Distributed control of multi-agent systems under communication constraints : application to robotics

Syed Ali Ajwad

Thèse
Pour obtenir le grade de
Docteur de l'université de Poitiers
École Nationale Supérieure d'ingénieurs de Poitiers

Diplôme National - Arrêté du 25 mai 2016

École Doctorale :
Sciences et Ingénierie des Systèmes, Mathématiques, Informatique de la ComUE de l'Université Confédérale Léonard de Vinci

Spécialité :
Automatique, Productique et Robotique

Présentée par
Syed Ali Ajwad

Distributed control of multi-agent systems under communication constraints: application to robotics

---

Directeur de thèse	:	Emmanuel MOULAY	Chargé de recherche CNRS à XLIM
Co-directeur	:	Patrick COIRAULT	Professeur à l'Université de Poitiers
Co-encadrant	:	Michael DEFOORT	Maître de conférences à l'Université Polytechnique Hauts-de-France

Soutenue le 4 Septembre 2020

Thèse préparée au sein du
Laboratoire d'Informatique et d'Automatique pour les Systèmes

Composition du Jury

Rapporteurs :	Elena PANTELEY	Directrice de recherche CNRS au L2S
	Franck PLESTAN	Professeur à l'École Centrale de Nantes
Examinateurs :	Mohamed DJEMAI	Professeur à l'Université Polytechnique Hauts-de-France
	Tomas MÉNARD	Maître de conférences à l'Université de Caen
	Emmanuel MOULAY	Chargé de recherche CNRS à XLIM
	Patrick COIRAULT	Professeur à l'Université de Poitiers
	Michael DEFOORT	Maître de conférences à l'Université Polytechnique Hauts-de-France

Dedicated to

my dearest father Saghir Hussain

my loving mother Tahira Hussain

&

my beautiful wife Sanna Tauseef

Contents

List of Figures

List of Tables

Notations
Following are the notations used in throughout the manuscript:

- $\mathbb{N}$: Set of all natural numbers.
- $\mathbb{R}$: Set of all real numbers.
- $\stackrel{\triangle}{=}$: equal by definition.
- $\mathbf{1}_n = (1, \ldots, 1)^T \in \mathbb{R}^n$.
- I_n: n dimensional identity matrix.
- $\mathcal{G}$: Communication graph.
- $\mathcal{A}$: Adjacency matrix.
- $\mathcal{L}$: Laplacian matrix.
- $\lambda_{\min}(A)$: Minimum eigenvalue of symmetric matrix A.
- $\lambda_{\max}(A)$: Maximum eigenvalue of symmetric matrix A.
- $\mathcal{D}_i^n$: An $n \times n$ matrix with all entries equal to zero except the i^{th} diagonal entry which is 1.
- $\|.\|_2$: Euclidean norm.
- $\|.\|_F$: Frobenius norm.
- $A \otimes B$: Kronecker product of A and B.

Acronyms
The list of acronyms is as below:

- APF: Artificial Potential Function
- LIAS: Laboratoire d'Informatique et d'Automatique pour les Systèmes
- LMI: Linear Matrix Inequality
- MAS: Multi-Agent System
- MIMO: Multi-Input-Multi-Output
- ROS: Robot Operating System
- UAV: Unmanned Air Vehicle

Acknowledgements

First of all, I am thankful to One who says "Can they who know and they who do not know be deemed equal? Only those endowed with pure understanding do take heed".

I have so many people to thank for their help throughout my PhD thesis. Firstly, my sincere gratitude to my supervisor **Dr. Emmanuel Moulay** for believing in me and providing me the opportunity to start with. I am thankful for his invaluable help and patience during the course of this research project. I am grateful to **Prof. Patrick Coirault** for sharing his experience, valuable suggestions and providing me the a conducive environment. My heartfelt thanks to **Dr. Michael Defoort** for his greatest help and guidance throughout this research journey. He provided me the opportunity of working in his lab on robotic platforms which was an extremely knowledgeable experience. Special thanks to **Dr. Tomas Ménard** for his great help in complex mathematical analysis of the theorems.

I am also very thankful to **Dr. Elena Panteley**, **Prof. Franck Plestan** and **Prof. Mohamed Djemai** for agreeing to be part of the jury and evaluating my thesis. I really appreciate their valuable discussion and feedback.

During my PhD, I have been very fortunate to have brilliant colleagues and friends at the laboratory. Thanks Achraf, Bilal, Fayrouz, Florence and all others for your cooperation, friendship and all the fun we had together in the last 3 years. My special gratitude to Pipit from LAMIH lab.

Finally but most importantly, my very special thanks to my wife Sanna for her love and persistent support. She made countless sacrifices to help me get to this point. I would also like to express my deepest gratitude to my parents and siblings for their unconditional love and prayers.

Part I

General introduction

General introduction

A Multi-Agent System (MAS) consists of multiple autonomous subsystems which can interact with each other and the environment to complete tasks in a cooperative manner. MAS has been an important subject of research since the last decade. The study of MAS is mostly inspired by the natural phenomena exhibited by various animal species. Fish schooling, bird flocking or flying in V shape, bacteria swarming and mammal herds are few examples of cooperative behaviour in nature. Researchers have developed various control methods for cooperative operation and applied them in different applications like distributed computing, sensor networks, satellite constellation, power grids and robotics. In robotics, cooperative control techniques are applied to achieve different objectives in vast areas including but not limited to military, automated industry, heavy payload transportation, exploration, rescue operations and entertainment.

Cooperative control of MAS heavily depends on communication and information exchange among the agents. This gives rise to two natural choices of communication network which are centralised and distributed control network. In centralized communication, as apparent from the name, there is a central unit to which all the agents are connected to and send their information. The central station handles all the information exchange and sends control commands to the agents. This kind of scheme has some major drawbacks. For example, if the central unit fails, the whole network will collapse. Moreover, a centralized control scheme cannot handle large number of agents due to network saturation and/or limited computing and processing capability etc. On the other hand, a distributed control scheme does not require a central control unit. Agents communicate directly to their neighboring agents and exchange information. The information received from neighbors is also called local information. Each agent in the distributed network uses this local information to compute its own control input. Distributed control schemes offer many advantages over the centralized control scheme in terms of efficiency, adaptability, robustness and scalability.

Communication topology among the agents can be represented by a graph which describes the links between the agents. Therefore, graph theory plays an important role in analysis of MAS. In fact, adjacency and Laplacian matrices associated with the graph provides useful information about the network topology which dictates the behaviour of the MAS. A communication topology in a distributed MAS network could be directed or undirected. If the communication among the neighbors is one way then it is called directed and undirected if the communication is bidirectional. Moreover, communication links between the agents could remain either fixed (fixed topology) or they can evolve with time (switching topology).

In distributed cooperative control of MAS, consensus and formation control are considered the most fundamental problems. In consensus control, the states of all agents of a MAS are required to converge to a common value, for example, same altitude and speed of all drones in the network or same power levels of batteries in an energy storage facility etc. The common state which is achieved in result of consensus is known as consensus state. In some cases, the agents not only have to reach to the same states but are also required to follow a reference trajectory produced by a real or a virtual leader. This is known as leader-following consensus or consensus tracking. The reference trajectory generated by the leader is independent of other agents in the network and it is usually designed separately according to the application requirement. Various distributed control algorithms are proposed by researchers to deal with the problem of both leaderless and leader-following consensus.

Formation control is another important topic in distributed control of MAS. In formation control, the states of agents like position, velocity and orientation are controlled in a way that they produce a geometric shape. This geometric pattern could either be fixed or time-varying depending on the application. The required formation shape can be defined by a formation vector. In some cases agents are required to make a formation and track the trajectory of the leader while maintaining the shape. This is known as formation tracking. Various formation control and formation tracking algorithms are proposed by the research community using different techniques. It is also shown that the consensus-based algorithm can be used to achieve formation in MAS by choosing an appropriate deviation from the consensus state.

Information exchange among the neighboring agents is vital for distributed consensus and forma-

tion control algorithms since the computation of control law usually depends on the state information of the neighbors. Mostly, it is considered that the neighbors' information is available continuously. However, in reality, this is not possible because continuous transmission of data requires infinite bandwidth of the communication channel. Moreover, the available communication and processing equipment is digital in nature. Therefore, sampled-data transmission is inevitable.

In practical applications, sampled-data MAS also suffered from irregular and nonuniform sampling periods which are normally ignored in most of the existing literature on distributed cooperative control of MAS. Similarly, another important issue related to real MAS is asynchronous transmission between the agents. Though generally it is assumed that the agents transmit their data at the same instant. However, to achieve synchronized data transmission, clocks of the agents must be precisely synchronized and processing delays must be the same. However this is not possible in real world scenarios.

Distributed control algorithm design usually requires information of all states of the neighbors. However, in some cases, measuring every state is not feasible. This may be due to various reasons like compact size of the agent, cost reduction or unavailability of appropriate sensors. Therefore, these limited on-board resources result in partially available states. In addition, even if all states can be measured, more communication resources will be required to transmit them. Therefore, it is more economical if we communicate only some of the states and estimate the unavailable states from the available data.

Motivated by the above discussion, this thesis is focused on the study of leader-following consensus and formation tracking problems of second-order MAS with communication constraints. These constraints include:

- Each agent can only measure its position state.
- Agents are not equipped with sensors to measure their velocity.
- Agents do not have access to the input (acceleration) of their neighbors.
- The measured state is transmitted to the neighbors at irregular and non-uniform time intervals.
- The transmission among the agents is asynchronous and totally independent of other agents in the network.
- The communication topology among the agents is directed.

In Chapter2, first, we investigate the case of leader-following consensus of MAS with fixed topology. The main idea is to use a continuous-discrete time observer to reconstruct both position and velocity states in continuous time from the available discrete position data. An agent not only estimates its own states but also estimates the states of its neighbors. The observer must take into account irregular and asynchronous sampling. These estimated states then can be used to design the distributed leader-following control law.

Secondly, the results of the fixed-time leader-following control algorithm are extended for the case of switching topology. The main issue in switching topology is that if the switching is occurring too fast, the MAS can suffer with chattering and zeno effects. Therefore, it is important to calculate the conditions for the minimum required dwell time in which the topology remains constant.

In Chapter 3, we investigate the problem of formation tracking. we again considered the MAS with the above mention communication constraints. The designed leader-following consensus algorithm can be modified to achieve formation tracking by including some offsets in the position and the velocity consensus states. The value of offset depends on the desired geometric shape and can be represented through formation vector. Both cases of fixed and time-varying formation shapes are considered.

In the next step, we study another important issue of collision avoidance among the agents. which is related to MAS with moving agents like ground vehicles and drones etc. If the formation vector in formation tracking scheme is chosen appropriately, i.e keeping in mind the inter-agent distance, the agents will not collide once they achieve the desired shape. However, the agents may collide while converging to the required positions. Therefore, we need some collision avoidance mechanism

which restrains agents from coming too close to each other. We introduce a repulsive force between the agents through Artificial Potential Function (APF). The force acts on the agents when distance between them becomes less than some threshold and causing them to move away from each other.

Finally in Chapter 4, we apply the designed algorithms of leader-following consensus, formation tracking and collision avoidance on a multi-robot network to verify their efficacy in real applications. The robotic network consists of wheeled mobile robots called Mini-Lab from Enova robotics. The algorithms are implemented through Robot Operating System (ROS).

ROS is an open-source meta-operating robotic framework. Since robotics is a multi-disciplinary domain that requires skills of mechanical engineering, electronics embedded systems and computer programming, therefore, a decade ago, robotic engineers had to spend a lot of time in designing the basic software and hardware architecture of the robot before testing their designed algorithm. Due to the lack of any platform which provides the essential software architecture with re-usable programs, robot designing was a cumbersome job and mostly involved reinvention of the wheel before implementing the advanced algorithm. ROS was developed to fill this gap. It provides software tools and libraries specifically designed for robotic applications. Since it is an open-source platform, one can modify and upgrade the software to apply and test any algorithm on the robot. Moreover, ROS-gazebo is a robotic simulator which provides realistic simulations by incorporating real-world factors like friction, sensor feedback and collision detection etc.

A distributed network is established for the implementation purpose of the proposed control law. Since wheeled mobile robots are usually subject to motion constraints known as nonholonomic constraints, in order to apply the designed distributed algorithms of second-order MAS on the multi-robotic network, a new control strategy is required that can deal with the nonholonomic constraints. We use the robot flatness property based technique for this purpose. ROS-Gazebo simulation and hardware results are obtained to examine the efficiency and effectiveness of the proposed algorithms for real applications.

Part II

Cooperative control of multi-agent systems

Chapter 1

Introduction

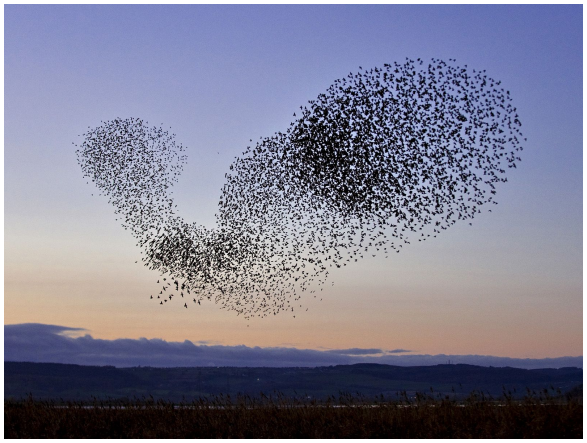
Contents

1.1 Background and motivation

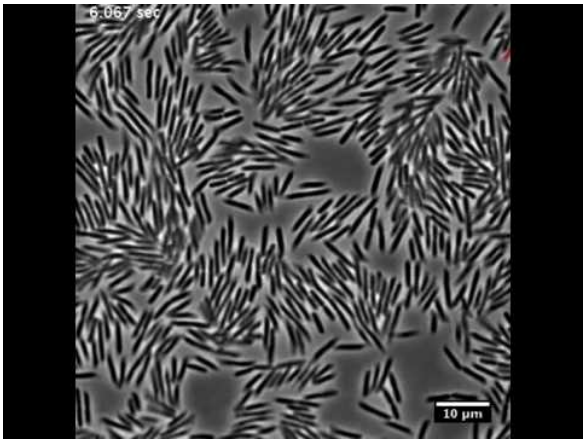
A Multi-Agent System (MAS) is defined as a group of autonomous sub-systems, called agents, which can interact with each other and with their environment. The study of MAS is essentially inspired and motivated by the collective behavior of various biological species. In nature, some animals make use of social grouping, acting in a cooperative manner to achieve a desired common goal. Widely observed examples of such natural behaviors are fish schooling, bird flocking, bacteria swarming and mammal herds (Figure 1.1).

(a) Fish schooling

(b) Mammal Herd



(c) Birds flocking



(d) Bacteria swarming

Figure 1.1: Cooperative behaviour of biological species

Inspiration from these natural phenomenon has served as the basis for the study and development of cooperative control of MAS by researchers and scientists of various fields. As such, cooperative control of MAS has seen a vast range of applications including but not limited to sensor networks [1], satellite constellation [2], electric power systems [3], distributed computing [4], synchronization [5] and intoferometers [6].

Robotics is one of the most important area of application of networked systems [7]. While advancement in technology has enabled advanced operations for autonomous robots, the performance and effectiveness of robots has seen further improvement by deploying them in a group to work in a cooperative manner. In fact, a multi-robot system can accomplish complex tasks which would not be possible with a single robot operation. Classical examples of cooperative control in robotics can be found in heavy payload transportation, search and rescue operations, automated industries, exploration, and defence [8].

Though each of these applications has its own complexity and challenges, they do share some underlying common characteristics. Some important MAS related problems include synchronization, rendezvous, flocking, consensus and formation tracking etc. One of the most crucial and vital aspect of cooperative control is communication and information sharing between the agents. Unlike a single agent system, in MAS, we have sub-systems which are required to share their information to achieve

a global task effectively. Due to the limited resources like bandwidth, sensors and processing power, information availability and its communication are not straightforward. Hence, it is quite difficult to control the behaviour of the agents to complete a global objective. In this thesis, we study two fundamental problems of cooperative control, namely consensus and formation tracking, for the MAS under communication constraints.

1.2 Communication network in cooperative control

Communication and information exchange are keys to achieve a global task in MAS and lend itself to the classification of cooperative control of MAS into two classes, namely centralized control and distributed control. In the case of **centralized** control, it is considered that all the agents are fully connected to a central location and that they transmit their state information to the central controlling unit. This central unit evaluates the states and computes the control inputs for all the individual agents and in turn, transmits the computed control inputs to the respected agents through the same connected network. In such a centralized network topology, agents are not required to communicate with each other but only with the central control unit. However, there are many serious drawbacks associated with the centralized control strategies. For instance, if the communication system of centralized control unit fails due to any reason (e.g. hardware failure or distortion in communication signals), the whole MAS will break down. Such a failure could be catastrophic especially when the agents are moving vehicles. Furthermore, as the number of agents increases the amount of data exchange between the agents and the central controller is expected to increase significantly. This in itself presents further issues given real-life constraints such as the limited capacity of installed communication equipment, or budget, thereby making it difficult to accommodate a huge number of agents. As a result, the centrally controlled network scheme faces serious scalability issues. Moreover, as all the computations are carried out in the central unit, the induction of a large number of agents significantly impacts the calculation time of control inputs. Therefore, in the presence of real-world problems, cooperative control with a centralized scheme becomes more challenging.

Figure 1.2: Centralized control scheme

The issues attached with the centralized control schemes favor the need for a **distributed** control technique. Unlike centralized control schemes, a distributed scheme does not have any common control unit. Instead, agents are required to exchange their information with their neighbors only (Figure 1.3) and use the received information of the neighbor, also known as local information, to compute their control input. Study of distributed control of MAS has gained much popularity in the research community during the last decade because of its advantages over centralized control such as efficiency, scalability, flexibility, robustness and adaptability [9].

The distributed communication topology can further be divided in two types based on whether the network remains the same or changes with time.

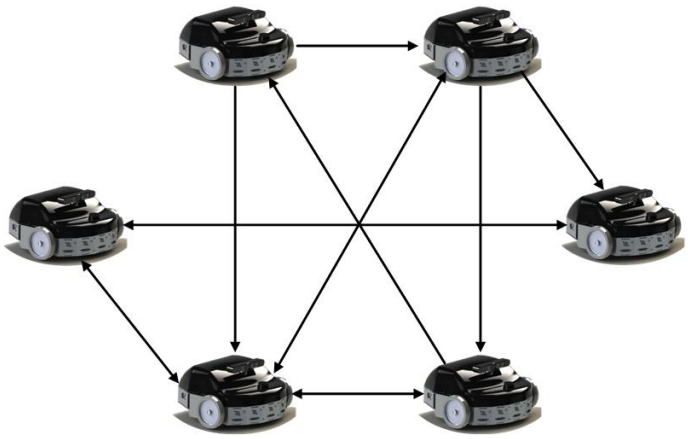


Figure 1.3: Distributed control scheme

- **Fixed topology:** If the communication network among the agents is not changing and remains fixed then it is called a fixed topology.
- **Switching topology:** In some practical cases, it is not possible for the agents to keep a fixed network and they may need to change their neighbors due to different physical limitations of the communication hardware. The topology where the communication network is switching with time is called switching topology.

The distributed communication network has also been classified into undirected and directed types as defined below:

- **Undirected topology:** If all the agents in the network are able to send and receive information to and from neighbors then the communication topology is undirected.
- **Directed topology:** If an agent can send its information to the another agent but cannot necessarily receive information from that other agent then the communication topology is called a directed topology.

1.3 Algebraic graph theory

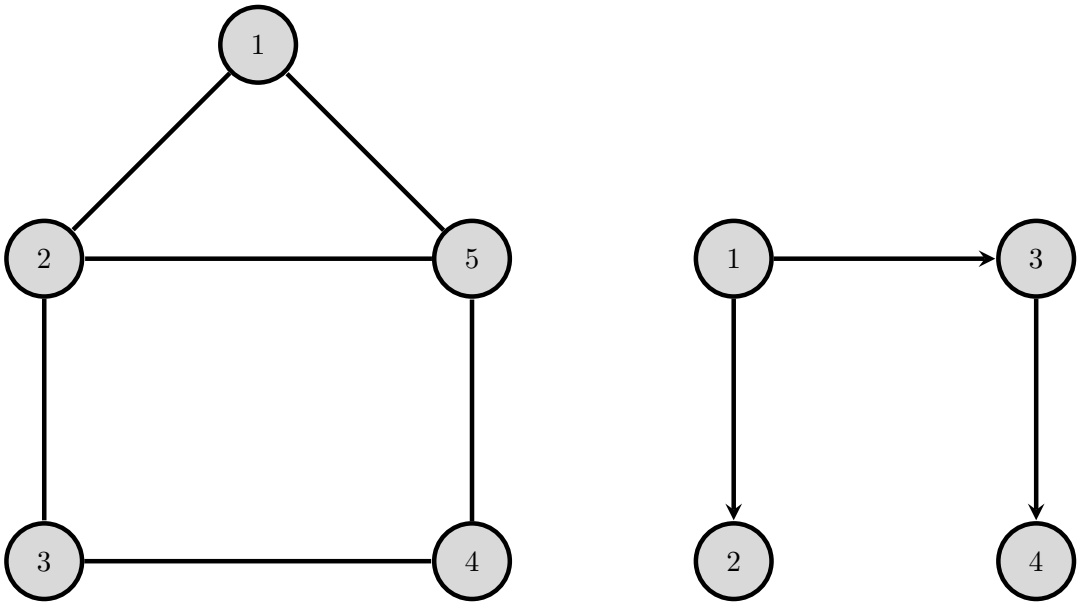
A communication topology that defines connections and information flow paths between the agents in a MAS, is usually modelled by algebraic graphs. Algebraic graphs and graph theory have been intensively investigated in literature [10–12]. In the following sections, some important definitions and properties of algebraic graphs have been discussed which will be used throughout the thesis.

1.3.1 Basics of graph theory

Mathematically, a graph is a pair of a nonempty finite set of nodes also called vertices and a finite set of edges or links. In MAS, each agent is represented as a node, whereas the communication link between two agents is shown as an edge. For a MAS consisting of N agents, the corresponding graph can be defined as $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$ is a set of N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. Let v_i and v_j be two nodes of the graph. If information is being shared between v_i and v_j and the information flows from v_i to v_j, there exists a pair (v_i, v_j) and this pair is an element of $\mathcal{E}$. If $(v_i, v_j) \in \mathcal{E}$, then node v_i is a neighbor of v_j and v_i is called the parent node while v_j is known as a child node. The set of neighbors of node v_i is described as $\mathcal{N}_i = \{i \neq j : (v_i, v_j) \in \mathcal{E})\}$.

A graph is called an undirected graph if for a pair of nodes with an edge (v_i, v_j), there also exists the edge (v_j, v_i) otherwise the graph is known as a directed graph. In the case of undirected graphs, sometimes, arrows are not shown in the visual representation. In a weighted graph, a weight w_{ij} is assigned to each edge (v_i, v_j). For an undirected weighted graph, w_{ij} is equal to w_{ji}.

For visualization, nodes in a graphs are represented as dots and circles while if the edge (v_i, v_j) exists, it is shown by an arrow directed from v_i to v_j (for undirected graphs, arrows are not required). A visual representation of both undirected and directed graphs is shown in Figure 1.4.



(a) Undirected graph (b) Directed graph

Figure 1.4: Visual representation of graph

Definition 1. *A directed path between nodes v_a and v_b is a sequence of edges in a directed graph such that there exists an edge between all consecutive nodes i.e. $(v_a, v_{a+1}), (v_{a+1}, v_{a+2}), \dots (v_{b-1}, v_b)$.*

Definition 2. *A cycle is a directed path in a directed graph that starts and ends at the same node.*

Definition 3. *A directed graph is strongly connected if all the nodes of the graph are connected with each other through a directed path. Similarly, an undirected graph is connected if there is an undirected path between each pair of distinct nodes.*

Definition 4. *A directed graph in which all nodes have exactly one parent node except one node is called a directed tree. The node which does not have any parent node is known as the root.*

It is to note that a directed tree does not have any cycle because all the edges are oriented away for/from the node.

A subgraph of $\mathcal{G}$, represented as $\mathcal{G}^s \triangleq (\mathcal{V}^s, \mathcal{E}^s)$ is a graph with the set of node $\mathcal{V}^s \subseteq \mathcal{V}$ and set of edges $\mathcal{E}^s \subseteq \mathcal{E} \cap \mathcal{V}^s \times \mathcal{V}^s$.

Definition 5. *A directed spanning tree is a subgraph $(\mathcal{V}^s, \mathcal{E}^s)$ of the directed graph $(\mathcal{V}, \mathcal{E})$ such that the subgraph $(\mathcal{V}^s, \mathcal{E}^s_N)$ has a directed tree and the subgraph $(\mathcal{V}^s, \mathcal{E}^s)$ must contain all the nodes of graph $(\mathcal{V}, \mathcal{E})$ i .e. $\mathcal{V}^s = \mathcal{V}$.*

It is to note that a directed graph $\mathcal{G}$ has a directed spanning tree if and only if there exists at least one node which has a directed path to all other nodes. In the case of an undirected graph, having an undirected spanning tree is equivalent to a connected graph. The directed graph shown in Figure 1.5 has more than one directed spanning tree. Nodes 1 and 2 are two separate roots since both have at least one directed path to all other nodes.

1.3.2 Adjacency matrix

The structure of a graph can be represented by a matrix called adjacency matrix or connectivity matrix. Adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{N \times N}$ of a directed graph $\mathcal{G}$ is defined such that $a_{ij} > 0$ if

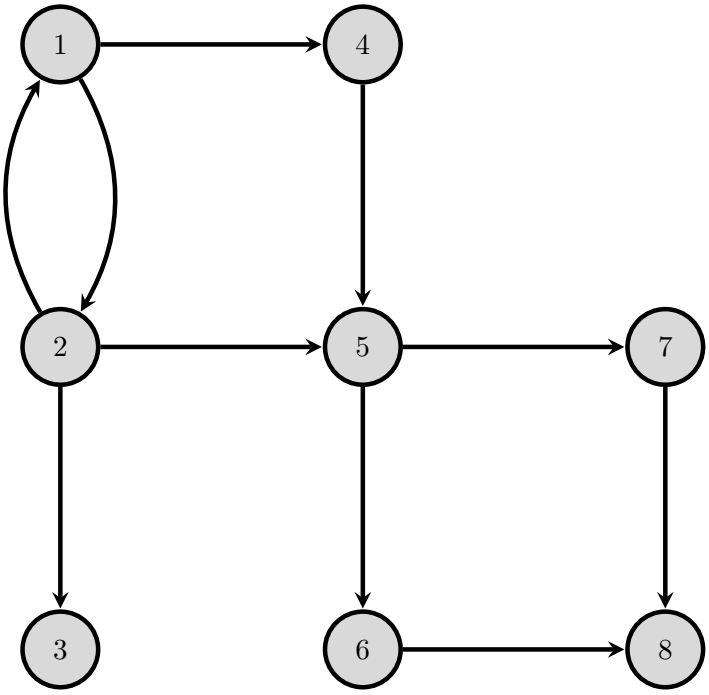


Figure 1.5: A directed graph with multiple directed spanning trees

$(v_j, v_i) \in \mathcal{E}$ and $a_{ij} = 0$ if $(v_j, v_i) \notin \mathcal{E}$. In other words, if agent v_j can receive information from agent v_i then $a_{ij} > 0$ and zero otherwise. Self edges are not allowed i.e $a_{ii} = 0$ unless specified. The case of undirected graph is similar except that $a_{ij} = a_{ji}\ \forall i \neq j$ since $(v_i, v_j) \in \mathcal{E}$ implies that $(v_j, v_i) \in \mathcal{E}$. Through out this thesis we choose unweighted graphs which means $a_{ij} = 1$ if $(v_j, v_i) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise.

The adjacency matrix of the undirected graph in Figure 1.4a is given as

$$\mathcal{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

while the adjacency matrix of the directed graph in Figure 1.4b is given as

$$\mathcal{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Adjacency matrix of an undirected graph is always symmetric.

Definition 6. *A graph is balanced if* $\sum_{j=1}^{N} a_{ij} = \sum_{j=1}^{N} a_{ji}$ *for all* i.

Therefore, every undirected graph is balanced.

1.3.3 Laplacian matrix

Another important matrix that one may assign to a graph is the Laplacian matrix. It is a very useful matrix in the study of MAS and defined as

$$\mathcal{L} = [l_{ij}] \in \mathbb{R}^{N \times N} \tag{1.1}$$

where

$$\begin{aligned} l_{ij} &= -a_{ij}, \qquad \forall i, j = 1, 2, \ldots, N, i \neq j \\ l_{ii} &= \sum_{j=1}^{N} a_{ij}, \qquad j \neq i \end{aligned}$$

It is to note that if for any edge $(v_j, v_i) \notin \mathcal{E}$ then $l_{ij} = -a_{ij} = 0$. Matrix $\mathcal{L}$ satisfies the followings:

$$\begin{aligned} l_{ij} &< 0, \qquad \forall i, j = 1, 2, \ldots, N, i \neq j \\ \sum_{j=1}^{N} l_{ij} &= 0, \qquad i = 1, 2, \ldots, N \end{aligned}$$

Similarly to adjacency matrix, Laplacian matrix of an undirected graph is always symmetric. Laplacian matrix of a directed graph is not symmetric and often known as directed Laplacian matrix [13] or non-symmetric Laplacian matrix [14]. In the current thesis, Laplacian matrix of a direct graph is simply refereed as Laplacian matrix without any prefix. The Laplacian matrix of the undirected graph in Figure 1.4a is given below

$$\mathcal{L} = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 & -1 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & -1 & 0 & -1 & 3 \end{bmatrix}$$

while the Laplacian matrix of directed graph in Figure 1.4b is given as

$$\mathcal{L} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

Laplacian matrix can also be defined as $\mathcal{L} = A - D$, where $D = [d_{ij}]$ is the in-degree matrix of size $N \times N$ given as $d_{ii} = \sum_{j=1}^{N} a_{ij}$, $i = 1, 2, \ldots, N$ and $d_{ij} = 0$ for $i \neq j$.

Remark 7. *Laplacian matrix $\mathcal{L}$ always has zero row sum and 0 is an eigenvalue of $\mathcal{L}$ with associate eigenvector $\mathbf{1}_N$ where $\mathbf{1}_N$ is $N \times 1$ vector with all entries equal to 1. Moreover, all non-zero eigenvalues of $\mathcal{L}$ are positive.*

Remark 8. *For an undirected graph, 0 is simple eigenvalue of $\mathcal{L}$ if and only if the undirected graph is connected. For a directed graph, 0 is a simple eigenvalue of $\mathcal{L}$ if and only if the graph is strongly connected. In other words, if directed graph has at least one spanning tree, then 0 is a simple eigenvalue of $\mathcal{L}$.*

1.4 Consensus control problem

Consensus seeking is considered as one of the fundamental problems of cooperative control of MAS and that is why it has attracted great attention from researchers in the recent years. In consensus, agents are required to reach an agreement for some common feature. These common features may vary in accordance with applications and requirements. For instance, the feature of interest in the case of mobile robots could be positions and velocities or for UAVs, altitude could also be a feature of interest where all the UAVs may require to maintain a common height.

The consensus problem can be divided into two categories: leaderless consensus and leader-following consensus. To understand these two cases, we first introduce a leader.

Definition 9. *A **leader** is an agent which produces a reference state that represents a control objective or a feature of common interest for the other agents in the MAS. A leader could either be real or virtual.*

1.4.1 Leaderless consensus

A leaderless consensus is also known as consensus producing. If agents are not required to track any reference trajectory then the consensus problem is called leaderless consensus. The final consensus state in this case is inherent and generally depends on the initial states of the agents and the communication topology. Let us consider a MAS consisting of N agents whose dynamics evolves with the following differential equation:

$$\dot{x}_i(t) = F_i\left(x_i(t), u_i(t)\right), \qquad i = 1, 2, \ldots N \tag{1.2}$$

where $x_i(t) \in \mathbb{R}^n$ and $u_i(t) \in \mathbb{R}^p$ represents the state and the input of agent i respectively. The leaderless consensus control then can be defined as follows.

Definition 10. *The **leaderless consensus** or consensus producing is achieved by the MAS if for all* $x_i(0)$ *and for all* $i, j = 1, 2, \ldots N$, $\|x_i(t) - x_j(t)\| \to 0$ *as* $t \to \infty$.

In the consensus control problem, the goal is to design a distributed control protocol that drives all or a few states of the agents such that they reach some common value as described in Definition 10. The final value at which the states of the agents reach is called a consensus value. Let the agents in the MAS have simple single-integrator dynamics as described below

$$\dot{r}_i(t) = u_i(t), \qquad i = 1, 2, \ldots N \tag{1.3}$$

where $r_i(t), u_i(t) \in \mathbb{R}^m$ with $m \in \mathbb{N}$ are the agent position and control input respectively, then a trivial distributed control input for the i^{th} agent can be designed as below [15–17].

$$u_i(t) = -\sum_{j=1}^{N} a_{ij} K[r_i(t) - r_j(t)], \qquad \forall i, j = 1, \ldots, N, j \neq i \tag{1.4}$$

where a_{ij} is the ij^{th} entry of the corresponding adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$ and K is a positive gain.

Example: Let us consider a MAS consisting of 4 agents. The initial conditions of the agents are $x_1(0) = -4$, $x_2(0) = -2.5$, $x_3(0) = 1$ and $x_4(0) = 3$. Two possible communication networks are shown in Figure 1.6. The corresponding consensus results are shown in Figure 1.7. From these results, it can been seen that the final consensus value depends on the initial condition of the agents and the communication topology. However, the final consensus value is constant in any case.

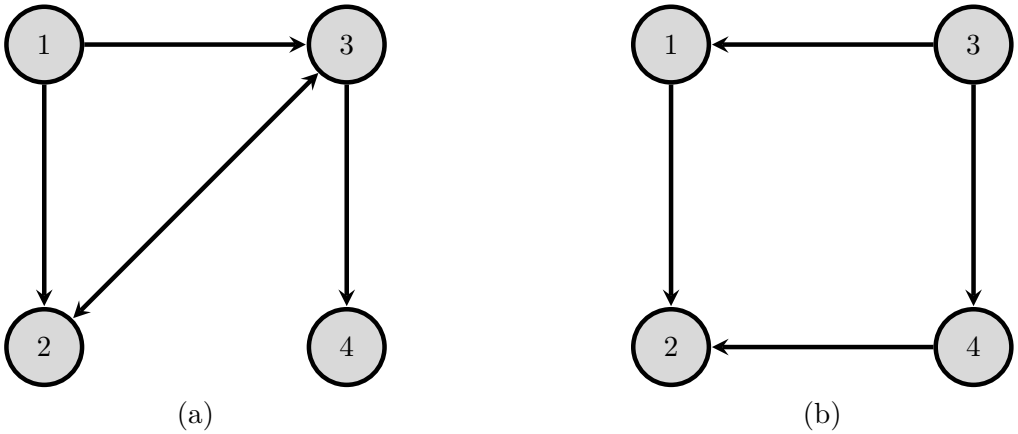


Figure 1.6: Communication topology for leaderless consensus

Problem of leaderless consensus for single integrator MAS has remained a focus of interest for the research community and has been investigated from various aspects. One of the most important conclusion from these studies for directed fixed topology is that the MAS achieves consensus if the corresponding communication graph has a directed spanning tree [18]. Olfati-Saber and Murray studied the average consensus problem of MAS with both fixed and switching topologies [15]. They also

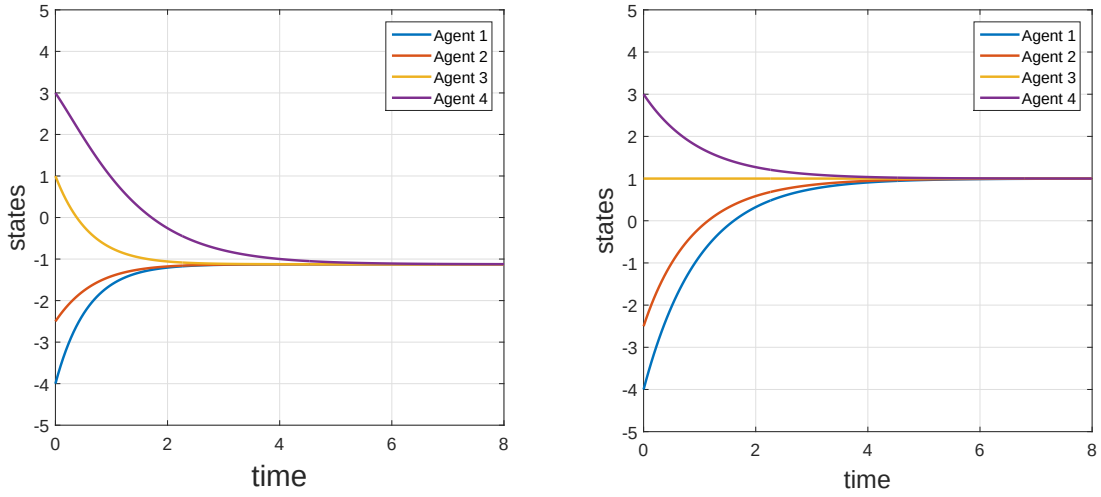


Figure 1.7: Leaderless consensus

investigated the problem of time delays in the communication network. Authors in [19] also proposed an algorithm to deal with switching topology and time delays. The issue of constant and time-varying delays in the communication of state information has been discussed in [20]. Sufficient conditions for the consensus of heading angles of the agents with undirected switching topology are shown in [16]. Ren and Beard presented consensus algorithms with more relaxed conditions as compared to [16] and [15] for both continuous and discrete-time first order MAS [18]. Fixed-time consensus protocols for single integrator MAS can be found in [21, 22].

A broad class of MAS consists of mechanical mobile agents. The basic equation of motion for such agents normally requires double integrator dynamics [23, 24]. For example, physical systems like robotic vehicles can be represented as linearized double integrator dynamic systems where position and velocity are the information states [15, 16]. Double integrator dynamics of i are presented as

$$\begin{cases} \dot{r}_i(t) = v_i(t), \\ \dot{v}_i(t) = u_i(t) \end{cases} \tag{1.5}$$

where $r_i, v_i, u_i \in \mathbb{R}^m$ represent respectively the position, velocity and control input of the i-th agent ($m \in \mathbb{N}$). Unlikely leaderless consensus of MAS with single integrator dynamics, double integrator dynamics might have a time-varying (dynamic) consensus value. Therefore, the dynamics of the system also plays an important role in leaderless consensus. A simple distributed leaderless consensus controller for double integrator MAS can be designed as follows [25, 26]

$$u_i(t) = -\sum_{j=1}^{N} a_{ij}[K_1(r_i(t) - r_j(t)) + K_2(v_i(t) - v_j(t))] \tag{1.6}$$

for all $i, j = 1, \ldots N$ and $i \neq j$, where K_1 and K_2 are the positive gains. The consensus is said to be reached if $r_i(t) \to r_j(t)$ and $v_i(t) \to v_j(t)$ as $t \to \infty$.

Due to the vast applications of second-order MAS, it has been widely studied in the literature for different perspectives. For instance, Xie and Wang, in [27], extended the single-integrator consensus protocol of [15] for double-integrator MAS with undirected communication graph. Sufficient and necessary conditions associated with the communication topology and Laplacian matrix for a general consensus protocol for MAS with double integrator dynamics are obtained in [28, 29]. It has been shown that how the real and the imaginary parts of eigenvalues of Laplacian matrix plays a vital role in achieving consensus. Ren and Atkins discussed MAS consensus protocols in [25] and showed that condition of spanning tree is necessary rather than sufficient in case of double integrator dynamics.

Consensus algorithms for double integral agents with heterogeneous inertias are proposed in [30, 31] and it was further improved by introducing heterogeneous controller gains by Mei *et al.* in [23]. Authors in [24] proposed a distributed consensus algorithm for double-integrator MAS that relies on relative position and absolute velocity of the agent. The final consensus is achieved with zero final velocity of all the agents.

The case of switching topology with arbitrary communication delays has been investigated in [32]. Conditions for reaching consensus are derived in the form of matrix inequalities through Lyapunov functions with delay partitioning. Leaderless consensus for switching topology is also studied by Ren and Beard in their article [18]. Consensus with switching topologies for nonlinear double-integral dynamical system has been investigated in [33] where a transformation is applied to convert the consensus problem into a stabilization problem of a switched system. Sliding-mode-control based consensus algorithm for second-order MAS is presented by Yu *et al.* in [34]. Results of fixed and finite time leaderless consensus of second-order MAS can be found in [35–38], to name a few. Similarly, the leaderless consensus problem for higher-order integrator MAS is also well investigated e.g [39–42].

1.4.2 Leader-following consensus

Contrary to consensus producing, in various practical applications, it is required that the agents follow a constant or time-varying reference trajectory, where the reference trajectory is produced by a leader (real or virtual). Therefore, it is known as leader-following consensus or consensus tracking. The leader is generally labeled as 0. Let us assume that the leader dynamics are

$$\dot{x}_0(t) = F_i\left(x_0(t), u_0(t)\right) \tag{1.7}$$

where the leader input $u_0(t)$ can be designed to achieve any desired time-varying reference trajectory depending upon the application. In the leader-following consensus, the objective is not only to ensure that each agent of MAS reaches consensus on a common state but that common state must also converge to the reference trajectory produced by the leader.

Mathematically, the leader-following consensus can be defined as follows.

Definition 11. *The **leader-following consensus** is said to be achieved by the MAS if for all $x_i(0)$, $\|x_i(t) - x_0(t)\| \to 0$ as $t \to \infty$ for all $i = 1, 2, \ldots, N$.*

In distributed leader-following consensus control, the leader's state information is only available to a small group of the agents. It is a proven fact that to achieve leader-following consensus for a directed communication network, the corresponding graph must have a directed spanning tree with the leader as a root. A trivial continuous-time distributed leader-following control law for a single-integrator dynamic MAS can be be designed as follows:

$$u_i(t) = -\sum_{j=1}^{N} a_{ij}[r_i(t) - r_j(t)] - b_i[r_i(t) - r_0(t)], \qquad j \neq i \tag{1.8}$$

where b_i is the i^{th} diagonal entry of **pinning matrix** $\mathcal{B} = diag\{b_1, \ldots, b_N\}$ and $b_i = 1$ if agent i can receive information from the leader and zero otherwise.

Definition 12. *A communication topology which contains the leader and the followers and one or more followers are pinned to the leader (receive the leader's data) is called pinning joint communication topology.*

Consider a MAS with 4 followers and one leader. Figure 1.8 shows four different cases of communication network between the leader and the followers. In case (a), $b_1 = 1$ and $b_j = 0$, $\forall j \neq 1$ since the leader can only transmit its state information to follower 1. In case (b), the leader can send information to follower 4 hence $b_4 = 1$ while $b_j = 0$, $j \neq 4$. The leader can communicate to more than one followers as shown in case (c) where the leader is connected to follower 1 and 2. Therefore, $b_j = 1$ for $j = 1, 2$ and $b_j = 0$ for $j = 3, 4$. Case (d) is a special case where $b_j = 1$, for all j corresponding to

the topology where leader can send its information to all the followers. Case (d) does not fulfill the core objective of distributed control as leader is acting like a central unit communicating to all the other agents and if the followers do not communicate among them, they can reach the the leader's trajectory.

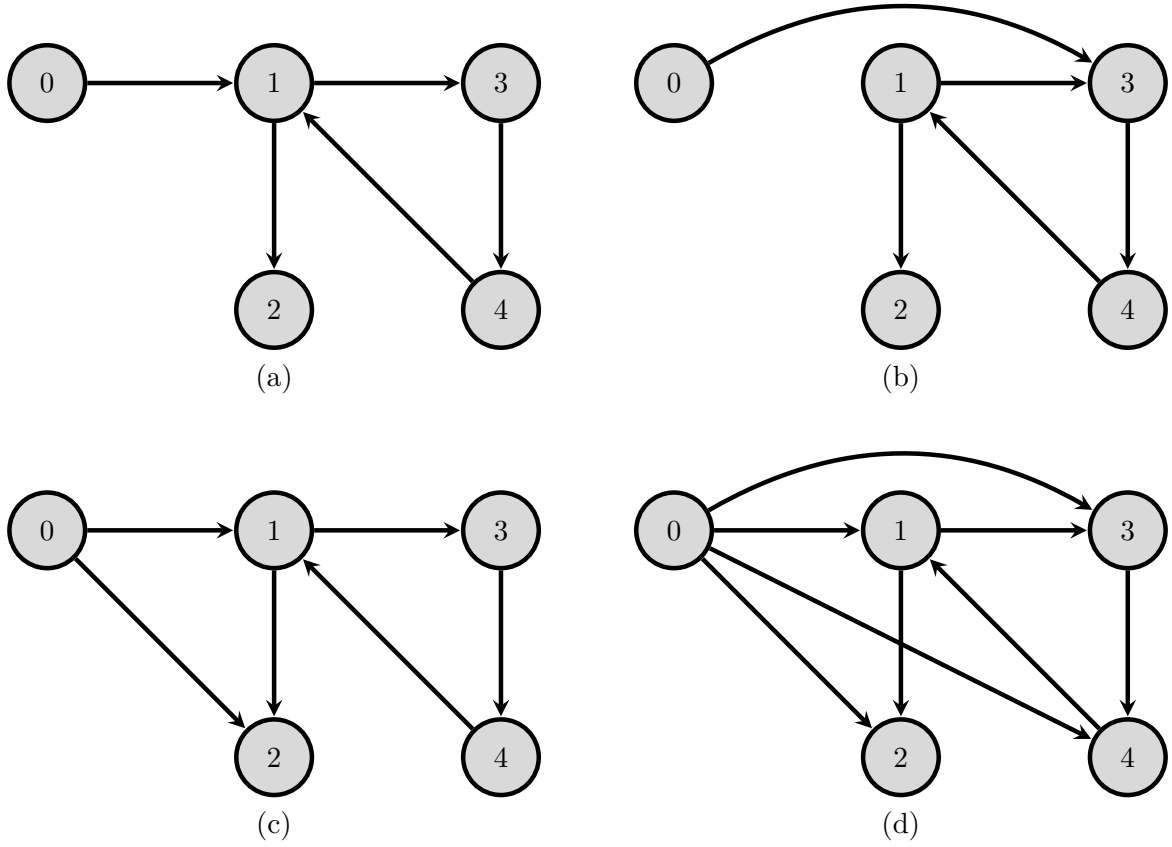


Figure 1.8: Communication topology among 4 followers and a leader

In [43], a leader-following algorithm for second-order system with time-varying leader velocity is given as below:

$$\begin{aligned} u_i(t) &= \frac{1}{\kappa_i}\sum_{j=1}^{N} a_{ij}\left[\dot{v}_j(t) - K_1(r_i(t) - r_j(t)) + K_2(v_i(t) - v_j(t))\right] \\ &\quad \frac{1}{\kappa_i} b_i\left[\dot{v}_0(t) - K_1(r_i(t) - r_0(t)) + K_2(v_i(t) - v_0(t))\right] \end{aligned} \tag{1.9}$$

for $\forall i, j = 1, \ldots, N$, $i \neq j$, a_{ij} is the ij^{th} entry of the corresponding adjacency matrix, $\kappa_i \triangleq \sum_{j=1}^{N} a_{ij} + b_i \neq 0$ and $K_1, K_2 > 0$. It is to note that in the above control law, each agent not only requires the position and velocity of its neighbors but also the derivative of the velocity which, in fact, is the input of the neighbor (i.e. $\dot{v}_j = u_j$). Therefore, the above algorithm can only be used for leader-following consensus if full states and input of the agent is available to the neighbors. Figure 11 illustrates leader-following results for all four cases of communication topology shown in Figure 1.8 using the above control law (1.9). It can be seen that in all cases, the followers track the leaders trajectory. The only condition for a communication topology is to have a directed spanning tree with the leader as a root.

The leader-following consensus problem for MASs with general single, double, and higher order integrator has been extensively studied in last few years. In [44], a first-order consensus seeking algorithm was modified to track a time-varying trajectory of the leader. Hong *et al.* proposed a leader-following algorithm for MAS where the followers dynamics are governed by a first-order integrator

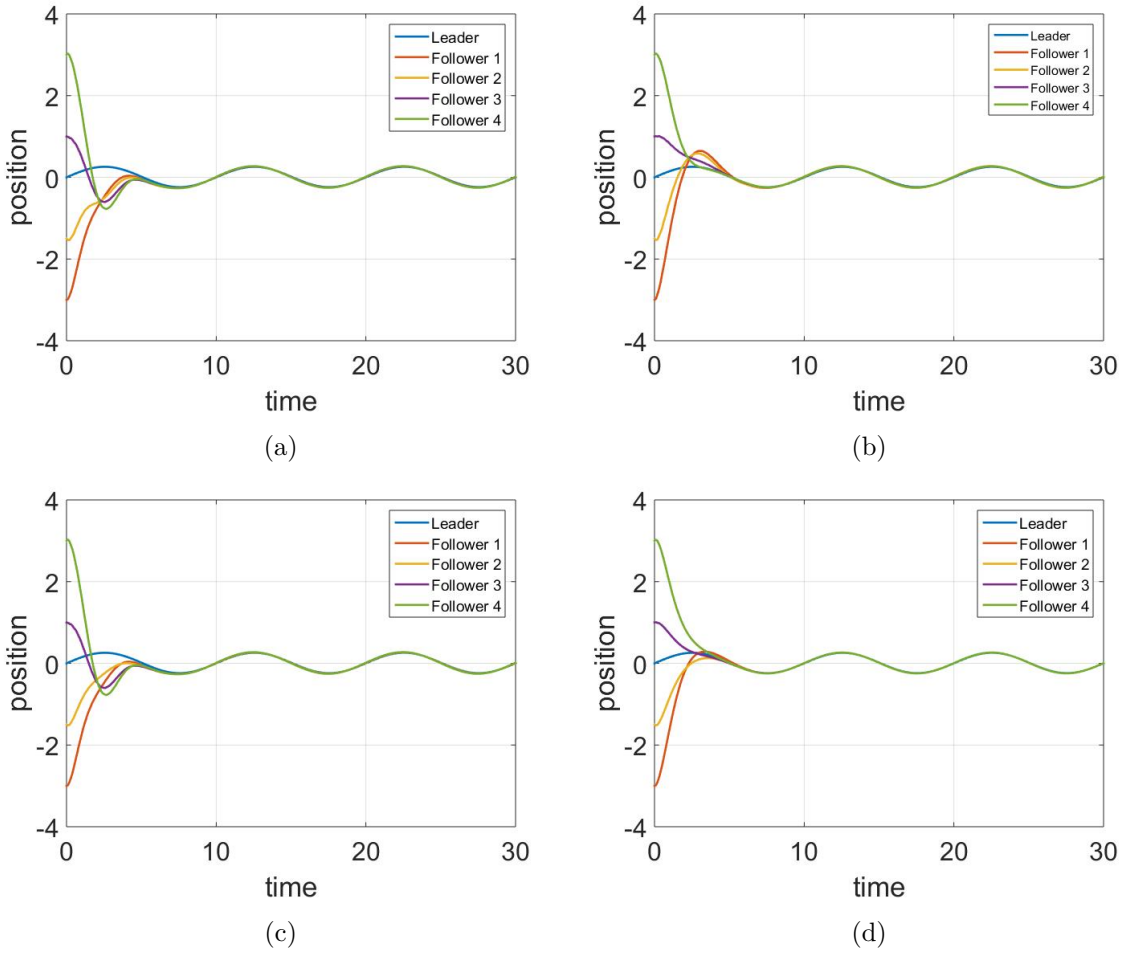


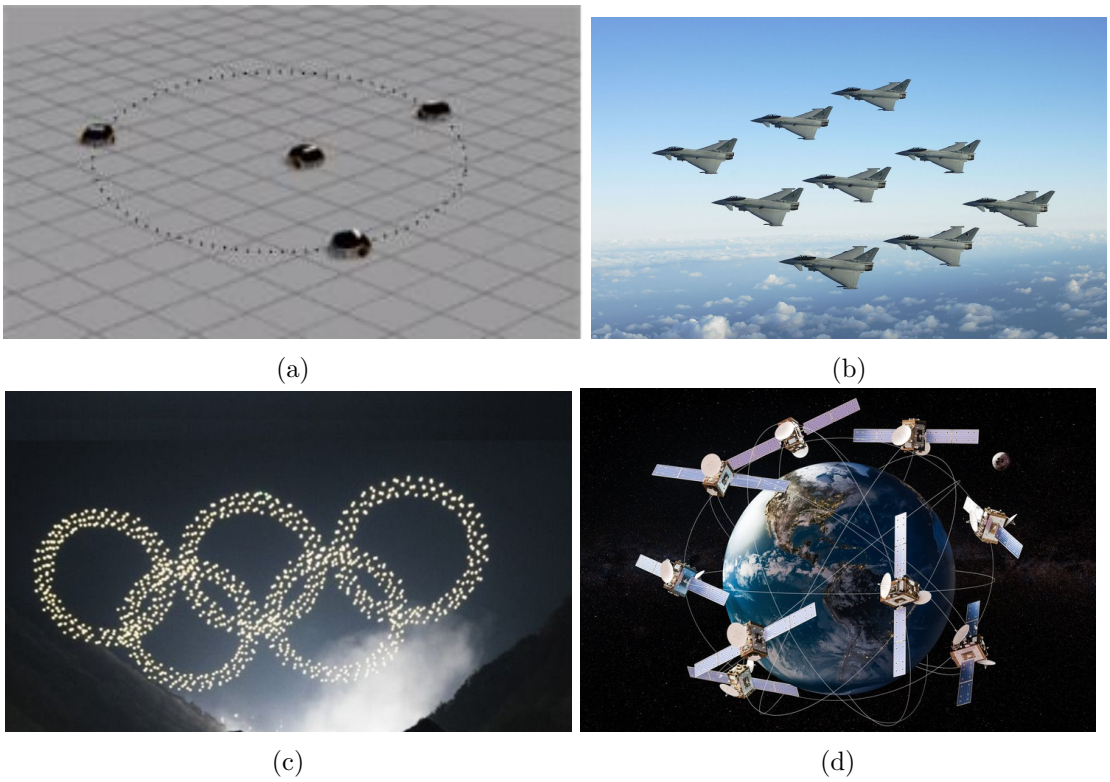
Figure 1.9: Leader-following consensus

while the leader is active with second order dynamics [45]. The results of [45] were further extended for second-order followers' dynamics in [46]. However, it is considered that the input (acceleration) of the virtual leader is available to all the agents. This condition is relaxed in [47] where the authors proposed virtual structure based approach. The problem of switching topology with time-varying delays has been studied in [48] for a second-order MAS. Leader-following with higher-order dynamics agents has been investigated in [49] for both fixed and switching topologies. In [50], the authors proposed a leader-following control algorithm for general time-varying linear MASs with both fixed and switching topologies. Fixed-time leader-following consensus has been discussed in [51–53] while the leader-following problem for nonlinear MAS is investigated in [54–57] and references within. Similarly, researchers have also proposed leader-following consensus schemes for nonholonomic robots and UAVs [58–62].

1.5 Formation control problem

Formation control is another important area of interest in the domain of MAS. It refers to the problem of controlling positions, velocities and/or orientation such that they produce a desired geometric shape. Formation control has various applications in exploration and mapping, environment monitoring, satellite communication, secure surveillance, military, heavy payload transportation and entertainment etc.

The first step in formation control problem is to select a desired geometric pattern while keeping in mind the physical constraints of the agents. The formation shape can either be fixed or time-varying depending on the application. Like consensus control, formation control can be separated into two



(a) (b)

(c) (d)

Figure 1.10: Applications of formation control of MAS (a) mobile robots encircling the leader (b) fighter jets formation for defence and surveillance (c) drones making Olympic rings (d) satellites formation to cover maximum earth coverage

classes based on whether the formation has any reference trajectory or not, called formation producing and formation tracking [63].

- In **formation producing**, the agents are only required to produce the desired geometric shape and maintain it for all future time.
- On the other hand, in **formation tracking**, the agents are not only required to make the desired shape but also track a reference trajectory produced by a leader while maintaining the formation.

A simple example of formation producing and formation tracking is shown in Figure 1.11.

In the literature, formation control strategies are also characterized based on agent sensing capability and interaction topology as mentioned in a survey by Oh *et al.* [64]. Note that these approaches are not characterized based on centralized of distributed but rather the characterization is based on the variables which are sensed and controlled by the agents to achieve the formation. Three very common approaches are:

- **Position-based approach:** [65–68] Agent measures it own position through the sensor in a global frame and controls the position actively to converge to the desired position which is also defined in the global frame. The desired formation is described by specifying the desired position of each agent. Hence, for a general position-based approach, interaction among the agents is not be an obligation since the formation can be achieved through individual position control of an agent [69]. However, neighbors' position information can be used to enhance performance of the control scheme and to achieve additional goals. The main drawback of this approach is that the agents require advance sensors to measure their position accurately in the global frame and sometime require a prior trajectory to be evaluated for each agent.
- **Displacement-based approach:** [70–72] The formation is defined by the desired displacement of the agents in a global frame. The displacement of the agent is controlled under the assumption

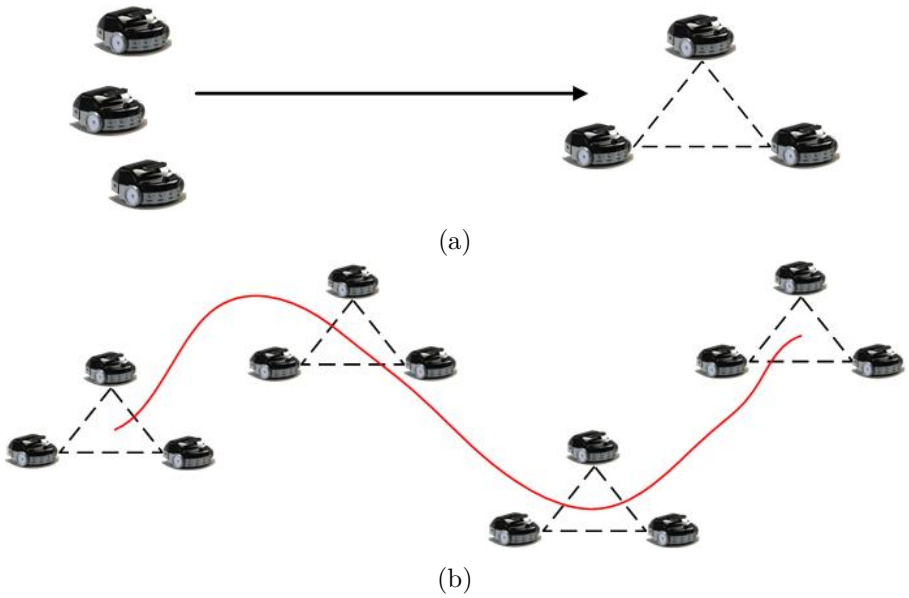


Figure 1.11: Formation control (a) formation producing (b) formation tracking

that the agent can sense the relative distance of the neighbors in the global frame. No absolute position is required but the orientation of the global frame is needed to be known. This approach is moderate in term of interaction topology and sensing capability.

- **Distance-based approach:** [66, 70, 73–75] In this approach the formation pattern is specified by inter-agent distance. The inter-agent distance is actively controlled to achieve a desired formation. The agents need to sense the neighbors' relative distance in their local frames which is not necessarily aligned with the local frames of other agents. Therefore, the disadvantage of distance-based approach is that the orientation and translation of the formation are not unique anymore.

Researchers have also distinguished formation control techniques based on leader-follower architecture [76–82], virtual structure [64, 83–88], and behavioral control [89–92]. In leader-follower architecture, one agent acts like a leader and the other agents follow the leader's position with some predefined offset. The formation is considered as a single virtual structure object in virtual structure based approaches. The required motion of the whole virtual structure is provided and then the movement of the agents is decided based on the virtual structure. Distance-based control and displacement-based control are examples of virtual structure. Sometimes a leader-follower approach and virtual structure approach are combined to drive the formation to a desired target. In behavioral-based formation control, various behaviors of the agents are defined. Such behaviors might incorporate collision and obstacle avoidance or cohesion. Formation is not explicitly defined in the behavioral-based approach rather the prescribed behavior describes the formation.

Consensus-based formation control design is another interesting approach especially when distributed control is the main objective. According to the definition of consensus, agents converge to a common desired value. By choosing appropriate deviation from the consensus state, a consensus algorithm can be applied to achieve a desired formation. Moreover, all the above mentioned formation control techniques can be unified and incorporated in consensus-based formation control [93, 94] by choosing appropriate consensus state and they usually provide a more reliable and robust solution even if some of the agents are subject to a failure [95]. Various consensus-based formation control strategies have been proposed in the literature. For instance, Ren proposed a consensus based formation controller for a second-order MAS [93]. [96] presents a distributed algorithm for formation flight of UAVs. The algorithm is based on feedback linearization and consensus protocol along with the failure detection logic. The proposed algorithm only deals with formation producing. The consensus-based formation producing problem for UAVs has also been studied in [97]. A time-varying formation

tracking algorithm for second-order MAS with switching topology is proposed by Dong *et al.* in [94]. The proposed algorithm has been applied on a team of quadrotors to enclose a target in xy-plane. Liu *et al.* have proposed a formation control technique which deals with the problem of input saturation [98]. They use low gain feedback and Lyapunov function theory to prove that with their proposed algorithm, semi-global formation tracking can be achieved even in the presence of switching topology and input saturation. Consensus-based formation tracking laws for various communication topologies which include spanning tree shaped, complete and ring shaped are presented in [99] for the systems whose kinematics evolve on Lie group. Mondal *et al.* combined collision avoidance and consensus approach to propose a distance-based formation controller [72]. Consensus-based formation tracking control for a team of mobile robots has been presented in [100]. The authors provided a transformation to convert the formation control problem into a trivial consensus problem. A consensus-based, leader-follower structured controller has been proposed by Manoharan and Chiu in [101] for formation tracking of automatic vehicles. [102] discusses a nonlinear formation controller for a team of mobile robots. A distributed observer has been used to estimate the reference trajectory as it is considered that the reference state is not continuously available.

1.6 Issues and challenges in distributed cooperative control design

Despite the fact that a lot of research has been done in the field of cooperative control, there are still many open problems that need to be addressed, especially when it comes to practical applications. Designing a distributed control law for real world scenarios requires various factors to be taken into account which pose serious challenges. Most of these challenges are related to restricted communication capabilities as well as hardware and physical limitations of the agent.

Communication constraints: In most of the previously mentioned literature on consensus and formation control problems, it is considered that each agent in the MAS receives its neighbors' information in continuous time. However, it requires infinite bandwidth to transmit continuous data. Additionally, due to digital nature of the of communication equipment, the data is always communicated in discrete-time [103].

Several control techniques to deal with sampled data issues in cooperative control have been proposed by the research community. For instance, Pan and Qiao proposed a discrete-time consensus algorithm for double integrator MAS in [104] where communication delay was considered and conditions for consensus under such delays were derived using linear matrix inequalities. Several sufficient and necessary conditions on controller gain, communication topology and sampling time were proposed in [105, 106] for the convergence of sampled-data coordination protocols for agents with double integrator dynamics. In [107], average consensus was achieved asymptotically for a dynamic network provided that there always exists a directed spanning tree. Integral sliding mode control based consensus algorithm was proposed in [108] for both leaderless and leader tracking of second-order MAS with disturbance. The authors in [109] investigated a discrete-time consensus algorithm for second-order MAS. It was shown that agents achieve consensus in the presence of switching communication topology with non-uniform time delays and sampled data. Eichler and Werner discussed the optimization of convergence speed for only fixed communication topology [110]. A leader-following protocol for second-order MAS in a sampling setting with Markovian switching topology has been presented in [111]. Wu *et al.* derived mean square consensus tracking with a virtual leader in [112]. In this paper, the tracking error not only depends on the sampling period and delay but also on the velocity and acceleration of the virtual leader. The problem of intermittent communication was addressed by Liu *et al.* in [113]. The authors proposed a consensus algorithm based on persistent-hold techniques for first order systems.

While the above mentioned sampled data techniques consider that the system has a constant sampling rate T i.e. the data is transmitted at regular time intervals, in real world applications, irregular and nonuniform sampling rates are inevitable due to various factors like time delays, packet loss etc. Moreover, exact clock synchronization of the agents is also impossible. Therefore, in practice, the agents transmit their data asynchronously and the transmission is totally independent of other agents. Asynchronous transmission can prove to be useful as well. For example, less frequency

bandwidth is required when agents send data at different instants from the other agents. Inspired by this, researchers have suggested event-triggering based control algorithms in which the information is transmitted by an agent only when it is required. This is achieved by defining some conditions that will trigger the communication hence the name, event-triggering based control. A huge amount of literature is available on this type of distributed cooperative control schemes. In [114], Dimarogonas *et al.* proposed an event-triggering based distributed control for MAS. The next triggering time is calculated when a particular event takes place. An event-triggering based algorithm with a state-dependant triggering function was proposed in [115]. An observer based event-triggering consensus protocol was proposed in [116–118] where the control law is updated using the estimated states. Other examples can be found in [119–122] and the references inside them. One of the main difficulties in such control techniques is to determine the triggering function which can ensure the quality of task completion by MAS.

Another problem which is not considered in most of the existing literature on cooperative control of sampled-data MAS is related to the nature of the control input. Generally, the control input of the agent is kept constant between two sampling instants. However, time-varying control input is more advantageous and can be taken into consideration since the transmitted data has to be sampled not the control input.

Agent sensing constraints: Another common assumption in MAS distributed control design is that the full states of the agents are available. Whereas, in practice, it is not always easy to measure every state of an agent due to limited on-board resources. Therefore, some states are not accessible for the agents. For instance, to reduce the cost and size of a robot, it is usually equipped with only position sensors like the odometer. Hence the velocity state cannot be measured directly. Moreover, even if both position and velocity states are available, transmitting them will also require more communication resources as compared to sharing only position states. However, in order to design an effective state feedback control law to steer the robot to the desired path, both position and velocity states are required. Such scenarios give rise to the need for state estimators for the dynamic systems to achieve desired goals in safe and economical manners. Unavailable states can be reconstructed through state observers which use available output data. State observers are discussed in detail in the next section.

Switching topology: Usually it is considered that the communication topology among the agents of a MAS remains fixed during the cooperative process. However, in many real applications, it is sometime not practically feasible for the agents to throughout maintain a fixed topology. This could be due to various reasons like transmission link failure, communication rage limitation or collision avoidance etc. Therefore, switching topology is more suitable option in these scenarios. Various cooperative control techniques for MAS with switching topology are studied in literature for instance in [94, 98, 123, 124]. However, these research work mostly do not consider the above mentioned communication and sensing constraints.

Collision among agents: Inter-agent collision is another important issue in MAS cooperative operation. Agents can be damaged due to collision and this could be catastrophic as well. Therefore, it is highly essential to ensure collision avoidance when agents are performing some cooperative task. For mobile physical agents, position consensus may not be suitable as agents will collide with each other when they reach the final consensus state. In formation producing and formation tracking it is clear that an agent will not collide if the formation shape is chosen with appropriate inter-agent distances. However, agents may still collide while converging to the final shape from their initial positions. Collision avoidance during transient can be managed by incorporating some intelligent mechanism along with the formation controller to keep a minimum distance between the agents.

1.7 State observers

A state observer estimates the state variables from the available information. Observers were first developed by Luenberger more than half a century ago [125]. To understand the basic design of the Luenberger observer, let us consider a continuous time linear time-invariant system with single agent

(also known as a plant). The dynamics of the plant is given as

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{1.10}$$

where $x \in \mathbb{R}^m$ and $u \in \mathbb{R}^n$ are state and input vectors while $y \in \mathbb{R}^p$ is the output vector and m, n and $p \in \mathbb{N}$. A, B and C are state, input and output matrices of appropriate size respectively. A typical linear observer uses the input and the output of the system to estimate the system states. An observer can only be developed if the system is observable. For a linear system with n states, the system is observable if the observability matrix $O = [C, CA, CA^2, \ldots, CA^{n-1}]^T$ has row rank equal to n. A simple linear observer for state estimation can be given as

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x} + Bu + L(y - \hat{y}) \\ &= A\hat{x} + Bu + LC(x - \hat{x}) \end{aligned}$$

where $L \in \mathbb{R}^n$ is the observer gain matrix and estimated output $\hat{y}$ is

$$\hat{y} = C\hat{x}$$

The estimation error dynamics can be be written as follows

$$\begin{aligned} \dot{\tilde{x}} &= x - \hat{x} \\ \dot{\tilde{x}} &= (A - LC)\tilde{x} \end{aligned}$$

If L is selected such that the real part of the eigenvalues of $A - LC$ are negative then the estimation error will converge to zero as $t \to \infty$ which implies that $\hat{x} = x$.

These kinds of observers work very well only if the exact system model is available as it can be seen from the observer design. However, in practice, exact system modelling is not possible. In the case of modelling uncertainty, choosing observer gain L such that $A - LC$ becomes Hurwitz may not lead to estimation error convergence. To cope with this problem, researchers have proposed high gain observers [126, 127]. High gain observers became more famous to reconstruct the states of nonlinear systems [128, 129] since the non-linearity can be considered as uncertainty [130]. The gains of the observer are chosen to be high enough (therefore named high gain observer) to ensure that the uncertain terms vanish with time. High gains not only attenuate the uncertainty but also make the observer dynamics fast.

In practical applications, the output measurements are usually discrete rather than continuous i.e one gets output y only at sampling instants t_k with $k = 1, 2, \ldots$. Therefore, continuous time observers are not quite useful for real applications especially when the sampling time of the output is comparatively high. Moreover, irregular and asynchronous sampling rate is also inevitable in real world scenarios. To deal with these issues appropriately, continuous time observers are required to be redesigned for discrete output measurement. This leads to the development of continuous-discrete time observers. Earlier works on continuous-discrete time estimation can be found in [131] where observer of [128] is extended to drive continuous-discrete time Kalman filtering. A constant gain continuous-discrete time observer for uniformly observable systems is presented in [132]. The constant gain is calculated through stationary discrete Lyapunov equations. A continuous-discrete time observer based on a simple model for a complex system is proposed by Astorga *et al.* in [133]. The designed observer is applied on chemical reactors. In the aforementioned continuous-discrete time observer, state prediction is provided between two consecutive intervals through a dynamical system which is similar to the underlying system. The state prediction is updated when output measurement is available at the sampling instant. Raff *et al.* proposed an impulsive observer design for a specific class of nonlinear systems with nonuniform sampled output data [134]. In this approach, the correcting term is expressed in the form of the difference between the estimation and the last output sample. The correcting term is multiplied with a constant gain and only updates when an output sample is available. The conditions for gain and observer convergence analysis are obtained through Linear

Matrix Inequalities (LMI) method. [135] presents another interesting approach for continuous-discrete time observers where an output predictor is used to predict the system output when it is not available between two samples. Then this output predictor is coupled with a continuous time observer as shown in Figure 1.12. Ordinary differential equations have been used to predict the output between samples with the last measured output as the initial condition.

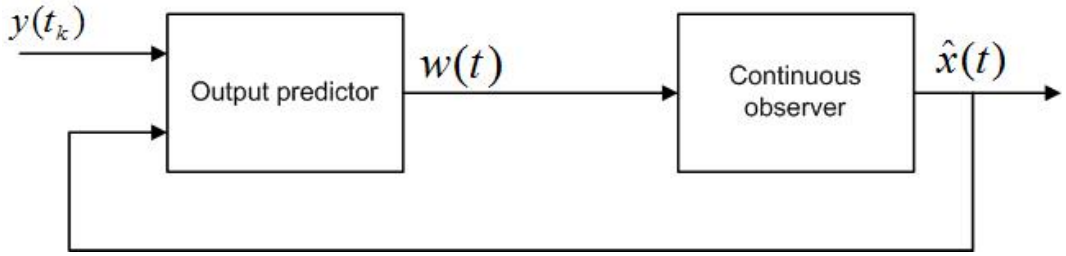


Figure 1.12: Sampled data observer using output predictor

Combining the concepts of [134] and [135], authors of [136] proposed a continuous-discrete time controller for a Multi-input-Multi-Output (MIMO) system with Lipschitz non-linearities and variable sampling period. The considered system can take the following form

$$\begin{aligned} \dot{x}(t) &= Ax(t) + \phi(u(t), x(t)) \\ y(t_k) &= Cx(t_k) \end{aligned}$$

with $x = [x_1, x_2 \dots x_q]^T \in \mathbb{R}^n$ with $x_i \in \mathbb{R}^p$ for $i = 1, \dots q$ and $C = [I_p, 0_p \dots 0_p]$ of compatible size. ϕ is a triangular shaped function and output $y(t_k) = x_1(t_k) \in \mathbb{R}^p$. Instead of using a fixed observer gain like in [134], a time-varying and sampling time dependant gain is introduced. The proposed observer dynamics are given as follows

$$\dot{\hat{x}}(t) = A\hat{x}(t) + \phi(u(t), \hat{x}(t)) - \Delta_\theta^{-1} K e^{-\theta K_1 (t - t_k)} \left(C\hat{x}(t_k) - y(t_k)\right)$$

where $K = [K_1, \dots, K_q]^T$ is selected such that matrix $A - KC$ is Hurwitz. $\Delta_\theta = diag\{I_p, \frac{1}{\theta} I_p \dots \frac{1}{\theta^{q-1}} I_p\}$ with design parameter $\theta \geq 1$. It has been shown through Lyapunov analysis that if sampling periods remain below a certain bound, the observer error will converge exponentially. The use of exponential time-varying gains for the estimation and stabilization of a variable sampling time output system can also be found in [137–139].

Using a state observer to reconstruct the unknown states of the agent and its neighbor in MAS is naturally a promising choice. Distributed observers for MAS have been well investigated in the literature. Mostly, the motivation behind such approaches is to modify and expand the conventional observer design methods to obtain distributed observers. For instance, a traditional linear observer is used to reconstruct an agent's own unknown state in [140] and the estimated states are then transmitted to the neighbors. In [46] an observer based leader-following protocol is proposed for a continuous time MAS with second order agent dynamics. It is considered that the leader only has a position sensor and its velocity cannot be measured. Furthermore, it is considered that the leader input is some common policy known by each agent. A reduced order (first-order) continuous time observer is designed for an agent to estimate the unknown leader velocity. A high gain sliding mode based observer has been proposed in [141] to estimate the unknown velocity of the leader in finite time. The estimated velocity is then used to design finite-time cooperative control law for leader tracking. Authors in [142] studied consensus problem of double-integrator MAS when the velocity information is unavailable. Observers have been used to reconstruct the velocity to be used for a feedback consensus controller. Hu *et al.* studied an observer-based consensus protocol for nonlinear MAS [143]. Two observers are designed, a local observer from local output information of the agent and a distributed observer through the received output information of the neighbors. A few other examples of observer-based distributed cooperative controls can be found in [144–146]. Observer-based cooperative control protocols for MAS with switching topology are discussed in [147–150]. A position estimation-based

formation controller with collision avoidance capability is presented in [151]. The proposed algorithm is tested on a group of small mobile robots.

The above mentioned papers consider that the information is available in continuous time. However, for the cases where only partial states are accessible in discrete-time, only a few results are available in literature. A distributed observer is proposed in [152] to address the consensus problem of high order nonlinear MAS in which the only available data are sampled and delayed outputs. Du *et al.* studied consensus problem for MAS with stochastic sampling output [153]. It is assumed that the sampling period switches strictly between only two values. Observer-based event-triggering protocols for sampled data MAS have been discussed in [154, 155]. The first attempt to solve the problem of leaderless consensus for double-integrator MAS with nonuniform sampling was made in [156]. A continuous-discrete time observer has been used to estimate the agent's and neighbors' state in continuous time from available data. However, the leader-following consensus and formation tracking of MAS with nonuniform and asynchronous sampling are still open problems.

1.8 Contribution of thesis

The aim of this thesis is to study distributed consensus and formation of MAS. It is clear from the discussion and literature presented in the previous sections that the available cooperative control schemes do not take various limitations into account. Motivated by this, the current thesis focused on the design and implementation of distributed cooperative control laws for a double-integrator MAS with communication and sensor constraints. These considered constraints are given below:

- Each agent can only measure its position state.
- Agents are not equipped with sensors to measure their velocity.
- Agents do not have access to the input (acceleration) of their neighbors.
- The measured state is transmitted to the neighbors at irregular and non-uniform time intervals.
- The transmission among the agents is asynchronous and totally independent of other agents in the network.
- The communication topology among the agents is directed.

The following control algorithms have been proposed for the MAS with above mentioned constraints:

- Firstly, a continuous-time distributed leader-following consensus algorithm is proposed for fixed communication topology. It is assumed that only a small portion of the agents have access to the leader position. A continuous-discrete time observer with time varying exponential gain has been used to estimate neighbors' and the agent's own position and velocity in continuous-time from the available sampled-data. These estimated states are then used to design the leader-following consensus protocol. A Lyapunov-based convergence analysis is carried out to achieve conditions for observer and controller parameters.
- The results of the designed leader-following controller for fixed topology is then extended for switching communication topology. A convergence analysis of the proposed controller is carried out which provides conditions for the switching rate between different communication graphs.
- In this thesis, we also study formation tracking algorithm for the MAS with above mentioned constraints. Both time-varying and fixed formation shape cases are investigated. The observer dynamics are also modified to deal with time-varying formation problems.
- Since inter-agent collision is another important issue in formation tracking control, a potential function based collision avoidance algorithm is incorporated with the proposed formation tracking controller. The collision avoidance algorithm ensures that the agents converge to produce the desired geometric shape without colliding with each other. Convergence analysis is based

on Lyapunov theory and it provides various conditions for the stability of the system with a proposed hybrid controller.

- Finally, all the proposed distributed control schemes are tested on a fleet of differential-drive mobile robots. Robot Operating System (ROS) has been used to implement algorithms on the robots. ROS is a complete operating system for robots which comes with very useful built-in libraries with high-level functionalities to operate a robot. Therefore, it eliminates the requirement of building robot software architecture from scratch.

  If the consensus algorithm is applied on all the states of the multi-robot network, it will cause a collision among the robots since they will converge to the same position. In order to avoid the collision in the consensus tracking experiment, we applied the proposed algorithm to obtain consensus only in the x-positions of the robots. This is achieved by restricting the motion of the robot in a a straight line in x-direction (i.e. 1-D) with some constant offset in their y-positions. By limiting the motion of the robot to 1-D, the nonholonomic constraints the nonholonomic constraints can be ignored and robots can be modelled as double integrator systems with position and velocity as states.

  A new control scheme, based on robot-flatness properties, is proposed to implement the designed second-order formation tracking algorithm with collision avoidance. Through the new proposed scheme, any double-integral control law can be applied on robots moving in a 2-D plane. Gazebo simulator is used for robot simulations as it provides a very realistic environment while taking into account various factors like collision and friction etc.

1.9 Thesis Layout

The remaining of this manuscript is organized as follows:

- **Chapter 2:** In this chapter, we investigate the problem of leader-following consensus for a double-integrator MAS with communication constraints. First an observer based distributed consensus protocol is proposed for fixed communication topology. Then the results are expanded for the case of switching topology.

- **Chapter 3:** This chapter comprises of two parts. In the first part, distributed formation tracking problem is discussed. A distributed consensus based formation tracking algorithm is presented to deal with both fixed and time-varying formation patterns. In the second part of the chapter, problem of collision avoidance is studied to achieve collision-free formation tracking of the MAS.

- **Chapter 4:** Application of proposed cooperative control algorithms on a multi-robot network is studied in this chapter. Experimental platform along with ROS based distributed robot network is presented. Moreover, control scheme for robots is also discussed. The efficiency of the proposed algorithm on real applications is investigated through simulations and hardware implantation results.

- **General conclusion and future work:** In this chapter, the obtained results are summarized and various possible future research directions are identified.

1.10 Scientific publications

International refereed Journals

- **Syed Ali Ajwad**, Tomas Ménard, Emmanuel Moulay, Michael Defoort and Patrick Coirault, "Observer based leader-following consensus of second-order multi-agent systems with nonuniform sampled position data", *Journal of the Franklin Institute*, Vol. 356 (16), pp. 10031-10057, 2019.

- Tomas Ménard, **Syed Ali Ajwad**, Emmanuel Moulay, Patrick Coirault and Michael Defoort, "Leader-following consensus for multi-agent systems with nonlinear dynamics subject to additive bounded disturbances and asynchronously sampled outputs", *Automatica*, Vol. 121, November 2020.

- **Syed Ali Ajwad**, Emmanuel Moulay, Michael Defoort, Tomas Ménard and Patrick Coirault, "Collision-free formation tracking of multi-agent systems under communication constraints: application to robotics", *IEEE Control Systems Letters*, 2020, (under review).

- **Syed Ali Ajwad**, Emmanuel Moulay, Michael Defoort, Tomas Ménard and Patrick Coirault, "Leader-following consensus of second-order multi-agent systems with switching topology and partial aperiodic sampled data", *IEEE Control Systems Letters*, 2020, (under review).

International Conferences

- **Syed Ali Ajwad**, Emmanuel Moulay, Michael Defoort, Tomas Ménard and Patrick Coirault, "Output-feedback formation tracking of second-order multi-agent systems with asynchronous variable sampled data", *IEEE* 58^{th} *Conference on Decision and Control (CDC)*, pp. 4483-4488, 2019.

Chapter 2

Leader-following consensus

This chapter is focused on consensus tracking of MAS under communication constraints. First, a continuous-discrete time observer based algorithm is proposed to achieve leader-following consensus in the presence of various communication constraints with fixed topology. These results are published in Journal of Franklin Institute [157]. In the second part of the chapter, the obtained results are extended to the case of switching communication topology. These results are submitted in IEEE Control Systems Letters [158].

Contents

2.1 Introduction

In this chapter, the problem of leader-following consensus for double-integrator MAS with communication constraints is studied. The dynamics of the leader is not necessarily constant and can be controlled with an external input. The leader sends its information to one or more followers while all the followers in the network are required to follow the leader trajectory.

The problem of leader-following consensus has been widely studied in the literature, for example in [123, 159–163] to name a few. However, there are still various issues that need to be addressed. Communication constraints in MAS, raised due to the nature of available sensors and communication equipment, are among those unsolved problems. Contrary to the existing literature, the focus of this chapter is to design a distributed leader-following control algorithm for MAS with the following communicating constraints:

- An agent only transmits its position state to its neighbors (nor its velocity nor its input).
- The agents transmit their data with irregular sampling times.
- The sampling instants for each agent are totally independent which means that no synchronization is required for the transmission.
- The leader can communicate its position to only a small portion of the followers.
- The communication topology among agents is directed.

The objective of this chapter is to design a distributed leader-following control algorithm to deal with all these constraints. For this purpose, an observer-based control scheme is proposed. A local high-gain continuous-discrete time observer is used to reconstruct the position and velocity of an agent and its neighbors in continuous time from discrete position data. Then these estimated states are used to design the control input. Such an approach has been motivated by [156] for a leaderless consensus problem. However, the results of [156] cannot be directly applied for the leader-following case since the inclusion of an active leader influences the stability of the whole system. A Lyapunov-based detailed stability analysis of the proposed algorithm is carried out and the theoretical results are validated through MATLAB simulations. The obtained results are also extended to the case of switching communication topology.

The chapter is organized as follows. Firstly, some preliminaries are given in Section 2.2 which include some important definitions and results. The main problem is formulated in Section 2.3. The observer-based controller design for leader-following consensus for fixed topology is presented in Section 2.4. It also gives the stability analysis of the proposed algorithm along with the numerical results. An extension of the designed algorithm for switching topology is presented in Section 2.5. Finally, Section 2.6 gives a brief conclusion.

2.2 Preliminaries

Before formulating the problem, we first introduce the notations used in this chapter, followed by some important definitions and results.

The set of $n \times n$ real matrices is denoted $\mathbb{R}^{n\times n}$. $I_n \in \mathbb{R}^{n\times n}$ is an n-dimensional identity matrix. For any symmetric matrix A, $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ represent minimum and maximum eigenvalues of A respectively. $\triangleq$ means equal by definition. $\mathcal{D}_i^N$ denotes an $N \times N$ matrix with all entries equal to zero except the i^{th} diagonal entry which is 1. $\|.\|_2$ and $\|.\|_F$ represent the Euclidean and Frobenius norms respectively. If nothing is specific then $\|.\|$ denotes the Euclidean norm. $\text{diag}(b_1, \ldots, b_q)$, with $b_i \in \mathbb{R}^{m\times m}$, $i = 1, \ldots, q$, $q, m \in \mathbb{N}$, is the diagonal by block matrix having $b_1, \ldots, b_q$ on its diagonal. $\mathbf{1}_N \in \mathbb{R}^N$ represents a vector with all entries equal to 1. The Kronecker product of two matrices A and B is denoted as $A \otimes B$.

Definition 13. *[164, Def. 1.2 p. 133] If a non-singular real matrix $W = (w_{ij}) \in \mathbb{Z}^{n\times n}$ with $1 \leq i, j \leq n$ has all positive diagonal elements ($w_{ii} > 0$), non-positive off-diagonal entries ($w_{ij} \leq 0, \forall i \neq j$) and all the eigenvalues have positive real part then such matrix is called M-matrix.*

Lemma 14. *[164, Th. 2.3-H24 p. 134] Let $W \in \mathbb{Z}^{n\times n}$ is an M-matrix then there exists a positive vector $\omega = [\omega_1 \dots \omega_n]^T$ such that $\Omega W + W^T\Omega > 0$ where $\Omega = diag(\omega_1 \dots \omega_n)$.*

Lemma 15. *We have the following results:*

(a) $\|A\|_2 \leq \|A\|_F \leq \sqrt{n}\|A\|_2$ *for all* $A \in \mathbb{R}^{n\times n}$;

(b) $\|x\|_2 \leq \|x\|_1 \leq \sqrt{n}\|x\|_2$ *for all* $x \in \mathbb{R}^n$;

(c) $\sum_{i=1}^n \sqrt{x_i} \leq \sqrt{n}\sqrt{\sum_{i=1}^n x_i}$ *for all* $x_i \in \mathbb{R}^+$ *with* $i = 1, \dots, n$;

(d) $\sum_{i=1}^n (x_i)^2 \leq (\sum_{i=1}^n x_i)^2$ *for all* $x_i \in \mathbb{R}^+$ *with* $i = 1, \dots, n$;

(e) let μ_i, $i = 1, \dots, m$ *and* ν_j, $j = 1, \dots, n$ *be respectively the eigenvalues of* $A \in \mathbb{R}^{m\times m}$ *and* $B \in \mathbb{R}^{n\times n}$, *then the eigenvalues of* $A \otimes B$ *are* $\mu_i\nu_j$ *with* $i = 1, \dots, m$ *and* $j = 1, \dots, n$;

(f) $\|A \otimes B\|_2 = \|A\|_2 \|B\|_2$ *for all* $A \in \mathbb{R}^{n\times n}$ *and* $B \in \mathbb{R}^{m\times m}$;

(g) let $A, B \in \mathbb{R}^{n\times n}$ *be positive definite symmetric matrices, then*

$$\begin{aligned} \lambda_{\max}(A \otimes B) &\leq \lambda_{\max}(A)\lambda_{\max}(B) \\ \lambda_{\min}(A \otimes B) &\geq \lambda_{\min}(A)\lambda_{\min}(B) \end{aligned}$$

(h) for any symmetric definite positive matrix $M \in \mathbb{R}^{n\times n}$ *and* $x, y \in \mathbb{R}^n$, $x^T My \leq \sqrt{x^T Mx}\sqrt{y^T My}$ *(Cauchy-Schwarz inequality);*

(i) for any symmetric matrix $M \in \mathbb{R}^{n\times n}$ *and* $x \in \mathbb{R}^n$, $\lambda_{\min}(M)x^T x \leq x^T Mx \leq \lambda_{\max}(M)x^T x$ *(Rayleigh inequality);*

(j) let $(u^i)_{1\leq i\leq n}$ *a basis of* $\mathbb{R}^n$ *and* $(v^j)_{1\leq j\leq m}$ *a basis of* $\mathbb{R}^m$, *then* $(u^i \otimes v^j)_{1\leq i\leq n, 1\leq j\leq m}$ *is a basis of* $\mathbb{R}^{nm}$;

(k) let $A \in \mathbb{R}^{n\times n}$ *be a symmetric definite positive matrix and* $B \in \mathbb{R}^{\times m}$ *be a symmetric semi-definite matrix, then the following inequality hold*

$$\lambda_{\min}(A)I_n \otimes B \leq A \otimes B \leq \lambda_{\max}(A)I_n \otimes B.$$

Proof. (a) See [165, p.304], section 5.6.23

(b) See [165, p.304], section 5.6.23

(c) Consider $X = \begin{pmatrix} \sqrt{x_1} \\ \vdots \\ \sqrt{x_n} \end{pmatrix}$, then we have

$$\begin{aligned} \|X\|_1 &= \sum_{i=1}^n \sqrt{x_i} \\ \|X\|_2 &= \sqrt{\sum_{i=1}^n (\sqrt{x_i})^2)} = \sqrt{\sum_{i=1}^n x_i} \end{aligned}$$

Applying Lemma 15-b) gives the result.

(d) The proof is straightforward and then not reported here.

(e) See [166, p.27], property *IX*.

(f) Denoting $\rho(A)$ the spectral radius of matrix A, one has

$$\begin{aligned}\|A\otimes B\|_2^2 &= \rho\left((A\otimes B)^T(A\otimes B)\right)\\ &= \rho((A^TA)\otimes(B^TB))\\ &= \rho(A^TA)\rho(B^TB)) \quad \text{(by applying Lemma 15-e)}\\ &= \|A\|_2^2\,\|B\|_2^2\end{aligned}$$

(g) The two inequalities are obtained directly from Lemma 15-e).

(h) See [165, p.15], subsection 0.6.3.

(i) See [165, p.234], Theorem 4.2.2.

(j) From property X in [166, p. 27], one has $det(A\otimes B) = (det(A))^m(det(B))^n$ for any matrices $A\in\mathbb{R}^{n\times n}$ and $B\in\mathbb{R}^{m\times m}$. Then taking $A=(u^1,\dots,u^n)$ and $B=(v^1,\dots,v^m)$, one has $det(A\otimes B) = det([u^1\otimes v^1,\dots,u^1\otimes v^m,u^2\otimes v^1,\dots,u^n\otimes v^m]) = (det(A))^n(det(B))^m\neq 0$ since (u^i) and (v^j) are basis of $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively.

(k) Let x be a non zero vector $\mathbb{R}^{mn}$. Let (u^i) (resp. (v^j)) be a basis of orthogonal eigenvectors of A (resp. B) of $\mathbb{R}^n$ (resp. $\mathbb{R}^m$), that is $Au^i=\mu^iu^i$, with μ^i an eigenvalue of A (resp. $Bv^j=\lambda^jv^j$, with λ^j an eigenvalue of B). Since, according to point *h)*, $(u^i\otimes v^j)_{1\leq i,1\leq j\leq m}$ is a basis of $\mathbb{R}^{mn}$, there exist reals α_{ij} such that

$$x=\sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}u^i\otimes v^j.$$

One has

$$\begin{aligned}x^T(A\otimes B)x &= \sum_{i_1,i_2=1}^n\sum_{j_1,j_2=1}^m\alpha_{i_1j_1}\alpha_{i_2j_2}(u^{i_1}\otimes v^{j_1})^T(A\otimes B)(u^{i_2}\otimes v^{j_2})\\ &= \sum_{i_1,i_2=1}^n\sum_{j_1,j_2=1}^m\alpha_{i_1j_1}\alpha_{i_2j_2}((u^{i_1})^TAu^{i_2})\otimes((v^{j_1})^TBu^{j_2})\\ &= \sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}^2((u^i)^TAu^i)\otimes((v^j)^TBu^j) \text{ since } (u^i) \text{ and } (v^j) \text{ are orthogonal basis,}\\ &= \sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}^2\mu^i\lambda^j((u^i)^Tu^i)\otimes((v^j)^Tu^j)\end{aligned}$$

Then, since B is semidefinite positive symmetric, one has $\lambda^j\geq 0$, $j=1,\dots,m$. Furthermore $((u^i)^Tu^i)\otimes((v^j)^Tv^j) = ((u^i)^Tu^i)((v^j)^Tv^j)\geq 0$, then

$$\begin{aligned}\sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}^2\mu^i\lambda^j((u^i)^Tu^i)\otimes((v^j)^Tv^j) &\leq \max_i\{\mu^i\}\sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}^2\lambda^j((u^i)^Tu^i)\otimes((v^j)^Tv^j)\\ &= \max_i\{\mu^i\}\sum_{i=1}^n\sum_{j=1}^m\alpha_{ij}^2((u^i)^Tu^i)\otimes((v^j)^TBv^j)\\ &= \max_i\{\mu^i\}x^T(I_n\otimes B)x\end{aligned}$$

The same can be done for the other inequality.

□

Theorem 24.

Lemma 16. *Let $v_1(t)$ and $v_2(t)$ be real valued functions verifying*

$$\frac{d}{dt}\left(v_1^2(t)+v_2^2(t)\right) \leq -av_1^2(t) - bv_2^2(t) + c\int_{t-\delta}^{t} v_2^2(s)ds + k, \tag{2.1}$$

for all $t \geq 0$, where $a, b, c, \delta > 0$ and $k \geq 0$. There exists $\varrho > 0$, independent of a, b, c, k, and $\bar{\alpha} \geq 0$ such that if

$$\delta < \varrho \min\left(\frac{b}{c}, \frac{1}{\sigma}\right)$$

then $v_1(t)$ and $v_2(t)$ verify the following inequality

$$v_1^2(t) + v_2^2(t) \leq \bar{\alpha}e^{-\sigma t} + \frac{k}{\sigma}, \quad \forall t \geq 0 \tag{2.2}$$

where σ is given by

$$\sigma = \frac{1}{2}\min(a, b) \tag{2.3}$$

Proof. Let $v = \min\left(\frac{a}{\sqrt{2}}, \frac{b}{\sqrt{2}}\right)$, $\xi = 2\frac{c\delta}{b}$ and $\kappa = 1 - \xi$. Since $\delta \in \left(0, \varrho\min\left(\frac{b}{c}, \frac{1}{\sigma}\right)\right)$, one has

$$0 < 2\frac{c\delta}{b} < 2\frac{c}{b}\varrho\min\left(\frac{b}{c}, \frac{1}{\sigma}\right) \leq 2\varrho \quad \Rightarrow \quad 1 - 2\varrho < \underbrace{1-\xi}_{=\kappa} < 1 \tag{2.4}$$

$$0 < v\kappa\delta < v\delta < v\varrho\min\left(\frac{b}{c}, \frac{1}{\sigma}\right) \leq \sqrt{2}\varrho \tag{2.5}$$

Then $\varrho > 0$ can be chosen, independently of a, b, c, k such that for all $\delta \in \left(0, \varrho\min\left(\frac{b}{c}, \frac{1}{\sigma}\right)\right)$

$$\kappa \in \left(\frac{1}{\sqrt{2}}, 1\right) \tag{2.6}$$

$$e^{v\kappa\delta} \leq 1 + 2v\kappa\delta \tag{2.7}$$

Consider the following candidate Lyapunov function

$$W(v_t) = v_1^2(t) + v_2^2(t) + c\int_0^{\delta}\int_{t-s}^{t} e^{v\kappa(\mu-t+s)}v_2^2(\mu)d\mu ds \tag{2.8}$$

where $v_t(s) = [v_1(t+s), v_2(t+s)]^T$, $s \in [-\delta, 0]$. One has

$$\dot{W}(v_t) = \frac{d}{dt}\left(v_1^2(t)+v_2^2(t)\right) + c\int_0^{\delta}\frac{d}{dt}\int_{t-s}^{t} e^{v\kappa(\mu-t+s)}v_2^2(\mu)d\mu ds.$$

Applying Leibniz integration leads to

$$\dot{W}(v_t) = \frac{d}{dt}\left(v_1^2(t)+v_2^2(t)\right) - v\kappa c\int_0^{\delta}\int_{t-s}^{t} e^{v\kappa(\mu-t+s)}v_2^2(\mu)d\mu ds + c\int_0^{\delta} e^{v\kappa s}v_2^2(t) - v_2^2(t-s)ds$$

By using (2.1), one has

$$\begin{aligned}\dot{W}(v_t) &\leq -av_1^2(t) - bv_2^2(t) + c\int_{t-\delta}^{t} v_2^2(s)ds + k - v\kappa c\int_0^{\delta}\int_{t-s}^{t} e^{v\kappa(\mu-t+s)}v_2^2(\mu)d\mu ds \\ &\quad + c\int_0^{\delta} e^{v\kappa s}v_2^2(t)ds - c\int_0^{\delta} v_2^2(t-s)ds \\ &\leq -av_1^2(t) - bv_2^2(t) + k + c\left(\frac{e^{v\kappa\delta}-1}{v\kappa}\right)v_2^2(t) - v\kappa\left(W(v_t) - v_1^2(t) - v_2^2(t)\right)\end{aligned}$$

Since $\frac{e^{v\kappa\delta}-1}{v\kappa} \leq 2\delta$ and given the definition of v, the following inequalities are achieved

$$\begin{aligned}
\dot{W}(v_t) + v\kappa W(v_t) &\leq (-a + v\kappa)\, v_1^2(t) + (-b + 2c\delta + v\kappa) v_2^2(t) + k \\
\dot{W}(v_t) + v\kappa W(v_t) &\leq -a\left(1 - \frac{1}{\sqrt{2}}\right) v_1^2(t) - b\left(1 - (1-\kappa) - \frac{\kappa}{\sqrt{2}}\right) v_2^2(t) + k \\
\dot{W}(v_t) + v\kappa W(v_t) &\leq -a\left(\frac{\sqrt{2}-1}{\sqrt{2}}\right) v_1^2(t) - b\kappa\left(\frac{\sqrt{2}-1}{\sqrt{2}}\right) v_2^2(t) + k \\
\dot{W}(v_t) &\leq -v\kappa W(v_t) + k \\
\dot{W}(v_t) &\leq -\sigma W(v_t) + k
\end{aligned}$$

In order to get an over-valuation of W, one uses the comparison Lemma 2.5 p85 [167]. The solution of the following ODE

$$\dot{z}(t) = -\sigma z(t) + k$$

is given by

$$z(t) = \left(z(0) - \frac{k}{\sigma}\right) e^{-\sigma t} + \frac{k}{\sigma}$$

Taking $\bar{\alpha} = (W(v_0) - \frac{k}{\sigma})$ ends the proof. □

2.3 Problem formulation

Consider a group of N followers labeled from 1 to N and one leader labeled 0. The followers have the following second-order dynamics:

$$\begin{cases} \dot{r}_i(t) = v_i(t), \quad i = 1, \ldots, N \\ \dot{v}_i(t) = u_i(t) \end{cases} \tag{2.9}$$

where $r_i \in \mathbb{R}^m$ and $v_i \in \mathbb{R}^m$ represent the position and the velocity of i-th agent respectively while $u_i \in \mathbb{R}^m$ is its control input with $m \in \mathbb{N}$. The dynamics of the leader is as followers and is given by:

$$\begin{cases} \dot{r}_0(t) = v_0(t) \\ \dot{v}_0(t) = u_0(t) \end{cases} \tag{2.10}$$

where $p_0, v_0, u_0 \in \mathbb{R}^m$ represent respectively the position, velocity and control input of the leader. The input of the leader u_0 is independent and is not affected by the followers. It could be designed to achieve any desired reference trajectory for the followers. Only the following assumption is considered.

Assumption 17. *It is assumed that the control input u_0 is bounded, that is, there exists $\delta_0 \geq 0$ such that*

$$\|u_0(t)\| \leq \delta_0, \qquad \forall t \geq 0$$

2.3.1 Communication constraints

The communication connection between N followers is described by a graph $\mathcal{G}$ and the corresponding adjacency and Laplacian matrices are $\mathcal{A}$ and $\mathcal{L}$, respectively. It is considered that each agent transmits only its position r_j to its neighbor i at times $t_k^{i,j}$ with $k \in \mathbb{N}$, but neither its velocity v_j nor its input u_j. The sampling instants $t_k^{i,j}$ are supposed to verify

$$0 = t_0^{i,j} < t_1^{i,j} < \cdots < t_k^{i,j} < \ldots \tag{2.11}$$

Furthermore, one assumes that there exist constants $\tau_m, \tau_M > 0$, called respectively the minimum sampling period and the maximum sampling period, such that

$$\tau_m < t_{k+1}^{i,j} - t_k^{i,j} < \tau_M \tag{2.12}$$

for all $k \in \mathbb{N}$ and $i = 1, \ldots, N$, $j = 0, \ldots, N$.

Example. *Consider a MAS with communication topology as shown in Figure 2.1. Agent 2 receives position data of agent 1 at time instants* $t_k^{2,1}$ *for* $k \in \mathbb{N}$*. These samples are used by agent 2 to estimate the position and velocity of agent 1 in continuous time. Agent 2 also measures its own position at time instants* $t_k^{2,2}$ *to reconstruct it own states in continuous time. Similarly, agent 3 receives position information of agent 1 and agent 2 at time instants* $t_k^{3,1}$ *and* $t_k^{3,2}$ *respectively and its own position at* $t_k^{3,3}$*. Agent 1 does not have any neighbor but it uses its own position information at time instants* $t_k^{1,1}$*. It can be noted that the time instants* $t_k^{i,j}$ *for* $i, j = 1, 2, 3$ *are independent and can be chosen freely as far as they verify* (2.11) *and* (2.12)*.*

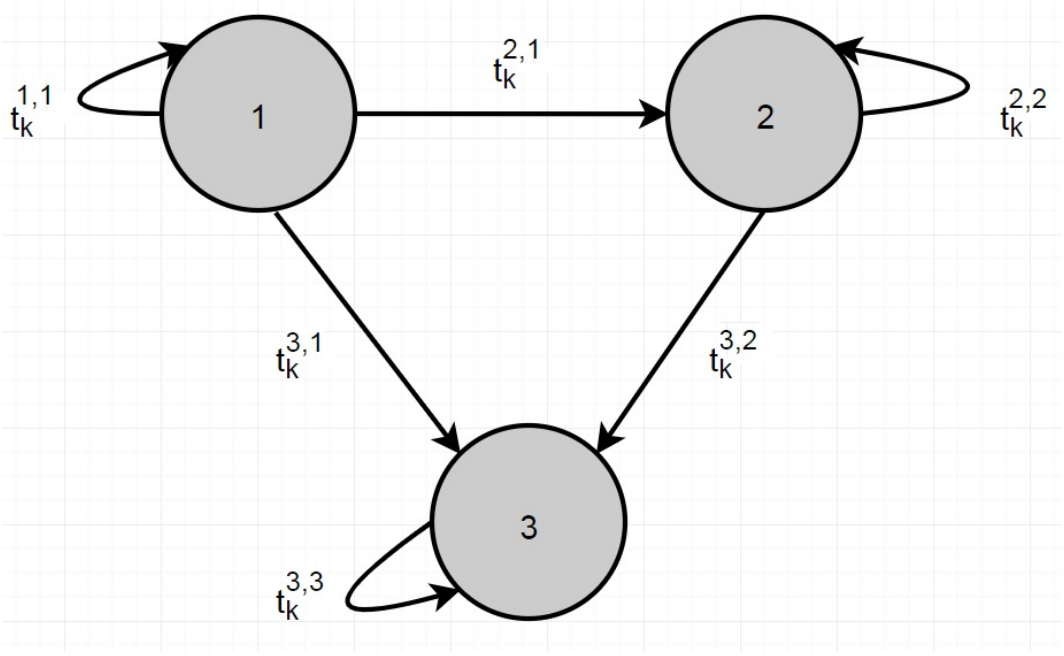


Figure 2.1: Example of sampling instants for data transmission under a directed graph.

The position of the leader is considered to be transmitted to a small portion of the followers. Let the diagonal matrix $\mathcal{B} = diag(b_1, b_2, \ldots, b_N)$ be the interconnection relationship between the leader and followers, where $b_i = 1$ if the information of the leader is accessible by the i^{th} follower, otherwise $b_i = 0$. The communication graph including the followers and the leader is denoted $\tilde{\mathcal{G}}$. The corresponding Laplacian matrix $\tilde{\mathcal{L}}$ is given by:

$$\tilde{\mathcal{L}} = \begin{pmatrix} 0 & 0_{1\times N} \\ -\tilde{b} & \mathcal{H} \end{pmatrix}$$

where

$$\tilde{b} = (b_1, \ldots, b_N)^T$$

and

$$\mathcal{H} = \mathcal{L} + \mathcal{B}$$

Lemma 18. *[168] Matrix* $\mathcal{H}$ *is a nonsingular M-matrix if and only if the pinning joint communication topology* $\tilde{\mathcal{G}}$ *has a directed spanning tree.*

Assumption 19. *The communication topology* $\tilde{\mathcal{G}}$ *between the leaders and the followers has a directed spanning tree with the leader as a root.*

Note that if $\tilde{\mathcal{G}}$ has a directed spanning tree, then according to Lemma 14, there exists a diagonal matrix $\Omega = \text{diag}(\omega_1, \ldots, \omega_N)$ such that $\mathcal{H}^T\Omega + \Omega\mathcal{H} > 0$. Furthermore, the following notations will be used hereafter.

$$\omega_{\max} = \max\{\omega_1, \ldots, \omega_N\}, \tag{2.13}$$
$$\omega_{\min} = \min\{\omega_1, \ldots, \omega_N\}, \tag{2.14}$$
$$\rho = \lambda_{\min}(\mathcal{H}^T\Omega + \Omega\mathcal{H}) \tag{2.15}$$
$$h_{\max} = \max_{i,j} |\mathcal{H}_{ij}| \tag{2.16}$$

The objective here is to achieve the leader-following consensus in a MAS subject to the previously mentioned communication constraints and with a dynamic leader according to the following definition.

Definition 20. *The leader-following practical exponential consensus is achieved if*

$$\sum_{i=1}^{N} \|e_i(t)\| \leq \alpha e^{-\beta t} + \gamma, \quad \forall t \geq 0$$

where $e_i = x_i - x_0$ *with* $x_i = \left[r_i^T, v_i^T\right]^T$, $x_0 = \left[r_0^T, v_0^T\right]^T$, $\alpha, \beta > 0$ *and* γ *is a positive constant.*

2.4 Observer based leader-following consensus

In this work, the basic idea is to use the classical continuous linear consensus controller for double integrator MAS [28,29,43] with discrete position measurements only. In a classical consensus protocol, it is assumed that all states of the neighbors are available in continuous time (please see Section 1.4 of Chapter 1 for details). However, in our case, only discrete position data is available to the neighbors at irregular and asynchronous time instants. Therefore, first, it is needed to reconstruct the position and velocity of the agents in continuous time from the available discrete position data. We use a continuous-discrete time high gain observer [139] to estimate the states in continuous time from the received discrete data. As mentioned in Chapter 1, continuous-discrete time observers have been widely studied specially for single agent system [139, 169, 170] and have proven their effectiveness for the estimation of both available and unavailable states in continuous time for a system with discrete output. We use the same idea to reconstruct the position and velocity of an agent and its neighbors and then the leader-following controller is designed using these estimated states.

The proposed observer-based consensus protocol is given for $t \geq 0$ by

$$\begin{aligned} u_i(t) = & -\bar{c}\lambda^2 \sum_{j=1}^{N} a_{ij} \left[\hat{r}_{i,i}(t) - \hat{r}_{i,j}(t)\right] - \bar{c}2\lambda \sum_{j=1}^{N} a_{ij} \left[\hat{v}_{i,i}(t) - \hat{v}_{i,j}(t)\right] \\ & -\bar{c}\lambda^2 b_i \left[\hat{r}_{i,i}(t) - \hat{r}_{i,0}(t)\right] - \bar{c}2\lambda b_i \left[\hat{v}_{i,i}(t) - \hat{v}_{i,0}(t)\right], \quad i = 1, \ldots, N \end{aligned} \tag{2.17}$$

where $\hat{r}_{i,j}$ and $\hat{v}_{i,j}$ are the estimated position and velocity of agent j by agent i. Their dynamics are given by

$$\dot{\hat{r}}_{i,j}(t) = \hat{v}_{i,j}(t) - 2\theta e^{-2\theta(t-t_k^{i,j})} \left(\hat{r}_{i,j}(t_k^{i,j}) - r_j(t_k^{i,j})\right) \tag{2.18}$$
$$\dot{\hat{v}}_{i,j}(t) = -\theta^2 e^{-2\theta(t-t_k^{i,j})} \left(\hat{r}_{i,j}(t_k^{i,j}) - r_j(t_k^{i,j})\right) \tag{2.19}$$

for $i = 1, \ldots, N$, $j = 0, 1, \ldots, N$ and $t \in \left[t_k^{i,j}, t_{k+1}^{i,j}\right)$, $k \in \mathbb{N}$, where $\bar{c} > 0$ is the coupling strength, a_{ij} is the (i,j)-th entry of the adjacency matrix $\mathcal{A}$ of the directed graph $\mathcal{G}$ while θ and $\lambda > 0$ are the observer and controller tuning parameter respectively. The initial conditions $\hat{r}_{i,j}(0), \hat{v}_{i,j}(0) \in \mathbb{R}^m$ of the observers can be chosen arbitrarily. Figure 2.2 shows the block diagram of the proposed observer based leader-following control algorithm for agent i.

Remark 21. *The term* $e^{-2\theta(t-t_k^{i,j})} \left(\hat{r}_{i,j}(t_k^{i,j}) - r_j(t_k^{i,j})\right)$, *in the observer dynamics* (2.18), (2.19), *acts as output error* $(r_{i,j}(t) - r_j(t))$ *predictor between two samples, i.e. over an interval* $t \in \left[t_k^{i,j}, t_{k+1}^{i,j}\right)$. *More details on such output predictor can be found in [169] Section III.B.*

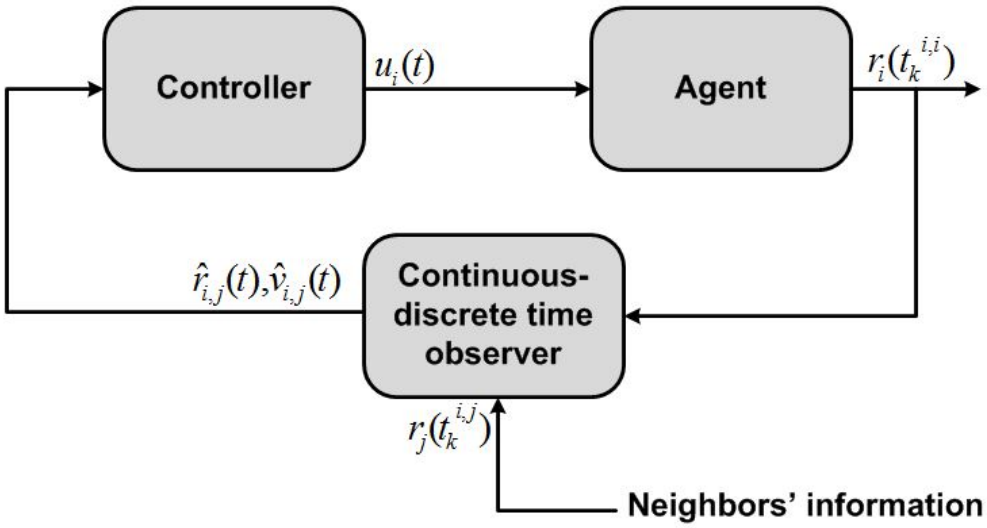


Figure 2.2: Block diagram of proposed observer based leader-following controller for agent i

Remark 22. *Each follower has a local observer which not only estimates the agent's own states but also the states of its neighbors.*

Remark 23. *The structure of the proposed observer-based leader-following algorithm also has some advantages when it comes to communication delays and data packet dropouts. As each controller is using the estimated states provided by the local observer, by time stamping the measured position data, the estimation can be provided as soon as the data is received, even with a delay, by compensating it through increasing the computation speed of the observer. Moreover, if the information packet is lost during communication, the observer could still provide the estimation if the next data is available within τ_M duration with respect to the last available data.*

Theorem 24. *Consider the MAS* (2.9)-(2.10) *with the consensus protocol* (2.17)-(2.19) *and assume that the communication topology $\tilde{\mathcal{G}}$ contains a directed spanning tree. If the control parameters $\theta, \lambda, \bar{c} > 0$ satisfy the following*

$$\theta < \frac{\bar{\varrho}}{\tau_M} \tag{2.20}$$

$$\lambda < \varepsilon^* \theta \tag{2.21}$$

$$\bar{c} \geq \frac{\omega_{\max}}{\rho} \tag{2.22}$$

where $\bar{\varrho}$ is a positive constant, $\varepsilon^* \in (0,1)$, ω_{max} and ρ are given by (2.13) *and* (2.15)*, respectively, then the leader-following practical consensus problem is solved in the sense of Definition 20. i.e*

$$\sum_{i=1}^{N} \|e_i\| \leq \alpha e^{-\frac{\lambda}{8}t} + \frac{\beta\delta_0}{\lambda} \tag{2.23}$$

with α and $\beta > 0$.

The proof of Theorem 24 is provided in Appendix A.

2.4.1 Discussion on Theorem 24

The conditions provided in Theorem 24 to achieve the leader-following consensus are sufficient and obtained through a Lyapunov-based stability analysis. They provide some useful information about the choice of the gains. For instance, it is clear from inequality (2.20) that the maximum sampling period directly influences the choice of θ. If the maximum sampling time is high, then θ must be chosen small

enough and vice versa. It should be noted that θ and λ dictate the convergence speed of the observer and controller, respectively. In order to guarantee closed-loop stability, the controller dynamics must be slower than the observer dynamics which is represented by the fact $\varepsilon^* < 1$. Therefore, for a large maximum sampling period, the system convergence rate will be slow.

It is clear from inequality (2.23) that the error will gradually converge and will enter in a ball centered at the origin. This verifies that practical consensus is achieved. The radius of the convergence ball of the tracking error is directly proportional to the bound of the leader input/acceleration δ_0. The size of this ball can also be reduced by increasing the controller gain λ (while keeping in mind that the controller dynamics must remain slower than the observer dynamics in order to guarantee stability of the closed-loop system). Furthermore, if the leader is moving with constant velocity i.e. $u_0 = 0$, then the MAS achieves exponential consensus.

One can remark that the conditions on the control parameters given in Theorem 24 require some global information like $\mathcal{H}$ and N. Hence, each agent must have some global knowledge about the communication topology similarly to many existing works on consensus. Nevertheless, the tuning parameters θ, λ and $\bar{c}$ are chosen beforehand and then they remain constant for all $t \geq 0$. Once the gains are set, only local information is needed to achieve leader-following consensus.

2.4.2 Simulation results

Consider a MAS with a leader denoted as 0 and 10 followers labeled from 1 to 10. The communication topology among the agents is shown in Figure 2.3. It can be seen that only follower 2 and follower 5 have access to the leader. Moreover, Follower 5, 6, and 9 have multiple neighbors. It is also to note that follower 8 can receive information from follower 7 and 9 but it is not transmitting its own data to any other agent in the network.

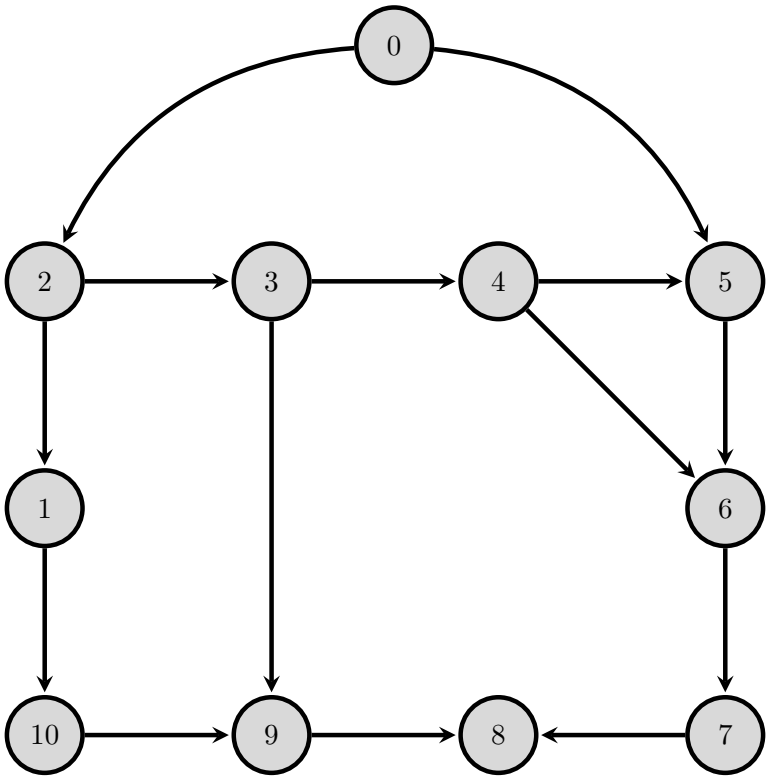


Figure 2.3: Communication topology for collision-free formation tracking.

The adjacency and pinning matrices, corresponding to the communication, topology are

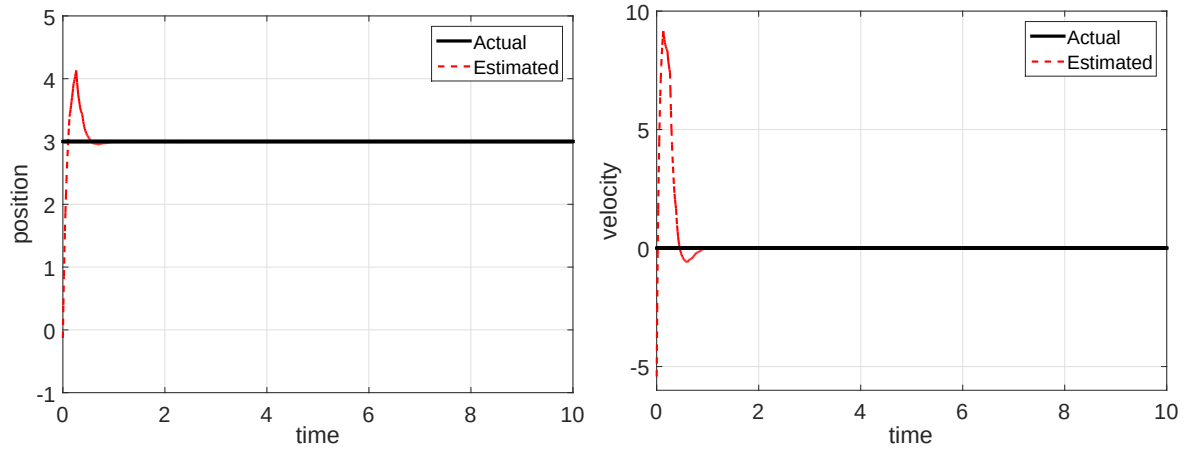
$$\mathcal{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \mathcal{B} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

while the Laplacian matrix can be written as

$$\mathcal{L} = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 2 & -1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

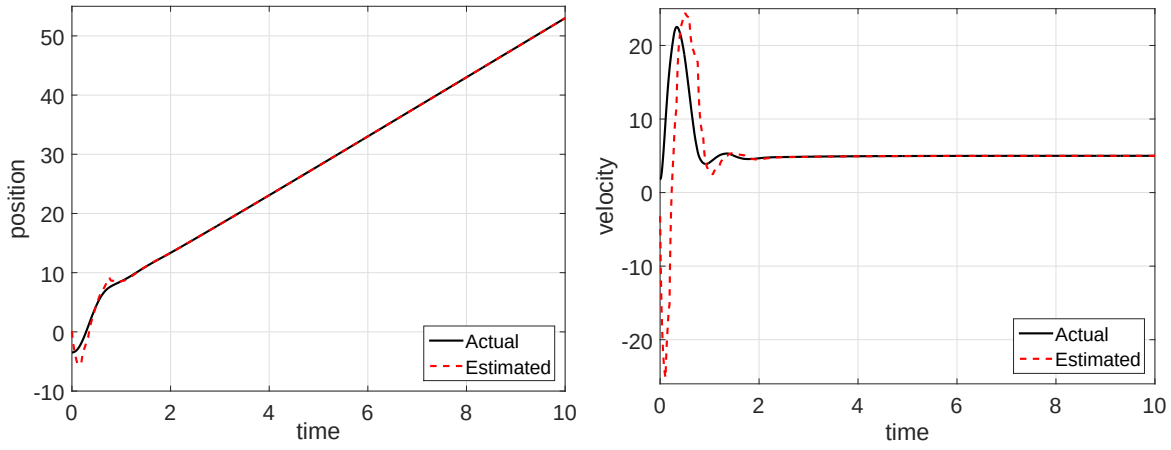
The simulation of the leader-following consensus protocol (2.17)-(2.19) are carried out in MATLAB. The minimum and the maximum sampling period bound is chosen as $\tau_m = 10ms$ and $\tau_M = 100ms$. Using the insight achieved by Theorem 1 and Remark 1, the gains for simulation purpose are chosen through trial and error method as $\bar{c} = 1$, $\lambda = 1.2$ and $\theta = 13$. The initial conditions of the agents and their local observers are chosen randomly.

Figures 2.4 to 2.6 show state estimation by different agents for various trajectories. Figure 2.4 depicts the position and velocity estimation of the leader by follower 2 in the network. Similarly, Figure 2.5 shows the state estimation of follower 2 by follower 1 while Figure 2.6 illustrates the estimation of follower 3's states by follower 9. An example of the sampling time is depicted in Figure 2.7 where it can be seen that each agent has an irregular sampling rate and also these sampling instants are independent of the other agents in the network. It is clear from these results that the designed observer effectively estimates the position and the velocity states in continuous time.



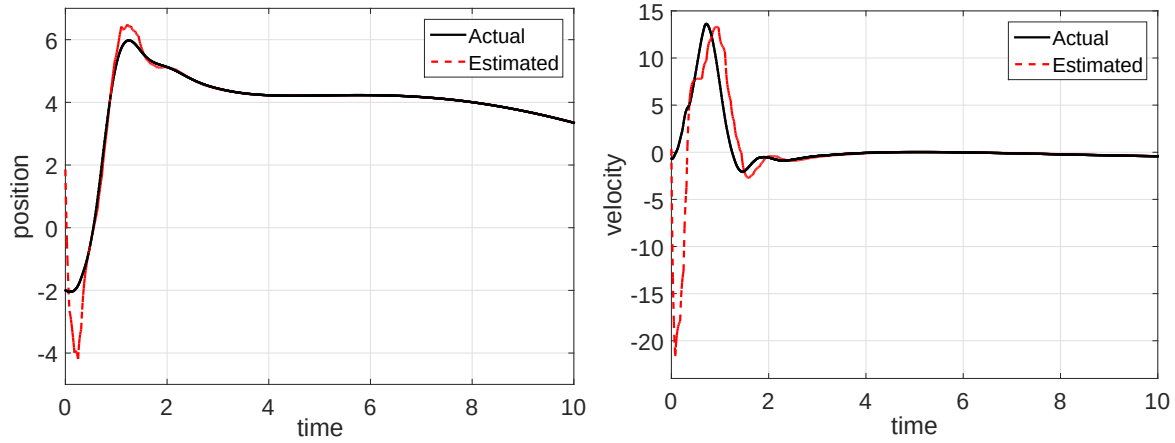
(a) (b)

Figure 2.4: Estimation of leader's states by follower 2 (a) position $\hat{r}_{1,0}$ (b) velocity $\hat{v}_{1,0}$.



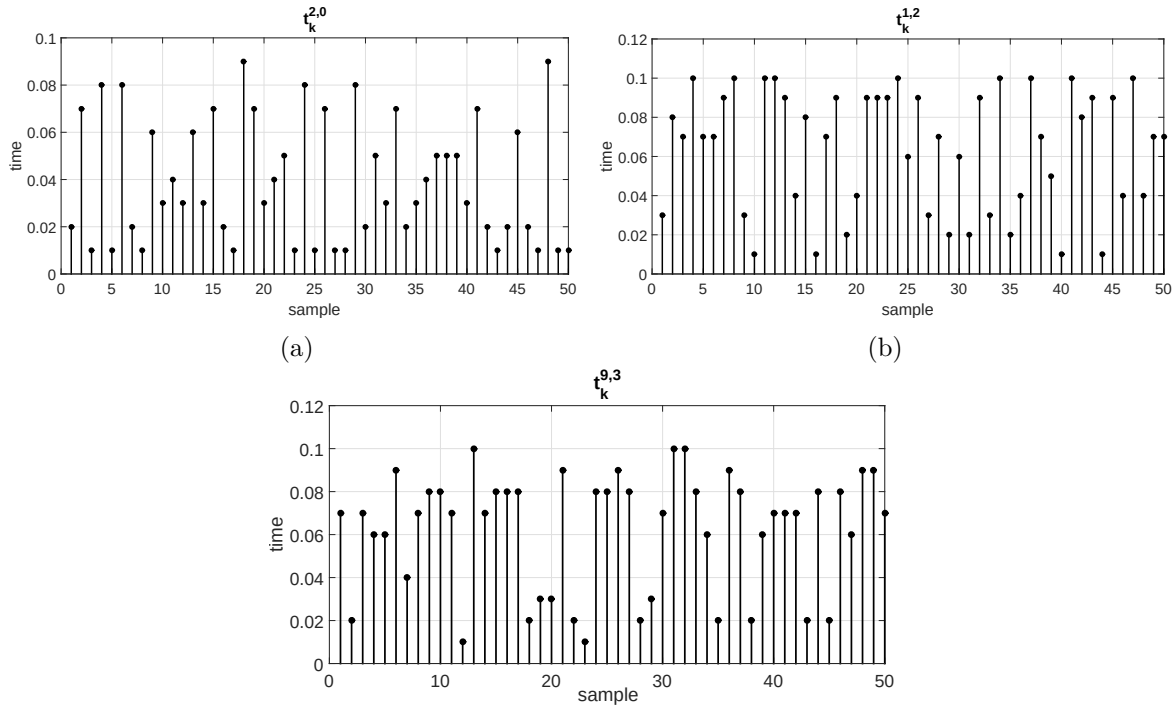
(a) (b)

Figure 2.5: Estimation of follower 2 states by follower 1 (a) position $\hat{r}_{2,1}$ (b) velocity $\hat{v}_{2,1}$.



(a) (b)

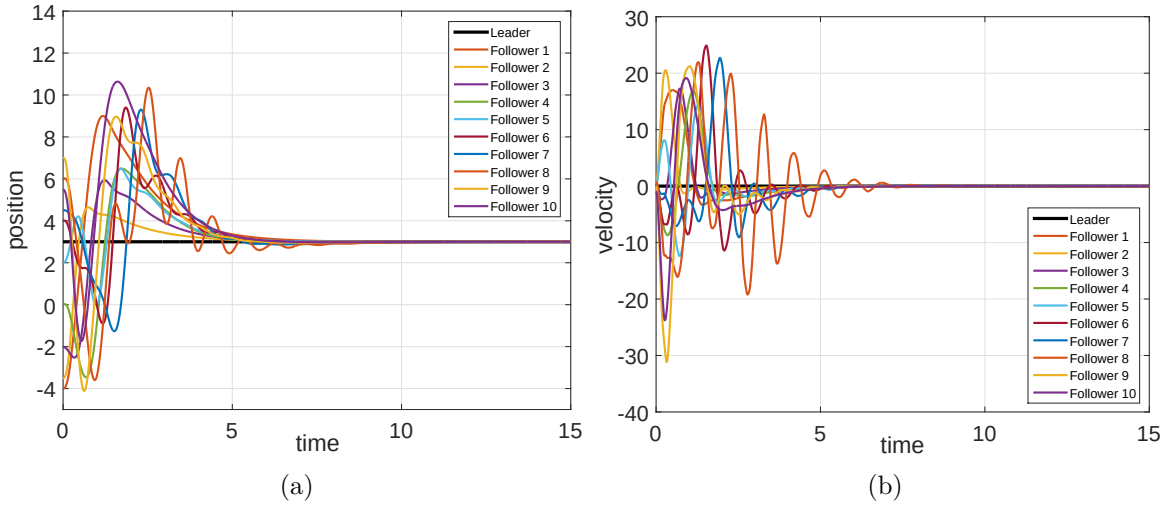
Figure 2.6: Estimation of follower 3 states by follower 9 (a) position $\hat{r}_{3,9}$ (b) velocity $\hat{v}_{3,9}$.



(a) (b)

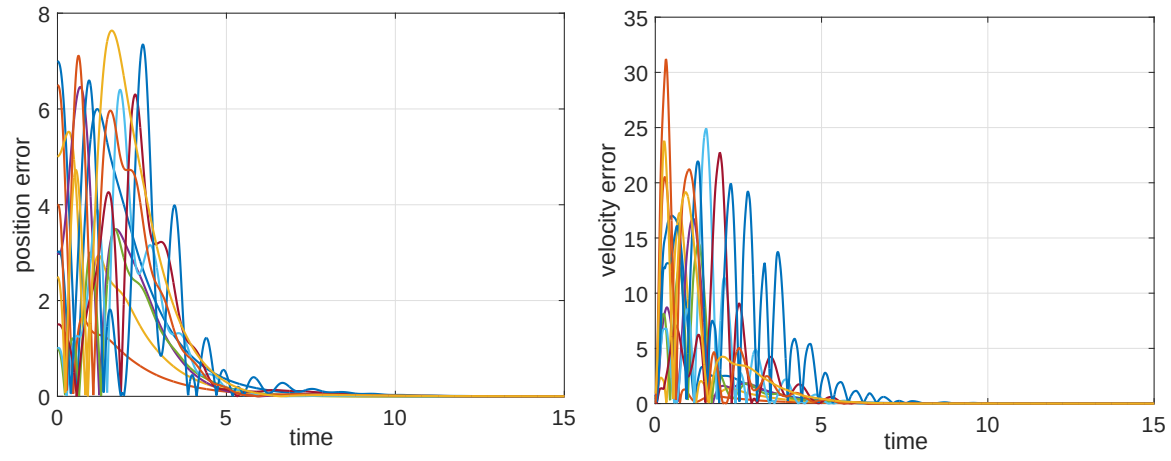
(c)

The proposed distributed leader-following algorithm is validated for various cases of position trajectories of the leader. The first case is considered for step leader trajectory. Figure 2.8 shows the position and the velocity consensus results while Figure 2.9 shows the corresponding tracking error. In the second case, ramp position trajectory of the leader is considered which means that the leader moves with a constant velocity. We set this constant velocity at 5m/s. The leader-following consensus results for this scenario are illustrated in Figure 2.10 for both position and velocity states. The related tracking errors are shown in Figure 2.11.



(a) (b)

Figure 2.8: Consensus tracking with a stationary leader (a) position (b) velocity.



(a) (b)

Figure 2.9: Tracking error with a stationary leader (a) position error $\|r_i - r_0\|$ (b) velocity error $\|v_i - v_0\|$.

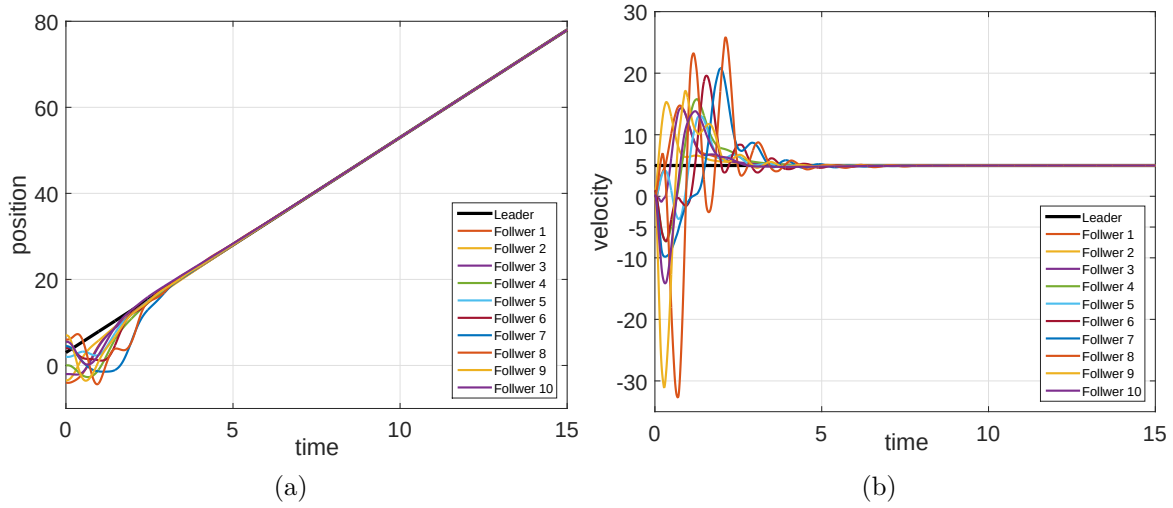


Figure 2.10: Consensus tracking with constant leader velocity (a) position (b) velocity.

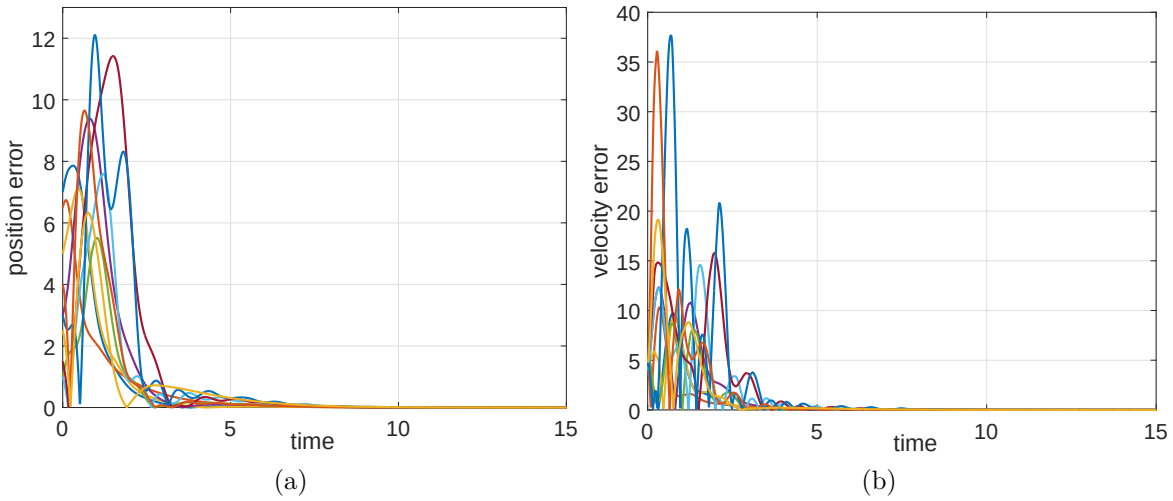
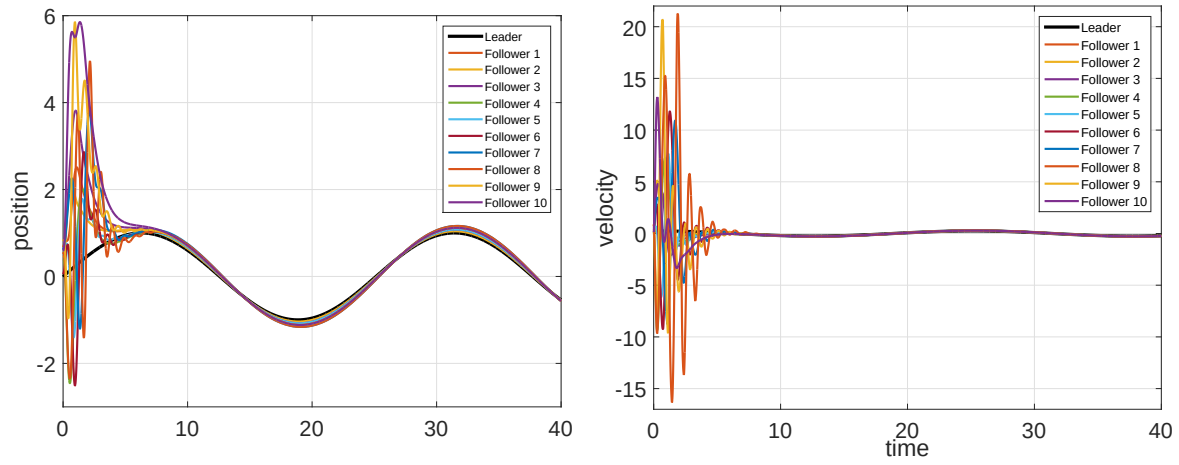


Figure 2.11: Tracking error with constant leader velocity (a) position error $\|r_i - r_0\|$ (b) velocity error $\|v_i - v_0\|$.

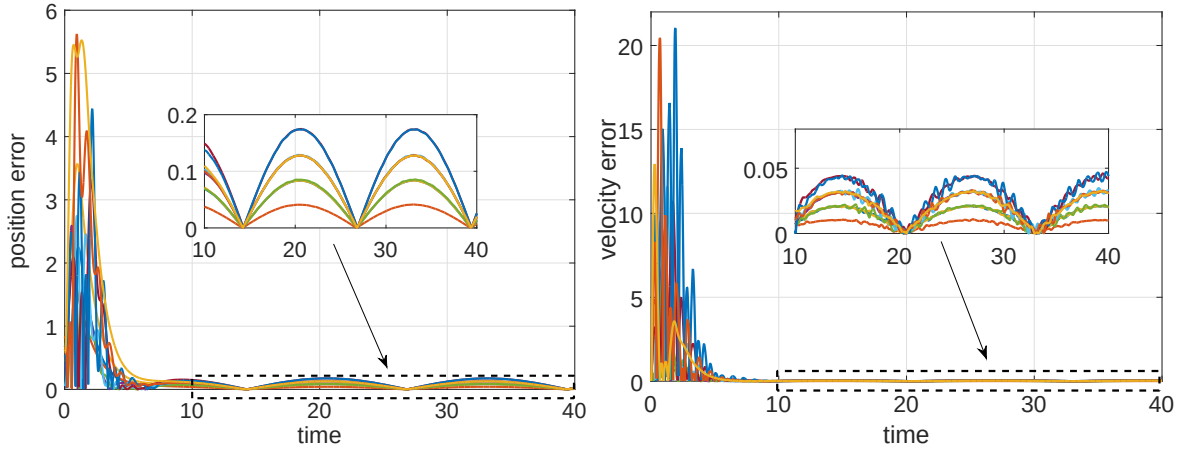
In the final scenario, the leader has a sinusoidal trajectory. To achieve this reference trajectory, the leader's input is selected as $u_0(t) = 0.0625sin(0.25t)\text{m/s}^2$. The consensus tracking results are shown in Figure 2.12 for both position and velocity. Figure 2.13 shows the results of corresponding tracking errors. It can be seen that only practical stability is achieved in the case of non-zero leader's input as expected and discussed in Section 2.4.1. Figures 2.14 and 2.15show the consensus results for the sinusoidal leader input with tuning gains changed to $\theta = 8$ and $\lambda = 0.8$. As mentioned earlier that these gains not only dictate the convergence speed of the system but also, the final error depends on λ (inequality 2.23). Therefore, one can note that the final error increases as λ decreases.

It is worth noting that in all the cases, only position information is transmitted through the communication network, given in Figure 2.3, at nonuniform and asynchronous sampling instants. Velocities and inputs are completely unknown to the neighbors. Despite these constraints, the system achieved leader-following consensus.



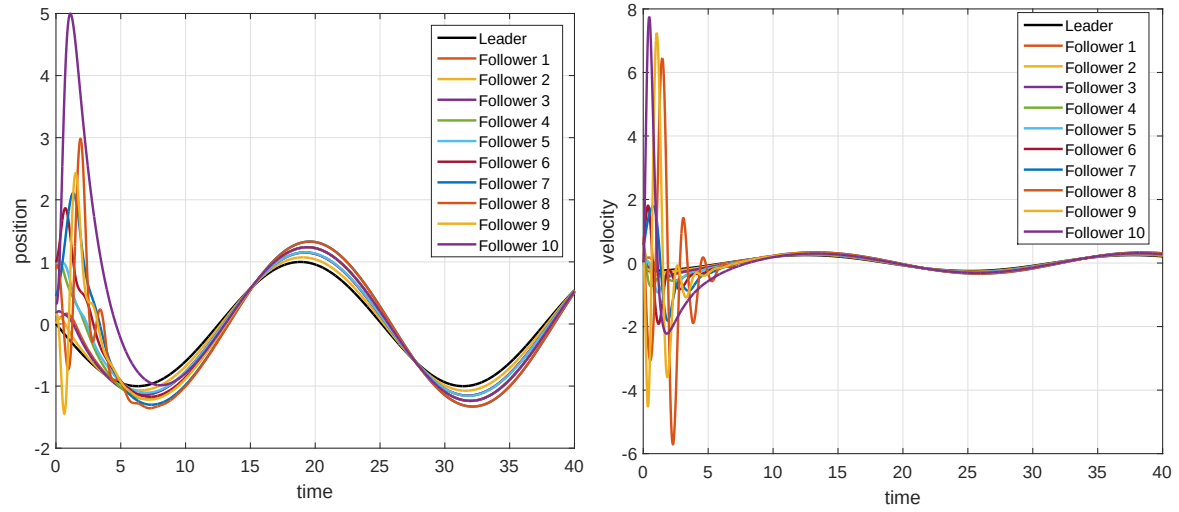
(a) (b)

Figure 2.12: Consensus tracking with sinusoidal leader velocity (a) position (b) velocity.



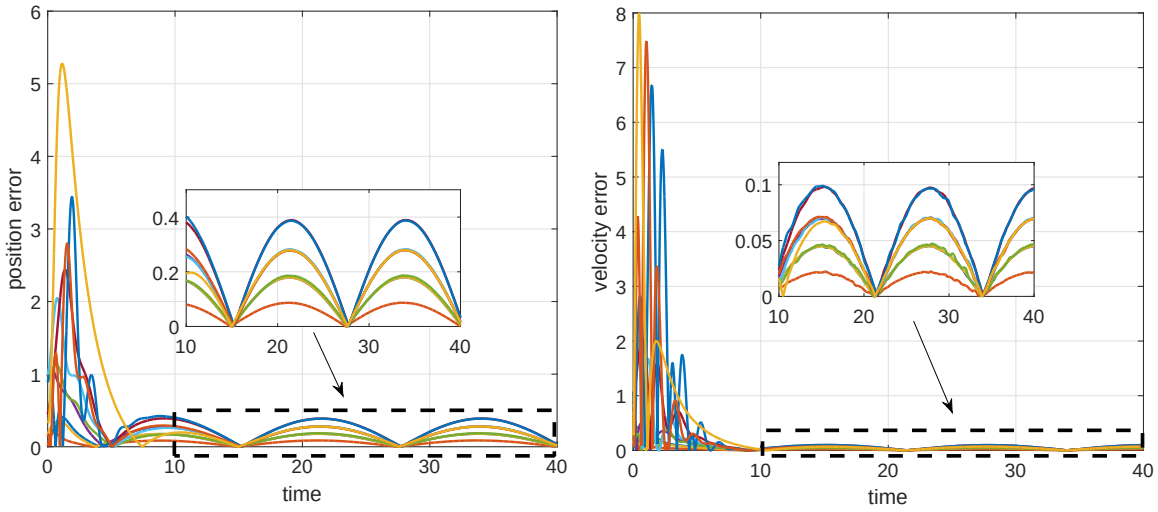
(a) (b)

Figure 2.13: Tracking error with sinusoidal leader input (a) position error $\|r_i - r_0\|$ (b) velocity error $\|v_i - v_0\|$.



(a) (b)

Figure 2.14: Consensus tracking with sinusoidal leader velocity with $\theta = 8.0$ and $\lambda = 0.8$ (a) position (b) velocity.



(a) (b)

Figure 2.15: Tracking error with sinusoidal leader input with $\theta = 8.0$ and $\lambda = 0.8$ (a) position error $\|r_i - r_0\|$ (b) velocity error $\|v_i - v_0\|$.

2.5 Leader-following consensus with switching topology

In this section, the obtained results of leader-following consensus with fixed topology are extended to the case where the communication topology among the agents changes over time. In many practical applications, it is sometime not feasible for the agents to maintain a fixed communication topology due to various reasons like collision avoidance, communication link failure or communication range limitations etc. Therefore, switching topology is a more suitable option in these scenarios.

Let us consider a MAS with N followers. Denote $\underline{\mathcal{G}} = \{\mathcal{G}^1, \mathcal{G}^2, \ldots, \mathcal{G}^M\}$ the finite set of all possible topology graphs and $\mathcal{M} = \{1, 2, \ldots, M\}$ represents the set of indices. Each graph in $\underline{\mathcal{G}}$ has same nodes (agents) but can have different edges. The switching between the graphs is time dependant and is modelled by a switching function $\sigma(t) : (0, \infty] \to \mathcal{M}$ which is a piece-wise constant function. Let $0 = t_0 < t_1 < t_2 \ldots$ be the switching instants of $\sigma(t)$. Furthermore, the intervals $(t_l, t_{l+1}]$, $l = 0, 1, \ldots$ are bounded and contiguous. Denote the directed switching graph as $\mathcal{G}^{\sigma(t)} \in \underline{\mathcal{G}}$ with $A^{\sigma(t)}$ and $L^{\sigma(t)}$ as corresponding adjacency and Laplacian matrices respectively. Let the diagonal matrix $\mathcal{B}^{\sigma(t)} = diag(b_1^\sigma(t), \ldots, b_N^\sigma(t))$ which represents the switching interconnection between the leader and the followers. $b_i^\sigma(t)$, for $i = 1, \ldots, N$ is equal to 1 if agent i can receive information from the leader and zero otherwise. It is to note that the leader does not change, however, its connection with the followers can change. The switching communication graph including the followers and the leader is denoted $\bar{\mathcal{G}}^{\sigma(t)}$.

Assumption 25. *Each switching graph $\bar{\mathcal{G}}^{\sigma(t)}$ has a directed spanning tree with the leader as a root.*

Let us define another matrix

$$\mathcal{H}^{\sigma(t)} = \mathcal{L}^{\sigma(t)} + \mathcal{B}^{\sigma(t)} \tag{2.24}$$

If graph $\bar{\mathcal{G}}^{\sigma(t)}$ has directed spanning tree, then according to Lemma 14 there exists a diagonal matrix $\Omega^\sigma = diag(\omega_1^\sigma, \ldots, \omega_N^\sigma)$ such that

$$\mathcal{H}^{\sigma T}\Omega^\sigma + \Omega^\sigma \mathcal{H}^\sigma > 0 \tag{2.25}$$

Define the following notations

$$\omega_{\max}^\sigma = \max\{\omega_1^\sigma, \ldots, \omega_N^\sigma\}, \tag{2.26}$$

$$\omega_{\min}^\sigma = \min\{\omega_1^\sigma, \ldots, \omega_N^\sigma\}, \tag{2.27}$$

$$\rho^\sigma = \lambda_{\min}(\mathcal{H}^{\sigma T}\Omega^\sigma + \Omega^\sigma \mathcal{H}^\sigma). \tag{2.28}$$

To avoid chattering and zeno behaviour, let us consider dwell time $\tau_d > 0$ such that for all $t \geq 0$, $t_{l+1} - t_l \geq \tau_d$ where $l = 0, 1, 2, \ldots$.

Definition 26. *[171] For any switching signal $\sigma(t)$ and time instants t_1 and t_2 such that $t_2 > t_1 \geq t_0$, let $N_{\sigma(t_2,2_1)}$ describes the number of switching of $\sigma(t)$ over the time interval $[t_1, t_2)$. For any $\tau_a > 0$ and an integer $N_0 \geq 0$, if*

$$N_{\sigma(t_2,t_1)} < N_0 + \frac{t_2 - t_1}{\tau_a} \tag{2.29}$$

holds, then τ_a is called an Average Dwell Time (ADT).

2.5.1 Controller design

The proposed distributed control law is

$$\begin{aligned} u_i(t) &= -\bar{c}\lambda^2 \sum_{j=1}^{N} a_{ij}^{\sigma(t)} \left[\hat{r}_{i,i}(t) - \hat{r}_{i,j}(t)\right] - \bar{c}2\lambda \sum_{j=1}^{N} a_{ij}^{\sigma(t)} \left[\hat{v}_{i,i}(t) - \hat{v}_{i,j}(t)\right] \\ &\quad -\bar{c}\lambda^2 b_i^{\sigma(t)} \left[\hat{r}_{i,i}(t) - \hat{r}_{i,0}(t)\right] - \bar{c}2\lambda b_i^{\sigma(t)} \left[\hat{v}_{i,i}(t) - \hat{v}_{i,0}(t)\right], \quad i = 1, \ldots, N \end{aligned} \tag{2.30}$$

where $a_{ij}^{\sigma(t)}$ is the ij^{th} entry of adjacency matrix $A^{\sigma(t)}$. One should note that as compared to the control input (2.17), $a_{ij}^{\sigma(t)}$ in (2.30) is not constant but changes with the topology. $\hat{r}_{i,j}(t)$ and $\hat{v}_{i,j}(t)$, $i = 1, \ldots, N$, $j = 0, \ldots, N$ are the position and velocity of agent j estimated by agent i and are computed through same continuous-discrete time observer given by (2.18) and (2.19).

Assumption 27. *At each switching instant t_l, $l = 0, 1, \ldots$, every agent of the MAS sends its own estimated states, $\hat{r}_{i,i}(t_l), \hat{v}_{i,i}(t_l)$, with i=0,...,N, to its new neighbors. The observer updates its value at $t = t_l$ based on the estimations it receives from the neighbors i.e $\hat{r}_{i,j}(t_l) = \hat{r}_{j,j}(t_l)$ and $\hat{v}_{i,j}(t_l) = \hat{v}_{j,j}(t_l)$.*

Remark 28. *Assumption 27 is important for the convergence of observer in the case of switching graphs. It ensures that once the observer error reaches zero, it will not diverge due to switching between the graphs. Also, other than switching instants i.e. when $t \neq t_l$, the observer dynamics are governed by (2.18) and (2.19). Furthermore, same observer can be used by a real leader to estimate its own states which it could transmit to the neighbor at the switching instant.*

Theorem 29. *Consider the MAS (2.9)-(2.10) with control input (2.30). Let Assumptions 25 and 27 hold and if the control parameters $\theta, \lambda, \bar{c} > 0$ satisfy the following*

$$\theta < \frac{\bar{\varrho}}{\tau_M} \tag{2.31}$$

$$\lambda < \varepsilon^* \theta \tag{2.32}$$

$$\bar{c} \geq \frac{\max_{p \in \mathcal{M}}\{\omega_{max}^p\}}{\min_{p \in \mathcal{M}}(\rho^p)} \tag{2.33}$$

where $\bar{\varrho}$ is a positive constant, $\varepsilon^* \in (0, 1)$, ω_{max}^p and ρ^p are given by (2.26) and (2.28), respectively and if the leader velocity is constant i.e. $u_0 = 0$ and ADT satisfies the following inequality

$$\tau_a > \frac{8 \ln(\beta K) - 1}{\lambda} \tag{2.34}$$

where $K \geq 0$ and $\beta \geq 1$ are defined in the proof, then the leader-following exponential consensus is achieved.

The proof of Theorem 29 is provided in Appendix B.

Remark 30. *The case when the leader velocity is not constant, i.e. $u_0 \neq 0$ is not discussed here and considered as future work. In fact, when the leader input is nonzero, the MAS will not achieve exponential stability but only achieves practical consensus as shown for the case of fixed topology. The stability proof for such a case is quite complicated due to the extra terms related to leader input u_0.*

2.5.2 Simulations

Let us consider a MAS consisting of one leader, labeled 0, and four followers, labeled $1, \ldots, 4$, and there are 3 possible communication topologies as shown in Figure 2.16. The minimum dwell time (τ_d) is chosen equal to 1 sec. The topologies are switching according to the switching signal shown in Figure 2.17. The adjacency and pinning matrices corresponding to graphs $\mathcal{G}^1$, $\mathcal{G}^2$ and $\mathcal{G}^3$ respectively are:

$$\mathcal{A}^1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \mathcal{B}^1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathcal{A}^2 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \mathcal{B}^2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathcal{A}^3 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \mathcal{B}^3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

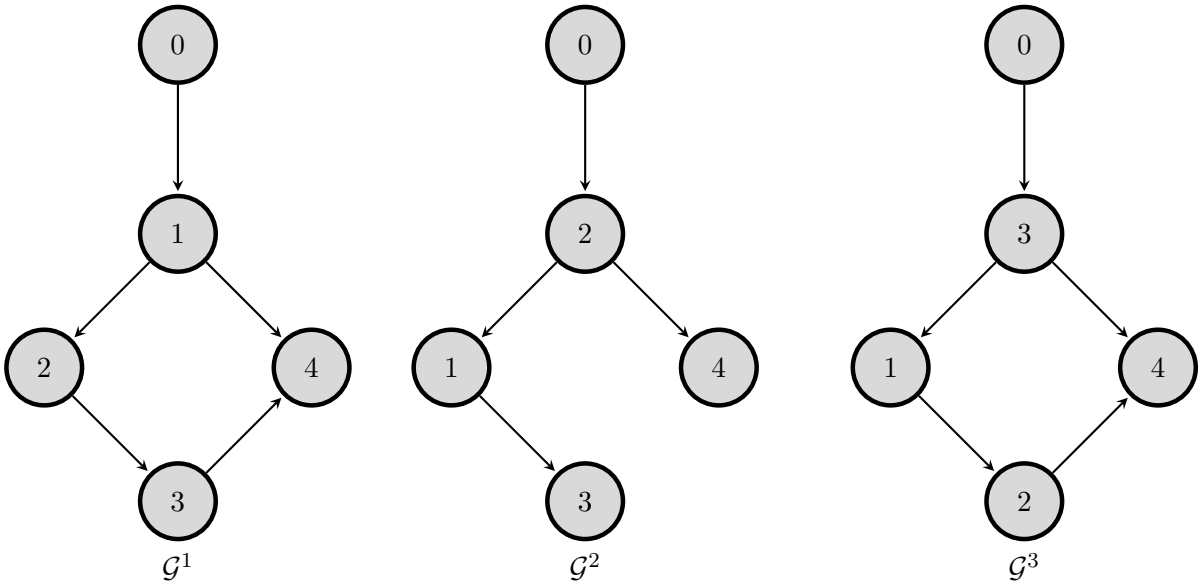


Figure 2.16: Communication topologies

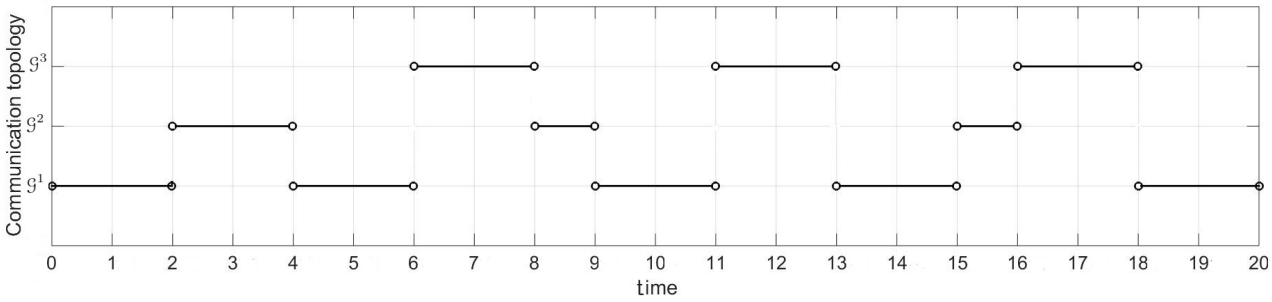
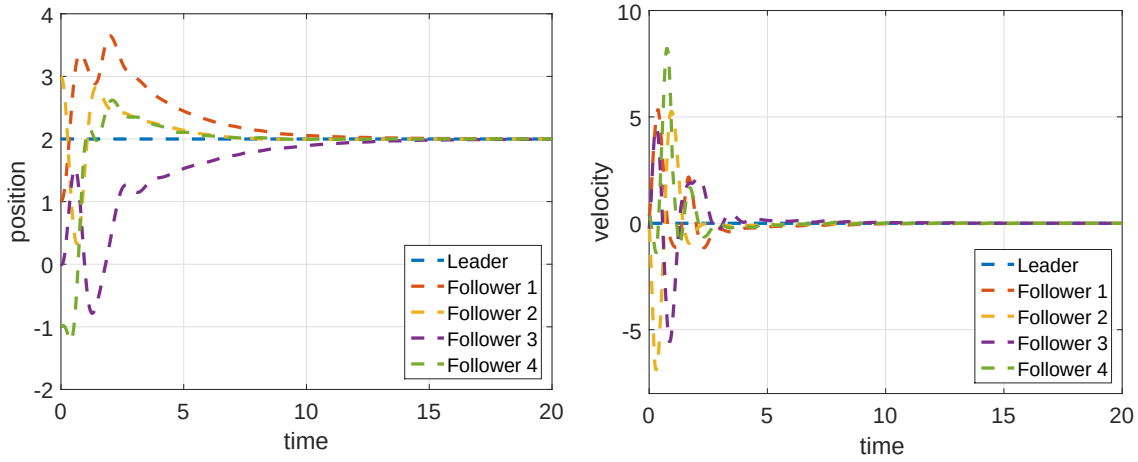


Figure 2.17: Switching signal

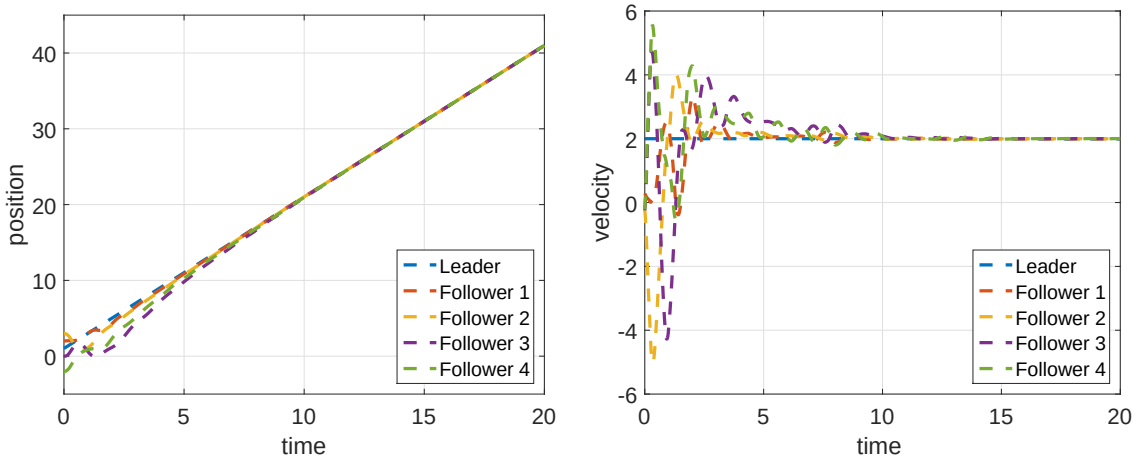
The minimum and the maximum sampling times for simulations are $\tau_m = 0.01$s and $\tau_M = 0.2$s respectively. The initial states of the agents are selected randomly. The tuning gains are chosen by

trial and error method as $\bar{c} = 1.6$, $\lambda = 0.7$ and $\theta = 10$. Leader-following consensus with the static leader is shown in Figure 2.18. Figure 2.18a shows the position tracking while Figure 2.18b depicts the velocity tracking. Similarly, consensus tracking results for ramp leader trajectory is shown in Figure 2.19. In this case, the leader is moving with a constant velocity $u_0 = 2\text{m/s}$. It is evident from these results that the MAS achieves the leader-following consensus with switching communication topology effectively.



(a) (b)

Figure 2.18: Leader-following consensus of MAS under switching topology with the static leader (a) position (b) velocity



(a) (b)

Figure 2.19: Leader-following consensus of MAS under switching topology with ramp leader trajectory (a) position (b) velocity

2.6 Conclusion

In this chapter, an observer-based leader following consensus protocol is developed for a MAS. The designed algorithm provides a leader-following consensus in the presence of communication and sensor constraints. These constraints include irregular and asynchronous sampling periods and unavailability of agent's velocity and input states. A high-gain continuous-discrete time observer is used for the

reconstruction of both position and velocity states from available discrete position information. These estimated states are then used to design the control input.

It is shown through Lyapunov-based stability analysis that the system achieves exponential stability with the proposed algorithm if the input of the leader is zero. However, for non-zero leader input, only practical stability can be achieved where the bound on final tracking errors depends on the maximum bound of the input of the leader.

In the case of practical stability, the final error bound can also be reduced by increasing the controller gain. However, the controller dynamics must remain slower than the observer dynamics. On the other hand, it has been shown that the choice of the observer gain depends on the maximum sampling time. If the system has high maximum sampling time then the observer gain must be low. In other words, the system response will be slower for higher sampling periods.

The obtained results are also extended to the case of switching communication topology. It is shown that if each communication graphs contains a spanning tree with the leader as a root and if ADT respects a certain threshold, the MAS achieves the leader-following consensus with switching topology.

The designed algorithms are verified through MATLAB simulations for different types of leader's trajectories. The obtained simulation results have validated the theoretical results and have shown the efficacy of the proposed protocol.

Chapter 3

Formation tracking and collision avoidance

This chapter deals with the problem of formation tracking and collision avoidance of a MAS with communication constraints. The results of formation tracking are published in IEEE Control and Decision Conference (CDC) 2019 [172], while the results of collision avoidance algorithm are submitted in IEEE Control Systems Letters [173].

Contents

3.1 Introduction

This chapter comprises of two parts. In the first part, the formation tracking problem of MAS under communication constraints is investigated. Same communication constraints are considered here as mentioned in the previous chapters which include availability of position state only and nonuniform and irregular transmission between the agents.

In many practical scenarios, it is required that the agents of MAS create and maintain a desired geometric shape. The required shape could either be fixed or time-varying. In some cases, it is further required that the agents follow a trajectory while maintaining the shape. The trajectory is produced by a virtual or a real leader. This is known as formation tracking. In this chapter, the results of the leader-following algorithm, provided in chapter 2, have been expanded to solve the formation tracking problem for both fixed and time-varying formations. The desired formation is achieved by introducing an offset in the states of the agents and the leader. The offset between the states of the agents and the leader depends on the required formation shape.

The second part of the chapter deals with another important issue in the formation tracking problem that is to avoid collision between the agents while they converge to the desired geometric shape. In this work, we use the Artificial Potential Field (APF) method for collision avoidance. APF method has been used widely for MASs due to its efficiency and simplicity [47, 174, 175]. In APF, an agent is considered as a point in a potential field. This point agent experiences a repulsion force from the obstacles and therefore, instead of colliding with them, it steers around them. Typically, potential functions are based on the relative distance between two agents hence do not require any global information. APF based repulsion mechanism is adequately combined with the proposed formation tracking algorithm to achieve collision-free formation tracking of second-order MAS.

The remaining of the chapter is organized as follows. Section 3.2 describes the formation tracking problem and the proposed algorithm to achieve the fixed and time-varying formation. It also provides the stability analysis of the designed algorithm followed by simulation results. Sections 3.3 discusses the collision avoidance algorithm along with the stability analysis and simulation results.

3.2 Formation Tracking

Let us first consider a group of N followers labeled from 1 to N and one leader labeled 0. The leader could either be real or virtual. The followers and the leader have same double-integrator dynamics as given in Chapter 2 by (2.9) and (2.10) respectively with consideration that leader input u_0 verifies Assumption 17.

Let us consider that the communication graph between the followers is denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V}$ represents the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set containing all the edges. An edge (i, j) for $i \neq j$ exists if agent j can receive data from agent i. The adjacency matrix $\mathcal{A} = (a_{ij}) \in \mathbb{R}^{N\times N}$ of $\mathcal{G}$ with N nodes satisfies that $a_{ij} = 1$ if $(j, i) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. The Laplacian matrix $\mathcal{L} \in \mathbb{R}^{N\times N}$ is defined as $l_{ii} = \sum_{j\neq i} a_{ij}$, $l_{ij} = -a_{ij}$ for $i \neq j$. Let the diagonal matrix $\mathcal{B} = \text{diag}(b_1, b_2, \ldots, b_N)$ be the interconnection relationship between the leader and the followers. If follower i can receive information from the leader then $b_i = 1$ and 0 otherwise. The communication graph including the followers as well as the leader is denoted by $\bar{\mathcal{G}}$. It is defined by

$$\mathcal{H} = \mathcal{L} + \mathcal{B}$$

It is considered that the pinning joint communication topology $\bar{\mathcal{G}}$ has a directed spanning tree where the leader is a root of the tree which means Assumption 19 holds.

Similar to the previous chapter, it is assumed that each agent only measures its position r_i and is unable to measure its velocity v_i as well as its control input u_i. Furthermore, in order to replicate the real world scenario, it is also considered that the agent position is transmitted on irregular and nonuniform time intervals. Moreover, the sampling intervals for transmission are asynchronous. If there is an edge (j, i) between these two agents then the sampling instant at which the position information is transmitted from agent j to agent i is denoted by $t_k^{i,j}$ where $k \in \mathbb{N}$, $i = 1, \ldots N$ and

$j = 0, \dots N$. The sampling instants verify condition (2.11) while the minimum sampling period τ_m and the maximum sampling period τ_M satisfy condition (2.12).

3.2.1 Formation vector

The desired time-varying geometric shape that the followers have to form is defined by:

$$f(t) = [f_1(t)^T, \dots, f_N(t)^T]^T \tag{3.1}$$

where $f_i(t)$ is the formation vector of agent i and is given as:

$$f_i(t) = [f_{i,r}(t)^T, f_{i,v}(t)^T]^T \tag{3.2}$$

for follower $i \in \{1, \dots, N\}$ and satisfies

$$\dot{f}_{i,r}(t) = f_{i,v}(t) \tag{3.3}$$

$f_{i,r}(t) \in \mathbb{R}^m$ and $f_{i,v}(t) \in \mathbb{R}^m$ correspond to the position and velocity offset, respectively. It is worth noting that $f_i(t)$ does not represent global formation coordinates but relative offset vectors with respect to the leader. It means that the desired geometric shape is defined with reference to the leader.

To further explain the formation vector, let us consider the following example. Consider that there are five vehicles in a 2D plane (xy plane) i.e $m = 2$. One of the vehicles is acting as a leader and it is labeled 0. The remaining four vehicles labeled from 1 to 4 are required to make a square around the leader. The distance between the two vehicles is predefined and should not change. Then the corresponding formation vectors for vehicles 1 to 4 with respect to the leader 0 can be chosen as

$$\begin{aligned} f_1 &= [5, 5, 0, 0]^T \\ f_2 &= [5, -5, 0, 0]^T \\ f_3 &= [-5, 5, 0, 0]^T \\ f_4 &= [-5, -5, 0, 0]^T \end{aligned}$$

It is to note that since it is a time invariant or fixed formation, the corresponding velocity offset components in the formation vectors are zero. Figure 3.1 provides a graphical representation of the desired formation in the leader frame (x_0, y_0).

Assumption 31. *It is considered that each agent already has the knowledge of the formation geometry $f(t)$.*

Remark 32. *The information of formation geometry is provided beforehand to the followers and is not shared during the tracking process.*

Remark 33. *Since, in this context of formation tracking, each follower has a distinct desired position to achieve the required shape, hence, the followers are not interchangeable. It is contrary to the consensus where all followers converge to a same position. in this context of formation tracking.*

Definition 34. *The formation tracking problem is said to be practically solved if there exists $\bar{\varepsilon} > 0$ such that*

$$\limsup_{t \to +\infty} \|(x_i(t) - f_i(t) - x_0(t))\| \leq \bar{\varepsilon} \tag{3.4}$$

where $x_i(t) = [r_i(t)^T, v_i(t)^T]^T$ and $x_0(t) = [r_0(t)^T, v_0(t)^T]^T$.

The goal is to design the protocol $u_i(t)$, $i \in \{1, \dots, N\}$ such that the MAS (2.9)-(2.10) achieves the practical fixed and time-varying formation specified in Definition 34 using asynchronous and aperiodic sampled position data.

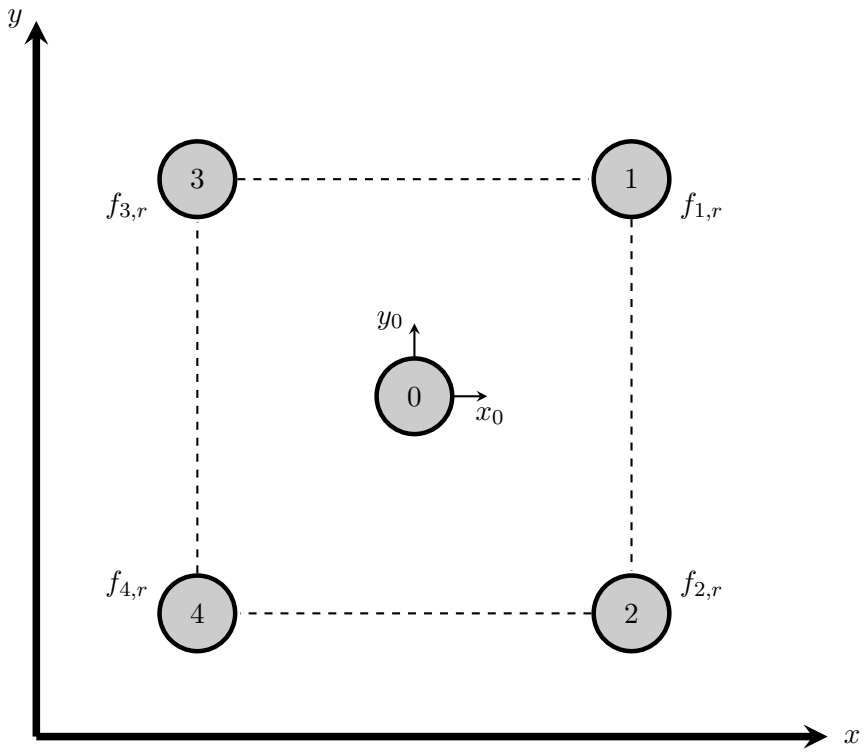


Figure 3.1: Example of square geometric shape.

3.2.2 Output-feedback formation tracking controller

In this section, a time-varying formation tracking controller is designed and analyzed. First, for each follower, a continuous-discrete time observer is proposed to estimate its state and the state of its neighbors from the available local asynchronous and aperiodic sampled position data. Using these estimates, an output-feedback formation tracking protocol is developed.

For each follower $i \in \{1, \ldots, N\}$, the following continuous-discrete time observer is used:

$$\dot{\hat{r}}_{i,j}(t) = \hat{v}_{i,j}(t) - 2\theta e^{-2\theta(t-t_k^{i,j})}\left(\hat{r}_{i,j}(t_k^{i,j}) - r_j(t_k^{i,j})\right) \tag{3.5}$$

$$\dot{\hat{v}}_{i,j}(t) = \dot{f}_{j,v}(t) - \theta^2 e^{-2\theta(t-t_k^{i,j})}\left(\hat{r}_{i,j}(t_k^{i,j}) - r_j(t_k^{i,j})\right) \tag{3.6}$$

where $j = 0, 1, \ldots, N$, $t \in \left[t_k^{i,j}, t_{k+1}^{i,j}\right)$, $k \in \mathbb{N}$ and $\theta > 0$ is the observer tuning parameter. Moreover, $\dot{f}_{0,v} = 0$ as leader does not have any offset. Using the available local asynchronous and aperiodic sampled position data, this observer guarantees the estimation of the state of agent j by agent i. $\hat{r}_{i,j}$ and $\hat{v}_{i,j}$ are the estimated position and speed of the agent j by the agent i. The initial conditions $\hat{r}_{i,j}(0), \hat{v}_{i,j}(0) \in \mathbb{R}^m$ of the observers can be chosen arbitrarily.

Using these estimates, for each follower $i \in \{1, \ldots, N\}$, the formation tracking algorithm is proposed as

$$\begin{aligned} u_i(t) = u_i^f(t) \;=\; & \dot{f}_{i,v}(t) - \bar{c}\lambda^2 \sum_{j=1}^{N} a_{ij}\left[\hat{r}_{i,i}(t) - f_{i,r}(t) - \hat{r}_{i,j}(t) + f_{j,r}(t)\right] \\ & -\bar{c}2\lambda \sum_{j=1}^{N} a_{ij}\left[\hat{v}_{i,i}(t) - f_{i,v}(t) - \hat{v}_{i,j}(t) + f_{j,v}(t)\right] \\ & -\bar{c}\lambda^2 b_i\left[\hat{r}_{i,i}(t) - f_{i,r}(t) - \hat{r}_{i,0}(t)\right] - \bar{c}2\lambda b_i\left[\hat{v}_{i,i}(t) - f_{i,v}(t) - \hat{v}_{i,0}(t)\right] \end{aligned} \tag{3.7}$$

where $\bar{c} > 0$ is the coupling strength and $\lambda > 0$ is the controller tuning parameter.

Remark 35. *It should be noted that due to the presence of the offset f_i, the controller* (3.7) *and the observer* (3.5) *are different from the one given in Chapter 2.*

Theorem 36. *Considering that Assumptions 17, 19 and 31 are satisfied. There exist constants* $\bar{\varrho} > 0$ *and* $\varepsilon \in (0, 1)$ *such that if the observer and controller gains verify*

$$\theta \leq \frac{\bar{\varrho}}{\tau_M} \tag{3.8}$$

$$\bar{c} \geq \frac{\omega_{\max}}{\rho} \tag{3.9}$$

$$\lambda = \varepsilon\theta \tag{3.10}$$

where $\omega_{\max}$ *and* ρ *are given by* (2.13) *and* (2.15) *respectively, then the time-varying formation tracking problem is solved in the sense of Definition 34 using the output-feedback controller* (3.5)-(3.7)*.*

The proof of Theorem 36 is given in Appendix C.

Remark 37. *It is clear from inequality* (C.7) *that the system achieves practical stability where the tracking error is gradually reduced and enters in a ball centered at the origin. The radius of the ball depends on* δ *which means that in case of a static leader and time-invariant formation, the tracking errors converge exponentially to zero. Furthermore, the radius of the convergence ball of the tracking errors can be reduced by increasing the controller gain* λ*. This will also increase the speed of the controller dynamics.*

3.2.3 Simulation results

Let us consider a MAS consisting of one leader, labeled 0, and six followers, labeled $1, \ldots, 6$, described by (2.9)–(2.10). The communication topology between agents is directed and is shown in Figure 3.2. The leader can send its position information only to agent 1. One can also note that agent 4 can receive data from both agent 1 and agent 3.

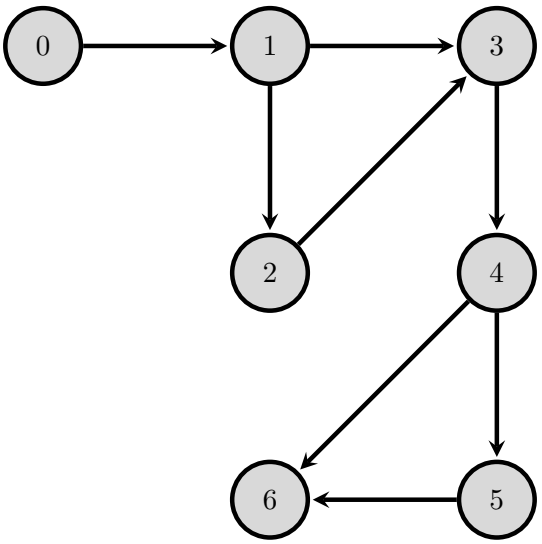


Figure 3.2: Communication topology.

The corresponding adjacency and pinning matrices are given below.

$$\mathcal{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \qquad \mathcal{B} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Hence the Laplacian matrix is computed as follows

$$\mathcal{L} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 2 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

The simulation results are divided into two parts depending on the desired geometric shape which is either time-invariant or time-varying. Different cases for the leader trajectory are also considered. All the simulations are carried out for a 2-dimensional space, i.e. $m = 2$. Therefore, position errors are considered both in $x-$position, and in $y-$position and defined as follow.

$$\begin{aligned} e_x &= |(r_i)_x - (f_{i,r})_x - (r_0)_x| \\ e_y &= |(r_i)_y - (f_{i,r})_y - (r_0)_y| \end{aligned}$$

Figure 3.3 shows the sampling instants at which the position information is transmitted between two agents. One can note that the agent position is transmitted asynchronously with nonuniform sampling periods. The maximum sampling period τ_M in these simulations is 130ms. The observer and controller parameters are chosen by trial and error method and selected as $\bar{c} = 1$ $\theta = 11$ and $\lambda = 1$.

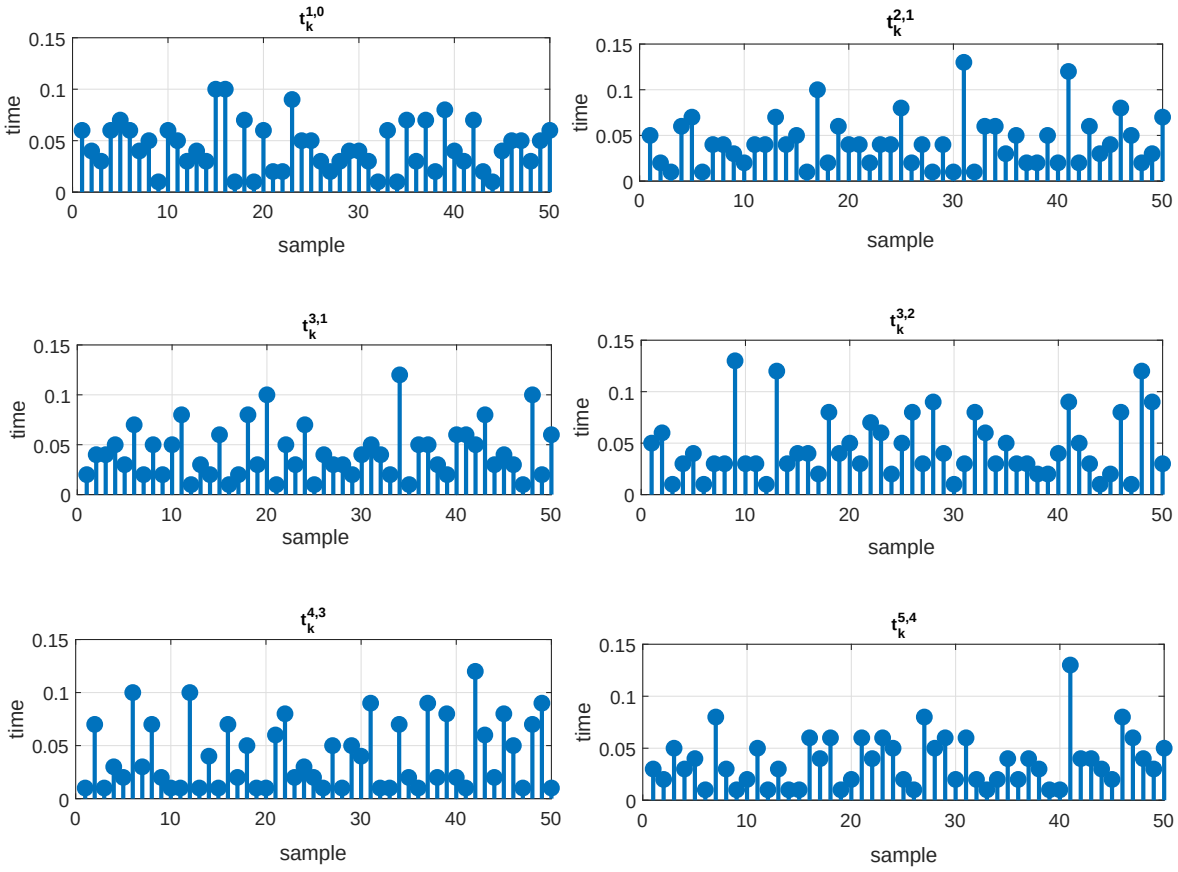


Figure 3.3: Sampling periods for data transmission among the agents

3.2.3.1 Time-invariant formation

A regular hexagon geometric shape is considered for the time-invariant formation case where six followers track the leader while maintaining a constant hexagonal shape around the leader. The

length of each side of the desired hexagonal is $2m$. The formation vector $f = [f_1^T, \dots f_N^T]^T$ is chosen such as:

$$\begin{aligned} f_1 &= [2, 1, 0, 0] \\ f_2 &= [1, \sqrt{3}, 0, 0] \\ f_3 &= [-1, \sqrt{3}, 0, 0] \\ f_4 &= [-2, 0, 0, 0] \\ f_5 &= [-1, -\sqrt{3}, 0, 0] \\ f_6 &= [1, -\sqrt{3}, 0, 0] \end{aligned}$$

Case 1: First, it is considered that the leader is stationary at $(0, 0)$. The tracking result for this case is shown in Figure 3.4a for different time instants. Both x and y position errors are depicted in Figure 3.5a and Figure 3.5b respectively. It is clear from these results that the formation pattern is successfully obtained and all the followers keep the formation for all future time.

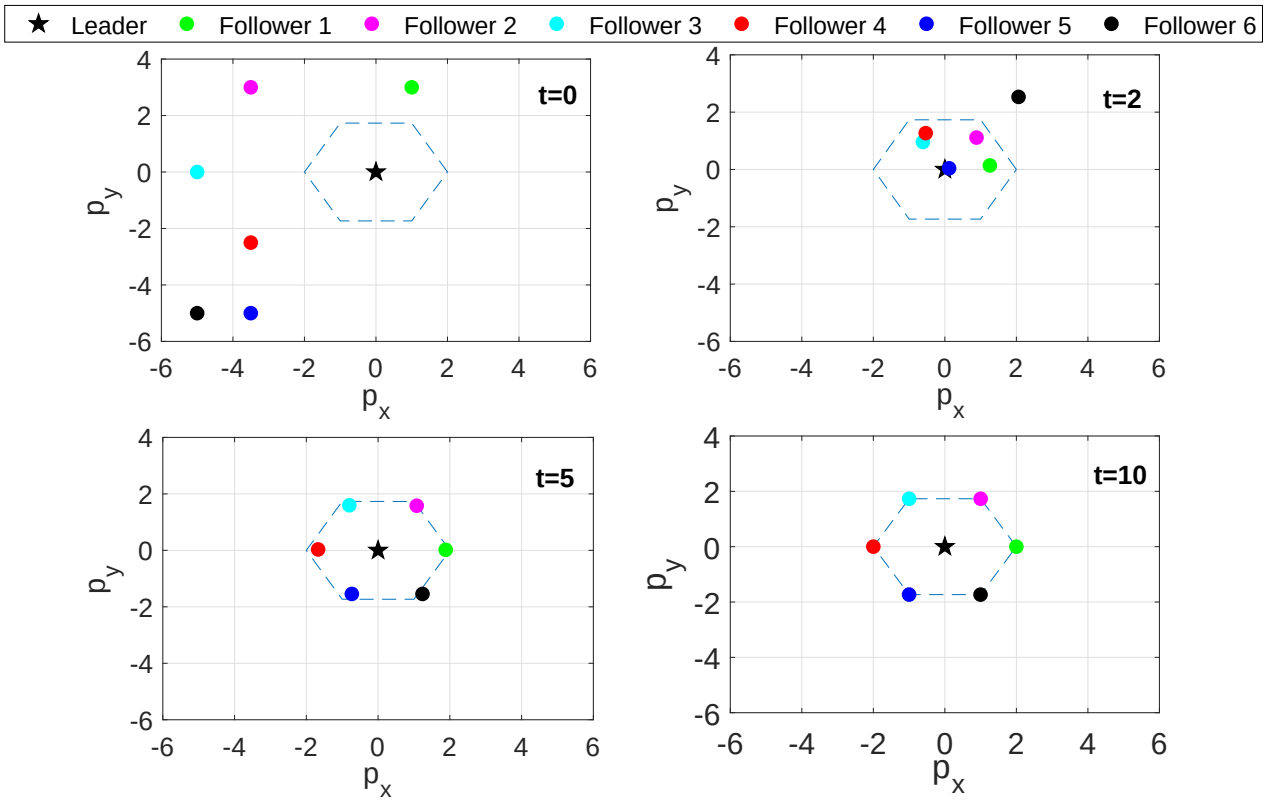
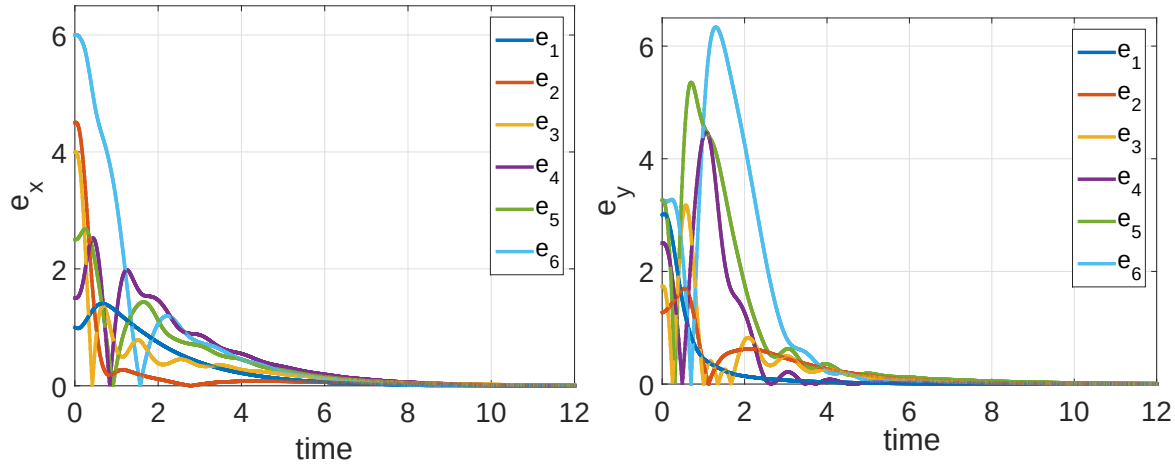


Figure 3.4: Fixed formation tracking with static leader

Case 2: Secondly, it is considered that the leader is moving with constant x and y velocities i.e. $u_o = [0, 0]^T$. The velocities are selected as $u_x = 2\text{m/s}$ and $u_y = 1\text{m/s}$ respectively. The tracking result for this scenario is shown in Figure 3.6 and tracking errors are depicted in Figure 3.7a and Figure 3.7b. It is to note that in both *Case 1* and *Case 2*, exponential stability is achieved.

Case 3: Finally, the case is considered where the leader is given some input. For these simulations, the leader's input is selected as $u_x = 0.03m/s^2$ and $u_y = 0.02m/s^2$. The tracking result for this scenario is shown in Figure 3.8. Since $u_0 \neq 0$, the system achieves only practical stability. This is evident seeing the tracking error results shown in Figure 3.9a and Figure 3.9b.



(a) (b)

Figure 3.5: Tracking error of fixed formation with static leader (a) x-position error (b) y-position error.

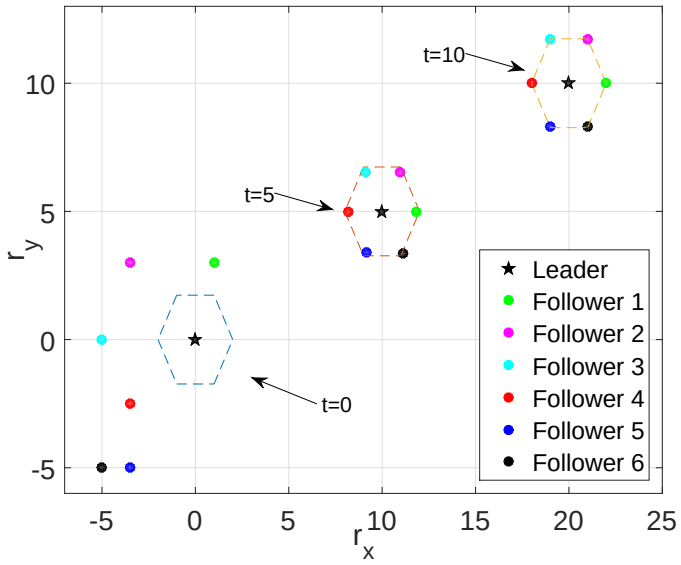


Figure 3.6: Fixed formation tracking with constant leader velocity

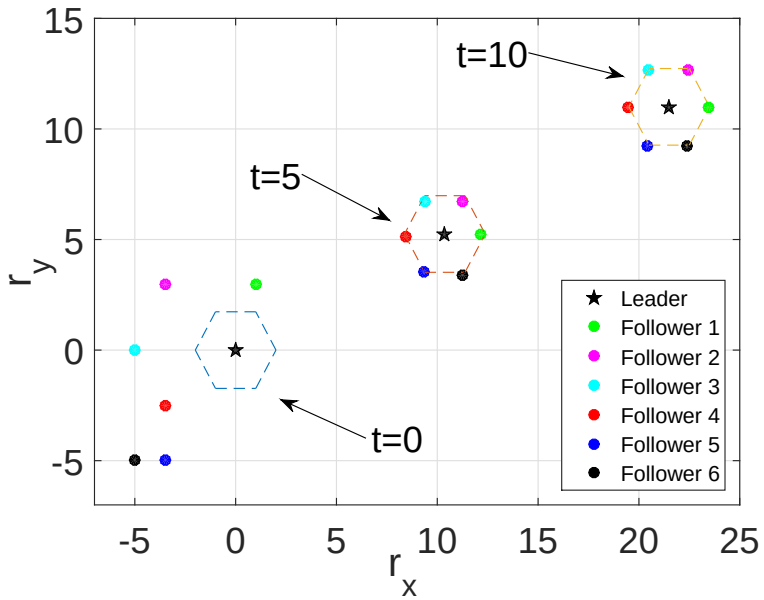
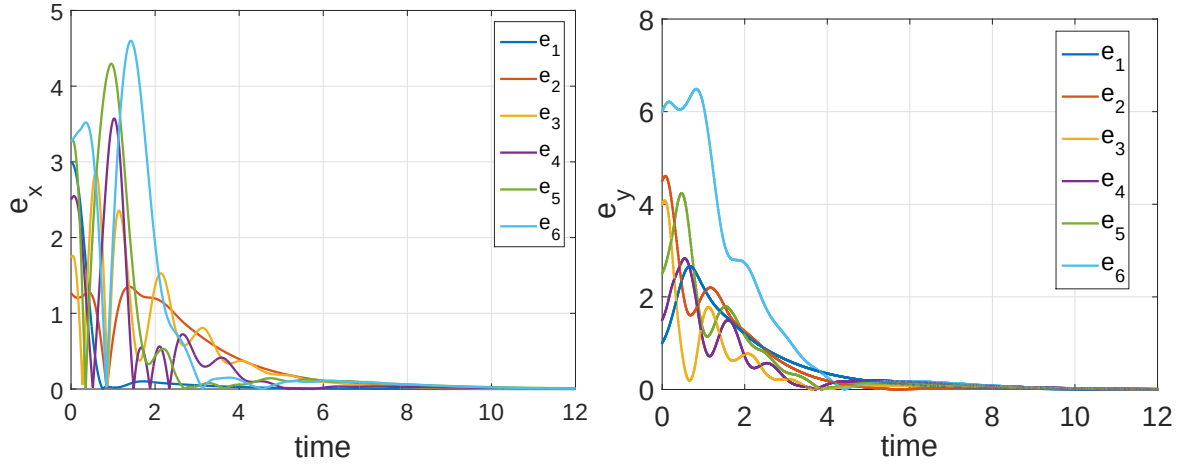
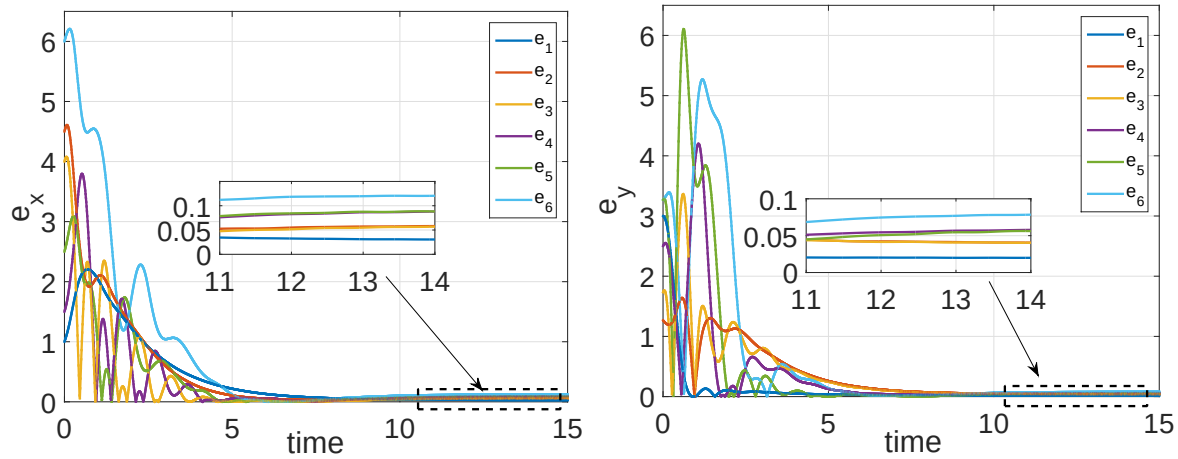


Figure 3.8: Fixed formation tracking with leader input



(a) (b)

Figure 3.7: Tracking error of fixed formation with constant leader velocity (a) x-position error (b) y-position error.



(a) (b)

Figure 3.9: Tracking error of fixed formation with leader input (a) x-position error (b) y-position error.

3.2.3.2 Time-varying formation

The time-varying formation vector is selected

$$f_i(t) = \begin{pmatrix} 10\cos(0.1t + 2\pi(i-1)/6) \\ 10\sin(0.1t + 2\pi(i-1)/6) \\ -\sin(0.1t + 2\pi(i-1)/6) \\ \cos(0.1t + 2\pi(i-1)/6) \end{pmatrix}$$

with $i = 1, \ldots, 6$. If this formation is achieved, the followers will turn in a circle around the leader keeping the distance of $10m$ from the leader. Again, three cases will be considered.
Case 1: In the first case, the leader is stationary and the followers are required to move from their initial positions to form the desired circle and keep moving in that circle for all future time. Figure 3.10 shows the tracking result. The tracking errors for both x and y positions are shown in Figure 3.11.

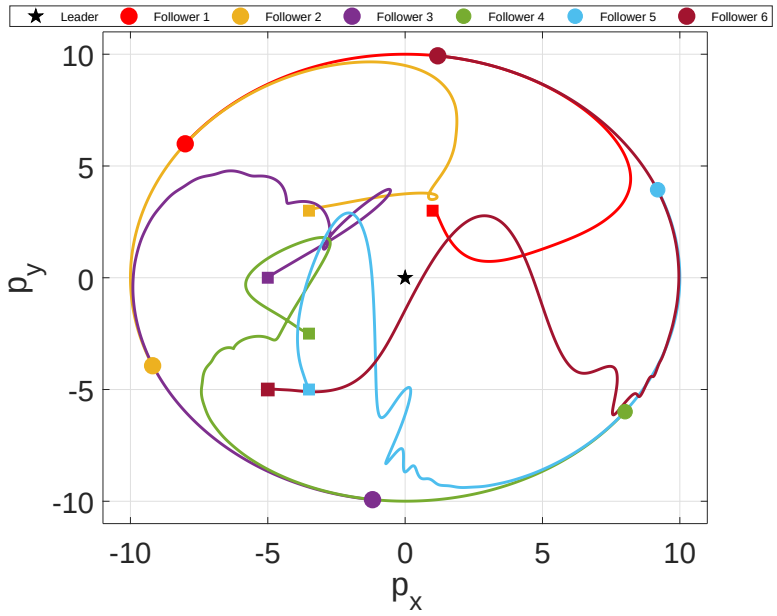
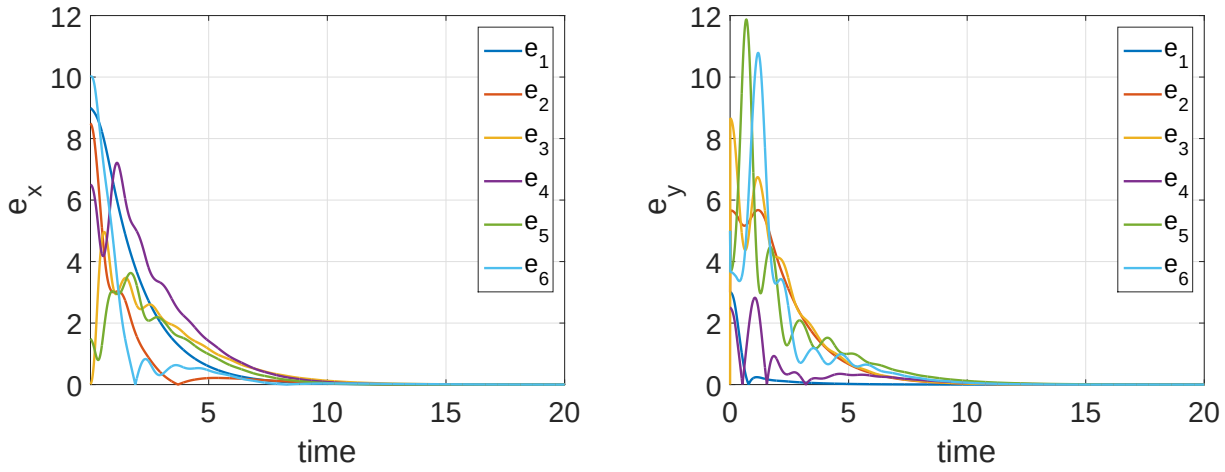


Figure 3.10: Time-varying formation tracking with static leader where □ = initial state and o = final state



(a) (b)

Figure 3.11: Tracking error of time-varying formation with static leader (a) x-position error (b) y-position error.

Case 2: The leader is then moved with constant x and y velocities which are $v_x = 1m/s$ and $v_y = 1m/s$ respectively. Figure 3.12 depicts the formation tracking results for different time instants while the corresponding tracking error results are illustrated in Figure 3.13.

Case 3: Finally, the leader is provided x and y inputs $u_x = 0.03m/s^2$ and $u_y = 0.02m/s^2$. The tracking result for this case is illustrated in Figure 3.14 while the tracking error results are shown in Figure 3.15a and Figure 3.15b. Again, it is clear from the results of time-varying formation tracking that the system achieves exponential stability if $u_0 = 0$ (*Case 1* and *Case 2*) and the system is practically stable only when $u_o \neq 0$ (*Case 3*).

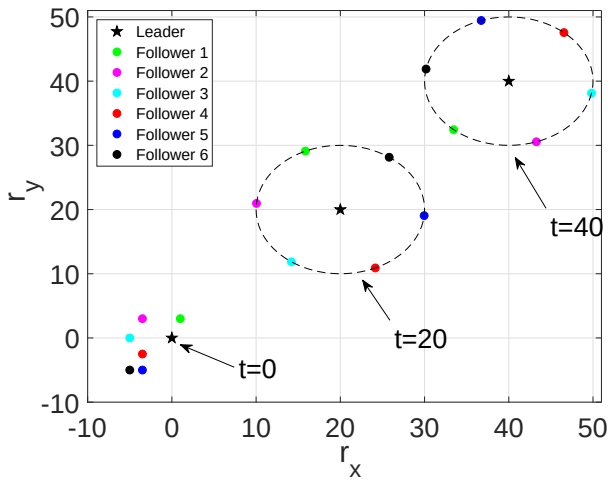
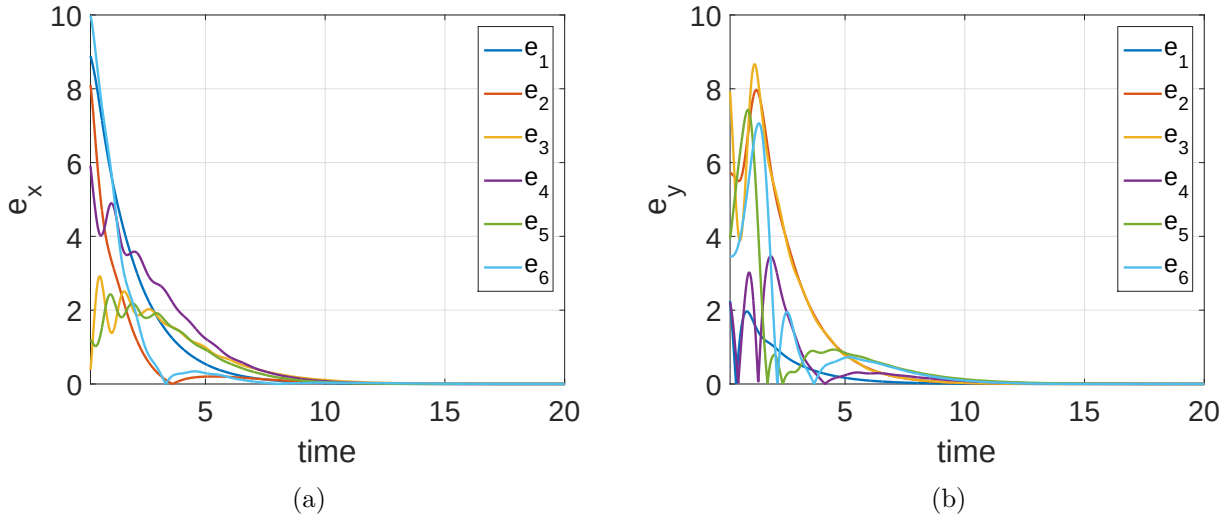


Figure 3.12: Time-varying formation tracking with constant leader velocity



(a) (b)

Figure 3.13: Tracking error of time-varying formation with constant leader velocity (a) x-position error (b) y-position error.

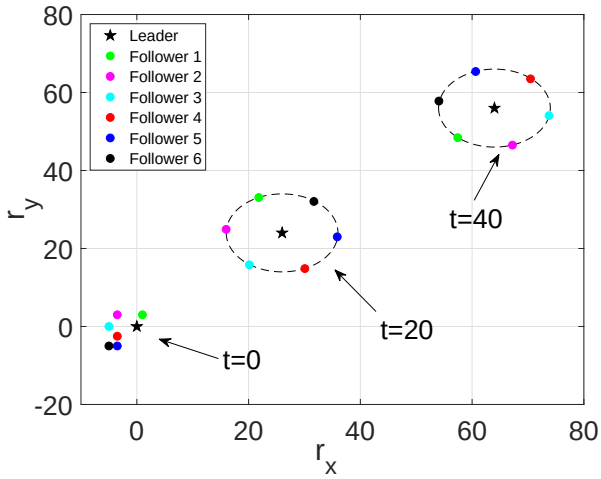
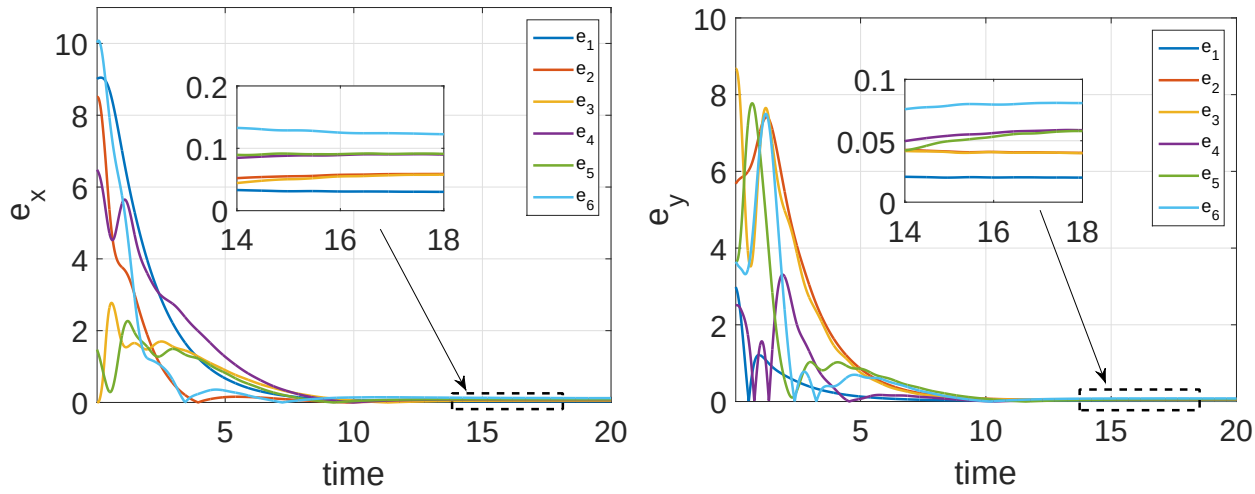


Figure 3.14: Time-varying formation tracking with leader input



(a) (b)

Figure 3.15: Tracking error of time-varying formation with leader input (a) x-position error (b) y-position error.

3.3 Collision avoidance

The distance between the agents is a very important aspect during the implementation of formation tracking controller. A safety distance between agents should be considered while agents converge to the desired position. Indeed, without this constraint, agents could collide and be damaged. For instance, we have shown that one can achieve a desired formation pattern by using a formation tracking control algorithm (3.7) and if the formation pattern is chosen appropriately, the agent will never collide once they build the formation as shown in the simulation results presented in Section 3.2.3. However, the agents can collide with each other during the transient. For example, the inter-agent distance for the formation tracking case presented in Figure 3.4 is shown in Figure 3.16. It is clear from this figure that the inter-agent distances become equal to zero several times during the transition phase which means that the agents have collided at those instants. In practical scenarios, if the agents are physical bodies, such collision could not only destroy the agents but it could be catastrophic in some cases. Therefore, it is of utmost importance to include some collision avoidance mechanisms in order to achieve collision-free formation tracking.

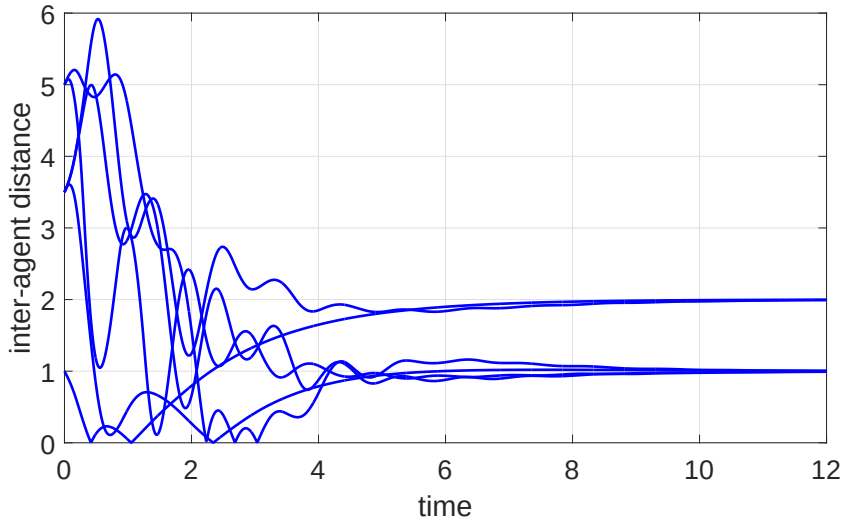


Figure 3.16: Inter-agent distance without collision avoidance mechanism

3.3.1 Artificial potential function

In this section, a repulsion force between agents to avoid collision is introduced using APF. It allows agents to repel each other when the distance between them becomes below a certain threshold. All the agents which are close and in the restricted vicinity of an agent are considered as obstacles. The APF base algorithms are simple in term of implementation and do not required extra computation resources. However, in such techniques the agent may get trapped in local minima and start oscillating instead of reaching the desired position. Nevertheless, in case of MAS when the potential field depends only on inter-agent distances, the chances of trapping in local minima are rare.

Based on the practical aspects, an ideal potential function must have the following properties:

- the range of the potential field must be bounded. Usually, it depends on the range of obstacle sensors mounted on the agent;
- the value of the potential field and the corresponding repulsion must be infinity at the boundary of the obstacle and must decrease with the increase in the distance;
- first and second derivatives of the potential function must exist in order to have a smooth repulsion force.

Assumption 38. *It is assumed that agents are equipped with proximity sensors and can detect any relative position of both non-cooperative and cooperative entities within a sensing range $R > 0$.*

Remark 39. *Sensing capability is required to sense the presence of any other agent in its close vicinity which may lead to a collision. These sensors only give the relative position of any agent within its range in the local frame and do not provide position information in the global frame. It is to note that this assumption is used for the purpose of collision avoidance only.*

3.3.2 Collision free formation tracking of MAS

Let us introduce the following inter-agent distance based potential function [151, 176]

$$q_{ij} = \left(\min \left\{ 0, \frac{\|r_{ij}\|^2 - R^2}{\|r_{ij}\|^2 - 4r^2} \right\} \right)^2 \tag{3.11}$$

where $r > 0$ is the radius of the safety disc around an agent and $r_{ij} = r_i - r_j$. From (3.11), one gets $q_{ij} > 0$ if the distance between agents i and j is less than R. It is also clear that q_{ij} tends to infinity if the inter-agent distance tends to r and $q_{ij} = q_{ji}$. Figure 3.17 shows evolution of the potential function with respect to the inter-agent distance.

The partial derivative of the potential function is

$$\frac{\partial q_{ij}^T}{\partial r_i} = \begin{cases} \frac{4\left(\|r_{ij}\|^2 - R^2\right)\left(R^2 - 4r^2\right)}{\left(\|r_{ij}\|^2 - 4r^2\right)^3} r_{ij}^T & \text{if } 2r \leq \|r_{ij}\| \leq R \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

and

$$\frac{\partial q_{ij}}{\partial r_i} = -\frac{\partial q_{ij}}{\partial r_j} = \frac{\partial q_{ji}}{\partial r_i} = -\frac{\partial q_{ji}}{\partial r_j} \tag{3.13}$$

The total repulsion force on one agent to avoid collision is

$$u_i^r(t) = -\sum_{j=0}^{N} \frac{\partial q_{ij}^T}{\partial r_i} \tag{3.14}$$

The overall formation controller with collision avoidance is then designed as follows

$$u_i(t) = u_i^f(t) + u_i^r(t) \tag{3.15}$$

where $u_i^f(t)$ is the formation tracking controller given in (3.7).

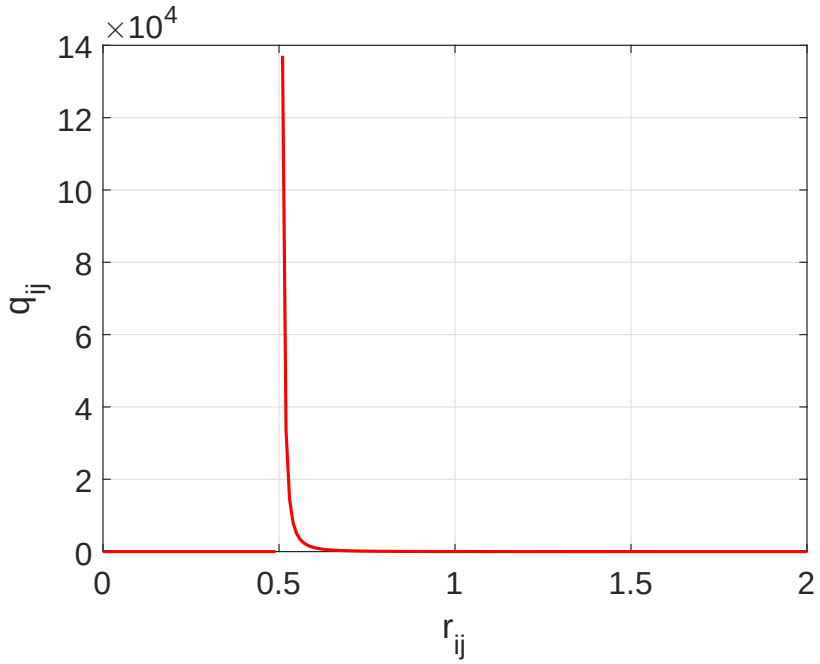


Figure 3.17: Collision avoidance potential function with $r = 0.25$ and $R = 2$

Assumption 40. *The initial configuration of the agents are outside of the detection radius of the others. It means that* $\|r_i(0) - r_j(0)\| > R$ *for all* i, j, $i \neq j$. *Moreover, the formation shape is chosen such that the inter-agent distance in the desired formation remains greater than* R.

Lemma 41. *For the potential function defined by* (3.11), *the following equality holds [177].*

$$\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial q_{ij}}{\partial r_i}v_i + \frac{\partial q_{ij}}{\partial r_j}v_j\right) = \sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i \tag{3.16}$$

Proof.

$$\begin{aligned}&\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial q_{ij}}{\partial r_i}v_i + \frac{\partial q_{ij}}{\partial r_j}v_j\right)\\ &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_j}v_j\end{aligned}$$

Since $\frac{\partial q_{ij}}{\partial r_i} = -\frac{\partial q_{ij}}{\partial r_j}$, one can note that

$$\begin{aligned}&\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_j}v_j\\ &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_j\\ &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i + \frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i\\ &= \sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i\end{aligned}$$

Hence the lemma holds. □

Theorem 42. *Consider the MAS* (2.9)–(2.10) *with formation tracking protocol* (3.15) *and suppose that Assumptions 17, 19, 31 and 40 hold. Then, if conditions* (3.8)-(3.10) *are satisfied, fixed formation tracking is practically achieved without any inter-agent collision.*

The proof of Theorem 42 is provided in Appendix D.

Remark 43. *The proof of Theorem 42 shows that the MAS achieves collision-free fixed formation in practical sense with control law* (3.15)*. In the case of a time-varying formation pattern, the velocity offset $f_{i,v}$ does not equal to zero. The position error defined by* (D.1) *will remain the same. On the other hand, the velocity error* (D.2) *for time-varying formation will become $\zeta_i = v_i - f_{i,v} - v_0$. Due to the extra terms appeared in the error dynamics related to the velocity offset of the formation, the proof becomes more complex. In fact, the chosen Lyapunov functions are not useful for analysis of time-varying formation case and finding a new suitable Lyapunov function is not straightforward. However, in the next section, it is shown through simulation results that collision-free formation tracking for both fixed and time-varying formations can be obtained through the proposed control law* (3.15)*.*

3.3.3 Simulation results

For simulation purposes, the considered network consists of four followers labelled from 1 to 4 and a leader labelled as 0. The directed communication topology among the agents is shown in Figure 3.18. One can note that the leader only sends its position data at random intervals to follower 1 while

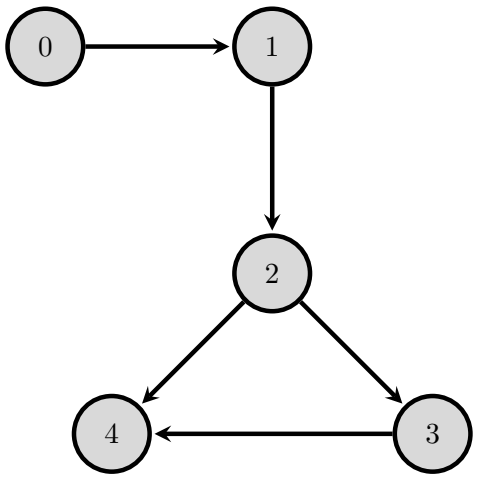


Figure 3.18: Communication topology

follower 4 can receive information from both followers 2 and 3 at irregular and asynchronous sampling times. The corresponding adjacency and pinning matrices are given below

$$\mathcal{A} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \qquad \mathcal{B} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

and the Laplacian matrix is computed as follows

$$\mathcal{L} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & -1 & 2 \end{pmatrix}$$

The minimum and maximum sampling time is $\tau_m = 0.01$s and $\tau_M = 0.2$s respectively. An example of the sampling instants for data transmission between the agents is shown in Figure 3.19.

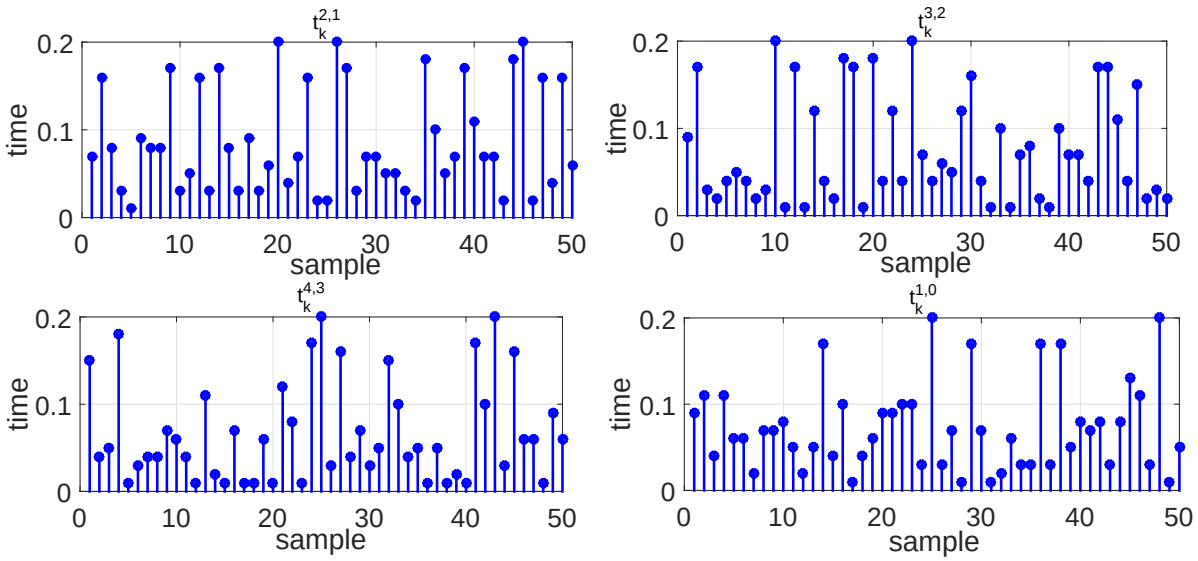


Figure 3.19: Sampling time for data transmission between agents.

The observer and controller gains are chosen as $\theta = 10$ and $\lambda = 0.6$ respectively while the coupling strength $\bar{c} = 1$. The detection region is $R = 2$m while the safety region is $r = 0.25$m for each agent. The simulations are performed for a 2-dimensional space which means $m = 2$. The initial conditions $x_i(0) = [p_{i_x}(0), p_{i_y}(0), v_{i_x}(0), v_{i_y}(0)]^T$ of the followers are:

$$\begin{aligned} x_1(0) &= [-8, -5.5, 0.3, 0.1]^T \\ x_2(0) &= [-6, 0, -0.2, 0.1]^T \\ x_3(0) &= [-4.5, 5, 0.3, -0.1]^T \\ x_4(0) &= [-6, 8, 0.2, 0]^T \end{aligned}$$

One can see that the initial distances between the agents are greater than R.

3.3.3.1 Collision-free fixed formation

The desired formation is chosen to produce a square geometric shape around the leader. The corresponding position offsets for the followers are:

$$\begin{aligned} f_{1,r} &= [6, 6]^T, \\ f_{2,r} &= [6, -6]^T, \\ f_{3,r} &= [-6, 6]^T, \\ f_{4,r} &= [-6, -6]^T \end{aligned}$$

For the first scenario, the leader is kept stationary at position coordinates $(0, 0)$ while the followers reach and maintain the desired square shape around it. Figure 3.20 illustrates the formation producing result. Fig 3.21 explains collision avoidance mechanism for follower 1. The distance between follower 1 and the other agents in the network is shown in Fig 3.21a while Fig 3.21b shows the control input. It can be seen that the repulsion term in the controller activates when the distance between agents is less than $R = 2$m otherwise it remains zero. Figure 3.22 depicts the distance between all the agents during the whole process of formation tracking. The inter-agent distance clearly shows that the agents do not collide during the formation producing. The x and y position errors for this case is shown in Figure 3.23

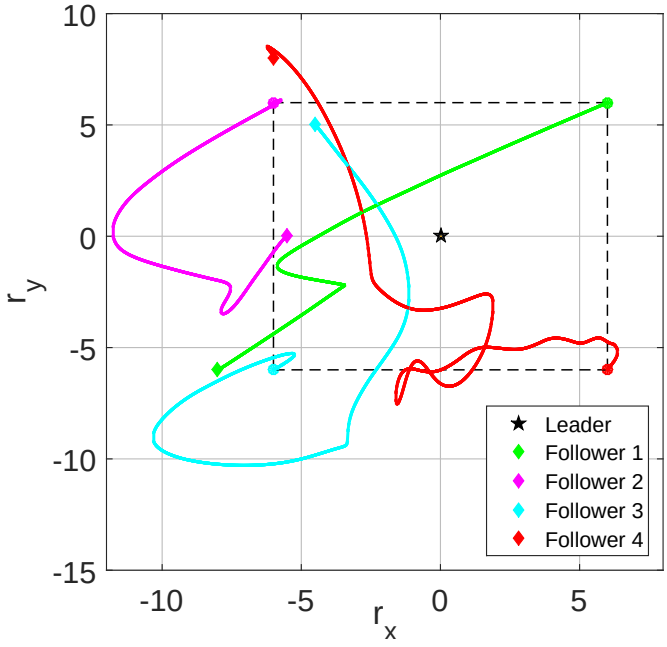


Figure 3.20: Square formation with static leader where $\diamond$ = initial state and o = final state.

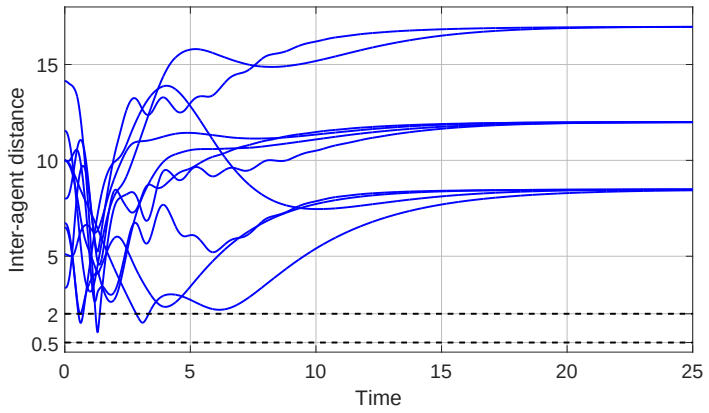


Figure 3.22: Inter-agent distance during formation tracking with static leader

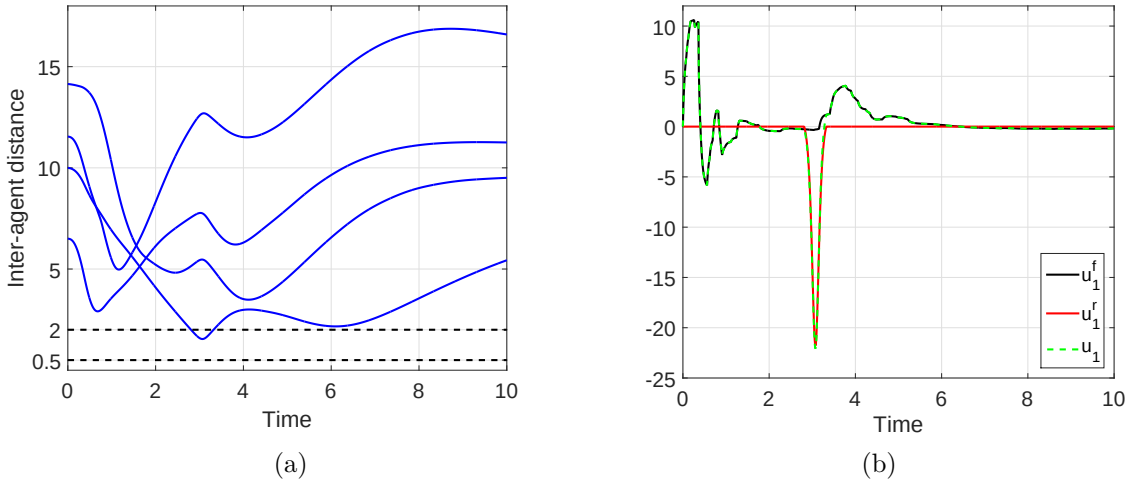
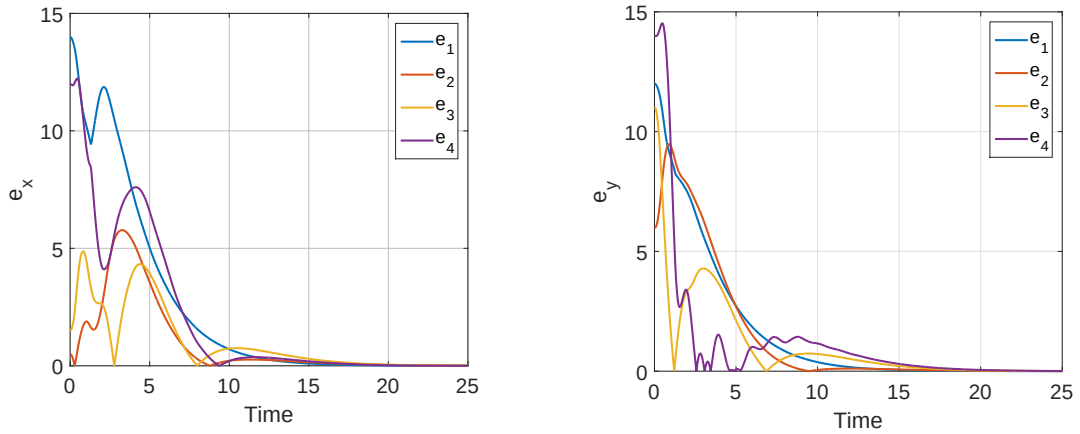


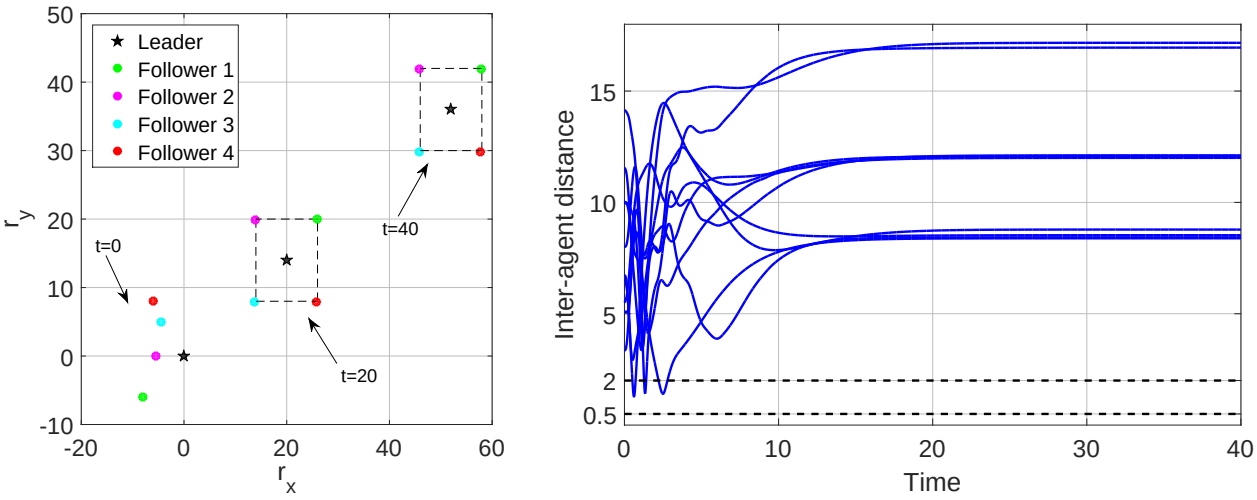
Figure 3.21: Follower 1 (a) distance with other agents (b) control input.



(a) (b)

Figure 3.23: Tracking error for square formation with static leader

For the second scenario, the leader is moving with constant acceleration i.e. $u_{x_0} = 0.03\text{m/s}^2$ and $u_{y_0}(0) = 0.02\text{m/s}^2$. Since the leader moves with constant acceleration, only practical stability is achieved as described in Remark 37. The formation tracking result is shown in Figure 3.24a while the inter-agent distances are depicted in Figure 3.24b. Figure 3.25 shows the corresponding tracking errors.



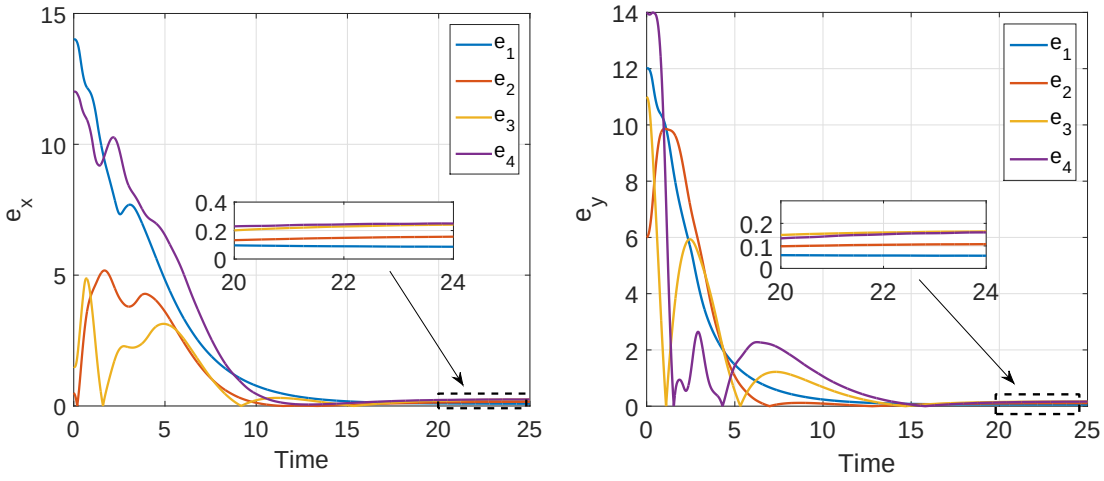
(a) (b)

Figure 3.24: Square formation with leader input (a) formation tracking (b) inter-agent distances.

3.3.3.2 Collision-free time-varying formation

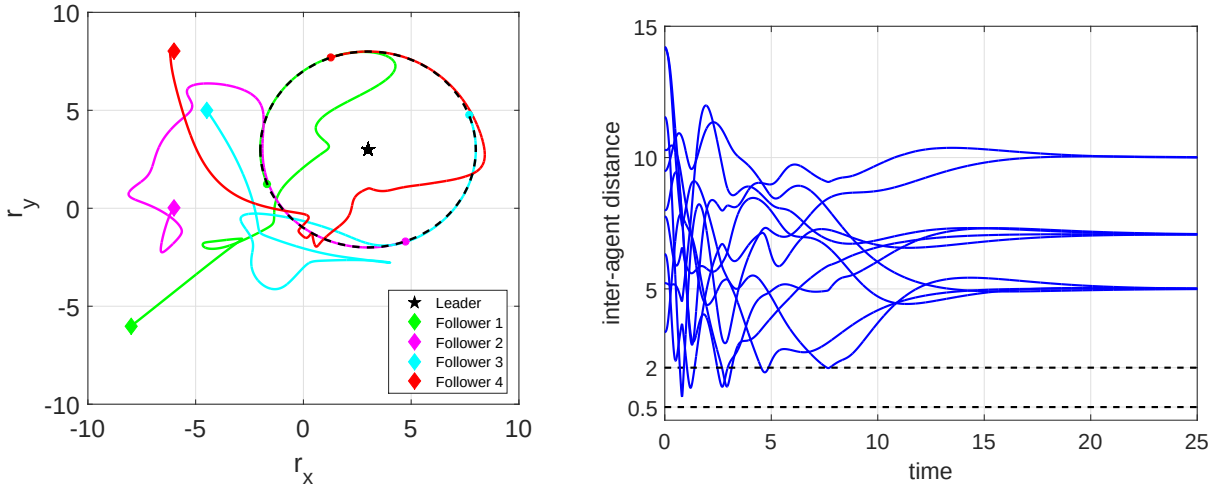
The desired formation for the followers is to make a circular pattern with a radius of 5m around the leader. Figure 3.26a and 3.26b show the formation tracking results and the inter-agent distances respectively when the leader is static at position coordinates $(3, 3)$. The corresponding position errors are shown in Figure 3.27.

In the second scenario, the leader is moving with constant acceleration i.e. $u_{x_0} = 0.03\text{m/s}^2$ and $u_{y_0}(0) = 0.02\text{m/s}^2$. Only practical stability is achieved in this case. The formation tracking result is shown in Figure 3.28a while the inter-agent distances are depicted in Figure 3.28b. Figure 3.29 illustrates the corresponding tracking errors. It can be seen from these results that the collision-free formation tracking is achieved even for time-varying formation shapes with the proposed algorithm.



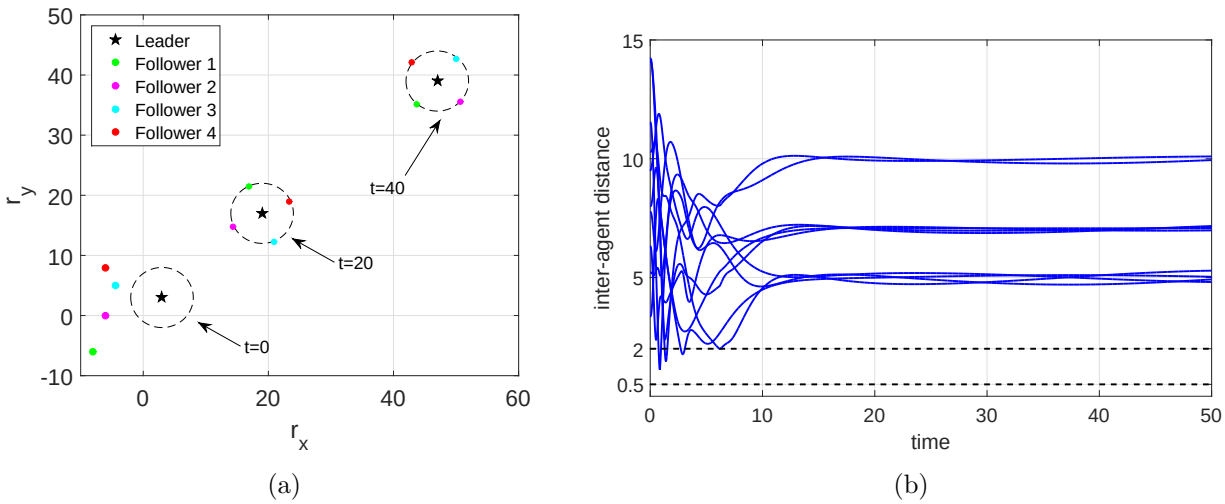
(a) (b)

Figure 3.25: Tracking error for square formation with constant leader acceleration



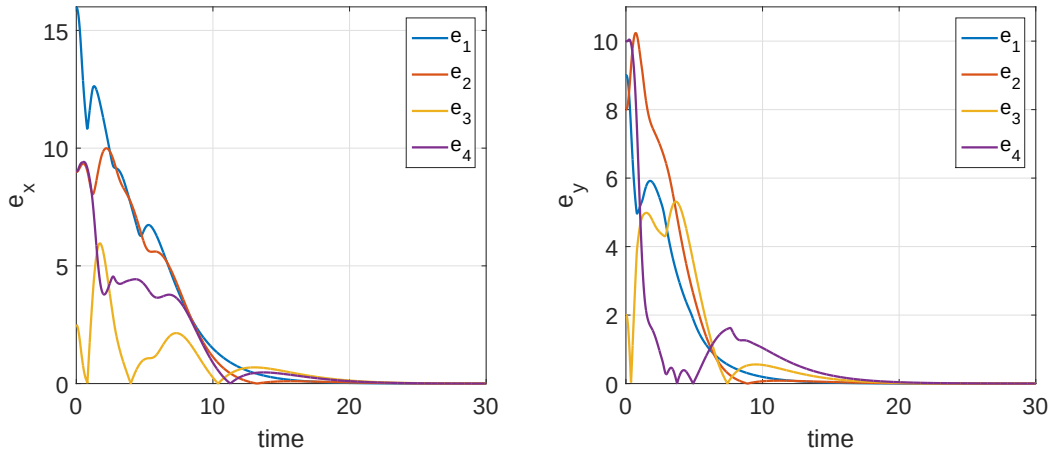
(a) (b)

Figure 3.26: Circular formation with static leader (a) formation tracking $\diamond$ = initial state and o = final state (b) inter-agent distance



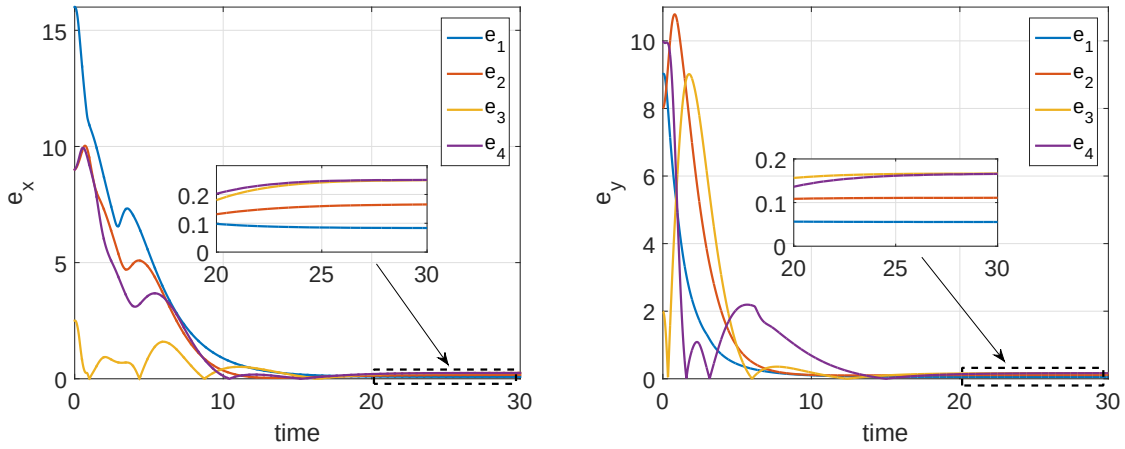
(a) (b)

Figure 3.28: Square formation with leader input (a) formation tracking (b) inter-agent distances.



(a) (b)

Figure 3.27: Tracking error for circular formation with static leader



(a) (b)

Figure 3.29: Tracking error for circular formation with constant leader acceleration

3.4 Conclusion

This chapter deals with the formation tracking problem of MAS under communication constraints. A consensus based distributed formation tracking algorithm is proposed to achieve both fixed and time-varying formation shapes. It is shown that the followers in a MAS achieve a desired geometric shape whether fixed or not and track the trajectory of the leader effectively while maintaining the desired pattern.

In the second part of this chapter, an APF based collision avoidance mechanism is combined with the proposed formation tracking algorithm to avoid any collision among the agents during their transition to the desired shape. Whenever the distance between the agents decreases below a certain value, the repulsive potential acts on the agents and move them away from each other. The repulsive force vanishes as soon as the inter-agent distance becomes sufficiently large. The effectiveness of the proposed collision-free formation tracking algorithm is shown through simulation results.

Chapter 4

Application to multi-robot network

Contents

4.1 Introduction

In this chapter, we study the application of the proposed algorithms of leader-following consensus, formation tracking and collision avoidance. The theoretical results are tested and validated on a fleet of differential drive mobile robots. An open-source software architecture Robot Operating System (ROS) is used for the implementation purpose. The simulations for the robotic fleet are carried out in Gazebo simulator while the experiments are performed in Laboratory of Industrial and Human Automation Control, Mechanical Engineering and Computer Science (LAMIH), Université Polytechnique Hauts-de-France. One robot in the fleet is considered as a leader while other robots are designated as followers. The robots communicate through a wireless network using the TCP/IP protocol. For the implementation of distributed algorithms, the limited information exchange is achieved by simply restricting the use of information received to a robot from only a certain member of the fleet. A control scheme has been designed to deal with the motion constraints of the robot in the Cartesian space. The proposed formation tracking and collision avoidance algorithms have been applied on a multi-robot system using the designed control scheme.

The remaining of the chapter is organized as follows. First, we present the robotic platform used for implementation purpose in Section 4.2 followed by a brief introduction of ROS and Gazebo in Section 4.3 and 4.4 respectively. Section 4.5 elaborates the multi-robot ROS network while Section 4.6 explains the experimental setup. Then, in Section 4.7, experimental results of the developed consensus algorithms are discussed. Robot nonholonomic model and its control scheme are presented in Section 4.8 and 4.9 respectively. Formation tracking and collision avoidance results are illustrated in Section 4.10 and 4.11. Finally Section 4.12 gives a brief conclusion.

4.2 Robotic Platform

The robotic platform used for the experiments is Mini-Lab by ENOVA ROBOTICS shown in Figure 4.1. It is a multi-functional differential drive robot particularly designed for educational and research purposes. The software architecture of the robot is open-source and based on ROS which makes it a good candidate to test the designed algorithms. The Mini-Lab is a compact-sized robot with dimensions 409mm $\times$ 364mm $\times$ 231mm (width$\times$length$\times$height) and weighs 11.5kg. With such body proportions, it is mostly useful for indoor operations. The schematic layout of the Mini-Lab robot is illustrated in Figure 4.2.

The robot can handle a payload of up to 3kg. Separate drive motors are directly attached with the axis of each wheel. The robot includes sensors which provide accurate odometry location and it also has a camera mounted on it for real-time video streaming. Specifications of Mini-Lab robot are provided in Table 4.1.

Figure 4.1: Mini-Lab robot.

The main hardware components of Mini-Lab are listed below.

- **Central processor:** The central processing unit for the control of Mini-Lab is based on the

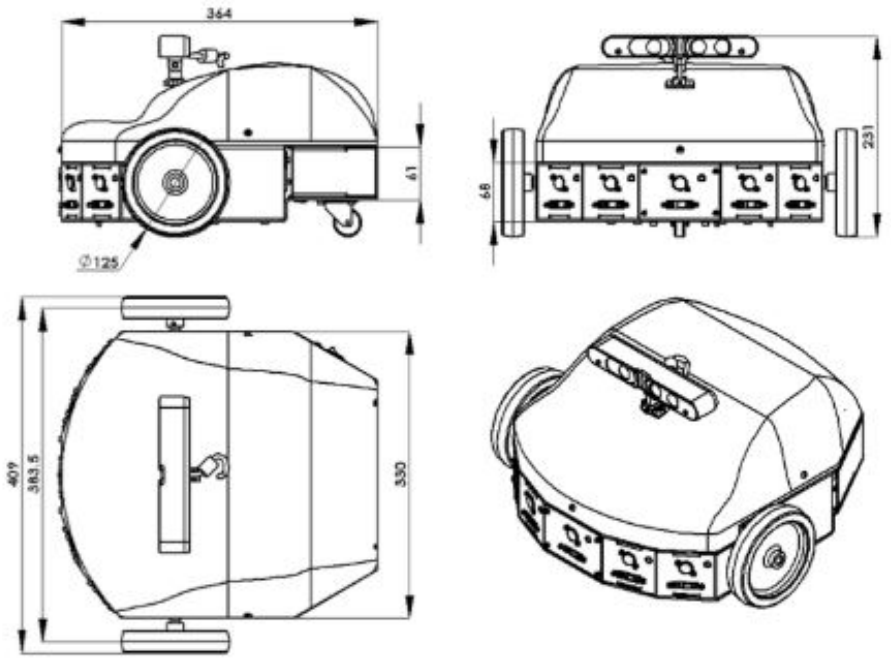


Figure 4.2: Mini-Lab physical dimensions.

DN2800MT Mini ITX Motherboard. The motherboard consists of an Intel Atom N2800 microprocessor. It is a 1.86GHz dual core 64bit microprocessor which supports 4GB DDR3-SDRAM Memory that is available on the motherboard. The board also has VGA, USB 2.0, USB 3.0 and HDMI ports. The processing unit is quite compact in size with dimensions 17cm × 17cm.

- **DC motors:** The robot gets its mechanical driving force through two 12V DC driving motors, attached on each wheel axis and can be controlled separately. Each motor provides a max torque of 0.07N. The robot can move with the speed of 1.5m/s when motors are running at their peak capacity. However, due to safety reasons, maximum allowable speed is limited to about 1m/s. These motors allow robots to climb a slope with maximum angle of 10 degrees.
- **Motor driver:** Dual channel Roboteq SDC2130 controller is used to drive the motors of the robot. It takes input command signals from the processor and generates a high current and high voltage output to drive the motors.
- **Ethernet router:** Each robot can connect to the network through an on-board TL-WR802N wireless router. The router provides wireless connection using TCP/IP protocol.
- **Sensors:** The robot is equipped with two types of proximity sensors. There are five Ultrasonic MaxSonar-EZ0 sensors with a range of 0m to 6.45m and a resolution of 2.54cm. The other sensor is Infrared SHARP 2Y0A21 F 46 with a range between 10 to 80cm. Each robot is also equipped with 16bit encoders.
- **Secondary processor:** The proximity sensors interface is provided through a secondary processor. An Arduino micro-controller board is used for this purpose.
- **Camera:** Each robot is also equipped with Orbbec Astra Pro Camera. The camera provides a 3D RGB image stream with depth measurements and ideal for image processing related applications. The range of the camera is from 0.6m to 8m.
- **USB HUB:** All the hardware devices communicate through a D-Link DUB-H7 USB HUB. It has a transfer rate of up to 480 Mbps, hence provides an ideal solution for transferring data between the central processor and other devices like motor driver and camera etc.

- **Battery:** Each robot has a 12V rechargeable, maintenance-free led-acid battery. It provides sufficient electric power to the robot for up to 4 hours.

Specifications		
Mechanical	Dimensions (W × L × H)	409 × 364 × 231 mm
	Weight	11.5kg
	Load Capacity	3kg
	Max Speed	1.5m/s
	Max Slope Angle	10 degree
Electronics	Processor	Intel Atom N2800
	Sensor Interface	Ardino
	Depth Camera	Asus Xtion Live Pro
	Sensors	Ultrasonic (×5) Infrared (×5)
Communication	Wireless	IEEE 802.11b/g/n-
	Extension with	USB, Ethernet
Power	Battery	12V
	Autonomy	4h
	On-board Voltages	5V/12V

Table 4.1: Specifications of Mini-Lab robot.

4.3 Robot operating system (ROS)

ROS is an open-source meta-operating framework which provides software tools and libraries specifically developed for robotic applications. The main idea behind ROS is to provide a standard framework which gives basic software architecture and tools that can allow robotic researchers to built their complex algorithms on top of it instead of starting from the scratch. ROS was initially developed by researchers at the Stanford Artificial Intelligence Laboratory in 2007 [178]. From 2008, the development of ROS was continued over the next six years at the research institute Willow Garage along with the collaboration of 20 other institutes. Currently, Open Source Robotics Foundation (OSRF), now known as Open Robotics, is the primary maintainer and manager of ROS. ROS attracted the interest of researchers from around the globe and robot manufacturers also opted their products to be compatible with ROS.

ROS provides various services which includes hardware abstraction, low-level device control (device drivers), implementation of common functions, package organization and inter-process message exchange. One of the main reasons that ROS has become popular among the robotic research community is its modular approach which has simplified the designing of complex robot behavior. ROS offers a framework that gathers the robotic tools and allows code sharing and reuse by setting the de facto standards of robot programming. In fact, ROS is a multi-lingual framework where modules can be written in various languages like C++, Python and Lisp. The underlying ROS open-source principle is that one can easily take a code from ROS repositories, use it, improve it and then share it again. One of the important features of ROS is that it supports a distributed computation environment which means ROS processes can be run on different machines and they can share information with each other. The main ROS client libraries are supported for Unix-based systems mainly due to their open-source software dependencies.

The main components of ROS architecture are discussed below

4.3.1 Package

Packages are the basic organization units of software in ROS. A package can consist of ROS processes (Nodes), configuration files, a dependant library, a data-set or any other files that might be useful for the package. The design and structure of packages provide a well-organized ease to use functionality such that the software can be easily reused for other projects as well. This kind of configuration and organization enables ROS modular approach with packages as building blocks of the module.

A typical package structure usually consists of the following files and folders.

- **package.xml file:** It is a manifest file of the package that includes information like package name, licence details, authors, dependencies on other packages etc.
- **CMakelist.txt file:** It contains a list of cmake rules describing how to build and compile the code.
- **src folder:** It is a source folder which contains all the source codes written in C or Python languages.
- **launch folder:** It contains launch files. The launch files are used to launch multiple processes simultaneously.
- **msg folder:** This folder contains message types that can be used to share information among processes. The message types are defined data structures depending on the information they carry.

Sometimes, it is more feasible to combine different packages in a **meta-package**. A meta-package is a stack that provides an aggregate functionality to achieve an overall goal by a robot. Each meta-package has its own manifest file.

4.3.2 Node

Every process in ROS is separately designed and programmed and is referred to as a node. Each node is an executable that performs computations and is written using ROS client libraries like rospy and roscpp. An overall robot control system could have multiple nodes for different tasks. For instance, one node for wheel motor control, one node for path planning, one node for sensor data acquisition, one node for localization and so on. ROS nodes communicate and exchange data with each other through ROS communication methods. A node uses publisher or subscriber to broadcast or receive data respectively. Since 'Node' has a different interpretation in graph theory, from here on-wards we will use ROS-node to refer to a node in ROS context hereafter in order to avoid confusion.

4.3.3 Master

The ROS master is responsible for the registration of the ROS-nodes. Whenever a ROS-node is activated, it looks for the ROS master and registers its name with it. Therefore, the ROS master has the details of all running ROS-nodes on the ROS network. If a ROS-node changes its details, it generates a callback and updates the latest details. If ROS-nodes are run on different machines, then at least one ROS master should run on one of the machines and then all ROS-nodes can find each other through it. ROS Master can be started by a command *roscore*. The command *roscore* not only runs ROS Master but also some other ROS-nodes and programs which are a prerequisite for any ROS based system.

4.3.4 Topics and messages

ROS-nodes in a ROS system communicate and share data in the form of messages. These messages are transported on named channels called topics. Whenever a ROS-node wants to share data, it will **publish** the data in messages on an appropriate topic and the ROS-node that requires this information will **subscribe** to this topic. Each topic has a unique name with a specific message type.

Any ROS-node can subscribe to this topic and also can publish messages with the right message type. A publisher ROS-node keeps publishing on a topic with a specified data rate and does not require any request from the subscriber. Similarly, a subscriber ROS-node can subscribe to any topic even if the publisher is not publishing any message. Usually a topic has one publisher and multiple subscriber ROS-nodes. The ROS master take cares of all the communication but messages are directly sent from publishers to subscribers. Figure 4.3 illustrates a basic communication architecture of ROS.

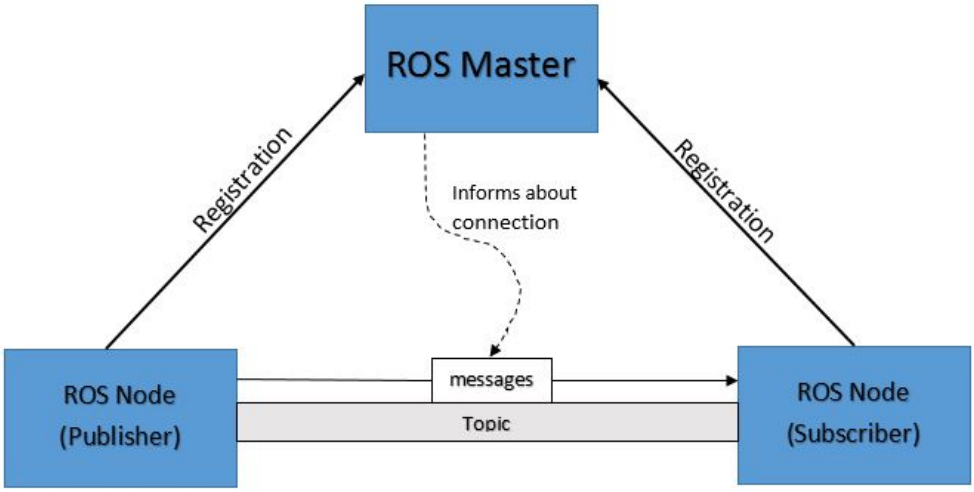


Figure 4.3: Basic ROS communication architecture.

A ROS message in fact is a simple data structure describing data types. These data type descriptions can be used to generate source codes. All standard data types are supported like int, float, double, boolean etc as well as arrays of primitive types. A message can also include a nested structure and arrays. Two other important builtin primitive types are `time` and `duration`. These types are provided by roslib (base dependency of client libraries and tools) as `ros::Time` and `ros::Duration`. A `time` is a specific moment while a `duration` is a period between two instants.

A ROS message description is saved in *.msg* files in the *msg* folder in a ROS package. A ROS message may include a special type called Header. The header contains some important message information like time, frame ID and sequence number. A simple header example is given below

```
uint32 seq
time stamp
string frame_id
```

Since the proposed algorithms of consensus and formation tracking in this research require exact time of position data, Header of the messages containing position information can be used to extract the exact time when that position is measured.

4.3.4.1 Graphical visualization about publishers and subscribers

The communication in a ROS system can be visualized graphically using `rqt_graph`. `rqt_graph` is a Graphic User Interface (GUI) plugin which provides a graphical visualization about publishers, subscribers and topics relationship. This tool is quite useful for debugging especially when a large number of ROS-nodes are running in the ROS application. Figure 4.4 shows an example of simple rqt_graph for a well-known turtle robot.

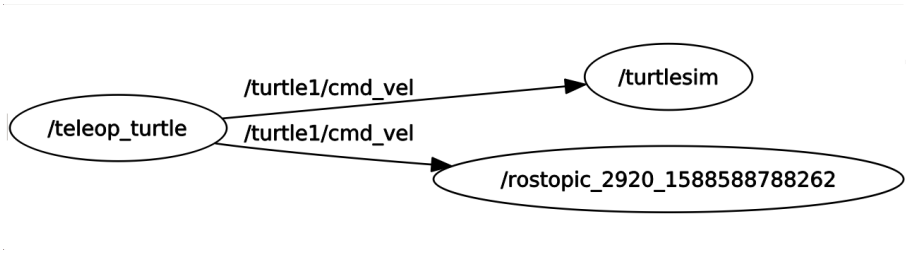


Figure 4.4: rqt_graph of turtle robot simulation.

4.3.5 Services

In some robot applications, the publish and subscribe method for sharing information is not enough, particularly when a request-response interaction is required. This is achieved by ROS services. A service definition consists of two components, one for request and other for response. One ROS-node acts like a server which responds to the request of a client ROS-node.

4.3.6 Bags

ROS message data can be logged and saved in bag files. Bags are important to store sensor data or any other information. Bags basically subscribe to one or more topics and store the data as received. These files can be played back again and the store messages will be published in the same sequence on the topics from which they are received.

4.3.7 Launch file

Launch files are used to launch multiple ROS-nodes simultaneously. Launch files are written in XLM and are saved in `launch` directory of a package with the extension `.launch`. It is run through `roslaunch` command. `roslaunch` automatically starts `roscore` if not activated already. The Launch file not only runs a ROS-node but it can configure ROS-node parameters as well. Moreover, a launch file can also run other launch files. Any package with more than one ROS-node likely to have a launch file to specify and configure them.

4.4 Gazebo Simulator

Gazebo [179] is an open-source robot simulator which provides realistic 3D simulation environments to test and verify robot algorithms. Gazebo allows to design a virtual dynamic environment which includes robots, sensors and other objects. Gazebo simulator mimics real-world scenarios to a great extent by providing a realistic sensor feedback and interaction between the robots and the objects. It incorporates various natural world features like gravity, inertia, collision or contact forces and friction by using ODE and physics engines. Gazebo also has a sophisticated GUI plugin which allows a user to monitor the robot behavior in real-time. It also allows a dynamic environment with multiple robots working simultaneously. Figure 4.5 shows a screen-shot of the Gazebo GUI with three Mini-Lab robots.

Gazebo basically is a stand-alone simulation software. However, now Gazebo provides a convenient interface with ROS. The integration between ROS and Gazebo is achieved through a specific ROS packages set called `gazebo_ros_pkgs`. These packages provide the required interfaces between Gazebo APIs and ROS messages and services. A Unified Robot Description Format (URDF) file can be used for the description of the robotic platform model. URDF is an XLM format file generally used in ROS to provide kinematics and dynamic description of a robot. The details of the URDF file can be found in [180]. Gazebo, by default, uses another format for robot description known as a Simulation Description Format (SDF). However, Gazebo can convert a URDF to a SDF automatically.

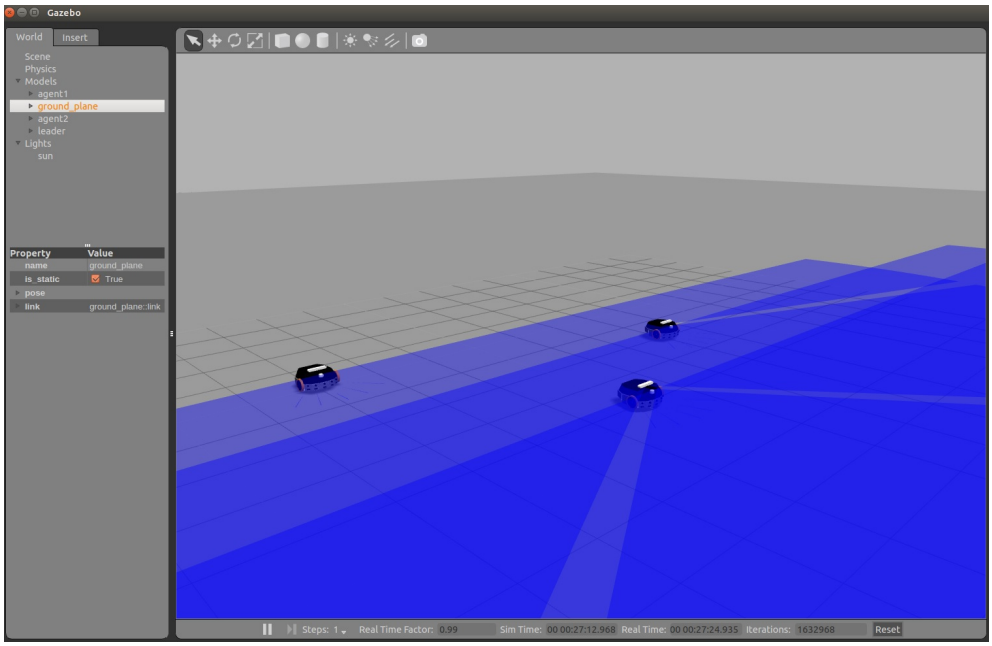


Figure 4.5: Gazebo simulation screen-shot.

4.5 Multi-robot ROS Network

Typically, in ROS framework, each Mini-Lab robot has its own ROS network with a single ROS master which handles all the communication between ROS-nodes in the network. Multiple ROS-nodes can either run on the same computer or they can be on different computers depending on the application requirements. The network could be wired, wireless or a combination of both. Figure 4.6 illustrates an example of a single robot network.

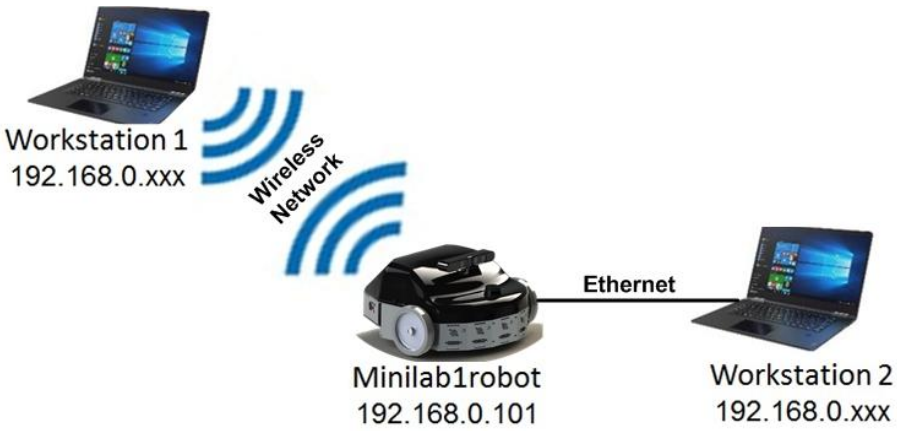


Figure 4.6: Single robot ROS network

When multiple robots are used in a system, there are two possible ROS network schemes. One is to establish a large network with a single ROS master handling all the ROS-nodes and topics. However, in such a case, all robots are dependant on a single `roscore` node. The `roscore` not only manages the communication between the ROS-nodes of different robots but also handles communication of ROS-nodes within a single robot. The other possibility for a multi-robot system is to develop such a mechanism where multiple ROS systems can exchange information with each other. In other words, each robot has a local ROS network with its own ROS master and each local network can communicate to the other ROS networks of other robots. This kind of setup is allowed under the ROS multi-master scheme in which two or more ROS networks can be combined. Such a scheme is essential for distributed communication topology.

A multi-master ROS network can be created through a special package called `multimaster_fkie`.

It consists of various ROS-nodes to establish and maintain the communication between ROS subsystems. It can automatically detect and synchronize the changes in the network. `multimaster_fkie` has two essential ROS-nodes called `master_discovery` and `master_sync` that must run simultaneously. `master_discovery` notifies its presence to local ROS networks by periodically sending the multicast messages. It is also responsible to detect any changes in the local networks and inform all ROS masters about these changes. On the other hand, `master_sync` allows the synchronization of remote services and topics to the local ROS master. Each remote ROS network should have its own `master_sync` ROS-node to achieve synchronization with other ROS networks. Following section discusses the experimental setup which is based on ROS multi-master.

4.6 Experimental setup

A fleet of three Mini-Lab robots is used for the experimental verification of our proposed algorithms. Since each robot has its own ROS master, a distributed ROS network is created for the robots to communicate using a ROS multi-master setup. Each of the robots and the workstation requires a unique and dedicated IP address such that information is received from the same address every time they connect to the network. The address reservation is configured in the router against the MAC addresses of the robots such that any robot gets same IP address when it becomes alive in the network. Host names are bounded with the IP addresses in the */etc/hosts* file. The network configuration can be verified by pinging each robot and the workstation. Following IP addresses have been assigned to to each machine in the network.

Minilab1robot	192.168.0.101
Minilab2robot	192.168.0.102
Minilab3robot	192.168.0.103
Workstation	192.168.0.111

Table 4.2: Assigned IP addresses.

Once the wireless network is established, multi-master ROS setup is configured using `multimaster_fkie`. First of all, local `roscore` is run on each robot. After that, `mutimaster_discovery` node is launched on each robot by the following command:

```
$ rosrun master_discovery_fkie master_discovery_mcast_group:=224.0.0.1
```

The parameter `mcast_group` is passed to specify the multicast address to be used. Afterwards, topics and services of all robots and workstation are synchronized by launching `master_sync` ROS-node. The list of all available ROS maters can be obtained using the following command:

```
$ rosservice call /master_discovery/list_masters
```

The time at which a robot measures its position is highly important in the proposed algorithms. Therefore, it is necessary that the clocks of all the machines on the network are synchronized. This has been achieved by incorporating Network Time Protocol (NTP) in the network. Figure 4.7 illustrates the multi-master ROS network with three Mini-Lab robots and a workstation each having a local ROS master.

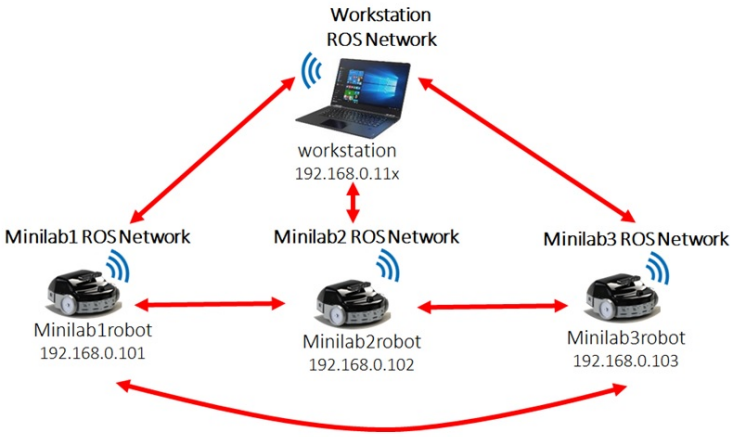


Figure 4.7: Multi-master ROS network with Mini-Lab robots.

Leader-following communication configuration

For the application of the leader-following control algorithms, the Minilab3 robot is considered as a leader while Minilab1 and Minilab2 are considered as follower 1 and follower 2 respectively. Though, in the designed multi-master ROS network, each robot can receive information of other robots, the distributed communication topology is emulated by simply restricting the use of information from certain members of the network. For instance, consider the communication topology shown in Figure 4.8. Node 0, representing the leader, does not have any subscriber ROS-node. It is only publishing its po-

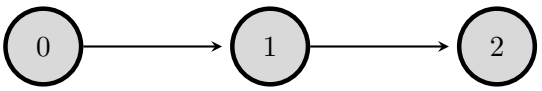


Figure 4.8: Communication topology.

sition data on the `minilab3/odom` topic. Follower 1 has a subscriber ROS-node receiving data from `minilab3/odom` topic and also has a publisher with topic `minilab1/odom`. Similarly, follower 2 is subscribed to `minilab1/odom` only. The distributed communication topology achieved by designing appropriate subscriber/publisher ROS-nodes is illustrated in Figure 4.9 while Figure 4.10 shows the actual experimental setup.

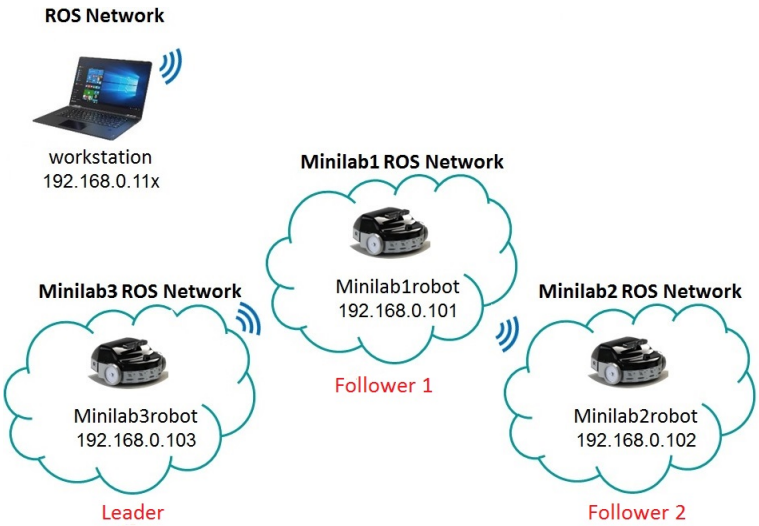


Figure 4.9: Distributed communication configuration.

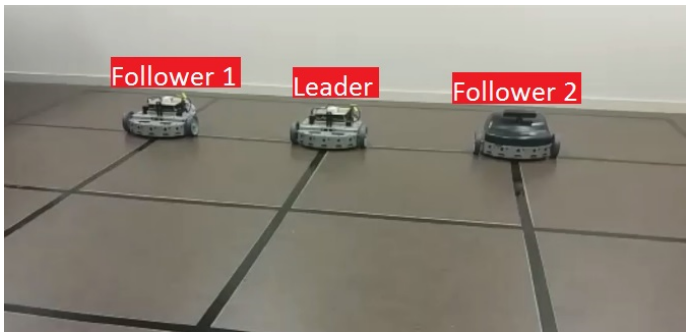


Figure 4.10: Experimental setup.

4.7 Consensus tracking

The leader-following consensus algorithm is applied on the robotic fleet to achieve consensus in the x-position state of the robots. For this purpose, the motion of robots is restricted in the x-axis only with some fixed offset in their y-positions as shown in Figure 4.11. In fact, consensus schemes are not applicable for robotic agents if both x and y positions are considered since to achieve consensus, robots will converge to the same position in the plane and will eventually collide. By restraining the robot motion to 1-D, one not only avoids collision but additionally the nonholonomic constrains can be ignored since robot dynamics can be reduced to a double integrator given below:

$$\begin{cases} \dot{r}_i(t) = v_i(t), \\ \dot{v}_i(t) = u_i(t) \end{cases} \tag{4.1}$$

where r_i, v_i and u_i represent respectively the position, velocity and control input of the i-th robot. The consensus tracking is achieved if the followers track x-position of the leader.

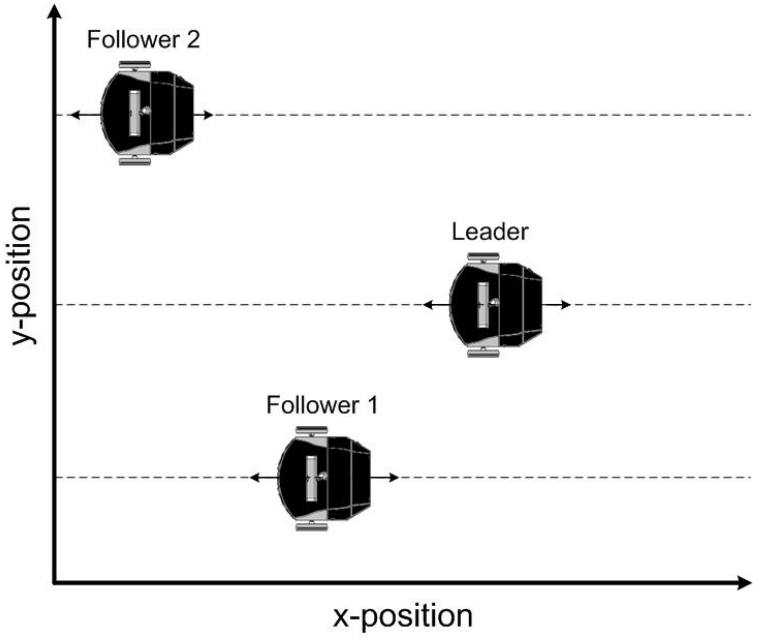


Figure 4.11: Robot motion is restricted to x-axis

The implementation algorithm for consensus tracking is given in Algorithm 1. Position and velocity of the robot and its neighbor is estimated using the most recent available data through the observer (2.18)-(2.19). These estimated values are used to compute the input $u_i(t)$ through consensus control law given in (2.17). Then the required robot linear velocity $v_i(t)$ is calculated and implemented. The while loop will continue until either the desired final time is achieved or `ROS_OK` returns false. `ROS_OK`

could return false if an interrupt signal SIGINT (Ctrl+C) is received, ROS-node handler is destroyed or a ROS-node is kicked off from the network due to any reason.

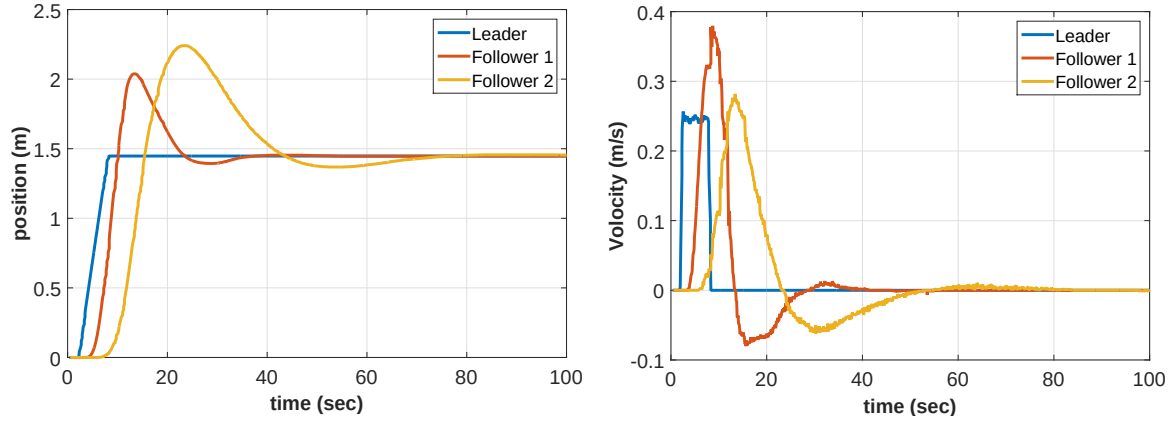
Algorithm 1: Leader-following consensus algorithm.

For each agent $i = 1, 2 \dots N$ **Input:** $r_i(t_k^{i,i})$, $r_0(t_k^{i,0})$ (if $b_i = 1$) and $r_j(t_k^{i,j})$ (if $a_{ij} = 1$)
Parameter: γ, θ and $\bar{c}$
Output: $v_i(t)$
initialize variables for agent i and its neighbor positions and respective sampling time;
while *time<final_time and ROS_OK* **do**
  Measure and broadcast own position;
  Receive position data of neighbors;
  if *new position sample available* **then**
    use new position and its sampling time;
  else
    use previously measured/received data;
  end
  compute the estimation of position and velocity of agent i and its neighbors using observer (2.18)-(2.19);
  compute $u_i(t)$ using controller (2.17);
  implement $v_i(t)$
end

Hardware limitations are kept in mind while choosing the controller and the observer gains such that the maximum speed limit is not exceeded. The communication topology among the robots is the same as shown in Figure 4.8. The related adjacency and pinning matrices are:

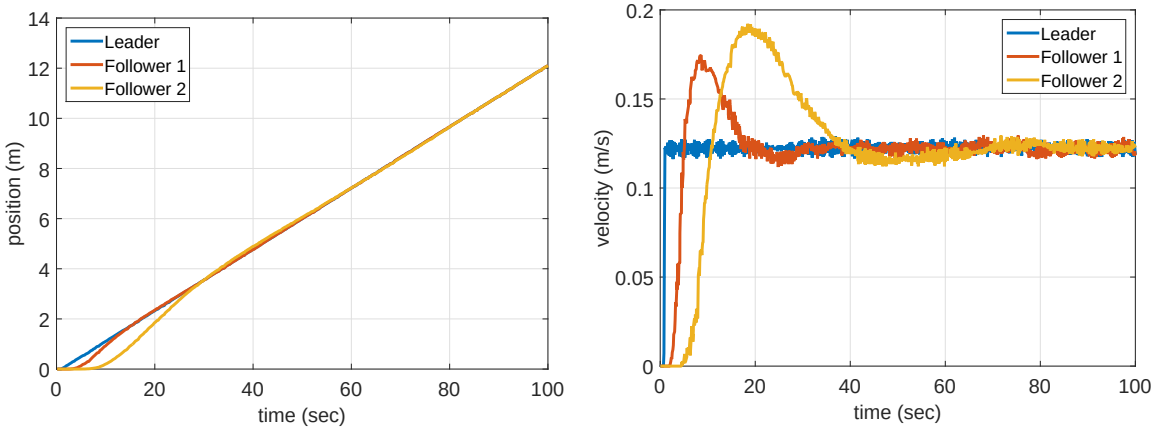
$$\mathcal{A} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \qquad \mathcal{B} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

The experiments are carried out for both step and ramp position trajectories of the leader. Figure 4.12 shows the position and velocity consensus results when the leader has step position trajectory while the Figure 4.13 depicts the position and velocity consensus results with ramp position trajectory of the leader. An example of sampling time for communication of data from follower 1 to follower 2 is given in Figure 4.14. It is clear from the results that the proposed observer based distributed consensus algorithm performs efficiently despite the presence of uncertainties and disturbances inherent for real applications.



(a) (b)

Figure 4.12: Leader-following consensus with step leader trajectory (a) position (b) velocity.



(a) (b)

Figure 4.13: Leader-following consensus with ramp leader trajectory (a) position (b) velocity.

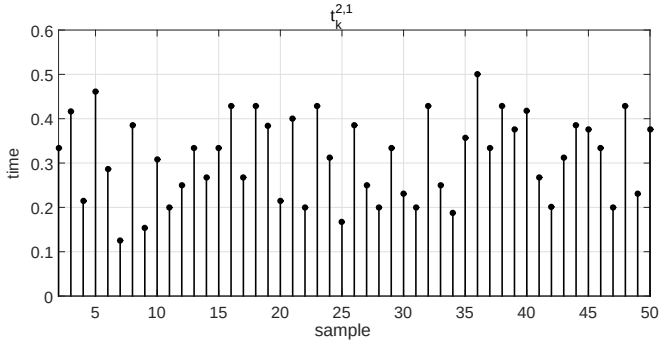


Figure 4.14: Sampling period for data transmission from follower 1 to follower 2.

4.8 Nonholonomic robot model

A differential drive mobile robot like Mini-Lab is kinematically equivalent to a unicyle vehicle. The configuration of such robots can be described by a generalized coordinates vector $q(t)$ of its centre of mass. Let us consider that $\mathcal{M}_i$ is the i^{th} robot in a multi-robot network. Its configuration vector can be expressed as:

$$q_i(t) = [r_{x_i}, r_{y_i}, \phi_i]^T$$

where r_{x_i} and r_{y_i} are the x-position and y-position of the center of mass of the robot in the Cartesian coordinates while ϕ_i is the heading angle or orientation of the robot as explained in Figure 4.15. These kinds of robots are subject to kinematic constraints due to the pure rolling and nonslipping conditions of the wheel. It means that a robot can move forward or backward in curved motion but cannot move sideways. This can be defined as [181]:

$$\dot{r}_{x_i} \sin\phi_i - \dot{r}_{y_i} \cos\phi_i = 0 \tag{4.2}$$

or in pfaffian form as:

$$[\sin\phi_i, -\cos\phi_i, 0]\dot{q}_i(t) = 0 \tag{4.3}$$

This constraint is called a nonholonomic constraint since it only restricts the motion of a robot but does not reduce the accessibility of the configuration space. With the above mentioned nonholonomic constraint, the unicycle robot model can be described as:

$$\dot{q}_i(t) = \begin{bmatrix} \dot{r}_{x_i} \\ \dot{r}_{y_i} \\ \dot{\phi}_i \end{bmatrix} = \begin{bmatrix} \cos\phi_i & 0 \\ \sin\phi_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{4.4}$$

where v_i and ω_i are the linear and angular velocities of the robot.

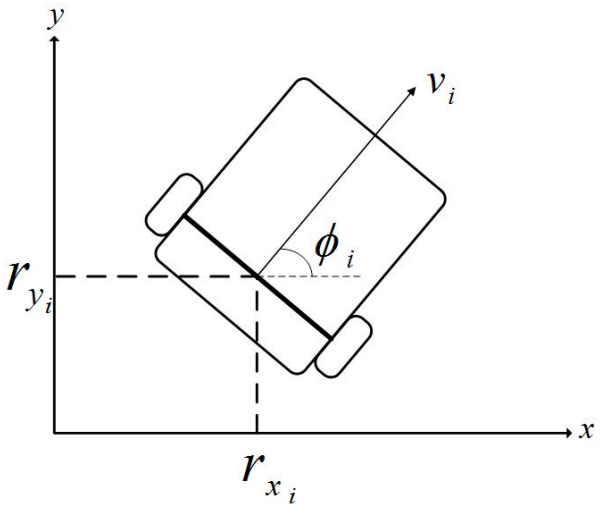


Figure 4.15: Unicycle type robot model.

4.9 Control scheme for nonholonomic robot

In order to apply the proposed formation tracking and collision avoidance algorithms on a fleet of nonholonomic mobile robots, the differential flatness properties [182, 183] have been used. Cartesian coordinates of the center of mass of a differential drive robot are flat outputs. Therefore, in case of trajectory tracking, the desired trajectory to be followed can be provided in the Cartesian coordinates as $[r_{x_d}, r_{y_d}]$. The associated configuration vector is $q_d(t) = [r_{x_d}, r_{y_d}, \phi_d]$ where ϕ_d can be computed as:

$$\phi_d = atan2(\dot{r}_{x_d}, \dot{r}_{y_d}) \tag{4.5}$$

where $atan2$ not only computes arctangent but also determines the quadrant of the result. Similarly, the required input linear velocity for the robot can also be calculated as:

$$v_d = \pm\sqrt{\dot{r}_{x_d}^2 + \dot{r}_{y_d}^2} \quad \text{or} \quad v_d = \dot{r}_{x_d}\cos\phi_d + \dot{r}_{y_d}\sin\phi_d \tag{4.6}$$

Flatness-based approach is valid for formation tracking goals since the required task is eventually to track the desired position in the Cartesian coordinates. Figure 4.16 illustrates the control scheme to apply the developed formation tracking and collision avoidance algorithms on a robot. The tracking control scheme of the robot is divided into an inner-loop and an outer-loop. The outer-loop can be modelled with double integrator dynamics since x and y coordinates are considered separately. Hence, the control algorithm for double integrator agents can be applied in this loop.

The designed distribution formation tracking and collision avoidance algorithms are implemented in the `'Formation tracking law'` block which will compute the desired acceleration u_x and u_y for each robot. The required heading angle ϕ_d and linear velocity v are computed using (4.5) and (4.6). Moreover, the estimated angle $\hat{\phi}$ of the robot can also be calculated through the estimated velocities $\hat{v}_x$ and $\hat{v}_y$. This is done in the inner-loop block `'robot angle'`. The inner-loop contains a PID controller which computes the required angular velocity ω using the error signal $e_\phi = \phi_d - \hat{\phi}$. The error is wrapped between $-\pi$ and π by using $e_\phi = atan2(\sin(e_\phi), \cos(e_\phi))$. The computed angular velocity ω steers the robot in the desired orientation as dictated by the formation tracking control law.

Remark 44. *The above mentioned control scheme is a practical solution to apply second-order algorithms on nonholonomic robots. The proposed schemes is valid for all cases when the linear velocity of the robot is non-zero. There exists a singularity when the velocity becomes zero. However, this singularity can be dealt easily in practical application since the objective is to achieve the require position state to make formation and the final angle of the robot does not matter. Whenever the linear velocity becomes zero, we can considered the current angle equal to the angle computed at the last instant.*

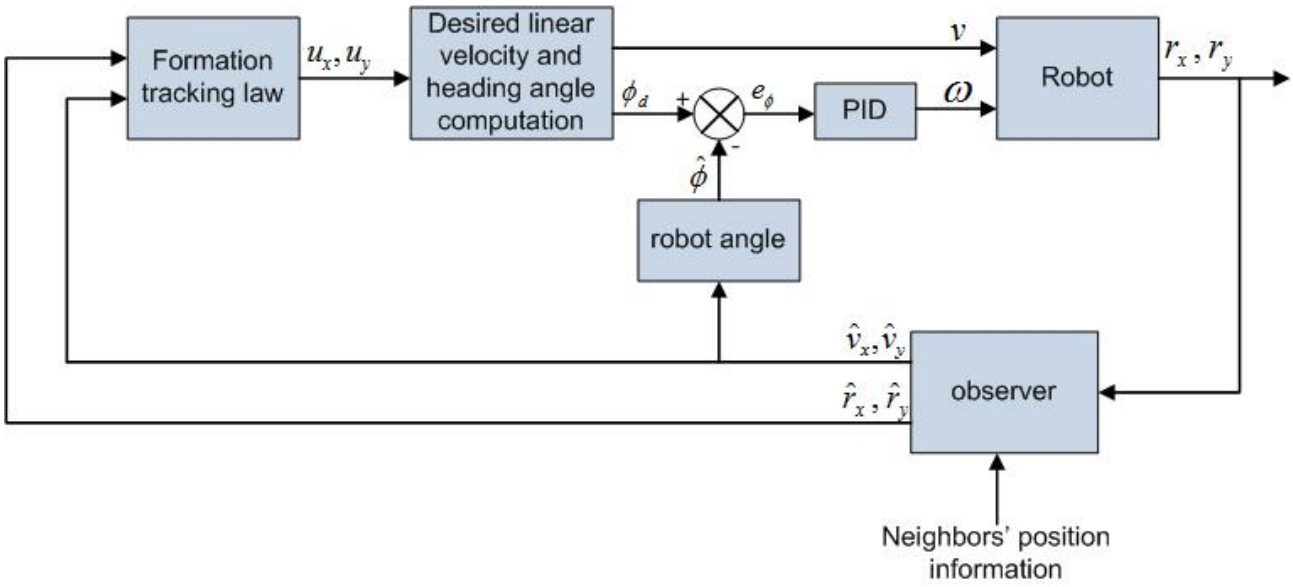


Figure 4.16: Control scheme for one robot in the nonholonomic multi-robot network.

4.10 Formation tracking control

The proposed distributed formation tracking algorithm has been applied to a multi-robot network using the the control scheme described in the previous section. The numerical results have been obtained through ROS gazebo simulator. The algorithm for the formation tracking of robots is given in Algorithm 2. The position and velocity of the robot and its neighbors are estimated using the latest available position data by implementing the observer described by (3.5)-(3.6). After calculating u_{x_i} and u_{y_i} using the formation tracking controller (4.10), linear and angular velocities are computed and implemented.

A fleet of four Mini-Lab robots is considered for testing and verification purposes. One robot is acting as a leader while the other three are considered the followers. Figure 4.17 shows the communication topology among the robots. The corresponding adjacency and pinning matrices are:

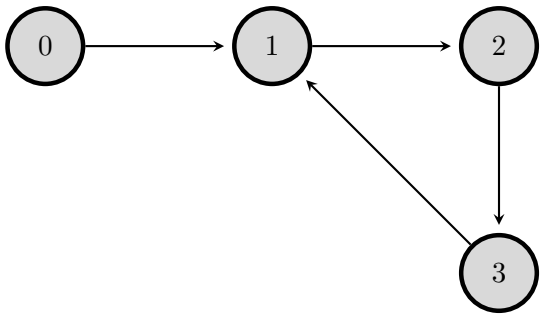


Figure 4.17: Communication topology for formation tracking.

$$\mathcal{A} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad \mathcal{B} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The initial conditions of the agents are chosen carefully such that they do not collide with each other while converging to the desired position. The initial state of the leader and three followers are $r_0(0) = (0, 0)$, $r_1(0) = (-3, 3)$, $r_2(0) = (-5, 0)$ and $r_3(0) = (3, -3.5)$ with initial angles as $\phi_0(0) = 0^o$, $\phi_1(0) = 45^o$, $\phi_2(0) = -45^o$ and $\phi_3(0) = 0^o$ respectively.

The sampling period of the robots in Gazebo is not regular or synchronous. However, to test the designed algorithm with more random sampling, separate ROS-nodes are written which sets the

Algorithm 2: Formation tracking algorithm.

For each agent $i = 1, 2 \dots N$ **Input:** $r_i(t_k^{i,i})$, $r_0(t_k^{i,0})$ (if $b_i = 1$) and $r_j(t_k^{i,j})$ (if $a_{ij} = 1$),
Parameter: γ, θ and $\bar{c}$
Output: $v_i(t)$
initialize variables for agent i and its neighbor positions and respective sampling time;
initialize variable for formation vector;
while *While time<final_time and ROS_OK* **do**
  Measure and broadcast own position;
  Receive position data of neighbors;
  if *new position sample available* **then**
    update position and sampling time value;
  else
    keep previously measured/received data;
  end
  compute formation vector;
  compute the estimation of position and velocity of agent i and its neighbors using observer (3.5)-(3.6);
  compute $u_{x_i}(t)$ and $u_{y_i}(t)$ using controller (4.10);
  compute $v_{x_i}(t)$ and $v_{y_i}(t)$ from obtained $u_{x_i}(t)$ and $u_{y_i}(t)$;
  compute the estimated angle;
  compute linear velocity $v_i(t)$ and angular velocity $\omega_i(t)$;
  implement $v_i(t)$ and $\omega_i(t)$
end

sampling rate to be more random. These ROS-nodes receive odom messages from the robot `\odom` topics and then publish only random position information with original sampling time on new topics, we named it `\data`. The robots subscribe to this new topic of the corresponding robot to receive position information. Since each odom message is time-stamped, the information of sampling time can be extracted from the message header. Figure 4.18 illustrates an example of sampling of instants of data communication between the robots.

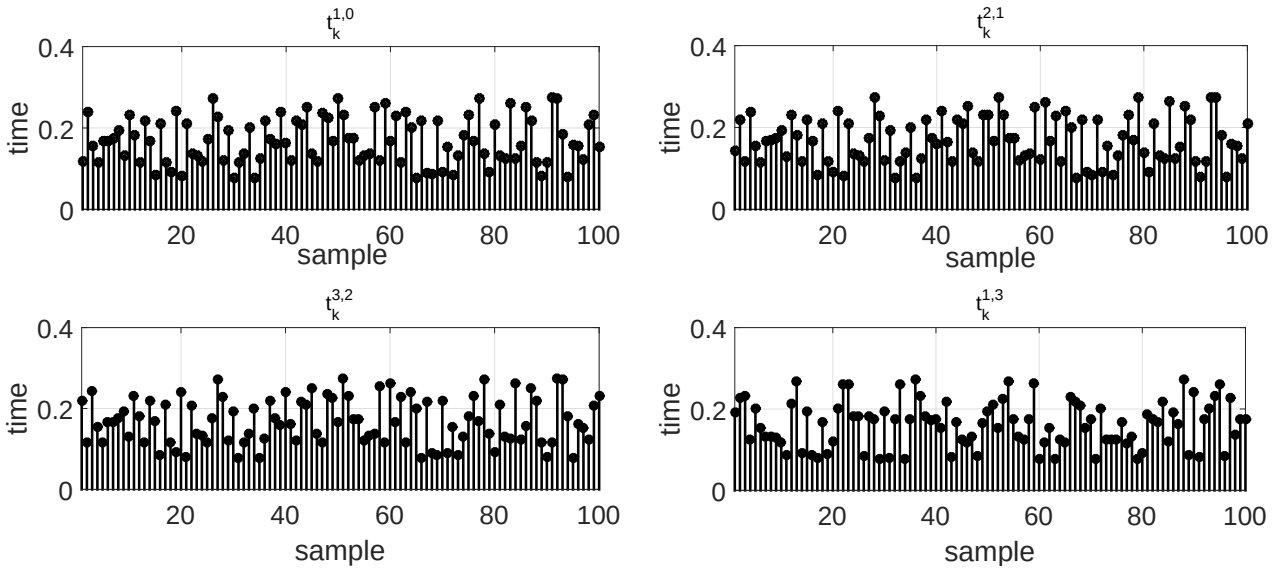


Figure 4.18: Sampling time between the agents.

The observer and controller gains are chosen as $\bar{c} = 1$, $\theta = 5$ and $\lambda = 0.4$ respectively while the proportional, integral and derivative gains of PID controller in the inner-loop are 1.6, 0.3, 0.6 respectively. The simulations are carried out for both fixed and time-varying formations.

4.10.1 Fixed-formation

In this formation, the followers are required to make a fixed triangular shape around the leader and follow its trajectory while maintaining the shape. The formation vectors for the followers are selected as:

$$\begin{aligned} f_1 &= [0, 3, 0, 0]^T \\ f_2 &= [-3, -3, 0, 0]^T \\ f_3 &= [3, -3, 0, 0]^T \end{aligned}$$

In the first scenario, the leader is kept static at position coordinates $(0, 0)$ and follower robots reach and maintain the desired formation around it. Figure 4.19 illustrates the formation tracking at different time instants for this scenario while Figure 4.20 shows the corresponding tracking errors of both x and y positions.

In the second scenario, the leader robot moves with a linear velocity $v_0 = 0.15m/s$ and heading angle $\phi_0 = 20^o$. Figure 4.21 and Figure 4.22 show the formation tracking results and tracking errors respectively. It can be clearly seen that followers make the triangular shape and maintain it while tracking the leader trajectory.

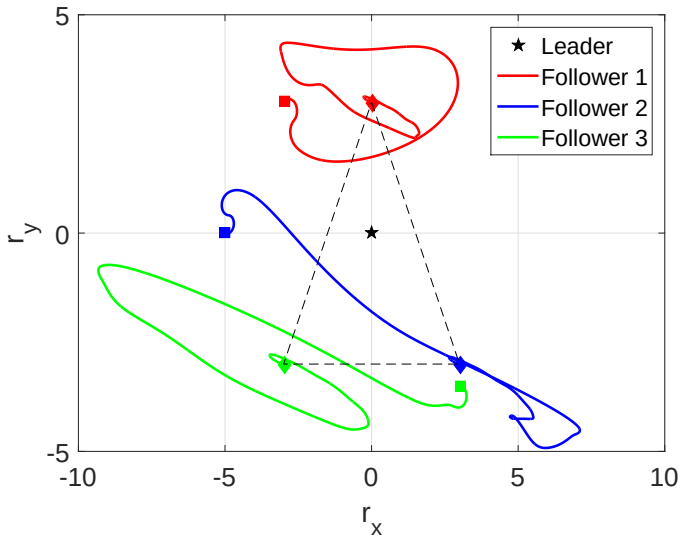


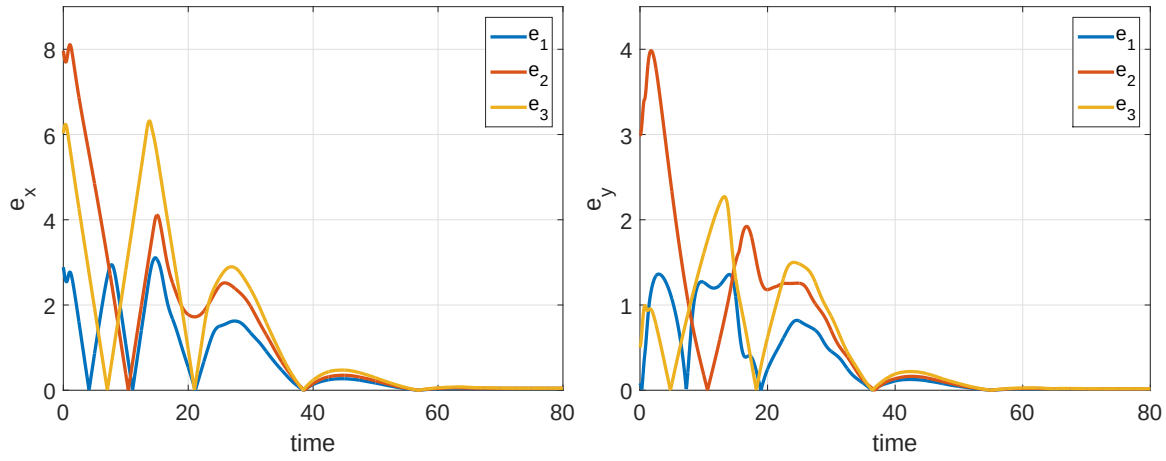
Figure 4.19: Fixed formation tracking with a stationary leader.

4.10.2 Time-varying formation

For the time-varying formation tracking, the formation vector is chosen as below:

$$f_i(t) = \begin{pmatrix} 3\cos(0.1t + 2\pi(i-1)/3) \\ 3\sin(0.1t + 2\pi(i-1)/3) \\ -0.3\sin(0.1t + 2\pi(i-1)/3) \\ 0.3\cos(0.1t + 2\pi(i-1)/3) \end{pmatrix}$$

with $i = 1, 2, 3$. If this formation is achieved, the followers will move in a circle around the leader. The radius of the circle is $3m$. Two cases are again considered. Firstly, the leader is stationary and three followers develop the circular formation around it and keep on rotating in that circle. Figure 4.23 depicts the formation tracking at different time instants and Figure 4.24 shows the related tracking error. Figure 4.25 and Figure 4.26 show formation tracking and its corresponding tracking error for the second scenario where the leader is moving at an angle of 20^o with linear velocity of $0.15m/s$.



(a) (b)

Figure 4.20: Tracking error of fixed formation with a stationary leader.

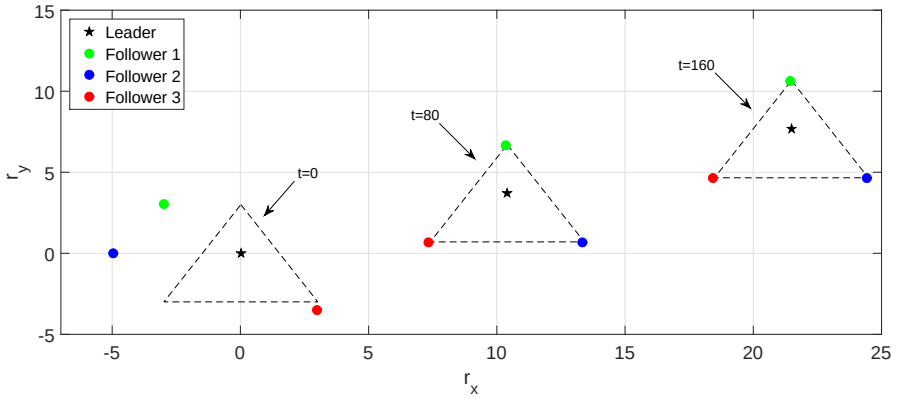
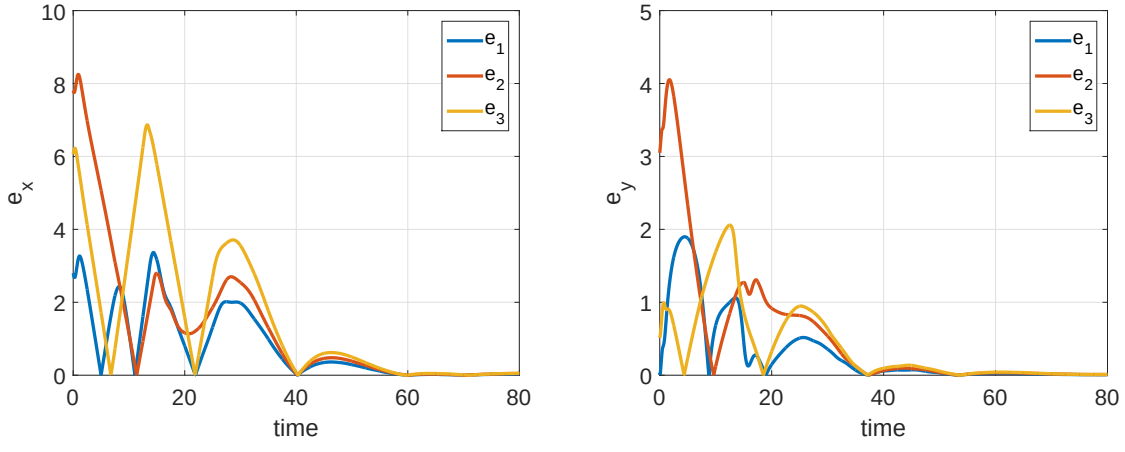


Figure 4.21: Fixed formation tracking with a moving leader.



(a) (b)

Figure 4.22: Tracking error for fixed formation with a moving leader.

4.11 Formation tracking control with collision avoidance

Before discussing the results of collision-free formation tracking of the robotic network, let us first discuss the concept of velocity cone.

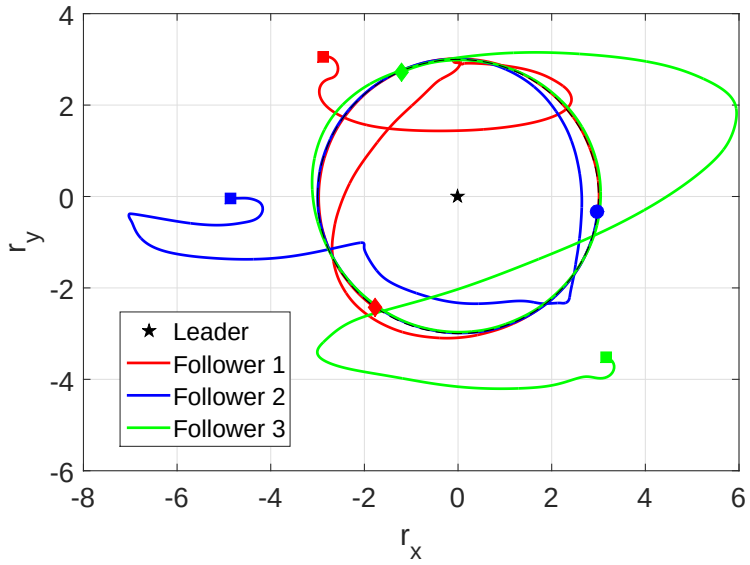
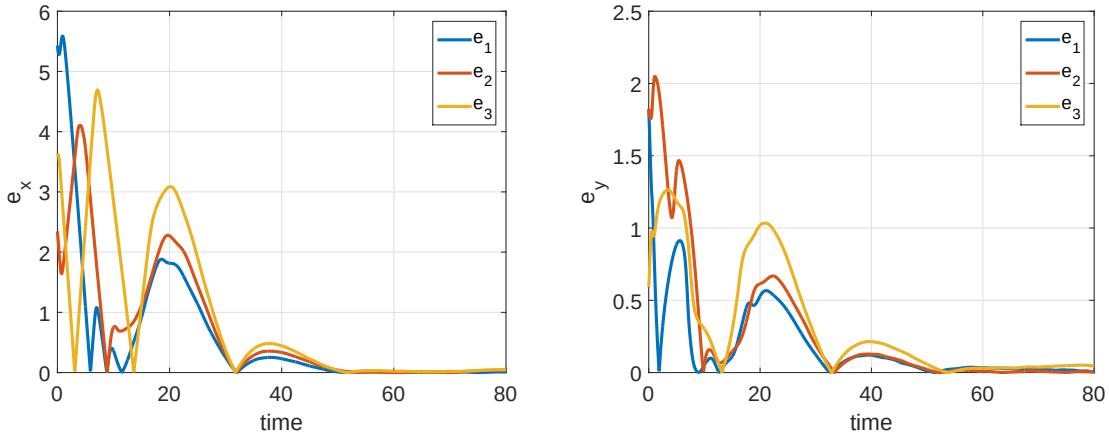


Figure 4.23: Time-varying formation tracking with a static leader.



(a) (b)

Figure 4.24: Tracking error for time-varying formation tracking with a static leader.

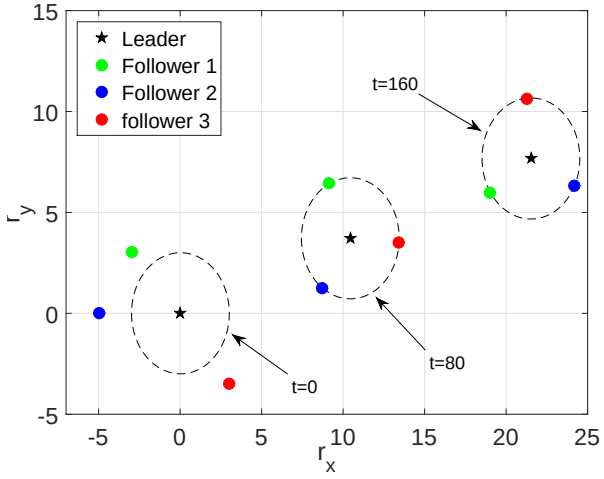
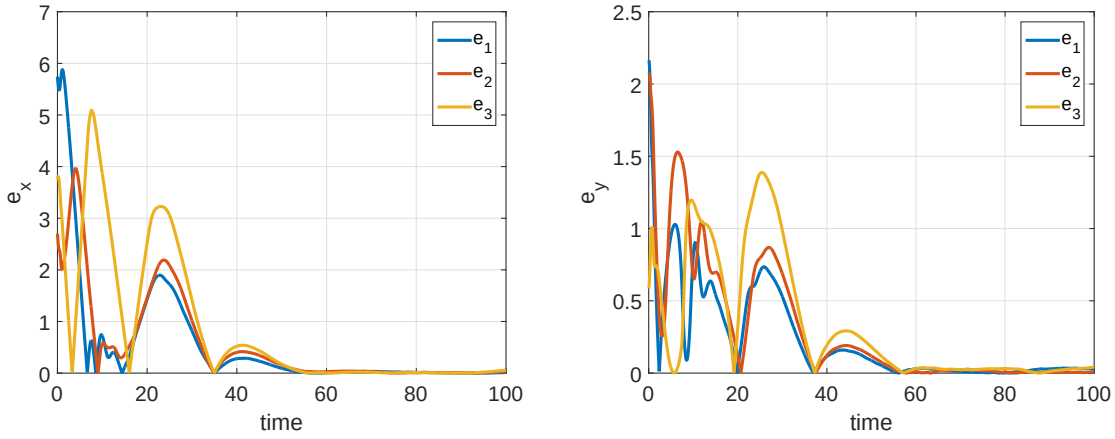


Figure 4.25: Time-varying formation tracking with a moving leader.



(a) (b)

Figure 4.26: Tracking error of time-varying formation tracking with a moving leader.

4.11.1 Velocity cone concept

The velocity cone concept provides a geometrical representation of velocities of dynamic objects which will eventually result in a collision [184, 185]. In our proposed framework, the unnecessary deviation of the robot from the actual path can be reduced by incorporating the velocity cone concept along with the Artificial Potential Function (APF) based collision avoidance algorithm. The repulsion force acts on the robot only when the velocity cone predicts a possible collision in the near future. The repulsion force keeps pushing the robot away while the robot velocity and heading angle could lead to a possible collision.

Let us consider that each robot has a protection region of radius r around it and $v_{ji} = v_j - v_i$ be the relative velocity of a pair of robots $(\mathcal{M}_i, \mathcal{M}_j)$. This relative velocity is obtained by taking the derivative of the relative position obtained by proximity sensor.

Let us define the following variables as shown in Fig. 4.27:

$$\beta_{ij} = \arg(v_j - v_i) - \arg(r_i - r_j) \tag{4.7}$$

$$\alpha_{ij} = \arcsin\left(\frac{2r}{\|r_{ij}\|}\right) \tag{4.8}$$

At a given time, a necessary and sufficient condition for no collision between robots $\mathcal{M}_i$ and $\mathcal{M}_j$ is

$$|\beta_{ij}| > \alpha_{ij} \tag{4.9}$$

4.11.2 Numerical results

The designed collision-free formation tracking algorithm has been applied to a multi-robot network consisting of a leader and four followers. The velocity cone is incorporated with the algorithm to avoid unnecessary deviation of the robots from there required path. The repulsion force only activates when the velocity cone detects any possible collision. Control law (3.15) along with the velocity cone is implemented `'Formation tracking law'` block in the outer-loop of the control scheme described in Section 4.9. The algorithm for the implementation is shown in Algorithm 3.

The communication topology among the robots is shown in Figure 4.28. The leader can send its position information to follower 1 only while follower 4 receives data from both follower 2 and follower 3.

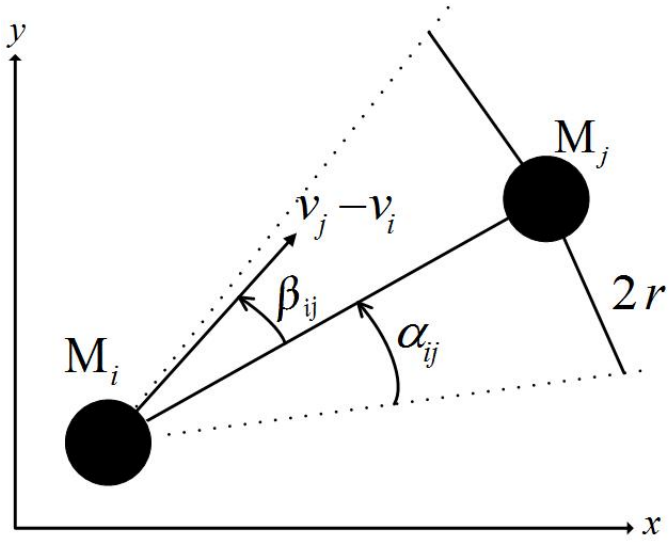


Figure 4.27: Velocity cone

Algorithm 3: Collision-free formation tracking algorithm.

For each agent $i = 1, 2 \dots N$ **Input:** $r_i(t_k^{i,i})$, $r_0(t_k^{i,0})$ (if $b_i = 1$) and $r_j(t_k^{i,j})$ (if $a_{ij} = 1$),
Parameter: γ, θ and $\bar{c}$
Output: $v_i(t)$
initialize variables for agent i and its neighbor positions and respective sampling time;
initialize variables for formation vector and velocity cone;
while *While time<final_time and ROS_OK* **do**
  Measure and broadcast own position;
  Receive position data of neighbors;
  if *new position sample available* **then**
    update the value of position ans sampling time;
  else
    keep previously measured/received data;
  end
  compute formation vector;
  compute the estimation of position and velocity of agent i and its neighbors using observer (3.5)-(2.19);
  compute $u_{x_i}^f(t)$ and $u_{y_i}^f(t)$ using controller (3.7);
  if *any other agent detected in side radius R* **then**
    compute relative velocity v_{ji} and angles β_{ij} and α_{ij} ;
    if $|\beta_{ij}| > \alpha_{ij}$ **then**
      compute $u_{x_i}^r(t)$ and $u_{y_i}^r(t)$ using controller (3.14);
    end
  end
  compute $u_{x_i}(t)$ and $u_{y_i}(t)$ using controller (3.15);
  compute $v_{x_i}(t)$ and $v_{y_i}(t)$ from obtained $u_{x_i}(t)$ and $u_{y_i}(t)$ and compute the estimated angle;
  compute linear velocity $v_i(t)$;
  compute angular velocity $\omega_i(t)$;
  implement $v_i(t)$ and $\omega_i(t)$
end

The matrices related to this communication graph are:

$$\mathcal{A} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \qquad \mathcal{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

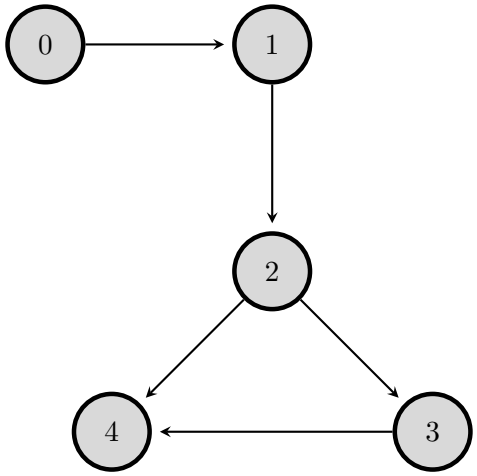


Figure 4.28: Communication topology for collision-free formation tracking.

The simulations are carried out in Gazebo. The initial position of the robots are $r_0(0) = (0, 0)$, $r_1(0) = (1, -5)$, $r_2(0) = (-5, 0)$, $r_3(0) = (3, 5)$ and $r_4(0) = (-5, 6)$ while the initial angles are $\phi_0(0) = 0$, $\phi_1(0) = 40^o$, $\phi_2(0) = -45^o$, $\phi_3(0) = 15^o$ and $\phi_4(0) = -25^o$. The observer and controller gains are chosen as $\bar{c} = 1$, $\theta = 5$ and $\lambda = 0.35$ while the PID controller gains are 1.6, 0.3 and 0.6 for proportional, integral and derivative terms respectively. The detection radius $R = 3m$ and the protection radius $r = 0.5m$. The results are obtained for both fixed and time-varying formations.

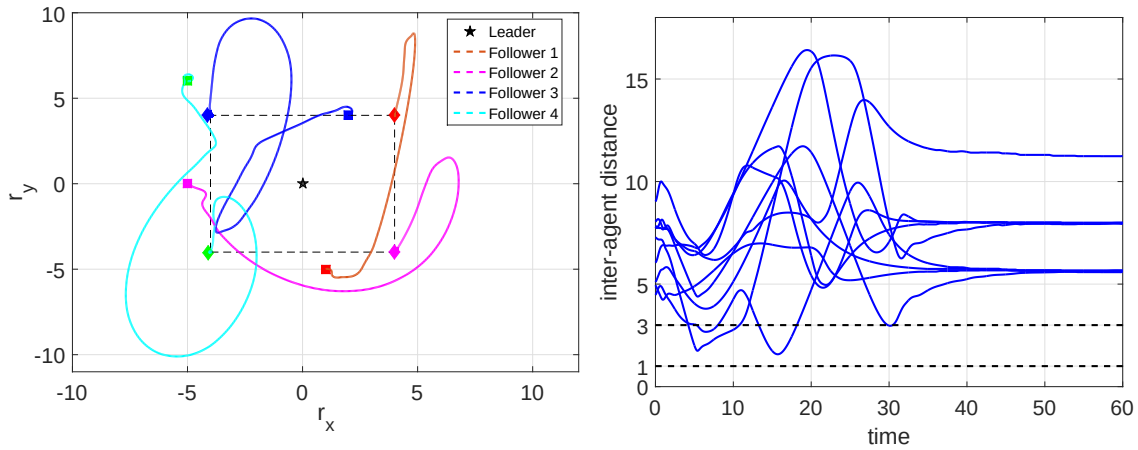
4.11.2.1 Fixed formation

The formation vector for fixed formation is chosen such that the followers make a square shape around the leader. The formation vectors are given as:

$$\begin{aligned} f_1 &= [4, 4, 0, 0]^T \\ f_2 &= [4, -4, 0, 0]^T \\ f_3 &= [-4, 4, 0, 0]^T \\ f_4 &= [-4, -4, 0, 0]^T \end{aligned}$$

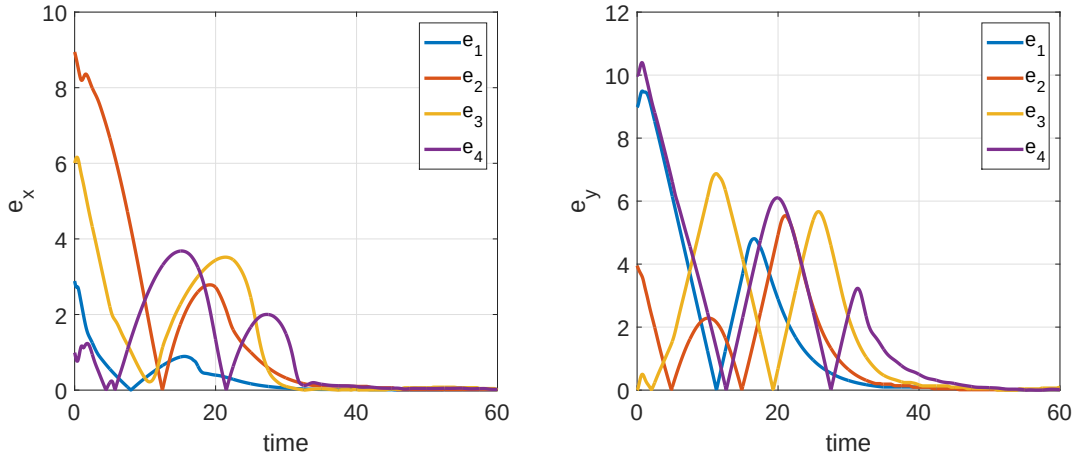
Figure 4.29a shows the formation tacking result when the leader is stationary with position coordinates $(0, 0)$ while Figure 4.29b shows the distance between the robots during the formation process. It is clear that the inter-agent distance remains great than $2r$ which means the robots do not collide. The tracking error results are shown in Figure 4.30.

Collision-free fixed formation tracking result with moving leader is shown in Figure 4.31. In this case, the leader is moving with linear velocity $v_0 = 0.15m/s$ with heading angle $\phi_0 = 30^o$. Figure 4.31 illustrates the formation tracking at different time instants and inter-agent distance is shown in Figure 4.31. The corresponding tracking errors are shown in Figure 4.32.



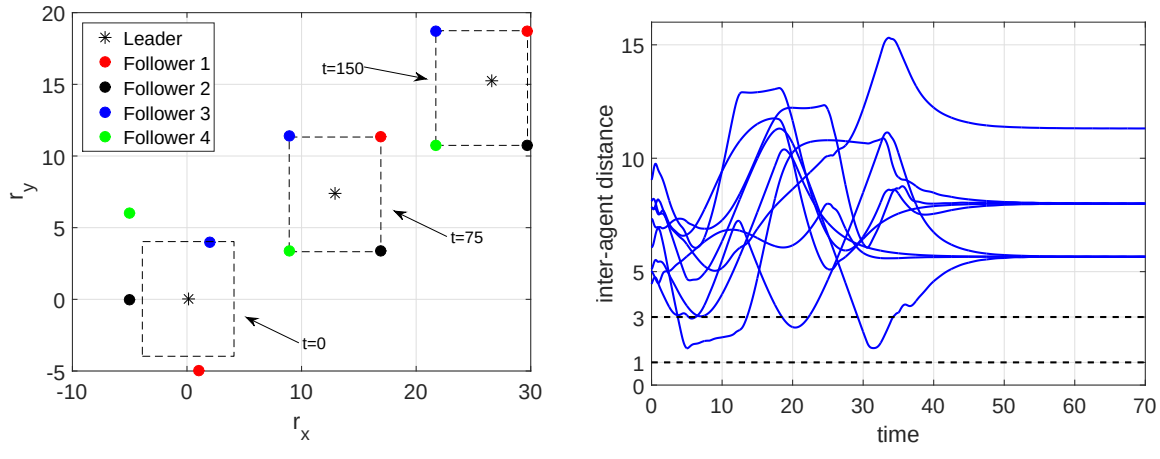
(a) (b)

Figure 4.29: Collision-free fixed formation with a stationary leader (a) formation tracking (b) inter-agent distance.



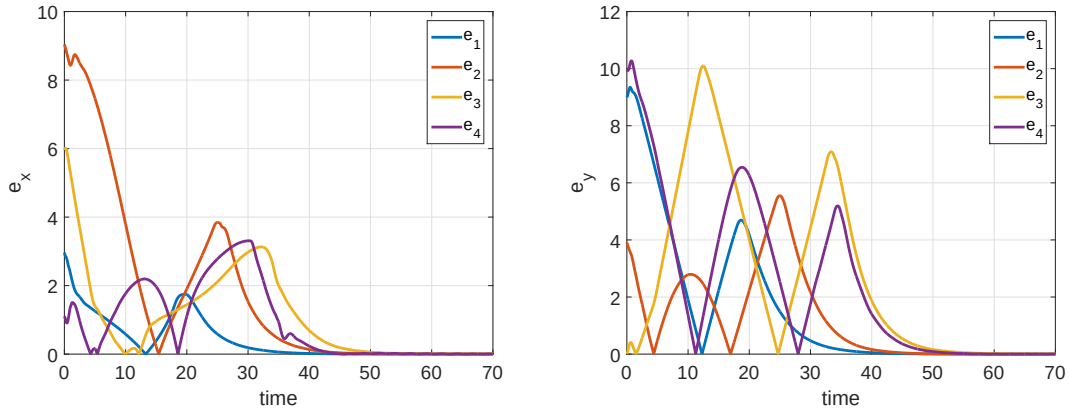
(a) (b)

Figure 4.30: Tracking error of square formation tracking with a stationary leader.



(a) (b)

Figure 4.31: Fixed formation with a moving leader (a) formation tracking (b) inter-agent distance.



(a) (b)

Figure 4.32: Tracking error of time-varying circular formation tracking with a moving leader.

4.11.2.2 Time-varying formation

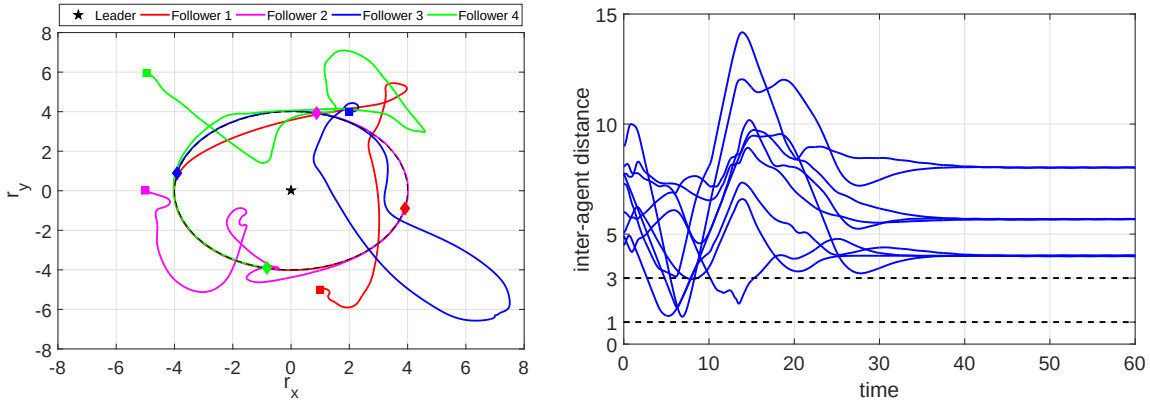
A circular shape is again chosen for time-varying collision-free formation tracking. The followers are supposed to move in a circle around the leader. The formation vector is chosen as:

$$f_i(t) = \begin{pmatrix} 4\cos(0.1t + 2\pi(i-1)/4) \\ 4\sin(0.1t + 2\pi(i-1)/4) \\ -0.4\sin(0.1t + 2\pi(i-1)/4) \\ 0.4\cos(0.1t + 2\pi(i-1)/4) \end{pmatrix}$$

for $i = 1, 2, 3, 4$. The formation vector is selected such that the distance between the robots, while in the formation, is greater than R.

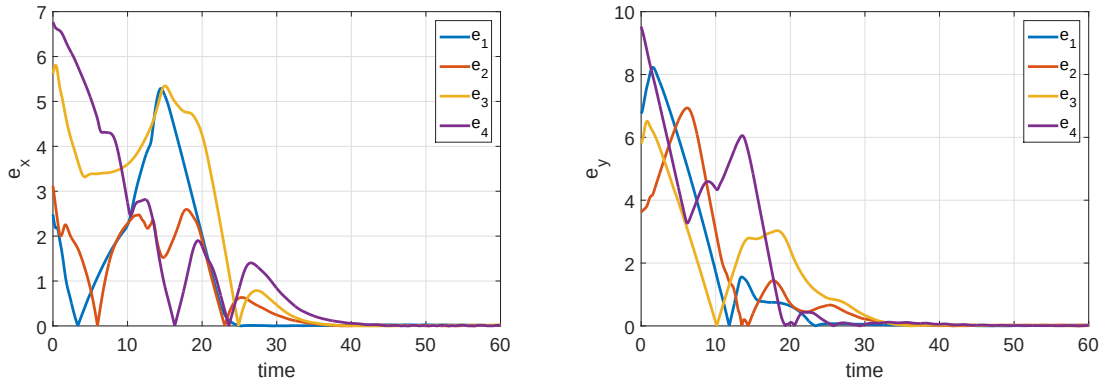
Both the cases with stationary and moving leader are verified. In the first case, the leader is stationary at position coordinates (0,0). Figure 4.33a shows collision-free formation tracking with a stationary leader and Figure 4.33b illustrates distance between the robots. It can be seen that the robots achieve the desired formation without any collision. Figure 4.34 depicts the tracking error of both x and y positions.

In the second case, the leader is moving at an angle of 30^o with a speed of $0.15m/s$. The formation tracking result is illustrated in Figure 4.35a and the distance between the robots is shown in Figure 4.35b. The corresponding tracking errors is shown in Figure 4.36.



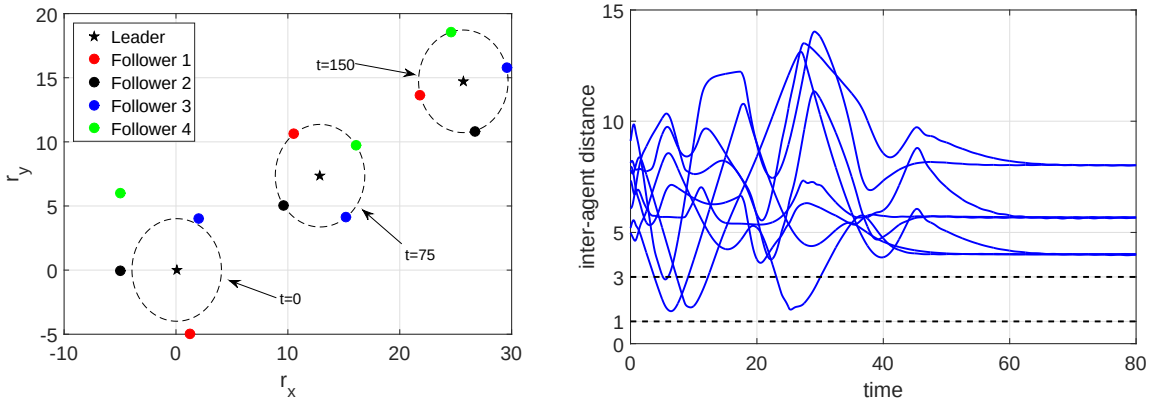
(a) (b)

Figure 4.33: Time-varying collision-free formation tracking with a static leader.



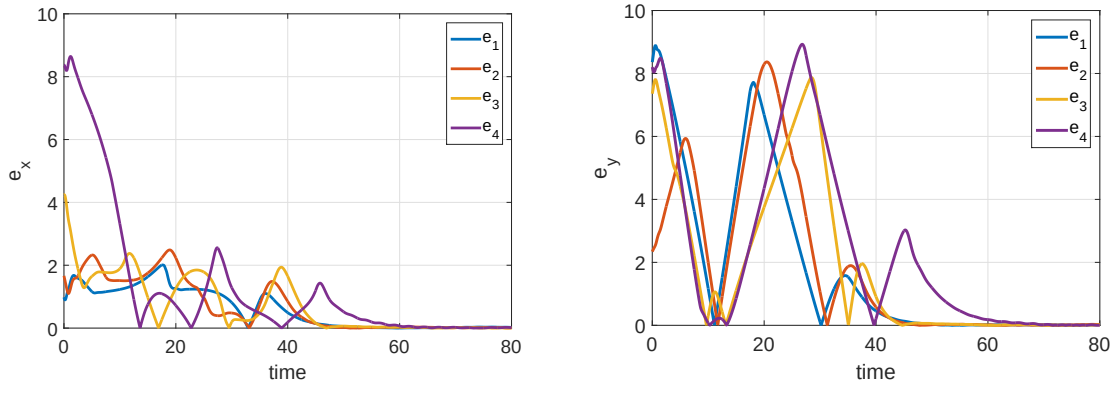
(a) (b)

Figure 4.34: Tracking error of time-varying circular formation tracking with a stationary leader.



(a) (b)

Figure 4.35: Time-varying collision-free formation with a moving leader (a) formation tracking (b) inter-agent distance.



(a) (b)

Figure 4.36: Tracking error of time-varying circular formation tracking with a moving leader.

4.12 Conclusion

In this chapter, application of the developed distributed cooperative controllers has been investigated. Leader-following consensus and formation tracking algorithms along with a collision avoidance scheme

have been applied to a robotic network consisting of differential drive mobile robots. The software architecture of the robots is based on an open-source platform ROS. The distributed multi-robot network is created using multi-master ROS strategy.

The observer based leader-following algorithm is implemented on a network of real robots. The experimental results showed that the consensus is achieved efficiently even in the existence of disturbances and uncertainties attached with real-world environment.

A new control scheme has been proposed to apply formation tracking and collision avoidance algorithms for nonholonomic agents. The proposed control scheme deals with the nonholonomic constraints of the robots and allows us to use the algorithms for double integrator on differential drive wheeled robots. The numerical results have been achieved through ROS Gazebo simulator. Gazebo provides a very realistic environment to test the algorithms on robots by incorporating real-world scenarios like friction, gravity etc. The proposed formation tracking and collision avoidance algorithms are tested for both fixed and time-varying formation patterns. The obtained results depict the efficacy of the algorithms in real-world applications and prove the robustness of these algorithms against the uncertainties and disturbances which are always present in actual applications such as measurement noise, communication noise, friction etc.

Part III

General conclusion and future prospects

General conclusion

This thesis has been focused on cooperative control of MAS. The considered MAS has been subjected to various real-life communication constraints. Generally, the communication constraints appear in the real systems due to the nature of the communication equipment and limited resources like communication bandwidth and lack of relevant sensors etc. Since the communication among the agents for cooperative control is of utmost importance, the constraints associated with it are required to be handled in order to achieve the desired goals of MAS in an efficient manner.

We considered the following communication constraints. Firstly, it has been assumed that each agent in the network can only measure and transmit its position information. It cannot measure and transmit either its velocity and acceleration (input) states. Secondly, the sampling time for data communication is irregular and nonuniform. In addition, these sampling periods are asynchronous which means that sampling time of each agent is totally independent of others. Furthermore, it has also been considered that the leader sends its position information to only a few agents and the communication topology among the agents in the network is direct.

The main results of the thesis are summarized below:

- First chapter provided a brief introduction of multi-agent systems and their applications. Some basic concepts related to MAS and communication network approaches to control it have also been presented. We also discussed two fundamental problems in cooperative control of MAS which include consensus and formation control. A detailed literature review has been provided on these problems. Issues and challenges in distributed cooperative control have also been discussed in detail.

  The communication constraints present in practical MAS networks motivated this research work to investigate and develop distributed control algorithms which must be able to deal with these constraints effectively.

- In the second chapter, leader-following consensus problem has been considered. A continuous-discrete time observer has been proposed to estimate the position and the velocity in continuous time through available discrete position data. Each agent not only estimates its own states but also the states of its neighbors. The proposed distributed control law uses these estimated states to compute the required input of the agent. It has been shown through Lyapunov stability analysis that the MAS achieves practical stability with the proposed algorithm if the leader input (acceleration) is nonzero and exponential stability is achieved for the case when the leader is static or moving with some constant velocity. The results have been further expanded for the case when the communication topology among the agents does not remain fixed and changes with time. It has been proved that the MAS under the communication constraints can achieve consensus with switching topologies if the average dwell time remains greater than a certain value. The developed control law has also been validated through MATLAB simulations.

- In chapter 3, firstly the problem of formation tracking has been studied. The results of the designed leader-following consensus protocol has been extended to achieve formation tracking of MAS with communication constraints. The formation tracking protocol is designed by introducing position and velocity offsets in the consensus controller. These offsets depend on the required formation shape. It has been shown that both fixed and time-varying formation shapes can be achieved through the developed protocol.

  In the second part of chapter 3, a collision avoidance mechanism has been incorporated with the formation controller. An APF based approach has been used to introduce a repulsion force between the agents if they are very close. A stability analysis has been provided along with simulation results to illustrate the efficacy of the developed algorithm.

- The main focus of chapter 4 was the application of developed algorithms to a fleet of wheeled mobile robots. To begin, the robotic platform and its specifications have been presented along with an introduction of ROS. Then a setup of ROS based distributed communication network

for multi-robot system has been presented which was developed for the implementation of the designed consensus tracking algorithm. A new control scheme has been developed to deal with the nonholonomic constraints of the robot. The proposed scheme allows us to use the algorithms for double integrator on differential drive wheeled robots. The results obtained through ROS-Gazebo simulations and hardware implementation have shown the effectiveness of the proposed consensus and formation tracking algorithms in real world scenarios.

Future prospects

Based on the results obtained in this thesis, following are some problems which can be considered in future work.

- The results in the current thesis are obtained for MAS with double-integrator agent dynamics. It is possible to extend these results for a larger scale of systems including higher-order, nonlinear and chained form dynamical models etc. Some preliminary results have already been obtained for MAS having general N-order dynamics with Lipschitz nonlinearities and bounded uncertainties [186].
- In chapter 2, the leader-following consensus algorithms have been developed for a MAS with either a fixed communication topology or when the topology is switching between the graphs having a spanning tree. It will be interesting to investigate the case where individual graphs do not necessarily have a spanning tree but only a joint-graph contains a spanning tree in a time period. Moreover, for switching topology, we considered the leader with constant velocity. The case of leader with acceleration is anticipated as future work. Similarly, we studied the formation tracking and collision avoidance for fixed graphs only. The extension of these results for the switching topology case is considered as future work. Another interesting research direction would be finding a mechanism to maintain a connected graph for a system time-varying topologies.
- We considered that the dynamics of the leader and the followers are the same. However, in practice, sometimes leader's dynamics could be different from the follower dynamics. Similarly, the followers could have different dynamics. Development of distributed consensus and formation tracking algorithm with heterogeneous agents can be considered in future work.
- It will be worth studying the case of formation tracking problem with multiple leaders where MAS can split in groups with different geometric shapes and then can merge to form a new shape. Such operations are quite useful to divide or combine objectives and to avoid collision with obstacles in the path of the desired trajectory.

Abstract

Multi-agent systems (MAS) have gained much popularity due to their vast range of applications. MAS is deployed to achieve more complex goals which could not be realized by a single agent alone. Communication and information exchange among the agents in a MAS is crucial to control its cooperative behavior. Agents share their information with their neighbors to reach a common objective, thus do not require any central monitoring unit. However, the communication among the agents is subject to various practical constraints. These constraints include irregular and asynchronous sampling periods and the availability of partial states only. Such constraints pose significant theoretical and practical challenges. In this thesis, we investigate two fundamental problems related to distributed cooperative control, namely consensus and formation control, of double-integrator MAS under these constraints. It is considered that each agent in the network can measure and transmit its position state only at nonuniform and asynchronous sampling instants. Moreover, the velocity and acceleration are not available. First, we study the problem of distributed control of leader-following consensus. A continuous-discrete time observer based leader-following algorithm is proposed. The observer estimates the position and velocity of the agent and its neighbor in continuous time from the available sampled position data. Then these estimated states are used for the computation of the control input. Both fixed and switching topology scenarios are discussed. Secondly, a consensus based distributed formation tracking protocol is designed to achieve both fixed and time-varying formation patterns. Collision avoidance problem is also studied in this thesis. An Artificial Potential Function (APF) based collision avoidance mechanism is incorporated with the formation tracking algorithm to prevent collisions between the agents while converging to a desired position. Finally, the proposed algorithms are applied on a multi-robot network, consisting of differential drive robots using Robot Operating System (ROS). A new scheme is proposed to deal with nonholonomic constraints of the robot. Efficiency of the designed algorithms and their effectiveness in real world applications are shown through both simulation and hardware results.

Keywords: Multi-agent system, Leader-following consensus, formation tracking, multi-robot network, sampled data, asynchronous sampling, nonuniform sampling

Appendix A

Proof of Theorem 24

The proof of Theorem 24 is divided into several steps. In the first step, system (2.9) along with the control law (2.17) and the observer (2.18)-(2.19) are re-written in a more compact form. Equations for tracking and observer errors are also derived in this step. New coordinates for high-gain design are introduced in the second step, while, in step 3, variables of the state of tracking and observer errors are combined in new variables. Candidate Lyapunov functions are introduced in step 4 and inequalities involving their derivatives are derived. Then in step 5, it is shown that Lemma 16 can be applied if the conditions given in Theorem 24 are satisfied. Finally, in the last step, an inequality involving the tracking error in original coordinates is obtained.

Step 1.

The dynamics of the agents can be written as

$$\begin{cases} \dot{x}_i = Ax_i + Bu_i \quad i = 0, \dots, N \\ r_i = Cx_i \end{cases} \tag{A.1}$$

where $A = \begin{pmatrix} 0_m & I_m \\ 0_m & 0_m \end{pmatrix}$, $B = \begin{pmatrix} 0_m \\ I_m \end{pmatrix}$ and $C = \begin{pmatrix} I_m & 0_m \end{pmatrix}$.

Denoting $\hat{x}_{i,j} = (\hat{r}_{i,j}^T, \hat{v}_{i,j}^T)^T$, system (2.18)-(2.19) can be written as

$$\dot{\hat{x}}_{i,j}(t) = A\hat{x}_{i,j}(t) - \theta\Delta_\theta^{-1}K_o e^{-2\theta(t-\kappa_{i,j}(t))}(\hat{r}_{i,j}(\kappa_{i,j}(t)) - r_j(\kappa_{i,j}(t))), \quad i = 1 \dots N, j = 0 \dots N$$

where

$$\kappa_{i,j}(t) = max\left\{t_k^{i,j} \mid t_k^{i,j} \leq t, k \in \mathbb{N}\right\}$$

represents the last instant, among all the sampling times represented by $t_k^{i,j}$, when the measurements of agent j have been received by agent i. While

$$\Delta_\theta = \begin{pmatrix} I_m & 0_m \\ 0_m & \frac{1}{\theta}I_m \end{pmatrix}$$
$$K_o = P^{-1}C^T = \begin{bmatrix} 2I_m & I_m \end{bmatrix}^T$$

with P the symmetric positive definite matrix solution of the following equation (see [187] for more details).

$$P + A^T P + PA = C^T C \tag{A.2}$$

The input u_i can also be written as

$$\begin{aligned}
u_i &= -\bar{c}\lambda^2 \sum_{k=1}^{N} \mathcal{L}_{ik}\hat{r}_k^i - \bar{c}\lambda^2 b_i(\hat{r}_i^i - \hat{r}_0^i) - \bar{c}2\lambda \sum_{k=1}^{N} \mathcal{L}_{ik}\hat{v}_k^i - \bar{c}2\lambda b_i(\hat{v}_i^i - \hat{v}_0^i) \\
&= -\bar{c}\lambda^2 \sum_{k=1}^{N} H_{ik}(\hat{r}_k^i - \hat{r}_0^i) - \bar{c}2\lambda \sum_{k=1}^{N} H_{ik}(\hat{v}_k^i - \hat{v}_0^i) \\
&= -\bar{c} \sum_{k=1}^{N} H_{ik} \begin{pmatrix} I_m & 2I_m \end{pmatrix} \begin{pmatrix} \lambda^2 I_m & 0_m \\ 0_m & \lambda I_m \end{pmatrix} \begin{pmatrix} \hat{x}_k^i - \hat{x}_0^i \end{pmatrix} \\
&= -\bar{c} \sum_{k=1}^{N} H_{ik} K_c \Gamma_\lambda \begin{pmatrix} \hat{x}_k^i - \hat{x}_0^i \end{pmatrix}
\end{aligned}$$

where

$$\begin{aligned}
\Gamma_\lambda &= \begin{pmatrix} \lambda^2 I_m & 0_m \\ 0_m & \lambda I_m \end{pmatrix} \\
K_c &= B^T Q = \begin{pmatrix} I_m & 2I_m \end{pmatrix}
\end{aligned}$$

with Q is the symmetric positive definite matrix solution of the following matrix equality (see [188] for more details).

$$Q + QA + A^T Q = QBB^T Q \tag{A.3}$$

Defining the estimation error

$$\tilde{x}_{i,j} = \hat{x}_{i,j} - x_j$$

and the tracking error

$$e_i = x_i - x_0$$

for $j = 0, \ldots, N$ and $i = 1, \ldots, N$ gives

$$\begin{aligned}
\dot{\tilde{x}}_{i,j}(t) &= A\tilde{x}_{i,j}(t) - \theta\Delta_\theta^{-1} K_o e^{-2\theta(t-\kappa_{i,j}(t))} \left(C\hat{x}_{i,j}(\kappa_{i,j}(t)) - Cx_j(\kappa_{i,j}(t))\right) - Bu_j(t) \\
&= (A - \theta\Delta_\theta^{-1} K_o C)\tilde{x}_{i,j}(t) - \theta\Delta_\theta^{-1} K_o z_{i,j}(t) - Bu_j(t)
\end{aligned}$$

where $z_{i,j}(t) = \left[e^{-2\theta(t-\kappa_{i,j}(t))} C\tilde{x}_{i,j}(\kappa_{i,j}(t)) - C\tilde{x}_{i,j}(t)\right]$ and

$$\begin{aligned}
\dot{e}_i(t) &= Ae_i(t) + Bu_i(t) - Bu_0(t) \\
u_i &= -\bar{c}K_c\Gamma_\lambda \sum_{k=1}^{N} \mathcal{H}_{ik} e_k - \bar{c}K_c\Gamma_\lambda \sum_{k=1}^{N} \mathcal{H}_{ik}\tilde{x}_{i,k} + b_i\bar{c}K_c\Gamma_\lambda\tilde{x}_{i,0}
\end{aligned}$$

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

Step 2.

Consider the coordinates for high-gain design $\bar{e}_i = \Gamma_\lambda e_i$ and $\bar{x}_{i,j} = \Delta_\theta\tilde{x}_{i,j}$ using the equalities,

$$\begin{aligned}
\Delta_\theta A\Delta_\theta^{-1} &= \theta A \\
C\Delta_\theta^{-1} &= C \\
\Gamma_\lambda A\Gamma_\lambda^{-1} &= \lambda A \\
\Gamma_\lambda B &= \lambda B \\
\Delta_\theta B &= \frac{1}{\theta} B \\
B^T\Delta_\theta^{-1} &= \theta B^T
\end{aligned}$$

yields

$$\begin{aligned}\dot{\bar{e}}_i(t) &= \lambda A\bar{e}_i(t) + \lambda Bu_i(t) - \lambda Bu_0(t)\\ \dot{\bar{x}}_{i,j}(t) &= \theta(A - K_oC)\bar{x}_{i,j}(t) - \theta K_o z_{i,j}(t) - \frac{1}{\theta}Bu_j(t)\end{aligned}$$

and

$$\begin{aligned}u_i &= -\bar{c}K_c\sum_{k=1}^{N}\mathcal{H}_{ik}\bar{e}_k - \bar{c}K_c\Gamma_\lambda\Delta_\theta^{-1}\sum_{k=1}^{N}\mathcal{H}_{ik}\bar{x}_{i,k} + b_i\bar{c}K_c\Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{i,0}\\ &= \bar{c}K_c\left(b_i\Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{i,0} - \sum_{k=1}^{N}\mathcal{H}_{ik}(\bar{e}_k + \Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{i,k})\right)\end{aligned}$$

Step 3.

Denoting $\eta^c = [\bar{e}_1^T \dots \bar{e}_N^T]^T$, $\eta_i^o = [(\bar{x}_{i,1})^T \dots (\bar{x}_{i,N})^T]^T$, $i = 1 \dots N$ and $\eta_0^o = [(\bar{x}_{1,0})^T \dots (\bar{x}_{N,0})^T]$, the tracking error can be written in compact form as

$$\begin{aligned}\dot{\eta}^c &= \lambda[I_N \otimes A]\eta^c - \bar{c}\lambda[\mathcal{H} \otimes (BK_c)]\eta^c - \bar{c}\lambda\sum_{i=1}^{N}[(\mathcal{D}_i^N\mathcal{H}) \otimes (BK_c\Gamma_\lambda\Delta_\theta^{-1})]\eta_i^o\\ &\quad +\bar{c}\lambda[I_N \otimes (BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B} \otimes I_{2m}]\eta_0^o - \lambda[\mathbf{1}_N \otimes B]u_0\end{aligned}$$

Step 4.

Let us consider the following Lyapunov functions

$$\bar{V}_c(\eta^c) = (\eta^c)^T[\Omega \otimes Q]\eta^c \tag{A.4}$$

$$V_o(\bar{x}_{i,j}) = (\bar{x}_{i,j})^T P(\bar{x}_{i,j}) \tag{A.5}$$

$$\bar{V}_o(\eta^o) = \sum_{i=1}^{N}\sum_{j=0}^{N} s_{ij}V_o(\bar{x}_{i,j}) \tag{A.6}$$

where $s_{ij} = 1$ if agent i receives information from agent j and 0 otherwise for $i = 1, \dots, N$, $j = 0, \dots, N$ and η^o is the vector containing all the $\bar{x}_{i,j}$ such that $s_{ij} = 1$.
Step 4.1.
The derivative of the Lyapunov function (A.4) can be written as

$$\begin{aligned}\dot{\bar{V}}_c(\eta^c) &= \lambda(\eta^c)^T[\Omega \otimes A^TQ]\eta^c + \lambda(\eta^c)^T[\Omega \otimes QA]\eta^c - \bar{c}\lambda(\eta^c)^T[(\mathcal{H}^T\Omega) \otimes ((BK_c)^TQ)]\eta^c\\ &\quad -\bar{c}\lambda(\eta^c)^T[(\Omega\mathcal{H}) \otimes (QBK_c)]\eta^c + 2\bar{c}\lambda(\eta^c)^T[\Omega \otimes (QBK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B} \otimes I_{2m}]\eta_0^o\\ &\quad -2\bar{c}\lambda\sum_{i=1}^{N}(\eta^c)^T[(\Omega\mathcal{D}_i^N\mathcal{H}) \otimes (QBK^c\Gamma_\lambda\Delta_\theta^{-1})]\eta_i^o - 2\lambda(\eta^c)^T[(\Omega\mathbf{1}_N) \otimes (QB)]u_0\\ &= \lambda(\eta^c)^T[\Omega \otimes (A^TQ + QA)]\eta^c - \bar{c}\lambda(\eta^c)^T[(\mathcal{H}^T\Omega) \otimes ((BK_c)^TQ) + (\Omega\mathcal{H}) \otimes (QBK_c)]\eta^c\\ &\quad +2\bar{c}\lambda(\eta^c)^T[\Omega \otimes (QBK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B} \otimes I_{2m}]\eta_0^o - 2\bar{c}\lambda\sum_{i=1}^{N}(\eta^c)^T[(\Omega\mathcal{D}_i^N\mathcal{H}) \otimes (QBK^c\Gamma_\lambda\Delta_\theta^{-1})]\eta_i^o\\ &\quad -2\lambda(\eta^c)^T[(\Omega\mathbf{1}_N) \otimes (QB)]u_0\end{aligned} \tag{A.7}$$

Since

$$(BK^c)^TQ = QBK^c = QBB^TQ$$

It can be shown by using point k) of Lemma 15 that

$$
\begin{aligned}
&(\mathcal{H}^T\Omega)\otimes((BK_c)^TQ)+(\Omega\mathcal{H})\otimes(QBK_c)\\
&=[\mathcal{H}^T\Omega+\Omega\mathcal{H}]\otimes[QBB^TQ]\\
&\geq\rho(I_N\otimes[QBB^TQ])\\
&\geq\frac{\rho}{\omega_{\max}}\Omega\otimes[QBB^TQ]
\end{aligned}
$$

therefore

$$
\begin{aligned}
&\lambda(\eta^c)^T[\Omega\otimes(A^TQ+QA)]\eta^c-\bar{c}\lambda(\eta^c)^T[(\mathcal{H}^T\Omega)\otimes((BK_c)^TQ)+(\Omega\mathcal{H})\otimes(QBK_c)]\eta^c)\\
&=\lambda(\eta^c)^T[\Omega\otimes(QBB^TQ-Q)]\eta^c-\bar{c}\lambda(\eta^c)^T[(\mathcal{H}^T\Omega+\Omega\mathcal{H})\otimes(QBB^TQ)]\eta^c\\
&\leq-\lambda\bar{V}_c(\eta^c)+\lambda(\eta^c)^T[\Omega\otimes QBB^TQ]\eta^c-\frac{\rho}{\omega_{\max}}\bar{c}\lambda(\eta^c)^T[\Omega\otimes QBB^TQ]\eta^c
\end{aligned}
$$

If $\bar{c}\geq\frac{\omega_{\max}}{\rho}$, then the above inequality can be reduced to

$$
\begin{aligned}
\lambda(\eta^c)^T[\Omega\otimes(AQ^T+QA)]\eta^c-\bar{c}\lambda(\eta^c)^T[(\mathcal{H}^T\Omega)\otimes((BK_c)^TQ)]\eta^c\\
-\bar{c}\lambda(\eta^c)^T[(\Omega\mathcal{H})\otimes(QBK_c)]\eta^c\leq-\lambda\bar{V}_c(\eta^c)
\end{aligned}
\tag{A.8}
$$

Moreover, one has the following inequality,

$$
\begin{aligned}
&2\bar{c}\lambda(\eta^c)^T[\Omega\otimes(QBK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\otimes I_{2m}]\eta_0^o\\
=\;&2\bar{c}\lambda(\eta^c)^T[(\Omega\otimes Q)(I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\otimes I_{2m}]\eta_0^o\\
\leq\;&2\bar{c}\lambda\sqrt{(\eta^c)^T[\Omega\otimes Q]\eta^c}\sqrt{((I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\eta_0^o])^T[\Omega\otimes Q](I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\eta_0^o])}\\
&\text{by using Lemma 15-h)}\\
\leq\;&2\bar{c}\lambda\sqrt{V_c(\eta^c)}\sqrt{\lambda_{\max}(\Omega\otimes Q)((I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\eta_0^o])^T(I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\eta_0^o])}\\
&\text{by using Lemma 15-i)}\\
\leq\;&2\bar{c}\lambda\sqrt{\bar{V}_c(\eta^c)}\sqrt{\omega_{\max}}\sqrt{\lambda_{\max}(Q)}\|[I_N\otimes BK_c\Gamma_\lambda\Delta_\theta^{-1}][\mathcal{B}\otimes I_{2m}]\eta_0^o\|_2\\
&\text{by using Lemma 15-g)}
\end{aligned}
\tag{A.9}
$$

Furthermore

$$
\begin{aligned}
\|[\mathcal{B}\otimes I_{2m}]\eta_0^o\|_2^2&=\sum_{i=1}^N b_i\|\bar{x}_{i,0}\|_2^2\\
&\leq\sum_{i=1}^N\frac{b_i}{\lambda_{\min}(P)}\left((\bar{x}_{i,0})^TP\bar{x}_{i,0}\right)\quad\text{by using Lemma 15-i)}\\
&\leq\frac{1}{\lambda_{\min}(P)}\sum_{i=1}^N s_{i0}V_o(\bar{x}_{i,0})
\end{aligned}
\tag{A.10}
$$

By Lemma 15-f) and since $\|I_N\|_2=1$, we obtain

$$
\begin{aligned}
&2\bar{c}\lambda(\eta^c)^T[\Omega\otimes(QBK_c\Gamma_\lambda\Delta_\theta^{-1})][\mathcal{B}\otimes I_{2m}]\eta_0^o\\
&\leq2\bar{c}\lambda\|\Gamma_\lambda\Delta_\theta^{-1}\|\sqrt{\omega_{\max}}\frac{\sqrt{\lambda_{\max}(Q)}}{\sqrt{\lambda_{\min}(P)}}\|BK_c\|\sqrt{\bar{V}_c(\eta^c)}\sqrt{\sum_{i=0}^N s_{i0}V_o(\bar{x}_{i0})}\\
&\leq2\bar{c}\lambda\|\Gamma_\lambda\Delta_\theta^{-1}\|\sqrt{\omega_{\max}}\frac{\sqrt{\lambda_{\max}(Q)}}{\sqrt{\lambda_{\min}(P)}}\|K_c\|\sqrt{\bar{V}_c(\eta^c)}\sqrt{\sum_{i=0}^N s_{i0}V_o(\bar{x}_{i0})}\\
&\text{since }\|B\|=1
\end{aligned}
\tag{A.11}
$$

For the second term, we have

$$
\begin{aligned}
& -2\bar{c}\lambda \sum_{i=1}^{N} (\eta^c)^T [(\Omega \mathcal{D}_i^N \mathcal{H}) \otimes (QBK^c \Gamma_\lambda \Delta_\theta^{-1})] \eta_i^o \\
= \; & -2\bar{c}\lambda \sum_{i=1}^{N} (\eta^c)^T (\Omega \otimes Q)[(\mathcal{D}_i^N \mathcal{H}) \otimes (BK_c \Gamma_\lambda \Delta_\theta^{-1})] \eta_i^o \\
\leq \; & -2\bar{c}\lambda \sum_{i=1}^{N} \sqrt{(\eta^c)^T (\Omega \otimes Q) \eta^c} \sqrt{([(\mathcal{D}_i^N \mathcal{H}) \otimes (BK_c \Gamma_\lambda \Delta_\theta^{-1})] \eta_i^o)^T (\Omega \otimes Q)[(\mathcal{D}_i^N \mathcal{H}) \otimes (BK_c \Gamma_\lambda \Delta_\theta^{-1})] \eta_i^o} \\
& \text{by Lemma 15-h)} \\
\leq \; & 2\bar{c}\lambda \sum_{i=1}^{N} \sqrt{\bar{V}_c(\eta^c)} \sqrt{\lambda_{\max}(\Omega \otimes Q)} \|(\mathcal{D}_i^N \mathcal{H}) \otimes (BK_c \Gamma_\lambda \Delta_\theta^{-1})\|_2 \sqrt{\sum_{j=1}^{N} s_{ij} \|\bar{x}_{i,j}\|} \quad \text{by Lemma 15-i)} \\
\leq \; & 2\bar{c}\lambda \sqrt{\omega_{\max} \lambda_{\max}(Q)} \|\mathcal{H}\| \, \|K_c\| \, \|\Gamma_\lambda \Delta_\theta^{-1}\| \sqrt{\bar{V}_c(\eta^c)} \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{N} s_{ij} \|\bar{x}_{i,j}\|} \\
& \text{by using Lemma 15-f) and } \|\mathcal{D}_i^N\|_2 = 1, \; \|B\|_2 = 1 \\
\leq \; & 2\bar{c}\lambda \sqrt{\omega_{\max}} \frac{\sqrt{\lambda_{\max}(Q)}}{\sqrt{\lambda_{\min}(P)}} \|\mathcal{H}\| \, \|K_c\| \, \|\Gamma_\lambda \Delta_\theta^{-1}\| \sqrt{\bar{V}_c(\eta^c)} \sum_{i=1}^{N} \sqrt{\sum_{j=1}^{N} s_{ij} V(\bar{x}_{i,j})} \qquad \text{(A.12)} \\
& \text{by using Lemma 15-i)}
\end{aligned}
$$

Finally, by using inequalities (A.11) and (A.12), one obtains

$$
\begin{aligned}
& 2\bar{c}\lambda (\eta^c)^T [\Omega \otimes (QBK_c \Gamma_\lambda \Delta_\theta^{-1})][b \otimes I_{2m}] \eta_0^o - 2\bar{c}\lambda \sum_{i=1}^{N} (\eta^c)^T [(\Omega \mathcal{D}_i^N \mathcal{H}) \otimes (QBK_c \Gamma_\lambda \Delta_\theta^{-1})] \eta_i^o \\
\leq \; & 2\bar{c}\lambda \sqrt{\omega_{\max}} \sqrt{\frac{\lambda_{\max}(Q)}{\lambda_{\min}(P)}} \max\{1, \|\mathcal{H}\|\} \, \|K_c\| \, \|\Gamma_\lambda \Delta_\theta^{-1}\| \sqrt{\bar{V}_c(\eta^c)} \sum_{i=0}^{N} \sqrt{V_o(\eta_i^o)} \\
\leq \; & 2\lambda k_1 \|\Gamma_\lambda \Delta_\theta^{-1}\| \sqrt{\bar{V}_c(\eta^c)} \sqrt{\bar{V}_o(\eta^o)} \\
& \text{by using Lemma 15-c)}
\end{aligned}
$$

where

$$
k_1 = \bar{c}\sqrt{N+1} \sqrt{\omega_{\max}} \sqrt{\frac{\lambda_{\max}(Q)}{\lambda_{\min}(P)}} \max\{1, \|\mathcal{H}\|\} \, \|K_c\| \qquad \text{(A.13)}
$$

Similarly, for the last term of (A.7), following can be shown,

$$
\begin{aligned}
& -2\lambda (\eta^c)^T [(\Omega \mathbf{1}_N) \otimes (QB)] u_0 \\
= \; & -2\lambda (\eta^c)^T (\Omega \otimes Q)(\mathbf{1}_N \otimes B) u_0 \\
\leq \; & 2\lambda \sqrt{(\eta^c)^T (\Omega \otimes Q) \eta^c} \sqrt{[(\mathbf{1}_N \otimes B) u_0]^T (\Omega \otimes Q)(\mathbf{1}_N \otimes B) u_0} \\
& \text{by using Lemma 15-h)} \\
\leq \; & 2\lambda \sqrt{\bar{V}_c(\eta^c)} \sqrt{\lambda_{\max}(\Omega \otimes Q)} \sqrt{N} \delta_0 \\
& \text{by using Lemma 15-i) and the fact that } \|\mathbf{1}_N\| = \sqrt{N}, \|u_0\| \leq \delta_0 \\
\leq \; & 2k_2 \lambda \delta_0 \sqrt{\bar{V}_c(\eta^c)} \qquad \text{(A.14)} \\
& \text{by using Lemma 15-g)}
\end{aligned}
$$

where

$$\bar{k}_2 = \sqrt{N}\sqrt{\omega_{\max}}\sqrt{\lambda_{\max}(Q)} \tag{A.15}$$

Using inequalities (A.8), (A.13) and (A.14) lead to

$$\dot{\bar{V}}_c(\eta^c) \leq -\lambda \bar{V}_c(\eta^c) + 2k_1\lambda\|\Gamma_\lambda \Delta_\theta^{-1}\|\sqrt{\bar{V}_c(\eta^c)}\sqrt{\bar{V}_o(\eta^o)} + 2\bar{k}_2\lambda\delta_0\sqrt{\bar{V}_c(\eta^c)} \tag{A.16}$$

Step 4.2.
For i, j such that $s_{ij} = 1$, the derivative of (A.5) is given by

$$\begin{aligned}\dot{V}_o(\bar{x}_{i,j}) &= \theta(\bar{x}_{i,j})^T[(A - K_oC)^TP + P(A - K_oC)](\bar{x}_{i,j}) \\ &\quad -2\theta(\bar{x}_{i,j})^TPK_oz_{i,j} - \frac{2}{\theta}(\bar{x}_{i,j})^TPBu_j\end{aligned} \tag{A.17}$$

One has

$$\begin{aligned}&\theta(\bar{x}_{i,j})^T[(A - K_oC)^TP + P(A - K_oC)](\bar{x}_{i,j}) \\ &= \theta(\bar{x}_{i,j})^T[A^TP - C^TC + PA - C^TC](\bar{x}_{i,j}) \\ &= -\theta(\bar{x}_{i,j})^TP(\bar{x}_{i,j}) - \theta(\bar{x}_{i,j})^TC^TC(\bar{x}_{i,j}), \qquad \text{by using the definition of } P \\ &\leq -\theta V_o(\bar{x}_{i,j})\end{aligned} \tag{A.18}$$

$$\begin{aligned}&-2\theta(\bar{x}_{i,j})^TPK_oz_{i,j}(t) \leq 2\theta\sqrt{\lambda_{\max}(P)}\|K_o\|\sqrt{V_o(\bar{x}_{i,j})}\|z_{i,j}(t)\| \\ &\text{by using Lemma 15-i)}\end{aligned} \tag{A.19}$$

Since

$$\begin{aligned}\dot{z}_{i,j}(t) &= -2\theta e^{-2\theta(t-\kappa_{i,j}(t))}C\tilde{x}^i_j(\kappa_{i,j}(t)) - C\dot{\tilde{x}}_{i,j}(t) \\ &= -B^T\tilde{x}_{i,j}(t)\end{aligned}$$

and $z_{i,j}(\kappa_{i,j}(t)) = 0$, $\forall t \geq 0$, then we have

$$\begin{aligned}z_{i,j}(t) &= -\int_{\kappa_{i,j}(t)}^{t} B^T\tilde{x}_{i,j}(s)ds \\ &= -\theta\int_{\kappa_{i,j}(t)}^{t} B^T\bar{x}_{i,j}(s)ds\end{aligned}$$

or

$$\begin{aligned}\|z_{i,j}\| &= \theta\left\|\int_{\kappa_{i,j}(t)}^{t} B^T\bar{x}_{i,j}(s)ds\right\| \\ &\leq \theta\int_{t-\tau_M}^{t}\|\bar{x}_{i,j}(s)\|ds\end{aligned}$$

The above inequality is achieved by using the definition of $\kappa_{i,j}(t)$ and the fact that $t - \kappa_{i,j}(t) \leq t^{i,j}_{k+1} - t^{i,j}_k \leq \tau_M$, for all $t \geq 0$, $k \in \mathbb{N}$ and $\left\|B^T\right\| = 1$. Furthermore, by using Lemma 15-i), we get

$$z_{i,j}(t) \leq \frac{\theta}{\sqrt{\lambda_{\min}(P)}}\int_{t-\tau_M}^{t}\sqrt{((\bar{x}_{i,j}(s))^TP(\bar{x}_{i,j}(s)))}ds$$

Therefore, (A.20) becomes

$$-2\theta(\bar{x}_{i,j})^TPK_oz_{i,j}(t) \leq 2\theta^2\bar{k}_3\sqrt{V_o(\bar{x}_{i,j}(t))}\int_{t-\tau_M}^{t}\sqrt{V_o(\bar{x}_{i,j}(s))}ds$$

where

$$\bar{k}_3 = \frac{\sqrt{\lambda_{\max}(P)}}{\sqrt{\lambda_{\min}(P)}}\|K_o\| \tag{A.20}$$

Furthermore,

$$\|u_j\| \leq \bar{c}\|K_c\|\left(\|b_i\Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{j,0}\| + \sum_{k=1}^{N}\|\mathcal{H}_{i,k}(\bar{e}_k + \Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{i,k})\|\right)$$

By using the definition of $s_{i,j}$ and $\mathcal{H}$, we get

$$\|u_j\| \leq \bar{c}\|K_c\|h_{\max}\left(\sum_{k=1}^{N}\|\bar{e}_k\| + \|\Gamma_\lambda\Delta_\theta^{-1}\|\sum_{k=0}^{N}s_{jk}\|\bar{x}_{j,k}\|\right)$$

where $h_{\max}$ is defined by (2.16). Since

$$\|\bar{e}_k\| = \sqrt{\bar{e}_k^T e_k}$$

so by using lemma 15-i), one has

$$\begin{aligned}\|\bar{e}_k\| &\leq \frac{1}{\sqrt{\lambda_{\min}(Q)}}\sqrt{\bar{e}_k^T Q\bar{e}_k}\\ \sum_{k=1}^{N}\|\bar{e}_k\| &\leq \frac{\sqrt{N}}{\lambda_{\min}(Q)\sqrt{\omega_{min}}}\sqrt{\bar{V}_c(\eta^c)} \quad \text{by using Lemma 15-c)}\end{aligned}$$

Similarly,

$$\|\bar{x}_{j,k}\| \leq \frac{1}{\sqrt{\lambda_{\min}(P)}}\sqrt{V_o(\bar{x}_{j,k})} \quad \text{by using Lemma 15-i)}$$

So it leads to

$$\begin{aligned}-\frac{2}{\theta}(\bar{x}_{i,j})^T PBu_j &\leq \frac{2}{\theta}\sqrt{V_o(\bar{x}_{i,j})}\sqrt{\lambda_{\max}(P)}\|u_j\|\\ &\leq \frac{2}{\theta}\bar{k}_4\sqrt{V_o(\bar{x}_{i,j})}\sqrt{\bar{V}_c(\eta^c)}\\ &\quad + \frac{2}{\theta}\bar{k}_5\|\Gamma_\lambda\Delta_\theta^{-1}\|\sqrt{V_o(\bar{x}_{i,j})}\sum_{k=0}^{N}s_{jk}\sqrt{V_o(\bar{x}_{j,k})}\end{aligned} \tag{A.21}$$

where

$$\bar{k}_4 = \frac{\bar{c}\|K_c\|h_{\max}\sqrt{N}\sqrt{\lambda_{\max}(P)}}{\sqrt{\lambda_{\min}(Q)}\sqrt{\omega_{\min}}} \tag{A.22}$$

$$\bar{k}_5 = \frac{\bar{c}\|K_c\|h_{\max}\sqrt{\lambda_{\max}(P)}}{\sqrt{\lambda_{\min}(P)}} \tag{A.23}$$

Moreover, by using Lemma 15-h)

$$-\frac{2}{\theta}(\bar{x}_{i,0})^T PBu_0 \leq \frac{2}{\theta}\sqrt{\lambda_{\max}(P)}\sqrt{(\bar{x}_{i,0})^T P(\bar{x}_{i,0})}\sqrt{(Bu_0)^T(Bu_0)}$$

which can be further simplified as

$$-\frac{2}{\theta}(\bar{x}_{i,0})^T PBu_0 \leq \frac{2}{\theta}\delta_0\bar{k}_6\sqrt{V_0(\bar{x}_{i,0})} \tag{A.24}$$

The above is achieved by using Lemma 15-i) and the fact that $\|B\| = 1$ and $\|u_0(t)\| \leq \delta_0$ for all $t \geq 0$ and

$$\bar{k}_6 = \sqrt{\lambda_{\max}(P)} \tag{A.25}$$

Using inequalities (A.18), (A.21) and (A.24) leads to

$$\begin{aligned}\dot{V}_o(\bar{x}_{i,j}) &\leq -\theta V_o(\bar{x}_{i,j}) + 2\theta^2\bar{k}_3\sqrt{V_o(\bar{x}_{i,j})}\int_{t-\tau_M}^{t}\sqrt{V_o(\bar{x}_{i,j}(s))}ds + 2\frac{\bar{k}_4}{\theta}\sqrt{V_o(\bar{x}_{i,j})}\sqrt{\bar{V}_c(\eta^c)} \\ &\quad +2\frac{\bar{k}_5}{\theta}\|\Gamma_\lambda\Delta_\theta^{-1}\|\sqrt{V_o(\bar{x}_{i,j})}\sum_{k=0}^{N}s_{jk}\sqrt{V_o(\bar{x}_{j,k})} \\ &\leq -\theta V_o(\bar{x}_{i,j}) + 2\theta^2\bar{k}_3\sqrt{V_o(\bar{x}_{i,j})}\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds + 2\frac{\bar{k}_4}{\theta}\sqrt{V_o(\bar{x}_{i,j})}\sqrt{\bar{V}_c(\eta^c)} \\ &\quad +2\frac{\bar{k}_5}{\theta}\|\Gamma_\lambda\Delta_\theta^{-1}\|\sqrt{V_o(\bar{x}_{i,j})}\sqrt{N+1}\sqrt{\bar{V}_o(\eta^o)} \qquad (A.26)\end{aligned}$$

for $j = 1, \ldots, N$, while

$$\dot{V}_o(\bar{x}_{i,0}) \leq -\theta V_o(\bar{x}_{i,0}) + 2\theta^2\bar{k}_3\sqrt{V_o(\bar{x}_{i,j})}\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds + 2\frac{\bar{k}_6}{\theta}\delta_0\sqrt{V_o(\bar{x}_{i,0})} \qquad (A.27)$$

Step 4.3.
Letting

$$\lambda = \varepsilon\theta$$

with $\varepsilon \in (0,1)$, then

$$\|\Gamma_\lambda\Delta_\theta^{-1}\| = \lambda\theta \qquad (A.28)$$

From the definition of $\dot{\bar{V}}_o(\eta^o)$

$$\dot{\bar{V}}_o(\eta^o) = \sum_{i=1}^{N}\sum_{j=1}^{N}s_{ij}\dot{\bar{V}}_o(\bar{x}_{i,j}) + \sum_{i=1}^{N}s_{ij}\dot{\bar{V}}_o(\bar{x}_{i,0}) \qquad (A.29)$$

then by using (A.26),(A.27), (A.28) and Lemma 15 e), one gets

$$\begin{aligned}\dot{\bar{V}}_o(\eta^o) &\leq -\theta\bar{V}_o(\eta^o) + 2\theta^2 k_3\sqrt{\bar{V}_o(\eta^o)}\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds + 2\lambda k_5\bar{V}_o(\eta^o) \qquad (A.30) \\ &\quad +2\frac{k_4}{\theta}\sqrt{\bar{V}_o(\eta^o)}\sqrt{\bar{V}_c(\eta^c)} + 2\frac{k_6}{\theta}\delta_0\sqrt{\bar{V}_o(\eta^o)} \qquad (A.31)\end{aligned}$$

where

$$\begin{aligned}k_3 &= \bar{k}_3\sqrt{N}\sqrt{N+1} \qquad (A.32)\\ k_4 &= \bar{k}_4 N \qquad (A.33)\\ k_5 &= \bar{k}_5 N\sqrt{N+1} \qquad (A.34)\\ k_6 &= \bar{k}_6\sqrt{N} \qquad (A.35)\end{aligned}$$

Step 5.

From (A.16), we obtain

$$\frac{d}{dt}\left(\sqrt{\bar{V}_c(\eta^c)}\right) = \frac{1}{2\sqrt{\bar{V}_c(\eta^c)}}\dot{\bar{V}}_c(\eta^c) \leq -\frac{\lambda}{2}\sqrt{\bar{V}_c(\eta^c)} + k_1\lambda^2\theta\sqrt{\bar{V}_o(\eta^o)} + \bar{k}_2\lambda\delta_0$$

and similarly from (A.31)

$$\frac{d}{dt}\left(\sqrt{\bar{V}_o(\eta^o)}\right) \leq -\frac{\theta}{2}\sqrt{\bar{V}_o(\eta^o)} + k_3\theta^2\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds + \frac{k_4}{\theta}\sqrt{\bar{V}_c(\eta^c)} + k_5\lambda\sqrt{\bar{V}_o(\eta^o)} + \frac{k_6}{\theta}\delta_0$$

We have

$$\begin{aligned}\frac{d}{dt}\left(\sqrt{\bar{V}_c(\eta^c)}+\varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)}\right) &\leq -\frac{\varepsilon\theta}{4}\left(1-4k_4\varepsilon^{\frac{1}{2}}\right)\sqrt{\bar{V}_c(\eta^c)}-\frac{\varepsilon^{\frac{3}{2}}\theta^3}{4}\left(1-4k_1\varepsilon^{\frac{1}{2}}-4k_5\varepsilon\right)\sqrt{\bar{V}_o(\eta^o)}\\ &\quad -\frac{\varepsilon\theta}{4}\sqrt{\bar{V}_c(\eta^c)}-\frac{\varepsilon^{\frac{3}{2}}\theta^3}{4}\sqrt{\bar{V}_o(\eta^o)}+k_3\varepsilon^{\frac{3}{2}}\theta^4\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds\\ &\quad +\bar{k}_2\varepsilon\theta\delta_0+k_6\varepsilon^{\frac{3}{2}}\theta\delta_0\end{aligned}$$

By choosing $\varepsilon<\varepsilon^*$ where $\varepsilon^*=\min\left\{1,\frac{1}{(4k_4)^2},\frac{1}{(8k_1)^2},\frac{1}{8k_5}\right\}$, we obtain

$$\frac{d}{dt}\left(\sqrt{\bar{V}_c(\eta^c)}+\varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)}\right)\leq -\frac{\varepsilon\theta}{4}\sqrt{\bar{V}_c(\eta^c)}-\frac{\varepsilon^{\frac{3}{2}}\theta^3}{4}\sqrt{\bar{V}_o(\eta^o)}+k_3\varepsilon^{\frac{3}{2}}\theta^4\int_{t-\tau_M}^{t}\sqrt{\bar{V}_o(\eta^o(s))}ds+k_2\varepsilon\theta\delta_0$$

where

$$k_2=\max\{\bar{k}_2,k_6\} \tag{A.36}$$

Applying Lemma 16 with

$$a = \frac{\varepsilon\theta}{4}=\frac{\lambda}{4} \tag{A.37}$$

$$b = \frac{\theta}{4} \tag{A.38}$$

$$c = k_3\varepsilon^{\frac{3}{2}}\theta^4 \tag{A.39}$$

$$k = k_2\varepsilon\theta\delta_0 \tag{A.40}$$

one obtains the existence of $\varrho>0$ and $\bar{\alpha}>0$ such that if

$$\tau_M<\varrho\min\left(\frac{b}{c},\frac{1}{\sigma}\right) \tag{A.41}$$

with $\sigma=\frac{1}{2}\min(a,b)=\frac{\lambda}{8}$ since $\theta>\lambda>0$, then

$$\sqrt{\bar{V}_c(\eta^c)}+\varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)}\leq\bar{\alpha}e^{-\sigma t}+\frac{k_2\varepsilon\theta\delta_0}{\sigma}$$

Since

$$\min\left(\frac{b}{c},\frac{1}{\sigma}\right)=\min\left(\frac{1}{4\theta k_3},\frac{8}{\lambda}\right)\geq\min\left(\frac{1}{4k_3},8\right)\min\left(\frac{1}{\theta},\frac{1}{\lambda}\right)\geq\frac{1}{4k_3\theta} \tag{A.42}$$

then, if θ verifies

$$\tau_M<\frac{\bar{\varrho}}{\theta} \tag{A.43}$$

with $\bar{\varrho}=\frac{\varrho}{4k_3}$, the following inequality holds true

$$\sqrt{\bar{V}_c(\eta^c)}+\varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)}\leq\bar{\alpha}e^{-\frac{\lambda}{8}t}+8k_2\delta_0 \tag{A.44}$$

Step 6.

One now comes back to the original coordinates of both observer and tracking errors. One has

$$\lambda\sqrt{\lambda_{\min}(Q)}\|e_i\| \leq \sqrt{\bar{e}_i^TQ\bar{e}_i}$$

since $\lambda>0$

$$\begin{aligned}\|e_i\| &\leq \frac{1}{\lambda\sqrt{\lambda_{\min}(Q)}\sqrt{\omega_i}}\sqrt{\omega_i\bar{e}_i^TQ\bar{e}_i}\\ \sum_{i=1}^{N}\|e_i\| &\leq \frac{1}{\lambda}\frac{\sqrt{N}}{\sqrt{\lambda_{\min}(Q)}\sqrt{\omega_{\min}}}\sqrt{\bar{V}_c(\eta^c)}\end{aligned}$$

The above inequality has been achieved using Lemma 15-c) and the fact that $\sum_{i=1}^{N} \omega_i \left(\bar{e}_i^T Q \bar{e}_i\right) = \bar{V}_c(\eta^c)$). Rearranging the above inequality, one has

$$\sqrt{\bar{V}_c(\eta^c)} \quad \geq \quad \lambda l_1 \sum_{i=1}^{N} \|e_i\| \tag{A.45}$$

where

$$l_1 \quad = \quad \frac{\sqrt{\lambda_{\min}(Q)}\sqrt{\omega_{\min}}}{\sqrt{N}} \tag{A.46}$$

Similarly, we have

$$\frac{\sqrt{\lambda_{\min}(P)}}{\theta} \|\tilde{x}_{i,j}\| \quad \leq \quad \sqrt{(\bar{x}_{i,j})^T P \bar{x}_{i,j}} \tag{A.47}$$

As $\theta > 0$, we have

$$\begin{aligned}
\sum_{j=0}^{N} \|\tilde{x}_{i,j}\| \quad &\leq \quad \frac{\theta}{\sqrt{\lambda_{\min}(P)}} \sum_{j=0}^{N} \sqrt{V_o(\bar{x}_{i,j})} \\
\sum_{i=1}^{N}\sum_{j=0}^{N} \|\tilde{x}_{i,j}\| \quad &\leq \quad \theta \frac{\sqrt{N}\sqrt{N+1}}{\sqrt{\lambda_{\min}(P)}} \sqrt{\bar{V}_o(\eta^o)} \quad \text{by using Lemma 15-c)} \\
\sqrt{\bar{V}_o(\eta^o)} \quad &\geq \quad \frac{l_2}{\theta} \sum_{i=1}^{N}\sum_{j=0}^{N} \|\tilde{x}_{i,j}\|
\end{aligned} \tag{A.48}$$

where

$$l_2 \quad = \quad \frac{\sqrt{\lambda_{\min}(P)}}{\sqrt{N}\sqrt{N+1}} \tag{A.49}$$

Using inequalities (A.45) and (A.48), one obtains

$$\sqrt{\bar{V}_c(\eta^c)} + \varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)} \quad \geq \quad \lambda l_1 \sum_{i=1}^{N} \|e_i\| + \varepsilon^{\frac{3}{2}}\theta l_2 \sum_{i=1}^{N}\sum_{j=0}^{N} \|\tilde{x}_{i,j}\|$$

Over-valuation of the tracking error e_i is given by

$$\sqrt{\bar{V}_c(\eta^c)} + \varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)} \quad \geq \quad \lambda l_1 \sum_{i=1}^{N} \|e_i\|$$

by using inequality (A.44), it gives

$$\sum_{i=1}^{N} \|e_i\| \quad \leq \quad \alpha e^{-\frac{\lambda}{8}t} + \frac{\beta\delta_0}{\lambda} \tag{A.50}$$

with

$$\alpha = \frac{\bar{\alpha}}{\lambda l_1} \tag{A.51}$$

$$\beta = \frac{8k_2}{l_1} \tag{A.52}$$

This ends the proof. □

Appendix B

Proof of Theorem 29

The proof is divided into two steps. In the first step, the results of fixed topology case are further expanded to obtain some useful inequalities and properties. In the second step, piece-wise candidate Lyapunov function is introduced and it is shown that if the conditions given in Theorem 29 are satisfied then the MAS with switching topology achieves stability.

Step 1:

First, suppose a fixed communication graph $\mathcal{G}^p$ ($p \in \mathcal{M}$). Using the same Lyapunov functions (A.4)-(A.6) with a small modification in the notation of $\bar{V}_c(\eta^c)$ as $\bar{V}_c^p(\eta^c) = (\eta^c)^T[\Omega^p \otimes Q]\eta^c$ to specify it for communication graph $\mathcal{G}^p$ since Ω depends on the communication graph. Then by following the same steps as given in proof of Theorem 24, one can show that if the conditions (2.31)-(2.33) are satisfied and leader's input is zero, i.e. $u_0 = 0$ then

$$\sqrt{\bar{V}_c^p(\eta^c)} + \varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_o(\eta^o)} \leq \bar{\alpha}(t_0)e^{-\frac{\lambda}{8}(t-t_0)}$$

where $\bar{\alpha} \geq 0$. In fact, from Lemma 16 one gets

$$\bar{\alpha}(t_0) = \sqrt{\bar{V}_c^p(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_0(t_0)} + c\varepsilon^{\frac{3}{2}}\theta^2\int_0^{\tau_M}\int_{t_0-s}^{t_0} e^{v\kappa(\mu-t_0+s)}\sqrt{\bar{V}_0(\mu)}d\mu ds \tag{B.1}$$

where v and κ are given in Lemma 16.

Now for $t \in [t_0, t_1)$, one can show that

$$\sqrt{\bar{V}_c^p(t)} + \varepsilon^{\frac{3}{2}}\theta^2\sqrt{\bar{V}_0(t)} \leq \left(\sqrt{\bar{V}_c^p(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 K \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_0)} \tag{B.2}$$

$$\begin{aligned}\sqrt{\bar{V}_c^p(t_1)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_1-\tau_M,t_1)}\sqrt{\bar{V}_0(s)} &\leq \left(\sqrt{\bar{V}_c^p(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 K \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}((t_1-\tau_M)-t_0)} \\ \sqrt{\bar{V}_c^p(t_1)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_1-\tau_M,t_1)}\sqrt{\bar{V}_0(s)} &\leq \left(\sqrt{\bar{V}_c^p(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 K \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)\left(e^{\frac{\lambda}{8}\tau_M}\right)e^{-\frac{\lambda}{8}(t_1-t_0)}\end{aligned} \tag{B.3}$$

where $K = 2\max\{1, c\tau_M^2 e^{v\kappa\tau_M}\}$ with $c \geq 0$.

Each $\bar{V}_c^p(\eta^c)$ is continuous and decreases exponentially. $\bar{V}_c^p(\eta^c)$ satisfies the following properties [147, 189].

- There exists $\bar{\beta} \geq 1$ such that

$$\bar{V}_c^p(\eta^c) \leq \bar{\beta}\bar{V}_c^q(\eta^c), \quad \forall p, q \in \mathcal{M} \tag{B.4}$$

or

$$\sqrt{\bar{V}_c^p(\eta^c)} \leq \beta\sqrt{\bar{V}_c^q(\eta^c)}, \quad \forall p, q \in \mathcal{M} \tag{B.5}$$

where $\beta = \sqrt{\bar{\beta}}$

- Let $\alpha_1 = \min_{p\in\mathcal{M}}(\lambda_{\min}(\Omega^p \otimes Q))$ and $\alpha_1 = \max_{p\in\mathcal{M}}(\lambda_{\max}(\Omega^p \otimes Q))$, then

$$\alpha_1\|\eta^c\|^2 \leq \bar{V}_c^p \leq \alpha_2\|\eta^c\|^2 \tag{B.6}$$

Step 2:

Let us now define a piece-wise Lyapunov function for a switching communication topology

$$\bar{V}_c^{\sigma(t)}(\eta^c) = (\eta^c)^T[\Omega^{\sigma(t)} \otimes Q]\eta^c \tag{B.7}$$

then from (B.5), for any switching instant t_l, $l = 1, 2, \ldots$

$$\sqrt{\bar{V}_c^{\sigma(t_l)}(t_l)} \leq \beta\sqrt{\bar{V}_c^{\sigma(t_l^-)}(t_l^-)} \tag{B.8}$$

and since $\bar{V}_0$ does not switch, therefore

$$\bar{V}_0(t_l) = \bar{V}_0(t_l^-) \tag{B.9}$$

so

$$\left(\sqrt{\bar{V}_c^{\sigma(t_l)}(t_l)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_l,t_l-\tau_M)}\sqrt{\bar{V}_o(s)}\right) \leq \beta\left(\sqrt{\bar{V}_c^{\sigma(t_l^-)}(t_l^-)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_l^-,t_l^--\tau_M)}\sqrt{\bar{V}_o(s)}\right) \tag{B.10}$$

For $t \in [t_k, t_{k+1})$, from (B.2) and (B.10), one have

$$\begin{aligned}
&\sqrt{\bar{V}_c^{\sigma(t)}(t)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t-\tau_M,t)}\sqrt{\bar{V}_0(s)} \\
\leq\ & e^{\frac{\lambda}{8}\tau_M}\left(\sqrt{\bar{V}_c^{\sigma(t_k)}(t_k)} + \varepsilon^{\frac{3}{2}}\theta^2 K \max_{s\in[t_k-\tau_M,t_k]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_k)} \quad \text{from (B.2)} && \text{(B.11)}\\
\leq\ & \beta K e^{\frac{\lambda}{8}\tau_M}\left(\sqrt{\bar{V}_c^{\sigma(t_k^-)}(t_k^-)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_k^--\tau_M,t_k^-]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_k)} \quad \text{from (B.10)} && \text{(B.12)}\\
\leq\ & \beta K^2 e^{2\frac{\lambda}{8}\tau_M}\left(\sqrt{\bar{V}_c^{\sigma(t_0)}(t_{k-1})} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_{k-1}-\tau_M,t_{k-1}]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_{k-1})} \\
\leq\ & \beta^{N_\sigma}(Ke^{\frac{\lambda}{8}\tau_M})^{N_\sigma+1}\left(\sqrt{\bar{V}_c^{\sigma(t_0)}(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_0)} \\
\leq\ & \beta^{N_0}(Ke^{\frac{\lambda}{8}\tau_M})^{N_0+1}(\beta K)^{\frac{t-t_0}{\tau_a}}e^{\frac{t-t_0}{\tau_a}}\left(\sqrt{\bar{V}_c^{\sigma(t_0)}(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)e^{-\frac{\lambda}{8}(t-t_0)}
\end{aligned}$$

where inequalities (B.11) and (B.12) are obtained using (B.2) and (B.10), respectively. Now by using Definition 26, one has

$$\begin{aligned}
&\sqrt{\bar{V}_c^{\sigma(t)}(t)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t-\tau_M,t)}\sqrt{\bar{V}_0(s)} \\
\leq\ & \beta^{N_0}(Ke^{\frac{\lambda}{8}\tau_M})^{N_0+1}\left(\sqrt{\bar{V}_c^{\sigma(t_0)}(t_0)} + \varepsilon^{\frac{3}{2}}\theta^2 \max_{s\in[t_0-\tau_M,t_0]}\sqrt{\bar{V}_o(s)}\right)e^{-\left(\frac{\lambda}{8}-\frac{\ln(\beta K)-1}{\tau_a}\right)(t-t_0)} && \text{(B.13)}
\end{aligned}$$

so if ADT is chosen $\tau_a > \frac{8\ln(\beta K)-1}{\lambda}$, the system achieves exponential stability. □

Appendix C

Proof of Theorem 36

The agent dynamics can be re-written as

$$\begin{cases} \dot{x}_i = Ax_i + Bu_i \quad i = 0, \dots, N \\ r_i = Cx_i \end{cases}$$

with $A = \begin{pmatrix} 0_m & I_m \\ 0_m & 0_m \end{pmatrix}$, $B = \begin{pmatrix} 0_m \\ I_m \end{pmatrix}$ and $C = \begin{pmatrix} I_m & 0_m \end{pmatrix}$. Similarly, the formation dynamics can be written as

$$\dot{f}_i = Af_i + B\dot{f}_{i,v}$$

Denoting $\hat{x}_{i,j} = (\hat{r}_{i,j}^T, \hat{v}_{i,j}^T)^T$, the observer (3.5)-(3.6) can be written as

$$\dot{\hat{x}}_{i,j}(t) = A\hat{x}_{i,j}(t) + B\dot{f}_{i,v} - \theta\Delta_\theta^{-1}K_o e^{-2\theta(t-\kappa_{i,j}(t))}(\hat{r}_{i,j}(\kappa_{i,j}(t)) - r_j(\kappa_{i,j}(t)))$$

for $i = 1, \dots, N$ and $j = 0, \dots, N$ and where $\kappa_{i,j}(t) = max\left\{t_k^{i,j} \mid t_k^{i,j} \leq t, k \in \mathbb{N}\right\}$ is the last instant when the position of agent j has been received by agent i, while

$$\begin{aligned} \Delta_\theta &= \begin{pmatrix} I_m & 0_m \\ 0_m & \frac{1}{\theta}I_m \end{pmatrix} \\ K_o &= \begin{bmatrix} 2I_m & I_m \end{bmatrix}^T \end{aligned}$$

From Definition 34, the tracking error can be defined as:

$$e_i(t) = x_i(t) - f_i(t) - x_0(t)$$

and denoting the estimation error

$$\tilde{x}_{i,j}(t) = \hat{x}_{i,j}(t) - x_j(t)$$

for $j = 0 \dots N$ and $i = 1 \dots N$. The input u_i can be written as

$$u_i = -\bar{c}K_c\Gamma_\lambda \sum_{k=1}^{N} \mathcal{H}_{ik}e_k - \bar{c}K_c\Gamma_\lambda \sum_{k=1}^{N} \mathcal{H}_{ik}\tilde{x}_{i,k} + b_i\bar{c}K^c\Gamma_\lambda\tilde{x}_{i,0} + \dot{f}_{i,v}(t)$$

where $\mathcal{H}_{ik}$ is the ik^{th} element of matrix $\mathcal{H}$ and

$$\begin{aligned} K_c &= \begin{pmatrix} I_m & 2I_m \end{pmatrix} \\ \Gamma_\lambda &= \begin{pmatrix} \lambda^2 I_m & 0_m \\ 0_m & \lambda I_m \end{pmatrix} \end{aligned}$$

Hence, the formation tracking error dynamics is

$$\dot{e}_i = Ae_i + Bu_i - Bu_0 - B\dot{f}_{i,v}$$

while the estimation error dynamics is

$$\dot{\tilde{x}}_{i,j}(t) = (A - \theta\Delta_\theta^{-1}K_oC)\tilde{x}_{i,j}(t) - \theta\Delta_\theta^{-1}K_oz_{i,j}(t) - Bu_j(t) + B\dot{f}_{j,v}(t)$$

where

$$z_{i,j}(t) = \left[e^{-2\theta(t-\kappa_{i,j}(t))}C\tilde{x}_{i,j}(\kappa_{i,j}(t)) - C\tilde{x}_{i,j}(t)\right]$$

Now using high-gain transformation $\bar{e}_i = \Gamma_\lambda e_i$ and $\bar{x}_{i,j} = \Delta_\theta\tilde{x}_{i,j}$. One gets

$$\dot{\bar{e}}_i(t) = \lambda A\bar{e}_i(t) + \lambda Bu_i(t) - \lambda Bu_0(t) - \lambda B\dot{f}_{i,v}(t) \tag{C.1}$$

$$\dot{\bar{x}}_{i,j}(t) = \theta(A - K_oC)\bar{x}_{i,j}(t) - \theta K_oz_{i,j}(t) - \frac{1}{\theta}Bu_j(t) - \frac{1}{\theta}B\dot{f}_{i,v}(t) \tag{C.2}$$

and

$$u_i = -\bar{c}K_c\sum_{k=1}^{N}\mathcal{H}_{ik}\bar{e}_k - \bar{c}K_c\Gamma_\lambda\Delta_\theta^{-1}\sum_{k=1}^{N}\mathcal{H}_{ik}\bar{x}_{i,k} + b_i\bar{c}K_c\Gamma_\lambda\Delta_\theta^{-1}\bar{x}_{i,0} + \dot{f}_{i,v}(t) \tag{C.3}$$

By using (C.3) in (C.1) and (C.2), one can see that the new tracking and observer error dynamics i.e. $\dot{\bar{e}}_i(t)$ and $\dot{\bar{x}}_{i,j}(t)$ are same as those presented in Section 1.4.1 of Chapter 2. Using

$$\begin{aligned}\eta^c &= [\bar{e}_1^T \dots \bar{e}_N^T]^T \\ \eta_i^o &= [(\bar{x}_{i,1})^T \dots (\bar{x}_{i,N})^T]^T, \qquad i = 1, \dots, N \\ \eta_0^o &= [\bar{x}_{1,0} \dots \bar{x}_{N,0}]\end{aligned}$$

and the following Lyapunov functions:

$$V_c(\eta^c) = (\eta^c)^T[\Omega \otimes Q]\eta^c \tag{C.4}$$

$$V_o(\tilde{X}_{i,j}) = (\tilde{X}_{i,j})^T P(\tilde{X}_{i,j}) \tag{C.5}$$

$$\bar{V}_o(\tilde{X}_{i,j}) = \sum_{i=1}^{N}\sum_{j=0}^{N}s_{ij}V_o(\tilde{X}_{i,j}) \tag{C.6}$$

where $s_{ij} = 1$ if agent i receives information from agent j and 0 otherwise for $i = 1, \dots, N$, $j = 0, \dots, N$, P is the symmetric positive definite matrix solution of the equation

$$P + A^TP + PA = C^TC$$

and Q is the symmetric positive definite matrix solution of the equation

$$Q + QA + A^TQ = QBB^TQ$$

By following same steps given in Chapter 2, one can show that if conditions (3.8)–(3.10) are satisfied, then

$$\sum_{i=1}^{N}\|e_i(t)\| \leq \alpha e^{-\frac{\lambda}{\gamma}t} + \frac{\beta\delta_0}{\lambda}, \qquad \forall t \geq 0 \tag{C.7}$$

where $\alpha, \beta, \gamma > 0$ and δ_0 is the upper bound of the leader input. Furthermore, β does not depend on $\theta, \lambda, \bar{c}, \tau_M$ and the initial conditions of the agents and observers. Details of these parameters are given in Chapter 2. □

Appendix D

Proof of Theorem 42

The position and velocity errors of agent i for fixed formation can be defined respectively as:

$$\xi_i = r_i - f_{i,r} - r_0 \tag{D.1}$$
$$\zeta_i = v_i - v_0 \tag{D.2}$$

Step 1

Consider system (2.9)–(2.10) with only formation controller (3.7) and choose Lyapunov function as:

$$V_1 = \frac{1}{2}\sum_{i=1}^{N}[\xi_i^T\xi_i + \zeta_i^T\zeta_i] \tag{D.3}$$
$$= \frac{1}{2}\sum_{i=1}^{N} e_i^T e_i \tag{D.4}$$
$$= \frac{1}{2}\sum_{i=1}^{N} \|e_i(t)\|^2$$

By using (C.7), one has

$$\sqrt{2V_1} \leq \alpha e^{-\frac{\lambda}{\gamma}t} + \frac{\beta\delta_0}{\lambda}, \qquad \forall t \geq 0 \tag{D.5}$$
$$V_1 \leq \frac{1}{2}(\alpha e^{-\frac{\lambda}{\gamma}t} + \frac{\beta\delta_0}{\lambda})^2 \tag{D.6}$$

Moreover, the time derivative of V_1 is given as:

$$\dot{V}_1 = \sum_{i=1}^{N}[\xi_i^T\dot{\xi}_i + \zeta_i^T\dot{\zeta}_i]$$
$$= \sum_{i=1}^{N}\left[\xi_i^T\dot{\xi}_i + \zeta_i^T\left(u_i^f - u_0\right)\right] \tag{D.7}$$

Step 2

Let us now consider system (2.9), (2.10) with collision-free formation tracking controller (3.15) and define the following Lyapunov function

$$V_2 = \frac{1}{2}\sum_{i=1}^{N}[\xi_i^T\xi_i + \zeta_i^T\zeta_i] + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} q_{ij}(r_i, r_j) + \sum_{i=1}^{N} q_{i0}(r_i, r_0)$$

The time derivative of V_2 along the trajectories of system (2.9) is

$$\begin{aligned}\dot{V}_2 &= \sum_{i=1}^{N}\left[\xi_i^T\dot{\xi}_i+\zeta_i^T\dot{\zeta}_i\right]+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial q_{ij}}{\partial r_i}v_i+\frac{\partial q_{ij}}{\partial r_j}v_j\right)\\ &\quad+\sum_{i=1}^{N}\left(\frac{\partial q_{i0}}{\partial r_i}v_i+\frac{\partial q_{i0}}{\partial r_0}v_0\right)\\ &= \sum_{i=1}^{N}\left[\xi_i^T\dot{\xi}_i+\zeta_i^T\left(u_i^f+u_i^r-u_0\right)\right]+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial q_{ij}}{\partial r_i}v_i+\frac{\partial q_{ij}}{\partial r_j}v_j\right)\end{aligned} \tag{D.8}$$

or

$$\begin{aligned}\dot{V}_2 &= \sum_{i=1}^{N}\left[\xi_i^T\dot{\xi}_i+\zeta_i^T\left(u_i^f-u_0\right)\right]+\sum_{i=1}^{N}\zeta_i^Tu_i^r+\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{\partial q_{ij}}{\partial r_i}v_i+\frac{\partial q_{ij}}{\partial r_j}v_j\right)\\ &\quad+\sum_{i=1}^{N}\left(\frac{\partial q_{i0}}{\partial r_i}v_i+\frac{\partial q_{i0}}{\partial r_0}v_0\right)\\ &= \dot{V}_1-\sum_{i=1}^{N}(v_i-v_0)^T\sum_{j=o}^{N}\frac{\partial q_{ij}^T}{\partial r_i}+\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i+\sum_{i=1}^{N}\frac{\partial q_{i0}}{\partial r_i}(v_i-v_0)\end{aligned} \tag{D.9}$$

By applying Lemma 41, one obtains

$$\begin{aligned}\dot{V}_2 &= \dot{V}_1-\sum_{i=1}^{N}\sum_{j=1}^{N}v_i^T\frac{\partial q_{ij}^T}{\partial r_i}+v_0^T\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}^T}{\partial r_i}-\sum_{i=1}^{N}(v_i-v_0)^T\frac{\partial q_{i0}^T}{\partial r_i}\\ &\quad+\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}v_i+\sum_{i=1}^{N}\frac{\partial q_{i0}}{\partial r_i}(v_i-v_0)\\ &= \dot{V}_1+v_0^T\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}^T}{\partial r_i}\end{aligned} \tag{D.10}$$

Furthermore, since $\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{\partial q_{ij}}{\partial r_i}=0$, it leads to

$$\dot{V}_2=\dot{V}_1 \tag{D.11}$$

It implies that if the initial positions of the agents satisfy Assumption 40, i.e. $\frac{\partial q_{ij}}{\partial r_i}=0$ at $t=0$ for $i=1\dots N$, $j=0\dots N$, consequently $V_1(0)=V_2(0)$ then

$$V_2=V_1\leq\frac{1}{2}(\alpha e^{-\frac{\lambda}{\gamma}t}+\frac{\beta\delta_0}{\lambda})^2,\qquad\forall t\geq 0$$

Hence, the MAS achieves the desired formation with controller (3.15) in practical sense. Also, from the structure of (3.12) and (3.14), one has

$$\begin{aligned}\lim_{\|r_{ij}\|\to r}q_{ij} &= \infty\\ \lim_{\|r_{ij}\|\to r}\frac{\partial q_{ij}}{\partial r_i} &= \infty\end{aligned}$$

for $\forall i\neq j$. Therefore, it can be concluded that the collision is avoided. □

Bibliography

[1] Sabato Manfredi and Edmondo Di Tucci. Decentralized control algorithm for fast monitoring and efficient energy consumption in energy harvesting wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 13(4):1513–1520, 2017.

[2] Chio-Zong Frank Cheng, Ying-Hwa Kuo, Richard A Anthes, and Lance Wu. Satellite constellation monitors global and space weather. *Eos, Transactions American Geophysical Union*, 87(17):166–166, 2006.

[3] Pengcheng Yang, Yanghong Xia, Miao Yu, Wei Wei, and Yonggang Peng. A decentralized coordination control method for parallel bidirectional power converters in a hybrid ac–dc microgrid. *IEEE Transactions on Industrial Electronics*, 65(8):6217–6228, 2018.

[4] Eric Boutin, Jaliya Ekanayake, Wei Lin, Bing Shi, Jingren Zhou, Zhengping Qian, Ming Wu, and Lidong Zhou. Apollo: Scalable and coordinated scheduling for cloud-scale computing. In *OSDI*, volume 14, pages 285–300, 2014.

[5] Anoop Jain and Debasish Ghose. Synchronization of multi-agent systems with heterogeneous controllers. *Nonlinear Dynamics*, 89(2):1433–1451, 2017.

[6] Liangming Chen, Yanning Guo, Chuanjiang Li, and Jing Huang. Satellite formation-containment flying control with collision avoidance. *Journal of Aerospace Information Systems*, pages 1–18, 2018.

[7] Francesco Bullo, Jorge Cortes, and Sonia Martinez. *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*, volume 27. Princeton University Press, 2009.

[8] Judith Ebegbulem and Martin Guay. Distributed control of multi-agent systems over unknown communication networks using extremum seeking. *Journal of Process Control*, 59:37–48, 2017.

[9] Judith Ebegbulem and Martin Guay. Distributed control of multi-agent systems over unknown communication networks using extremum seeking. *Journal of Process Control*, 59:37–48, 2017.

[10] Chris Godsil and Gordon F Royle. *Algebraic graph theory*, volume 207. Springer Science & Business Media, 2013.

[11] Norman Biggs, Norman Linstead Biggs, and Biggs Norman. *Algebraic graph theory*, volume 67. Cambridge university press, 1993.

[12] Reinhard Diestel. *Graph theory {graduate texts in mathematics; 173}*. Springer-Verlag Berlin and Heidelberg GmbH & amp, 2000.

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

[14] Rafig Agaev and Pavel Chebotarev. On the spectra of nonsymmetric laplacian matrices. *arXiv preprint math/0508176*, 399:157–178, 2005.

[15] Reza Olfati-Saber and Richard M Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on automatic control*, 49(9):1520–1533, 2004.

[16] Ali Jadbabaie, Jie Lin, and A Stephen Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on automatic control*, 48(6):988–1001, 2003.

[17] Zhiyun Lin, Mireille Broucke, and Bruce Francis. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on automatic control*, 49(4):622–629, 2004.

[18] Wei Ren and Randal W Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on automatic control*, 50(5):655–661, 2005.

[19] Ulrich Munz, Antonis Papachristodoulou, and Frank Allgower. Consensus in multi-agent systems with coupling delays and switching topology. *IEEE Transactions on Automatic Control*, 56(12):2976–2982, 2011.

[20] Pierre-Alexandre Bliman and Giancarlo Ferrari-Trecate. Average consensus problems in networks of agents with delayed communications. *Automatica*, 44(8):1985–1995, 2008.

[21] Sergey Parsegov, Andrey Polyakov, and Pavel Shcherbakov. Fixed-time consensus algorithm for multi-agent systems with integrator dynamics. In *4th IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pages 110–115, 2013.

[22] Zongyu Zuo and Lin Tie. A new class of finite-time nonlinear consensus protocols for multi-agent systems. *International Journal of Control*, 87(2):363–370, 2014.

[23] Jie Mei, Wei Ren, and Jie Chen. Consensus of second-order heterogeneous multi-agent systems under a directed graph. In *2014 American Control Conference*, pages 802–807. IEEE, 2014.

[24] Zhenhong Guo, Chunjing Jiang, Jie Mei, and Guangfu Ma. Fully distributed consensus for second-order uncertain multi-agent systems under a directed graph. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 492–496. IEEE, 2018.

[25] Wei Ren and Ella Atkins. Distributed multi-vehicle coordinated control via local information exchange. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 17(10-11):1002–1033, 2007.

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

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

[28] Jiandong Zhu, Yu-Ping Tian, and Jing Kuang. On the general consensus protocol of multi-agent systems with double-integrator dynamics. *Linear Algebra and its Applications*, 431(5-7):701–715, 2009.

[29] Wenwu Yu, Guanrong Chen, and Ming Cao. Some necessary and sufficient conditions for second-order consensus in multi-agent dynamical systems. *Automatica*, 46(6):1089–1095, 2010.

[30] Dongjun Lee and Mark W Spong. Stable flocking of multiple inertial agents on balanced graphs. *IEEE transactions on automatic control*, 52(8):1469–1475, 2007.

[31] Wei Li and Mark W Spong. Stability of general coupled inertial agents. *IEEE Transactions on Automatic Control*, 55(6):1411–1416, 2010.

[32] Jiahu Qin, Huijun Gao, and Wei Xing Zheng. Second-order consensus for multi-agent systems with switching topology and communication delay. *Systems & Control Letters*, 60(6):390–397, 2011.

[33] Wei Liu, Shaolei Zhou, Yahui Qi, and Xiuzhen Wu. Leaderless consensus of multi-agent systems with lipschitz nonlinear dynamics and switching topologies. *Neurocomputing*, 173:1322–1329, 2016.

[34] Wenwu Yu, He Wang, Fei Cheng, Xinghuo Yu, and Guanghui Wen. Second-order consensus in multiagent systems via distributed sliding mode control. *IEEE transactions on cybernetics*, 47(8):1872–1881, 2016.

[35] Wencheng Zou, Peng Shi, Zhengrong Xiang, and Yan Shi. Finite-time consensus of second-order switched nonlinear multi-agent systems. *IEEE transactions on neural networks and learning systems*, 2019.

[36] Junjie Fu, Guanghui Wen, Wenwu Yu, and Zhengtao Ding. Finite-time consensus for second-order multi-agent systems with input saturation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(11):1758–1762, 2017.

[37] Zongyu Zuo. Nonsingular fixed-time consensus tracking for second-order multi-agent networks. *Automatica*, 54:305–309, 2015.

[38] Xiaoyan He and Qingyun Wang. Distributed finite-time leaderless consensus control for double-integrator multi-agent systems with external disturbances. *Applied Mathematics and Computation*, 295:65–76, 2017.

[39] Shize Su and Zongli Lin. Distributed consensus control of multi-agent systems with higher order agent dynamics and dynamically changing directed interaction topologies. *IEEE Transactions on Automatic Control*, 61(2):515–519, 2015.

[40] Zongyu Zuo, Bailing Tian, Michael Defoort, and Zhengtao Ding. Fixed-time consensus tracking for multiagent systems with high-order integrator dynamics. *IEEE Transactions on Automatic Control*, 63(2):563–570, 2017.

[41] Guipu Li, Xiangyu Wang, and Shihua Li. Distributed composite output consensus protocols of higher-order multi-agent systems subject to mismatched disturbances. *IET Control Theory & Applications*, 11(8):1162–1172, 2017.

[42] Guipu Li, Xiangyu Wang, and Shihua Li. Consensus control of higher-order lipschitz non-linear multi-agent systems based on backstepping method. *IET Control Theory & Applications*, 2019.

[43] Wei Ren and Randal W Beard. *Distributed consensus in multi-vehicle cooperative control*. Springer, 2008.

[44] Wei Ren. Multi-vehicle consensus with a time-varying reference state. *Systems & Control Letters*, 56(7-8):474–483, 2007.

[45] Yiguang Hong, Jiangping Hu, and Linxin Gao. Tracking control for multi-agent consensus with an active leader and variable topology. *Automatica*, 42(7):1177–1182, 2006.

[46] Yiguang Hong, Guanrong Chen, and Linda Bushnell. Distributed observers design for leader-following control of multi-agent networks. *Automatica*, 44(3):846–850, 2008.

[47] Yongcan Cao and Wei Ren. Distributed coordinated tracking with reduced interaction via a variable structure approach. *IEEE Transactions on Automatic Control*, 57(1):33–48, 2011.

[48] Wei Zhu and Daizhan Cheng. Leader-following consensus of second-order agents with multiple time-varying delays. *Automatica*, 46(12):1994–1999, 2010.

[49] Wei Ni and Daizhan Cheng. Leader-following consensus of multi-agent systems under fixed and switching topologies. *Systems & Control Letters*, 59(3-4):209–217, 2010.

[50] Jiahe Jiang and Yangyang Jiang. Leader-following consensus of linear time-varying multi-agent systems under fixed and switching topologies. *Automatica*, 113:108804, 2020.

[51] Bailing Tian, Zongyu Zuo, and Hong Wang. Leader–follower fixed-time consensus of multi-agent systems with high-order integrator dynamics. *International Journal of Control*, 90(7):1420–1427, 2017.

[52] Yilun Shang and Yamei Ye. Leader-follower fixed-time group consensus control of multiagent systems under directed topology. *Complexity*, 2017, 2017.

[53] Alireza Khanzadeh and Mahdi Pourgholi. Fixed-time leader–follower consensus tracking of second-order multi-agent systems with bounded input uncertainties using non-singular terminal sliding mode technique. *IET Control Theory & Applications*, 12(5):679–686, 2017.

[54] Guoxing Wen, CL Philip Chen, Yan-Jun Liu, and Zhi Liu. Neural network-based adaptive leader-following consensus control for a class of nonlinear multiagent state-delay systems. *IEEE transactions on cybernetics*, 47(8):2151–2160, 2016.

[55] Chang-Chun Hua, Xiu You, and Xin-Ping Guan. Leader-following consensus for a class of high-order nonlinear multi-agent systems. *Automatica*, 73:138–144, 2016.

[56] Changchun Hua, Yafeng Li, and Xinping Guan. Leader-following consensus for high-order nonlinear stochastic multiagent systems. *IEEE transactions on cybernetics*, 47(8):1882–1891, 2017.

[57] Chang-Chun Hua, Kuo Li, and Xin-Ping Guan. Leader-following output consensus for high-order nonlinear multiagent systems. *IEEE Transactions on Automatic Control*, 64(3):1156–1161, 2018.

[58] Suiyang Khoo, Lihua Xie, and Zhihong Man. Leader-follower consensus control of a class of nonholonomic systems. In *2010 11th International Conference on Control Automation Robotics & Vision*, pages 1381–1386. IEEE, 2010.

[59] Michael Defoort, Guillaume Demesure, Zongyu Zuo, Andrey Polyakov, and Mohamed Djemai. Fixed-time stabilisation and consensus of non-holonomic systems. *IET Control Theory & Applications*, 10(18):2497–2505, 2016.

[60] Boda Ning and Qing-Long Han. Prescribed finite-time consensus tracking for multiagent systems with nonholonomic chained-form dynamics. *IEEE Transactions on Automatic Control*, 64(4):1686–1693, 2018.

[61] Jiankui Wang, Teng Gao, and Guoshan Zhang. Finite-time leader-following consensus for multiple non-holonomic agents. In *Proceedings of the 33rd Chinese Control Conference*, pages 1580–1585. IEEE, 2014.

[62] Zhicheng Hou and Isabelle Fantoni. Interactive leader–follower consensus of multiple quadrotors based on composite nonlinear feedback control. *IEEE Transactions on Control Systems Technology*, 26(5):1732–1743, 2017.

[63] Wei Ren and Yongcan Cao. *Distributed coordination of multi-agent networks: emergent problems, models, and issues*. Springer Science & Business Media, 2010.

[64] Kwang-Kyo Oh, Myoung-Chul Park, and Hyo-Sung Ahn. A survey of multi-agent formation control. *Automatica*, 53:424–440, 2015.

[65] Wenjie Dong and Jay A Farrell. Cooperative control of multiple nonholonomic mobile agents. *IEEE Transactions on Automatic Control*, 53(6):1434–1448, 2008.

[66] J Alexander Fax and Richard M Murray. Information flow and cooperative control of vehicle formations. *IEEE transactions on automatic control*, 49(9):1465–1476, 2004.

[67] KD Do and Jie Pan. Nonlinear formation control of unicycle-type mobile robots. *Robotics and Autonomous Systems*, 55(3):191–204, 2007.

[68] Sung-Mo Kang, Myoung-Chul Park, Byung-Hun Lee, and Hyo-Sung Ahn. Distance-based formation control with a single moving leader. In *2014 American Control Conference*, pages 305–310. IEEE, 2014.

[69] Jianhua Wu and Mingshun Qi. Research on formation handling of serial swarm robots. In *2012 Third Global Congress on Intelligent Systems*, pages 338–341. IEEE, 2012.

[70] Xiaoyu Cai and Marcio De Queiroz. Adaptive rigidity-based formation control for multirobotic vehicles with dynamics. *IEEE Transactions on Control Systems Technology*, 23(1):389–396, 2014.

[71] Teddy M Cheng and Andrey V Savkin. Decentralized control of multi-agent systems for swarming with a given geometric pattern. *Computers & Mathematics with Applications*, 61(4):731–744, 2011.

[72] Arindam Mondal, Laxmidhar Behera, Soumya Ranjan Sahoo, and Anupam Shukla. A novel multi-agent formation control law with collision avoidance. *IEEE/CAA Journal of Automatica Sinica*, 4(3):558–568, 2017.

[73] Viet Hoang Pham, Minh Hoang Trinh, and Hyo-Sung Ahn. Distance-based directed formation control in three-dimensional space. In *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 886–891. IEEE, 2017.

[74] Laura Krick, Mireille E Broucke, and Bruce A Francis. Stabilisation of infinitesimally rigid formations of multi-robot networks. *International Journal of Control*, 82(3):423–439, 2009.

[75] Dimos V Dimarogonas and Karl H Johansson. On the stability of distance-based formation control. In *2008 47th IEEE Conference on Decision and Control*, pages 1200–1205. IEEE, 2008.

[76] PKC Wang and Fred Y Hadaegh. Coordination and control of multiple microspacecraft moving in formation. *Journal of Astronautical Science*, 44(3):315–355, 1996.

[77] Ben Yun, Ben M Chen, Kai Yew Lum, and Tong H Lee. Design and implementation of a leader-follower cooperative control system for unmanned helicopters. *Journal of Control Theory and Applications*, 8(1):61–68, 2010.

[78] Zhou Chao, Shao-Lei Zhou, Lei Ming, and Wen-Guang Zhang. Uav formation flight based on nonlinear model predictive control. *Mathematical Problems in Engineering*, 2012, 2012.

[79] Shude He, Min Wang, Shi-Lu Dai, and Fei Luo. Leader–follower formation control of usvs with prescribed performance and collision avoidance. *IEEE Transactions on Industrial Informatics*, 15(1):572–581, 2018.

[80] Chengzhi Yuan, Haibo He, and Cong Wang. Cooperative deterministic learning-based formation control for a group of nonlinear uncertain mechanical systems. *IEEE Transactions on Industrial Informatics*, 15(1):319–333, 2018.

[81] Magnus B Egerstedt and Xiaoming Hu. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951, 2001.

[82] J Shao, G Xie, and L Wang. Leader-following formation control of multiple mobile vehicles. *IET Control Theory & Applications*, 1(2):545–552, 2007.

[83] M Anthony Lewis and Kar-Han Tan. High precision formation control of mobile robots using virtual structures. *Autonomous robots*, 4(4):387–403, 1997.

[84] Wei Ren and Randal W Beard. Decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control, and Dynamics*, 27(1):73–82, 2004.

[85] Norman HM Li and Hugh HT Liu. Formation uav flight control using virtual structure and motion synchronization. In *2008 American Control Conference*, pages 1782–1787. IEEE, 2008.

[86] Yang Qingkai, Cao Ming, Fang Hao, Chen Jie, and Huang Jie. Distributed formation stabilization for mobile agents using virtual tensegrity structures. In *2015 34th Chinese Control Conference (CCC)*, pages 447–452. IEEE, 2015.

[87] Qi Qin, Tie-Shan Li, Cheng Liu, CL Philip Chen, and Min Han. Virtual structure formation control via sliding mode control and neural networks. In *International Symposium on Neural Networks*, pages 101–108. Springer, 2017.

[88] Khac Duc Do. Formation control of multiple elliptical agents with limited sensing ranges. *Automatica*, 48(7):1330–1338, 2012.

[89] Tucker Balch and Ronald C Arkin. Behavior-based formation control for multirobot teams. *IEEE transactions on robotics and automation*, 14(6):926–939, 1998.

[90] Sreeja Nag and Leopold Summerer. Behaviour based, autonomous and distributed scatter manoeuvres for satellite swarms. *Acta astronautica*, 82(1):95–109, 2013.

[91] Giroung Lee and Dongkyoung Chwa. Decentralized behavior-based formation control of multiple robots considering obstacle avoidance. *Intelligent Service Robotics*, 11(1):127–138, 2018.

[92] Ismail Bayezit and Barış Fidan. Distributed cohesive motion control of flight vehicle formations. *IEEE Transactions on Industrial Electronics*, 60(12):5763–5772, 2012.

[93] Wei Ren. Consensus strategies for cooperative control of vehicle formations. *IET Control Theory & Applications*, 1(2):505–512, 2007.

[94] Xiwang Dong, Yan Zhou, Zhang Ren, and Yisheng Zhong. Time-varying formation tracking for second-order multi-agent systems subjected to switching topologies with application to quadrotor formation flying. *IEEE Transactions on Industrial Electronics*, 64(6):5014–5024, 2016.

[95] Travis Alan Dierks and Jagannathan Sarangapani. Neural network control of mobile robot formations using rise feedback. *IEEE Transactions on System, Man and Cybernetics: Part B*, 39(2):332–347, 2009.

[96] Joongbo Seo, Youdan Kim, Seungkeun Kim, and Antonios Tsourdos. Consensus-based reconfigurable controller design for unmanned aerial vehicle formation flight. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 226(7):817–829, 2012.

[97] Xiwang Dong, Bocheng Yu, Zongying Shi, and Yisheng Zhong. Time-varying formation control for unmanned aerial vehicles: Theories and applications. *IEEE Transactions on Control Systems Technology*, 23(1):340–348, 2014.

[98] Jing Liu, Jian-an Fang, Zhen Li, and Guang He. Time-varying formation tracking for second-order multi-agent systems subjected to switching topology and input saturation. *International Journal of Control, Automation and Systems*, pages 1–11, 2019.

[99] Runsha Dong and Zhiyong Geng. Consensus based formation control laws for systems on lie groups. *Systems & Control Letters*, 62(2):104–111, 2013.

[100] Zhaoxia Peng, Guoguang Wen, Ahmed Rahmani, and Yongguang Yu. Distributed consensus-based formation control for multiple nonholonomic mobile robots with a specified reference trajectory. *International Journal of Systems Science*, 46(8):1447–1457, 2015.

[101] Shri Harish Manoharan and Wei-Yu Chiu. Consensus based formation control of automated guided vehicles using dynamic destination approach. In *2019 58th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 902–907. IEEE, 2019.

[102] Jing Wang, Morrison Obeng, Tianyu Yang, Gennady Staskevich, and Brian Abbe. Formation control of multiple nonholonomic mobile robots with limited information of a desired trajectory. In *IEEE international conference on Electro/Information Technology*, pages 550–555. IEEE, 2014.

[103] Xiaohua Ge and Qing-Long Han. A brief survey of recent advances in consensus of sampled-data multi-agent systems. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 6758–6763. IEEE, 2016.

[104] Huan Pan and Wenjuan Qiao. Consensus of double-integrator discrete-time multi-agent system based on second-order neighbors' information. In *Control and Decision Conference (2014 CCDC), The 26th Chinese*, pages 1946–1951. IEEE, 2014.

[105] Yongcan Cao and Wei Ren. Multi-vehicle coordination for double-integrator dynamics under fixed undirected/directed interaction in a sampled-data setting. *International Journal of Robust and Nonlinear Control*, 20(9):987–1000, 2010.

[106] Yongcan Cao and Wei Ren. Sampled-data discrete-time coordination algorithms for double-integrator dynamics under dynamic directed interaction. *International Journal of Control*, 83(3):506–515, 2010.

[107] David W Casbeer, Randy Beard, and A Lee Swindlehurst. Discrete double integrator consensus. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 2264–2269. IEEE, 2008.

[108] Guodong Wang, Xiangyu Wang, and Shihua Li. Sliding-mode consensus algorithms for disturbed second-order multi-agent systems. *Journal of the Franklin Institute*, 355(15):7443–7465, 2018.

[109] Jiahu Qin and Huijun Gao. A sufficient condition for convergence of sampled-data consensus for double-integrator dynamics with nonuniform and time-varying communication delays. *IEEE Transactions on Automatic Control*, 57(9):2417–2422, 2012.

[110] Annika Eichler and Herbert Werner. Optimal convergence speed of consensus under constrained damping for multi-agent systems with discrete-time double-integrator dynamics. *Systems & Control Letters*, 108:48–55, 2017.

[111] Dongmei Xie and Yongli Cheng. Bounded consensus tracking for sampled-data second-order multi-agent systems with fixed and markovian switching topology. *International Journal of Robust and Nonlinear Control*, 25(2):252–268, 2015.

[112] Zhihai Wu, Li Peng, Linbo Xie, and Jiwei Wen. Stochastic bounded consensus tracking of leader–follower multi-agent systems with measurement noises based on sampled-data with small sampling delay. *Physica A: Statistical Mechanics and its Applications*, 392(4):918–928, 2013.

[113] Cheng-Lin Liu, Shuai Liu, Ya Zhang, and Yang-Yang Chen. Consensus seeking of multi-agent systems with intermittent communication: a persistent-hold control strategy. *International Journal of Control*, pages 1–7, 2018.

[114] Dimos V Dimarogonas, Emilio Frazzoli, and Karl H Johansson. Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 57(5):1291–1297, 2012.

[115] Lulu Li, Daniel WC Ho, and Shengyuan Xu. A distributed event-triggered scheme for discrete-time multi-agent consensus with communication delays. *IET Control Theory & Applications*, 8(10):830–837, 2014.

[116] Hao Zhang, Gang Feng, Huaicheng Yan, and Qijun Chen. Observer-based output feedback event-triggered control for consensus of multi-agent systems. *IEEE Trans. Industrial Electronics*, 61(9):4885–4894, 2014.

[117] Derui Ding, Zidong Wang, Daniel WC Ho, and Guoliang Wei. Observer-based event-triggering consensus control for multiagent systems with lossy sensors and cyber-attacks. *IEEE transactions on cybernetics*, 47(8):1936–1947, 2017.

[118] Jiangping Hu, Ji Geng, and Hong Zhu. An observer-based consensus tracking control and application to event-triggered tracking. *Communications in Nonlinear Science and Numerical Simulation*, 20(2):559–570, 2015.

[119] Wei Zhu, Zhong-Ping Jiang, and Gang Feng. Event-based consensus of multi-agent systems with general linear models. *Automatica*, 50(2):552–558, 2014.

[120] Wei Zhu, Huizhu Pu, Dandan Wang, and Huaqing Li. Event-based consensus of second-order multi-agent systems with discrete time. *Automatica*, 79:78–83, 2017.

[121] Rajiv Kumar Mishra and Abhinav Sinha. Event-triggered sliding mode based consensus tracking in second order heterogeneous nonlinear multi-agent systems. *European Journal of Control*, 45:30–44, 2019.

[122] Bin Hu, Zhi-Hong Guan, and Minyue Fu. Distributed event-driven control for finite-time consensus. *Automatica*, 103:88–95, 2019.

[123] Wei Ni and Daizhan Cheng. Leader-following consensus of multi-agent systems under fixed and switching topologies. *Systems & Control Letters*, 59(3-4):209–217, 2010.

[124] Ping Gong, Qing-Long Han, and Weiyao Lan. Finite-time consensus tracking for incommensurate fractional-order nonlinear multiagent systems with directed switching topologies. *IEEE Transactions on Cybernetics*, 2020.

[125] David Luenberger. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, 11(2):190–197, 1966.

[126] John Doyle and Guter Stein. Robustness with observers. *IEEE transactions on automatic control*, 24(4):607–611, 1979.

[127] Farzad Esfandiari and Hassan K Khalil. Output feedback stabilization of fully linearizable systems. *International Journal of Control*, 56(5):1007–1037, 1992.

[128] JP Gautheir, H Hammouri, and S Othman. A simple observer for nonlinear systems application to bioreactors. *IEEE Transactions on Automatic Control*, 37(6):875–880, 1992.

[129] Hassan K Khalil and Laurent Praly. High-gain observers in nonlinear feedback control. *International Journal of Robust and Nonlinear Control*, 24(6):993–1015, 2014.

[130] Hassan K Khalil and Jessy W Grizzle. *Nonlinear systems*, volume 3. Prentice hall Upper Saddle River, NJ, 2002.

[131] F Deza, E Busvelle, JP Gauthier, and D Rakotopara. High gain estimation for nonlinear systems. *Systems & control letters*, 18(4):295–299, 1992.

[132] Hassan Hammouri, Madiha Nadri, and Rafael Mota. Constant gain observer for continuous-discrete time uniformly observable systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5406–5411. IEEE, 2006.

[133] C-M Astorga, N Othman, Sami Othman, Hassan Hammouri, and T-F McKenna. Nonlinear continuous–discrete observers: application to emulsion polymerization reactors. *Control Engineering Practice*, 10(1):3–13, 2002.

[134] Tobias Raff, Markus Kogel, and Frank Allgower. Observer with sample-and-hold updating for lipschitz nonlinear systems with nonuniformly sampled measurements. In *2008 American Control Conference*, pages 5254–5257. IEEE, 2008.

[135] Iasson Karafyllis and Costas Kravaris. From continuous-time design to sampled-data design of observers. *IEEE Transactions on Automatic Control*, 54(9):2169–2174, 2009.

[136] Mondher Farza, Mohammed M'Saad, Mamadou Lamine Fall, Eric Pigeon, Olivier Gehan, and Krishna Busawon. Continuous-discrete time observers for a class of mimo nonlinear systems. *IEEE Transactions on Automatic Control*, 59(4):1060–1065, 2013.

[137] Tarek Ahmed-Ali, Emilia Fridman, Fouad Giri, Laurent Burlion, and Françoise Lamnabhi-Lagarrigue. Using exponential time-varying gains for sampled-data stabilization and estimation. *Automatica*, 67:244–251, 2016.

[138] Mondher Farza, Ibtissem Bouraoui, Tomas Menard, R Ben Abdennour, and Mohammed M'Saad. Sampled output observer design for a class of nonlinear systems. In *2014 European Control Conference (ECC)*, pages 312–317. IEEE, 2014.

[139] Ibtissem Bouraoui, Mondher Farza, Tomas Ménard, Ridha Ben Abdennour, Mohammed MSaad, and Henda Mosrati. Observer design for a class of uncertain nonlinear systems with sampled outputsapplication to the estimation of kinetic rates in bioreactors. *Automatica*, 55:78–87, 2015.

[140] Long Cheng, Zeng-Guang Hou, Yingzi Lin, Min Tan, and Wenjun Zhang. Solving a modified consensus problem of linear multi-agent systems. *Automatica*, 47(10):2218–2223, 2011.

[141] Suiyang Khoo, Lihua Xie, Zhihong Man, and Shengkui Zhao. Observer-based robust finite-time cooperative consensus control for multi-agent networks. In *2009 4th IEEE Conference on Industrial Electronics and Applications*, pages 1883–1888. IEEE, 2009.

[142] Abdelkader Abdessameud and Abdelhamid Tayebi. On consensus algorithms for double-integrator dynamics without velocity measurements and with input constraints. *Systems & Control Letters*, 59(12):812–821, 2010.

[143] Jianqiang Hu, Jinde Cao, Jie Yu, and Tasawar Hayat. Consensus of nonlinear multi-agent systems with observer-based protocols. *Systems & Control Letters*, 72:71–79, 2014.

[144] Fuyi Qu, Shaocheng Tong, and Yongming Li. Observer-based adaptive fuzzy output constrained control for uncertain nonlinear multi-agent systems. *Information Sciences*, 467:446–463, 2018.

[145] Junjie Fu and Jinzhi Wang. Observer-based finite-time coordinated tracking for general linear multi-agent systems. *Automatica*, 66:231–237, 2016.

[146] Peng Shi and QK Shen. Observer-based leader-following consensus of uncertain nonlinear multi-agent systems. *International Journal of Robust and Nonlinear Control*, 27(17):3794–3811, 2017.

[147] Lixin Gao, Bingbing Xu, Junwei Li, and Hui Zhang. Distributed reduced-order observer-based approach to consensus problems for linear multi-agent systems. *IET Control Theory & Applications*, 9(5):784–792, 2015.

[148] Guanghui Wen, Wenwu Yu, Yuanqing Xia, Xinghuo Yu, and Jianqiang Hu. Distributed tracking of nonlinear multiagent systems under directed switching topology: An observer-based protocol. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(5):869–881, 2016.

[149] Miaomiao Wu, Hao Zhang, Zhuping Wang, and Huaicheng Yan. Output regulation of asynchronously switched multi-agent systems by adaptive observer-based control. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2825–2829. IEEE, 2017.

[150] Mengqi Xue, Yang Tang, Wei Ren, and Feng Qian. Practical output synchronization for asynchronously switched multi-agent systems with adaption to fast-switching perturbations. *Automatica*, 116:108917, 2020.

[151] Yuanqing Xia, Xitai Na, Zhongqi Sun, and Jing Chen. Formation control and collision avoidance for multi-agent systems based on position estimation. *ISA Transactions*, 61:287–296, 2016.

[152] Fangting Chen, Hui Yu, and Xiaohua Xia. Output consensus of multi-agent systems with delayed and sampled-data. *IET Control Theory & Applications*, 11(5):632–639, 2016.

[153] Sheng-Li Du, Weiguo Xia, Wei Ren, Xi-Ming Sun, and Wei Wang. Observer-based consensus for multiagent systems under stochastic sampling mechanism. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(12):2328–2338, 2017.

[154] Hao Zhang, Gang Feng, Huaicheng Yan, and Qijun Chen. Observer-based output feedback event-triggered control for consensus of multi-agent systems. *IEEE Transactions on Industrial Electronics*, 61(9):4885–4894, 2013.

[155] Lian-Na Zhao, Hong-Jun Ma, Lin-Xing Xu, and Xin Wang. Observer-based adaptive sampled-data event-triggered distributed control for multi-agent systems. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 67(1):97–101, 2019.

[156] Tomas Menard, Emmanuel Moulay, Patrick Coirault, and Michael Defoort. Observer-based consensus for second-order multi-agent systems with arbitrary asynchronous and aperiodic sampling periods. *Automatica*, 99:237–245, 2019.

[157] Syed Ali Ajwad, Tomas Menard, Emmanuel Moulay, Michael Defoort, and Patrick Coirault. Observer based leader-following consensus of second-order multi-agent systems with nonuniform sampled position data. *Journal of the Franklin Institute*, 356(16):10031–10057, 2019.

[158] Syed Ali Ajwad, Emmanuel Moulay, Michael Defoort, Tomas Ménard, and Patrick Coirault. Leader-following consensus of second-order multi-agent systems with switching topology and partial aperiodic sampled data. *IEEE Control Systems Letters*, [Submitted], 2020.

[159] Chuanrui Wang and Haibo Ji. Leader-following consensus of multi-agent systems under directed communication topology via distributed adaptive nonlinear protocol. *Systems & Control Letters*, 70:23–29, 2014.

[160] Wangli He, Guanrong Chen, Qing-Long Han, and Feng Qian. Network-based leader-following consensus of nonlinear multi-agent systems via distributed impulsive control. *Information Sciences*, 380:145–158, 2017.

[161] B Madhevan and M Sreekumar. Tracking algorithm using leader follower approach for multi robots. *Procedia Engineering*, 64:1426–1435, 2013.

[162] Wei Liu and Jie Huang. Leader-following consensus for uncertain second-order nonlinear multi-agent systems. *Control Theory and Technology*, 14(4):279–286, 2016.

[163] Jia Wu, Huaqing Li, and Xin Chen. Leader-following consensus of nonlinear discrete-time multi-agent systems with limited communication channel capacity. *Journal of the Franklin Institute*, 354(10):4179–4195, 2017.

[164] Abraham Berman and Robert J Plemmons. *Nonnegative matrices in the mathematical sciences*. SIAM, 1994.

[165] R.A. Horn and C.R. Johnson. *Matrix analysis*. Cambridge university press, 1990.

[166] A. Graham. *Kronecker Products and Matrix Calculus: With Applications (Mathematics and its Applications) PDF*. Courier Dover Publications, 1981.

[167] Hassan K Khalil. Noninear systems. *Prentice-Hall, New Jersey*, 2(5):5–1, 1996.

[168] Qiang Song, Fang Liu, Jinde Cao, and Wenwu Yu. Pinning-controllability analysis of complex networks: an m-matrix approach. *IEEE Trans. on Circuits and Systems*, 59(11):2692–2701, 2012.

[169] Mondher Farza, Mohammed M'Saad, Mamadou Lamine Fall, Eric Pigeon, Olivier Gehan, and Krishna Busawon. Continuous-discrete time observers for a class of mimo nonlinear systems. *IEEE Transactions on Automatic Control*, 59(4):1060–1065, 2014.

[170] Omar Hernández-González, Mondher Farza, Tomas Menard, Boubekeur Targui, Mohammed MSaad, and Carlos-M Astorga-Zaragoza. A cascade observer for a class of mimo non uniformly observable systems with delayed sampled outputs. *Systems & Control Letters*, 98:86–96, 2016.

[171] Joao P Hespanha and A Stephen Morse. Stability of switched systems with average dwell-time. In *Proceedings of the 38th IEEE conference on decision and control (Cat. No. 99CH36304)*, volume 3, pages 2655–2660. IEEE, 1999.

[172] Syed Ali Ajwad, Emmanuel Moulay, Michael Defoort, Tomas Ménard, and Patrick Coirault. Output-feedback formation tracking of second-order multi-agent systems with asynchronous variable sampled data. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4483–4488. IEEE, 2019.

[173] Syed Ali Ajwad, Emmanuel Moulay, Michael Defoort, Tomas Ménard, and Patrick Coirault. Collision-free formation tracking of multi-agent robotic networks under communication constraints. *IEEE Control Systems Letters*, [Submitted], 2020.

[174] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles*, pages 396–404. Springer, 1986.

[175] Jen-Hui Chuang and Narendra Ahuja. An analytically tractable potential field model of free space and its application in obstacle avoidance. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(5):729–736, 1998.

[176] Yeong-Hwa Chang, Chun-Lin Chen, Wei-Shou Chan, Hung-Wei Lin, and Chia-Wen Chang. Fuzzy formation control and collision avoidance for multiagent systems. *Mathematical Problems in Engineering*, 2013, 2013.

[177] Yongcan Cao and Wei Ren. Distributed coordinated tracking with reduced interaction via a variable structure approach. *IEEE Transactions on Automatic Control*, 57(1):33–48, 2012.

[178] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source robot operating system. In *ICRA workshop on open source software*. Kobe, Japan, 2009.

[179] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154. IEEE, 2004.

[180] Anis Koubâa. *Robot Operating System (ROS)*. Springer, 2017.

[181] L Sciavicco, B Siciliano, L Villani, and G Oriolo. *Robotics: Modelling, planning and Control, ser. Advanced Textbooks in Control and Signal Processing.* Berlin, Germany: Springer, 2011.

[182] Michel Fliess, Jean Lévine, Philippe Martin, and Pierre Rouchon. Flatness and defect of nonlinear systems: introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, 1995.

[183] Yingchong Ma, Gang Zheng, Wilfrid Perruquetti, and Zhaopeng Qiu. Control of nonholonomic wheeled mobile robots via i-pid controller. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4413–4418. IEEE, 2013.

[184] Animesh Chakravarthy and Debasish Ghose. Obstacle avoidance in a dynamic environment: A collision cone approach. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 28(5):562–574, 1998.

[185] Michael Defoort, Arnaud Doniec, and Noury Bouraqadi. Decentralized robust collision avoidance based on receding horizon planning and potential field for multi-robots systems. In *Informatics in Control Automation and Robotics*, pages 201–215. Springer, 2011.

[186] Tomas Ménard, Syed Ali Ajwad, Emmanuel Moulay, Patrick Coirault, and Michael Defoort. Leader-following consensus for multi-agent systems with nonlinear dynamics subject to additive bounded disturbances and asynchronously sampled outputs. *Automatica*, 121, November 2020.

[187] J.P. Gauthier, H. Hammouri, and S. Othman. A simple observer for nonlinear systems applications to bioreactors. *IEEE Transactions on automatic control*, 37(6):875–880, 1992.

[188] A. Bédoui, M. Farza, M. MSaad, and M. Ksouri. Robust nonlinear controllers for bioprocesses. *IFAC Proceedings Volumes*, 41(2):15541–15546, 2008.

[189] Kairui Chen, Junwei Wang, Yun Zhang, and Zhi Liu. Second-order consensus of nonlinear multi-agent systems with restricted switching topology and time delay. *Nonlinear Dynamics*, 78(2):881–887, 2014.