I form the ”Superchicche” trio. I feel incredibly lucky to have had such wonderful colleagues, and together we’ve built unforgettable memories. From Namur, I would also like to thank Riccardo and Cédric for the pleasant conversations.

Lastly, I want to express my profound gratitude to my family for their unwavering support throughout this journey.

Contents

Chapter 1

Introduction

1.1 Background and state of the art

Multi-agent systems are of great interest within the scientific community for their ability to model systems across various domains, including biology, physics, economics, and engineering. In a multi-agent system, many agents, that can model diverse entities, from biological to robotic ones, interact and collaborate to achieve common goals or coordinate their actions. Due to their distributed nature, these systems ensure the achievement of collective behaviors that are both efficient and robust. In this context, a key phenomenon is *consensus*.

Consensus is achieved when the states of all units converge to a common value. This value may represent the position where robots of a team meet, the average value of a distributed measure, the temperature set point for a building with a centralized heating and conditioning system, and so on [68, 60]. It has been extensively studied across different agent dynamics. In particular, consensus has been first investigated for agents with single-integrator dynamics [83] and then extended to higher-order linear dynamics [81, 46, 117]. Protocols for consensus have been designed by using both undirected [83, 111, 110] and directed networks [82, 43]. Many of these works rely on algorithms based on the Laplacian matrix of the graph, hence characterizing the spectrum of this matrix is crucial [2]. Consensus dynamics has been also studied in the case of interaction networks that change in time or are affected by delays [59, 115, 69]. Other works have addressed fundamental issues such as actuator saturation [81, 98], robustness to uncertainties [49], and finite-time convergence [48]. In general, reaching a consensus in the shortest possible time is an important problem that has also been addressed by incorporating clustering techniques into the distributed algorithm used to control the agents [47].

While consensus requires all agents to converge to a single common value, more complex scenarios often require that different subgroups of agents each reach their consensus, leading to the concept of *multiconsensus*. This phenomenon is useful in all applications of multi-agent systems where the dynamics of the units is required to be differentiated into small subgroups. For instance, a team of robots may be split into smaller groups that perform several tasks, multiple rendezvous points may be assigned in an extended rendezvous problem, decision-making problems with groups adopting multiple solutions may be solved, and multiple distributed measurements with averaging restricted to subsets of sensors can be carried out using protocols for multiconsensus [4, 114, 109]. The first important question that has been investigated regards the conditions on the interaction network to achieve multiconsensus and the number and composition of groups. It has been found that diverse structural properties may yield multiconsensus. For instance, it occurs in graphs satisfying the in-degree balanced condition [114, 109, 79, 35, 55], admitting an external equitable partition [63, 29], or displaying symmetries [42, 52, 92]. Furthermore, multiconsensus can also be observed in special cases of graphs, such as signed and k-partite signed networks, when cooperative and antagonistic interactions are simultaneously present [17, 18], or conditions for interactive or structural balancing exist [120, 122]. Finally, in [71] the conditions for the emergence of multiconsensus are derived for varying structures networks characterized either by a finite or infinite set of possible configurations.

The understanding of the conditions for multiconsensus has enabled the development of control techniques to either modify the equilibrium value reached by each group or, in cases where the uncontrolled system does not display any multiconsensus, to affect the structure of the interactions to enforce one of the topological properties guaranteeing multiconsensus. Many techniques within the first class consider the application of control laws to a subset of the agents of the network. In more detail, they require to control, *i.e. to pin*, at least one node for each group of agents [54, 55]. The same approach can be adopted also for the control of multiconsensus in switching topologies [99, 79, 36, 39, 45], and extended to event-triggered or impulsive controllers which are useful in reducing the amount of communication required by the agents [112, 113]. Lastly, when the agents have no direct access to the state of the other units, then observer-based approaches can be used [38]. Instead, among the techniques acting on the structure of interactions, we find, for instance, the approach of [29], where, by using distributed controllers, the structure of interaction is modified with a minimum number of changes so that in the resulting graph the nodes can be clustered according to a desired external equitable partition.

Another very important and widespread phenomenon in multi-agent systems is *synchronization*, which refers to the alignment of internal states such as phase, frequency, amplitude, or even the complete trajectories of units, enabling them to operate in unison. Unlike consensus, typically associated with linear systems, synchronization encompasses a broader range of behaviors, applying also to nonlinear dynamics, including chaotic systems [78, 10, 11]. This phenomenon can be observed in natural systems, such as the coordinated flashing of fireflies [12]. But it also finds application in many engineered systems, such as the power grids in which the generators should be synchronized to guarantee a consistent energy supply [67].

Similarly to multiconsensus, *cluster synchronization* is a state wherein specific clusters of nodes within the network synchronize internally while exhibiting distinct dynamics from other clusters [91, 66, 41, 108, 75, 93]. The phenomenon is particularly relevant in systems that rely on parallel processing, such as brain networks [119]. It also finds application in engineered systems such as power grids, where different sections may operate in synchronized but distinct modes. Therefore, achieving full control of cluster synchronization is crucial for optimizing the performance of such systems. Despite the recognized importance of cluster synchronization, the development of control methods tailored for taming and regulating clustered states is currently lagging compared to techniques for controlling global synchronization. Several control strategies exist, indeed, for promoting or suppressing synchronization in entire networks [51, 118, 84, 116, 121]. In contrast, cluster synchronization approaches typically rely on determining the composition of the clusters based on the group of symmetries [30, 50, 24, 31], or the equitable partitions of the network nodes [28, 1, 88]. However, these approaches have not yet been extended to handle more generic or complex cluster-level dynamics.

Besides these phenomena, there are other complex dynamic behaviors of interest. Indeed, dynamics such as the face-to-face interaction and the response to synchronization provide additional understandings of the different ways in which collective behavior can emerge. However, these models often lack direct experimental validation, which is crucial as simulations frequently overlook real-world factors. Recent works have filled this gap by using robotic platforms, which not only allow for the validation of mathematical models but also enable experiments in settings where some parameters that are typically non-controllable can be tuned [107, 100, 77, 76, 19, 5, 6]. A pioneer work in this direction focuses on a special type of consensus dynamics, i.e., the naming game, and implements it on the Kilobot robotic platform [107]. The authors of the study analyze the effects of physical interference on the concurrent execution of the games and experimentally demonstrate the emergence of consensus in the naming process. Consensus dynamics has also been used to find the best-of-n solution through voter models using kilobot robots [15, 100]; interestingly, the robots can better adapt to environment changes when communication is constrained [100]. Nonequilibrium self-organization phenomena have also been studied with the help of robotic platforms, for instance, to test a predictive theory based on rattling [14]. Instead, in [77, 76] a swarm of e-puck robots is used to study the different regimes of synchronization that can appear in a system of mobile pulse-coupled oscillators as a function of agent speed, angle, and range of interaction. Real robot experiments have also been used to study the search efficiency and the ability to spread information within a swarm of random walkers [19]. The theoretical model of swarmalators that combines the synchronization and swarming dynamics has also been implemented in a robotic platform, in particular by using both small robots and drones [5, 6].

These works show how robotics can be useful for the study of complex systems. In fact, it provides a natural embodiment for theoretical models of complex systems [23, 96], and enables the experimental validation of the assumptions on the mechanisms underlying the system dynamical behavior, including the possibility of physical investigations with tunable parameters. However, the cross-fertilization between robotics and complexity [3] can also be beneficial for robotics, as algorithms drawn from the elementary principles of interactions unveiled in complex systems can be particularly useful for the control of multi-agent systems, where typically centralized methods are used instead [34]. An example is the distributed control of agents that sense and react to virtual forces inspired by natural physical laws in a framework enabling self-organization, fault-tolerance, and self-repair [94]. Another example is provided by the distributed control law introduced in [33]. This control law exploits agent elementary interactions and adaptation to design a strategy for geometric pattern formation that avoids the need for communication between agents, relying instead on the calculation of the displacement between the units that can be carried out with low sensor requirements. A third example is the self-organized and decentralized decision-making method that, being able to reach consensus on the fastest action, allows a swarm of robots to select the shortest of two paths [89].

1.2 Research contributions

In an increasingly interconnected world, coordinating large groups of units, such as robots, vehicles, or even biological entities, becomes crucial. This work is motivated by the need to develop robust, distributed, and practically implementable solutions for systems where agents interact through graphs of different natures, including undirected, directed, time-constant, and time-varying. Specifically, this thesis explores the role of agent interactions as a foundation for both controlled and emergent coordination in multi-agent systems.

In this context, the spectral properties of the interaction network are crucial, acting as tools for developing control strategies for the emergence of collective behaviors such as multiconsensus and synchronization through agent communication. In particular, the concept of eigenvector centrality forms the basis of our two communication protocols, which, unlike the traditional protocol relying on diffusive coupling, guide the system towards a subspace spanned by the leading eigenvector of the graph's adjacency matrix. This ensures that nodes with the same eigenvector centrality asymptotically achieve the same state. The main advantage of such protocols is that they provide a simple strategy to drive the whole multi-agent system to a desired multi-consensus solution. Indeed, to guide the system towards a specific solution, it is required a single control input acting on just a single node. Our first protocol leverages network symmetries, as symmetric nodes share the same eigenvector centrality, allowing multiconsensus to be achieved based on the inherent properties of the interaction network. Additionally, our approach, combined with [31], enables shaping cluster formations. However, while we can control the composition of each cluster, the multiconsensus solution remains constrained by the leading eigenvector of the adjacency matrix. Our second protocol addresses this limitation by allowing adjustments of interaction weights. This flexibility enables the selection of the leading eigenvector of the adjacency matrix, and thus, the eigenvector centralities of the nodes, broadening the range of achievable states for the system. Furthermore, we demonstrate that the spectral properties of the interaction network can be used to shape cluster synchronization dynamics. Specifically, through the introduction of additive control links, we can induce spectral blocks or shape the spectral properties of the groups of nodes associated with them. This approach not only allows the achievement of synchronization among clusters but also the control over the synchronizability of each cluster, which provides a significant impact on the system's collective dynamics.

We go beyond providing theoretical control strategies to induce collective behavior as we demonstrate the applicability of one of our mathematical models in a scenario that incorporates real-world factors. Specifically, we employed a team of robots (Elisa-3 and e-puck2 robots) to achieve a multiconsensus driven by the symmetries of the interaction graph through which they interact. These robotic platforms not only enable the validation of the theoretical models but also provide a controlled, scalable, and cost-effective platform for investigating collective behaviors, offering valuable insights into natural and engineered systems. In this context, scenarios in which coordination emerges from the local interactions among agents can be explored, emphasizing that phenomena such as aggregation or synchronization are driven by these interactions. For example, agents can form some group aggregation based on their reciprocal attractiveness, or agents carrying chaotic oscillators can achieve synchronization through local exchanges of state variables when close to each other, illustrating how local interaction can be leveraged to achieve coordination in complex settings. On the other hand, local interactions not only facilitate induced coordination but also allow investigation of the boundaries and possibilities of such phenomena. For instance, in scenarios where some agents cannot achieve synchronization due to their dynamics, introducing agents capable of synchronizing into the system can induce synchronization in others, highlighting the concept of the response to synchronization. In this context, the employment of the robotic platform allows the control of parameters like agent density to influence these emergent behaviors without directly altering the underlying dynamics or interaction networks.

Thus, integrating theoretical insights with robotic experimentation highlights the importance of multi-agent interactions in achieving both controlled and emergent coordination, demonstrating how local interactions can be exploited to influence collective behaviors in multi-agent systems.

1.3 Thesis structure

This thesis begins with the mathematical preliminaries, presented in Chapter 2, which introduces the notation and provides the definitions, lemmas, and theorems that are used throughout the thesis.

The rest of this thesis work is divided into two main parts. In the first part, constituted by Chapters 3−5, we focus on the mathematical models inducing multiconsensus and cluster synchronization, examining both the underlying theories and the associated control techniques.

The second part, covering Chapters 6−10, discusses the practical validation of mathematical models through experiments on robotic systems. Here, we begin with the validation of the multiconsensus, and then we extend the analysis to other models, such as the face-to-face interaction dynamics, the synchronization of moving chaotic agents, and the response to synchronization. The objective is to demonstrate how the developed theories can be implemented in real-world contexts, verifying their applicability and robustness.

Specifically, in Chapter 3, based on [101], we propose a communication protocol for multiconsensus that guides a system of agents with single-integrator dynamics, interacting through an undirected and connected graph, to reach a final state that reflects the symmetries of the interaction network. Similarly to [42], multiconsensus is induced by network symmetries. However, we use a different interaction protocol that ensures the convergence of the state variables of the multi-agent system to a vector parallel to the leading eigenvector of the graph adjacency matrix, allowing symmetric nodes to asymptotically converge to the same value. Nevertheless, this communication protocol does not allow reaching every possible multiconsensus solution, as its solution is constrained by the leading eigenvector.

In Chapter 4, derived from [102], we introduce a further communication protocol for multiconsensus which is applied to a second-order dynamics system where the agents interact through a directed and strongly connected graph. Specifically, this model drives the system to reach trajectories parallel to an arbitrary vector $\mathbf{v}$. To achieve this, we first assign weights to the interaction network such that the adjacency matrix has a target eigenvector centrality $\mathbf{v}$, enabling the formation of clusters of nodes sharing the same eigenvector centrality. Then, we adjust the gains to ensure the stability of the multiconsensus solution.

In Chapter 5 we move from multiconsensus to the broader concept of cluster synchronization. This chapter is built on the work presented in [103], where we leverage the concept of spectral blocks [7] to design controllers that shape the dynamics of clusters. Specifically, we introduce a control action able to induce the formation of such structures in networks where they do not naturally occur, thereby providing precise control over the synchronizability of individual clusters [103]. This approach also enables setting the sequence in which each cluster enters or exits the synchronization region as the coupling strength varies.

The second part begins with Chapter 6, where we describe the robotic platform used to perform our experimental investigation: the Elisa-3 and the e-puck2 robots.

In Chapter 7 we validate the mathematical model introduced in Chapter 3. Specifically, we apply the communication protocol to a team of Elisa-3 robots to address the rendezvous problem. Through the experimental results, we show that our model can be applied to a real-world system.

In Chapter 8, based on [104], we validate the mathematical model introduced in [95] which describes the social interaction occurring during human gatherings. Indeed, models devoted to the analysis of social systems can very often be validated against real data, but not compared with the results of controlled experiments. Therefore we propose our multi-robot system to facilitate experimental investigations in settings where the parameters can be tuned. In particular, the control action that we propose here is fully distributed as it relies solely on the local communication system onboard the robots.

In addition to local communication, Chapter 9 introduces another communication system based on virtual interaction to overcome the limitations due to this communication system. Specifically, we use both communication systems to validate with a team of robots the model discussed in [23] which studies the synchronization among units in motion, each provided by a chaotic oscillator and interacting through a time-varying graph as connections are established only when units are close. This validation is particularly significant because, although there are numerous experimental studies on synchronization—often involving mechanical systems, lasers, or electronic circuits [72, 75, 57, 61, 90, 27], most of these investigations deal with systems where the network topology remains fixed over time. This validation is also discussed in [105].

We conclude the second part with Chapter 10 which shows the experimental validation of the model describing the response to synchronization in fireflies [80]. Assessing the practical applicability of the model is crucial, as it demonstrates how units that typically cannot achieve synchronization on their own can synchronize through interaction with other units that, on the contrary, possess the capability to synchronize. Therefore, it can be applied to induce synchronization among units that normally cannot synchronize. The results shown in this chapter are derived from [106].

Chapter 2

Mathematical preliminaries

This chapter introduces the notation as well as some useful lemmas, definitions, and theorems that are used throughout this thesis. Additional theorems and definitions are provided in the chapters where they are relevant. We also introduce the concept of Master Stability Function (MSF) which is a useful tool to assess the stability of the synchronous solution in a multi-agent system.

2.1 Notation

Throughout this thesis, we adopt the following notation:

Vectors

Vectors are indicated in boldface. Commonly used symbols related to vectors include:

$\mathbf{0}_N$	N-dimensional column vector with entries that are equal to zero. When clear from the context, the subscript N is omitted
$\mathbf{1}_N$	N-dimensional column vector with entries that are equal to one
$\mathbf{v} > \mathbf{0}$	every element of the vector $\mathbf{v}$ is positive
$\mathbf{v} \geq \mathbf{0}$	every element of the vector $\mathbf{v}$ is non-negative
$\langle \mathbf{v}, \mathbf{w} \rangle$	scalar product between the vectors $\mathbf{v}$ and $\mathbf{w}$, i.e., $\langle \mathbf{v}, \mathbf{w} \rangle = \mathbf{v}^T \mathbf{w}$

Matrices

Matrices are indicated in roman text. Commonly used symbols related to matrices include:

I_N	$N \times N$ identity matrix
$0_{N,M}$	$N \times M$ matrix with each element equal to zero
$\rho(\mathrm{A})$	spectral radius of matrix A
$\mathrm{A} > 0$	every element of the matrix A is positive
$\mathrm{A} \geq 0$	every element of the matrix A is non-negative

Matrix operations

$\mathrm{A} \circ \mathrm{B}$ Hadamard product of the $N \times M$ matrices A and B, i.e., $(\mathrm{A} \circ \mathrm{B})_{ij} = a_{ij} b_{ij}$, where a_{ij} and b_{ij}, for $i = 1, \ldots, N$ and $j = 1, \ldots, M$, are the entries of the matrices A and B, respectively

$\mathrm{A} \otimes \mathrm{B}$ Kronecker product of an $N \times M$ matrix A and a $P \times Q$ matrix B, resulting in an $NP \times MQ$ block matrix with each block being $(\mathrm{A} \otimes \mathrm{B})_{ij} = a_{ij}\mathrm{B}$, namely:

$$\mathrm{A} \otimes \mathrm{B} = \begin{bmatrix} a_{11}\mathrm{B} & \ldots & a_{1M}\mathrm{B} \\ \vdots & \ddots & \vdots \\ a_{N1}\mathrm{B} & \ldots & a_{NM}\mathrm{B} \end{bmatrix}$$

$\mathrm{vec}(\mathrm{A})$ vectorization of the $N \times M$ matrix A obtained by stacking the columns of A into a single column as follows:

$$\mathrm{vec}(\mathrm{A}) = [a_{11}, \ldots, a_{N1}, a_{12}, \ldots, a_{N2}, \ldots, a_{1M}, \ldots, a_{NM}]^T$$

$\mathrm{diag}(\mathrm{A})$ function that returns an $N \times N$ matrix having in the diagonal the elements of the diagonal of A and zero elsewhere

2.2 Graphs

Here we introduce the concept of graph and some related definitions [44].

Definition 2.2.1 (Graph) *A graph is a mathematical structure described by the pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}(\mathcal{G}) = \{1, 2, \ldots, N\}$ is the set of the vertices/nodes, and $\mathcal{E}(\mathcal{G}) \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges, which are ordered pairs of vertices.*

To represent a graph $\mathcal{G}$, the adjacency matrix A can be used.

Definition 2.2.2 (Adjacency matrix) *It is an $N \times N$ matrix with elements a_{ij} such that $\mathrm{a}_{ij} > 0$ if there is a link between the vertices i and j, and $\mathrm{a}_{ij} = 0$ otherwise.*

Definition 2.2.3 (Laplacian matrix) *Given the adjacency matrix A, the Laplacian matrix L is a $N \times N$ zero-row sum matrix with entries $\mathrm{l}_{ij} = -a_{ij}$ if $i \neq j$ and $\mathrm{l}_{ii} = \sum_{i=1}^{N} a_{ij}$. This matrix is positive semidefinite.*

Definition 2.2.4 (Weighted graph) *A weighted graph is a graph where each edge is assigned a numerical value, known as a weight. If no weights are assigned to the edges, the graph is said unweighted.*

The adjacency matrix of an unweighted graph is binary: $\mathrm{a}_{ij} = 1$ if there is a link between the vertices i and j, and $\mathrm{a}_{ij} = 0$ otherwise.

Definition 2.2.5 (Undirected graph) *A graph is said to be undirected when for any $(i, j) \in \mathcal{E}$, then $(j, i) \in \mathcal{E}$. Otherwise, the graph is said directed (or digraph).*

The adjacency matrix of an undirected graph is symmetric: $a_{ij} = a_{ji}$ $\forall i, j = 1, \ldots, N$.

Definition 2.2.6 (Simple graph) *A simple graph is a graph that does not contain multiple edges between any pair of nodes and does not contain self-loops.*

Definition 2.2.7 (Neighborhood of a vertex) *The neighborhood of a vertex $i \in \mathcal{V}$, denoted as $\mathcal{N}_i$, is the set of all vertices that are adjacent to i. In other words, it is the set of vertices that are connected to i by an edge.*

Definition 2.2.8 (Path in a graph) *A path between two nodes, n_0 and n_k, is an alternating sequence of nodes and edges $n_0, e_1, n_1, \ldots, e_l, n_l$, that begins with n_0 and ends with n_l, such that $e_j = (n_{j-1}, n_j) \in \mathcal{E}$ for $j = 1, 2, \ldots, l$, and no node is visited more than once (i.e, $n_h \neq n_j$ for all $h = 0, 1, \ldots, l$ and $j = 0, 1, \ldots, l$, with $h \neq j$).*

Definition 2.2.9 (Connected graph) *An undirected graph is said to be connected if, for any pair of vertices, there is a path between them.*

If the graph is connected, then its adjacency matrix A is irreducible, that is, there is not a permutation matrix P such that PAP^{-1} is in block upper triangular form.

Definition 2.2.10 (Weakly connected digraph) *A digraph is weakly connected if there is a path between every pair of nodes, regardless of the direction of the edges.*

Definition 2.2.11 (Strongly connected digraph) *A digraph is strongly connected if, for every pair of vertices, there is a path between them.*

If the digraph is strongly connected, then its adjacency matrix A is irreducible.

Definition 2.2.12 (Graph partition [28]) *A partition π of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a map of the vertices that groups them into Q distinct cells, $C_1, C_2, \ldots, C_Q$, with $\bigcup_{k=1}^{Q} C_k = \mathcal{V}$ and $C_i \cap C_j = \emptyset$ for $i \neq j$.*

Definition 2.2.13 (Quotient graph) *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a partition π of the vertex set $\mathcal{V}$ into subsets $C_1, C_2, \ldots, C_Q$, the quotient graph G/π is a graph having the following properties:*

- *its vertices correspond to the subsets $C_1, C_2, \ldots, C_Q$*
- *an edge between two vertices C_i and C_j exists if, in the graph $\mathcal{G}$, there is at least one edge in $\mathcal{E}$ connecting a vertex in C_i to a vertex in C_j.*

2.3 The Perron-Frobenius theorem and other useful results

Some useful lemmas and definitions are here briefly recalled.

Theorem 2.3.1 (Perron-Frobenius [44]) *If* A *is an* $N \times N$*, non-negative, irreducible matrix, then one of its eigenvalues is positive and greater than or equal to all other eigenvalues, this eigenvalue is a simple root of the characteristic equation of* A*, and there is a positive eigenvector corresponding to it.*

Definition 2.3.2 (Metzler matrix [21]) *A matrix* $\mathrm{A} \in \mathbb{R}^{N \times N}$ *is Metzler if all its off-diagonal coefficients are non-negative.*

Definition 2.3.3 (M-matrix [8]) *A matrix* $\mathrm{A} \in \mathbb{R}^{N \times N}$ *is said to be an* M*-matrix, if it can be expressed as* $\mathrm{A} = s\mathrm{I}_N - \mathrm{B}$ *where* B *is a Metzler matrix and* s *is at least as large as the spectral radius of* B*.*

Lemma 2.3.4 ([21]) *A Metzler matrix* A *is Hurwitz stable if and only if the leading minors of* $-\mathrm{A}$ *are positive.*

Lemma 2.3.5 ([8]) *Suppose that* A *is a singular matrix that has a simple zero eigenvalue, and let* $\mathbf{v} \neq \mathbf{0}$ *and* $\mathbf{w} \neq \mathbf{0}$ *be the associated left and right eigenvector of* A*, respectively, i.e.,* $\mathbf{v}^T\mathrm{A} = \mathbf{0}$ *and* $\mathrm{A}\mathbf{w} = \mathbf{0}$*. Then,* $\mathrm{A} + \mathbf{x}\mathbf{y}^T$ *is non-singular if and only if the following inequality, called non-zero projection (NZP) condition, is satisfied:*

$$(\mathbf{v}^T\mathbf{x})(\mathbf{y}^T\mathbf{w}) \neq 0 \tag{2.1}$$

Lemma 2.3.6 ([8]) *Suppose that* $\mathrm{A} \in \mathbb{R}^{N \times N}$ *is a singular, irreducible* M*-matrix that has a simple zero eigenvalue with associated left and right eigenvectors* $\mathbf{v} > \mathbf{0}$ *and* $\mathbf{w} > \mathbf{0}$*, and let* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^N$ *such that* $\mathbf{x} \geq \mathbf{0}$*,* $\mathbf{y} \geq \mathbf{0}$ *and* $(\mathbf{v}^T\mathbf{x})(\mathbf{y}^T\mathbf{w}) \neq 0$*. Then, the matrix* $\mathrm{A} + \mathbf{x}\mathbf{y}^T$ *has all positive leading minors.*

Lemma 2.3.7 (Farka's lemma I [26]) *Let* $\mathrm{A} \in \mathbb{R}^{N \times M}$ *and* $\mathbf{b} \in \mathbb{R}^N$*, then only one of the following two assertions is true:*

- *there exists* $\mathbf{x} \in \mathbb{R}^M$ *such that* $\mathrm{A}\mathbf{x} = \mathbf{b}$ *and* $\mathbf{x} \geq \mathbf{0}$*;*

- *there exists* $\mathbf{y} \in \mathbb{R}^N$ *such that* $\mathrm{A}^T\mathbf{y} \geq \mathbf{0}$ *and* $\mathbf{y}^T\mathbf{b} < 0$.

Lemma 2.3.8 (Farka's lemma II [58]) *Let* $\mathrm{A} \in \mathbb{R}^{m\times n}$ *and* $\mathbf{b} \in \mathbb{R}^m$, *then only one of the following statements is true:*

- *There exists* $\mathbf{x} \geq 0$ *such that* $\mathrm{A}\mathbf{x} \leq \mathbf{b}$.

- *There exists* $\mathbf{y} \geq 0$ *such that* $\mathrm{A}^T\mathbf{y} \geq 0$ *and* $\mathbf{b}^T\mathbf{y} > 0$.

2.4 The Master Stability Function

Let us consider a network of N coupled oscillators described by the following dynamics:

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + \sigma \sum_{j=1}^{N} a_{ij} \left(\mathbf{h}(\mathbf{x}_j) - \mathbf{h}(\mathbf{x}_i)\right) \tag{2.2}$$

where $\mathbf{x}_i \in \mathbb{R}^n$, $i = 1, \ldots, N$, $\mathbf{f}$ is the uncoupled dynamics, σ is the coupling strength, a_{ij} are the entries of the adjacency matrix describing the interaction graph $\mathcal{G}$, and $\mathbf{h}$ is the inner coupling function.

Eq. (2.2) is such that the synchronization manifold, that is defined by $\mathbf{x}_1 = \mathbf{x}_2 = \ldots = \mathbf{x}_N = \mathbf{x}_s$, always exists and has dynamics described by the following equation:

$$\dot{\mathbf{x}}_s = \mathbf{f}(\mathbf{x}_s) \tag{2.3}$$

The stability analysis of this state has been subject of extensive study in the literature. A pioneer work in this context is [25], which introduces key concepts that form the foundation for analyzing the stability of synchronization in coupled-oscillator systems. Here, we follow the approach introduced in [74], which starts by considering a small perturbation $\delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_s$ around the synchronization manifold, and by linearizing the system dynamics around the synchronization manifold:

$$\dot{\delta\mathbf{x}} = \left[\mathrm{I} \otimes D\mathbf{f}|_{\mathbf{x}_s} - \sigma \mathrm{L} \otimes D\mathbf{h}|_{\mathbf{x}_s}\right] \delta\mathbf{x} \tag{2.4}$$

where $\delta\mathbf{x} = [\delta\mathbf{x}_1^T, \delta\mathbf{x}_2,^T \ldots, \delta\mathbf{x}_N^T]^T$. Here, $\mathrm{D}\mathbf{f}|_{\mathbf{x}_s}$ and $\mathrm{D}\mathbf{h}|_{\mathbf{x}_s}$ are the Jacobian matrix of $\mathbf{f}$ and $\mathbf{h}$ computed around the synchronous manifold $\mathbf{x}_s$, respectively. Let us suppose that L is diagonalizable, then Eq. (2.4) is block-diagonalized, resulting in a new set of equations where each block has the form $\dot{\xi}_i = [\mathrm{D}\mathbf{f}|_{\mathbf{x}_s} - \sigma\lambda_i(\mathrm{L})\mathrm{D}\mathbf{h}|_{\mathbf{x}_s}]\,\xi_i$. Since the blocks only differ for the eigenvalue appearing in it, by introducing the parameter $\nu = \sigma\lambda_i(\mathrm{L})$, a single Master Stability Equation (MSE), namely $\dot{\zeta} = [\mathrm{D}\mathbf{f}|_{\mathbf{x}_s} - \nu\mathrm{D}\mathbf{h}|_{\mathbf{x}_s}]\,\zeta$, can be considered. This is an important step, as it allows to separate the role of the unit dynamics (namely $\mathbf{f}$ and $\mathbf{h}$) from that of the structure of interactions (the eigenvalues of the Laplacian matrix) in the variational equation. From the MSE, the maximum Lyapunov exponent $\lambda_{\max}$ is calculated as a function of ν, thus obtaining the Master Stability Function (MSF), i.e., $\lambda_{\max} = \lambda_{\max}(\nu)$. The condition on stability of synchronization is then expressed as $\lambda_{\max}(\nu) = \lambda_{\max}(\sigma\lambda_i(\mathrm{L})) < 0\ \forall i = 2, \ldots, N$. Note that, although we assumed L to be diagonalizable, this is not a necessary condition. Indeed, the Jordan decomposition can be applied to block-diagonalize Eq. (2.4), leading to a similar stability analysis [67, 20]. Thus, the MSF provides a necessary condition for the stability of the synchronization manifold that is effective and easy to check, unveiling how network topology affects the property of synchronization stability.

In [9], three classes of MFSs have been identified for chaotic systems: type I MSF, type II MSF, and type III MSF that we now illustrate with reference to a network having a Laplacian with real eigenvalues (all undirected networks satisfy this property). In systems with type I MFS, $\lambda_{\max}(\nu) > 0$ for any value of ν, making the synchronization manifold unstable $\forall\sigma$. In systems with type II MSF, $\lambda_{\max}(\nu)$ turns from positive to negative values at the critical value ν^*, yielding a scenario where a transition from instability to stability of the synchronization manifold can be observed when the coupling strength is increased from zero. In this case, synchronization stability can be achieved if $\sigma\lambda_2(\mathrm{L}) > \nu^*$. For systems with type III MSF, $\lambda_{\max}(\nu) < 0$ for $\nu \in [\nu_1^*, \nu_2^*]$, where ν_1^* and ν_2^* are two threshold values. This yields the possibility of observing two transitions when the coupling strength is varied from zero: from instability to stability and from stability to instability. Also for systems with type III MSF, synchronization stability depends on the spectrum of L; more specifically, it requires that $d\lambda_i \in [\nu_1^*, \nu_2^*]\ \forall i = 2, \ldots, N$. This condition can be achieved only if $\frac{\lambda_N(\mathrm{L})}{\lambda_2(\mathrm{L})} < \frac{\nu_2^*}{\nu_1^*}$. Fig. 2.1 summarizes the three classes of MSF that can be observed.

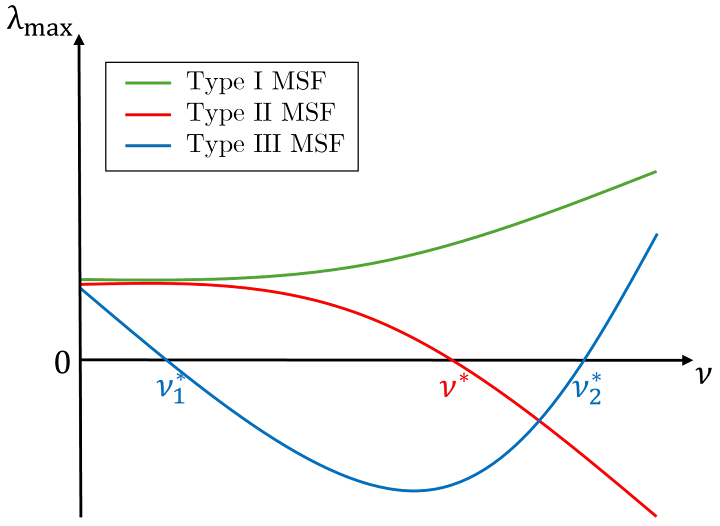


Figure 2.1: Possible classes of MSF.

Part I

Theoretical results

Chapter 3

Multiconsensus induced by network symmetries

In this chapter, we introduce a communication protocol for multiconsensus that induces a system of units with single integrator dynamics to achieve a state reflecting the interaction network's symmetries. In particular, this communication protocol leads the system to reach a state that is parallel to the leading eigenvector of the adjacency matrix representing the interaction network, such that symmetric nodes reach the same state. We also show that, by pinning just a single arbitrary node, it is possible to control the system such that it reaches a specific vector of the subspace spanned by the leading eigenvector, independently from the agents' initial conditions. Furthermore, we analyze the protocol in the presence of perturbations of the adjacency matrix describing the interaction network and provide an estimate for the difference between the final state reached by the nominal and perturbed systems. With this result, we revisit the notion of quasi-symmetries and propose a quantitative criterion for their definition. As an application, we use our protocol to address the rendezvous problem in a bi-dimensional space.

3.1 Symmetries in a graph

Before introducing our communication protocol, we discuss the concept of symmetries in a graph, including a lemma that is crucial for our technique.

Graphs are said to have a symmetry if there exists a permutation of the nodes that preserves the connectivity pattern. The symmetries of a graph form a mathematical group, denoted by $\mathbb{G}$, where each element is represented as a square permutation matrix R^g with $\mathrm{R}^g_{ij} = 1$ if node j is mapped to node i under the permutation, and $\mathrm{R}^g_{ij} = 0$ otherwise. Any permutation matrix of the group commutes with the adjacency matrix A of the graph, i.e., $\mathrm{R}^g\mathrm{A} = \mathrm{A}\mathrm{R}^g$.

The symmetry group induces a partition of the nodes into disjoint sets called orbits, which include all nodes that are mapped into each other after the application of all symmetries of the group.

The following Lemma expresses an important property related to symmetries (see for instance [87]).

Lemma 3.1.1 *Let* A *be the adjacency matrix of a connected graph and let* $\mathbb{G}$ *be the symmetry group of the graph. Assume thar* A *is irreducible. Then, for each permutation matrix* $\mathrm{R}^g \in \mathbb{G}$*, the eigenvector* $\mathbf{v}_1$ *associated to the maximum eigenvalue of* A *satisfies:*

$$\mathrm{R}^g\mathbf{v}_1 = \mathbf{v}_1 \tag{3.1}$$

Proof. By the Perron-Frobenius theorem, the maximum eigenvalue of A, namely $\rho(\mathrm{A})$, is simple. The associated eigenvector $\mathbf{v}_1$ is such that $\mathrm{A}\mathbf{v}_1 = \rho(\mathrm{A})\mathbf{v}_1$. Left-multiplying this relationship by R^g, one has that $\mathrm{R}^g\mathrm{A}\mathbf{v}_1 = \rho(\mathrm{A})\mathrm{R}^g\mathbf{v}_1$. Since $\mathrm{R}^g\mathrm{A} = \mathrm{A}\mathrm{R}^g$, it follows that $\mathrm{A}\mathrm{R}^g\mathbf{v}_1 = \rho(\mathrm{A})\mathrm{R}^g\mathbf{v}_1$, hence $\mathrm{R}^g\mathbf{v}_1$ belongs to the subspace generated by $\mathbf{v}_1$ and so can be expressed in a basis of $\mathbf{v}_1$: $\mathrm{R}^g\mathbf{v}_1 = a\mathbf{v}_1$. Now, since R^g is a permutation matrix, the transformation $\mathrm{R}\mathbf{v}_1$ only permutes the elements of $\mathbf{v}_1$, and thus $a = 1$. Consequently, $\mathrm{R}^g\mathbf{v}_1 = \mathbf{v}_1$. $\square$

Since the property of Lemma 3.1.1 holds for any permutation matrix of the symmetry group $\mathbb{G}$ of A, we can associate to $\mathbb{G}$ a partition $\pi_{\mathbb{G}}$ that is formed by the cells that group together the nodes of the same orbit, such that, for nodes of the same cell, the component in $\mathbf{v}_1$ is the same. Since the leading eigenvector $\mathbf{v}_1$ represents the eigenvector centrality of the graph [44], nodes of the same cell have the same eigenvector centrality value.

3.2 Multiconsensus protocol

Let us consider a multi-agent system formed by N agents with first-order integrator dynamics

$$\dot{x}_i = v_i \tag{3.2}$$

where $i = 1, \ldots, N$ and $x_i \in \mathbb{R}$. Here, we study the control protocol $v_i = \frac{1}{\rho(\mathrm{A})} \sum_{j=1}^{N} a_{ij} x_j - x_i + u_i$, where a_{ij} are the coefficients of the adjacency matrix $\mathrm{A} = \{a_{ij}\}$ of the interaction graph modeling how agents interact with each other, and u_i a further (additive) control signal. With this control protocol, the dynamics of the agents is given by:

$$\dot{x}_i = \frac{1}{\rho(\mathrm{A})} \sum_{j=1}^{N} a_{ij} x_j - x_i + u_i \tag{3.3}$$

with $i = 1, \ldots, N$.

In the rest of the chapter, we always assume that the graph is simple, undirected, and connected so that the adjacency matrix is symmetric, non-negative, with zero diagonal entries, and irreducible.

Eqs. (3.3) are rewritten in compact notation as:

$$\dot{\mathbf{x}} = \left(\frac{1}{\rho(\mathrm{A})} \mathrm{A} - \mathrm{I}_N \right) \mathbf{x} + \mathbf{u} \tag{3.4}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$ and $\mathbf{u} = [u_1, u_2, \ldots, u_N]^T$.

Definition 3.2.1 *Given a multi-agent system (3.3) and a partition* $\pi = \{C_1, C_2, \ldots, C_Q\}$ *of the agents, the multiconsensus manifold is defined as* $\mathbb{MC}(\pi) = \{\mathbf{x} : x_i = x_j, \forall i, j \in C_h, h = 1, \ldots, Q\}$.

Definition 3.2.2 *A multi-agent system (3.3) is said to reach the multiconsensus associated with the cell partition* π *if*

$$\lim_{t \to +\infty} \mathbf{x}(t) = \bar{\mathbf{x}} \tag{3.5}$$

with $\bar{\mathbf{x}} \in \mathbb{MC}(\pi)$ *or, equivalently, if*

$$\lim_{t \to +\infty} (x_i(t) - x_j(t)) = 0, \forall i, j \in C_h, h = 1, \ldots, Q \tag{3.6}$$

We begin with the analysis of the behavior for $\mathbf{u} = 0$. The next lemma illustrates that, under this condition, the multi-agent system converges to a solution that belongs to the multiconsensus manifold.

Lemma 3.2.3 *Consider the multi-agent system*

$$\dot{\mathbf{x}} = \left(\frac{1}{\rho(\mathrm{A})} \mathrm{A} - \mathrm{I}_N \right) \mathbf{x} \tag{3.7}$$

with A *the adjacency matrix of the interaction graph, assumed to be simple, undirected and connected, and let* $\mathbb{G}$ *be the group of symmetries of* A*.*

Then, given an arbitrary initial condition $\mathbf{x}(0)$*, the trajectory of the multi-agent system asymptotically converges to the multiconsensus manifold* $\mathbb{MC}(\pi_{\mathbb{G}})$*, where* $\pi_{\mathbb{G}}$ *is the partition induced by* $\mathbb{G}$*, that is:*

$$\lim_{t \to +\infty} \mathbf{x}(t) = \bar{\mathbf{x}}, \tag{3.8}$$

where $\bar{\mathbf{x}}$ *is a vector of the multiconsensus manifold, namely* $\bar{\mathbf{x}} \in \mathbb{MC}(\pi_{\mathbb{G}})$*, that depends on the initial condition* $\mathbf{x}(0)$*.*

Proof. Let $\mathrm{A}_\rho = \frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N$. Since the graph is connected and undirected, then the matrix $\frac{1}{\rho(\mathrm{A})}\mathrm{A}$ is symmetric, diagonalizable and has all real eigenvalues with their maximum being equal to one. It follows that $\mathrm{A}_\rho = \frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N$ is negative semidefinite and always has one zero eigenvalue. Let $\mathbf{v}_0$ be the eigenvector associated with this eigenvalue. Since $\left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{v}_0 = 0$, and so $\mathrm{A}\mathbf{v}_0 = \rho(\mathrm{A})\mathbf{v}_0$, we get that $\mathbf{v}_0 = \mathbf{v}_1$, where $\mathbf{v}_1$ is the eigenvector associated to the largest eigenvalue of A, namely λ_1. Since by Lemma 3.1.1, for any permutation matrix $\mathrm{R} \in \mathbb{G}$, we have that $\mathrm{R}\mathbf{v}_1 = \mathbf{v}_1$, then $\mathbf{v}_1 \in \mathbb{MC}(\pi_{\mathbb{G}})$ and $v_{1,i} = v_{1,j}$ for all nodes i and j that are symmetric.

The solution of Eqs. (3.7) is given by:

$$\mathbf{x}(t) = e^{\mathrm{A}_\rho t}\mathbf{x}(0) \tag{3.9}$$

Taking into account that A_ρ is diagonalizable, $\mathbf{x}(t)$ can be written as:

$$\mathbf{x}(t) = \left(e^{\lambda_1 t}\mathbf{v}_1\mathbf{v}_1^T + e^{\lambda_2 t}\mathbf{v}_2\mathbf{v}_2^T + \ldots + e^{\lambda_N t}\mathbf{v}_N\mathbf{v}_N^T \right)\mathbf{x}(0) \tag{3.10}$$

where λ_i, $i = 1, \ldots, N$ are the eigenvalues of A_ρ, with $\lambda_1 = 0$.

Since A_ρ is negative semidefinite and, by the Perron-Frobenius theorem, λ_1 is simple, then

$$\lim_{t \to +\infty} \mathbf{x}(t) = \mathbf{v}_1\mathbf{v}_1^T\mathbf{x}(0) \tag{3.11}$$

It follows that $\bar{\mathbf{x}} = c\mathbf{v}_1$ where $c = \mathbf{v}_1^T\mathbf{x}(0)$. Since $\mathbf{v}_1 \in \mathbb{MC}(\pi_{\mathbb{G}})$, then also $\bar{\mathbf{x}} \in \mathbb{MC}(\pi_{\mathbb{G}})$. □

Lemma 3.2.3 shows that a group of agents interacting through the protocol (3.7) asymptotically converges towards a multiconsensus solution that is parallel to the eigenvector $\mathbf{v}_1$, so that symmetric agents approach the same steady-state value. The constant of proportionality between the multiconsensus solution $\bar{\mathbf{x}}$ and the eigenvector $\mathbf{v}_1$, namely the parameter c in the proof of the theorem, depends on the initial conditions of the agents. The next example illustrates the application of Lemma 3.2.3 to a network of $N = 16$ agents.

Example 3.2.4 *Let us consider the multi-agent system (3.7) with $N = 16$ agents interacting according to the network shown in Fig. 3.1(a). The symmetries of the adjacency matrix* A *induce the following partition of the network nodes:*

$$\begin{aligned}\pi_{\mathbb{G}} &= \{C_1, C_2, C_3, C_4, C_5, C_6, C_7\} \\ &= \{\{1,2,5,6\},\{3\},\{4\},\{7,8\},\{9,10\},\{11,12\},\{13,14,15,16\}\}\end{aligned}$$

The leading eigenvector $\mathbf{v}_1$ *of* A *is:*

$$\begin{aligned}\mathbf{v}_1 = [&0.18, 0.18, 0.34, 0.40, 0.18, 0.18, 0.45, 0.45,\\ &0.14, 0.14, 0.24, 0.24, 0.10, 0.10, 0.10, 0.10]^T\end{aligned}$$

As expected from Lemma 3.1.1, we find that $v_{1,i} = v_{1,j}$ if nodes i and j belong to the same cell of the partition π.

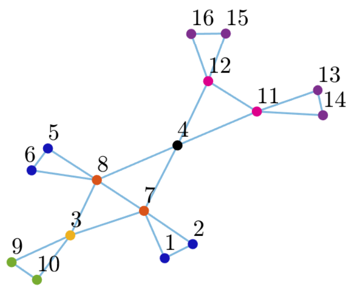
An example of the time evolution of the state variables of the multi-agent system is shown in Fig. 3.1(b). The variables asymptotically approach the equilibrium point:

$$\begin{aligned}\bar{\mathbf{x}} = [&0.40, 0.40, 0.75, 0.88, 0.40, 0.40, 0.99, 0.99,\\ &0.30, 0.30, 0.53, 0.53, 0.22, 0.22, 0.22, 0.22]^T\end{aligned}$$

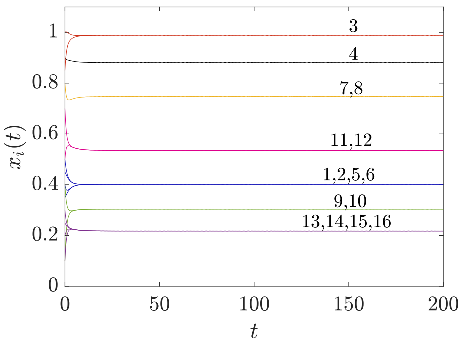
which is indeed parallel to $\mathbf{v}_1$ *as* $\frac{\bar{\mathbf{x}}^T\mathbf{v}_1}{\|\bar{\mathbf{x}}\|\|\mathbf{v}_1\|} = 1$.

3.3 Control of the multiconsensus state

As shown in Sec. 3.2, for $\mathbf{u} = \mathbf{0}$ the multi-agent system (3.4) converges to a multiconsensus solution $\bar{\mathbf{x}}$ that is parallel to the leading eigenvector $\mathbf{v}_1$



(a)



(b)

Figure 3.1: Multiconsensus in the multi-agent system (3.7). (a) Network of interactions among agents with symmetrical nodes indicated with the same color. (b) Time evolution of the state variables of the multi-agent system (3.7), showing that symmetric nodes converge to the same value.

of A, with the constant of proportionality c being a function of the initial conditions of the system. Here, we address the problem of controlling, via a non-zero input $\mathbf{u}$, the multi-agent system to converge to a predefined vector $\bar{\mathbf{v}} \in \text{span}(\mathbf{v}_1)$, independently from the initial condition of the agents. We show that this control problem can be solved by acting on a single (arbitrary) node of any cell of the partition of A.

Theorem 3.3.1 *Consider the multi-agent system described by the following equations*

$$\dot{x}_i = \frac{1}{\rho(\mathrm{A})} \sum_{j=1}^{N} a_{ij} x_j - x_i + u_i \tag{3.12}$$

with $i = 1, \ldots, N$ and A being the adjacency matrix of the interaction graph, assumed to be simple, undirected and connected. Let $C_{\bar{h}}$ be an arbitrary cell of the partition associated to the symmetry group of A, $\bar{i}$ an arbitrary node of this cell $C_{\bar{h}}$, and $\bar{\mathbf{v}} \in \text{span}(\mathbf{v}_1)$ a target multiconsensus solution. In addition, let $\bar{c} = \langle \bar{\mathbf{v}}, \mathbf{v}_1 \rangle$, that is, $\bar{\mathbf{v}} = \bar{c}\mathbf{v}_1$. Then, it is possible to control system (3.12) so that $\mathbf{x}(t)$ converges to $\bar{\mathbf{v}}$ by selecting as control input:

$$u_i = k\xi_i(x_{ref} - x_i) \tag{3.13}$$

where k is an arbitrary positive constant, $\xi_i = 1$ if $i = \bar{i}$, and $\xi_i = 0$, otherwise, and $x_{ref} = \bar{c}v_{1,\bar{i}}$.

Proof. We start from the compact form of system (3.12)

$$\dot{\mathbf{x}} = \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{x} + \mathbf{u} \tag{3.14}$$

and note that $[\xi_1, \ldots, \xi_N]^T = \mathbf{e}_{\bar{\mathbf{i}}}$, namely it is the $\bar{i}$-th canonical vector, so that we can write

$$\dot{\mathbf{x}} = \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{x} + k(x_{ref}\mathbf{e}_{\bar{\mathbf{i}}} - \mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\mathbf{x}) \tag{3.15}$$

As the target of the control is to reach the steady-state $\bar{\mathbf{v}}$, i.e., to obtain that $\lim\limits_{t\to+\infty}\mathbf{x}(t) = \bar{\mathbf{v}}$, we define the control error as $\mathbf{z} = \mathbf{x} - \bar{\mathbf{v}}$ and study its dynamics:

$$\dot{\mathbf{z}} = \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\right)(\mathbf{z} + \bar{\mathbf{v}}) + kx_{ref}\mathbf{e}_{\bar{\mathbf{i}}} \tag{3.16}$$

Taking into account that $\left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\bar{\mathbf{v}} = 0$ and that $\bar{\mathbf{v}} = \bar{c}\mathbf{v}_1$, Eq. (3.16) becomes

$$\dot{\mathbf{z}} = \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\right)\mathbf{z} - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\bar{c}\mathbf{v}_1 + kx_{ref}\mathbf{e}_{\bar{\mathbf{i}}} \tag{3.17}$$

Since $\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\bar{c}\mathbf{v}_1 - x_{ref}\mathbf{e}_{\bar{\mathbf{i}}} = 0$, the error dynamics is:

$$\dot{\mathbf{z}} = \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\right)\mathbf{z} \tag{3.18}$$

Now, we demonstrate that $\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T$ is negative definite. To show this, we notice that, since A is an irreducible non-negative matrix, then $\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N$ is an irreducible Metzler matrix with 0 as dominant eigenvalue. Consequently, $\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T$ is also an irreducible Metzler matrix. Since $\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N \geq \frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T$, then $\lambda_{max}\left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right) > \lambda_{max}\left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\right)$. This yields that all eigenvalues of $\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T$ are strictly negative.

Taking into account that $\left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N - k\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T\right)$ is negative definite, then $\lim\limits_{t\to+\infty}\mathbf{z}(t) = 0$, and hence $\lim\limits_{t\to+\infty}\mathbf{x}(t) = \bar{c}\mathbf{v}_1$. $\square$

Notice that the multi-agent system (3.12) with the control law (3.13) can also be viewed as a leader-follower scheme where the agents $i = 1, \ldots, N$ are the followers and there is a single leader l that is connected to unit $\bar{i}$ of the cell $C_{\bar{h}}$ and has dynamics $\dot{x}_l = 0$ with initial condition $x_l(0) = x_{ref}$.

Following [31], it becomes possible to change the set of symmetries of a network by perturbing its structure, e.g., adding and/or removing a minimum number of edges, then the characteristics of the multiconsensus solution can be fully controlled, first applying the techniques of Ref. [31] to control the network symmetries and, then, the results of Theorem 3.3.1 to steer the network towards a specific vector of the multiconsensus manifold.

Next, we show a numerical example of application of the control law of Theorem 3.3.1.

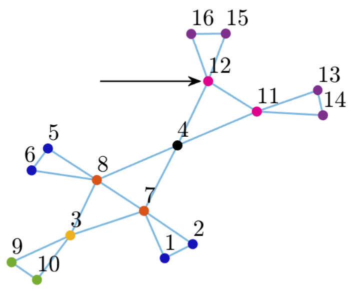
Example 3.3.2 *Let us consider again the multi-agent system analyzed in Example 3.2.4 and a target multiconsensus solution* $\bar{\mathbf{v}} = \bar{c}\mathbf{v}_1$ *with* $\bar{c} = 2$*. Control is applied as in Eq. (3.15), selecting without any lack of generality* $k = 2$ *and* $\bar{i} = 11$*. Hence,* $\xi_{11} = 1$ *and* $\xi_i = 0$ *for* $i = \{1, \dots, 10, 12, \dots, 16\}$*. This yields a non-zero control input effectively applied only to node 11 (Fig. 3.2(a)), whereas for all the other nodes the control input is zero. A typical time evolution of the system is reported in Fig. 3.2(b) which shows convergence of the variables to*

$$\begin{aligned}\bar{\mathbf{v}} = [&0.36, 0.36, 0.68, 0.80, 0.36, 0.36, 0.89, 0.89,\\ &0.27, 0.27, 0.48, 0.48, 0.20, 0.20, 0.20, 0.20]^T\end{aligned}$$

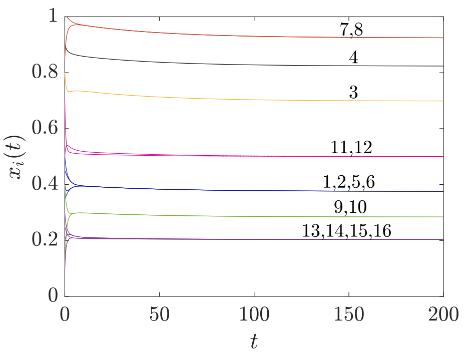
The controlled system reaches the control goal as $\langle \bar{\mathbf{v}}, \mathbf{v}_1 \rangle = 2$*.*

3.4 Multiconsensus in perturbed multi-agent systems

In this section, we study the behavior of the multi-agent system in the presence of a perturbation $\delta \mathrm{A} = \{\delta a_{ij}\}$ of the adjacency matrix A. We start with the case when $\mathbf{u} = 0$, i.e., Eqs. (3.7), to move then to the case when $\mathbf{u} \neq 0$, as in Eqs. (3.4). We refer to the multi-agent system with connectivity A as the *unperturbed* or *nominal* multi-agent system (Eq. (3.7)), and to the system with connectivity $\mathrm{A} + \delta\mathrm{A}$ as the *perturbed* multi-agent system. In the following we always make two assumptions: i) the perturbations are small enough to preserve the property that the graph is connected; ii) $\delta a_{ij} = \delta a_{ji}$, which represents symmetric perturbations, as for instance it occurs for reciprocal interactions among agents. Under these conditions,



(a)



(b)

Figure 3.2: Multiconsensus in the controlled multi-agent system (3.15). (a) Network of interactions among agents with symmetric nodes indicated with the same color and the controlled node indicated with an arrow ($\bar{i} = 11$). (b) Time evolution of the state variables of the multi-agent system (3.15), showing convergence to the target multiconsensus solution $\bar{\mathbf{v}} = \bar{c}\mathbf{v}_1$.

we can assume that the matrix $\mathrm{A} + \delta\mathrm{A}$ is symmetric with all nonnegative coefficients. Notice, however, that in general even the introduction of small perturbations changes the group of symmetries of the interaction network, with one or more of the existing symmetries that can eventually be removed from the group. With these positions, the perturbed multi-agent system is described by:

$$\dot{\mathbf{x}}_p = \left(\frac{1}{\rho(\mathrm{A} + \delta\mathrm{A})}(\mathrm{A} + \delta\mathrm{A}) - \mathrm{I}_N \right) \mathbf{x}_p \tag{3.19}$$

In addition, let $\bar{\mathbf{x}}$ be the multiconsensus solution reached by the nominal system ($\delta\mathrm{A} = 0$) and $\bar{\mathbf{x}}_p$ the steady state solution of the perturbed multi-agent system. Then, assuming that the two systems start from the same initial condition $\mathbf{x} \neq 0$, these two states will differ by a quantity $\delta\mathbf{x} = \bar{\mathbf{x}} - \bar{\mathbf{x}_p}$ with norm:

$$\|\delta\mathbf{x}\| = \|\mathbf{v}_1\mathbf{v}_1^T\mathbf{x}(0) - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\mathbf{x}(0)\| \tag{3.20}$$

where $\tilde{\mathbf{v}}_1$ is the leading eigenvector of $\mathrm{A} + \delta\mathrm{A}$. From Eq. (3.20), using consistent norms, one gets that:

$$\|\delta\mathbf{x}\| \leq \|\mathbf{v}_1\mathbf{v}_1^T - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\|\|\mathbf{x}(0)\| \tag{3.21}$$

Using matrix perturbation theory [97], we can write a first-order approximation of $\tilde{\mathbf{v}}_1$:

$$\tilde{\mathbf{v}}_1 \approx \mathbf{v}_1 + \sum_{j=2}^{N} \frac{\mathbf{v}_j^T \delta \mathrm{A} \mathbf{v}_1}{\lambda_1 - \lambda_j} \mathbf{v}_j, \tag{3.22}$$

which can be plugged into (3.21) to obtain an approximate expression for $\|\delta \mathbf{x}\|$:

$$\begin{aligned} \|\delta \mathbf{x}\| \quad \approx \quad & \left\| \sum_{j=2}^{N} \sum_{k=2}^{N} \frac{\mathbf{v}_j^T \delta \mathrm{A} \mathbf{v}_1 \mathbf{v}_k^T \delta \mathrm{A} \mathbf{v}_1}{(\lambda_1 - \lambda_j)(\lambda_1 - \lambda_k)} \mathbf{v}_j \mathbf{v}_k^T \mathbf{x}(0) \right. \\ & \left. + \sum_{j=2}^{N} \frac{\mathbf{v}_j^T \delta \mathrm{A} \mathbf{v}_1}{\lambda_1 - \lambda_j} \mathbf{v}_1 \mathbf{v}_j^T \mathbf{x}(0) + \sum_{j=2}^{N} \frac{\mathbf{v}_j^T \delta \mathrm{A} \mathbf{v}_1}{\lambda_1 - \lambda_j} \mathbf{v}_j \mathbf{v}_1^T \mathbf{x}(0) \right\| \triangleq \|\delta \mathbf{x}\|_{\text{approx}} \end{aligned} \tag{3.23}$$

Despite our approach being general, for simplicity here we only focus on numerical examples where the perturbations that are applied keep the interaction networks undirected and unweighted, considering the removal/addition of an increasing number of links. As a first example, we study multiconsensus in the presence of perturbations of the connectivity for systems with a number of agents N ranging from 50 to 200. For each N we consider 150 graphs. These networks have been generated with the following algorithm. We start from a random Erdös-Renyi model with $N = 200$ nodes, average degree equal to 11, and a symmetry group inducing a partition of the nodes into $Q = 50$ cells. We, then, progressively decrease the number of nodes by eliminating randomly selected vertices. The nodes are also removed from the original partition, thus obtaining a new partition which can eventually have a smaller number of cells. The final step consists of adding/removing links to the structure to obtain a graph with the symmetry group associated to the novel partition. This step is accomplished by readapting the connectivity preserving method of Ref. [31] where a minimum number of links is changed. For each of the graphs analysed, we have then considered a number $M = 200$ of perturbed networks obtained from the original graph by either removing a randomly selected link or adding a link between randomly chosen nodes. In the first case (removal of a link), only links that can be removed without disconnecting the graph are considered. In both cases, since the network is undirected, $\|\delta \mathrm{A}\| = \sqrt{2}$.

For each perturbed multi-agent system, we have calculated the error between the multiconsensus solution reached by the nominal multi-agent system and that reached by the perturbed multi-agent system either using the exact expression (3.20), i.e., $\|\delta\mathbf{x}\|$, or the approximation provided by Eq. (3.23), i.e., $\|\delta\mathbf{x}\|_{\text{approx}}$. The results are shown in Fig. 3.3(a) which prompts some interesting considerations. First, we find that the state reached by the perturbed multi-agent system differs by a small quantity $\|\delta\mathbf{x}\|$ from that of the nominal multi-agent system. In these conditions, therefore, one can conclude that the perturbed multi-agent system reaches a quasi-multiconsensus state, where the nodes of the same cell do not display exactly the same steady-state value but converge to very close values. Second, the approximation of Eq. (3.23) provides a good estimation of the exact error. Additionally, we analyzed the relationship between the exact and approximated values of $\delta\mathbf{x}$ and found a linear trend with a slope of 0.8999 and an intercept of 0.0027, as illustrated in Fig. 3.3(b).

We now study the multiconsensus state obtained for increasing values of the perturbation $\|\delta\mathrm{A}\|$, considering a multi-agent system described by Eq. (3.7) with $N = 200$ nodes and average degree equal to 11, and comparing the multiconsensus solution reached by the nominal system with that obtained by the perturbed system. The results are presented in Fig. 3.4 which shows how the multiconsensus error $\|\delta\mathbf{x}\|$ varies with $\|\delta\mathrm{A}\|$. As the number of links added to or removed from the original network is increased, and so is $\|\delta\mathrm{A}\|$, the difference between the nominal and perturbed multiconsensus solution increases as well. At the same time, the accuracy of the error estimate, via Eq. (3.23), decreases.

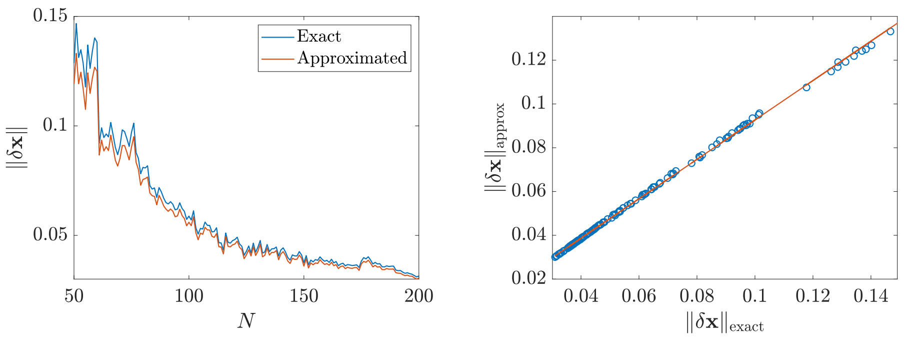


Figure 3.3: (a) Exact and approximated multiconsensus error $\|\delta\mathbf{x}\|$, calculated using Eq. (3.20) and Eq. (3.23), respectively, for a set of graphs with increasing size N. For each graph, $M = 200$ different perturbed networks with $\|\delta \mathrm{A}\| = \sqrt{2}$ have been considered and the results have been averaged over these perturbation instances. (b) Relationship between $\|\delta\mathbf{x}\|_{\text{exact}}$ and $\|\delta\mathbf{x}\|_{\text{approx}}$, revealing a linear trend with a slope of 0.8999 and an intercept of 0.0027.

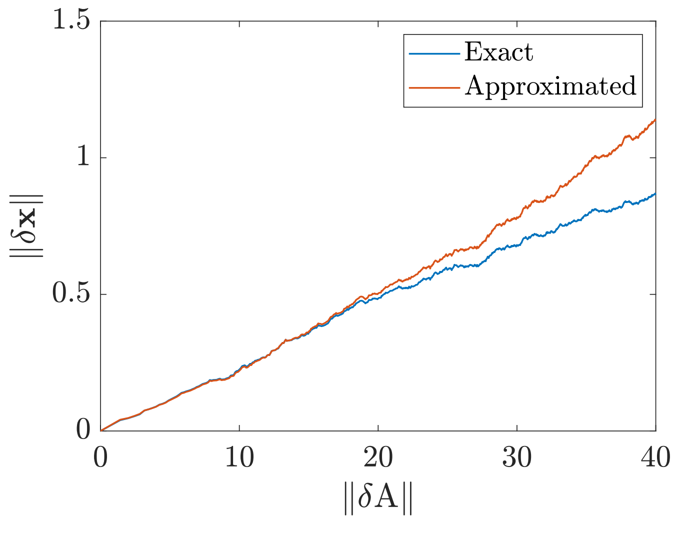


Figure 3.4: (a) Exact and approximated multiconsensus error $\|\delta\mathbf{x}\|$, calculated using Eq. (3.20) and Eq. (3.23), respectively, for a perturbed multi-agent systems with $N = 200$. The system is perturbed through a perturbation matrix having an increasing norm.

The considerations discussed above also apply to the multi-agent system

(3.4). In this case, the perturbed multi-agent system is described by this equation:

$$\dot{\mathbf{x}}_p = \left(\frac{1}{\rho(\mathrm{A} + \delta \mathrm{A})} (\mathrm{A} + \delta \mathrm{A}) - \mathrm{I}_N \right) \mathbf{x}_p + \mathbf{u} \tag{3.24}$$

Let us indicate with $\bar{\mathbf{v}}_p$ the target solution for the perturbed multi-agent system, then the input $\mathbf{u}$ can be selected according to Theorem 3.3.1. Considering again the difference between the final states of the nominal and perturbed multi-agent system, in this case we obtain:

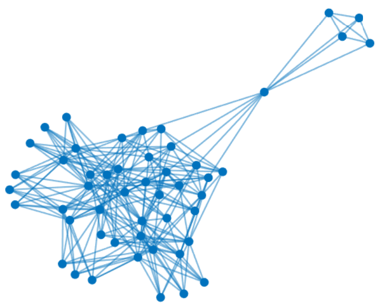
$$\|\delta \mathbf{x}\| = \|\langle \bar{\mathbf{v}}, \mathbf{v}_1 \rangle \mathbf{v}_1 - \langle \bar{\mathbf{v}}_p, \tilde{\mathbf{v}}_1 \rangle \tilde{\mathbf{v}}_1\| \tag{3.25}$$

which is no longer a function of the system initial conditions.

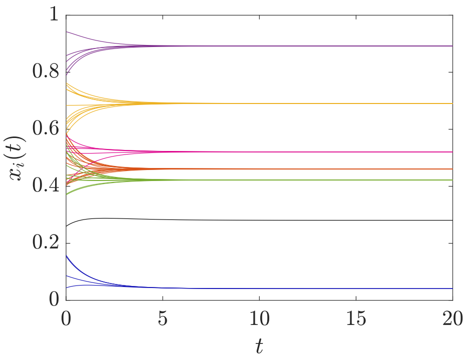
Next, we show an example of the effects of small perturbations affecting the interaction network on the multi-consensus dynamics.

Example 3.4.1 *Consider a multi-agent system (3.4) with* $N = 50$ *units, connected by a network with an average degree* $\langle k \rangle = 4$ *(Fig. 3.5(a)). In the absence of perturbations, the multi-agent system converges to the multi-consensus state of Fig. 3.5(b), where we observe seven different clusters that reflect the symmetries of the network.*

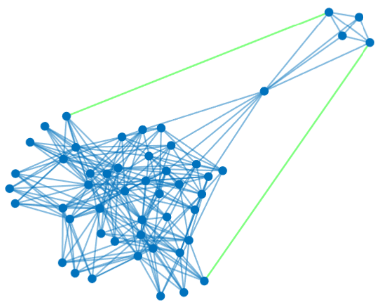
We then consider a perturbed network as in Fig. 3.5(c), obtained by adding to the original configuration the two links represented in green. The perturbed multi-agent system reaches a steady-state solution close to that of the nominal system (Fig. 3.5(d)), with $\|\delta \mathbf{x}\| = 0.087$. *This also provides an indication that the addition of the two links does not completely destroy the symmetry of the nominal network, as the difference between the steady-state solutions of the nominal and perturbed system is small.*



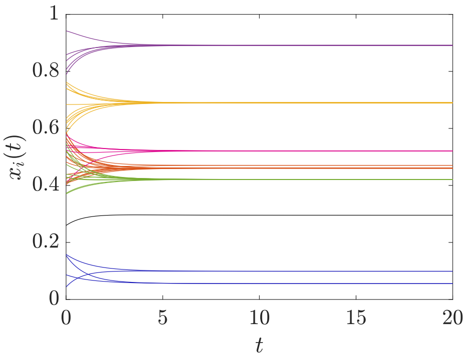
(a)



(b)



(c)



(d)

Figure 3.5: Multiconsensus and quasi-multiconsensus in a multi-agent system. Unperturbed graph (a) and perturbed graph (b). Dynamic behaviour of the unperturbed (c) and perturbed multi-agent system (d).

3.4.1 Revisiting the notion of quasi-symmetries

Quasi-symmetries or nearly symmetries naturally arise when networks are reconstructed from data that is noisy or inaccurate [64]. In such circumstances, it may happen that, when one swaps two or more links with similar weights (and the nodes connected to them), the resulting structure is almost the same as the original network. Similarly, it may happen that two or more nodes are not symmetric because of a single link (either missing or in excess), so that adding or removing an edge generates a network with an exact symmetry. Despite the importance of the notion of quasi-symmetries, it is not straightforward to provide a formal definition. As proposed in Ref. [64], recalling that a network admits an exact symmetry if its adjacency matrix A commutes with the permutation matrix R associated to the symmetry, i.e., when AR = RA, a possibility is to say that a quasi-symmetry exists if $\|\mathrm{AR} - \mathrm{RA}\| < \epsilon$, where ϵ is a small quantity. Similarly, the Authors of Ref. [65] define a correction cost associated with the perturbation that needs to be added to A to obtain an exact symmetry. However, in these definitions, the selection of the threshold value ϵ remains somewhat arbitrary. Taking into account the results of the previous section, it is possible to revisit the notion of quasi-symmetries and solve the ambiguity in the selection of this threshold. In fact, the relationship (3.21) links the perturbation on the agent interaction network, i.e., $\delta\mathrm{A}$, and the difference between the steady-states reached by the multi-agent system with unperturbed connectivity and that with perturbed connectivity. This relationship prompts a definition of quasi-symmetries that originates from their effect on the dynamics of the multi-agent system (3.3). In other words, in this way, one may recast the problem of defining a threshold for quasi-symmetries in the more practical terms of defining a bound for the deviation of the perturbed multiconsensus solution from the nominal case. In particular, from (3.21) it follows that

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}(0)\|} \leq \|\mathbf{v}_1\mathbf{v}_1^T - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\| \tag{3.26}$$

The previous inequality relates the difference in the steady-states of nominal and perturbed multi-agent system with a term, $\|\mathbf{v}_1\mathbf{v}_1^T - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\|$, that only depends on the topology of interaction. This allows us to define quasi-symmetries from their effect on the dynamical behavior of the multi-agent system (3.3). Suppose, in fact, that one can accept deviations from the nominal behavior bounded by $\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}(0)\|} < \epsilon_x$, then one can say that each symmetry of A is a quasi-symmetry for $\mathrm{A} + \delta\mathrm{A}$ if $\delta\mathrm{A}$ is such that $\|\mathbf{v}_1\mathbf{v}_1^T - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\| < \epsilon_x$.

3.5 Application to the rendezvous problem

As an application of the multi-consensus protocol introduced here, we consider a rendezvous problem in a bi-dimensional space where multiple rendezvous points have to be negotiated by the agents. Agents have single integrator dynamics and may rely on information on the locations of their neighbors to find an agreement on the point where to meet with the other units of the same cluster. The problem may be solved by applying, to each of the two variables describing the agent position, the multiconsensus protocol (3.7) if the rendezvous point is not given, or the multiconsensus protocol (3.4) if agents need to converge to a specific rendezvous point (of the multiconsensus manifold). Accordingly, we can indicate with x_i and y_i the coordinates of the position of agent i in the plane and with $\mathbf{x}$ and $\mathbf{y}$ the corresponding stack vectors, then the dynamics of the multi-agent system is written as:

$$\begin{aligned} \dot{\mathbf{x}} &= \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{x} + \mathbf{u}_x \\ \dot{\mathbf{y}} &= \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{y} + \mathbf{u}_y \end{aligned} \tag{3.27}$$

As the two variables $\mathbf{x}$ and $\mathbf{y}$ are independent, Lemma 3.2.3 and Theorem 3.3.1 may be applied to each of Eqs. (3.27), guaranteeing convergence to a multiconsensus solution.

Example 3.5.1 *Let us consider a system of $N = 15$ mobile agents connected via the network of Fig. 3.6(a), with $\mathbf{u}_x = \mathbf{u}_y = 0$. The symmetries of this network induce a partition $\pi = \{C_1, C_2, C_3\}$ with three cells: $C_1 = \{1, 4, 5, 6, 11, 13, 14\}$, $C_2 = \{2, 3, 7, 10, 15\}$, and $C_3 = \{8, 9, 12\}$.*

Since the network is connected, for Lemma 3.2.3 the final positions, towards which the agents will converge, here indicated as $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$, are determined by the agents initial conditions and the leading eigenvector of the adjacency matrix, that is: $\bar{\mathbf{x}} = c_1\mathbf{v}_1$ and $\bar{\mathbf{y}} = c_2\mathbf{v}_1$, where $c_1 = \mathbf{v}_1^T\mathbf{x}(0)$ and $c_2 = \mathbf{v}_1^T\mathbf{y}(0)$.

The trajectories described by the agents are illustrated in Fig. 3.6(b), where, in agreement with Lemma 3.2.3, agents belonging to the same cluster, converge to the same rendezvous point.

Suppose now that a pair of agents, for instance agents 8 and 14, becomes disconnected as shown in Fig. 3.6(c). As result of the missing link, the symmetry group of the network of interaction varies. However, from Eq. (3.20) we derive that:

$$\|\delta\mathbf{x}\| = \|\mathbf{v}_1\mathbf{v}_1^T\mathbf{x}(0) - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\mathbf{x}(0)\| = 0.095$$

$$\|\delta\mathbf{y}\| = \|\mathbf{v}_1\mathbf{v}_1^T\mathbf{y}(0) - \tilde{\mathbf{v}}_1\tilde{\mathbf{v}}_1^T\mathbf{y}(0)\| = 0.087$$

indicating that the rendezvous points of the units of the perturbed multi-agent system are close to those of the nominal multi-agent system:

$$\begin{aligned}\bar{\mathbf{x}} &= [0.28, 0.55, 0.55, 0.28, 0.28, 0.28, 0.55, 0.76,\\ &\quad 0.76, 0.55, 0.28, 0.76, 0.28, 0.28, 0.55]^T;\\ \bar{\mathbf{y}} &= [0.25, 0.49, 0.49, 0.25, 0.25, 0.25, 0.49, 0.69,\\ &\quad 0.69, 0.49, 0.25, 0.69, 0.25, 0.25, 0.49]^T\end{aligned}$$

This is confirmed by the analysis of the trajectories of the perturbed multi-agent system shown in Fig. 3.6(d), where agents in the same cluster reach points that are very close to each other and far from those reached by the agents belonging to other clusters.

Note that we applied our protocol under the assumption that the units interact according to a fixed interaction network. The latter does not depend on the agents' positions, meaning that it does not change as the agents move. As a result, our protocol can be applied in scenarios where the units are capable of communicating over long distances, ensuring that the interaction topology remains static regardless of the agents' relative positions.

In this chapter, by leveraging symmetry group theory, we have proved the convergence of the state variables of the multi-agent system to a steady-state solution parallel to the leading eigenvector of the graph adjacency matrix. As the entries of the leading eigenvector are the same for symmetrical nodes, the system splits into clusters formed of nodes that are symmetric from one to the other and converge to the same consensus value. Although this technique enables the system to reach multiconsensus, it does not allow for reaching a state that is parallel to an arbitrary vector belonging to the multiconsensus manifold. To address this issue, we have developed the communication protocol described in the next chapter.

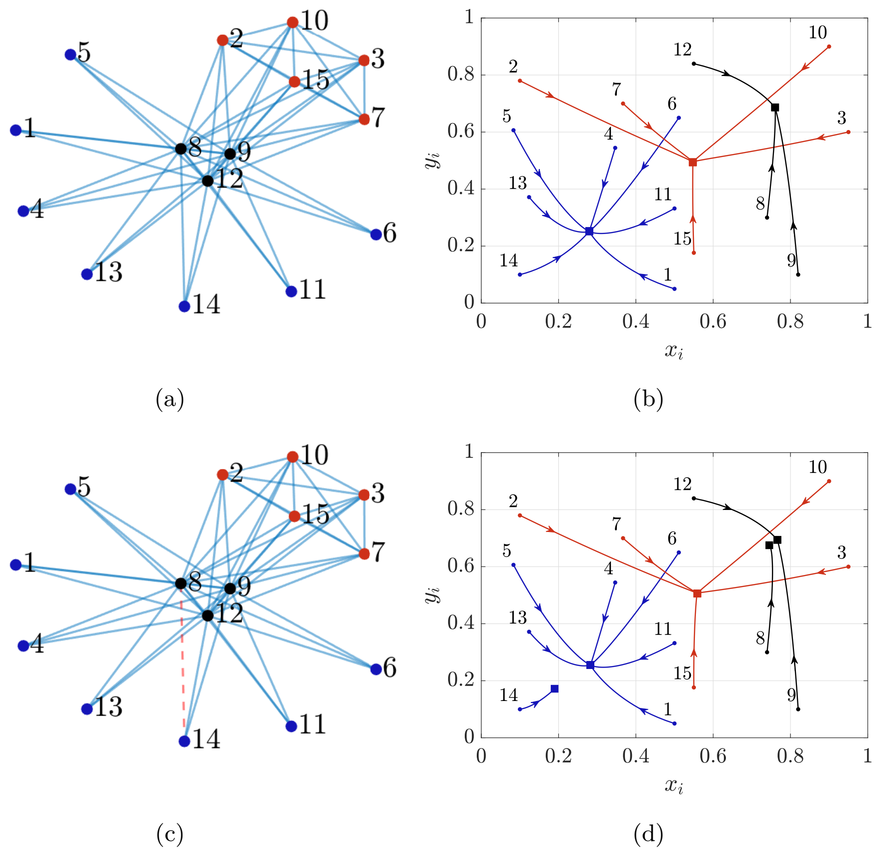


Figure 3.6: Multiple rendezvous problem addressed by the multiconsensus protocol (3.5.1). (a) Interaction network of the nominal multi-agent system. (b) Trajectories followed by the agents of the nominal multi-agent system. (c) Interaction network of the perturbed multi-agent system. (d) Trajectories followed by the agents of the perturbed multi-agent system. Circles and squares represent the initial and final points of the agent trajectory, respectively. Nodes belonging to the same cluster, and therefore symmetric, are shown with the same color.

Chapter 4

Control of multiconsensus in multi-agent systems based on eigenvector centrality

Here we focus on directed and weighted interaction graphs, where the units exhibit second-order dynamics, and we introduce a control strategy that allows the system to achieve an arbitrary multiconsensus solution. This strategy relies on a communication protocol for multiconsensus that induces the system to reach a state reflecting the eigenvector centrality of the nodes. In particular, our approach consists of two steps. First, the weights of the interaction network are assigned so that the network adjacency matrix has a given leading eigenvector, which corresponds to associating given values of eigenvector centrality to the nodes. This allows the formation of clusters of nodes that have the same value of eigenvector centrality. Second, the gains of the communication protocols are selected to guarantee the stability of the error dynamics. As in the previous chapter, we study both the cases of leaderless and leader-follower multiconsensus, providing analytical conditions for multiconsensus, and discussing a few numerical examples to illustrate our theoretical results.

4.1 Problem statement

Here, unlike in Chapter 3, where a multi-agent system with simple integrator dynamics is considered, we focus on a more general, yet still linear, configuration with second-order dynamics. This enables the study of more complex behaviors while preserving the linear nature of the system. The dynamics of each agent are described in controlled canonical form as follows:

$$\begin{aligned}
\dot{x}_i &= v_i \\
\dot{v}_i &= ax_i + bv_i + k_1\left(\frac{1}{\rho(\mathrm{A})}\sum_{j=1}^{N} a_{ij}w_{ij}x_j - x_i\right) + k_2\left(\frac{1}{\rho(\mathrm{A})}\sum_{j=1}^{N} a_{ij}w_{ij}v_j - v_i\right) + u_i
\end{aligned} \tag{4.1}$$

where the term u_i is an additive input. In the context of vehicle dynamics, the variables of the model x_i and v_i may represent, respectively, the position and the velocity of agent i, and the parameters a and b are the stiffness and damping factor. Since for $a < 0$ and $b < 0$ the isolated system is stable and the problem of multiconsensus is trivial, in the following we always consider that $a \geq 0$ or $b \geq 0$.

We assume that the agents interact according to a weighted graph with weighted adjacency matrix $\mathrm{A}' = \mathrm{A} \circ \mathrm{W}$, where the matrix A encodes the topology (i.e., $a_{ij} = 1$ if the information of agent j is used by agent i to update its status, and $a_{ij} = 0$ otherwise) and $w_{ij} \geq 0$ represents the weight of the link (i, j). We suppose that the topology is fixed but the edge weights w_{ij} can be tuned. The parameters k_1 and k_2 are also assumed to be tunable.

We assume that the agents know with which agents they communicate and the weights associated with those connections before the control protocol is implemented. At each step, the agents only need to recover the information on the status of the other units they are connected to, thus realizing a distributed control law. This approach is suitable for applications where the interaction graph is fixed and designed before running the control protocol.

System (4.1) can be equivalently written in matrix-vector form as follows:

$$\dot{\mathbf{y}}_i = \mathrm{A}_s\mathbf{y}_i + \mathbf{BK}\left(\sum_{j=1}^{N} a_{ij}w_{ij}\mathbf{y}_j - \mathbf{y}_i\right) + \mathbf{B}u_i \tag{4.2}$$

where $\mathbf{y}_i = \begin{bmatrix} x_i \\ v_i \end{bmatrix}$, $\mathrm{A}_s = \begin{bmatrix} 0 & 1 \\ a & b \end{bmatrix}$, $\mathbf{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $\mathbf{K} = \begin{bmatrix} k_1 & k_2 \end{bmatrix}$.

Finally, system (4.2) is rewritten in compact form:

$$\dot{\mathbf{y}} = (\mathrm{I}_N \otimes \mathrm{A}_s)\mathbf{y} + (\mathrm{A}_\rho \otimes \mathbf{BK})\mathbf{y} + (\mathrm{I}_N \otimes \mathbf{B})\mathbf{u} \tag{4.3}$$

where $\mathbf{y} = [\mathbf{y}_1^T, \dots, \mathbf{y}_N^T]^T$, $\mathbf{u} = [u_1, \dots, u_N]^T$, and $\mathrm{A}_\rho = \frac{1}{\rho(\mathrm{A})}(\mathrm{A} \circ \mathrm{W}) - \mathrm{I}_N$. Similarly, we define $\mathbf{x} = [x_1, \dots, x_N]^T$ and $\mathbf{v} = [v_1, \dots, v_N]^T$.

The classical notion of multiconsensus applied to the multi-agent system (4.1) requires that, given a partition $\pi = \{C_1, \dots, C_Q\}$, $\forall i, j \in C_h, h = 1, \dots, Q$ one has that

$$\begin{aligned} &\lim_{t \to +\infty} (x_i(t) - x_j(t)) = 0 \\ &\lim_{t \to +\infty} (v_i(t) - v_j(t)) = 0. \end{aligned} \tag{4.4}$$

We extend this notion by defining the multiconsensus associated with π and $\hat{\mathbf{p}}$. Indeed, as discussed in the previous chapter, a partition π can be associated with the multiconsensus manifold $\mathbb{MC}(\pi)$, defined by Def. 3.2.1. Here we want that the system reaches a trajectory reflecting a specific vector $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$, therefore we introduce the concept of $(\pi, \hat{\mathbf{p}})$-multiconsensus.

Definition 4.1.1 *Given a partition* $\pi = \{C_1, \dots, C_Q\}$ *and a vector* $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$*, we say that system (4.1) reaches the multiconsensus associated to* π *and* $\hat{\mathbf{p}}$*, shortly indicated as* $(\pi, \hat{\mathbf{p}})$*-multiconsensus, if* $\forall i, j = 1, \dots, N$ *we have that*

$$\begin{aligned} &\lim_{t \to +\infty} \left(\tfrac{x_i(t)}{\hat{p}_i} - \tfrac{x_j(t)}{\hat{p}_j}\right) = 0 \\ &\lim_{t \to +\infty} \left(\tfrac{v_i(t)}{\hat{p}_i} - \tfrac{v_j(t)}{\hat{p}_j}\right) = 0. \end{aligned} \tag{4.5}$$

This definition generalizes the notion of scaled consensus introduced in [86], which proposed a general protocol for consensus, where the aim is to reach an agreement for scaled variables. The case of multiconsensus is recovered by selecting equal scaling factors.

By setting $\hat{p}_i = \hat{p}_j \ \forall i, j \in C_h, h = 1, \dots, Q$, nodes belonging to the same cell will asymptotically follow the same trajectory in time as in the classical multiconsensus. However, in the case of $(\pi, \hat{\mathbf{p}})$-multiconsensus it is also required that the system trajectory becomes asymptotically parallel to the vector $\hat{\mathbf{p}}$. In this way, one can specify not only which agents cluster together, but also the scaling ratios [86] between trajectories of agents in different clusters. Note also that both Eqs. (4.4) and (4.5) do not necessarily require that the system state asymptotically converges to an equilibrium point, but only that nodes of the same cell asymptotically converge on the same trajectory.

The first problem that we investigate is the *leaderless multiconsensus problem* defined as follows:

Problem 4.1.2 (Leaderless multiconsensus problem) *Given a partition π and a vector $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$, find the edge weights w_{ij} and the gains k_1 and k_2 such that the system* (4.1) *reaches the* $(\pi, \hat{\mathbf{p}})$*-multiconsensus.*

We will show that a proper selection of the weights guarantees the existence of a solution with the state parallel to $\hat{\mathbf{p}}$, while it is the choice of k_1 and k_2 that guarantees that the system state effectively converges to a solution of this type. In the case of the $(\pi, \hat{\mathbf{p}})$-multiconsensus not only two agents of the same cell will converge to the same trajectory, but, in addition, if two agents i and j belong to different cells, that is, $p_i \neq p_j$, then they will converge to different trajectories.

The second problem that we study is the *leader multiconsensus problem* for which the goal is to control the multi-agent system to have the agents within the same cell follow a specific trajectory generated by a leader/reference extra node.

Problem 4.1.3 *Given a partition π, a vector $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$, an arbitrary scalar constant $\bar{c} \in \mathbb{R}$, and a leader with dynamics expressed by:*

$$\begin{aligned} \dot{x}_{ref} &= v_{ref} \\ \dot{v}_{ref} &= a x_{ref} + b v_{ref} \end{aligned} \tag{4.6}$$

find the edge weights w_{ij} and the gains k_1 and k_2 such that

$$\begin{aligned} \lim_{t \to +\infty} \mathbf{x}(t) = \bar{c} x_{ref}(t) \hat{\mathbf{p}} \\ \lim_{t \to +\infty} \mathbf{v}(t) = \bar{c} v_{ref}(t) \hat{\mathbf{p}} \end{aligned} \tag{4.7}$$

We anticipate that the first problem can be solved using $u_i = 0$, whereas a non-zero input is required to solve the second problem.

4.2 Leaderless multiconsensus

In this section, we study the problem of leaderless multiconsensus, by which we mean the case in which u_i is set to zero in Eq. (4.1). We will first introduce a lemma that shows how to set the link weights of a graph in order to achieve an arbitrary leading eigenvector of the adjacency matrix. When applied to system (4.1) with zero input, we demonstrate the existence of an invariant solution that is parallel to this arbitrary eigenvector. For brevity, we refer to this solution as the *multiconsensus solution*. We will move then to prove how to select the gains k_1 and k_2 such that the system state asymptotically converges to the multiconsensus solution.

Let us consider a graph $\mathcal{G}$ that is unweighted and strongly connected. Therefore, its adjacency matrix A is non-negative and irreducible, and, for the Perron-Frobenius theorem (Theorem 2.3.1), it has a leading eigenvector $\mathbf{v}_1 > \mathbf{0}$. Next we prove that it is possible to assign weights to the links of the graph such that the adjacency matrix of the resulting weighted graph has an arbitrary leading eigenvector $\mathbf{v}_1' > \mathbf{0}$.

Lemma 4.2.1 *Let* A *be the adjacency matrix of an unweighted, strongly connected graph* $\mathcal{G}$*,* $\rho(\mathrm{A})$ *its leading eigenvalue, and* $\mathbf{v}_1'$ *a vector with positive elements. Then, it is possible to find a matrix* W *such that the adjacency matrix* $\mathrm{A}' = \mathrm{A} \circ \mathrm{W}$ *of a weighted graph with the same topology as* $\mathcal{G}$ *has leading eigenvector equal to* $\mathbf{v}_1'$ *and leading eivenvalue equal to* $\rho(\mathrm{A})$*.*

Proof. $\mathbf{v}_1'$ is the leading vector of $\mathrm{A}' = \mathrm{A} \circ \mathrm{W}$, with associated eigenvalue given by $\rho(\mathrm{A})$, if

$$(\mathrm{A} \circ \mathrm{W})\mathbf{v}_1' = \rho(\mathrm{A})\mathbf{v}_1' \tag{4.8}$$

Eq. (4.8) can be recast by vectorization [56] as follows:

$$({\mathbf{v}_1'}^T \otimes \mathrm{I}_N)(\boldsymbol{\alpha} \circ \boldsymbol{\omega}) = \rho(\mathrm{A})\mathbf{v}_1' \tag{4.9}$$

where $\boldsymbol{\alpha} = vec(\mathrm{A})$ and $\boldsymbol{\omega} = vec(\mathrm{W})$. Now, let $\mathrm{M} = ({\mathbf{v}_1'}^T \otimes \mathrm{I}_N)$. Since $\boldsymbol{\alpha}$ is a binary vector, then $\mathrm{M}(\boldsymbol{\alpha} \circ \boldsymbol{\omega}) = \hat{\mathrm{M}}\hat{\boldsymbol{\omega}}$, where $\hat{\mathrm{M}}$ is obtained from M by removing each column whose index i is such that $\alpha_i = 0$, and, similarly, $\hat{\boldsymbol{\omega}}$ is obtained from $\boldsymbol{\omega}$ by eliminating the coefficients with row index i such that $\alpha_i = 0$.

Hence, in order to find W such that $\mathbf{v}'_1$ is the leading eigenvector, associated with $\rho(\mathrm{A})$, the following system of linear equations where the unknowns are the elements of $\hat{\boldsymbol{\omega}}$ must be solved:

$$\hat{\mathrm{M}}\hat{\boldsymbol{\omega}} = \rho(\mathrm{A})\mathbf{v}'_1 \tag{4.10}$$

Notice that $\hat{\mathrm{M}} \in \mathbb{R}^{N \times L}$, where L indicates the number of links. This matrix has a peculiar structure. In fact, since $\mathrm{M} = {\mathbf{v}'_1}^T \otimes \mathrm{I}_N$ and since, by the Perron-Frobenius theorem, $\mathbf{v}'_1$ is a vector of positive elements, then each column of M has a single non-zero element. Consequently, also $\hat{\mathrm{M}}$ has at most a single non-zero element in each column and is a non-negative matrix. Then, it follows that, when, by multiplying in Eq. (4.10) each row of $\hat{\mathrm{M}}$ by the vector $\hat{\boldsymbol{\omega}}$, which contains the unknowns of our problem, we get a set of independent equations where each of the L unknowns appears only in one equation. In addition, as $L > N$ the system of equations (4.10) is underdetermined. Since $\rho(\mathrm{A})\mathbf{v}'_1 > \mathbf{0}$, for Farka's lemma (Lemma 2.3.7), a non-negative solution always exists. In particular, the solution with minimum L2 norm is given by:

$$\hat{\boldsymbol{\omega}} = \hat{\mathrm{M}}^{+}\rho(\mathrm{A})\mathbf{v}'_1 \tag{4.11}$$

where $\hat{\mathrm{M}}^{+}$ is the Moore-Penrose inverse of the matrix $\hat{\mathrm{M}}$. In addition, since $\hat{\boldsymbol{\omega}} > \mathbf{0}$, the matrix $\mathrm{A}' = \mathrm{A} \circ \mathrm{W}$ is also non-negative and irreducible. □

From Lemma 4.2.1 it follows that $\mathbf{y} = \mathbf{v}'_1 \otimes \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$, where c_1 and c_2 are functions that depend on the initial conditions, is an invariant solution for Eq. (4.3) with $\mathbf{u} = 0$. In fact, for $\mathbf{y} = \mathbf{v}'_1 \otimes \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$ the coupling term in Eq. (4.3) vanishes, as

$$(\mathrm{A}_\rho \otimes \mathbf{BK}) \left(\mathbf{v}'_1 \otimes \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \right) = \left(\tfrac{1}{\rho(\mathrm{A})}(\mathrm{A} \circ \mathrm{W}) - \mathrm{I}_N \right) \mathbf{v}'_1 \otimes \mathbf{BK} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = 0 \tag{4.12}$$

Next, in the following theorem, we provide conditions on k_1 and k_2 such that the system state asymptotically converges to the multiconsensus solution.

Theorem 4.2.2 *Given an arbitrary* $(\pi, \hat{\mathbf{p}})$*-multiconsensus with* $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$*, the multi-agent system* (4.3)*, with* W *as in Lemma 4.2.1 and* $\mathbf{u} = 0$*, reaches the* $(\pi, \hat{\mathbf{p}})$*-multiconsensus, if and only if* $(k_1, k_2) \in S$*, where*

$$S = \{(k_1, k_2) \in \mathbb{R}^2 \mid \gamma_j < 0,\ \alpha_j \gamma_j + \beta_j^2 + \beta_j \gamma_j \delta_j < 0,\ \forall j = 2, \dots, N\}$$

and $\alpha_j = Re(a + \lambda_j k_1)$*,* $\beta_j = Im(a + \lambda_j k_1)$*,* $\gamma_j = Re(b + \lambda_j k_2)$*,* $\delta_j = Im(b + \lambda_j k_2)$*, with* λ_j $(j = 2, \dots, N)$ *being the ordered non-zero eigenvalues of* A_ρ*.*

Proof. Let $\mathrm{R} = \mathrm{I}_N - \mathbf{v}_1' \mathbf{v}_1'^T$ and let $\mathbf{z}$ be the error variables defined by

$$\mathbf{z} = (\mathrm{R} \otimes \mathrm{I}_2)\mathbf{y} \tag{4.13}$$

The previous relationship yields that $\mathbf{z} = \mathbf{y} - \mathbf{v}_1' \otimes \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$, where $c_1 = \mathbf{v}_1'^T \mathbf{x}$ and $c_2 = \mathbf{v}_1'^T \mathbf{v}$. Thus, $\mathbf{z} = \mathbf{0}$ if and only if the two vectors $\mathbf{x}(t)$ and $\mathbf{v}(t)$ are parallel at any time t to $\mathbf{v}_1'$.

Since $\mathrm{R}\mathbf{v}_1' = \mathbf{0}$ and $\mathrm{A}_\rho \mathbf{v}_1' = \mathbf{0}$, then the error dynamics can be expressed as:

$$\begin{aligned} \dot{\mathbf{z}} &= (\mathrm{R} \otimes \mathrm{I}_2)\dot{\mathbf{y}} = (\mathrm{R} \otimes \mathrm{A}_s)\mathbf{y} + (\mathrm{R}\mathrm{A}_\rho \otimes \mathbf{BK})\mathbf{y} \\ &= (\mathrm{R} \otimes \mathrm{A}_s)\mathbf{z} + (\mathrm{R}\mathrm{A}_\rho \otimes \mathbf{BK})\mathbf{z} \end{aligned} \tag{4.14}$$

For simplicity, let us suppose now that A_ρ is diagonalizable. We can define new variables $\bar{\mathbf{z}} = (\mathrm{T}^{-1} \otimes \mathrm{I}_2)\mathbf{z}$, where $\mathrm{T} \in C^{N \times N}$ is the matrix containing, in each column, the left eigenvectors of A_ρ. The error dynamics in the new variables becomes:

$$\dot{\bar{\mathbf{z}}} = (\mathrm{T}^{-1}\mathrm{R}\mathrm{T} \otimes \mathrm{A}_s)\bar{\mathbf{z}} + (\mathrm{T}^{-1}\mathrm{R}\mathrm{A}_\rho\mathrm{T} \otimes \mathbf{BK})\bar{\mathbf{z}} \tag{4.15}$$

Since $\mathbf{v_1}' = \mathrm{T}[1, 0, 0, \dots, 0]^T$, it follows that:

$$\mathrm{T}^{-1}\mathrm{R}\mathrm{T} = \begin{bmatrix} 0 & -\hat{\mathbf{v}} \\ \mathbf{0_{N-1}} & \mathrm{I}_{N-1} \end{bmatrix},\ \mathrm{T}^{-1}\mathrm{R}\mathrm{A}_\rho\mathrm{T} = \begin{bmatrix} 0 & -\hat{\mathbf{v}}\Lambda \\ \mathbf{0_{N-1}} & \Lambda \end{bmatrix}$$

with $\hat{\mathbf{v}} = [\mathbf{v}_1^T\mathbf{v}_2, \ldots, \mathbf{v}_1^T\mathbf{v}_N]$ and $\Lambda = \mathrm{diag}(\lambda_2, \ldots, \lambda_N)$. It is worth noticing that since these two matrices are lower-triangular, then $\mathrm{P} = \mathrm{T}^{-1}\mathrm{RT} \otimes \mathrm{A}_s + \mathrm{T}^{-1}\mathrm{RA}_\rho\mathrm{T} \otimes \mathbf{BK}$ is a block lower-triangular matrix where each block P_j ($j = 1, \ldots, N$) in the diagonal has the following form:

$$\mathrm{P}_j = \begin{bmatrix} 0 & 1 \\ a + \lambda_j k_1 & b + \lambda_j k_2 \end{bmatrix} \tag{4.16}$$

In particular, the block P_1 is associated to the error dynamics along the multiconsensus manifold, while the other ones ($j = 2, \ldots, N$) are related to the error dynamics along the motions which are transverse to this manifold. Since the target is the multiconsensus, k_1 and k_2 must be selected so that all the blocks P_j, except the first one, are Hurwitz stable.

The characteristic polynomial of P_j is

$$p_j(s) = s^2 - (b + \lambda_j k_2)s - a - \lambda_j k_1, \tag{4.17}$$

that can be rewritten as

$$p_j(s) = s^2 - (\gamma_j + i\delta_j)s - \alpha_j - i\beta_j, \tag{4.18}$$

where $\alpha_j = Re(a + \lambda_j k_1)$, $\beta_j = Im(a + \lambda_j k_1)$, $\gamma_j = Re(b + \lambda_j k_2)$ and $\delta_j = Im(b + \lambda_j k_2)$. As stated in [73], the polynomial (4.18) has negative roots if and only if the following conditions hold:

$$\begin{cases} f_{1,j}(k_2) = \gamma_j < 0 \\ f_{2,j}(k_1, k_2) = \alpha_j\gamma_j + \beta_j^2 + \beta_j\gamma_j\delta_j < 0 \end{cases} \tag{4.19}$$

Since $f_{1,j}$ and $f_{2,j}$ depend on k_1 and k_2, for each $j = 1, \ldots, N$, we can define the following sets:

$$S_j = \{(k_1, k_2) \in \mathbb{R}^2 \mid f_{1,j} < 0,\ f_{2,j} < 0\} \tag{4.20}$$

Therefore, the system described by equation (4.3) reaches multiconsensus if and only if $(k_1, k_2) \in S$, where S is the set defined as:

$$S = \bigcap_{j=2}^{N} S_j = \{(k_1, k_2) \in \mathbb{R}^2 \mid f_{1,j} < 0,\ f_{2,j} < 0,\ \forall j = 2, \ldots, N\} \tag{4.21}$$

In the case when A_ρ is not diagonalizable, the same result follows by considering the Jordan decomposition of A_ρ rather than its diagonalization. This yields a matrix P that is still block triangular, but eventually contains some blocks of dimension larger than 2. The analysis of stability of the error dynamics can then be carried out following arguments similar to those in [67]. $\square$

Example 4.2.3 *We consider a multi-agent system as in Eq.* (4.1) *in the absence of control, i.e.,* $u_i = 0$*, with* $a = -0.01$ *and* $b = 0$*. The graph associated to the adjacency matrix* A *is shown in Fig. 4.1(a). Consider also the partition*

$$\pi = \{\{1,2\},\{3,4\},\{5,6\},\{7,8\}\} \tag{4.22}$$

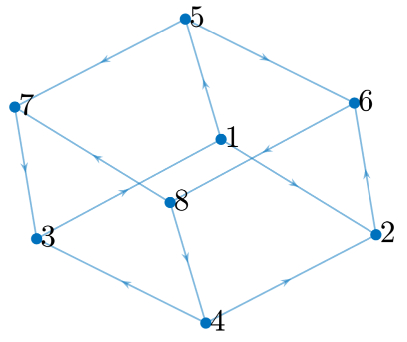
and the associated vector:

$$\hat{\mathbf{p}} = [0.13, 0.13, 0.39, 0.39, 0.52, 0.52, 0.26, 0.26]^T \tag{4.23}$$

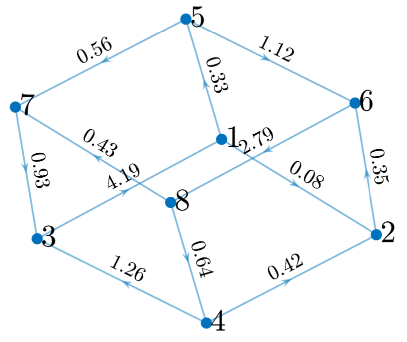
Notice that, when all weights are one, i.e., when $w_{ij} = 1\ \forall i,j$*, the multi-agent system* (4.1) *does not reach the multiconsensus associated to* π*, but instead converges to a different multiconsensus solution, defined by* $\pi' = \{\{1,8\},$ $\{2,7\},\ \{3,6\},\{4,5\}\}$ *and induced by the existing symmetries of the interaction graph. In order to ensure the existence of the multiconsensus solution* π*, one can act on the weights* w_{ij} *according to Lemma* 4.2.1*. In more detail, solving* (4.11) *yields the weights reported in Fig. 4.1(b).*

Notice that, at variance from methods for multiconsensus based on external equitable partitions [63, 29], here only a single eigenvalue is zero.

The next step is to select the values of k_1 *and* k_2 *such that the system state asymptotically converges to the multiconsensus solution. To this purpose, Theorem 4.2.2 is applied, obtaining the set* S *of all pairs* (k_1, k_2) *such that the error* (4.13) *asymptotically converges to zero. Fig. 4.2(a) shows two regions of the* (k_1, k_2) *plane, a stability region in blue where the conditions* (4.19) *are satisfied and an instability region in red where the conditions* (4.19) *are not satisfied. Fig. 4.2(b) shows the temporal evolution of* $\|\mathbf{z}(t)\|$ *(main panel) and of the state variables* $x_i(t)$ *(inset) for a point* $P_1 = (k_1, k_2) = (0.24, 3)$ *of the stability region. Here the multi-agent system reaches multiconsensus and agents belonging to the same cluster have oscillations of equal amplitude. Furthermore, once multiconsensus has been* *reached, the trajectories are at any time parallel to the vector* $\hat{\mathbf{p}}$. *Finally, Fig. 4.2(c) shows the system temporal evolution corresponding to the point* $P_2 = (k_1, k_2) = (2, 1.4)$, *that lies in the instability region, wherein the system does not reach multiconsensus. Correspondingly,* $\|\mathbf{z}(t)\|$ *increases over time (main panel of Fig. 4.2(c)) and the state variables* $x_i(t)$ *of agents within each cluster (inset of Fig. 4.2(c)) display oscillations that are different from each other and have growing amplitudes.*



(a)



(b)

Figure 4.1: (a) Unweighted interaction graph of the multi-agent system (4.1) in Example 4.2.3. (b) Corresponding graph with weights obtained applying Lemma 4.2.1 to guarantee the existence of multiconsensus solution associated to π as in (4.22) and $\hat{\mathbf{p}}$ in (4.23).

4.3 Leader multiconsensus

In this section, we consider the problem of leader multiconsensus, namely the problem of finding the values of the edge weights w_{ij} and of the gains k_1 and k_2 such that Eq. (4.7) holds, where x_{ref} and v_{ref} are the state variables of a reference node having dynamics described by (4.6).

Theorem 4.3.1 *Consider the multi-agent system* (4.2) *and assume that the interaction graph* $\mathcal{G}$ *is strongly connected. Let the reference trajectory (target of the control) be given by:*

$$\hat{\mathbf{y}} = \hat{\mathbf{p}} \otimes \bar{c}\mathbf{y_{ref}} \tag{4.24}$$

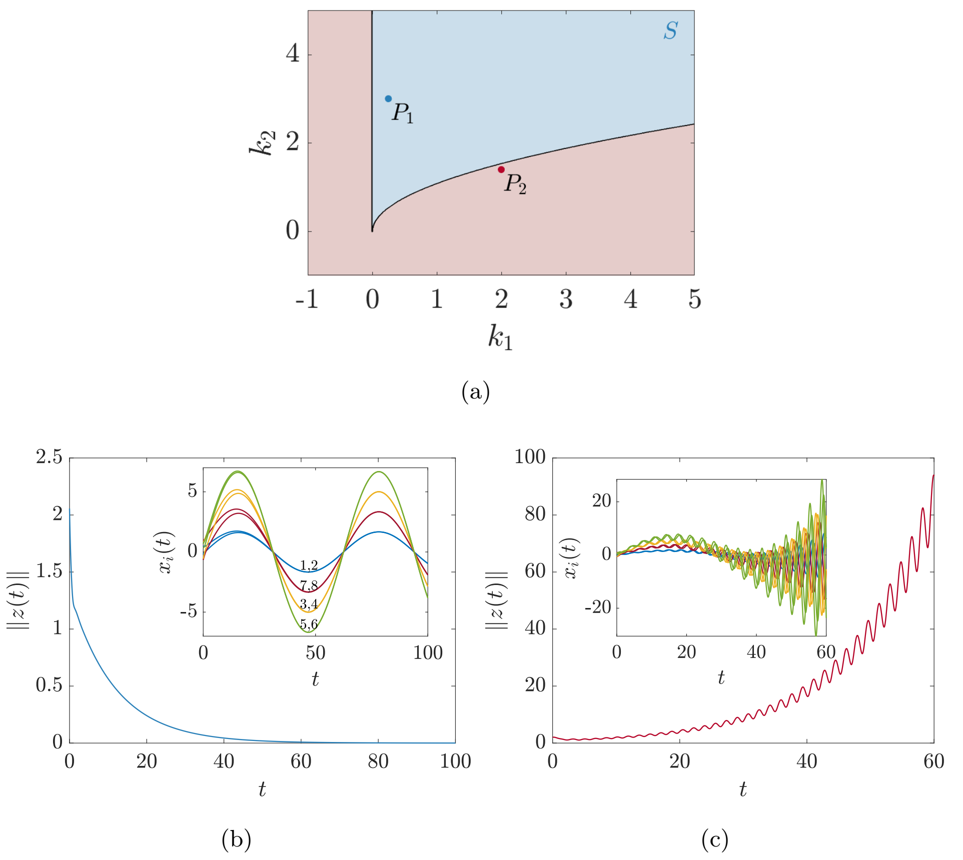


Figure 4.2: Leaderless multiconsensus in the multi-agent system of Example 4.2.3: stability region (a), temporal evolution of $\|\mathbf{z}(t)\|$ (main panels) and $x_i(t)$ (insets) corresponding to points P_1 in the stability region (b) and P_2 in the instability region (c).

where $\hat{\mathbf{p}} \in \mathbb{MC}(\pi)$ *is a vector associated to the partition* π *and* $\mathbf{y_{ref}} = \begin{bmatrix} x_{ref} \\ v_{ref} \end{bmatrix}$ *with* x_{ref} *and* y_{ref} *being the state variables of a reference node evolving based on Eq.* (4.6)*. In addition, let the matrix* W *be chosen as in Lemma 4.2.1,* $\bar{i}$ *be an arbitrary node of the graph and the control input* u_i *be defined as follows:*

$$u_i = \xi_i[k_1(\bar{c}\hat{p}_{\bar{i}}x_{ref} - x_i) + k_2(\bar{c}\hat{p}_{\bar{i}}v_{ref} - v_i)] \tag{4.25}$$

where $\xi_i = 1$ *if* $i = \bar{i}$*,* $\xi_i = 0$ *otherwise. Then, the error* $\hat{\mathbf{z}} = \mathbf{y} - \hat{\mathbf{y}}$

asymptotically converges to $\mathbf{0}$ *if and only if* $(k_1, k_2) \in \hat{S}$*, where:*

$$\hat{S} = \{(k_1, k_2) \in \mathbb{R}^2 \mid \hat{\gamma}_j < 0,\ \hat{\alpha}_j \gamma_j + \hat{\beta}_j^2 + \hat{\beta}_j \hat{\gamma}_j \hat{\delta}_j < 0,\ \forall j = 1, \ldots, N\}$$

and $\hat{\alpha}_j = Re(a + \mu_j k_1)$*,* $\hat{\beta}_j = Im(a + \mu_j k_1)$*,* $\hat{\gamma}_j = Re(b + \mu_j k_2)$*,* $\hat{\delta}_j = Im(b + \mu_j k_2)$*, with* μ_j *(*$j = 1, \ldots, N$*) being the ordered eigenvalues of* $\mathrm{A}_\rho - \mathrm{diag}(\xi_1, \ldots \xi_N)$*.*

Proof. The control law in Eq. (4.25) can be rewritten as

$$\mathbf{u} = (\bar{c}\hat{p}_i \mathbf{e}_{\bar{\mathbf{i}}} \otimes \mathbf{K})\mathbf{y_{ref}} - (\mathrm{D}_c \otimes \mathbf{K})\mathbf{y} \tag{4.26}$$

where $\mathrm{D}_c = \mathrm{diag}(\xi_1, \xi_2, \ldots, \xi_N)$, and $\mathbf{e}_{\bar{\mathbf{i}}} = [\xi_1, \xi_2, \ldots, \xi_N]^T$ that is the $\bar{i}$-th canonical vector. Consequently, Eq. (4.3) becomes:

$$\dot{\mathbf{y}} = (\mathrm{I}_N \otimes \mathrm{A}_s)\mathbf{y} + ((\mathrm{A}_\rho - \mathrm{D}_c) \otimes \mathbf{BK})\mathbf{y} + (\bar{c}\hat{p}_{\bar{i}} \mathbf{e}_{\bar{\mathbf{i}}} \otimes \mathbf{BK})\mathbf{y_{ref}}$$

Substituting $\mathbf{y} = \hat{\mathbf{z}} + \hat{\mathbf{y}}$ in the previous expression, we obtain:

$$\begin{aligned} \dot{\hat{\mathbf{z}}} + \dot{\hat{\mathbf{y}}} &= (\mathrm{I}_N \otimes \mathrm{A}_s)(\hat{\mathbf{z}} + \hat{\mathbf{y}}) + ((\mathrm{A}_\rho - \mathrm{D}_c) \otimes \mathbf{BK})(\hat{\mathbf{z}} + \hat{\mathbf{y}}) \\ &\quad + (\bar{c}\hat{p}_{\bar{i}} \mathbf{e}_{\bar{\mathbf{i}}} \otimes \mathbf{BK})\mathbf{y_{ref}} \end{aligned} \tag{4.27}$$

Since $(\mathrm{D}_c \otimes \mathbf{BK})\hat{\mathbf{y}} = (\bar{c}\hat{p}_{\bar{i}} \mathbf{e}_{\bar{\mathbf{i}}} \otimes \mathbf{BK})\mathbf{y_{ref}}$, $\dot{\hat{\mathbf{y}}} = (\mathrm{I}_N \otimes \mathrm{A}_s)\hat{\mathbf{y}}$, and $(\mathrm{A}_\rho \otimes \mathbf{BK})\hat{\mathbf{y}} = \mathbf{0}$, Eq. (4.27) can be written as:

$$\dot{\hat{\mathbf{z}}} = (\mathrm{I}_N \otimes \mathrm{A}_s)\hat{\mathbf{z}} + ((\mathrm{A}_\rho - \mathrm{D}_c) \otimes \mathbf{BK})\hat{\mathbf{z}} \tag{4.28}$$

This equation represents the error dynamics. We now provide proof of the stability of this dynamics under the assumption that $\mathrm{A}_\rho - \mathrm{D}_c$ is diagonalizable. In this case, we can define new variables $\tilde{\mathbf{z}} = (\hat{\mathrm{T}}^{-1} \otimes \mathrm{I}_2)\hat{\mathbf{z}}$, where $\hat{\mathrm{T}} \in \mathbb{C}^{N \times N}$ is the matrix containing, in each column, the left eigenvectors of $\mathrm{A}_\rho - \mathrm{D}_c$, and rewrite the error dynamics as:

$$\dot{\tilde{\mathbf{z}}} = (\mathrm{I}_N \otimes \mathrm{A}_s)\tilde{\mathbf{z}} + (\mathrm{G} \otimes \mathbf{BK})\tilde{\mathbf{z}} \tag{4.29}$$

where $\mathrm{G} = \hat{\mathrm{T}}^{-1}(\mathrm{A}_\rho - \mathrm{D}_c)\hat{\mathrm{T}}$ is the diagonal matrix containing the eigenvalues of $\mathrm{A}_\rho - \mathrm{D}_c$.

We now prove that $\mathrm{A}_\rho - \mathrm{D}_c$ (and so G) is Hurwitz stable. To this aim, let us introduce an M-matrix, namely $\hat{\mathrm{A}}_\rho$, defined as:

$$\hat{\mathrm{A}}_\rho = \rho(\mathrm{A})\mathrm{I}_N - (\mathrm{A} \circ \mathrm{W}) = -\rho(\mathrm{A})\mathrm{A}_\rho \tag{4.30}$$

and let us also consider the matrix F:

$$\mathrm{F} = \hat{\mathrm{A}}_\rho + \rho(\mathrm{A})\mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T = -\rho(\mathrm{A})(\mathrm{A}_\rho - \mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T) \tag{4.31}$$

As $\mathrm{A} \circ \mathrm{W}$ is non-negative and irreducible, the zero eigenvalue of $\hat{\mathrm{A}}_\rho$ is associated with positive right and left eigenvectors, therefore $\mathbf{e}_{\bar{\mathbf{i}}}$ satisfies the NZP condition (2.1). Therefore, all the hypotheses of Theorem 2.3.6 are verified, and all the leading minors of F are positive. For Theorem 2.3.4, it follows that the matrix $\mathrm{A}_\rho - \mathbf{e}_{\bar{\mathbf{i}}}\mathbf{e}_{\bar{\mathbf{i}}}^T = \mathrm{A}_\rho - \mathrm{D}_c$ is Hurwitz stable.

The values of k_1 and k_2 have to be selected to enforce stability of the error dynamics. This can be done by finding the conditions guaranteeing that the matrix $\mathrm{J} = \mathrm{I}_N \otimes \mathrm{A}_s + \mathrm{G} \otimes \mathbf{BK}$ is Hurwitz stable. Since J is block diagonal, its stability can be studied by considering the eigenvalues of each block:

$$\mathrm{J}_j = \begin{bmatrix} 0 & 1 \\ a + \mu_j k_1 & b + \mu_j k_2 \end{bmatrix} \tag{4.32}$$

The characteristic polynomial of J_j is

$$\hat{p}_j(s) = s^2 - (b + \mu_j k_2)s - a - \mu_j k_1 \tag{4.33}$$

that can be rewritten as

$$\hat{p}_j(s) = s^2 - (\hat{\gamma}_j + i\hat{\delta}_j)s - \hat{\alpha}_j - i\hat{\beta}_j \tag{4.34}$$

where $\hat{\alpha}_j = a + k_1 Re(\mu_j)$, $\hat{\beta}_j = a + k_1 Im(\mu_j)$, $\hat{\gamma}_j = b + k_2 Re(\mu_j)$ and $\hat{\delta}_j = b + k_2 Im(\mu_j)$. The polynomial (4.33) has roots with negative real part if and only if the following conditions hold [73]:

$$\begin{cases} \hat{f}_{1,j}(k_2) = \hat{\gamma}_j < 0 \\ \hat{f}_{2,j}(k_1, k_2) = \hat{\alpha}_j\hat{\gamma}_j + \hat{\beta}_j^2 + \hat{\beta}_j\hat{\gamma}_j\hat{\delta}_j < 0 \end{cases} \tag{4.35}$$

Since $\hat{f}_{1,j}$ and $\hat{f}_{2,j}$ depend on k_1 and k_2, for each $j = 1, \dots, N$, we can define the following sets:

$$\hat{S}_j = \{(k_1, k_2) \in \mathbb{R}^2 \mid \hat{f}_{1,j} < 0,\ \hat{f}_{2,j} < 0\} \tag{4.36}$$

We conclude that $\lim_{t \to +\infty} \hat{\mathbf{z}}(t) = \mathbf{0}$ if and only if $(k_1, k_2) \in \hat{S}$, where $\hat{S}$ is given by:

$$\hat{S} = \bigcap_{j=1}^{N} \hat{S}_j = \{(k_1, k_2) \in \mathbb{R}^2 \mid \hat{f}_{1,j} < 0,\ \hat{f}_{2,j} < 0,\ \forall j = 1, \dots, N\} \tag{4.37}$$

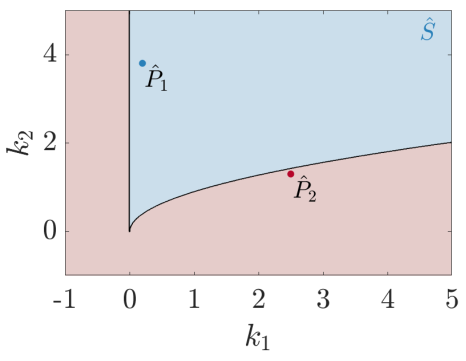
□

Example 4.3.2 *Let us consider again the multi-agent system of Example 4.2.3, with the aim of designing a control action u_i to obtain the desired multiconsensus dynamics given by:*

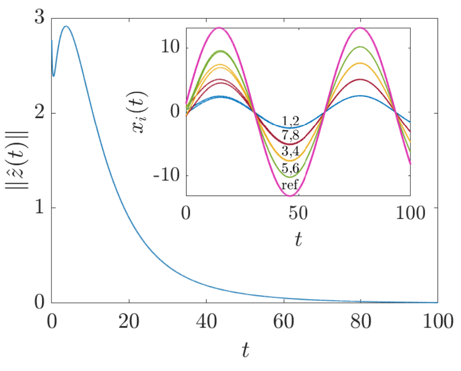
$$\hat{\mathbf{y}} = \hat{\mathbf{p}} \otimes \bar{c} \begin{bmatrix} x_{ref} \\ y_{ref} \end{bmatrix} \tag{4.38}$$

where $\bar{c} = 1.5$, $\hat{\mathbf{p}}$ as in (4.23), and x_{ref} and y_{ref} are the state variables of a reference system having dynamics as in (4.6).

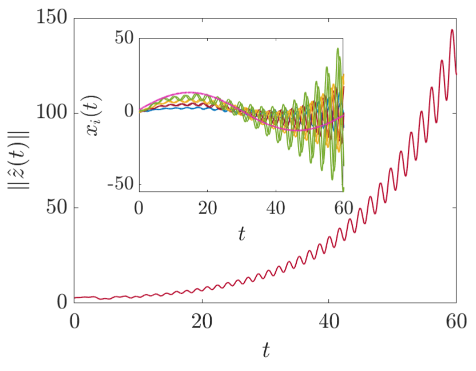
To this purpose, we use the control input u_i from (4.25). In particular, we arbitrarily select $\bar{i} = 1$, such that a non-zero control input is applied only to node 1. The first step is to apply Lemma 4.2.1; as $\hat{\mathbf{p}}$ is the same as in Example 4.2.3, the same weights of Fig. 4.1(b) are obtained. The next step is to apply Theorem 4.3.1 to find the region of k_1 and k_2 where the error $\hat{\mathbf{z}} = \mathbf{y} - \hat{\mathbf{y}}$ asymptotically converges to zero. This yields the region shown in blue in Fig. 4.3(a). Selecting, for instance, $(k_1, k_2) = (0.2, 3.8) \in \hat{S}$ (point $\hat{P}_1$ in Fig. 4.3(a)) leads to the temporal evolution of Fig. 4.3(b) (main panel), where $\|\hat{\mathbf{z}}\|$ vanishes. Correspondingly, the state variables $x_i(t)$ (inset of Fig. 4.3(b)) follow the reference trajectory with amplitude ratios reflecting the division in clusters of the nodes, achieving the desired dynamics specified by Eq. (4.38). On the contrary, if $(k_1, k_2) \in \mathbb{R}^2 \setminus \hat{S}$ (point $\hat{P}_2$ in Fig. 4.3(a)), the norm of the multiconsensus error increases over time (main panel of Fig. 4.3(c)), and the state variables $x_i(t)$ (inset of Fig. 4.3(b)) neither follow the reference nor achieve the desired clustering.



(a)



(b)



(c)

Figure 4.3: Leader multiconsensus for the multiagent system of Example 4.3.2: region of convergence to the desired dynamics (a), temporal evolution of $\|\mathbf{z}(t)\|$ (main panels) and $x_i(t)$ (insets) corresponding to point $\hat{P}_1$ in the region of convergence (b), and to point $\hat{P}_2$, that is outside the region of convergence (c).

Summing up, in this chapter, we have studied how to control multiconsensus in a multi-agent system by exploiting a communication protocol that allows the system states to converge to trajectories parallel to the leading eigenvector of the adjacency matrix representing the interaction graph. Multiconsensus is induced by selecting the weights associated to the links, but without modifying the structure of interactions. This has the advantage that the set of neighbors of each agent is not altered, which is relevant e.g. in situations in which the connectivity is constrained by the adopted communication system. Although we have focused the analysis on agents that have second-order linear dynamics, our results can be generalized to higher-order dynamics. In fact, this step requires the analysis of the eigenvalues of blocks P_j of larger size with respect to Eq. (4.16) which can be accomplished using the generalization of the Routh-Hurwitz criterion to polynomials with complex coefficients [73]. On the one hand, our technique requires a stronger assumption on the connectivity of the underlying graph compared to previous approaches where multiconsensus is achieved in weakly connected directed balanced graphs [114, 109, 79, 55]. On the other hand, it presents the strong advantage that control can be achieved by pinning a single arbitrary node.

We conclude the theoretical part by discussing the more general problem of cluster synchronization. In particular, in the next chapter, we will introduce control techniques aiming to tame cluster synchronization in systems where units interact through networks characterized by the presence of spectral blocks.

Chapter 5

Taming Cluster Synchronization

In this chapter, we discuss how spectral blocks can be leveraged to design controllers able to manipulate cluster synchronization. In particular, we demonstrate how to induce the formation of spectral blocks in networks where such structures do not exist, and how to control the synchronizability of individual clusters, setting also the sequence in which each of them enters or exits the synchronization stability region as the coupling strength increases.

5.1 Spectral block and MSF for cluster synchronization

We start by introducing the concept of spectral blocks and describing how the approach based on the MSF (see Sec. 2.4) can be expanded to study cluster synchronization in the presence of spectral blocks.

Definition 5.1.1 (Spectral block) *A spectral block $\mathcal{S}$ localized at nodes $\{i_1, i_2, \ldots, i_{N'}\}$ is a subset of $(N'-1)$ eigenvectors of the Laplacian matrix* L *having the following properties [7]: i) all $\mathbf{v} \in \mathcal{S}$ are such that $\nu_i = 0$ $\forall i \notin i_1, i_2, \ldots, i'_N$; ii) all $\mathbf{v} \notin \mathcal{S}$ are such that $\nu_i = \nu_j$ $\forall i, j \in i_1, i_2, \ldots, i'_N$.*

As shown in [7], a group of nodes C is associated with a spectral block $\mathcal{S}$ if and only if they are equally connected (i.e., with the same weight) to each other node that is not in C, i.e. $a_{ik} = a_{jk}$, $\forall i, j \in C$ and $\forall k \notin C$.

When a graph is equipped with the spectral blocks $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$, a partition $\pi = \{C_1, C_2, \ldots, C_M, C_{M+1}, \ldots C_{N_\pi}\}$ can be considered, where $C_1, C_2, \ldots, C_M$ are the clusters associated with $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M$, and $C_{M+1}, \ldots, C_{N_\pi}$ are singletons, each containing one of the remaining nodes. This partition can be associated with a quotient graph, denoted by $\mathcal{G}/\pi$, that has vertices $1, 2, \ldots, N_\pi$ and edges with weights $s_{lm} = a_{ij}$ connecting nodes l and m $\forall i \in C_l$, $\forall j \in C_m$. The quotient graph can be represented by its adjacency matrix $\mathrm{S} = \{s_{lm}\}$ with $l, m = 1, \ldots, N_\pi$.

The approach based on the MSF can be expanded to encompass the study of cluster synchronization [7]. Here we consider the case in which the clusters are induced by the presence of spectral blocks. Specifically, given a cluster C formed by M nodes, taking into account the defining structural property of a spectral block, for each node $l \in C$, Eq. (2.2) from Chapter 2 can be rewritten as follows:

$$\dot{\mathbf{x}}_l = \mathbf{f}(\mathbf{x}_l) - \sigma \sum_{m \in C} l_{lm} \mathbf{h}(\mathbf{x}_m) - \sigma \sum_{m \notin C} l_{lm} \mathbf{h}(\mathbf{x}_m) \tag{5.1}$$

where the coupling term is split into two sums, one including extra-cluster interactions and one including intra-cluster interactions. Here, for convenience, we formulated the equations using the coefficients l_{ij} of the graph Laplacian matrix. The dynamics of the cluster synchronous solution, defined by $\mathbf{x}_l = \mathbf{x}_m = \ldots = \mathbf{x}_C$ $\forall l, m \in C$, is given by the following equation:

$$\dot{\mathbf{x}}_C = \mathbf{f}(\mathbf{x}_C) - \sigma \sum_{m \notin C} l_{lm} \mathbf{h}(\mathbf{x}_m) \tag{5.2}$$

By considering the perturbation around the cluster synchronous state $\delta \mathbf{x}_{l,C} = \mathbf{x}_l - \mathbf{x}_C$ and by performing linearization of Eq. (5.1), we obtain:

$$\dot{\delta \mathbf{x}_C} = [\mathrm{I} \otimes D\mathbf{f}|_{\mathbf{x}_C} - \sigma \mathrm{L} \otimes D\mathbf{h}|_{\mathbf{x}_C}]\, \delta \mathbf{x}_C \tag{5.3}$$

with $\delta \mathbf{x}_C = [\delta \mathbf{x}_{1,C}^T, \delta \mathbf{x}_{2,C}^T, \ldots, \delta \mathbf{x}_{N,C}^T]^T$. Eq. (5.3) is very similar to Eq. (2.4), with the difference that, in this case, the Jacobians of $\mathbf{f}$ and $\mathbf{g}$ are evaluated around the cluster synchronous solution $\mathbf{x}_C$, whose dynamics is defined by Eq. (5.2). In this scenario, the trajectories of the cluster synchronous state are affected by the last term of Eq. (5.2), therefore they also depend on the dynamics of the rest of the network. Here, we assume that this term has a negligible effect on the trajectories of the cluster synchronous state, or more precisely on the maximum transverse Lyapunov exponent that results from the use of such trajectories in Eq. (5.3). Under this assumption, we can replace the trajectories followed by the clustered synchronous nodes with the ones of global synchronization in Eq. (5.3). This allows assessing the stability of each cluster's synchronous state by studying the MSF associated with global synchronization.

5.2 Problem statement

Consider a network $\mathcal{G}$ of N identical dynamical systems described by

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + \sigma \sum_{j=1}^{N} a_{ij} \left(\mathbf{h}(\mathbf{x}_j) - \mathbf{h}(\mathbf{x}_i)\right) + \mathbf{u}_i \tag{5.4}$$

Here we focus on the synchronous behavior of the network and, in particular, on the onset of clusters of synchronous nodes. Specifically, we consider a scenario where, given a set of M network clusters (denoted by $C_1, C_2, \ldots, C_M$), one or more of them display synchronous dynamics i.e., $\lim\limits_{t\to+\infty} \|\mathbf{x}_i(t) - \mathbf{x}_j(t)\| = 0 \ \forall i, j \in C_l$ for some $l \in \{1, \ldots, M\}$. Ref. [7] has proved rigorously that such clustered states correspond to the presence of spectral blocks in the structure of $\mathcal{G}$.

Since the nodes associated with spectral blocks receive the same input from the rest of the network, they form a cluster that can synchronize independently on the dynamics of all the other nodes. As discussed in Sec. 5.1, the stability of the synchronous clustered states associated with spectral blocks can be assessed with good approximation by using the MSF. In particular, here we focus on the challenging case of a type III MSF. In this case, denoting by L' the Laplacian matrix of the subgraph $\mathcal{G}'$ associated to the cluster C and calling s the strength through which each node in C is connected with the rest of the graph, the condition for synchronization is given by $\sigma\lambda_i(\mathrm{L}') \in [\nu_1^*, \nu_2^*] \ \forall i = 2, \ldots, N'$, and $\frac{\lambda_{N'}(\mathrm{L}')+s}{\lambda_2(\mathrm{L}')+s} < \frac{\nu_2^*}{\nu_1^*}$. Here ν_1^* and ν_2^* are the two critical values at which the MSF $\lambda_{\max}(\nu)$ crosses the x-axis, namely $\lambda_{\max}(\nu) < 0$ for $\nu \in [\nu_1^*, \nu_2^*]$.

The control input $\mathbf{u}_i$ in Eq. (5.4) is written as

$$\mathbf{u}_i = \sum_{j=1}^{N} w'_{ij} \left(\mathbf{h}(\mathbf{x}_j) - \mathbf{h}(\mathbf{x}_i)\right), \tag{5.5}$$

where w'_{ij} are the weights of the control links added to the pristine network. The spectral blocks's properties can be leveraged to design controllers of the type (5.5) able to shape the synchronous dynamics of the clusters. More specifically, we will concentrate on three different control tasks. The first deals with the case in which $\mathcal{G}$ does not display spectral blocks in the absence of control (i.e., when $\mathbf{u}_i = 0$), and the controllers yield thus the formation of new spectral blocks. The second task is related to the problem of rendering synchronizable the clusters of $\mathcal{G}$ (possibly created through the solution of the first task) for a given, desirable, value of the coupling strength σ. Finally, the third task involves the control of the entire synchronization/desynchronization sequence, namely the order in which the clusters synchronize/desynchronize in class III, as the coupling strength σ increases from zero.

For convenience, in what follows, the weights $w'{}_{ij}$ are normalized by σ (i.e., $w_{ij} = w'_{ij}/\sigma$) so that Eq. (5.4) is rewritten as

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + \sigma \sum_{j=1}^{N} a_{ij} \left(\mathbf{h}(\mathbf{x}_j) - \mathbf{h}(\mathbf{x}_i)\right) + \sigma \sum_{j=1}^{N} w_{ij} \left(\mathbf{h}(\mathbf{x}_j) - \mathbf{h}(\mathbf{x}_i)\right) \tag{5.6}$$

with w_{ij} being selected such that $w_{ij} + a_{ij} \geq 0 \ \forall i, j = 1, \ldots, N$. The adjacency matrix of the controlled network is, therefore, given by $\mathrm{A}' = \mathrm{A} + \mathrm{W} \geq 0$.

5.3 Methods for the solution to the control problems

The control tasks introduced in the previous section are addressed by finding a set of links achieving the specific goal considered, or equivalently, by finding a suitable matrix W. In general, multiple choices of these links can be performed. For this reason, we look for a solution satisfying an optimization problem. In more detail, three different optimization objectives can be considered: minimizing $\|\mathrm{W}\|_2$, preserving the connectedness of the network, and maximizing the sparsity of the solution. We now briefly discuss the three techniques.

Minimizing the L_2 norm of the solution

This method is based on solving the following optimization problem:

$$\min\ ||\hat{\mathbf{y}}||_2 \text{ subject to: } \begin{cases} \mathbf{z} + \hat{\mathbf{y}} \geq \mathbf{0} \\ \mathcal{C}_1 \hat{\mathbf{y}} = \mathbf{q}_1 \\ \mathcal{C}_2 \hat{\mathbf{y}} \geq \mathbf{q}_2 \end{cases} \tag{5.7}$$

where $\hat{\mathbf{y}}$, $\mathbf{z}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ depend on the specific control problem under consideration. This problem can be solved through linear programming and allows to find the solution that requires the least changes in the interaction network, as measured by the L_2 norm of the solution matrix.

Preserving network connectedness

This technique preserves the connectivity of the graph and is based on the following optimization problem:

$$\min\ ||\hat{\mathbf{y}}||_2 \text{ subject to: } \begin{cases} \hat{\mathbf{y}} \geq \mathbf{0} \\ \mathcal{C}_1 \hat{\mathbf{y}} = \mathbf{q}_1 \\ \mathcal{C}_2 \hat{\mathbf{y}} \geq \mathbf{q}_2 \end{cases} \tag{5.8}$$

where, also in this case, $\hat{\mathbf{y}}$, $\mathbf{z}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ are tailored on the specific control problem under consideration. The optimization problem can be solved through linear programming, allowing to find an optimal solution for which the network remains connected.

Maximizing the sparsity of the solution

This approach consists of adding/removing unweighted links to the network and is described by the following optimization problem:

$$\min\ ||\hat{\mathbf{y}}||_1 \text{ subject to: } \begin{cases} \hat{\mathbf{y}} + \mathbf{z} \geq \mathbf{0} \\ \hat{\mathbf{y}} \in \mathbb{Z} \\ \mathcal{C}_1 \hat{\mathbf{y}} = \mathbf{q}_1 \\ \mathcal{C}_2 \hat{\mathbf{y}} \geq \mathbf{q}_2 \end{cases} \tag{5.9}$$

This algorithm allows to find the solution by adding/removing the minimum number of control links and can be solved using integer linear programming.

5.4 Creation of spectral blocks

Consider the multi-agent system in Eq. (5.6) and suppose that the original network of interaction has no spectral blocks. Given a number of M arbitrary groups of nodes, indicated as $C_1, C_2, \ldots, C_M$, we want the nodes in these clusters to form spectral blocks in the controlled multi-agent system, that is, we want to control the interaction network so that it has the spectral blocks $\mathcal{S}_1, \ldots, \mathcal{S}_M$.

To address the problem, we notice that the condition associating nodes i and j to a spectral block C_l (i.e., $a_{ik} = a_{jk}$ $\forall k \notin C_l$) is similar to the condition guaranteeing that two nodes are symmetric, i.e., $a_{ik} = a_{jk} \forall k = 1, \ldots, N$. Therefore, one can follow the approach described in Ref. [31] for inducing symmetries in a graph, and apply it to a fictitious network obtained by neglecting all connections within each cluster. The fictitious network is described by the $N \times N$ adjacency matrix B with entries $b_{ij} = 0$ if $i, j \in C_l$ $\forall l$, and $b_{ij} = a_{ij}$ otherwise. In order to accomplish the control goal, one can ultimately select the entries of W such that:

$$\mathrm{R}_i(\mathrm{B} + \mathrm{W}) - (\mathrm{B} + \mathrm{W})\mathrm{R}_i = 0 \qquad \forall i = 1, 2, \ldots, M \tag{5.10}$$

where R_i is the permutation matrix that maps the nodes of the cluster $C_i = \{i_1, i_2, \ldots, i_{N_i}\}$ into $\{i_2, i_3, \ldots, i_{N_i}, i_1\}$.

By vectorization, Eq. (5.10) can be rewritten as:

$$\mathcal{R}_i \mathrm{vec}(\mathrm{W}) = \mathrm{vec}(\mathrm{R}_i\mathrm{B} - \mathrm{B}\mathrm{R}_i) \tag{5.11}$$

with $\mathcal{R}_i = \mathrm{I}_N \otimes \mathrm{R}_i - \mathrm{R}_i^T \otimes \mathrm{I}_N$. To simultaneously satisfy the condition $\forall i = 1, \ldots, M$, we consider the following equation:

$$\mathcal{R} \mathrm{vec}(\mathrm{W}) = \mathbf{b} \tag{5.12}$$

where $\mathbf{b} = \begin{bmatrix} (\mathrm{vec}(\mathrm{R}_1\mathrm{B} - \mathrm{B}\mathrm{R}_1))^T & \ldots & (\mathrm{vec}(\mathrm{R}_M\mathrm{B} - \mathrm{B}\mathrm{R}_M))^T \end{bmatrix}^T$ and $\mathcal{R} = \begin{bmatrix} \mathcal{R}_1^T & \ldots & \mathcal{R}_M^T \end{bmatrix}^T$. As shown in [31], a solution to (5.12) always exists. Indeed, it is sufficient to ensure that $a_{ik} + w_{ik} = a_{jk} + w_{jk} = 1$ $\forall i, j \in C_l$ and $\forall k \notin C_l$ $\forall l = 1, \ldots, M$. However, this solution is inefficient since it requires adding many control links. To find the optimal solution, one of the three algorithms discussed in Sec. 5.3 can be used, with $\hat{\mathbf{y}} = \mathrm{vec}(\mathrm{W})$, $\mathbf{z} = \mathrm{vec}(\mathrm{A})$, $\mathcal{C}_1 = \mathcal{R}$, $\mathbf{q}_1 = \mathbf{b}$, $\mathcal{C}_2 = 0$, and $\mathbf{q}_2 = \mathbf{0}$.

Example 5.4.1 *As an illustrative example, we consider the unweighted graph of Fig. 5.1(a), with $N = 35$ nodes. The graph has no spectral blocks, and the task is here to induce two spectral blocks (S_1 and S_2) formed by the (arbitrarily chosen) nodes marked as blue squares and orange triangles in Fig. 5.1(a), respectively. We proceed with the optimization problem aiming at maximizing the sparsity of the solution. This either adds or removes links from the pristine network, i.e., $-a_{ij} \leq w_{ij} \leq 1 - a_{ij}$, $w_{ij} \in \mathbb{Z}$. The resulting controlled network is depicted in Fig. 5.1(b), where the nodes of S_1 and S_2 are connected with a bulk of other nodes with strengths $s_1 + w_1 = 1$ and $s_2 + w_2 = 3$, respectively.*

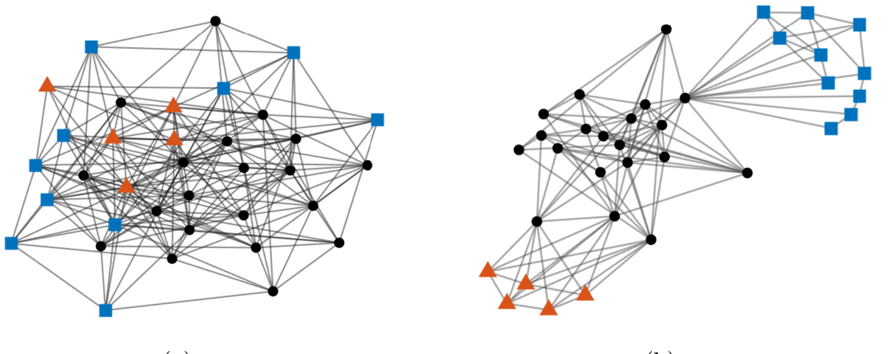

(a) (b)

Figure 5.1: Creating spectral blocks. (a) A graph with $N = 35$ nodes and no spectral blocks. The nodes associated with the spectral blocks S_1 and S_2 to be induced by the control are marked by blue squares and orange triangles, respectively. (b) The controlled network. The control is performed by adding/removing links, and leads to the two desired groups of nodes connected to a bulk with strengths $s_1 + w_1 = 1$ and $s_2 + w_2 = 3$, respectively.

5.5 Controlling cluster synchronizability

Consider a graph with M spectral blocks $(S_1, S_2, \ldots, S_M)$ associated to the clusters $(C_1, C_2, \ldots, C_M)$, and assume that $M' \leq M$ of such clusters, namely $C_1, C_2, \ldots, C_{M'}$, cannot synchronize at any value of σ, as they do not satisfy the eigenvalue ratio condition. In other terms, taking into account the properties of synchronization of the spectral blocks, the $M' \leq M$ subgraphs $\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_{M'}$ associated with $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{M'}$ are such that:

$$\frac{\lambda_{N_l}(\mathcal{L}_l) + s_l}{\lambda_2(\mathcal{L}_l) + s_l} \geq \frac{\nu_2^*}{\nu_1^*} \qquad \forall l = 1, \ldots, M' \tag{5.13}$$

where $N_l = |C_l|$, $\mathcal{L}_l$ is the Laplacian matrix of $\mathcal{G}_l$ and s_l is the strength through which each node of the cluster C_l associated with $\mathcal{S}_l$ is connected to the rest of the graph, i.e., $s_l = \sum\limits_{j \notin C_l} a_{ij}$ for $i \in C_l$.

The goal is now rendering such clusters synchronizable. For this purpose, control links are added to the network in a way that *i)* the spectral block condition on $S_1, S_2, \ldots, S_M$ is preserved, and *ii)* the criterion on the eigenvalue ratio becomes fulfilled. In practice, the weights of the control links have to be selected such that $\mathrm{W} = \mathrm{W}^T$, $w_{ik} = w_{jk}\ \forall i, j \in C_l, \forall k \notin C_l, \forall l = 1, \ldots, M$, and:

$$\frac{\lambda_{N_l}(\mathrm{L}_l) + s_l + w_l}{\lambda_2(\mathrm{L}_l) + s_l + w_l} < \frac{\nu_2^*}{\nu_1^*} \qquad \forall l = 1, \ldots, M, \tag{5.14}$$

with w_l being defined as $w_l = \sum\limits_{j \notin C_l} w_{ij}$ for $i \in C_l$. By rearranging the terms of (5.14), we obtain:

$$w_l + s_l - \left(\frac{\nu_2^*}{\nu_1^*} - 1\right)^{-1} \left(\lambda_{N_l}(\mathrm{L}_l) - \frac{\nu_2^*}{\nu_1^*}\lambda_2(\mathrm{L}_l)\right) > 0, \qquad \forall l = 1, \ldots, M' \tag{5.15}$$

Next we consider the quotient graph $\mathcal{G}/\pi$ and its associated adjacency matrix S, and we introduce the matrix $\mathrm{X} \in \mathbb{R}^{N_\pi \times N_\pi}$ whose elements are $x_{lm} = w_{ij}\ \forall i \in C_l$, $\forall j \in C_m$, $l \neq m$. Let $\mathrm{O}_{K,\Omega} = \begin{bmatrix} \mathrm{I}_K & 0_{K,\Omega-K} \end{bmatrix}$ be a $K \times \Omega$ matrix such that the product $\mathrm{O}_{K,\Omega}\mathrm{Z}$ returns the first K rows of Z. Then, Eqs. (5.15) can be written in compact, matrix-vector form as follows:

$$\begin{cases} \mathrm{O}_{M',N_\pi}\left(\mathrm{X} + \mathrm{S}\right)\gamma - \left(\frac{\nu_2^*}{\nu_1^*} - 1\right)^{-1} \mathrm{O}_{M',M}\left(\mathbf{\Lambda_N} - \frac{\nu_2^*}{\nu_1^*}\mathbf{\Lambda_2}\right) > \mathbf{0} \\ \mathrm{X} - \mathrm{X}^T = 0 \\ \mathrm{diag}(\mathrm{X}) = 0 \end{cases} \tag{5.16}$$

where $\gamma = [N_1, N_2, \ldots, N_{N_\pi}]^T$, $\mathbf{\Lambda_2} = [\lambda_2(\mathrm{L}_1), \lambda_2(\mathrm{L}_2), \ldots, \lambda_2(\mathrm{L}_M)]^T$ and $\mathbf{\Lambda_N} = [\lambda_{N_1}(\mathrm{L}_1), \lambda_{N_2}(\mathrm{L}_2), \ldots, \lambda_{N_M}(\mathrm{L}_M)]^T$.

Let $\mathbf{e_l}$ be the l-th canonical vector. By vectorization, Eq. (5.16) is rewritten as:

$$\begin{cases} \mathrm{F}\mathbf{x} > \mathbf{g} \\ \mathrm{E}\mathbf{x} = \mathbf{0} \end{cases} \tag{5.17}$$

where $\mathbf{x} = \mathrm{vec}\,(\mathrm{X})$, $\mathbf{g} = -\mathrm{O}_{M',N_\pi}\mathrm{S}\gamma + \left(\frac{\nu_2^*}{\nu_1^*} - 1\right)^{-1}\mathrm{O}_{M',M}\left(\mathbf{\Lambda_N} - \frac{\nu_2^*}{\nu_1^*}\mathbf{\Lambda_2}\right)$,

$\mathrm{F} = \gamma^T \otimes \mathrm{O}_{M',N_\pi}$, and $\mathrm{E} = \begin{bmatrix} \mathrm{E}_1 \\ \\ \mathrm{E}_2 \end{bmatrix}$ is a block matrix with:

$$\mathrm{E}_1 = \mathrm{I}_{N_\pi} - \begin{bmatrix} \mathrm{I}_{N_\pi} \otimes \mathbf{e_1} \\ \mathrm{I}_{N_\pi} \otimes \mathbf{e_2} \\ \vdots \\ \mathrm{I}_{N_\pi} \otimes \mathbf{e_{N_\pi}} \end{bmatrix}, \qquad \mathrm{E}_2 = \begin{bmatrix} \mathbf{e_1} & & & \\ & \mathbf{e_2} & & \\ & & \ddots & \\ & & & \mathbf{e_{N_\pi}} \end{bmatrix} \tag{5.18}$$

where $\mathbf{e_l}$ is the l-th canonical vector, the matrix E_1 is used to set $\mathrm{X} = \mathrm{X}^T$, whereas E_2 to set $\mathrm{diag}(\mathrm{X}) = 0$.

We solve (5.17) for the unknowns x_{lm} and then, once we have obtained them, we compute the entries of W as follows:

$$w_{ij} = x_{lm} \qquad \forall i \in C_l, \forall j \in C_m, \forall l, m = 1, \ldots, N_\pi \tag{5.19}$$

Notice that a solution of Eq. (5.17) always exists since

$$\lim_{w_l \to \infty} \frac{\lambda_{N_l}(\mathrm{L}_l) + s_l + w_l}{\lambda_2(\mathrm{L}_l) + s_l + w_l} = 1 < \frac{\nu_2^*}{\nu_1^*} \qquad \forall l = 1, \ldots, M \tag{5.20}$$

To determine the optimal solution, one of the three algorithms described in Sec. 5.3 can be used, where $\hat{\mathbf{y}} = \mathrm{vec}(\mathrm{X})$, $\mathbf{z} = \mathrm{vec}(\mathrm{S})$, $\mathcal{C}_1 = E$, $\mathbf{q}_1 = \mathbf{0}$, $\mathcal{C}_2 = \mathrm{F}$, $\mathbf{q}_2 = \mathbf{g} + \epsilon\mathbf{1}$, where $\epsilon > 0$ is an arbitrary small number. It is worth noticing that $\mathbf{q}_2$ is not equal to $\mathbf{g}$ because the inequality in Eq. (5.17) is strict, unlike those in the optimization problems in Sec. 5.3.

Example 5.5.1 *As an example, we consider the graph of Fig. 5.1(b) i.e., the result of the first control task, displaying two spectral blocks (S_1, S_2) with associated clusters C_1, C_2. Without lack of generality, we consider the node dynamics regulated by the Lorenz system [53]. Therefore Eqs.* (5.6) *read:*

$$\begin{cases} \dot{x}_{i,1} = a\left(x_{i,2} - x_{i,1}\right) + \sigma \sum\limits_{j=1}^{N} \left(a_{ij} + w_{ij}\right)\left(x_{j,2} - x_{i,2}\right), \\ \dot{x}_{i,2} = x_{i,1}\left(b - x_{i,3}\right) - x_{i,2}, \\ \dot{x}_{i,3} = x_{i,1}x_{i,2} - cx_{i,3}, \end{cases} \tag{5.21}$$

where the parameters are $a = 10$, $b = 28$, *and* $c = 2$, *so as the uncoupled dynamics is chaotic. The system has a type III MSF with* $\nu_1^* = 4.173$ *and* $\nu_2^* = 22.535$ *[40]. The smallest and the largest non-zero eigenvalues of* L_1 *are* $\lambda_2(\mathrm{L}_1) = 0.21$ *and* $\lambda_{10}(\mathrm{L}_1) = 5.93$, *whereas those of* L_2 *are* $\lambda_2(\mathrm{L}_2) = 1.38$ *and* $\lambda_5(\mathrm{L}_2) = 4.62$ *(see SM for all details on* L_1 *and* L_2*). Furthermore, the clusters* C_1 *and* C_2 *are connected to the rest of the graph with strengths* $s_1 = 1$ *and* $s_2 = 3$, *respectively. Since* $\frac{\lambda_{10}(\mathrm{L}_1)+s_1}{\lambda_2(\mathrm{L}_1)+s_1} > \frac{\nu_2^*}{\nu_1^*}$, *the nodes of* C_1 *cannot synchronize at any value of* σ. *For instance, a large synchronization error* $\delta = \frac{1}{N_1}\left(\sum_{i\in C_1} ||\mathbf{x}_i - \bar{\mathbf{x}}_1||^2\right)^{\frac{1}{2}}$ *(with* $N_1 = |C_1|$ *and* $\bar{\mathbf{x}}_1 = \frac{1}{N_1}\sum_{j\in C_1} \mathbf{x}_j$*) is obtained for* $\sigma = 2$, *as shown in Fig. 5.2(b). Also in this case, we adopt the optimization problem that maximizes the sparsity of the solution. Fig. 5.2(a) shows the controlled network, in which* C_1 *is connected to the bulk with a strength* $s_1 + w_1 = 2$. *Since* $\frac{\lambda_{10}(\mathrm{L}_1)+s_1+w_1}{\lambda_2(\mathrm{L}_1)+s_1+w_1} < \frac{\nu_2^*}{\nu_1^*}$, C_1 *now satisfies the eigenvalue ratio condition and can therefore synchronize for* $1.89 = \frac{\nu_1^*}{\lambda_2(\mathrm{L}_1)+s_1+w_1} < \sigma < \frac{\nu_2^*}{\lambda_{10}(\mathrm{L}_1)+s_1+w_1} = 2.84$. *Consequently,* C_1 *reaches synchronization for* $\sigma = 2$, *as confirmed from the time evolution of the error* $\delta(t)$ *shown in Fig. 5.2(c).*

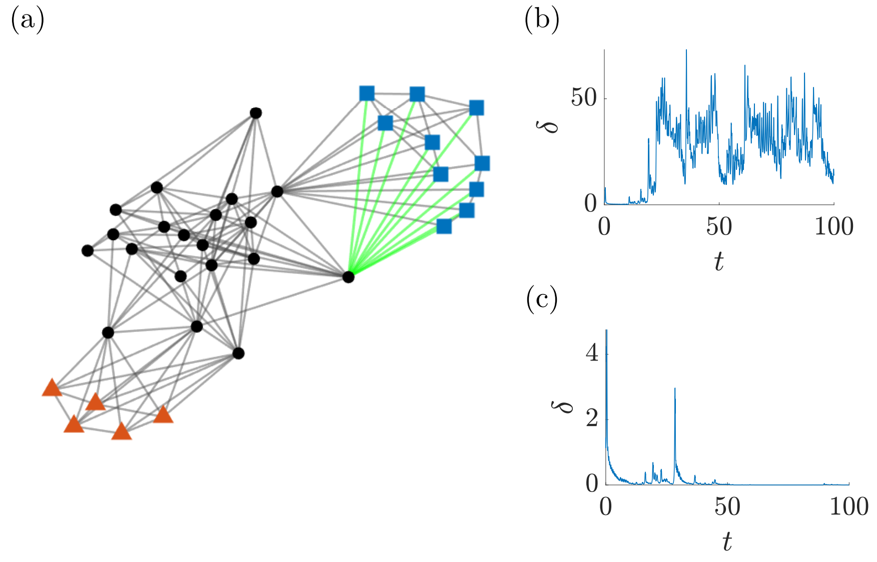


Figure 5.2: Taming cluster synchronizability. (a) Controlled network with two synchronizable spectral blocks S_1 and S_2. The nodes where the spectral blocks are localized are now connected with the bulk with strengths $s_1+w_1 = 2$ and $s_2 + w_2 = 3$, respectively. (b) Time evolution of the synchronization error $\delta(t)$ within the cluster C_1, for $\sigma = 2$ in the absence of control (i.e., using the network of Fig. 5.1(b)). (c) Time evolution of $\delta(t)$ within the cluster C_1, for $\sigma = 2$ when control is applied (i.e., using the network of panel (a)).

5.6 Shaping the synchronization/desynchronization sequence

Consider a graph equipped with M spectral blocks $(\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M)$ forming the clusters $C_1, C_2, \ldots, C_M$, and let $s_1, s_2, \ldots, s_M$ and $\mathrm{L}_1, \mathrm{L}_2, \ldots, \mathrm{L}_M$ be, respectively, the strengths through which the clusters are connected with the rest of the graph and the Laplacian matrices of the subgraphs associated with the clusters. One can define the synchronization sequence $\mathcal{I}$ as the set that contains the cluster indices j, ordered in decreasing order with respect to the sum of the largest non-zero eigenvalue of L_j and the corresponding s_j. In other words, $\mathcal{I} = \{i_1, i_2, \ldots, i_M\}$ if $\lambda_2(\mathrm{L}_{i_1}) + s_{i_1} \geq \lambda_2(\mathrm{L}_{i_2}) + s_{i_2} \geq \ldots \geq \lambda_2(\mathrm{L}_{i_M}) + s_{i_M}$. Similarly, one may define the desynchronization sequence

$\mathcal{D}$ as $\mathcal{D} = \{i_1, i_2, \dots, i_M\}$ if $\lambda_{N_{i_1}}(\mathrm{L}_{i_1}) + s_{i_1} \geq \lambda_{N_{i_2}}(\mathrm{L}_{i_2}) + s_{i_2} \geq \dots \geq \lambda_{N_{i_M}}(\mathrm{L}_{i_M}) + s_{i_M}$, with $N_i = |C_i|$. Clusters, indeed, will enter (exit) one after the other the stability region following the order specified in the sequence $\mathcal{I}$ ($\mathcal{D}$).

The goal of the control is to induce synchronization and desynchronization sequences arbitrarily chosen, say $\mathcal{I}' = \mathcal{D}' = \{1, 2, \dots, M\}$. To that purpose, the weights of the control links have to be selected such that: *i)* the clusters $C_1, C_2, \dots, C_M$ remain associated with the spectral blocks $S_1, S_2, \dots, S_M$, *ii)* they satisfy the eigenvalue ratio condition, *iii)* they synchronize and desynchronize according to the desired sequences $\mathcal{I}'$ and $\mathcal{D}'$. In practice, the entries of W have to be selected such that Eq. (5.14) holds, $\mathrm{W} = \mathrm{W}^T$, $w_{ik} = w_{jk}$ $\forall i, j \in C_l, \forall k \notin C_l, \forall l = 1, \dots, M$, and:

$$\begin{cases} \dfrac{\nu_1^*}{\lambda_2(\mathrm{L}_l) + s_l + w_l} < \dfrac{\nu_1^*}{\lambda_2(\mathrm{L}_{l+1}) + s_{l+1} + w_{l+1}} \\ \dfrac{\nu_2^*}{\lambda_{N_l}(\mathrm{L}_l) + s_l + w_l} < \dfrac{\nu_2^*}{\lambda_{N_{l+1}}(\mathrm{L}_{l+1}) + s_{l+1} + w_{l+1}} \end{cases} \tag{5.22}$$

$\forall l = 1, \dots, M-1$. By rearranging the terms in Eq. (5.22), we obtain that:

$$\begin{cases} s_l + w_l - s_{l+1} - w_{l+1} + \lambda_2(\mathrm{L}_l) - \lambda_2(\mathrm{L}_{l+1}) > 0 & \forall l = 1, \dots, M-1 \\ s_l + w_l - s_{l+1} - w_{l+1} + \lambda_{N_l}(\mathrm{L}_l) - \lambda_{N_{l+1}}(\mathrm{L}_{l+1}) > 0 & \forall l = 1, \dots, M-1 \end{cases} \tag{5.23}$$

We now proceed as we did for the control problem illustrated in Sec. 5.5, rewriting Eqs. (5.23) in compact form via the quotient graph $\mathcal{G}/\pi$, its adjacency matrix S and the matrix X, as follows:

$$\begin{cases} (\mathbf{1}_2 \otimes \mathrm{Q})\, \mathrm{O}_{M,N_\pi}\, (\mathrm{S} + \mathrm{X})\, \gamma + (\mathrm{I}_2 \otimes \mathrm{Q}) \begin{bmatrix} \mathbf{\Lambda_2} \\ \mathbf{\Lambda_N} \end{bmatrix} > \mathbf{0} \\ \mathrm{X} - \mathrm{X}^T = 0 \\ \mathrm{diag}(\mathrm{X}) = 0 \end{cases} \tag{5.24}$$

where

$$\mathrm{Q} = \begin{bmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{bmatrix} = \begin{bmatrix} \mathrm{I}_{M-1} & \mathbf{0}_{M-1} \end{bmatrix} - \begin{bmatrix} \mathbf{0}_{M-1} & \mathrm{I}_{M-1} \end{bmatrix} \tag{5.25}$$

We note that the matrix $\mathrm{Q} \in \mathbb{R}^{(M-1)\times M}$ has the following property. Given a $M \times K$ matrix Z, then the $i-$th row of the product QZ is $(\mathrm{QZ})_i = \mathrm{Z}_i - \mathrm{Z}_{i+1}$, where Z_i denotes the $i-$th row of Z.

By performing the vectorization of Eq. (5.24) and by considering Eq. (5.17), which is the vectorization of the condition *ii*), we obtain the following:

$$\begin{cases} \mathrm{F}\mathbf{x} > \mathbf{g} \\ \mathrm{P}\mathbf{x} > \mathbf{c} \\ \mathrm{E}\mathbf{x} = \mathbf{0} \end{cases} \tag{5.26}$$

where $\mathrm{P} = \left(\gamma^T \otimes \mathbf{1}_2 \otimes \mathrm{QO}_{M,N_\pi}\right)$,

and $\mathbf{c} = -\left(\mathbf{1}_2 \otimes \mathrm{Q}\right)\mathrm{O}_{M,N_\pi}\mathrm{S}\gamma - (\mathrm{I}_2 \otimes \mathrm{Q}) \begin{bmatrix} \mathbf{\Lambda_2} \\ \mathbf{\Lambda_N} \end{bmatrix}$.

Once the entries of X are obtained, we compute the coefficients w_{ij} using Eq. (5.19).

Here we do not provide a formal proof of the existence of a solution $\mathbf{x}$ of Eq. (5.26), such that $\mathbf{x} + \mathrm{vec}(\mathrm{S}) \geq 0$, but only sketch the arguments behind it. Given the vector $\hat{\mathbf{c}} = \mathbf{c} + (1_2 \otimes \mathrm{Q})\mathrm{O}_{M,N_\pi}\mathrm{S}\gamma$, the second inequality of Eq. (5.26) is equivalent to $\mathrm{P}\left(\mathbf{x} + \mathrm{vec}(\mathrm{S})\right) > \hat{\mathbf{c}}$. Now, let us consider the matrix $\mathrm{G} = \mathrm{PT}$ where T is defined as

$$\mathrm{T} = \mathrm{I}_{N_\pi} + \begin{bmatrix} \mathrm{I}_{N_\pi} \otimes \mathbf{e_1} \\ \mathrm{I}_{N_\pi} \otimes \mathbf{e_2} \\ \vdots \\ \mathrm{I}_{N_\pi} \otimes \mathbf{e_{N_\pi}} \end{bmatrix} \tag{5.27}$$

and let us then remove from G the columns that correspond to the elements x_{lm} with $l \geq m$, thus obtaining another matrix that we indicate as $\hat{\mathrm{G}}$. By construction, the matrix $\hat{\mathrm{G}}$ is such that there is no $\mathbf{y} \geq \mathbf{0}$ that satisfies $-\hat{\mathrm{G}}^T\mathbf{y} \geq 0$ and $-\hat{\mathbf{c}}^T\mathbf{y} > 0$. Therefore, for Lemma 2.3.8, Eq. (5.26) always has a solution $\mathbf{x} \geq -\mathrm{vec}(\mathrm{S})$. Finally, we can replace $\hat{\mathbf{c}}$ with $\hat{\mathbf{c}} + \epsilon\mathbf{1}$ to obtain a solution strictly positive.

Analogously to the control problems discussed above, the solution of Eq. (5.26) can be obtained through one of the three approaches shown in Sec. 5.3, with $\hat{\mathbf{y}} = \mathrm{vec}(\mathrm{X})$, $\mathbf{z} = \mathrm{vec}(\mathrm{S})$, $\mathcal{C}_1 = \mathrm{E}$, $\mathbf{q}_1 = \mathbf{0}$, $\mathcal{C}_2 = \begin{bmatrix} \mathrm{F} \\ \mathrm{P} \end{bmatrix}$, $\mathbf{q}_2 = \begin{bmatrix} \mathbf{g} \\ \mathbf{c} \end{bmatrix} + \epsilon\mathbf{1}$.

Let us now discuss an example where the optimal solution is obtained using the method based on minimizing the L_2 norm, as in Sec. 5.3.

Example 5.6.1 *We consider the weighted interaction network shown in Fig. 5.3(a), where the width of each link is proportional to its weight. This network is composed of three spectral blocks, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$, localized at the nodes of the clusters $C_1 = \{1, \ldots, 5\}$, $C_2 = \{6, \ldots, 10\}$, and $C_3 = \{11, \ldots, 15\}$. To show how general the applicability of our method is, this time we take for each node the dynamics of the Rössler oscillator [85]. The network evolution is therefore governed by the following equations:*

$$\begin{cases} \dot{x}_{i,1} = -x_{i,2} - x_{i,3} + d \sum_{j=1}^{N} (a_{ij} + w_{ij})(x_{j,1} - x_{i,1}), \\ \dot{x}_{i,2} = x_{i,1} + a x_{i,2}, \\ \dot{x}_{i,3} = b + x_{i,3}(x_{i,1} - c), \end{cases} \tag{5.28}$$

with $a = 0.2$, $b = 0.2$ and $c = 7$, such that the uncoupled dynamics is chaotic. System (5.28) has a type III MSF with $\nu_1^* = 0.186$ and $\nu_2^* = 4.614$ [40]. The adjacency matrix associated with the quotient graph (in the absence of control) is:

$$
\mathrm{A} = \begin{bmatrix}
0 & 0.5 & 0 & 0 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.5 & 0 & 0.5 & 0 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0 & 0.5 & 0 & 0.5 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0 & 0 & 0.5 & 0 & 0.5 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0 & 0 & 0 & 0.5 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0.5 & 0.5 & 0.5 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0 & 0 & 0 & 0.5 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0 & 0 & 0.5 & 0.5 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0 & 0.5 & 0 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0.5 & 0.5 & 0 & 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0.5 & 0 & 0.5 & 0.5 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0 & 0.5 & 0.5 & 0.5 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0 & 0.5 & 0 & 0.5 & 0.5 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0.5 & 0.5 & 0 & 0.5 \\
0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.5 & 0.5 & 0.5 & 0.5 & 0
\end{bmatrix}
$$

The strengths with which the clusters C_1, C_2, *and* C_3 *are connected to the rest of the graph are* $s_1 = s_2 = s_3 = 1$, *since* $a_{ij} = 0.1\ \forall i \in C_l, j \in C_m, l \neq m$. *In addition, the smallest and the largest non-zero eigenvalues of* L_1, L_2, L_3 *are* $\lambda_2(\mathrm{L}_1) = 0.19$, $\lambda_2(\mathrm{L}_2) = 0.69$, $\lambda_2(\mathrm{L}_3) = 1.51$, *and* $\lambda_5(\mathrm{L}_1) = 1.81$, $\lambda_5(\mathrm{L}_2) = 2.31$, $\lambda_5(\mathrm{L}_3) = 2.5$. *Therefore, since* $\lambda_2(\mathrm{L}_1) + s_1 < \lambda_2(\mathrm{L}_2) + s_2 < \lambda_2(\mathrm{L}_3) + s_3$ *and* $\lambda_5(\mathrm{L}_1) + s_1 < \lambda_5(\mathrm{L}_2) + s_2 < \lambda_5(\mathrm{L}_3) + s_3$, *in the absence of control, the synchronization and desynchronization sequences are* $\mathcal{I} = \mathcal{D} = \{3, 2, 1\}$. *This is confirmed by the numerical simulations of the multi-agent system in Eqs.* (9.8) *illustrated in Fig. 5.3(c). The time evolution of the system variables has been computed for a period of time equal to* $5T$, *with* $T = 10$. *After discarding a transient of* $4T$, *we have calculated the average value (on a window of time* T*) of the cluster synchronization error* $\langle \delta_h \rangle_T = \langle \frac{1}{N'} \left(\sum_{i \in C_h} ||\mathbf{x}_i - \bar{\mathbf{x}}_\mathbf{h}||^2 \right)^{\frac{1}{2}} \rangle_T$, *with* $h = \{1, 2, 3\}$, *for each of the three clusters of the network, namely* C_1 *(blue curve),* C_2 *(orange curve) and* C_3 *(green curve), as function of the coupling strength* σ. *Fig. 5.3(c) also shows* *the critical values predicted by the MSF approach marked as blue, orange or green triangles for the three clusters, C_1, C_2, and C_3.*

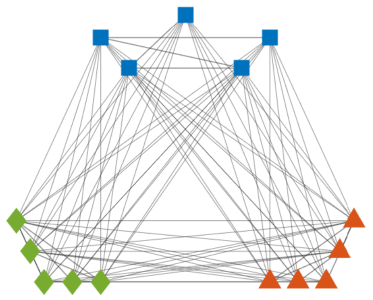
Next, we apply the control of the multi-agent system, using minimization of $||\mathrm{W}||_2$ *to find a solution of Eq.* (5.26) *with desired synchronization/desynchronization sequence* $\mathcal{I}' = \mathcal{D}' = \{1, 2, 3\}$*.*

This leads to the following matrix X*:*

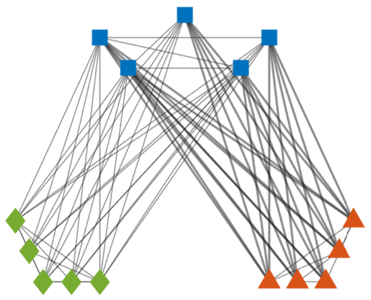
$$\mathrm{X} = \begin{bmatrix} 0 & 0.96 & 0.4 \\ 0.96 & 0 & -0.1 \\ 0.4 & -0.1 & 0 \end{bmatrix}$$

The resulting controlled network is shown in Fig. 5.3(b) with the clusters C_1, C_2 and C_3 being connected to the bulk with strengths $s_1 + w_1 = 7.8$, $s_2 + w_2 = 5.3$ and $s_3 = 2.5$, respectively. Consequently, we have that $\lambda_2(\mathrm{L}_1)+s_1+w_1 > \lambda_2(\mathrm{L}_2)+s_2+w_2 > \lambda_2(\mathrm{L}_3)+s_3+w_3$ and $\lambda_5(\mathrm{L}_2)+s_2+w_2 < \lambda_5(\mathrm{L}_1)+s_1+w_1 < \lambda_5(\mathrm{L}_3)+s_3+w_3$, yielding $\mathcal{I}' = \mathcal{D}' = \{1, 2, 3\}$. The curves of the synchronization error $\langle\delta_h\rangle_T$ for the three clusters C_1, C_2 and C_3 ($h = \{1, 2, 3\}$) as function of the coupling strength σ, shown in Fig. 5.3(d), demonstrate that the controlled multi-agent system displays the predicted synchronization/desynchronization sequence.

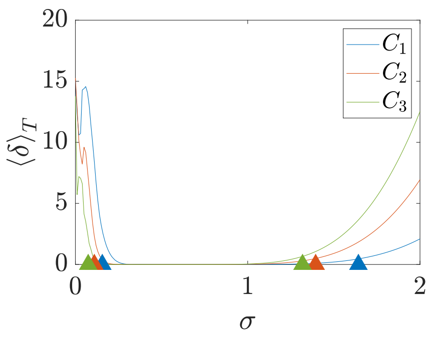
With this chapter, we conclude the theoretical part. Now we move to the experimental one, which deals with the robotic implementations of the theoretical models discussed in this first part.



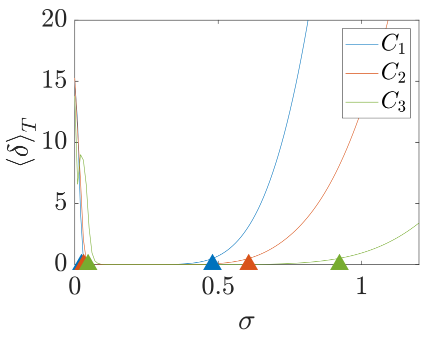
(a)



(b)



(c)



(d)

Figure 5.3: Shaping the synchronization/desynchronization sequence. (a) Pristine weighted network, where the thickness of each edge is proportional to the edge weight. The network is formed by three spectral blocks $\mathcal{S}_1, \mathcal{S}_\in, \mathcal{S}_3$ localized at the nodes of the clusters C_1 (blue squares), C_2 (orange triangles) and C_3 (green diamonds). These clusters are connected with the rest of the graph with strength $s_1 = s_2 = s_3 = 1$. (b) Controlled network, where the clusters are connected with the rest of the graph with strength $s_1 + w_1 = 7.8$, $s_2 + w_2 = 5.3$, and $s_3 + w_3 = 2.5$, respectively. (c,d) $\langle\delta_h\rangle_T$ vs. σ for the uncontrolled (panel c) and controlled (panel d) network for the three clusters C_1 (blue line), C_2 (orange line) and C_3 (green line). The triangles mark the transition values for synchronization stability predicted by the MSF approach. Controlling the multi-agent system makes it possible to change the synchronization and desynchronization sequence from $\mathcal{I} = \mathcal{D} = \{3, 2, 1\}$ to $\mathcal{I}' = \mathcal{D}' = \{1, 2, 3\}$.

Part II

Robotic implementation

Chapter 6

Robotic platforms: Elisa-3 and e-puck2

We start the experimental part of this thesis by describing the two robotic platforms used to perform the experiments: the Elisa-3 and the e-puck2 robots. Both of them are small-sized robots designed by the company GC-tronic[1] and intended for education and research purposes. Since they are equipped with a local communication system, these robots are well-suited for experiments that involve distributed models.

6.1 Elisa-3

The Elisa-3 robots[2] are small-sized robots with a circular shape, diameter $d = 5$ cm, height $h = 3$ cm, and weight $w = 39$ g. These robots are differential drive platforms with two wheels, each driven by a DC motor with a 25:1 reduction gear and a maximum speed equal to 60 cm/s. The main features of the robot are shown in Fig. 6.1. Each robot is equipped with an accelerometer for detecting its relative position via odometry, an IR receiver for remote control, and three IR emitters, one of which is used for tracking by an IR camera. Furthermore, each robot has eight IR sensors, uniformly distributed along the external circumference of the chassis and able to detect obstacles at a distance up to 5 cm, and four ground sensors,

[1] `https://www.gctronic.com/doc/index.php?title=GCtronic_Wiki`

[2] `https://www.gctronic.com/doc/index.php/Elisa-3`

located on the front side of the robot and used for cliff avoidance. The IR sensors can also be used for local communication with other robots. As the maximum distance for this communication system is $d_M = 5$ cm, each robot can communicate with any other robot that is at a center-to-center distance between 5 cm and 10 cm. This system has no transmission/reception queue and provides a throughput of approximately 1 byte/s, allowing robots to transmit packets of one byte each.

The robots can also communicate with the computer through a 2.4 GHz radio module that allows them to send/receive data to/from the computer at a distance of up to 10 m. The radio link bandwidth is $B_w = 1$ kHz, and the throughput depends on the number of robots, as the computer sends and receives packets to and from 4 robots simultaneously. As a result, the communication period is given by:

$$t_d = \left\lceil \frac{N}{4} \right\rceil \frac{1}{B_w} \tag{6.1}$$

where N is the number of robots and $\lceil \cdot \rceil$ denotes the ceiling function, which rounds the argument up to the nearest integer.

Each robot is controlled by an onboard 8-bit Atmel ATmega2460 microcontroller that handles motor control, sensor data acquisition, and communication with other robots and the PC.

The experimental setup, shown in Fig. 8.1 includes $N = 6$ robots and an arena of size $L_{y_1} = 80$ cm and $L_{y_2} = 60$ cm. An IR and an RGB camera are positioned above the arena to capture the whole area where the robots move. Specifically, the IR camera tracks and localizes each robot by detecting the IR emitter mounted on top. The camera data are acquired, processed, and recorded by the computer.

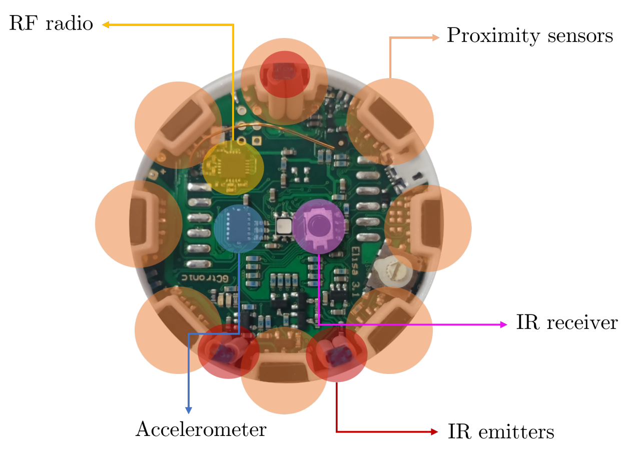


Figure 6.1: Elisa-3 robot including 8 proximity sensors to detect obstacles and locally communicate with other units, 3 IR emitters to be detected by the IR camera, an accelerometer, an IR receiver, and an RF module to communicate with the computer.

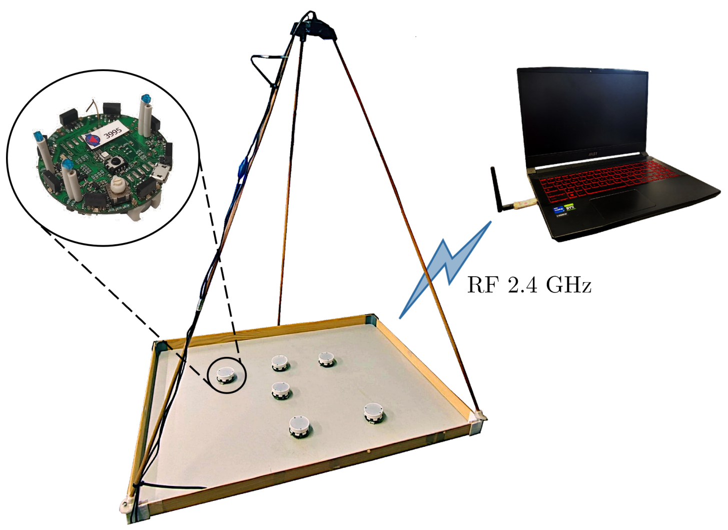


Figure 6.2: Experimental setup including 6 Elisa-3 robots in an arena of dimensions 80×60 cm. A computer receives data from the robots through a 2.4 GHz radio link and both RGB and IR cameras mounted over the arena through USB connection.

6.2 E-puck2

In this section, we describe the e-puck2 robot[3] and two of its extensions: the range & bearing board[4] for interacting with other robots, and the Pi-puck[5], which enables the mounting of a Raspberry Pi Zero single-board computer on the robot. Besides these extensions, the system also offers additional modules, including the ground module with 4 ground sensors, the cliff module with 2 cliff sensors, and the Omnivision Module V3, which features a 5 Mpixel color camera providing a 360° view around the robot, suitable for basic image processing tasks such as blob detection, color recognition, and light detection. However, since these additional extensions will not be used in our experiments, they will not be further explored. Fig. 6.3 shows the robot equipped with the extensions mentioned above.

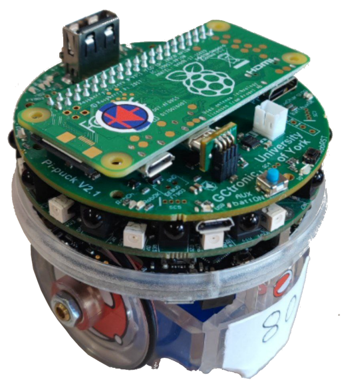


Figure 6.3: E-puck2 and two of its extensions mounted on it: range & bearing board and Pi-puck.

6.2.1 E-puck2 main board

The e-puck2 robot is a swarm robot characterized by a diameter $d = 7$ cm, a height of $h = 5.5$ cm, and a weight equal to $w = 150$ g. It moves using a differential drive system with two wheels powered by stepper motors,

[3] `https://www.gctronic.com/doc/index.php?title=GCtronic_Wiki`

[4] `https://www.gctronic.com/doc/index.php?title=Others_Extensions#Range_and_bearing`

[5] `https://www.gctronic.com/doc/index.php?title=Pi-puck`

allowing control of motion by setting either the speed or the motor steps. Each robot is equipped with a microcontroller (STM32F4 at 168MHz) that manages its sensors and actuators, including eight proximity sensors for obstacle avoidance, a time of flight (TOF) sensor, a speaker, an IR receiver to be remotely controlled, a CMOS camera for onboard image processing, an IMU (3D accelerometer + 3D gyro + 3D magnetometer) for detecting its relative position and orientation, and the stepper motors for motion control allowing a maximum speed of 15.4 cm/s. The proximity sensors enable also local communication with robots. In addition, the robot is equipped with a radio module, that handles the wireless communication (WiFi, BLE, BT) with the computer and controls the RGB LEDs that are connected with this module due to the limited number of pins in the microcontroller board. The main components of the e-puck2 main board are highlighted in Fig. 6.4.

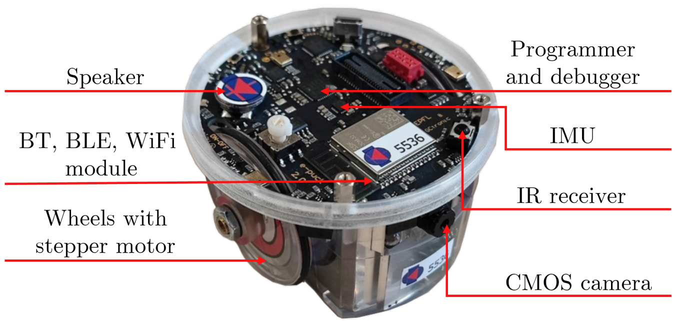


Figure 6.4: E-puck2 main board with its main components, including the speaker, the radio module for communicating with the computer, the IMU for detecting the relative position and orientation of the robot, the programmer and debugger to change the microcontroller firmware, an IR receiver to be remotely controlled, the CMOS camera to onboard image processing. The eight proximity sensors are uniformly distributed in the robot's external circumference and the TOF sensor is located in the frontal area of the robot.

6.2.2 Range & bearing board

Although the e-puck2 can interact with other units through its proximity sensors, local communication can be enhanced by the range & bearing board.

This board features 12 IR emitters and 12 IR receivers uniformly distributed along its circumference, providing a decentralized communication system based on infrared technology with frequency modulation. This allows the robot to send messages of two bytes at a frequency of approximately 40 Hz and to determine the range and bearing of the transmitter at distances of up to about $d_M = 1$ m, which can be tuned by adjusting the transmission power of the IR transmitters.

Given the nature of this communication system, the performance of the range & bearing board can be influenced by light conditions, as well as by the proximity sensors and the TOF sensor integrated into the e-puck2 main board. Therefore, in applications requiring a reliable and distributed communication system, it is recommended to calibrate the range & bearing board's sensors to account for ambient light conditions and minimize interference by reducing the sampling frequency of proximity sensors and deactivating the TOF sensor on the main board.

6.2.3 Pi-puck

The Pi-puck allows the attachment of a Raspberry Pi Zero single-board computer to the robot, providing Linux support, additional peripherals, and expanded functionality. The Pi-puck allows the remote control of the robot from the computer. In particular, there are three different configurations:

1. *WiFi for Pi-puck and Bluetooth for e-puck2*: The computer connects to the Pi-puck via WiFi for data processing and to the e-puck2 via Bluetooth for control. This approach allows the control of a maximum number of 7 robots, and it is characterized by a double latency (Pi-puck to PC and PC to robot).

2. *WiFi for Pi-puck and e-puck2*: The computer connects to both the Pi-puck and e-puck2 via WiFi. In this case, there is also a double latency (Pi-puck to PC and PC to robot).

3. *WiFi for Pi-puck and I^2C for e-puck2*: The computer connects to the Pi-puck via WiFi, which then controls the e-puck2 through I^2C. This is the best approach in terms of latency and simplicity.

In the third configuration, the Pi-puck acts as the master device, while the e-puck2 and the range & bearing board (if used) function as slaves. The Pi-puck controls the actuators and processes sensor data on the main board, while on the range & bearing board, it handles incoming messages (including the range and the bearing of the transmitter), configures outgoing messages, calibrates the IR sensors, and sets the transmission power. Fig. 6.5 illustrates a schematic of this configuration.

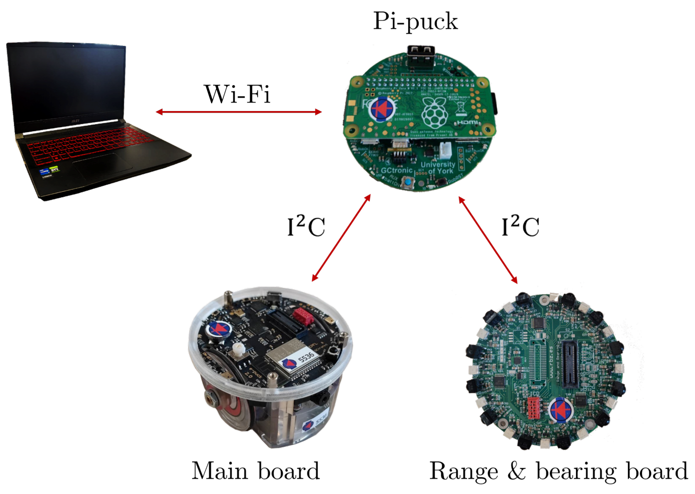


Figure 6.5: The three main modules of the e-puck2: Pi-puck, main board, and range & bearing board interacting according to the *WiFi for Pi-puck and I^2C for e-puck2* configuration. In this setup, the Pi-puck acts as the master device, controlling both the main board and the range & bearing board, which function as slaves. Data is exchanged between the master and its slaves via the I^2C protocol, while communication with the computer is handled over Wi-Fi.

6.3 Comparison between Elisa-3 and e-puck2 robots

Both robotic platforms, Elisa-3 and e-puck2, are well-suited for conducting experiments that aim to validate and explore complex systems phenomena. However, they exhibit distinct characteristics and advantages adapted to different experimental needs, as highlighted in Table 6.1.

The Elisa-3 is designed for tasks requiring less computational complexity and in smaller environments. Its compact size and light weight make it well-suited for experiments that require small space and short-distance interaction. In addition, due to its drive system, can reach velocities greater than the one reached by the e-puck2. Despite its hardware limitations, these can be overcome through the employment of external tools such as an IR camera that allow to detect the robots' positions with an high accuracy.

The e-puck2, on the other hand, offers more advanced capabilities, especially when equipped with its various extensions. These modules, such as extra sensors and the board for enhanced communication, significantly expand its range of applications. This makes the e-puck2 a better choice for experiments that require higher processing power and long-range interaction capabilities. Its more powerful microcontroller and its more reliable odometry system handle a wider variety of tasks.

In summary, both robots are versatile and well-suited for validating coordination phenomena. However, the e-puck2's potential is greatly enhanced by its extensions, making it more adaptable to experiments requiring higher computational complexity and longer communication distances. The Elisa-3, while simpler, remains a solid option for confined experiments.

Table 6.1: Feature comparison: Elisa-3 vs. e-puck2 robots

Aspect	**Elisa-3**	**e-puck2**
Diameter	5 cm	7 cm
Height	3 cm	5.5 cm
Weight	39 g	150 g
Drive system	Differential drive with DC motors and 25:1 reduction gear	Differential drive with stepper motors
Maximum speed	60 cm/s	15.4 cm/s
Obstacle avoidance	8 IR sensors distributed along the circumference	8 IR sensors distributed along the circumference

Aspect	**Elisa-3**	**e-puck2**
Cliff avoidance	4 ground sensors	4 ground sensors (with additional ground module) and 2 cliff sensors (with additional cliff module)
Additional components	3D accelerometer, an IR receiver, 3 IR emitters	IMU (3D accelerometer + 3D gyro + 3D magnetometer), an IR receiver, TOF sensor, CMOS camera
Local communication	IR sensors with a range of up to 5 cm, 1 byte/s throughput	IR sensors supporting communication at distances of up to 6 cm. Range & bearing board with a range of up to 1 m
Long-range communication	2.4 GHz radio module, 1 kHz bandwidth	WiFi, BLE, Bluetooth; Pi-puck module for additional functionality
Processing unit	8-bit Atmel ATmega2460 microcontroller	STM32F4 microcontroller at 168 MHz
Additional modules	None	Range & bearing board, Pi-puck (Raspberry Pi Zero), ground module, cliff module, Omnivision Module V3

Chapter 7

Robotic implementation of multiconsensus

In this chapter, we focus on the robotic implementation of the multiconsensus protocol introduced in Chapter 3. Specifically, we validate the mathematical model by addressing the rendezvous problem with a team of 6 Elisa-3 robots. We begin with a detailed description of our implementation, covering the setup, configuration, and operational aspects of the control action. Then, we present the experimental results, demonstrating the effectiveness and applicability of the communication protocol.

7.1 Multiconsensus induced by network symmetries

Here we briefly recall the communication protocol for multiconsensus introduced in Chapter 3. In particular, we will focus on the case in which there is no pinner.

Let us consider a system of N agents having single-integrator dynamics described by the following equation:

$$\dot{x}_i = \frac{1}{\rho(\mathrm{A})} \sum_{j=1}^{N} a_{ij} x_j - x_i \tag{7.1}$$

for $i = 1, \dots, N$, where a_{ij} are the entries of the adjacency matrix A describing the interaction network, which is assumed to be undirected and connected. Assume that the interaction graph has a set of symmetries forming the group $\mathbb{G}$ and inducing a partition of the vertex set into different clusters (one for each symmetric orbit). Then, according to Lemma 3.1.1, the leading eigenvector $\mathbf{v}_1$ of A satisfies the relation $\mathrm{R}^g\mathbf{v}_1 = \mathbf{v}_1 \ \forall \mathrm{R}^g \in \mathbb{G}$. In Lemma 3.2.3 we proved that, given the state vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$, the system (7.1) evolves as follows:

$$\lim_{t\to+\infty} \mathbf{x}(t) = c\mathbf{v}_1 \tag{7.2}$$

where $c \in \mathbb{R}$ is a scalar depending on the agents' initial conditions. This result implies that the system reaches a state that is parallel to the leading eigenvector of the adjacency matrix describing the interaction network. In this state, symmetric nodes assume the same value that typically differs from cluster to cluster, resulting in a multiconsensus.

When the nodes belonging to the same cluster do not display identical steady-state values but converge to very close values, the system achieves a quasi-multiconsensus.

In Sec. 3.5 we applied Eq. (7.1) to solve the rendezvous problem involving different clusters, in which the units belonging to the same group have to find an agreement on the point where to meet. Specifically, we dealt with the scenario in which the target points are not given. In the case of a bidimensional space, the problem can be solved by applying Eq. (7.1) to each of the two variables describing the agent position. Therefore, by indicating with $\mathbf{x}$ and $\mathbf{y}$ the vectors containing the x and y coordinates of each agent, the dynamics of the multi-agent system is the following:

$$\begin{aligned} \dot{\mathbf{x}} &= \left(\tfrac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{x} \\ \dot{\mathbf{y}} &= \left(\tfrac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right)\mathbf{y} \end{aligned} \tag{7.3}$$

As the two variables $\mathbf{x}$ and $\mathbf{y}$ are independent, Lemma 3.2.3 may be applied to each of Eqs. (7.3), ensuring the convergence to a multiconsensus solution. Specifically, let $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ denote the x and y coordinates of the final positions. The agents will reach positions such that $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are parallel to $\mathbf{v}_1$, leading to symmetric units converging at the same point.

We considered also the case in which the units achieved quasi-multiconsensus, where the agents within the same cluster reached points very close to each other and far from those reached by the agents belonging to other clusters. In this scenario, the units reach positions where $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are nearly parallel to $\mathbf{v}_1$, i.e., $1 - \frac{|\langle \mathbf{v}_1, \bar{\mathbf{x}} \rangle|}{\|\mathbf{v}_1\| \|\bar{\mathbf{x}}\|} < \epsilon_x$, and $1 - \frac{|\langle \mathbf{v}_1, \bar{\mathbf{y}} \rangle|}{\|\mathbf{v}_1\| \|\bar{\mathbf{y}}\|} < \epsilon_y$, with ϵ_x and ϵ_y being small positive scalars.

7.2 Robotic implementation

For the robotic implementation, we use the Elisa-3 robots and the experimental setup detailed in Sec. 6.1.

Given that the implementation of Eq. (7.3) requires the agents communicate based on a fixed and undirected interaction network, relying on local communication is not possible. This is because this communication system, with its maximum communication distance of $d_M = 10$ cm and its limited throughput, generates a time-varying network where the proximity of agents regulates the presence or absence of a link. Additionally, the robots need to continuously share their current positions with the other units, but due to the absence of a reliable odometry system, this is not feasible. Consequently, due to these hardware limitations, the control action cannot be fully distributed, and an IR camera for tracking and a computer to manage the control via a radio link are employed. To ensure that each robot properly receives the control directives from the computer at each time step $t_h = ht_s$ (with $h = 1, 2, \ldots$), t_s has to be selected such that $t_s > t_d$, where t_d is define in Chapter 6 by Eq. 6.1. Specifically, we select $t_s = 100$ ms.

To properly operate, the tracking system needs calibration. First, a linear calibration of the 2D vision system is carried out to transform the robot position from pixel to real-world coordinates [37]. This step needs to be accomplished only once the setup is established. Instead, at the beginning of each experimental session, the exposure, brightness, and contrast of the IR camera are all tuned. Finally, at the beginning of each run, a third calibration step is performed: each robot spins one at a time so that the tracking software can associate the moving robot with the corresponding particle.

Robot motion is controlled in the following way. At each time step t_h, the IR camera tracks the current positions of the robots, represented by the vectors $\mathbf{x}^{(P)}(t_h) = \left[x_1^{(P)}(t_h), x_2^{(P)}(t_h), \ldots, x_N^{(P)}(t_h)\right]^T$ and $\mathbf{y}^{(P)}(t_h) = \left[y_1^{(P)}(t_h), y_2^{(P)}(t_h), \ldots, y_N^{(P)}(t_h)\right]^T$, containing the x and y coordinates of the robots, respectively. The positions of all the robots are recorded in a file and processed by the computer, which integrates with a discrete-time step Eqs. (7.1), using the adjacency matrix of the interaction graph to compute the target positions for the robots at the next time step t_{h+1}, denoted by $\mathbf{x}^{(T)}(t_{h+1}) = \left[x_1^{(T)}(t_{h+1}), x_2^{(T)}(t_{h+1}), \dots, x_N^{(T)}(t_{h+1})\right]^T$ and $\mathbf{y}^{(T)}(t_{h+1}) = \left[y_1^{(T)}(t_{h+1}), y_2^{(T)}(t_{h+1}), \dots, y_N^{(T)}(t_{h+1})\right]^T$, as follows:

$$\begin{aligned} \mathbf{x}^{(T)}(t_{h+1}) &= \mathbf{x}^{(P)}(t_h) + t_s \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right) \mathbf{x}^{(P)}(t_h) \\ \mathbf{y}^{(T)}(t_{h+1}) &= \mathbf{y}^{(P)}(t_h) + t_s \left(\frac{1}{\rho(\mathrm{A})}\mathrm{A} - \mathrm{I}_N\right) \mathbf{y}^{(P)}(t_h) \end{aligned} \tag{7.4}$$

Once the target positions are determined, the computer calculates the required velocities of both the right and left wheels for each robot. These velocities are organized into two vectors: $\mathbf{v}_r(t_h)$ for the right wheel and $\mathbf{v}_l(t_h)$ for the left wheel. The computer then sends these values to the robots, enabling them to move toward the calculated target positions. However, there is no guarantee that the points reached by the robots at time t_{h+1}, represented by the vectors $\mathbf{x}^{(P)}(t_h)$ and $\mathbf{y}^{(P)}(t_h)$, will correspond to the target points computed at time t_h, expressed by $\mathbf{x}^{(T)}(t_{h+1})$ and $\mathbf{y}^{(T)}(t_{h+1})$, as the robots may not be able to reach the target positions in the time window t_s. Figure 7.1 presents the block diagram that describes the control process in the robotic implementation.

The control goal is accomplished when the following conditions are fulfilled:

$$\begin{aligned} 1 - \frac{|\langle \mathbf{v}_1, \mathbf{x}^{(P)}(t_h)\rangle|}{\|\mathbf{v}_1\| \; \|\mathbf{x}^{(P)}(t_h)\|} &< \epsilon_x \\ 1 - \frac{|\langle \mathbf{v}_1, \mathbf{y}^{(P)}(t_h)\rangle|}{\|\mathbf{v}_1\| \; \|\mathbf{y}^{(P)}(t_h)\|} &< \epsilon_y \end{aligned} \tag{7.5}$$

These conditions imply that the system reaches a quasi-multiconsensus. Notice that, in contrast to the scenario studied in Sec. 3.5, the system does not achieve a *perfect* multiconsensus not due to a perturbation of the interaction graph, but because reaching this condition is unfeasible. Specifically, achieving such a condition would require all robots within the same cluster to occupy the same position, leading to overlap. However, in other applications where there are no physical limits, the multiconsensus can be achieved.

Once the conditions in Eq. (7.5) are fulfilled, $\mathbf{v}_r$ and $\mathbf{v}_l$ are set to $\mathbf{0}$, resulting in the robots stopping their motion. Algorithm 1 summarizes all the steps performed by the computer during the control action.

We emphasize that in a scenario where experiments are conducted with robots equipped with accurate odometry sensors and a local communication system that allows for long-distance communication with a larger bandwidth, the control action could be distributed. Indeed, with knowledge of their positions and the ability to exchange this information with their neighbors (based on a fixed interaction network with symmetries), it is possible to solve the rendezvous problem without employing central tools such as the computer and the camera.

Algorithm 1: Control action aiming to induce the robots to reach multiconsensus: central station

```
   Parameter      : A, $t_s$, $\epsilon_x$, $\epsilon_y$
   Initialization : $t$ = getCurrentTime
1  Perform linear calibration
2  while true do
3      if (getCurrentTime $- t) \geq t_s$ then
4          Track the robot positions $\mathbf{x}^{(P)}$
5          Compute the target positions $\mathbf{x}^{(T)}$ as in Eq. (7.4)
6          Determine $\mathbf{v}_l$ and $\mathbf{v}_r$ to reach $\mathbf{x}^{(T)}$
7          if conditions Eqs. (7.5) holds then
8              $\mathbf{v}_r = \mathbf{0}$
9              $\mathbf{v}_l = \mathbf{0}$
10         end
11         for $i = 1 : N$ do
12             Send $v_{r,i}$ and $v_{l,i}$ to robot $i$
13         end
14         $t$ = getCurrentTime
15     end
16 end
```

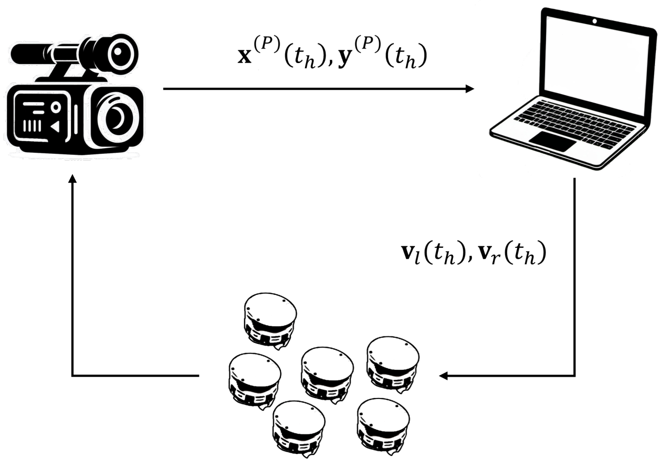

$\mathbf{x}^{(P)}(t_h), \mathbf{y}^{(P)}(t_h)$
$\mathbf{v}_l(t_h), \mathbf{v}_r(t_h)$

Figure 7.1: Block diagram describing the control action. At each time step t_h the IR camera sends the robots' positions to the computer, which, using the control law in Eq. (7.1), computes the velocities required for the robots to reach their target positions of the next time step t_{h+1}.

7.3 Results

Here we show the results of an experiment carried out using the implementation described in the previous section.

We consider $N = 6$ units moving in an arena of size $L_{y_1} = 80$ cm and $L_{y_2} = 60$ cm and we set $\epsilon_x = \epsilon_y = 0.005$. The robots interact according to the graph shown in Fig. 7.2(a), where we can identify three different clusters induced by the network symmetries: $C_1 = \{1, 2\}$ (in red), $C_2 = \{3\}$ (in green), and $C_3 = \{4, 5, 6\}$ (in blue). The adjacency matrix describing this graph has a maximum eigenvalue $\rho(\mathrm{A}) = 2.51$ and an associated leading eigenvector $\mathbf{v}_1 = [0.43,\ 0.43,\ 0.65,\ 0.26,\ 0.26,\ 0.26]^T$. Notice that, as we expect from Lemma 3.1.1, the entries corresponding to symmetric nodes are the same. At the beginning of the experiment, the agents are manually placed in the following positions:

$$\begin{aligned} \mathbf{x}^{(P)}(0) &= [34.51,\ 69.66,\ 68.99,\ 19.88,\ 20.39,\ 59.75]^T \\ \mathbf{y}^{(P)}(0) &= [45.53,\ 23.93,\ 40.79,\ 34.11,\ 9.05,\ 3.18]^T \end{aligned} \tag{7.6}$$

These positions are represented by circles in Fig. 7.2(b). Once the experiment begins, the control action described in the previous section leads the robots to perform the trajectories shown in Fig. 7.2(b). After $T = 8$ s, the robots stop their motion at the positions:

$$\begin{aligned} \mathbf{x}^{(P)}(T) &= [44.14,\ 45.81,\ 64.38,\ 22.57,\ 25.44,\ 28.4]^T \\ \mathbf{y}^{(P)}(T) &= [29.43,\ 32.11,\ 41.31,\ 17.6,\ 13.84,\ 14.39]^T \end{aligned} \tag{7.7}$$

which are represented by the squares in Fig. 7.2(b). Specifically, the system reaches a quasi-multiconsensus, as $1 - \frac{|\langle \mathbf{v}_1, \mathbf{x}^{(P)}(T)\rangle|}{\|\mathbf{v}_1\|\|\mathbf{x}^{(P)}(T)\|} = 0.0013$ and $1 - \frac{|\langle \mathbf{v}_1, \mathbf{y}^{(P)}(T)\rangle|}{\|\mathbf{v}_1\|\|\mathbf{y}^{(P)}(T)\|} = 0.0042$. Fig. 7.3 reports five snapshots taken from a video recorded by the RGB camera placed above the arena at times $t_h = 0$ s, $t_h = 2$ s, $t_h = 4$ s, $t_h = 6$ s, $t_h = T = 8$ s, capturing the progression of the system towards the solution of the rendezvous problem.

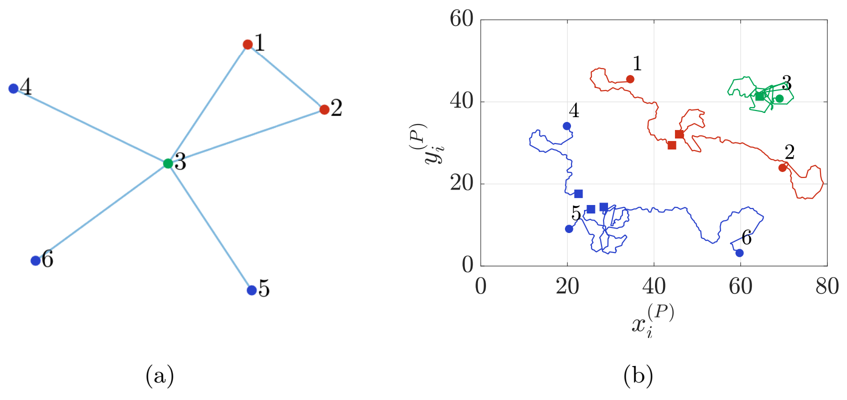


Figure 7.2: Experimental results. (a) Graph describing the interaction among the robots. (b) Trajectories performed by the robots to achieve the multiconsensus. The circles represent the initial points, whereas the squares the final positions.

In this chapter, we have demonstrated the feasibility of applying our communication protocol introduced in Chapter 3 in a real-world scenario. Specifically, we have addressed the rendezvous problem using a team of Elisa-3 robots, showing that they can reach a quasi-multiconsensus in a short amount of time. Due to the hardware limitation of these robots, our control strategy is not fully distributed and requires a reference frame to compute the agent position. However, with robots equipped with accurate odometry

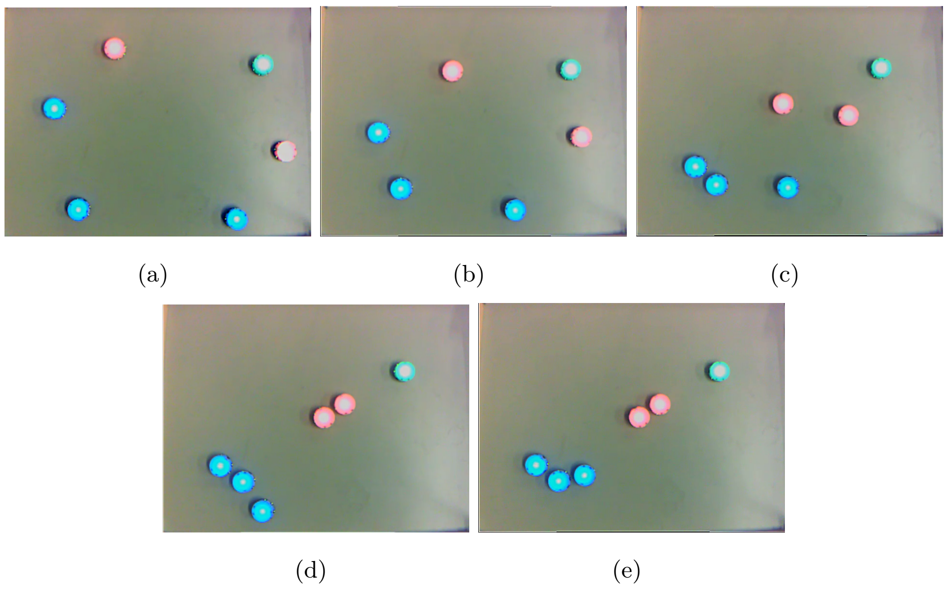


Figure 7.3: Dynamical evolution of multiconsensus in an experiment made with $N = 6$ Elisa-3 robots captured by the RGB camera. The different snapshots are taken at the following times: (a) $t_h = 0$ s, (b) $t_h = 2$ s, (c) $t_h = 4$ s, (d) $t_h = 6$ s, and (e) $t_h = 8$ s. The units interact according to the graph shown in Fig. 7.2(a). In particular, each robot has an RGB LED that flashes the same color as the corresponding node in the graph. At $t_h = 0$ s the robots start from different positions. Then, they follow the trajectories shown in Fig. 7.2(b) with snapshots (b), (c), and (d) capturing their progression. Finally, the robots reach their final positions at $t_h = 8$ s, achieving the solution of the rendezvous problem.

sensors and a local communication system allowing for long-distance communication with higher bandwidth, a fully distributed control action can be implemented.. Due to the generality of our communication protocol, it can also be exploited in other applications requiring multiconsensus.

In the next chapter, we will introduce a distributed control action that allows the experimental investigation of the face-to-face interaction dynamics in settings where the parameters can be controlled.

Chapter 8

A multi-robot system for the study of face-to-face interaction dynamics

Here, we leverage the theoretical insights from mathematical models based on complex networks of face-to-face interaction dynamics to propose a multi-robot system that facilitates experimental investigations of these dynamics in settings where the parameters are controllable. Specifically, we consider a team of Elisa-3 robots and implement a distributed and decentralized control law that takes into account the key mechanisms of interaction giving rise to face-to-face dynamics.

8.1 The attractiveness-based model for face-to-face interaction networks

The attractiveness-based model for face-to-face interaction networks [95] considers a group of N agents distributed on a plane where they move and interact according to the following rules. Each agent is characterized by a parameter a_i, representing its attractiveness, namely how likely other agents, which get in touch with it, will be engaged in a face-to-face interaction with it. At each time step $t_k = k\Delta h$ with Δh constant and $k = 0, 1, \ldots, K$, an agent can either perform a random walk or remain in its previous position to interact with one or more agents that are attracting its interest. In particular, a stochastic process regulates the action performed by the agent, such that with probability $p_i(t_k)$ the agent moves and with probability $1 - p_i(t_k)$ it does not change its position and interacts face-to-face with each of neighbors. The probability $p_i(t_k)$ is a function of the attractiveness of the neighboring agents, namely

$$p_i(t_k) = 1 - \max_{j \in \mathcal{N}_i(t_k)} a_j \tag{8.1}$$

where $\mathcal{N}_i(t_k)$ is the neighborhood of agent i at time t_k. In more detail, if we indicate with $\mathbf{y}_i(t_k)$ the position of agent i in the plane at time t_k and with r the sensing radius of each agent, then, $\mathcal{N}_i(t_k)$ is the set of agents that at time t_k are at a distance smaller than r, i.e., $\mathcal{N}_i(t_k) = \{j : \|\mathbf{y}_j(t_k) - \mathbf{y}_i(t_k)\|^2 \leq r\}$.

To introduce the motion equations, let us indicate with $\mathbf{v}_i(t_k) = ve^{\mathrm{i}\theta(t_k)}$ the linear velocity of agent i. Here, v denotes the modulus of the velocity that is maintained constant in time, while $\theta(t_k)$, the agent heading, is a quantity that changes randomly at each time step Δ. Then, with probability $p_i(t_k)$ agent i performs a random walk and its position is updated as follows

$$\mathbf{y}_i(t_{k+1}) = \mathbf{y}_i(t_k) + \mathbf{v}_i(t_k)\Delta h \tag{8.2}$$

while with probability $1 - p_i(t_k)$ its position $\mathbf{y}_i(t_{k+1})$ remains the same of the previous step, i.e., $\mathbf{y}_i(t_{k+1}) = \mathbf{y}_i(t_k)$.

In view of a robotic implementation of the model, there are several aspects to consider. First of all, notice that $p_i(t_k)$ changes over time, as the neighborhood does. In the original model [95], p_i is updated with the same step size, namely Δh, of the random walk process. However, in a robotic implementation, where agents are no more dimensionless particles, this strategy is not suitable. On the contrary, obstacle avoidance should always be active in order to avoid collisions with other robots or with the physical boundaries of the arena where they move. In addition, turning/heading update is not instantaneous (as this would imply an infinite angular velocity), but requires a finite amount of time. Finally, the third important ingredient to take into consideration is that the bandwidth of the communication between agents is limited.

To account for these important factors in the physical implementation, here we extend the original attractiveness-based model by revisiting some of the model assumptions and including some further parameters. In the original formulation of the model periodic boundary conditions are considered; here, for the sake of comparison with the experiments, also in the mathematical model we consider that at the boundaries of the arena there are fixed walls. Furthermore, in our numerical simulations, rather than performing the motion step of the random walk in a single time interval, we consider a smaller step size $\delta h < \Delta h$ and check after each interval of fixed length δh whether during its motion the agent finds an obstacle or not (the obstacle can be one of the arena boundaries or another unit). If there are no encounters, then the full motion step of length $v\Delta h$ is performed, otherwise two situations may occur. If the obstacle is one of the arena walls, the robot stops, rotates in a random direction and, then, continues its random walk. Otherwise, if the obstacle is another unit of the team, the agent stops at the position of the encounter, effectively performing a random walk step of a smaller length. At this point, the agent has to 'decide' whether to engage in an interaction or not, according to the probability p_i. To take into account the limited bandwidth of the communication link between the units, the agent remains at a fixed position for a time interval equal to t_c and updates its decision at intervals of t_c. Hence, if an agent decides to engage in an interaction, then the duration of such an interaction will not be shorter than t_c. Summing up, our model includes two new parameters, δh and t_c, and the additional rules to account for the obstacle avoidance protocol that needs to be always active in our robotic implementation.

8.2 Robotic implementation

The robotic implementation of the attractiveness-based model for face-to-face interactions has been developed using a team of Elisa-3 robots, which are described in Sec. 6.1. Robots move in an arena of size L_{y_1} and L_{y_2} as schematically represented in Fig. 8.1, which also illustrates the radius of local communication, r. This is an important parameter for the dynamics as interactions can occur only with robots within this radius. On top of the arena, an RGB camera allowing the recording of robot trajectories is mounted.

The control of the robot's autonomous behavior is carried out through a finite state machine that implements the mechanisms of interaction and

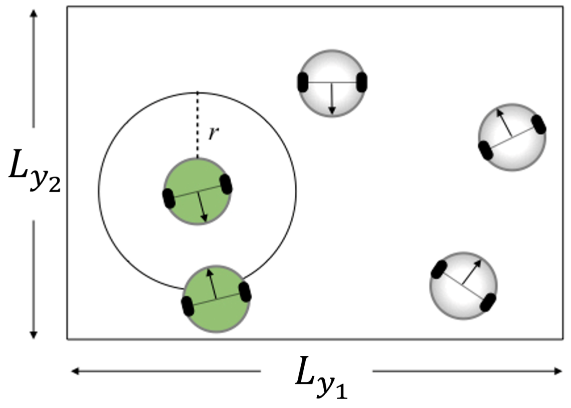


Figure 8.1: Schematic representation of the team of Elisa-3 robots implementing attractiveness-based face-to-face dynamics. Robots represented in green color are communicating with each other through the local communication system (with d_M indicating the maximum distance for such communication), while the other units (in grey color) are not detecting other robots or obstacles within their communication/sensing radius and, hence, are moving as random walkers.

movement of the attractiveness-based model for face-to-face dynamics. As shown in Fig. 8.2, the state machine has four states: 'Random walk & obstacle avoidance', 'Stop & check', 'Compute probability', and 'Engaged & listening'. In the first state, the robot moves as a random walker along the arena, while avoiding potential obstacles and continuously checking whether a message from other robots is received. When this occurs, the state machine moves to the 'Stop & check' state, where the robot stops at the current position for a period of time equal to t_c, searching for eventual messages from other robots and retrieving the value of the attractiveness of the neighboring robots. After that, it moves to the 'Compute probability' state where the robot calculates p_i according to Eq. (8.1). At this point, with probability $1 - p_i$ moves to the 'Engaged & listening' state, thus starting a face-to-face interaction, or with probability p_i goes back to the 'Random walk & obstacle avoidance' state, thus moving away from the current position as the other robots (if any) have not raised its interest. The 'Engaged & listening' state, therefore, represents the condition where the robot is effectively interacting face-to-face with other robots, forming a group of two or more units. From this state, where messages from other robots are also continuously checked, after a period of time equal to t_c, the robot moves back to 'Compute probability' from which the robot decides to continue the face-to-face interaction (with probability $1-p_i$) or leave the group (with probability p_i). In the two states 'Stop & check' and 'Engaged & listening' the robot's LED turns green and red respectively, in order to help visually detect in which state the robot is. The finite state machine is implemented in the robot via Algorithm 2.

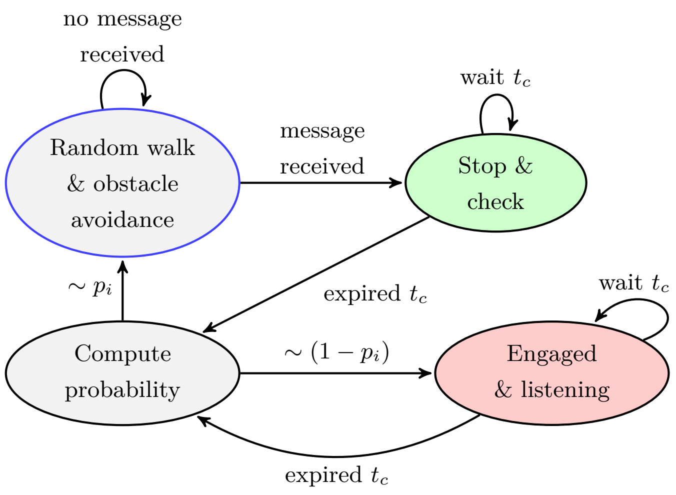


Figure 8.2: Finite state machine used to control the autonomous behavior of each robot of the team. The finite state machine is initialized in the 'Random walk & obstacle avoidance' state (visually highlighted in the scheme with a blue contour).

When the robot is not interacting with other units, it moves as a random walker with obstacle avoidance control always active. Taking into account that the robots are controlled by differential drive, we implemented the random walk in two steps. In the first step, we randomly set the direction of rotation of the robot (left or right) and then rotate it for a time equal to t_r which is randomly drawn with uniform distribution in the interval $[0.1s, 1s]$. This results in an in-place rotation of the robot by an angle randomly drawn with uniform distribution in the interval $[-\pi, \pi]$. Then, in the second step, the robot moves forward for a period of time equal to t_f (which corresponds to Δh in Eq. (8.2)) with fixed velocity v. Notice

Algorithm 2: Face-to-face dynamics for agent i

```
Parameter       : $a_i$, $t_c$, $t_f$, $v$
Initialization  : State = RandomWalk
while true do
    Broadcast $a_i$
    Check IR sensors
    if (messageReceived==true) then
        Stop moving
        State = MsgReceived
    end
    switch State do
        case RandomWalk do
            Move as a random walker with obstacle avoidance
        end
    end
    case MsgReceived do
        Turn on green light
        Wait $t_c$
        State = ComputeProb
    end
    case ComputeProb do
        Compute $p_i$ as in Eq. (8.1)
        Generate a random number $\xi \in [0, 1]$
        if $\xi < p_i$ then
            State = RandomWalk
        else
            State = Engaged
        end
    end
    case Engaged do
        Turn on red light
        Wait $t_c$
        State = ComputeProb
    end
end
```

that, since obstacle avoidance is always active, vt_f represents the maximum distance of the random walk step, while it is the exact distance only when no obstacles (either the arena walls or other robots) are encountered during this motion step. Obstacle avoidance is implemented by continuously checking eventual obstacles via the IR sensors. When the sensors signal the presence of an obstacle, then either this obstacle is a robot (as simultaneously a message has been received) and the finite state machine moves to the 'Stop & check' state, or it is a wall of the arena. In this latter case, the robot heading is randomly changed in the $[-\pi, \pi]$ interval and, then, the robot continues its motion step.

An important parameter of our experiments is the time that a robot has to wait to correctly receive a message after the IR sensors have detected it. This parameter is briefly indicated as the time for local communication t_c. The nominal value of the local communication throughput for the Elisa-3 robot is about 1 byte/s; however, based on a series of preliminary experiments that we have run to investigate whether the value of t_c could be reduced, we have selected $t_c = 700$ ms as a trade-off between message loss and the time the robot has to spend while waiting for messages.

During the experiments, the robots also communicate their status, including potential interactions with other units, to a PC through the radio link, at time intervals of 100 ms. Communication with the PC is solely used to record the relevant information to analyze the collective behavior of the system, but not for the robot control law, which is fully decentralized and distributed.

Two other important parameters of the setup are v and t_f, which we have empirically set to 6 cm/s and 2 s, respectively. These are trade-off values between the size of the area explored by a robot during its motion (which becomes larger as the two parameters increase) and the probability of receiving messages from other robots (which decreases as the two parameters increase).

Finally, two other parameters influence the collective behavior of the whole system of interacting robots. They are the robot density ρ and the values of the attractiveness of each robot, stored in a single vector $\mathbf{a} = [a_1, a_2, \dots, a_6]^T$. As the number of robots is kept fixed, the density of the robots is determined by the dimensions of the arena, namely

$\rho = N/(L_{y_1}L_{y_2})$. The values of attractiveness are set in the interval $[0, 1]$ and are assigned to each robot of the team at the beginning of each trial and then kept constant. The main symbols used in this chapter are summarized in Table 8.1.

Table 8.1: List of symbols

Symbol	Parameter
t_c	time for local communication
t_r	rotation time
t_f	forward time
v	robot velocity
a_i	attractiveness of robot i
L_{y_1}, L_{y_2}	length and width of the arena
ρ	robot density
T	test duration

8.3 Experimental results

To illustrate our results, we first discuss an experiment for a fixed setting of the system parameters. In particular, here we have considered $N = 6$ robots moving in an arena with $L_{y_1} = 80$ cm and $L_{y_2} = 60$ cm, thus resulting in a density of robots equal to $\rho = 1.3 \times 10^{-3}$ cm^{-2}. The robot attractiveness is arbitrarily fixed as: $\mathbf{a} = [0.75, 0.88, 0.93, 0.67, 0.72, 0.67]$. Fig. 8.3 shows five snapshots of a video recorded by the RGB camera. It shows the dynamic evolution of the formation of groups in the system. Initially, two robots form a group (Fig. 8.3(a)). Later, another robot communicates with one of the group units and joins it (Fig. 8.3(b-c)). Subsequently, a different unit communicates with a member of the group (Fig. 8.3(d)), and engages in a face-to-face interaction to form a new group of two robots (Fig. 8.3(e)), while the previous group breaks apart.

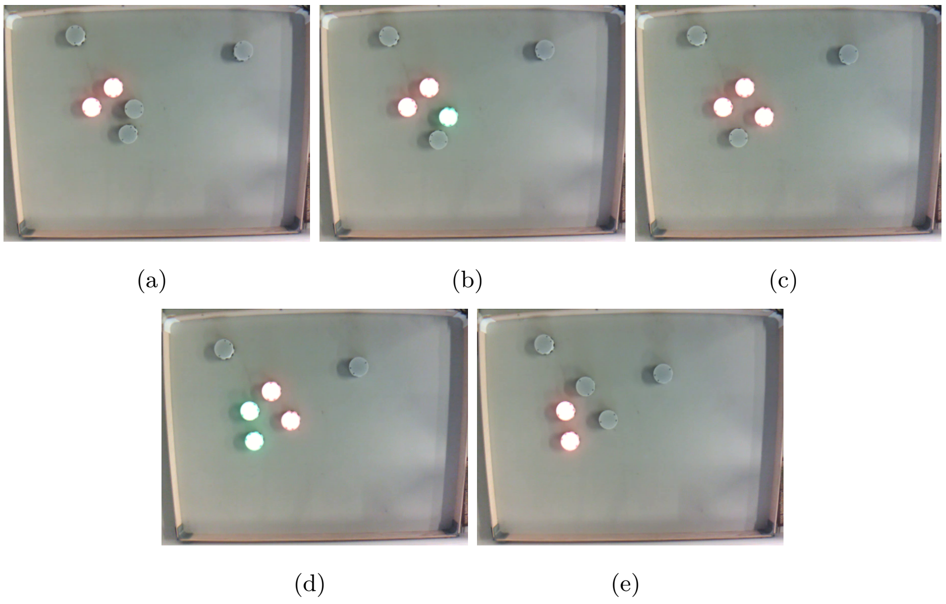

(a) (b) (c) (d) (e)

Figure 8.3: Dynamical evolution of group formation in an experiment made with $N = 6$ Elisa-3 robots. The different snapshots are taken from a video recorded by the RGB camera at the following times: (a) $t = 63$ s, (b) $t = 64$ s, (c) $t = 65$ s, (d) $t = 66$ s, and (e) $t = 67$ s. The units lighting up with red (green) light are interacting (communicating) with each other, while no light indicates that the robot is performing a random walk. (a) The robots form a group of two units that are engaged in a face-to-face interaction. (b) A third robot is communicating with one of the two units of the group. (c) The group is now formed by three interacting units. (d) Another robot in the area is communicating with units in the group. (e) The previous group breaks apart and a new group of two units forms, while the other two robots, belonging to the previous group, move away from the area of the meeting.

We have then conducted a more systematic investigation by exploring various parameter settings and performing a statistical analysis of the data gathered during our experiments to determine the distribution of the contact duration, denoted as $P(\Delta t)$, as well as the distribution of the time interval, denoted as $P(\tau)$, between two consecutive interactions of a robot with some other unit. Here, with the term contact we indicate the engagement in a face-to-face interaction of two or more robots. In particular, we have considered exemplificative settings with different values of the density ρ and the attractiveness vector $\mathbf{a}$. To change ρ, we have kept fixed the number of robots and changed the size of the arena (in particular, varying L_{y_1}), whereas the attractiveness vector $\mathbf{a}$ has been set via robot programming. The time for local communication between the robots has been set to $t_c = 700$ ms in all the experiments where not differently mentioned. The same holds for the speed module that is fixed to $v = 6$ cm/s.

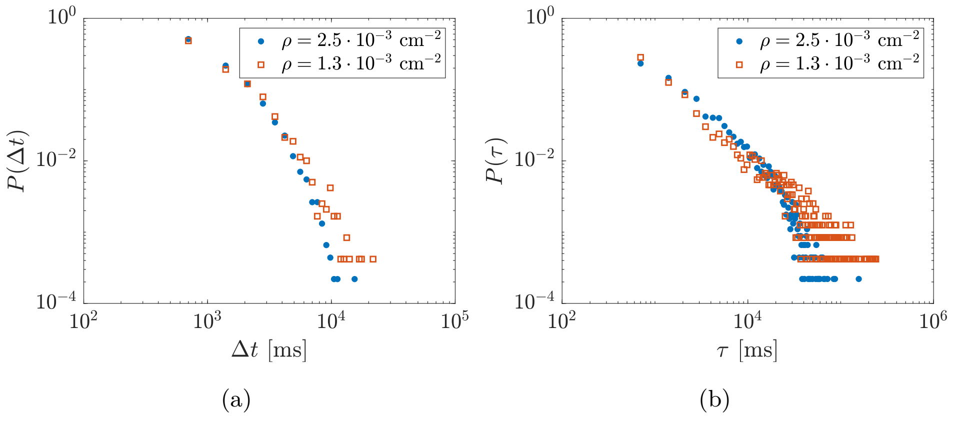


Figure 8.4: Experimental results. Effect of the density on the distribution of the contact duration, $P(\Delta t)$, (a) and on the distribution of the time interval between consecutive contacts, $P(\tau)$, (b). The results are the average of ten runs for each density value, which is controlled by changing L_{y_1} ($L_{y_1} = 80$ cm for the first set of runs, represented with blue circles, $L_{y_1} = 40$ cm for the second one, represented with orange squares). The remaining parameters are fixed to: $t_c = 700$ ms, $L_{y_2} = 60$ cm, $T = 10$ min, $t_f = 2$ s, $v = 6$ cm/s, $t_r \in [100$ ms, 1 s$]$, $N = 6$, $\langle \mathbf{a} \rangle = 0.77$.

We first discuss the effect of the density, considering two different values of ρ, namely $\rho_1 = 1.3 \times 10^{-3}$ cm^{-2}, obtained by setting $L_{y_1} = 80$ cm and $L_{y_2} = 60$ cm, and $\rho_2 = 2.5 \times 10^{-3}$ cm^{-2}, obtained by resizing the arena such that $L_{y_1} = 40$ cm and $L_{y_2} = 60$ cm. The results are illustrated in Fig. 8.4 that shows the distribution of the contact duration, $P(\Delta t)$, and that of the time intervals, $P(\tau)$, for $\rho = \rho_1 =$ (blue circles), and $\rho = \rho_2$ (orange squares). Both distributions are obtained by dividing the empirical data into bins and reporting the number of observations for each bin divided by the total number of observations. The results are averaged over 10 different runs, in each of which the behavior of the robots was monitored for a period of time equal to $T = 10$ min. The robot attractiveness is fixed to $\mathbf{a} = [0.75, 0.88, 0.93, 0.67, 0.72, 0.67]$, such that the average value is equal to $\langle \mathbf{a} \rangle = 0.77$.

We notice that, for both values of ρ, the two distributions $P(\Delta t)$ and $P(\tau)$ display a long-tailed power-law form spanning a few orders of magnitude. This is in agreement with the analysis of the original mathematical model where this behavior is consistently observed for different settings of the parameters [95]. From the analysis of $P(\Delta t)$ (Fig. 8.4(a)), we find that the number of contacts with a small duration is high, whereas there are few cases where the duration of the contact between robots is large. Analogously, the behavior of $P(\tau)$ (Fig. 8.4(b)) is also characterized by a long-tailed power-law distribution, meaning that there are few contacts occurring after long time intervals and many after small time intervals. The density ρ seems to significantly affect the distribution of τ (Fig. 8.4(b)), where we notice that, for the larger value of density, it is more difficult to observe larger time intervals compared to the case of smaller density. Instead, the density has a smaller effect on the distribution of contact duration, with a more noticeable impact on the tail of the distribution.

During their motion, robots interact with other robots in a dynamical way, forming groups of different sizes. This can be illustrated by considering the number of interactions between each pair of robots and counting the occurrence of the group size during a robot experiment. Fig. 8.5(a)-(c) reports the aggregated network obtained by considering all interactions taking place between pairs of robots in time windows of increasing duration. The aggregated network is represented with edges of thickness proportional to the number of interactions, normalized with respect to the total number of interactions in that time window. For a time window of small duration, not all pairs of nodes are connected, as during their motion the robots did not have the chance to meet all the other units. As more time elapses, more links emerge, up to the point where the aggregated network is a complete graph, with the weights reflecting the stochasticity in the robot dynamics. Fig. 8.5(d) reports the occurrence of groups of different sizes n. As expected, smaller groups occur more frequently than larger ones.

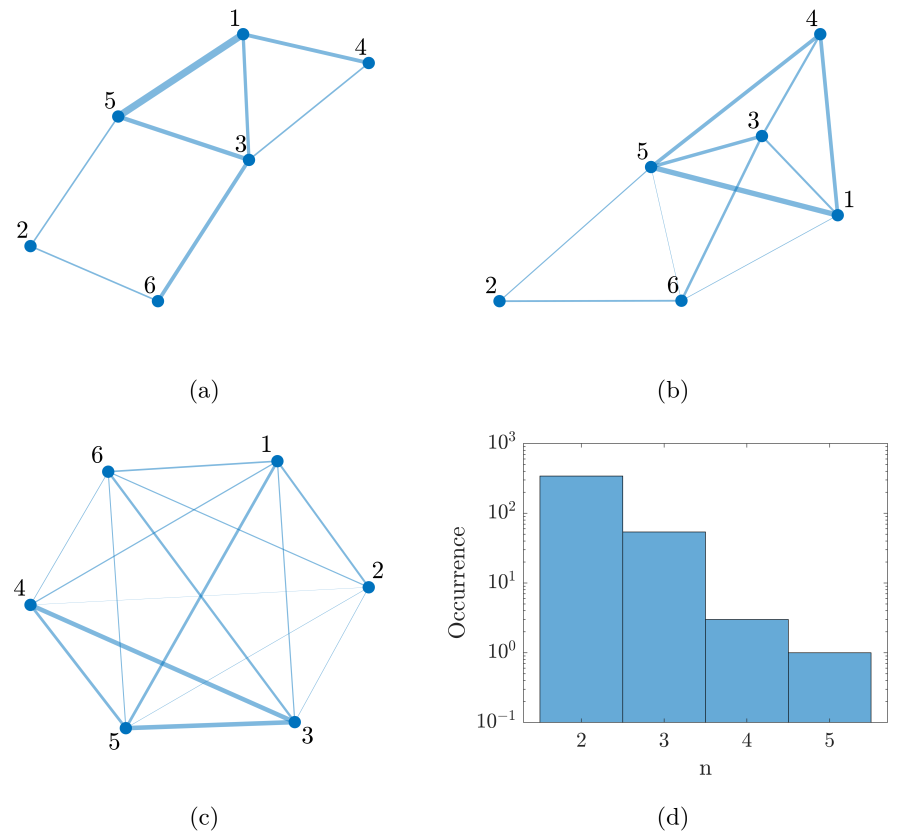
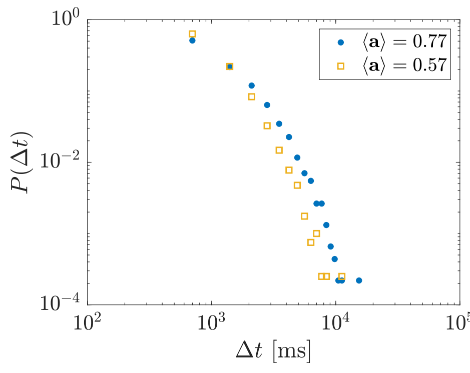
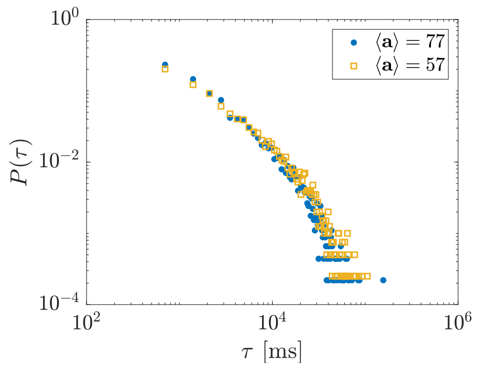


Figure 8.5: Interactions among agents and group formation in an experiment with $N = 6$ Elisa-3 robots. Aggregated network illustrating the number of interactions among agents after 1 minute (a), 3 minutes (b), and 10 minutes (c), and occurrences of groups of size n during robot experiment (d). The experiment has been carried out using the following parameters: $t_c = 700$ ms, $L_{y_1} = 40$ cm, $L_{y_2} = 60$ cm, $T = 10$ min, $t_f = 2$ s, $v = 6$ cm/s, $t_r \in$ [100 ms, 1 s], $\langle \mathbf{a} \rangle = 0.77$.

Next, we discuss the experiments we have carried out to investigate the effects of different values on the robot attractiveness. In particular, we have considered two sets of values for $\mathbf{a}$: $\mathbf{a} = \mathbf{a}_1 = [0.75, 0.88, 0.93, 0.67, 0.72, 0.67]$, representing a scenario where the average attractiveness is high, i.e., $\langle \mathbf{a} \rangle = 0.77$, and $\mathbf{a} = \mathbf{a}_2 = [0.40, 0.76, 0.62, 0.55, 0.42, 0.67]$, representing a scenario with a lower value of average attractiveness, i.e., $\langle \mathbf{a} \rangle = 0.57$. The distributions $P(\Delta t)$ and $P(\tau)$ obtained under these conditions are reported in Fig. 8.6. Here, we have considered $\rho = 2.5 \times 10^{-3}$ cm^{-2}, while the other parameters, such as the number of agents N, the robot velocity v, the experiment duration T, and the time for local communication t_c are fixed as in the previous set of experiments. Also in this case, we find that, for all values of $\mathbf{a}$, both $P(\Delta t)$ and $P(\tau)$ follow a long-tailed power law. Here, we notice that, while changing the attractiveness has a low impact on $P(\tau)$ (Fig. 8.6(b)), it significantly affects the distribution of the contact duration $P(\Delta t)$ (Fig. 8.6(a)). When the average value of attractiveness is lower, the number of contacts with short duration increases, while the number of contacts with long duration decreases, compared to the case where the attractiveness values are higher.



(a)



(b)

Figure 8.6: Experimental results. Effects of the robot attractiveness, $\mathbf{a}$, on the distribution of the contact duration, $P(\Delta t)$, (a) and on the distribution of the time interval between consecutive contacts, $P(\tau)$, (b). The results are the average of ten runs for each value of $\mathbf{a}$ (in the first case, represented with blue circles, the robot attractiveness is such that $\langle \mathbf{a} \rangle = 0.77$, while in the second case, represented with yellow squares, is such that $\langle \mathbf{a} \rangle = 0.57$). The remaining parameters are fixed to: $t_c = 700$ ms, $L_{y_1} = 40$ cm, $L_{y_2} = 60$ cm, $T = 10$ min, $t_f = 2$ s, $v = 6$ cm/s, $t_r \in [100$ ms, 1 s$]$, $N = 6$.

The face-to-face dynamics experimentally obtained has been then compared with numerical simulations of the model discussed in Sec. 8.1. We have found that the model can produce contact durations and time intervals between consecutive contacts having distributions similar to those observed experimentally, when faults in the local communication among agents are explicitly taken into account in the model. To this aim, we have introduced in the model a further parameter, p, mimicking the effect of loss of messages. This parameter represents the probability that each neighboring agent of a generic unit i is correctly perceived as such and, therefore, able to exchange information with it. Hence, with probability $1-p$, agent i misses the message sent by the other agent, despite it being one of its neighbors, and, thus, does not take into account its attractiveness in evaluating Eq. (8.1). In the robotic experiments, the loss of messages depends on the time for local communication, t_c, on the robot motion and on the asynchronous communication, and, as such, it is difficult to estimate. For this reason, in our analysis, the parameter p has been empirically set to the value $p = 0.1$.

We illustrate the comparison between experiments and simulations with reference to the case $\rho = \rho_2$ and $\mathbf{a} = \mathbf{a}_1$, noting that similar results have been obtained for other settings of the parameters. The distributions $P(\Delta t)$ and $P(\tau)$, obtained experimentally and numerically for these values of the parameters, are shown in Fig. 8.7. Although the distributions are similar, there are still some differences. In particular, in the experimental case, we find slightly longer time intervals between consecutive contacts. This suggests that other quantities, not explicitly taken into account in the model (such as the finite time for heading change or robot deviations from straight motion), are also influencing the behavior of the multi-robot system.

Finally, we discuss an example where robots are operated under challenging conditions. Specifically, we set $t_c = 500$ ms, which allocates a very short time window for message reception in the local communication system. This time window is below the inverse of the nominal throughput, resulting in a large loss of messages. Despite this, the main features of face-to-face interaction dynamics still emerge. As shown in Fig. 8.7(a), the distribution of contact durations is characterized by only a few points, as long-duration contacts become unlikely. However, these points are still distributed according to a power law. On the other hand, as shown in Fig. 8.7(b), the adverse operating conditions considered in this experiment do not seem to impact the distribution of time intervals between successive contacts, which contains many points and displays a long-tailed power-law form similar to those observed for other settings of the system parameters.

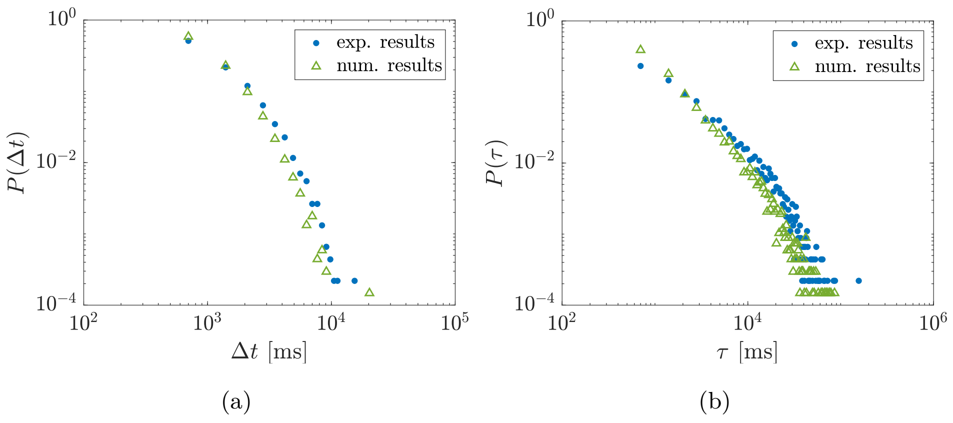


Figure 8.7: Comparison of experimental and numerical results. Distribution of the contact duration, $P(\Delta t)$, (a) and distribution of the time interval between consecutive contacts, $P(\tau)$, (b) of experimental and numerical data with $p = 0.1$. The results are the average of ten runs for each type of test, the experimental data are represented by blue circles, while the numerical ones are represented by green triangles. The system parameters are fixed as: $t_c = 700$ ms, $L_{y_1} = 40$ cm, $L_{y_2} = 60$ cm, $T = 10$ min, $t_f = 2$ s, $v = 6$ cm/s, $N = 6$, $\langle \mathbf{a} \rangle = 0.77$. For numerical simulations, we used $\Delta h = 2$ s, $\delta h = 100$ ms, and $K = 300$, which corresponds to $T = K\Delta h = 10$ min.

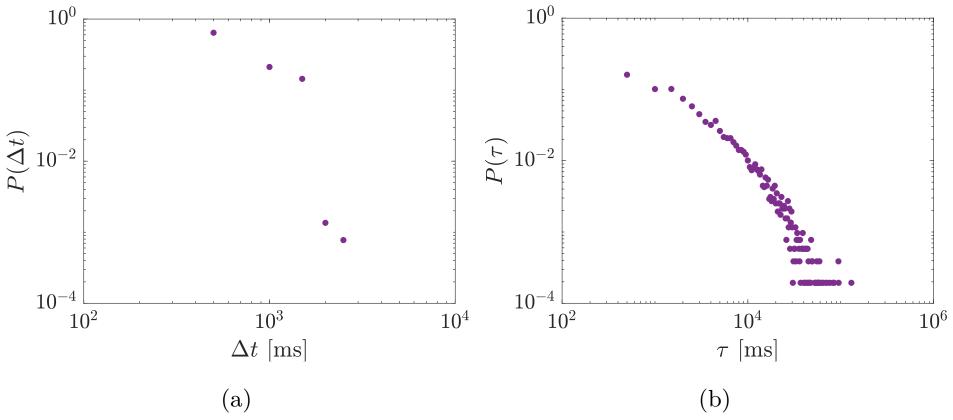


Figure 8.8: Experimental results. Distribution of the contact duration, $P(\Delta t)$, (a) and distribution of the time interval between consecutive contacts, $P(\tau)$, (b). The results are the average of ten runs with the following parameters: $t_c = 500$ ms, $L_{y_1} = 40$ cm, $L_{y_2} = 60$ cm, $\delta t = 100$ ms, $T = 10$ min, $t_f = 2$ s, $v = 6$ cm/s, $t_r \in [100 \text{ ms}, 1 \text{ s}]$, $N = 6$, $\langle \mathbf{a} \rangle = 0.77$.

In this chapter, we have introduced a robotic implementation of the attractiveness-based model for face-to-face interaction dynamics. The model considers elementary mechanisms of interactions that are locally implemented in each robot in a fully distributed and decentralized manner. Our results demonstrate the robustness of the approach despite the real-world factors that are usually neglected in numerical simulations.

In the next chapter, we will use the same robotic platform (Elisa-3 robot) to validate the mathematical model for synchronization of moving chaotic agents, discussed in [23]. Besides local communication, we will show another communication system able to overcome the limits of local communication.

Chapter 9

Synchronization of moving chaotic robots

In this chapter, we show a robotic system ruled by the interaction mechanisms of the mathematical model for synchronization of moving chaotic agents discussed in [23]. Our aim is to provide a proof-of-concept demonstrating the applicability of the theoretical model in a real-world scenario where factors commonly neglected in numerical simulations, such as parameter mismatches, noise, message loss, and deviations from planned movements, are present. The experimental setup is based on a team of Elisa-3 robots, each of them carrying an internal variable with chaotic dynamics. We contrast our results with a theoretical analysis based on the MSF and show that the mismatches and non-idealities of the system do not hamper the synchronization of the units.

9.1 Synchronization in networks of mobile oscillators

Before introducing the model of coupled mobile oscillators, we first briefly recall the notion of proximity graph and temporal proximity graph [32]. In proximity graphs, also known as random geometric graphs, the nodes are distributed uniformly in a random way in a two-dimensional Euclidean space and connected to each other if their relative distance is smaller than a given threshold, namely the interaction radius r. Let us indicate with

$\mathbf{y}_i = [y_{i,1}, y_{i,2}]^T$ $(i = 1, \ldots, N)$ the positions of the N nodes of the proximity graph in a two-dimensional rectangular space of size $L_{y_1} \times L_{y_2}$, and let us indicate with a_{ij} $(i, j = 1, \ldots, N)$ the coefficients of the $N \times N$ adjacency matrix of the graph, then

$$a_{ij} = 1 \Leftrightarrow \|\mathbf{y}_i - \mathbf{y}_j\| \leq r \tag{9.1}$$

and $a_{ij} = 0$ otherwise.

Temporal proximity graphs extend the notion of proximity graphs to account for time-varying links [32]. In more detail, nodes are now agents able to move in time, and therefore their position and neighborhood can change in time. Let $\mathbf{y}_i(t) = [y_{i,1}(t), y_{i,2}(t)]^T$ with $i = 1, \ldots, N$ indicate the agent positions at time t, and let us extend the rule for linking two nodes in the time-varying case. We consider two agents connected at time t if, at that time, their distance is less than the interaction radius r. In this way, we obtain a time-varying matrix $\mathrm{A}(t)$ whose coefficients are given by:

$$a_{ij}(t) = 1 \Leftrightarrow \|\mathbf{y}_i(t) - \mathbf{y}_j(t)\| \leq r \tag{9.2}$$

and $a_{ij}(t) = 0$ otherwise. Notice that temporal proximity graphs represent a class of temporal networks, as the model is fully specified only when the rule of motion for the agents is given. The model is particularly suited for agents equipped with limited sensing/communication capabilities, where r represents the maximum distance at which the communication system between robots can operate [60].

Let us now illustrate the model of coupled mobile oscillators at the basis of our robotic implementation. Following [23], we now associate a dynamical state $\mathbf{x}_i(t) \in \mathbb{R}^n$ to each agent $i = 1, \ldots, N$ of the system. The evolution of these variables is ruled by the dynamical equations of a nonlinear oscillator that is coupled with the other agent variables according to a temporal proximity graph:

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i) + \sigma \sum_{j=1}^{N} a_{ij}(t)\mathrm{H}(\mathbf{x}_j - \mathbf{x}_i) \tag{9.3}$$

for $i = 1, \ldots, N$. Here, $\mathbf{f}$ is the uncoupled dynamics of the dynamical oscillator, $\mathrm{H} \in \mathbb{R}^{n \times n}$ is the inner coupling matrix, and σ the coupling strength. In the model, the evolution of the dynamical variables of the oscillators is influenced by the agent motion through the matrix $\mathrm{A}(t)$. In this way, the motion type and characteristics influence the dynamical process, while no interaction from the dynamical process to the motion is considered, as, instead, occurs in swarmalators systems [70]. In the original work [23], agents are let move as random walkers in a planar space of size L and periodic boundary conditions. At each step of the random walk, each agent is moving with a direction of motion $\theta_i(t)$ drawn from a uniform distribution in $[0, 2\pi[$, and fixed velocity modulus v. In addition, with a given probability, named jumping probability, agents were allowed to perform jumps into fully random positions of the plane.

In view of the robotic implementation, the model that we consider is slightly different. First, rather than periodic boundary conditions we assume that, at the boundaries of the arena, agents turn in random directions to avoid hitting the walls delimiting the space where they move. Second, we focus on the case of zero jumping probability, as its effect is equivalently reproduced by using a large enough agent velocity v. Taking into account these considerations, the dynamical equations for the update of the agent positions become:

$$\mathbf{y}_i(t_{k+1}) = \mathbf{y}_i(t_k) + \mathbf{v}_i(t_k)\Delta h \tag{9.4}$$

where $i = 1, \ldots, N$, $\mathbf{v}_i(t_k) = ve^{\mathrm{i}\theta_i(t_k)}$ and $\Delta h = t_{k+1} - t_k \ \forall k$. The heading of agent i, $\theta_i(t_k) = \eta_i(t_k)$, is updated at each time step t_k through $\eta_i(t_k)$, an independent random variable drawn with uniform probability in $[-\pi, \pi]$. In the mathematical model, Eqs. (9.4) are used to simulate the motion of robots/agents considered as mass-less points. In the experiments, the robot velocity is controlled to realize random walk, by fixing the modulus and randomly updating the headings.

From the positions of the agents, the matrix $\mathrm{A}(t)$ can be obtained using Eq. (9.2). Specifically, since the motion direction is updated at time intervals of length equal to Δh, in Eq. (9.3) we consider that $\mathrm{A}(t) = \mathrm{A}(t_k)$ for $t_k \leq t < t_{k+1}$. In addition, the neighborhood of agent i at time t_k can be calculated as

$$\mathcal{N}_i(t_k) = \{j : \|\mathbf{y}_j(t_k) - \mathbf{y}_i(t_k)\| < r\} \tag{9.5}$$

from which we get $\mathcal{N}_i(t) = \mathcal{N}_i(t_k)$ for $t_k \leq t < t_{k+1}$.

As result of the time-varying interactions that arise during agent motion, the coupled oscillators of Eqs. (9.3) may synchronize, that is, their state variables may converge to a common trajectory where $\mathbf{x}_1(t) = \mathbf{x}_2(t) = \ldots = \mathbf{x}_N(t)$. The level of synchronization of the system can be evaluated through the synchronization error, defined as:

$$\delta(t) = \frac{1}{N}\sum_{i=1}^{N}\sum_{\substack{j=1,\\ j\neq i}}^{N} \|\mathbf{x}_i(t) - \mathbf{x}_j(t)\| \tag{9.6}$$

The system synchronizes if $\lim_{t\to+\infty} \delta(t) = 0$. In [23] the conditions for the onset of synchronization in this system have been derived by using an approach combining together the fast switching method and the local analysis of stability via the master stability function. A key role is played by the parameter $p_{int} = \frac{\pi r^2 \rho}{N}$, representing the probability that two agents interact. In particular, this parameter can be changed by varying the agent density $\rho = N/(L_{y_1}L_{y_2})$ or the radius r. Two scenarios, exemplified by the following case studies, emerge for synchronization. To illustrate them, we fix as the nodal dynamics the Rössler oscillator [85], i.e., a paradigmatic chaotic system for the study of synchronization, that, by varying the way in which the units are coupled, can display the different scenarios we are interested in. In our robotic implementation each of the Rössler oscillators is implemented in a robot of the team, which numerically integrates the oscillator equations and memorizes its state variables.

Consider first the following system of coupled Rössler oscillators:

$$\begin{cases} \dot{x}_{i,1} = -x_{i,2} - x_{i,3} \\ \dot{x}_{i,2} = x_{i,1} + a x_{i,2} + \sigma \sum\limits_{j=1}^{N} a_{ij}(t)(x_{j,2} - x_{i,2}) \\ \dot{x}_{i,3} = b + x_{i,3}(x_{i,1} - c) \end{cases} \tag{9.7}$$

with $i = 1, \ldots, N$. The parameters a, b, and c are set to $a = 0.2$, $b = 0.2$, $c = 7$, such that the uncoupled dynamics is chaotic. When coupled through a network with static links, this system has a master stability function of type II, which means that synchronization may be achieved for any network with large enough coupling [40]. In the case of mobile oscillators, the analysis carried out in [23] leads to the conclusion that synchronization is obtained when the agent density ρ is such that $\rho > \frac{\nu^*}{\pi r^2 \sigma}$, where $\nu^* = 0.157$. This result is particularly interesting as a similar density-dependent behavior is found in some real systems, such as yeast cell populations [16].

Consider now the following system of Rössler oscillators, where, at variance of Eqs. (9.7), coupling occurs through the first variable:

$$\begin{cases} \dot{x}_{i,1} = -x_{i,2} - x_{i,3} + \sigma \sum_{j=1}^{N} a_{ij}(t)(x_{j,1} - x_{i,1}) \\ \dot{x}_{i,2} = x_{i,1} + a x_{i,2} \\ \dot{x}_{i,3} = b + x_{i,3}(x_{i,1} - c) \end{cases} \tag{9.8}$$

where the parameters are set as above. In this case, when coupled through a network with static links, the system has a master stability function of type III, which indicates that synchronization may eventually be achieved only if some conditions on the spectrum of their Laplacian matrix and on the coupling strength σ are satisfied [40]. For mobile oscillators, synchronization in systems with type III master stability function requires the following condition on the agent density: $\rho \in [\frac{\nu_1^*}{\pi r^2 \sigma}, \frac{\nu_2^*}{\pi r^2 \sigma}]$, where $\nu_1^* = 0.186$ and $\nu_2^* = 4.614$.

Notice that, although the condition for synchronization in both cases is expressed as a function of the agent density, also the agent velocity is an important parameter of the system. In fact, for low values of the agent velocity, the system deviates from the assumption of fast switching and the previous results are no longer valid. Under these circumstances, the synchronization in the system can be analyzed using the framework discussed in [13].

9.2 Experimental setup

The experiments are conducted using the Elisa-3 robots and the experimental setup illustrated in Sec. 6.1. Specifically, for this implementation, the size of the arena is varied to change the agent density, $\rho = N/(L_{y_1} L_{y_2})$.

Robots are controlled to move as random walkers, while always checking for potential obstacles (either the walls of the arena or other units of the team). Each step of the random walk is realized as described in Sec. 8.2, thus by first rotating the robot in a randomly selected direction (left or right), and second by moving forward for a fixed duration of time $t_f = 2$ s at a constant velocity $v = 6$ cm/s. If, during this motion, an obstacle is encountered, the robot heading is changed, and then the robot proceeds straight in the new direction for the remaining time up to t_f. Notice that in this case, unlike the implementation described in Chapter 8, when the robot encounters another unit, it does not stop its motion but simply changes its direction.

During their motion, the robots iterate the calculation of the state variables of the chaotic oscillators associated with them, namely Eqs. (9.3). The calculation is carried out in a distributed way, as each robot i only integrates, in single precision, the equations related to its state variables $\mathbf{x}_i$. To this aim, either a fourth-order Runge-Kutta method or a forward Euler method with a fixed step size equal to $\Delta t = 0.025$ has been used (here time is expressed in the arbitrary time unit of the dynamical evolution of the chaotic oscillators). The robots provide a new value of the state variables after an interval of time equal to t_s. In the following, we indicate as $\mathbf{x}_i^h$, with $h = 1, 2, \ldots$, the calculated samples. At each iteration h, namely at time $t_h = ht_c$, the sample $\mathbf{x}_i^h$, which represents the value of the state variables after a period of time equal to $h\Delta t$, will be available. The value of t_s is, thus, a limiting factor for the time required to integrate the chaotic trajectory of the oscillators associated to the robots. Robots send the values of all their state variables to the computer at each $t_p = 100$ ms. In this way, experimental data are collected and available for post-processing, including the computation of the level of synchronization achieved by the multi-agent system.

9.3 Communication among robots

Interactions among robots occur through either the local communication system or the vision-based virtual communication, yielding two different implementations of the system of coupled mobile oscillators. In the first case, using the local communication system, a fully decentralized and autonomous team of robots is obtained. In the second case, the communication among robots is not physical, but virtually simulated through the vision-based localization and the central communication with the computer via the radio link. The purpose of this second implementation is to overcome the limitations of the local communication system on board of the Elisa-3 robots and prove the general validity of the proposed method.

9.3.1 Local communication system

In the case of the fully decentralized implementation, the communication among robots occurs through the local communication system on board of each Elisa-3 robot. As already discussed in Sec. 6.1, this communication system has no transmission/reception queue and allows to send one-byte packets with a throughput of about 1 byte/s. A single scalar signal, encoding the coupling variable in a one-byte packet, is sent via this communication system. In more detail, each robot continuously checks if new messages are received to detect nearby units and, when it detects another unit, it allocates a sufficiently large window of time to retrieve the information on the coupling variable of the other units. Therefore, in contrast to the mathematical model presented in Sec. 9.1, which first computes the positions of the agents and then uses Eq. (9.5) to calculate each agent neighborhood, the robotic implementation takes into account all data received from nearby units within time intervals of length $t_s = 700\ ms$. Notice that the value of t_s is chosen with the same criteria adopted in Sec. 8.2. Assuming that robot i has correctly received the data from all its neighbors during the time interval of length t_s, the neighborhood of agent i at time step t_h is, thus, given by:

$$\mathcal{N}_i(t_h) = \{j : \|\mathbf{y}_j(t') - \mathbf{y}_i(t')\| < r \text{ for some } t' \in [t_{h-1}, t_h]\} \tag{9.9}$$

where r is set to 10 cm, which corresponds to the maximum communication distance achievable through the use of the IR sensors of the local communication system.

However, since communication is not perfect and some of the messages may be lost, the actual neighborhood of each robot is a subset of the ideal one, namely $\tilde{\mathcal{N}}_i(t_h) \subseteq \mathcal{N}_i(t_h)$. Consequently, unlike the original mathematical model where interactions are mutual and the adjacency matrix is symmetric, in the robotic implementation, when a unit receives a message from another robot, it cannot be taken for granted that the latter also receives the message from the former, and, in general, the adjacency matrix describing interactions is no longer symmetric.

Algorithm 3: Implementation based on the local communication system: robot i

```
Parameter      : $a$, $b$, $c$, $\sigma$, $t_f$, $t_s$, $\Delta t$
Initialization : $\mathbf{x}_i^h = \mathbf{rand}$, $t$ = getCurrentTime
while true do
    Random walk with obstacle avoidance
    Send $\mathbf{x}_i^h$ to the central station
    Broadcast $x_{i,1}^h$ to neighboring units via the local communication system
    Check IR sensors
    if messageReceivedFromRobot == true then
        Update $\mathcal{N}_i(t_h)$ as Eq. (9.9)
    end
    if (getCurrentTime − t) $\geq t_s$ then
        Compute the coupling term $\gamma_i(t_h) = \sigma \sum_{j \in \mathcal{N}_i(t_h)} (x_{j,1} - x_{i,1})$
        Update $\mathbf{x}_i^h$ through the integration algorithm
        Reset the coupling term
        $t$ = getCurrentTime
    end
end
```

Algorithm 3 summarizes the main steps of the robot control law in the case of the implementation based on the local communication system. In illustrating the algorithm, we consider the case in which the evolution of the robot state variables is described by Eq. (9.8). The steps are the same when the dynamics is ruled by Eq. (9.7), with the exception that each robot now broadcasts $x_{i,2}^h$ (rather than $x_{i,1}^h$) and uses a coupling term given by $\gamma_i = \sigma \sum\limits_{j \in \mathcal{N}_i(t_h)} (x_{j,2} - x_{i,2})$ (rather than $\gamma_i = \sigma \sum\limits_{j \in \mathcal{N}_i(t_h)} (x_{j,1} - x_{i,1})$).

Notice that the communication between the computer and the robots is only used to collect the experimental data, so that Algorithm 3 is fully decentralized.

9.3.2 Virtual inter-robot communication system

In the second implementation the robots do not communicate in a direct way, but through the central communication via the radio link. The robots periodically send the value of their status to the central station, namely the computer. In addition, the robots are tracked through the IR camera mounted above the arena, in order to determine their positions. From them, the neighborhood of each robot is calculated by the computer that sends the appropriate information about the state of the neighbors to each robot of the team. In more detail, the computation of the neighborhood occurs in a way similar to the mathematical model discussed in Sec. 9.1, namely by computing:

$$\mathcal{N}_i(t_h) = \{j : \|\hat{\mathbf{y}}_j(t_h) - \hat{\mathbf{y}}_i(t_h)\| < r\} \tag{9.10}$$

where $\hat{\mathbf{y}}_k(t_h)$ is the position of each robot k (with $k = 1, \ldots, N$) detected by the tracking software at time step t_h. In contrast to the fully decentralized implementation of Sec. 9.3.1, where r is constrained by the local IR-based communication system, here r is a free parameter. The importance of varying this parameter is twofold. On the one hand, it allows to theoretically study the effect of the radius on synchronization in the coupled mobile oscillators. On the other hand, in phase of design and development of a more close-to-the-market prototype, it can be used to determine the requirements of the local communication system to operate the robots.

Using the virtual inter-robot communication system results in all interactions being mutual, such that the adjacency matrix is symmetric. In addition, the time window for neighborhood detection is no longer constrained by the limited throughput of the local communication system. Based on the considerations discussed in Chapter 7, to ensure that each robot properly receives the coupling term from the computer at each time step t_h, t_s has been selected equal to 100 ms.

As discussed in Chapter 7, the tracking system requires calibration. First, a linear calibration of the 2D vision system is performed to convert the robot's position from pixel to real-world coordinates. Then the camera settings including the brightness, contrast, and saturation are adjusted. Finally, at the beginning of each experiment, each robot rotates one at a time, allowing the tracking software to match the moving particle with the corresponding robot ID. Algorithms 4 and 5 summarize the tasks performed by each robot and by the computer in this implementation.

9.4 Results

In this section, we discuss the results of a series of experiments carried out using the setup described in Sec. 9.2. To quantitatively evaluate syn-

Algorithm 4: Implementation based on the virtual inter-robot communication system: robot i

```
Parameter       : $a$, $b$, $c$, $t_f$, $t_s$, $\Delta t$
Initialization  : $\mathbf{x}_i^h = \mathbf{rand}$, $t$ = getCurrentTime
1  while true do
2      Random walk with obstacle avoidance
3      Send $\mathbf{x}_i^h$ to central station
4      if A message from the central station is received then
5          Get the value of the coupling term $\gamma_i$ by decoding the message
6      end
7      if (getCurrentTime $-$ t) $\geq t_s$ then
8          Update $\mathbf{x}_i^h$ through the integration algorithm
9          $t$ = getCurrentTime
10     end
11 end
```

chronization of the oscillators associated with the robots, we consider the temporal average of the synchronization error $\delta(t)$ defined in Eq. (9.6) in the last part of the experiment, namely we calculate $\langle\delta\rangle = \frac{1}{\Delta T}\int_{\frac{4}{5}T}^{T}\delta(t)dt$, where T is the overall duration of each experiment.

9.4.1 Experiments with the local communication system

We start by illustrating the experiments with the implementation employing the local communication system described in Sec. 9.3.1. First, we show the effect of the coupling strength σ on the system dynamics, and then discuss how the robot density affects the synchronization error.

In all the experiments, the robots perform a random walk at a velocity of $v = 12$ cm/s, with $t_f = 2$ s and $t_r \in [0.1\text{ s}, 1\text{ s}]$. Moreover, due to the low throughput of the local communication system, t_s has been selected as $t_s = 0.7$ s. This is a quite large value which constrains the duration of the whole experiment, set to $T = 2400$ s, in order to observe a sufficiently large number of oscillations of the chaotic systems associated to the robots.

Preliminarily to a systematic analysis of the system behavior, we discuss two experiments with two different values of σ ($\sigma = 0$ and $\sigma = 2$), while maintaining the robot density fixed, using $N = 6$ robots and an arena of dimensions $L_{y1} = L_{y2} = 30$ cm. In these experiments, the dynamics of the

Algorithm 5: Implementation based on the virtual inter-robot communication system: central station

```
Parameter       : $\sigma$, $N$, $r$, $t_p$, $t_s$
Initialization  : $t$ = getCurrentTime
1  Perform linear calibration
2  while true do
3      if A message from robots is received then
4          Get $\mathbf{x^h}$ by decoding the message
5      end
6      if (getCurrentTime − t) $\geq t_s$ then
7          Track the robot positions
8          for $i = 1 : N$ do
9              Compute $\mathcal{N}_i(t_h)$ as in Eq. (9.10)
10             Compute the coupling term $\gamma_i(t_h) = \sigma \sum_{j \in \mathcal{N}_i(t_h)} (x_{j,1} - x_{i,1})$
11             Send $\gamma_i$ to robot $i$
12         end
13         $t$ = getCurrentTime
14     end
15 end
```

oscillators associated to the robot are ruled by Eqs. (9.7). After proving here the viability of the approach with these dynamics, in the rest of the chapter, we will then consider only Eqs. (9.8), as a master stability function of type III represents the most challenging and general case.

Fig. 9.1(a) illustrates the time evolution of the state variables $x_{i,1}(t)$ for $\sigma = 0$. Under this condition, the dynamical oscillators are decoupled from each other, regardless of robot motion, such that each oscillator evolves without synchronizing with the ones at the other units. Accordingly, the value of the average synchronization error is large, $\langle \delta \rangle = 11.6$.

The time evolution of the state variables $x_{i,1}(t)$ in the second experiment, where the coupling strength is fixed to $\sigma = 2$, is illustrated in Fig. 9.1(b). After a transient, the state variables converge to a common trajectory displaying a small synchronization error, $\langle \delta \rangle = 0.99$.

Next, we contrast the experimental results with the behavior of the mathematical model discussed in Sec. 9.1. In the model, rather than performing the motion step of the random walk in a single time interval, we consider a smaller step size $t_q < \tau_M$ and check after each interval of fixed

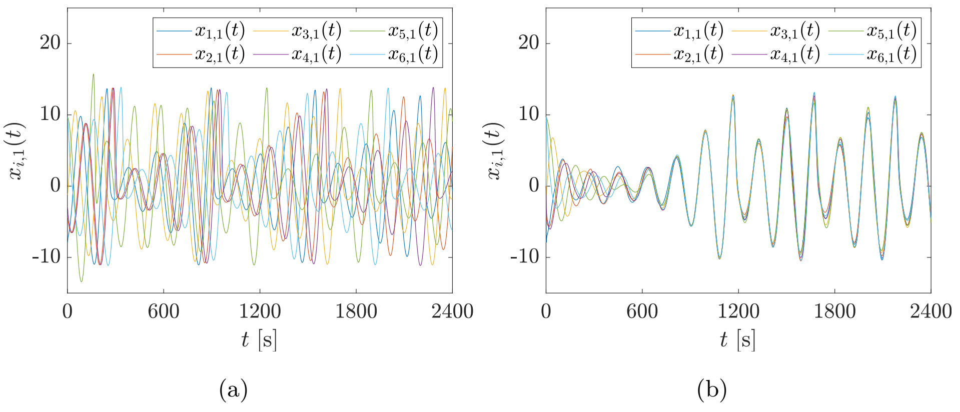
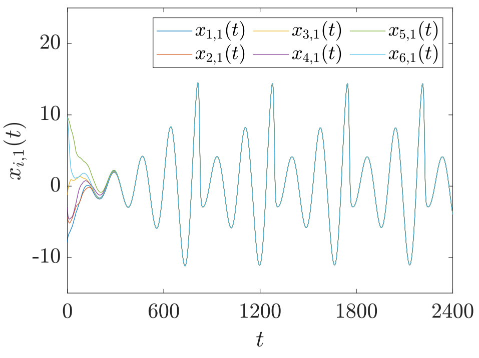


Figure 9.1: Implementation based on the local communication system: temporal evolution of the state variables $x_{i,1}(t)$ (experimental results on a team of $N = 6$ Elisa-3 robots). (a) $\sigma = 0$. (b) $\sigma = 2$. The other parameters are fixed as $L_{y_1} = L_{y_2} = 30$ cm, $t_s = 0.7$ s, $N = 6$, $t_f = 2$ s, $v = 6$ cm/s, $T = 2400$ s.

length t_q whether during its motion the agent finds an obstacle, which can be one of the arena walls or another unit to be considered as a neighbor. If no obstacle is encountered, then the full motion step of length $v\tau_M$ is performed; otherwise, the agent stops, rotates to a random direction and, then, continues its random walk.

For the purpose of comparison, we fix the model parameters so that the agents move with a velocity of $v = 0.6$ at each time step $t_q = 0.1$, whereas $\tau_M = 2$. The interaction radius used in Eq. (9.9) is fixed as $r = 10$ and the parameters for the integration of Eqs. (9.3) as $\Delta t = 0.025$ and $t_s = 0.7$. In addition, to emulate the characteristics of the local communication system, in the calculation of the coupling terms the values of the state variables of the neighboring units are encoded in 1 byte. Finally, a further parameter, indicated as p and representing the reception probability of a message, is introduced in the model to account for the possibility that a message is lost during communication.

The results of the numerical simulations are illustrated in Fig. 9.2 which shows the case $\sigma = 2$ for two different values of p: $p = 1$ in Fig. 9.2(a) representing the ideal scenario where no messages are lost, and $p = 0.7$ in Fig. 9.2(b) where the value of the reception probability has been empirically



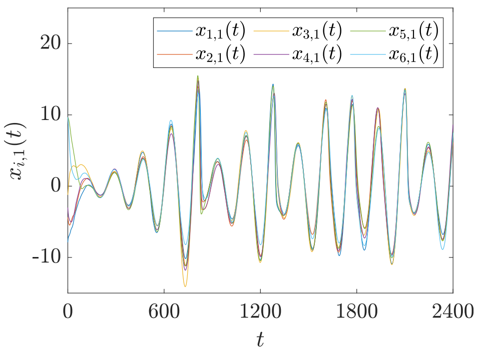


Figure 9.2: Implementation based on the local communication system: temporal evolution of the state variables $x_{i,1}(t)$ obtained for $\sigma = 2$ (numerical results). (a) $p = 1$. (b) $p = 0.7$. The other parameters are fixed as $L_{y_1} = L_{y_2} = 30$, $t_s = 0.7$, $N = 6$, $\tau_M = 2$, $t_r = 0$, $T = 2400$, $v = 0.6$, $t_q = 0.1$.

tuned to match the experimental situation. When $p = 1$, we obtain a synchronization error equal to $\langle\delta\rangle = 0$, whereas, when $p = 0.7$, $\langle\delta\rangle = 1.56$. These results suggest that the non-zero synchronization error observed in the experiment of Fig. 9.1(b) can be attributed to the loss of messages during transmission between robots. Despite the non-idealities of the local communication system, however, the chaotic oscillators associated with the robots still reach a quite small synchronization error, demonstrating the robustness of the interaction mechanisms.

We now move to illustrate the effect of different values of the robot density. To tune this parameter, the dimensions of the arena, namely L_{y_1} and L_{y_2}, have been changed while keeping fixed the number of robots to six, i.e., $N = 6$. For these experiments, the dynamics of the chaotic oscillators are given by Eqs. (9.8), with the coupling strength set to $\sigma = 2$.

Figure 9.3 shows the synchronization error $\langle\delta\rangle$ vs. agent density ρ obtained from a series of experiments (black squares) contrasted with the results of numerical simulations for $p = 1$ (depicted in blue) and $p = 0.7$ (depicted in red). For the numerical simulations, the average error across 50 trials is presented for each robot density value, with the shaded area proportional to the corresponding standard deviation. We notice that in the

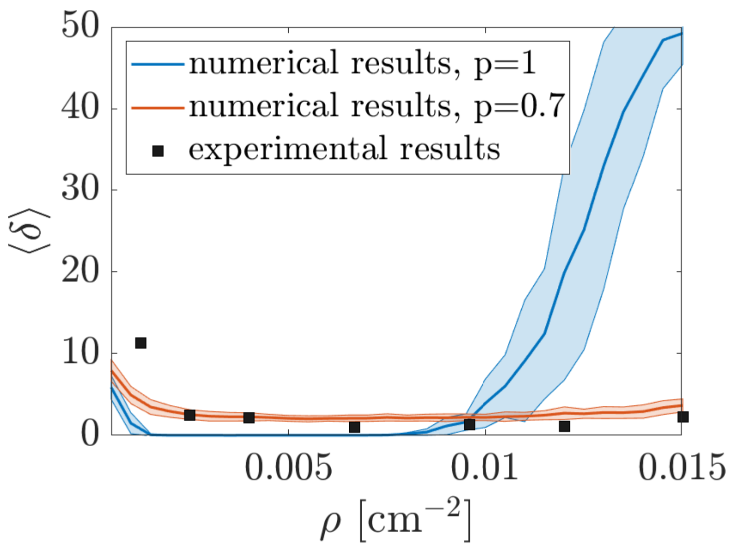


Figure 9.3: Implementation based on the local communication system: synchronization error $\langle\delta\rangle$ vs. robot density ρ. Parameter values: $t_s = 0.7$ s, $t_f = 2$ s, $v = 6$ cm/s, $T = 2400$ s, $\sigma = 2$.

ideal scenario of no loss of messages during robot interaction ($p = 1$) there are two transitions (from incoherent behavior to synchronization and from synchronization back to a disordered state), while in the experiments and in the numerical simulations with $p = 0.7$ only the first transition appears. We conclude that the loss of messages has a threefold effect: a shift of the point where the first transition occurs, the absence of a second transition in the considered range of values of the agent density, and a small, but non-zero, synchronization error.

9.4.2 Experiments with the virtual inter-robot communication system

In this section, we illustrate the results obtained using the virtual inter-robot communication system, as described in Sec. 9.3.2. In particular, we first discuss the effect of the robot density on the synchronization error, and then that of the interaction radius.

As in Sec. 9.4.1, the parameters regulating the robot random walk are set to $v = 12$ cm/s, $t_f = 2$ s and $t_r \in [0.1\text{ s}, 1\text{ s}]$. The dynamics of the chaotic oscillators associated to the robots is here given by Eq. (9.8) with $\sigma = 2$, to

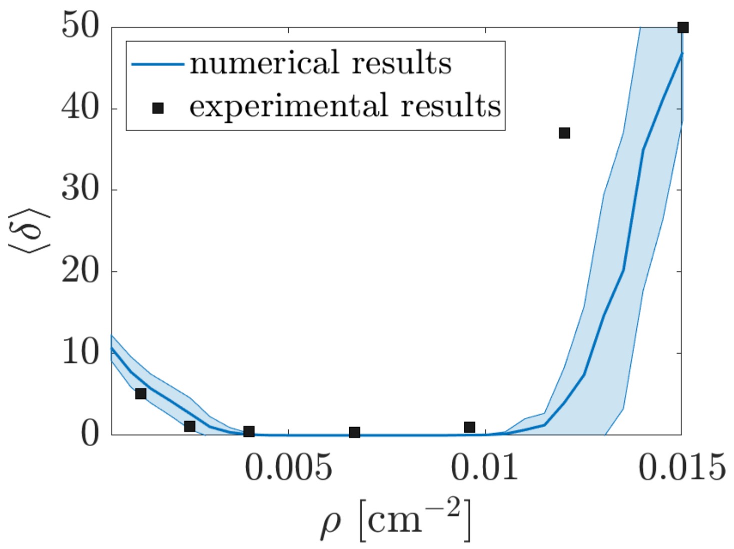


Figure 9.4: Implementation based on the virtual inter-robot communication system: synchronization error $\langle\delta\rangle$ vs. robot density ρ. Parameter values: $t_s = 0.1$ s, $t_f = 2$ s, $v = 6$ cm/s, $T = 600$ s, $\sigma = 2$.

study the dynamic behavior characterizing a system with a master stability function of type III. Finally, t_s is selected according to the considerations discussed in Sec. 9.3.2, that is, $t_s = 0.1$ s $> t_d$.

In this set of robot experiments, similarly to the experimental campaign illustrated in Sec. 9.4.1, we adjust the robot density by changing the dimensions of the arena, without changing the number of robots N. During the experiments, each lasting $T = 600$ s, the interaction radius is kept fixed to $r = 10$ cm. Notice that, in this case, a smaller duration T for the experiments can be selected in virtue of the smaller value of t_s, which allows to have a longer trajectory of the dynamical oscillators associated to the robots in a shorter time window.

Also in this case, the experimental results are compared with the outcomes of the mathematical model simulating the scenario of Sec. 9.3.2. The model parameters have been set such that the units perform a movement with a velocity of $v = 0.6$ at each time step $t_q = 0.1$. Eqs. (9.3) are integrated with $\Delta t = 0.025$, and the neighborhood is calculated from Eq. (9.10). The remaining parameters are set as in the robot experiments, namely $\tau_M = 2$ and $t_s = 0.1$. In contrast to the scenario discussed in Sec. 9.4.1, here there

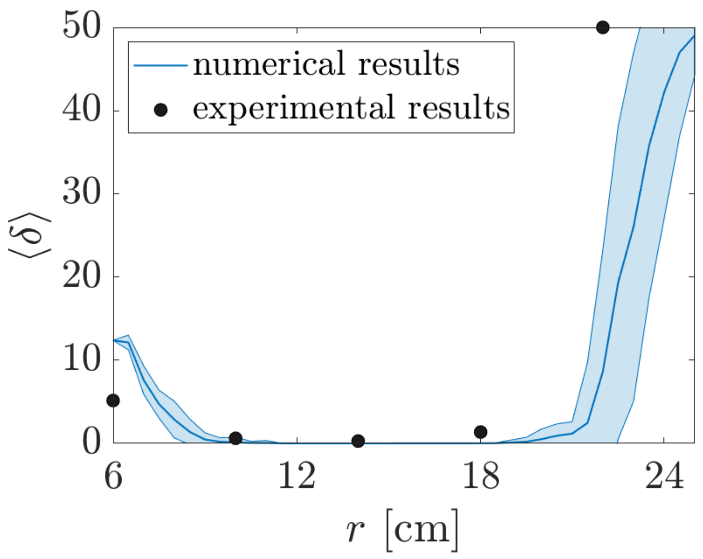


Figure 9.5: Implementation based on the virtual inter-robot communication system: synchronization error $\langle\delta\rangle$ vs. interaction radius r. Parameter values: $L_{y_1} = L_{y_2} = 40$ cm, $t_s = 0.1$ s, $N = 5$, $t_f = 2$ s, $v = 6$ cm/s, $T = 420$ s, $\sigma = 2$.

is no need to introduce a parameter to account for the loss of messages, as, using the virtual inter-robot communication system, this appears to be a negligible factor.

The results are illustrated in Fig. 9.4, which shows the experimental synchronization error $\langle\delta\rangle$ vs. agent density ρ (black squares) contrasted with the results of the numerical simulations (the blue curve represents the average synchronization error across 50 trials for each value of the robot density, while the shaded area is proportional to the standard deviation). The analysis of these results reveals two important differences with the case dealt with in Sec. 9.4.1. First, in the range of values of ρ where synchronization is attained, the synchronization error is smaller than the previous case: for instance in correspondence of $\rho = 6.7 \times 10^{-3}$ cm^{-2} we obtain $\langle\delta\rangle = 0.32$. Second, the two transitions characterizing synchronization in class III systems are clearly visible in Fig. 9.4. Altogether, these findings reinforce the conclusion that avoiding or at least reducing the loss of messages in inter-robot communication is important to closely reproduce the behavior of the theoretical model of synchronization in moving agents.

Finally, we illustrate the effect of the interaction radius r. To this purpose, we consider a team of $N = 5$ robots moving in an arena of dimensions $L_{y_1} = L_{y_2} = 40$ cm, corresponding to $\rho = 3.1 \times 10^{-3}$ cm^{-2}, that interact with a coupling strength $\sigma = 2$ for a time window $T = 420$ s.

The results are illustrated in Fig. 9.5, which shows the synchronization error $\langle \delta \rangle$ vs. the interaction radius r for the robot experiments (black circles) and the numerical simulations (the continuous blue line is the average of 50 trials for each value of r, while the shaded area is proportional to the standard deviation). Also in this case, the synchronization error first decreases, reaches a region where the oscillators are fully synchronized, and then increases again.

Overall, the experiments described in this section yield the conclusion that synchronization can be either induced or hampered by tuning the interaction radius r or the robot density ρ. Such an experimental observation corroborates the theoretical expectations, as, under the assumption of very fast motion [23], both r and ρ affect the probability that two robots interact at some time. Specifically, the interaction occurs if the robots lie in the area delimited by the radius r of their communication system. Thus, the probability of interaction in a $L_{y_1} \times L_{y_2}$ arena is given by $p_{int} = \frac{\pi r^2}{L_{y_1} L_{y_2}}$. Taking into account that agent density is $\rho = N/(L_{y_1} L_{y_2})$, one gets the well-known expression $p_{int} = \frac{\pi r^2 \rho}{N}$. Therefore, synchronization ultimately depends on the probability of interaction, p_{int}, as expected.

In this chapter, we have introduced a robotic system for the study of synchronization of chaotic oscillators associated with mobile agents, starting from the theoretical model introduced in [23]. This system can be useful both for studying the applicability of the mathematical model in a real system where there are also real-world factors often overlooked in simulations and for performing validations of the fundamental interaction mechanisms in real complex systems that have some uncontrollable parameters. We have proposed two different implementations of the mechanisms of interaction among the robots. The first was the local communication system on board the robots, which allows the control law to be fully distributed. However, this approach has the drawback of limited bandwidth. Indeed, while the robots were able to reach synchronization, the full repertoire of dynamic behaviors of the original model could not be observed. To demonstrate that a local communication system with a larger bandwidth can overcome the observed limitations, a second implementation based on a virtual communication system among the robots has been investigated. This system simulates interactions among robots through centralized communication with a computer via a radio link and a camera monitoring the arena where the robots move. Using this approach, we have been able to reproduce the dynamic behaviors of the original model. Additionally, we have shown that this method allows the communication radius to become a controllable parameter, proving that the key factor for synchronization is the interaction probability.

In the next chapter, we will focus on the phenomenon of the response to synchronization which involves two types of agents. The first type of agent is capable of synchronizing directly with other units through their interactions. In contrast, the second type of agent requires interaction with the first type to achieve synchronization among its kind.

Chapter 10

Response to synchronization in a robot team

This chapter concludes the experimental part of this thesis work. Here we validate the mathematical model concerning the response to synchronization of fireflies using a team of e-puck2 robots. Validating the model's applicability is crucial, as it shows that units that are not able to reach synchronization on their own can still achieve synchronization through interactions with agents that are capable of doing so. Based on the findings discussed in [22], we divide the population into males and females with different characteristics.

10.1 Response to synchronization in fireflies

The flashing of fireflies is a paradigmatic example of synchronization involving many interacting individuals. However, a closer analysis of the firefly system reveals distinct oscillatory behaviors between males and females, indicating that the system is composed of two different types of units. In this context, the concept of *response to synchronization* is introduced. This phenomenon is observed in the *Photinus carolinus* firefly species [62], where real females demonstrate an enhanced response to synchronous signals emitted by virtual males that share the same oscillatory characteristics as real ones.

In [80], both male and female fireflies are modeled using oscillators, with each male represented by a bursting oscillator (BO) and each female by a non-bursting oscillator (NBO). These oscillators show an active phase whose duration is affected by the interaction with other units and a resting phase whose duration remains constant.

We begin by describing the dynamics of these oscillators in isolation. For NBOs, the active phase consists of a long charging phase lasting $T_{c,f}$, followed by a short discharging phase lasting $T_{d,f}$. In contrast, the active phase of the BOs is characterized by n_m bursts, each with a charging time $T_{c,m}$ and a discharging time $T_{d,m}$. Thus, the total duration of the active phase in a BO is $n_m(T_{c,m} + T_{d,m})$. Following the active phase, both BOs and NBOs enter a resting phase, lasting $T_{s,f}$ for NBOs and $T_{s,m}$ for BOs, in which the state variable is kept at a constant value V_l. The total duration of a complete cycle (active phase + resting phase) is denoted by $T_{p,m}$ for the BOs and $T_{p,f}$ for the NBOs. In both oscillators, the charging phase ends when the state variable $v_i(t)$ reaches the value $V_u = \frac{2}{3}V_M$, whereas the discharging phase ends when $v_i(t) = V_l = \frac{1}{3}V_M$, with V_M being a constant value. When the oscillator is isolated, its dynamics is described by the following equation:

$$\begin{aligned}
\dot{v}_i(t) &= \left[\frac{\ln 2}{T_{c,i}}\left(V_M - v_i(t)\right)\epsilon_i(t) - \frac{\ln 2}{T_{d,i}}\left(1 - \epsilon_i(t)\right)\right] y_i(t) \\
y_i(t) &= \sum_{k=0}^{M_i(t)} \left[H(t - kT_{p,i} - \Delta\Phi_i) - H(t - (k+1)T_{p,i} + T_{s,i} - \Delta\Phi_i)\right]
\end{aligned} \tag{10.1}$$

where $M_i(t)$ is an integer obtained from the integer division of t by $T_{p,i}$, H is the Heaviside step function, $\Delta\Phi_i$ is a phase delay, and the parameters $T_{c,i}$, $T_{d,i}$, $T_{s,i}$, and $T_{p,i}$ depend on whether the oscillator i is a BO or an NBO. Specifically, these parameters correspond to $T_{c,m}$, $T_{d,m}$, $T_{s,m}$, and $T_{p,m}$ for a BO, and to $T_{c,f}$, $T_{d,f}$, $T_{s,f}$, and $T_{p,f}$ for an NBO. The function $y_i(t)$ determines the phase in which the oscillator is: when $y_i(t) = 1$, the oscillator is in the active phase, whereas when $y_i(t) = 0$, it is in the silent phase. During the active phase, the dynamics of $v_i(t)$ is constituted by two terms: the first that rules the charging phase, and the second that regulates the discharging term. The contribution of these terms is controlled by the binary variable $\epsilon_i(t)$, which describes the current state of the oscillator. Specifically, when $\epsilon_i(t) = 0$, the oscillator is in the discharging phase and the unit emits a flash, whereas when $\epsilon_i(t) = 0$, the oscillator is in the charging or in the resting phase, during which no flash is emitted. The temporal evolution of $\epsilon_i(t)$ is given by:

$$\epsilon_i(t^+) = \epsilon_i(t) - \epsilon_i(t)H(v_i(t) - V_u)\epsilon_i(t) + (1 - \epsilon_i(t))H(V_l - v_i(t)) \quad (10.2)$$

which indicates that $\epsilon_i(t)$ switches to 0 when $v_i(t)$ reaches the upper threshold V_u, and switch to 1 when $v_i(t)$ reaches the lower threshold V_l.

In Fig. 10.1 shows the temporal behavior of $v_i(t)$ for both an isolated BO (Fig. 10.1(a)) and NBO (Fig. 10.1(b)) oscillator, together with the binary function $\epsilon_i(t)$ and the associated flashing event (Fig. 10.1(c) for BO and Fig. 10.1(d) for NBO) denoted by a rectangular mark displayed each time a flash is emitted.

When N the oscillators interact with each other, the equations describing the dynamics become the following:

$$\begin{aligned} \dot{v}_i(t) &= \left[\frac{\ln 2}{T_{c,i}}\left(V_M - v_i(t)\right)\epsilon_i(t) - \frac{\ln 2}{T_{d,i}}\left(1 - \epsilon_i(t)\right) + \gamma_i(t)\right] z_i(t) \\ z_i(t) &= H(t - \Delta\Phi_i) + \sum_{k=1}^{M_i'(t)} \left[H(t - t_{k,i} - T_{s,i}) - H(t - t_{k,i})\right] \end{aligned} \quad (10.3)$$

where $\gamma_i(t)$ is the coupling term, $z_i(t)$ is a function that determines the oscillator phase when coupled ($z_i(t) = 0$ during the resting phase, $z_i(t) = 1$ during the active phase) and $t_{k,i}$ is the time instant in which, for the $k-$th time, the following conditions are satisfied:

- $mod(\eta_i(t), n_i) = 0$
- $v_i(t) = V_l$
- $\epsilon_i(t) = 0$

with $\eta_i(t)$ being the number of peaks reached by $v_i(t)$ in the time interval $[0, t]$, and $n_i = n_m$ if i is a BO, 1 otherwise. The term $M'(t)$ indicates the number of times these three conditions are satisfied in the time interval $[0, t]$. Notice that in this case, the oscillator phase is not determined by the function $y(t)$ since, due to the coupling, the complete cycle duration of the oscillator no longer has a constant duration.

The coupling term is expressed as:

$$\gamma_i(t) = \theta_i\sigma_i \sum_{j=1}^{N} a_{ij}\left(1 - \epsilon_j(t)\right) , \quad (10.4)$$

where σ_i is the coupling strength, a_{ij}, $i, j = 1, \dots, N$, are the entries of the adjacency matrix of the interaction graph, assumed to be undirected and fully connected, and θ_i is a term that assumes the value of 1 if i is a BO, -1 otherwise, meaning that coupling term increases the discharging time if i is BO or the charging time if i is an NBO. Notice that the coupling is defined in terms of the variable $\epsilon(t)$, in particular when $\epsilon_j(t) = 0$, i.e., the j-th oscillator is flashing, the latter influences the i-th one.

In the mathematical model, two scenarios have been considered: one where the oscillators are stationary and interact through an unweighted adjacency matrix and another where the oscillators are mobile and interact through a weighted time-varying interaction graph, with the weights depending on the distance between agents. In both cases, the coupling strength is uniform across all agents, i.e., $\sigma_i = \sigma \ \forall i = 1, \dots, N$. In this work, we focus on a scenario where the units are stationary and interact through an unweighted adjacency matrix, but the coupling strength depends on the type of oscillator: $\sigma_i = \sigma_m$ if i is a BO and $\sigma_i = \sigma_f$ if i is an NBO.

In [80], several configurations of oscillators have been examined. Here, we focus on the following four configurations:

1. *Mutually Coupled NBOs*: In this scenario, N NBOs interact among themselves.

2. *Mutually Coupled BOs*: This configuration considers N BOs interacting among themselves.

3. *Master (BO) - Slave (NBO)*: In this configuration, both BOs and NBOs are present. The BOs are identical but do not interact with other units ($\sigma_m = 0$), whereas the NBOs interact among themselves and are influenced by the BOs ($\sigma_f \neq 0$). Notice that the condition $\sigma_m = 0$ implies that the dynamics of the BOs is described by Eq. (10.1).

4. *Mutually Coupled BOs and NBOs*: Here, both BOs and NBOs interact with each other and their behavior is governed by Eq. (10.3).

According to the mathematical model, BOs can synchronize among themselves, NBOs cannot, and when BOs and NBOs are mingled, BOs can still synchronize and their synchronization induces the response of NBOs. Synchronization in this context refers not to identical trajectories but to the near alignment of active phases and silent times across oscillators of the same type.

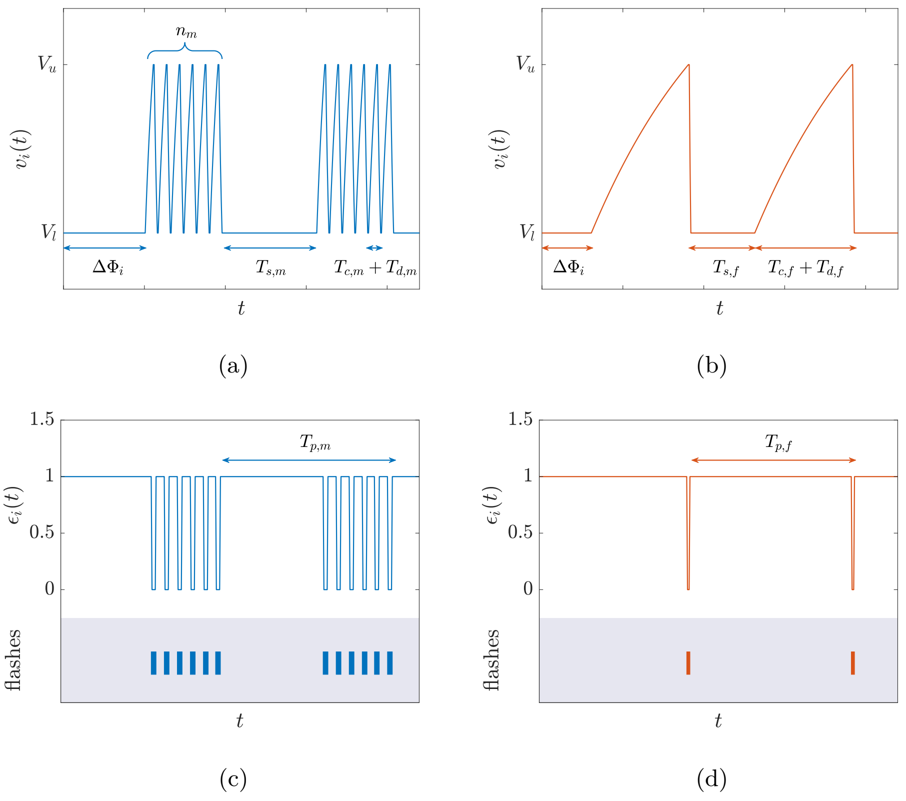


Figure 10.1: Dynamics of a BO oscillator (a) and the corresponding evolution of $\epsilon_i(t)$ (c). Dynamics of an NBO oscillator (b) and the corresponding evolution of $\epsilon_i(t)$ (d).

10.2 Robotic implementation

For the robotic implementation, we employ the e-puck2 robots equipped with the range & bearing board and the Pi-puck module, all detailed in Sec. 6.2. Specifically, in this setup, the three modules interact according to the *WiFi for Pi-puck and I^2C for e-puck2* configuration, described in Sec. 6.2.3. The experimental setup, shown in Fig. 10.2, consists of N robots placed in an arena of hexagonal shape, with each side measuring $L_y = 60$ cm. The setup also includes a computer that communicates with the robots via WiFi to initiate experiments and collect data from them at the end of each experiment.

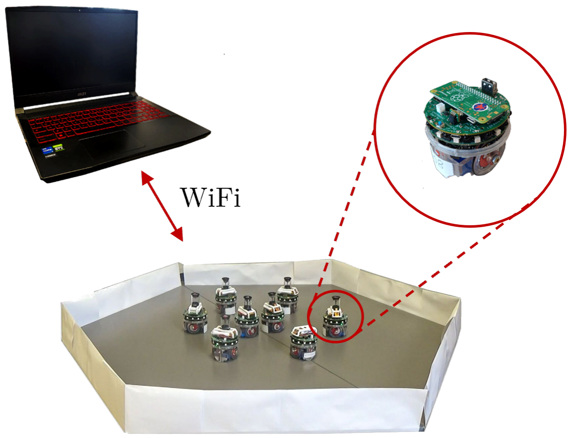


Figure 10.2: Experimental setup including N robots within a hexagonal arena with sides measuring $L_y = 60$ cm, and a computer that interacts with the robots via WiFi at the beginning and at the conclusion of each experiment to gather the robots information.

During the experiments, the robots remain stationary and, based on the values of the parameters (θ_i, σ_i, $T_{c,i}$, $T_{d,i}$, $T_{s,i}$, V_M, n_i, $\Delta\Phi_i$) loaded from a file stored on their memory, they are assigned either to BO, i.e., "male", or to NBO, i.e., "female". At each time step $t_h = ht_s$ ($h = 1, 2, \ldots$), each robot updates its state variable by performing an integration step of Eq. (10.3), by using the 4th-order Runge-Kutta method with a fixed step size of Δt. The robot then updates the value of $\epsilon_i(t_h)$ by using Eq. (10.2), and based on this value, it controls 4 green LEDsaaa (activated if $\epsilon_i(t_h) = 0$, deactivated otherwise) and broadcast its current value of $\epsilon_i(t_h)$. Both $v_i(t_h)$ and $\epsilon_i(t_h)$ are recorded in a file stored inside the memory of each robot, which will be transmitted via Wi-Fi to the computer at the end of each experiment, to post-process the produced data.

To prevent time drift during integration, before starting each experiment, the computer sends a file to the robots containing an absolute reference time, t_{ref}. This ensures that, at the beginning of the experiment, the robots, equipped with a universal clock, start their integration process once $t \geq t_{ref}$. As a result, the robots will perform the integration steps at times $t = t_{ref} + ht_s$ for $h = 1, \ldots, T/t_s$, where T denotes the total duration of the experiment.

In the mathematical model, the interaction graph is fully connected. On the other hand, in the experiment the neighbor detection relies on local communication provided by the range & bearing board, with a transmission power set to enable communication between robots up to a distance of $40\,[cm]$, to minimize battery consumption. As discussed in Sec. 6.2.2, this board is sensitive to light conditions and to interference caused by the proximity sensors and the TOF sensor integrated into the e-puck2 main board. To mitigate these issues, the sensors are calibrated at the beginning of each experiment to account for ambient light conditions, the sampling frequency of the proximity sensors is reduced, and the TOF sensor is deactivated.

Given that the communication system is not immune to message loss and message corruption, the neighborhood of the robot i is defined as follows:

$$\mathcal{N}_i(t_h) = \{j : \text{a valid message from } j \text{ is received by } i \text{ in } t' \in]t_{h-1}, t_h]\}$$

where "valid" refers to messages that are not corrupted due to interference or that do not come from previous time steps or by the robot itself. To ensure the reliability of communication, the robots broadcast messages containing information that allows the receiving robots to check their validity. Specifically, the message, consisting of two bytes, is structured as follows: the most significant byte contains the robot ID, i, which helps prevent the receiving robot from considering the same neighbor information multiple times during an integration step; the least significant byte is divided into 4-bit segments: the upper segment containing $r_i(t_h) = \text{mod}(h, 5)$, and the lower one containing $\epsilon_i(t_h)$.

Each time a robot i receives a message from another unit j, it performs a set of checks, including:

- verifying that the message is not corrupted (for example, if $\epsilon_j(t_h)$ is not equal to 0 or 1, the message is considered corrupted because $\epsilon_j(t_h)$ can assume only values of 0 or 1)
- ensuring that the message comes from the current time step, i.e., $r_i(t_h) = r_j(t_h)$

- checking that the sender j does not already belong to $\mathcal{N}_i(t_h)$.

If a message does not meet all these conditions, it will be neglected by the receiver. As a result, there is no guarantee that the adjacency matrix describing the interaction graph among the robots is time-constant and undirected. Consequently, the value of a_{ij} in Eq. (10.3) becomes time-varying, with $a_{ij}(t_h) = 1$ if and only if $j \in \mathcal{N}_i(t_h)$. All the steps here described are summarized in Algortihm 6, where << 4 (>> 4) indicates the shift on the left (right) of 4 bits, and the operation "x & 0x0f" takes the 4 least significant bits of the 1-byte variable x.

Algorithm 6: Implementation of the response to the synchronization: robot i

```
   Parameter      : $i$, $V_u$, $V_l$, $T_{c,i}$, $T_{d,i}$, $T_{s,i}$, $n_i$, $t_f$, $t_s$, $\sigma_i$, $\theta_i$, $\Delta t$
   Initialization : $v_i(t_h) = rand$, $\epsilon_i(t_h) = 1$
1  Do the light calibration
2  Read the value of $t_{ref}$ from the file sent by the computer
3  $t = t_{ref} + t_s$
4  while true do
5      if getCurrentTime $\geq t_{ref}$ then
6          msg[0] = $i$
7          msg[1] = $(r_i(t_h) <<4)\ |\ \epsilon(t_h)$
8          Broadcast msg to neighboring units via the range & bearing board
9          Check IR sensors
10         if messageReceivedFromRobot == true then
11             j = MsgReceived[0]
12             $\epsilon_j(t_h)$ = MsgReceived[1] & 0x0f
13             $n_j(t_h)$ = MsgReceived[1] >> 4
14             if $((\epsilon_j(t_h) == 1 \mid \epsilon_j(t_h) == 0)$ & $r_i(t_h) = r_j(t_h)$ & $j \notin N_i(t_h))$ then
15                 Update $\mathcal{N}_i(t_h)$
16             end
17         end
18         if (getCurrentTime $- t) \geq t_s$ then
19             Compute the coupling term $\gamma_i(t_h) = \theta_i\sigma_i \sum_{j\in\mathcal{N}_i(t_h)}(1 - \epsilon_j(t_h))$
20             Update $v_i(t_h)$ through the integration algorithm
21             Update $\epsilon_i(t_h)$
22             Write the values of $v_i(t_h)$ and $\epsilon_i(t_h)$ in a file
23             Reset the coupling term and $N_i(t_h)$
24             $t$ = getCurrentTime
25         end
26     end
27 end
```

Selection of t_s

The time t_s must be chosen large enough to allow the robots to detect their neighbors. However, it cannot be too large, as the use of the range & bearing board is a significant drain on the battery. To determine this value, we conducted experiments using two robots communicating at a distance $d = 15$ cm, testing different values of t_s. During these experiments, the robots continuously sent messages, and, at intervals of t_s, they recorded a value of 1 in a file if a message was received within the interval, or 0 otherwise. These files were then processed to compute the probability of message reception. The outcomes of these experiments are shown in Fig. 10.3(a), which depicts the likelihood of receiving messages as a function of t_s. Based on these findings, we select $t_s = 500$ ms which provides a message reception probability $p = 0.85$. Notice that this probability refers to the case in which communication occurs between two robots. Indeed, when the number of robots, denoted by N, increases, the value of p decreases. This is shown in Fig. 10.3(b). We also investigated the impact of the distance among the robots on the communication probability. For this purpose, we conducted experiments by using $N = 2$ robots and an elementary time step of $t_s = 500$ ms; as shown in Fig. 10.3(c), the probability remains constant for distances from $d = 10$ cm to $d = 35$ cm, after which it starts to decrease, reaching zero at $d = 45$ cm. Let us observe that this behavior is expected, given that the transmission power of the robots is configured to allow communication up to a maximum distance of approximately 40 cm.

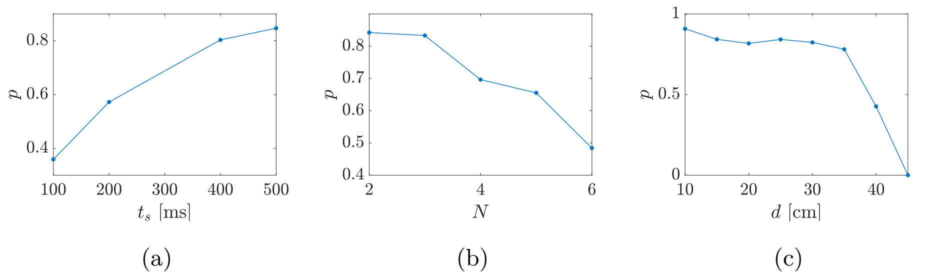


Figure 10.3: Reception probability p as function of t_s (a), N (b), and d (c).

10.3 Results

Here we show the results obtained from the experiments conducted by using the robotic implementation described in the previous section. Specifically, we carried out four types of experiments: the first shows that the NBOs are not able to synchronize in the absence of BOs; the second demonstrates that BOs can achieve synchronization; the third characterizes the response of the NBOs to the BOs synchronization in a master-slave configuration; and the fourth one investigates the NBOs response to BO synchronization in a mutually coupled BO-NBO configuration.

Throughout all the experiments, the parameters have been selected as follows: $T_{c,m} = 0.5$, $T_{d,m} = 0.2$, $T_{s,m} = 5.8$, $n_m = 6$, $T_{c,f} = 0.6$, $T_{d,f} = 0.1$, $T_{s,f} = 3$, and $\Delta t = 0.05$. In addition, V_M has been set equal to 9, resulting in $V_l = 3$ and $V_u = 6$.

In the first experiment, lasting $T = 1000$ s, $N = 2$ NBOs interact at a distance $d = 20$ cm. Each oscillator is initialized to a different initial condition and, at each time step t_h, it integrates Eq. (10.3) with $\sigma_f = 2$. Fig. 10.4 shows the dynamics of the oscillators and the corresponding flashes, revealing that, despite starting from very similar initial conditions, the two oscillators do not synchronize. This demonstrates that, as expected from the mathematical model, NBOs cannot synchronize on their own.

For the second experiment, a group of $N = 6$ BOs is arranged in a hexagonal formation with a radius $d_r = 25$ cm, and interact with each other with a coupling strength $\sigma_m = 0.05$. The experiment lasts $T = 800$ s and, as shown in Fig. 10.5(a), the system reaches synchronization despite a reception probability estimated to be ~ 0.5, according to the results shown in Fig. 10.3(b). This result is compared to the numerical simulation where we introduced the parameter p to emulate the message loss characterizing the local communication provided by the range & bearing board. We can observe that the results reflect those obtained from experiments, as shown in Fig. 10.5(b).

The third set of experiments involves $N = 8$ oscillators, consisting of 6 BO and 2 NBO units. In particular, the BOs are arranged in a hexagonal formation, with the NBOs placed at 3 cm from the center of the hexagon, facing each other, as shown in Fig. 10.6(a). In each run, the distance between the center and the vertices of the hexagon, denoted by d_r, is kept constant, but it varies across experiments, increasing incrementally from one to the next. The BOs start from the same initial condition and do not interact with any other units ($\sigma_m = 0$). Consequently, they remain synchronized throughout the entire experiment. It is important to note that synchronization does not arise from coupling but is due to the fact that the oscillators are identical and share the same initial condition. The NBOs, on the other hand, interact with each other and are susceptible to the BOs state, with a coupling strength of $\sigma_f = 2$. The synchronization among the NBOs is quantitatively evaluated using the synchronization error, computed as follows:

$$\delta(t) = \frac{1}{|C_f|} \sum_{i \in C_f} \sum_{\substack{j \in C_f, \\ j \neq i}} |v_i(t) - v_j(t)| \tag{10.5}$$

where C_f is the set identifying the NBOs. Fig. 10.6(b) shows the average synchronization error among the NBOs computed over the time interval $\left[\frac{4}{5}T, T\right]$. We can appreciate a low synchronization error for many values of d_r, i.e., up to $d_r = 50$ cm. For $d_r = 55$ cm, however, the synchronization error is high, indeed the NBOs, being very far from the BOs, do not detect the latter. The dynamics of the oscillators and corresponding flashes for $d_r = 20$ cm and for $d_r = 55$ cm are shown in Fig. 10.6(c) and Fig. 10.6(d), respectively. We notice that, for $d_r = 20$ cm, the NBOs respond to the synchronization and synchronize between themselves in a very short time.

The fourth experiment also includes $N = 8$ oscillators: 6 BOs arranged in a hexagonal formation with radius $d_r = 25$ cm and 2 NBOs placed 2 cm from the center facing each other. We hereby consider a scenario in which BOs and NBOs start from different initial conditions and interact among themselves. Specifically, the BOs interact with a coupling strength of $\sigma_m = 0.5$, while the NBOs have a coupling strength of $\sigma_f = 2$. The response to synchronization in NBOs occurs once the BOs synchronize among themselves, i.e., about at $t \sim 1000$ s or 10 bursts of the BOs, as shown in Fig. 10.7.

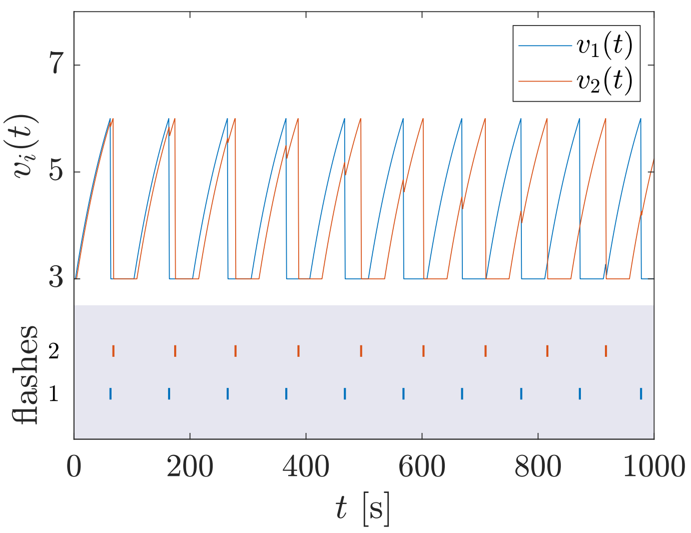
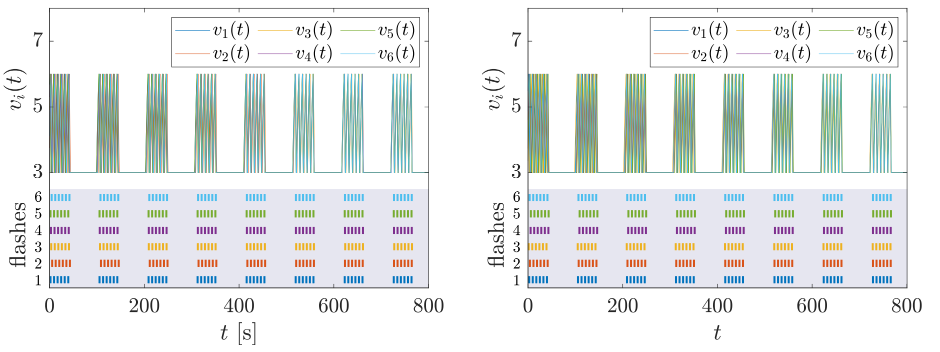


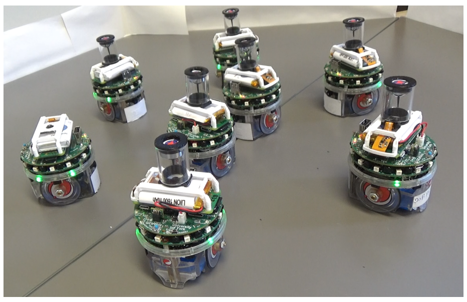
Figure 10.4: Experimental results: 2 interacting NBOs. Dynamics and corresponding flashes of 2 NBOs interacting at a distance of $d = 20$ cm. Despite starting from very close initial conditions, the NBOs do not achieve synchronization.



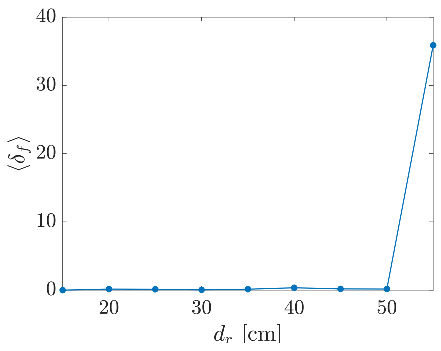
(a)

(b)

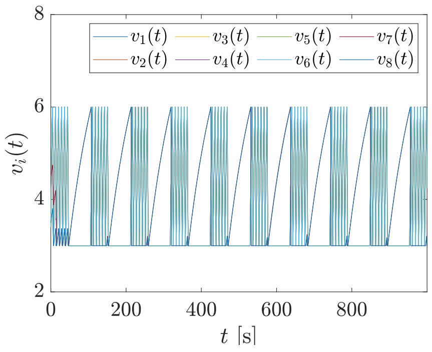
Figure 10.5: Experimental and numerical results: 6 interacting BOs. (a) Dynamics and corresponding flashes of $N = 6$ BOs arranged in a hexagonal formation with a radius $d_r = 25$ cm. (b) Results obtained from a simulation in which a probability of message reception $p = 0.5$ is considered. In both cases, a coupling strength $\sigma_m = 0.05$ has been used.



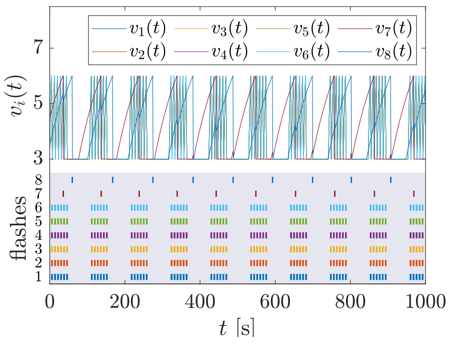
(a)



(b)



(c)



(d)

Figure 10.6: Experimental setup and results: 6 BOs (Masters) and 2 NBOs (Slaves). (a) Experimental setup used for the experimental investigation of the response of 2 NBOs, placed at 3 cm from the center of the hexagonal formation made of 6 synchronous BOs. (b) Synchronization error of the NBOs as a function of the hexagon radius d_r. (c) Dynamics and corresponding flashes of the oscillators for $d_r = 20$ cm, showing that the NBOs respond to synchronization after a short time. (d) Dynamics and corresponding flashes of the oscillators for $d_r = 55$ cm, showing that NBOs do not synchronize.

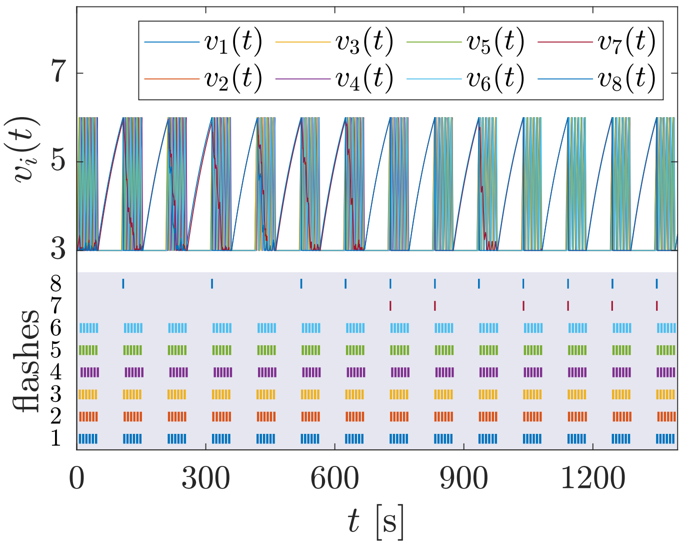


Figure 10.7: Experimental results: 6 BOs and 2 NBOs. The experiment employs the same configuration shown in Fig. 10.1 with $d_r = 25$ cm. In this experiment all the oscillators interact with each other, the BOs with a coupling strength of $\sigma_m = 0.5$ and the NBOs with $\sigma_f = 2$. Each oscillators start from different initial conditions. The figure shows the time evolution of the oscillators' state variables and their corresponding flashes.

In this chapter, we have discussed the experimental validation of the synchronization phenomenon observed in fireflies. As already mentioned in the introduction, validating the model's applicability is crucial, as it demonstrates that units not able to achieve synchronization independently can still do so through interactions with agents that can synchronize. For the robotic implementation, we have relied on a distributed control action based solely on the local communication system provided by the range & bearing board. We have considered two scenarios: one where only BO interact with each other, and another where BOs act as leaders for NBOs, who then interact with both the BOs and each other. Despite the hardware limitations, we have successfully reproduced the dynamics predicted by the mathematical model, demonstrating the model robustness.

Chapter 11

Conclusions

In this thesis, we investigated the theoretical and experimental aspects of complex systems composed of interacting units, encompassing both controlled and emergent collective behaviors.

Our approach focuses on utilizing properties of the interaction graph to develop control strategies that induce states where agents divide into groups and align their behaviors within each group. In this context, one of the main contributions of this thesis is the development of two communication protocols for inducing multiconsensus, tailored for different types of agent dynamics: single-integrator and second-order systems. Unlike conventional methods, our protocols leverage the absolute state information of the agents rather than diffusive coupling. While this reliance on absolute measurements could be seen as a limitation, it is feasible in applications where such measurements are available. We proved this by validating our first communication protocol using a team of robots, showing that, through a virtual global positioning system, we are able to address the rendezvous problem, in which agents belonging to the same group find an agreement on the point where to meet. We demonstrated also that, despite the strict constraints on the interaction network topology (connected graph for the first protocol, strongly connected digraph for the second), our protocols allow steering the system towards a desired solution on the multiconsensus manifold by pinning just a single node and using a single leader. This represents a significant advancement in reducing the complexity and resources needed for control in large-scale systems. In addition, to address the previously unresolved problem of shaping cluster synchronization in multi-agent systems, we introduced a novel control strategy for shaping cluster synchronization, exploiting the concept of spectral blocks. This strategy provides unprecedented control over the synchronizability of each cluster, including the sequencing of cluster synchronization as coupling strength varies.

Besides validating our communication protocol, we exploited the robotic platform to conduct experiments in settings where we could precisely adjust parameters that are typically beyond control. This approach highlighted not only the platform's versatility but also opened new pathways for managing and adapting otherwise uncontrollable variables, significantly expanding our research contributions. In this context, we conducted experiments aiming at the exploration of collective behaviors emerging from the local interaction among the agents, ranging from the aggregation induced by face-to-face interaction to the response to synchronization phenomenon. Indeed, in this context, the control does not rely on the action on the interaction topology or in the agents' dynamics, but on other parameters which indirectly affect the agents' interaction, such as the agent density. These validations required the use of two communication systems: local and virtual communication. In most cases, local communication proved sufficiently effective in replicating the main features of the mathematical models. This demonstrated that the models are robust enough to keep their primary characteristics even in the presence of real-world factors. However, to exactly replicate the predicted behaviors, we employed the virtual communication system, showing that robots equipped with a more refined local communication system can precisely replicate the model. These findings demonstrate that, in real-world scenarios, the interaction among agents has a more significant impact on the emergence of collective behavior than the other real-world factors. Summarizing, this approach underscores the flexibility and robustness of the proposed models in achieving desired coordination outcomes, highlighting the key role of the agents' interaction. In addition, the robotic implementation of complex system models paves the way for other applications requiring distributed control, highlighting the significant relevance of the units' interaction.

Looking forward, this thesis opens several promising avenues for future research. Studying the dynamic behavior in multi-agent systems with network topologies resembling spectral blocks could provide deeper insights into the interplay between network structure and agent behavior. Further experimental work with advanced robotic platforms will be essential to address the hardware limitations encountered and to refine our control strategies for real-world applications. Finally, another promising direction is to leverage the robotic implementations used for validating the mathematical models to explore other potential real-world applications or some variation, such as the response to synchronization in which the units are in motion or in which the units interact according to different configurations.

Overall, this thesis contributes significantly to the understanding and control of collective behaviors in multi-agent systems by highlighting the critical role of agent interactions. By combining rigorous mathematical modeling with experimental validation, we have provided a comprehensive framework that can be applied to a wide range of systems and scenarios involving local interactions.

Publications

- C. Tomaselli, L. V. Gambuzza, F. Sorrentino, and M. Frasca. Multiconsensus induced by network symmetries. *Systems & Control Letters*, 181:105629, 2023.
- C. Tomaselli, L. V. Gambuzza, F. Sorrentino, and M. Frasca. Control of multiconsensus in multi-agent systems based on eigenvector centrality. *Automatica*, 164:111638, 2024.
- C. Tomaselli, L. V. Gambuzza, G.-Q. Sun, S. Boccaletti, and M. Frasca. Taming cluster synchronization. *arXiv preprint arXiv: 2407.10638* and submitted to *Physical Review Letter*, 2024.
- C. Tomaselli, D. C. Guastella, G. Muscato, M. Frasca, and L. V. Gambuzza. A multi-robot system for the study of face-to-face interaction dynamics. *IEEE Robotics and Automation Letters*, 8(10):6715–6722, 2023.
- C. Tomaselli, D. C. Guastella, G. Muscato, L. Minati, M. Frasca, and L. V. Gambuzza. Synchronization of moving chaotic robots. *IEEE Robotics and Automation Letters*, 2024.
- C. Tomaselli, G. M. Ramírez-Ávila, L. V. Gambuzza, M. Frasca, T. Carletti, and E. Tuci. Implementation of the response to Synchronization in e-puck2 Robots. Submitted to *Proceedings of WIVACE2024*, 2024.

Bibliography

[1] M. A. Aguiar and A. P. S. Dias. Synchronization and equitable partitions in weighted networks. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 28(7), 2018.

[2] S. Ahmadizadeh, I. Shames, S. Martin, and D. Nešić. On eigenvalues of laplacian matrix for a class of directed signed graphs. *Linear Algebra and its Applications*, 523:281–306, 2017.

[3] M. Alhafnawi, E. R. Hunt, S. Lemaignan, P. O'Dowd, and S. Hauert. Deliberative democracy with robot swarms. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7296–7303. IEEE, 2022.

[4] M. Andreasson, D. V. Dimarogonas, H. Sandberg, and K. H. Johansson. Distributed control of networked dynamical systems: Static feedback, integral action and consensus. *IEEE Transactions on Automatic Control*, 59(7):1750–1764, 2014.

[5] A. Barciś, M. Barciś, and C. Bettstetter. Robots that sync and swarm: A proof of concept in ros 2. In *2019 Int Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 98–104. IEEE, 2019.

[6] A. Barciś and C. Bettstetter. Sandsbots: Robots that sync and swarm. *IEEE Access*, 8:218752–218764, 2020.

[7] A. Bayani, F. Nazarimehr, S. Jafari, K. Kovalenko, G. Contreras-Aso, K. Alfaro-Bittner, R. J. Sánchez-García, and S. Boccaletti. The transition to synchronization of networked systems. *Nature Communications*, 15(1):4955, 2024.

[8] J. Bierkens and A. Ran. A singular M-matrix perturbed by a nonnegative rank one matrix has positive principal minors; is it D-stable? *Linear Algebra and its Applications*, 457:191–208, 2014.

[9] S. Boccaletti, D.-U. Hwang, M. Chavez, A. Amann, J. Kurths, and L. M. Pecora. Synchronization in dynamical networks: Evolution along commutative graphs. *Physical Review E*, 74(1):016102, 2006.

[10] S. Boccaletti, J. Kurths, G. Osipov, D. Valladares, and C. Zhou. The synchronization of chaotic systems. *Physics reports*, 366(1-2):1–101, 2002.

[11] S. Boccaletti, A. N. Pisarchik, C. I. Del Genio, and A. Amann. *Synchronization: from coupled systems to complex networks.* Cambridge University Press, 2018.

[12] J. Buck and E. Buck. Synchronous fireflies. *Scientific American*, 234(5):74–85, 1976.

[13] T. Carletti and D. Fanelli. Theory of synchronisation and pattern formation on time varying networks. *Chaos, Solitons & Fractals*, 159:112180, 2022.

[14] P. Chvykov, T. A. Berrueta, A. Vardhan, W. Savoie, A. Samland, T. D. Murphey, K. Wiesenfeld, D. I. Goldman, and J. L. England. Low rattling: A predictive principle for self-organization in active collectives. *Science*, 371(6524):90–95, 2021.

[15] M. Crosscombe, J. Lawry, S. Hauert, and M. Homer. Robust distributed decision-making in robot swarms: Exploiting a third truth state. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 4326–4332. IEEE, 2017.

[16] S. Danø, P. G. Sørensen, and F. Hynne. Sustained oscillations in living cells. *Nature*, 402(6759):320–322, 1999.

[17] G. De Pasquale and M. E. Valcher. Tripartite and sign consensus for clustering balanced social networks. In *2021 American Control Conference (ACC)*, pages 3056–3061. IEEE, 2021.

[18] G. De Pasquale and M. E. Valcher. Consensus for clusters of agents with cooperative and antagonistic relationships. *Automatica*, 135:110002, 2022.

[19] C. Dimidov, G. Oriolo, and V. Trianni. Random walks in swarm robotics: an experiment with kilobots. In *International conference on swarm intelligence*, pages 185–196. Springer, 2016.

[20] M. Dorchain, R. Muolo, and T. Carletti. Pattern reconstruction through generalized eigenvectors on defective networks. *Europhysics Letters*, 144(1):11004, 2023.

[21] X. Duan, S. Jafarpour, and F. Bullo. Graph-theoretic stability conditions for metzler matrices and monotone systems. *SIAM Journal on Control and Optimization*, 59(5):3447–3471, 2021.

[22] L. F. Faust. Natural history and flash repertoire of the synchronous firefly photinus carolinus (coleoptera: Lampyridae) in the great smoky mountains national park. *Florida Entomologist*, pages 208–217, 2010.

[23] M. Frasca, A. Buscarino, A. Rizzo, L. Fortuna, and S. Boccaletti. Synchronization of moving chaotic agents. *Physical Review Letters*, 100(4):044102, 2008.

[24] C. Fu, Z. Deng, L. Huang, and X. Wang. Topological control of synchronous patterns in systems of networked chaotic oscillators. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 87(3):032909, 2013.

[25] H. Fujisaka and T. Yamada. Stability theory of synchronized motion in coupled-oscillator systems. *Progress of theoretical physics*, 69(1):32–47, 1983.

[26] D. Gale, H. W. Kuhn, and A. W. Tucker. Linear programming and the theory of games. *Activity analysis of production and allocation*, 13:317–335, 1951.

[27] L. Gambuzza, L. Minati, and M. Frasca. Experimental observations of chimera states in locally and non-locally coupled stuart-landau oscillator circuits. *Chaos, Solitons & Fractals*, 138:109907, 2020.

[28] L. V. Gambuzza and M. Frasca. A criterion for stability of cluster synchronization in networks with external equitable partitions. *Automatica*, 100:212–218, 2019.

[29] L. V. Gambuzza and M. Frasca. Distributed control of multiconsensus. *IEEE Transactions on Automatic Control*, 66(5):2032–2044, 2020.

[30] L. V. Gambuzza, M. Frasca, and V. Latora. Distributed control of synchronization of a group of network nodes. *IEEE Transactions on Automatic Control*, 64(1):365–372, 2018.

[31] L. V. Gambuzza, M. Frasca, F. Sorrentino, L. M. Pecora, and S. Boccaletti. Controlling symmetries and clustered dynamics of complex networks. *IEEE Transactions on Network Science and Engineering*, 8(1):282–293, 2020.

[32] D. Ghosh, M. Frasca, A. Rizzo, S. Majhi, S. Rakshit, K. Alfaro-Bittner, and S. Boccaletti. The synchronized dynamics of time-varying networks. *Physics Reports*, 949:1–63, 2022.

[33] A. Giusti, G. C. Maffettone, D. Fiore, M. Coraggio, and M. di Bernardo. Distributed control for geometric pattern formation of large-scale multirobot systems. *arXiv:2207.14567*, 2022.

[34] D. C. Guastella, L. Cantelli, G. Giammello, C. D. Melita, G. Spatino, and G. Muscato. Complete coverage path planning for aerial vehicle flocks deployed in outdoor environments. *Computers & Electrical Engineering*, 75:189–201, 2019.

[35] T. Han, Z.-H. Guan, M. Chi, B. Hu, T. Li, and X.-H. Zhang. Multi-formation control of nonlinear leader-following multi-agent systems. *ISA transactions*, 69:140–147, 2017.

[36] Y. Han, W. Lu, and T. Chen. Achieving cluster consensus in continuous-time networks of multi-agents with inter-cluster non-identical inputs. *IEEE Transactions on Automatic Control*, 60(3):793–798, 2014.

[37] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2 edition, 2004.

[38] B. Hou, F. Sun, H. Li, Y. Chen, and G. Liu. Observer-based cluster consensus control of high-order multi-agent systems. *Neurocomputing*, 168:979–982, 2015.

[39] B. Hou, F. Sun, H. Li, Y. Chen, and J. Xi. Cluster consensus of high-order multi-agent systems with switching topologies. *International Journal of Systems Science*, 47(12):2859–2868, 2016.

[40] L. Huang, Q. Chen, Y.-C. Lai, and L. M. Pecora. Generic behavior of master-stability functions in coupled nonlinear dynamical systems. *Physical Review E*, 80(3):036204, 2009.

[41] P. Ji, T. K. D. Peron, P. J. Menck, F. A. Rodrigues, and J. Kurths. Cluster explosive synchronization in complex networks. *Physical review letters*, 110(21):218701, 2013.

[42] I. Klickstein, L. Pecora, and F. Sorrentino. Symmetry induced group consensus. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(7):073101, 2019.

[43] G. Lafferriere, A. Williams, J. Caughman, and J. Veerman. Decentralized control of vehicle formations. *Systems & control letters*, 54(9):899–910, 2005.

[44] V. Latora, V. Nicosia, and G. Russo. *Complex networks: principles, methods and applications*. Cambridge University Press, 2017.

[45] M. Li and F. Deng. Cluster consensus of nonlinear multi-agent systems with markovian switching topologies and communication noises. *ISA transactions*, 116:113–120, 2021.

[46] S. Li, H. Du, and X. Lin. Finite-time consensus algorithm for multi-agent systems with double-integrator dynamics. *Automatica*, 47(8):1706–1712, 2011.

[47] W. Li and H. Dai. Cluster-based distributed consensus. *IEEE Transactions on Wireless Communications*, 8(1):28–31, 2009.

[48] X. Li, M. Z. Chen, and H. Su. Finite-time consensus of second-order multi-agent systems via a structural approach. *Journal of the Franklin Institute*, 353(15):3876–3896, 2016.

[49] Z. Li, Z. Duan, and F. L. Lewis. Distributed robust consensus control of multi-agent systems with heterogeneous matching uncertainties. *Automatica*, 50(3):883–889, 2014.

[50] W. Lin, H. Fan, Y. Wang, H. Ying, and X. Wang. Controlling synchronous patterns in complex networks. *Physical Review E*, 93(4):042209, 2016.

[51] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási. Controllability of complex networks. *nature*, 473(7346):167–173, 2011.

[52] F. Lo Iudice, A. Di Meglio, F. Della Rossa, and F. Sorrentino. Controlling consensus in networks with symmetries. *International Journal of Control*, pages 1–17, 2021.

[53] E. N. Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

[54] X.-Q. Lu and S.-H. Chen. Cluster consensus of second-order multi-agent systems via pinning control. *Chinese Physics B*, 19(12):120506, 2010.

[55] S. Luo and D. Ye. Cluster consensus control of linear multi-agents systems under directed topology with general partition. *IEEE Transactions on Automatic Control*, 2021.

[56] H. D. Macedo and J. N. Oliveira. Typing linear algebra: A biproduct-oriented approach. *Science of Computer Programming*, 78(11):2160–2191, 2013.

[57] S. Mahler, A. A. Friesem, and N. Davidson. Experimental demonstration of crowd synchrony and first-order transition with lasers. *Physical Review Research*, 2(4):043220, 2020.

[58] J. Matoušek and B. Gärtner. *Understanding and using linear programming*, volume 1. Springer, 2007.

[59] Z. Meng, W. Ren, Y. Cao, and Z. You. Leaderless and leader-following consensus with communication and input delays under a directed network topology. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(1):75–88, 2010.

[60] M. Mesbahi and M. Egerstedt. *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[61] L. Minati. Remote synchronization of amplitudes across an experimental ring of non-linear oscillators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 25(12):123107, 2015.

[62] A. Moiseff and J. Copeland. Firefly synchrony: a behavioral strategy to minimize visual clutter. *Science*, 329(5988):181–181, 2010.

[63] S. Monaco and L. R. Celsi. On multi-consensus and almost equitable graph partitions. *Automatica*, 103:53–61, 2019.

[64] F. Morone and H. A. Makse. Symmetry group factorization reveals the structure-function relation in the neural connectome of caenorhabditis elegans. *Nature Communications*, 10(1):1–13, 2019.

[65] C. Nathe, L. V. Gambuzza, M. Frasca, and F. Sorrentino. Looking beyond community structure leads to the discovery of dynamical communities in weighted networks. *Scientific Reports*, 12(1):1–12, 2022.

[66] V. Nicosia, M. Valencia, M. Chavez, A. Díaz-Guilera, and V. Latora. Remote synchronization reveals network symmetries and functional modules. *Physical review letters*, 110(17):174102, 2013.

[67] T. Nishikawa and A. E. Motter. Synchronization is optimal in nondiagonalizable networks. *Physical Review E*, 73(6):065106, 2006.

[68] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

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

[70] K. P. O'Keeffe, H. Hong, and S. H. Strogatz. Oscillators that sync and swarm. *Nature Communications*, 8(1):1504, 2017.

[71] L. Pan, H. Shao, M. Mesbahi, D. Li, and Y. Xi. Cluster consensus on matrix-weighted switching networks. *Automatica*, 141:110308, 2022.

[72] J. Pantaleone. Synchronization of metronomes. *American Journal of Physics*, 70(10):992–1000, 2002.

[73] P. C. Parks and V. Hahn. *Stability theory.* Prentice-Hall, Inc., 1993.

[74] L. M. Pecora and T. L. Carroll. Master stability functions for synchronized coupled systems. *Physical review letters*, 80(10):2109, 1998.

[75] L. M. Pecora, F. Sorrentino, A. M. Hagerstrom, T. E. Murphy, and R. Roy. Cluster synchronization and isolated desynchronization in complex networks with symmetries. *Nature Communications*, 5(1):4079, 2014.

[76] F. Perez-Diaz, S. M. Trenkwalder, R. Zillmer, and R. Groß. Emergence and inhibition of synchronization in robot swarms. In *Distributed Autonomous Robotic Systems*, pages 475–486. Springer, 2018.

[77] F. Perez-Diaz, R. Zillmer, and R. Groß. Firefly-inspired synchronization in swarms of mobile agents. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015)*, pages 279–286, 2015.

[78] A. Pikovsky, M. Rosenblum, J. Kurths, and A. Synchronization. A universal concept in nonlinear sciences. *Self*, 2:3, 2001.

[79] J. Qin and C. Yu. Cluster consensus control of generic linear multi-agent systems under directed topology with acyclic partition. *Automatica*, 49(9):2898–2905, 2013.

[80] G. M. Ramírez-Ávila and J. Kurths. Unraveling the primary mechanisms leading to synchronization response in dissimilar oscillators. *The European Physical Journal Special Topics*, 225:2487–2506, 2016.

[81] W. Ren. On consensus algorithms for double-integrator dynamics. *IEEE Transactions on Automatic Control*, 53(6):1503–1509, 2008.

[82] W. Ren and R. W. Beard. Consensus algorithms for double-integrator dynamics. *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications*, pages 77–104, 2008.

[83] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control systems magazine*, 27(2):71–82, 2007.

[84] M. G. Rosenblum and A. S. Pikovsky. Controlling synchronization in an ensemble of globally coupled oscillators. *Physical Review Letters*, 92(11):114102, 2004.

[85] O. E. Rössler. An equation for continuous chaos. *Physics Letters A*, 57(5):397–398, 1976.

[86] S. Roy. Scaled consensus. *Automatica*, 51:259–262, 2015.

[87] R. J. Sánchez-García. Exploiting symmetry in network analysis. *Communications Physics*, 3(1):1–15, 2020.

[88] M. T. Schaub, N. O'Clery, Y. N. Billeh, J.-C. Delvenne, R. Lambiotte, and M. Barahona. Graph partitions and cluster synchronization in networks of oscillators. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 26(9), 2016.

[89] A. Scheidler, A. Brutschy, E. Ferrante, and M. Dorigo. The $\{k\}$-unanimity rule for self-organized decision-making in swarms of robots. *IEEE Trans. Cybernetics*, 46(5):1175–1188, 2015.

[90] R. Sevilla-Escoboza, J. Buldú, S. Boccaletti, D. Papo, D.-U. Hwang, G. Huerta-Cuellar, and R. Gutierrez. Experimental implementation of maximally synchronizable networks. *Physica A: Statistical Mechanics and its Applications*, 448:113–121, 2016.

[91] F. Sorrentino and E. Ott. Network synchronization of groups. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 76(5):056114, 2007.

[92] F. Sorrentino, L. Pecora, and L. Trajković. Group consensus in multilayer networks. *IEEE Transactions on Network Science and Engineering*, 7(3):2016–2026, 2020.

[93] F. Sorrentino, L. M. Pecora, A. M. Hagerstrom, T. E. Murphy, and R. Roy. Complete characterization of the stability of cluster

synchronization in complex dynamical networks. *Science Advances*, 2(4):e1501737, 2016.

[94] W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil. Distributed, physics-based control of swarms of vehicles. *Autonomous robots*, 17(2-3):137–162, 2004.

[95] M. Starnini, A. Baronchelli, and R. Pastor-Satorras. Modeling human dynamics of face-to-face interaction networks. *Physical Review Letters*, 110(16):168701, 2013.

[96] M. Starnini, M. Frasca, and A. Baronchelli. Emergence of metapopulations and echo chambers in mobile agents. *Scientific reports*, 6(1):1–8, 2016.

[97] G. W. Stewart. *Matrix perturbation theory.* Citeseer, 1990.

[98] H. Su, G. Jia, and M. Z. Chen. Semi-global containment control of multi-agent systems with intermittent input saturation. *Journal of the Franklin Institute*, 352(9):3504–3525, 2015.

[99] Y. G. Sun, L. Wang, and G. Xie. Average consensus in networks of dynamic agents with switching topologies and multiple time-varying delays. *Systems & Control Letters*, 57(2):175–183, 2008.

[100] M. S. Talamali, A. Saha, J. A. Marshall, and A. Reina. When less is more: robot swarms adapt better to changes with constrained communication. *Science Robotics*, 6(56):eabf1416, 2021.

[101] C. Tomaselli, L. V. Gambuzza, F. Sorrentino, and M. Frasca. Multi-consensus induced by network symmetries. *Systems & Control Letters*, 181:105629, 2023.

[102] C. Tomaselli, L. V. Gambuzza, F. Sorrentino, and M. Frasca. Control of multiconsensus in multi-agent systems based on eigenvector centrality. *Automatica*, 164:111638, 2024.

[103] C. Tomaselli, L. V. Gambuzza, G.-Q. Sun, S. Boccaletti, and M. Frasca. Taming cluster synchronization. *arXiv preprint arXiv:2407.10638* and submitted to *Physical review letters*, 2024.

[104] C. Tomaselli, D. C. Guastella, G. Muscato, M. Frasca, and L. V. Gambuzza. A multi-robot system for the study of face-to-face interaction dynamics. *IEEE Robotics and Automation Letters*, 8(10):6715–6722, 2023.

[105] C. Tomaselli, D. C. Guastella, G. Muscato, L. Minati, M. Frasca, and L. V. Gambuzza. Synchronization of moving chaotic robots. *IEEE Robotics and Automation Letters*, 2024.

[106] C. Tomaselli, G. M. Ramírez-Ávila, L. V. Gambuzza, M. Frasca, T. Carletti, and E. Tuci. Implementation of the response to synchronization in e-puck2 robots. Submitted to *Proceedings of WIVACE2024*, 2024.

[107] V. Trianni, D. De Simone, A. Reina, and A. Baronchelli. Emergence of consensus in a multi-robot network: from abstract models to empirical validation. *IEEE Robotics and Automation Letters*, 1(1):348–353, 2016.

[108] C. R. Williams, T. E. Murphy, R. Roy, F. Sorrentino, T. Dahms, and E. Schöll. Experimental observations of group synchrony in a system of chaotic optoelectronic oscillators. *Physical review letters*, 110(6):064104, 2013.

[109] W. Xia and M. Cao. Clustering in diffusively coupled networks. *Automatica*, 47(11):2395–2405, 2011.

[110] G. Xie and L. Wang. Consensus control for a class of networks of dynamic agents: Fixed topology. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 96–101. IEEE, 2005.

[111] G. Xie and L. Wang. Consensus control for a class of networks of dynamic agents. *International Journal of Robust and Nonlinear Control*, 17(10-11):941–959, 2007.

[112] B. Xu and W. He. Event-triggered cluster consensus of leader-following linear multi-agent systems. *Journal of Artificial Intelligence and Soft Computing Research*, 8(4):293–302, 2018.

[113] Z. Yaghoubi and H. A. Talebi. Cluster consensus for nonlinear multi-agent systems. *Journal of Intelligent & Robotic Systems*, 100(3-4):1069–1084, 2020.

[114] J. Yu and L. Wang. Group consensus in multi-agent systems with switching topologies and communication delays. *Systems & Control Letters*, 59(6):340–348, 2010.

[115] W. Yu, G. Chen, M. Cao, and W. Ren. Delay-induced consensus and quasi-consensus in multi-agent dynamical systems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 60(10):2679–2687, 2013.

[116] W. Yu, G. Chen, and J. Lü. On pinning synchronization of complex dynamical networks. *Automatica*, 45(2):429–435, 2009.

[117] W. Yu, G. Chen, W. Ren, J. Kurths, and W. X. Zheng. Distributed higher order consensus protocols in multiagent dynamical systems. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(8):1924–1932, 2011.

[118] W. Yu, P. DeLellis, G. Chen, M. Di Bernardo, and J. Kurths. Distributed adaptive control of synchronization in complex networks. *IEEE Transactions on Automatic control*, 57(8):2153–2158, 2012.

[119] L. Zemanová, C. Zhou, and J. Kurths. Structural and functional clusters of complex brain networks. *Physica D: Nonlinear Phenomena*, 224(1-2):202–212, 2006.

[120] J. Zhan and X. Li. Cluster consensus in networks of agents with weighted cooperative–competitive interactions. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(2):241–245, 2017.

[121] G. Zhang, Z. Liu, and Z. Ma. Synchronization of complex dynamical networks via impulsive control. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 17(4), 2007.

[122] M. Zhao, C. Peng, Q.-L. Han, and X.-M. Zhang. Cluster consensus of multiagent systems with weighted antagonistic interactions. *IEEE Transactions on Cybernetics*, 2020.