KTH ROYAL INSTITUTE
OF TECHNOLOGY

Doctoral Thesis in Electrical Engineering

Task-oriented control and coordination of multi-agent systems under varying constraints

PIAN YU

Stockholm, Sweden 2021

Task-oriented control and coordination of multi-agent systems under varying constraints

PIAN YU

Academic Dissertation which, with due permission of the KTH Royal Institute of Technology, is submitted for public defence for the Degree of Doctor of Philosophy on Thursday the 11th of February 2021, at 3:00 p.m. in Q2, Malvinas Väg 10, KTH Campus, Stockholm

Doctoral Thesis in Electrical Engineering
KTH Royal Institute of Technology
Stockholm, Sweden 2021

ISBN: 978-91-7873-733-8
TRITA-EECS-AVL-2021:2

Printed by: Universitetsservice US-AB, Sweden 2020

Abstract

Multi-agent systems (MAS) offer a tremendous potential to improve the quality of modern society life. For instance, robot networks have been widely used for providing services such as search and rescue missions, surveillance and data collection, healthcare and entertainment. One challenge for MAS is the design of control and coordination strategies that enable each agent to perform operations safely and efficiently while achieving group or individual motion objectives. This dissertation addresses this challenge by developing different task-oriented control and coordination strategies for MAS.

The first part of the thesis is devoted to the control of MAS under cooperative tasks. Firstly, we investigate the distributed control of multi-agent consensus. Event-triggered control strategies are developed to reduce communication burden. It is proven that consensus can be achieved with the proposed strategies. Next, we tackle the problem of leader-follower multi-agent target tracking, where the group of agents is assigned a set of dynamically activated tasks, each of which has to be completed within a deadline. A dynamic scheduling strategy is proposed and distributed control laws are designed respectively for the leader and follower agents. It is shown that the proposed control laws guarantee the satisfaction of each task.

In the second part of the thesis, we develop control and coordination schemes for single- or multi-agent systems under temporal logic specifications. Firstly, the symbolic control of continuous-time uncertain nonlinear systems is studied. A new stability notion called approximate controlled globally practically stable is introduced. Building on this notion, we provide for the first time a behavioral relationship between the original continuous-time system and its discrete state-space symbolic model. After that, we consider the robust satisfiability check and online control synthesis of uncertain systems under signal temporal logic specifications. A sufficient condition is obtained for the robust satisfiability check of the uncertain systems. An online control synthesis algorithm is designed, which is shown to be sound for uncertain systems and both sound and complete for deterministic systems. Finally, the motion coordination of MAS is investigated, where each agent is assigned a linear temporal logic specification. Based on the realistic assumptions that each agent is subject to both state and input constraints and can have only local view and local information, a provably safe and fully distributed multi-agent motion coordination strategy is proposed.

Sammanfattning

Multiagentsystem (MAS) utlovar en enorm potential att förbättra kvaliteten på det moderna samhällslivet. Till exempel har robotnätverk använts i stor utsträckning för att tillhandahålla tjänster som sök- och räddningsuppdrag, övervakning och datainsamling, vård och underhållning. En utmaning för MAS är utformningen av regler- och samordningsstrategier som gör det möjligt för varje agent att utföra operationer säkert och effektivt samtidigt som grupp- eller individuella rörelsemål uppnås. Denna avhandling behandlar denna utmaning genom att utveckla olika uppgiftsorienterade regler- och samordningsstrategier för MAS.

Den första delen av avhandlingen ägnas åt regleringen av MAS under samarbetsuppgifter. För det första undersöks den distribuerade regleringen av samarbete med flera agenter. Händelsestyrda reglerstrategier utvecklas för att uppnå kommunikationsreduktion. Det är bevisat att konsensus kan uppnås med de föreslagna strategierna. Därefter hanterar vi problemet med målföljning med flera agenter som är antingen ledare eller följare, där gruppen av agenter tilldelas en uppsättning dynamiskt aktiverade uppgifter, som var och en måste slutföras inom en tidsfrist. En dynamisk schemaläggningsstrategi föreslås. Därefter utformas distribuerade styrlagar för ledare och följare. Det visas att de föreslagna styrlagarna garanterar att varje uppgift slutförs.

I den andra delen av avhandlingen utvecklar vi regler- och samordningsmetoder för system med en eller flera agenter under tidslogiska specifikationer. För det första studeras den symboliska regleringen av olinjära system med osäkerheter i kontinuerlig tid. Ett nytt stabilitetsbegrepp som kallas approximativ reglerad globalt praktisk stabilitet introduceras. Baserat på dessa begrepp tillhandahåller vi för första gången ett beteendemässigt förhållande mellan det ursprungliga tidskontinuerliga systemet och dess diskreta symboliska tillståndsmodell. Därefter beaktar vi den robusta lösbarhetskontrollen och online-styrlagssyntesen av system med osäkerhet under signaltemporallogiska specifikationer. Ett tillräckligt villkor erhålls för en robust lösbarhetskontroll av de osäkra systemen. Dessutom utformas en online-styrlagssyntesalgoritm, som visas vara sund för osäkra system och både sund och komplett för deterministiska system. Slutligen undersöks rörelsekoordinering av MAS, där varje agent tilldelas en linjär temporallogikspecifikation. Baserat på de realistiska antagandena att varje agent lyder under både tillstånds- och insignalsbegränsningar och endast kan ha lokal vy, föreslås en säker och helt distribuerad multi-agent-strategi för samordning av rörelser.

To my mother and to my father.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor Prof Dimos Dimarogonas, for offering the unique opportunity to work at KTH. His continuous support and encouragement, insightful feedback, and close collaboration has made this thesis possible. I would also like to thank my co-supervisor, Prof Carlo Fischione, for his great assistance and guidance, especially at the beginning of my PhD career. It is really a great honor and pleasure to work with them.

I wish to express my gratitude to Prof Christos G. Cassandras from Boston University for being the opponent, and Prof Ming Cao from University of Groningen, Prof Jie Fu from Worcester Polytechnic Institute, and Prof Karl-Erik Årzén from Lund University for being the committee members. Many thanks to Prof Jana Tumova for acting as the advanced reviewer and the substitute. I am also grateful to Prof Xiaoming Hu for willing to be the substitute. Thank you Prof Henrik Sandberg for chairing the defence.

Sincere thanks to Prof Karl H. Johansson, Prof Lihua Xie, and Yulong Gao for the great collaborations. Thank you, Prof Alessandro Abate, for offering the opportunity to present my work in your group. The help of Erik Berglund for translating the thesis abstract into Swedish, and Wenceslao Shaw Cortez, Péter Várnai, Maryam Sharifi, Wei Ren, and Xiao Tan for proofreading this thesis has been highly appreciated.

Many thanks to all my colleagues (current and former) at the Division of Decision and Control Systems for creating such a friendly and active working atmosphere. Especially, I thank Lars Lindemann for being a great office partner and for co-organizing the reading group. I thank Andrea Bisoffi, Pouria Tajvar, Wenceslao Shaw Cortez, Xiao Tan, Péter Várnai, Luis Guerrero Bonilla, Maryam Sharifi, and Pushpak Jagtap for joining the reading group regularly and for all the research discussions. I thank Dimitris Boskos, Soulaimane Berkane, Pierre-Jean Meyer, Meng Guo, Dionysios Theodosis Palimeris, Shahab Heshmati-alamdari, Antonio Adaldo, Pedro Pereira, Alexandros Nikou, Wei Ren, Christos K. Verginis, Pedro Alexandre Delgado Roque, Fei Chen, Sofie Ahlberg, Maria Charitidou, Adrian Wiltz, Mayank Sewlia, Ami Sakakibara, and Robin Baren for the exciting group meetings. I want to further extend my thanks to David Umsonst, Rui Oliveira, Ehsan Nekouei, Erik Berglund for being great office partners during my time in 6th floor; Junfeng Wu, Xiaoqiang Ren, Jieqiang Wei, Kuizhe Zhang, Song Fang, Wei Ren, Xinkang He, Xinlei Yi, Xiao Tan, Fei Chen, and Yuchao Li for the lunch time we had to-

gether; Joana Filipa Gouveia Fonseca, Mina Ferizbegovic, Ines De Miranda De Matos Lourenço, Matin Jafarian, Urszula Libal, Michelle Chong, Linnea Persson, Hanxiao Liu, and Malin Andersson for organizing and/or joining the Reglerteknik girl dinner. Many thanks also to all the professors and administrators (current and former) of the division for their assistance and support.

Last but not least, I would like to thank my parents and sisters, for always believing in me and for their unconditional love and constant support. A special thanks goes to my husband Yulong, who has been accompanying me and taking care of me for the past four years. I am so lucky to meet you in Stockholm!

Pian Yu
Stockholm, Sweden
December 2020

Contents

List of Abbreviations

MAS	Multi-Agent System(s)
ARE	Algebraic Riccati Equation
TTC	Time-Triggered Control
ETC	Event-Triggered Control
PETC	Periodic Event-Triggered Control
MASP	Maximum Allowable Sampling Period
QoSa	Quality-of-Satisfaction
EDF	Earliest Deadline First
FC	Forward Complete
δ-GAS	Incrementally Globally Asymptotically Stable
δ-GPS	Incrementally Globally Practically Stable
δ-ISS	Incrementally Input-to-State Stable
δ-FC	Incrementally Forward Complete
η-C-Ω-GPS	η-approximate Controlled Globally Practically Stable with respect to set Ω
CTS	Controlled Transition System
LTL	Linear Temporal Logic
NBA	Nondeterministic Büchi Automaton
PBA	Product Büchi Automaton
STL	Signal Temporal Logic
TLT	Temporal Logic Tree
tTLT	tube-based Temporal Logic Tree
MTF	maximal temporal fragment
MAMC	Multi-Agent Motion Coordination
TPP	Trajectory Planning Problem

List of Symbols

$\top$	true
$\bot$	false
$\mathbb{N}$	set of natural numbers
$\mathbb{Z}$	set of integers
$\mathbb{R}$	set of real numbers
$\mathbb{R}_{\geq 0}$	set of nonnegative real numbers
$\mathbb{R}_{>0}$	set of positive real numbers
$\mathbb{R}^n$	Euclidean space of dimension n
$\mathbb{R}^{n\times m}$	space of n-by-m real matrices
$n!$	the factorial of n
1_n	n-by-1 vector of all ones
I_n	n-by-n identity matrix
$\mathbf{0}$	matrix with proper dimensions and all elements equal to 0
$\lvert\lambda\rvert$	absolute value of a real number λ or modulus of a complex number λ
$\lfloor\lambda\rfloor$	floor of λ
$\mathrm{Re}(\lambda)$	the real part of a complex number λ
$\lVert\cdot\rVert$	Euclidean norm for vectors or induced 2-norm for matrices
$\lambda_{\max}(A)$	the largest eigenvalue of A; A has real eigenvalues
$\lambda_{\min}(A)$	the smallest eigenvalue of A; A has real eigenvalues
$\lambda_2(A)$	smallest positive eigenvalue of matrix A; A has nonnegative eigenvalues
A^T	transpose of real matrix A
x^T	transpose of real vector x
$(x_1, x_2, \ldots, x_m)$	$[x_1^T, x_2^T, \ldots, x_m^T]^T$
$M \succ 0$	M is a positive definite matrix
$A \otimes B$	Kronecker product of two matrices A and B

$\emptyset$	empty set
$\|S\|$	cardinality of a set S
2^S	set of all subsets of S
$\mathrm{int(S)}$	interior of S
$F_r(S)$	boundary of S
$\bar{S}$	complement of S
$S_1 \setminus S_2$	set difference of two sets S_1 and S_2
$\mathcal{F}(S_1, S_2)$	set of all measurable functions that map from S_1 to S_2
$\mathcal{B}(x, r)$	ball area centered at point $x \in \mathbb{R}^n$ and with radius $r > 0$
$\mathcal{B}(A, r)$	$\cup_{x \in A} \mathcal{B}(x, r)$
$\cup$	set union
$\cap$	set intersection
$\neg$	negation operator
$\wedge$	logical operator AND
$\vee$	logic operator OR
I	interval of the form $[a, b], [a, b), (a, b], (a, b), a \leq b$
$\mathrm{dom}(\cdot)$	domain of a function or signal
$\nabla_v f(\cdot)$	gradient of a function $f : \mathbb{R}^n \to \mathbb{R}^n$ with respect to $v \in \mathbb{R}^n$
$\|f\|_\infty$	$\sup\{\|f(t)\|, t \in \mathrm{dom}(f)\}$, supremum of a function $f : \mathbb{R} \to \mathbb{R}^n$
$\mathrm{dist}(x, S)$	$\inf_{y \in S}\{\|x - y\|\}, x \in \mathbb{R}^n, S \subseteq \mathbb{R}^n$, point-to-set distance
$\mathrm{dist}(S_1, S_2)$	$\inf_{x \in S_1, y \in S_2}\{\|x - y\|\}, S_1, S_2 \subseteq \mathbb{R}^n$, set-to-set distance
$\mathrm{proj}_m(x)$	projection of a vector $x \in \mathbb{R}^n$ or a signal $x : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ to its first $m, m \leq n$ components

List of Figures

Part I

Introduction and Preliminaries

Chapter 1

Introduction

A multi-agent system (MAS) is a collection of agents cooperating or competing with each other in order to fulfill common or individual goals [1]. It is a powerful model to describe a wide array of phenomena in nature, society, and technology. This thesis is concerned with the task-oriented control and coordination for MAS under communication constraints and/or state and input constraints. The task-oriented design means that the design of the control and the coordination strategy is dependent on the type of tasks, *e.g.,* cooperative or temporal logic, global or local, under consideration.

An agent in a MAS is defined as an autonomous entity that is endowed with sensing and actuation capabilities. The identification of individual agents depends on scenarios. Throughout this thesis, an agent can be exemplified as a (mobile) robot, which a robot is modeled as a dynamical system.

The rest of this chapter is organized as follows. Section 1.1 dedicates to discuss the motivations for the research work presented in this thesis. In Section 1.2, we formulate the research questions that are addressed in the thesis. Finally, Section 1.3 gives a detailed outline of the thesis and a list of publications that the thesis is based on.

1.1 Motivation

MAS offer a tremendous potential to improve the quality of modern society life. For instance, robot networks have been widely used for providing services such as search and rescue missions, surveillance and data collection, healthcare and entertainment. The deployment of connected automated cars have

the potential to improve vehicle safety and energy efficiency of transportation systems [2]. Having made great progress in the development of MAS in control, robotics, and computer science communities, many researchers have recently shifted their focus to multi-agent control under more complex objectives.

Cooperative control

Cooperative control is one of the most active research topics within the MAS. It is concerned with a collection of agents, usually with limited sensing and communication capabilities, all seeking to achieve a common objective. Due to the broad applications in various fields, for instance, attitude alignment of satellites [3], [4], robots formation and synchronization [5], [6], estimation over sensor networks [7], and distributed computing [8], cooperative control has gained great attention for the last decades.

Typical objectives of cooperative control include consensus [9], target tracking [10], and formation [11]. One example is the space interferometry mission [12], see Figure 1.1, where a fleet of networked satellites are required to perform a sequence of formation maneuvers while maintaining relative attitude accurately. The communication pattern for the multiple satellite system can be different. In [3], the formation keeping and attitude alignment for the multiple satellites are achieved by local information exchange with adjacent neighbors. In [13], a leader-follower approach is proposed, where each follower satellite tracks its leader's position and attitude.

Consensus is the benchmark problem of multi-agent cooperative control, in which agents are required to reach an agreement. At the early stage, control laws for multi-agent consensus require either continuous or high frequency interaction between neighboring agents, which is not resource efficient in terms of communication. This fact has motivated a recent interest in event-triggered control (ETC) methodology. Different from continuous-time and sampled-data control, where the states of each agent are transmitted continuously or at every sampling instant, the time instants in ETC (at which the states are transmitted) is determined by a pre-defined triggering condition [14]. In this way, a substantial reduction of communication rate can be achieved. Although the literature on ETC is rich, there are still some issues to be investigated. On one hand, most of the existing ETC strategies require the triggering condition to be monitored continuously, which results in excessive use of sensing and computational resources. On the other hand, ETC

Figure 1.1: A fleet of satellites. Source: https://aerospaceamerica.aiaa.org, Public domain.

for MAS is largely unexplored in the presence of communication constraint. In this thesis, we aim at addressing the above mentioned issues by designing resource-efficient communication and control strategies for MAS.

Leader-follower target tracking is more complex than consensus, in which the leaders (agents with target information) is responsible for tracking the target set/trajectory while the followers (agents without target information) track its leader's or adjacent neighbors' states. In this way, both target tracking and consensus/formation can be achieved by the MAS. Existing literature concerning leader-follower multi-agent control has usually focused on the design of control laws such that the objective is achieved in an asymptotical manner. However, in practical applications, it is reasonable to require a task being completed before a deadline. To the best of our knowledge, explicit time constraint is rarely considered in the context of MAS. In this thesis, we consider time-constrained leader-follower target tracking for MAS.

Control and coordination under temporal logic specifications

Another topic that has grown significantly in importance in recent years is the control and coordination of MAS such that each agent fulfills tasks given by high-level specifications expressed as temporal logic formulas. This is own to

the technical advancements in manufacturing, sensing, communication, and digital processing, which empower the agents in more efficient decision making and more accurate actuation. In addition, the rapid growth of robotic applications, *e.g.,* autonomous vehicles [15] and service robotics [16], stimulates the need for accomplishing more complex objectives such as nondeterministic, periodic, or sequential tasks. Temporal logics, such as linear temporal logic (LTL) [17] and signal temporal logic (STL) [18], have shown in the last decade capability in expressing such objectives for dynamical systems.

LTL focuses on the Boolean satisfaction of properties by given signals. It is expressive enough to capture many important properties, *e.g.,* safety (nothing bad will ever happen) and liveness (something good will eventually happen), and more complex combinations of Boolean and temporal statements [17]. Existing single-agent control approaches that use LTL mainly rely on a finite abstraction of the system dynamics and a language equivalent automata [19] representation of the LTL specification. The controller is synthesized by solving a game over the product automata [20]–[22]. In this process, discrete abstraction of the (infinite) continuous system plays a central role and the resulting controller guarantees the satisfaction of the LTL formula for the abstract finite system. However, to enforce the correctness of the controller for the (original) infinite system, an equivalence or inclusion relation has to be established between the abstract finite system and the infinite system.

STL is a more recently developed temporal logic, which allows the specification of properties over dense-time. Due to a number of advantages, such as explicitly treating real-valued signals [18], and admitting qualitative semantics [23], control synthesis under STL specifications has gained popularity in the last few years. Existing approaches that deal with the control synthesis under STL specifications include: barrier function methods [24], [25], optimization methods [26]–[28], sampling-based methods [29], and learning-based methods [30], [31]. We note that although various methods exist for the control synthesis under STL specifications, guaranteeing robustness under uncertainties is still a challenging problem.

In addition to single-agent control under temporal logic specifications, the design of motion coordination strategy for MAS that enables each agent to perform operations safely and efficiently in a shared workspace is also of great practical importance. One typical example is the intelligent transportation system, where automated vehicles that are capable of sensing and navigation are deployed, as seen in Figure 1.2. Each automated vehicle may be subject to a

local high-level task, *e.g.,* "first drive to A, then drive to B within 30 minutes, and eventually always stay in C", which can be expressed as a temporal logic formula. In addition, efficient coordination among them is required at, *e.g.,* traffic intersections, to ensure that the global task, *i.e.,* safety, is guaranteed.

Figure 1.2: Einride's self-driving truck tested in public road in Sweden. Picture: Einride

1.2 Problem Statement

This thesis is concerned with the control and coordination of MAS under cooperative or temporal logic, global or local specifications. Based on the specifications under consideration, different control laws or algorithms are proposed. In addition, various constraints, such as communication constraint, time constraint, and state and input constraints are taken into account. More specifically, we consider cooperative control tasks, such as consensus (Chapters 3, 4) and leader-follower target tracking (Chapter 5), and temporal logic specifications, such as LTL (Chapters 6, 8) and STL (Chapter 7).

The overall thesis problem is broken down in the following four research questions.

Communication-Efficient Multi-Agent Consensus

Both Chapters 3 and 4 are devoted to investigate the distributed control of multi-agent consensus. Different from existing literature, asynchronous (pe-

riodic) event-triggered communication and control strategies are proposed to answer the following question:

Q1. How to design a distributed, resource-efficient communication and control strategy such that consensus of the MAS is achieved?

Leader-Follower Target Tracking under Online Task Scheduling

Chapter 5 generalizes the cooperative control task to multi-agent target tracking. The leader-follower communication framework is considered and the MAS is subject to a sequence of deadline-constrained dynamically activated tasks. Due to the dynamically activated nature of the task set, an online task scheduling process is needed. In addition, the time constraint on the completion of each task poses new challenges for the distributed control synthesis. The question answered in this chapter is:

Q2. How to jointly schedule the dynamically activated tasks and design distributed controllers such that each task is completed?

Robust Control under Temporal Logic Specifications

In Chapters 6 and 7, we consider the robust control of uncertain systems under temporal logic specifications. As mentioned previously, although various approaches exist for the control synthesis under temporal logic specifications, guaranteeing robustness under uncertainties is still a challenging problem. In this thesis, we tackle this problem by developing new tools. The question to be addressed is:

Q3. Consider an uncertain system and a specification expressed as a temporal logic formula. How to do control synthesis such that the behaviors of the system satisfy the given specification under all possible uncertainties?

Distributed Motion Coordination under LTL Specifications

In Chapter 8, we extend the temporal logic control to the MAS. Consider a group of agents working in a shared workspace, each of which is assigned an LTL specification. We answer the following question:

Q4. How to coordinate, in a distributed manner, the motion of the agents such that the LTL specification of each gent is satisfied on the premise that safety (no collision with obstacles in the workspace and no inter-agent collision) is guaranteed?

1.3 Thesis Outline and Contributions

In Chapter 2, we introduce preliminaries that are used in the thesis. The topics include in this chapter are: graph theory, system properties, transition systems, and temporal logics. The results that are mentioned in Chapter 2 are then used directly in the rest of the thesis.

Chapter 3: Distributed Event-Triggered Consensus

Chapter 3 studies the distributed control of multi-agent consensus, where an asynchronous ETC strategy is designed for linear MAS. Firstly, an event-triggered communication scheme is designed for the communication between neighbors. Under such a communication scheme, a distributed event-triggered output feedback controller is further implemented for each agent. It is proven that the consensus is achieved asymptotically. In addition, it is shown that Zeno behavior is excluded.

The covered material is based on the following contribution.

- P. Yu, C. Fischione, and D. V. Dimarogonas, "Distributed event-triggered communication and control of linear multi-agent systems under tactile communication," IEEE Transactions on Automatic Control, vol. 63, no. 11, pp: 3979-3985, 2018.

Chapter 4: Periodic Event-Triggered Consensus under Limited Data Rate

Chapter 4 further addresses the distributed control of multi-agent consensus. In order to relax the requirement of continuous sensing and computation in ETC, a periodic event-triggered control (PETC) strategy is proposed. One of the main difficulties for implementing PETC is to obtain an explicit formula for the maximum allowable sampling period (MASP). To this end, an approach on finding the MASP is proposed first. After that, an asynchronous PETC strategy is formulated, a communication function and a control function are designed

for each agent to determine respectively whether or not the sampled state and control input should be transmitted at each sampling instant. Finally, the constraint of limited data rate is considered. An observer-based encoder-decoder and a finite-level quantizer are designed respectively for the sensor-controller communication and the controller-actuator communication such that certain constraint on the data rate is satisfied. It is shown that exponential consensus can be achieved under the PETC strategy.

The covered material is based on the following contributions.

- P. Yu and D. V. Dimarogonas, "Explicit computation of sampling period in periodic event-triggered multi-agent control under limited data rate," IEEE Transactions on Control and Network Systems, vol. 6, no. 4, pp. 1366-1378, 2019.
- P. Yu and D. V. Dimarogonas, "Explicit computation of sampling period in periodic event-triggered multi-agent control," in 2018 American Control Conference (ACC), pp. 3038-3043, Milwaukee, WC, 2018.

Chapter 5: Leader-Follower Target Tracking under Time Constraint

In Chapter 5, we consider a leader-follower MAS subject to a sequence of dynamically activated tasks. Each task is associated with a relative deadline and can be completed at several Quality-of-Satisfaction (QoSa) levels. By taking into account the reward and cost of satisfying the tasks, a dynamic scheduling strategy is proposed. Based on the dynamic plan, distributed control laws are designed for the leader and follower agents, respectively. It is shown that the proposed control laws guarantee the completion of each task at its desired QoSa level.

The covered material is based on the following contributions.

- P. Yu and D. V. Dimarogonas, "Time-constrained leader-follower multi-agent task scheduling and control synthesis," IEEE Transactions on Control and Network Systems. (under review)
- P. Yu and D. V. Dimarogonas, "Time-constrained multi-agent task scheduling based on prescribed performance control," in 2018 IEEE 57th Annual Conference on Decision and Control (CDC), pp. 2593-2598, Miami beach, 2018.

Chapter 6: Symbolic Control of Continuous-Time Uncertain Nonlinear Systems

In Chapter 6, we study the construction of symbolic models for continuous-time uncertain nonlinear systems. A novel stability notion called η-approximate controlled globally practically stable with respect to a set, and a new simulation relation called robust approximate (bi)simulation relation are proposed. It is shown that an uncertain system, under the condition that there exists an admissible control interface such that the augmented system (composed of the concrete system and its abstraction) can be made η-approximate controlled globally practically stable with respect to the given set, robustly approximately simulates its discrete state-space abstraction.

The covered material is based on the following contributions.

- P. Yu and D. V. Dimarogonas, “Robust approximately symbolic models for a class of continuous-time uncertain nonlinear systems via a control interface," Automatica. (under review)
- P. Yu and D. V. Dimarogonas, “Approximately symbolic models for a class of continuous-time nonlinear systems," in 2019 IEEE 58th Annual Conference on Decision and Control (CDC), pp. 4349-4354, Nice, 2019.

Chapter 7: Robust Satisfiability Check and Control Synthesis under STL Specifications

Chapter 7 focuses on the robust satisfiability check and online control synthesis of uncertain discrete-time systems under STL specifications. Different from existing techniques, we propose an approach based on STL, reachability analysis, and temporal logic trees. Firstly, a real-time version of STL semantics and a tube-based temporal logic tree are proposed. We show that such a tree can be constructed from every STL formula. Secondly, using the tube-based temporal logic tree, a sufficient condition is obtained for the robust satisfiability check of the uncertain system. When the underlying system is deterministic, a necessary and sufficient condition for satisfiability is obtained. After that, an online control synthesis algorithm is designed. It is shown that control synthesis algorithm is sound for uncertain systems, and both sound and complete for deterministic systems.

The covered material is based on the following contributions.

- P. Yu, Y. Gao, K. H. Johansson, and D. V. Dimarogonas, “Robust satisfiability check and online control synthesis for uncertain systems under signal temporal logic specifications," Nonlinear Analysis: Hybrid Systems. (under review)

Chapter 8: Distributed Motion Coordination under LTL Specifications

Chapter 8 is concerned with the motion coordination of MAS moving in a shared workspace, where each agent is assigned an LTL specification. Based on the realistic assumptions that each agent is subject to both state and input constraints and can have only local view and local information, a fully distributed multi-agent motion coordination strategy is proposed. For each agent, the motion coordination strategy consists of three layers. An offline layer pre-computes the braking area in the workspace, the controlled transition system, and a so-called potential function. An initialization layer outputs an initially safely satisfying trajectory. A coordination layer resolves conflicts online. The online coordination layer is further decomposed into three steps. Firstly, a conflict detection algorithm is implemented, which detects conflicts with neighboring agents. Whenever conflicts are detected, a rule is designed to assign dynamically a planning order to each pair of neighboring agents. Finally, a sampling-based algorithm is designed to generate local collision-free trajectories for the agent which at the same time guarantees the feasibility of the specification. Safety is proven to be guaranteed for all the agents at any time.

The covered material is based on the following contributions.

- P. Yu and D. V. Dimarogonas, “Distributed motion coordination for multi-robot systems under LTL specifications," IEEE Transactions on Robotics. (under review)

- P. Yu and D. V. Dimarogonas, “A fully distributed motion coordination strategy for multi-robot systems with local information," in American Control Conference (ACC), 2020. **Best Student Paper Award Finalist**

Finally, in Chapter 9, we present a summary of the results and discuss directions for future research.

Contributions not included in this thesis

The following publications are not covered in this thesis, but are related to the work presented here:

- Y. Gao, P. Yu, D. V. Dimarogonas, K. H. Johansson, and L. Xie, “Robust self-triggered control for time-varying and uncertain constrained systems via reachability analysis," Automatica , vol. 107, pp. 574-581, 2019.
- P. Yu, C. Fischione, and D. V. Dimarogonas, “Event-Triggered Output Feedback Control for Linear Systems under Tactile Communication," in 2017 IEEE 56th Annual Conference on Decision and Control (CDC), pp. 5451-5456, Melbourne, 2017.

Contribution by the Author

The order of authors reflects their contribution to each paper. The first author has the most important contribution, while the author D.V. Dimarogonas has taken the supervisory role. In all the listed publications, all the authors were actively involved in formulating the problems, developing the solutions, evaluating the results, and writing the paper.

Chapter 2

Preliminaries

In this chapter, we present the key definitions and results that will be used throughout this thesis.

2.1 Class $\mathcal{K}$, $\mathcal{K}_\infty$, $\mathcal{KL}$ functions

Before moving on, we first introduce the class of $\mathcal{K}$, $\mathcal{K}_\infty$, and $\mathcal{KL}$ functions. A continuous function $\gamma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\gamma(0) = 0$. γ is said to belong to class $\mathcal{K}_\infty$ if $\gamma \in \mathcal{K}$ and $\gamma(r) \to \infty$ as $r \to \infty$. A continuous function $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to belong to class $\mathcal{KL}$ if for each fixed s, the map $\beta(r, s)$ belongs to class $\mathcal{K}_\infty$ with respect to r and, for each fixed r, the map $\beta(r, s)$ is decreasing with respect to s and $\beta(r, s) \to 0$ as $s \to \infty$.

2.2 Graph Theory

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be a graph with the set of nodes $\mathcal{V} = \{1, 2, \ldots, N\}$, and $\mathcal{E} = \{(i, j) : i, j \in \mathcal{V}, j \neq i\}$ being the set of edges. If $(i, j) \in \mathcal{E}$, then node j is called a neighbor of node i and node j can receive information from node i. The neighboring set of node i is denoted by $\mathcal{N}_i = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$ and $\mathcal{N}_i^+ = \mathcal{N}_i \cup \{i\}$.

A graph is called undirected if $(i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$. A graph is called directed if the condition $(i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$ does not hold in general. Given an edge $e_k := (i, j) \in \mathcal{E}$, i is called the head of e_k and j is called the tail of e_k. A directed path from node i to node j is a sequence of ordered

edges connecting i and j, such that the head of each edge is equal to the tail of the following edge. A graph is called connected if for every pair of nodes (i, j), there exists a path which connects i and j. A directed graph contains a directed spanning tree if there exists a node called the root such that there exist directed paths from this node to every other node.

The adjacency matrix is denoted by $\mathcal{A} = (a_{ij})_{N\times N}$ and is given by $a_{ij} = 1$, if $(i, j) \in \mathcal{E}$, otherwise $a_{ij} = 0$. Let $\mathcal{D} = (d_{ij})_{N\times N}$ represent the degree matrix which is a diagonal matrix with entries $d_{ii} = \sum_{j=1,j\neq i}^{N} a_{ij}$. Then the Laplacian matrix of the graph $\mathcal{G}$ is defined as $L = (l_{ij})_{N\times N} = \mathcal{D} - \mathcal{A}$.

Let

$$\begin{aligned}\tilde{L} &= (\tilde{l}_{ij})_{(N-1)\times(N-1)} \\ &= \begin{bmatrix} l_{22} - l_{12} & \cdots & l_{2N} - l_{1N} \\ \cdots & \ddots & \cdots \\ l_{N2} - l_{12} & \cdots & l_{NN} - l_{1N} \end{bmatrix}.\end{aligned} \tag{2.1}$$

Lemma 2.1. *[32] Denote the eigenvalues of Laplacian matrix L and the matrix $\tilde{L}$, respectively by $\lambda_1, \lambda_2, \ldots, \lambda_N$ and $\mu_1, \mu_2, \ldots, \mu_{N-1}$, where $0 = |\lambda_1| \leqslant |\lambda_2| \leqslant \ldots \leqslant |\lambda_N|$ and $|\mu_1| \leqslant |\mu_2| \leqslant \ldots \leqslant |\mu_{N-1}|$. Then $\lambda_2 = \mu_1, \lambda_3 = \mu_2, \cdots, \lambda_N = \mu_{N-1}$.*

Lemma 2.2. *[33] Suppose that the matrix $A \in \mathbb{R}^{n\times n}$ is Hurwitz. Then, for all $t \geq 0$, it holds that $\left\|e^{At}\right\| \leq \|P_A\| \left\|P_A^{-1} c_A e^{a_A t}\right\|$, where P_A is a nonsingular matrix such that $P_A^{-1} A P_A = J_A$ with J_A being the Jordan canonical form of A, c_A is a positive constant determined by A, and $\max_i Re(\lambda_i(A)) < a_A < 0$.*

2.3 System Properties

Consider an autonomous system of the form

$$\dot{x}(t) = f(x(t)), \tag{2.2}$$

where $x(t) \in \mathbb{R}^n$.

Definition 2.1. *A function $f : \mathbb{R}^n \to \mathbb{R}^n$ is said to satisfy a global Lipschitz condition if there exists a constant ρ such that*

$$\|f(x_1) - f(x_2)\| \leq \rho\|x_1 - x_2\|, \quad \forall x_1, x_2 \in \mathbb{R}^n.$$

Definition 2.2. *Let f be a function whose domain is the set X and whose image is the set Y. Then f is invertible if there exists a function g with domain Y and image X, with the property*

$$f(x) = y \iff g(y) = x.$$

That function g is called the inverse of f, and denoted by f^{-1}.

Consider a continuous-time dynamical system, given by

$$\dot{x}(t) = f(x(t), u(t)) \tag{2.3}$$

where $x(t) \in \mathbb{R}^n, u(t) \in U \subseteq \mathbb{R}^m$ are the state and control input at time t, respectively. We assume that $f : \mathbb{R}^n \times U \to \mathbb{R}^n$ is continuous in x and u, and the vector field f is such that for any input in U, any initial condition $x_0 \in \mathbb{R}^n$, this differential equation has a unique solution.

Let $\mathcal{U}$ be the set of all measurable functions that take their values in U and are defined on $\mathbb{R}_{\geq 0}$. A curve $\xi : [0, \tau[\to \mathbb{R}^n$ is said to be a solution/trajectory of (2.3) if there exists an input signal $u \in \mathcal{U}$ satisfying (2.3) for almost all $t \in [0, \tau[$. We use $\xi(x_0, u, t)$ to denote the trajectory point reached at time t under the input signal $u \in \mathcal{U}$ from initial state x_0.

Definition 2.3. *[34] The system (2.3) is called forward complete (FC) if for every initial condition $x_0 \in \mathbb{R}^n$ and every input signal $u \in \mathcal{U}$, the corresponding solution is defined for all $t \geq 0$.*

Definition 2.4. *[35] The system (2.3) is called incrementally globally asymptotically stable (δ-GAS) if it is FC and there exists a class $\mathcal{KL}$ function β such that for any $t \in \mathbb{R}_{\geq 0}$, any initial conditions $x_0, x_0' \in \mathbb{R}^n$, and any input signal $u \in \mathcal{U}$,*

$$\|\xi(x_0, u, t) - \xi(x_0', u, t)\| \leq \beta(\|x_0 - x_0'\|, t).$$

Definition 2.5. *[35] The system (2.3) is called incrementally input-to-state stable (δ-ISS) if there exists a class $\mathcal{KL}$ function β and a class $\mathcal{K}_\infty$ function κ such that for any $t \geq 0$, any initial conditions $x_0, x_0' \in \mathbb{R}^n$, and any input signals $u, u' \in \mathcal{U}$,*

$$\|\xi(x_0, u, t) - \xi(x_0', u', t)\| \leq \beta(\|x_0 - x_0'\|, t) + \kappa(\|u - u'\|_\infty). \tag{2.4}$$

Definition 2.6. *[36] The system (2.3) is called incrementally forward complete (δ-FC) if it is FC and there exist continuous functions $\beta : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and $\kappa : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ such that for every $s \in \mathbb{R}_{\geq 0}$, the functions $\beta(\cdot, s)$ and $\kappa(\cdot, s)$ belong to class $\mathcal{K}_\infty$, and for any initial conditions $x_0, x_0' \in \mathbb{R}^n$, any $T \in \mathbb{R}_{\geq 0}$, and any input signals $u, u' \in \mathcal{U}$, the following condition is satisfied for all $t \in [0, T]$:*

$$\|\xi(x_0, u, t) - \xi(x_0', u', t)\| \leq \beta(\|x_0 - x_0'\|, t) + \kappa(\|u - u'\|_\infty, t). \tag{2.5}$$

Lemma 2.3. *Let $V : [0, \infty) \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ be a continuously differentiable function such that*

$$\underline{\alpha}(\|x\|) \leq V(t, x, u) \leq \bar{\alpha}(\|x\|)$$
$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t, x, u) \leq -\gamma V(t, x, u), \quad \forall \|x\| \geq \mu > 0,$$

$\forall t \geq 0$ and $\forall x \in \mathbb{R}^n$, where $\underline{\alpha}, \bar{\alpha}$ are class $\mathcal{K}_\infty$ functions, $\mu > 0, \gamma > 0$ are constants. Then, the solution $x(t)$ to the differential equation $\dot{x} = f(t, x, u)$ exists and satisfies

$$\|x(t)\| \leq \beta(\|x(0)\|, t) + \underline{\alpha}^{-1}(\bar{\alpha}(\mu)),$$

where

$$\beta(r, t) = \underline{\alpha}^{-1}(e^{-\gamma t} \bar{\alpha}(r))$$

is a class $\mathcal{KL}$ function.

Proof. The proof follows from Lemma 4.4 and Theorem 4.18 of [37], and hence omitted. □

Let $\phi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ be the solution to the following differential equation

$$\dot{\phi} = -m_1 \phi^2 - m_2 \phi - m_3 \quad \phi(0) = \phi_0 \tag{2.6}$$

where $\phi_0 > 0, m_i > 0, i = 1, 2, 3$. Then, one can compute that

$$\phi^{-1}(0) = \begin{cases} \frac{2}{r} \arctan\left(\frac{\phi_0 \sqrt{r}}{\phi_0 m_2 + 2m_3}\right), & \text{if } 4m_1 m_3 > {m_2}^2, \\ \frac{4m_1 \phi_0}{2m_1 m_2 \phi_0 + {m_2}^2}, & \text{if } 4m_1 m_3 = {m_2}^2, \\ \frac{2}{r} \text{arctanh}\left(\frac{\phi_0 \sqrt{r}}{\phi_0 m_2 + 2m_3}\right), & \text{if } 4m_1 m_3 < {m_2}^2. \end{cases}$$

where

$$r := \sqrt{|4m_1 m_3 - m_2^2|}.$$

Lemma 2.4. *There exists $\phi_0 > 0$, such that $\phi(\tau) \geq 0$ for all $\tau \in [0, \mathcal{T}(m_1, m_2, m_3))$, where $\mathcal{T}(m_1, m_2, m_3)$ is given by*

$$\mathcal{T}(m_1, m_2, m_3) := \begin{cases} \frac{2}{r} \arctan\left(\frac{r}{m_2}\right), & \text{if } 4m_1m_3 > {m_2}^2, \\ \frac{2}{m_2}, & \text{if } 4m_1m_3 = {m_2}^2, \\ \frac{2}{r} \operatorname{arctanh}\left(\frac{r}{m_2}\right), & \text{if } 4m_1m_3 < {m_2}^2. \end{cases}$$

Proof. It follows from the fact that $\phi^{-1}(0)$ increases as ϕ_0 increases, and $\mathcal{T}(m_1, m_2, m_3) = \lim_{\phi_0 \to \infty} \phi^{-1}(0)$. □

2.4 Transition Systems

Consider a continuous-time dynamical system, given by

$$\begin{cases} \dot{x} = f(x, u), \\ y = g(x), \end{cases} \tag{2.7}$$

where $x \in X \subseteq \mathbb{R}^n$ is the state, $u \in U \subseteq \mathbb{R}^m$ is the control input, $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ describes the dynamics, $y \in Y \subseteq \mathbb{R}^p$ is the output, and $g : \mathbb{R}^n \to \mathbb{R}^p$ is the output function.

A curve $\xi : [0, \infty) \to \mathbb{R}^n$ is said to be a trajectory of (2.7) if there exists an input $u \in \mathcal{U}$ satisfying $\dot{\xi}(t) = f(\xi(t), u(t))$ for almost all $t \in [0, \infty)$. A curve $\zeta : [0, \infty) \to \mathbb{R}^l$ is said to be an output trajectory of (2.7) if $\zeta(t) = g(\xi(t))$ for almost all $t \in [0, \infty)$, where $\xi(t)$ is a trajectory of (2.7). We use $\xi(\xi_0, u, t)$ and $\zeta(\zeta_0, u, t)$ to denote the trajectory and output trajectory point reached at time t under the input $u \in \mathcal{U}$ from initial condition ξ_0 and ζ_0, respectively.

The continuous-time system (2.7) can be represented as an (infinite) transition system $\mathcal{T} = (X, X_0, \Sigma, \to, f, O, g)$, where

- X is the set of states,
- $X_0 \subseteq X$ is the set of initial states,
- $\Sigma = \mathcal{U}$ is the set of inputs,
- $\to$: $X \times \Sigma \to 2^X$ is the transition relation,
- $O = Y$ is the set of observations, and

- g is the observation map.

The transition relation $x' \in\rightarrow (x, u)$ if and only if $x' = \xi(x, u, \tau)$, where ξ is the trajectory of (2.7) and $\tau > 0$ is a chosen constant. For convenience, $x' \in\rightarrow (x, u)$ will be denoted as $x \xrightarrow{u} x'$.

2.5 Temporal Logics

2.5.1 Linear Temporal Logic

Let AP be a set of atomic propositions. Linear temporal logic (LTL) is based on atomic propositions (state labels $p \in AP$), Boolean connectors like negation $\neg$ and conjunction $\wedge$, and two temporal operators X ("next") and U ("until"), and is formed according to the following syntax [17]:

$$\varphi ::= \top | p | \neg\varphi | \varphi_1 \wedge \varphi_2 | \mathsf{X}\varphi | \varphi_1 \mathsf{U} \varphi_2, \tag{2.8}$$

where $\varphi, \varphi_1, \varphi_2$ are LTL formulas. The Boolean connector disjunction $\vee$, and temporal operators F ("eventually") and G ("always") can be derived as $\varphi_1 \vee \varphi_2 := \neg(\neg\varphi_1 \wedge \neg\varphi_2)$, $\mathsf{F}\varphi := \top \mathsf{U} \varphi$ and $\mathsf{G}\phi := \neg\mathsf{F}\neg\phi$. Formal definitions for the LTL semantics and model checking can be found in [17].

Definition 2.7. *[38] A nondeterministic Büchi automaton (NBA) is a tuple $\mathcal{B} = (S, S_0, 2^{AP}, \delta, F)$, where*

- *S is a finite set of states,*
- *$S_0 \subseteq S$ is the set of initial states,*
- *2^{AP} is the input alphabet,*
- *$\delta : S \times 2^{AP} \to 2^S$ is the transition function, and*
- *$F \subseteq S$ is the set of accepting states.*

An infinite *run* $\mathbf{s}$ of a NBA is an infinite sequence of states $\mathbf{s} = s_0 s_1 \ldots$ generated by an infinite sequence of input alphabets $\sigma = \sigma_0 \sigma_1 \ldots \in (2^{AP})^\omega$, where $s_0 \in S_0$ and $s_{k+1} \in \delta(s_k, \sigma_k), \forall k \geq 0$. An infinite run $\mathbf{s}$ is called *accepting* if $\mathrm{Inf}(s) \cap F \neq \emptyset$, where $\mathrm{Inf}(s)$ is the set of states that appear in $\mathbf{s}$ infinitely often. Given a state $s \in S$, define

$$\mathtt{Post}(s) := \{s' \in S : \exists \sigma \in 2^{AP}, s' \in \delta(s, \sigma)\}. \tag{2.9}$$

Given an LTL formula φ over AP, there is a union of infinite words that satisfy φ, that is,

$$\texttt{Words}(\varphi) = \{\sigma \in (2^{AP})^\omega | \sigma \models \varphi\},$$

where $\models \subseteq (2^{AP})^\omega \times \varphi$ is the satisfaction relation [17].

Lemma 2.5. *[19] Any LTL formula φ over AP can be algorithmically translated into a NBA $\mathcal{B}_\varphi$ over the input alphabet 2^{AP} such that $\mathcal{B}_\varphi$ accepts all and only those infinite runs over AP that satisfy φ.*

Definition 2.8 (Controlled transition system)**.** *Given a transition system $\mathcal{T} = (X, X_0, \Sigma, \to, f, O, g)$ and a set of atomic propositions AP, we define the controlled transition system (CTS) $\mathcal{T}_c = (X, X_0, AP, \to, L_c)$, where*

- *$L_c : X \to 2^{AP}$ is a labelling function.*

The labelling function $L_c(x)$ maps a state x to the finite set of AP which are true at state x. Given a state $x \in X$, define

$$Post(x) := \{x' \in X : \exists u \in \Sigma, x \xrightarrow{u} x'\}. \tag{2.10}$$

An infinite *path* of the CTS $\mathcal{T}_c$ is a sequence of states $\boldsymbol{\varrho} = x_0 x_1 x_2 \ldots$ generated by an infinite sequence of inputs $u_0 u_1 u_2 \ldots$ such that $x_0 \in X_0$ and $x_k \xrightarrow{u_k} x_{k+1}$ for all $k \geq 0$. Its *trace* is the sequence of atomic propositions that are true in the states along the path, *i.e.,* $\texttt{Trace}(\boldsymbol{\varrho}) := L_c(x_0)L_c(x_1)L_c(x_2)\ldots$. The satisfaction relation $\boldsymbol{\varrho} \models \varphi$ if and only if

$$\texttt{Trace}(\boldsymbol{\varrho}) \in \texttt{Words}(\varphi).$$

Definition 2.9 (Product Büchi automaton [17])**.** *Given a CTS $\mathcal{T}_c = (X, X_0, AP, \to, L_c)$ and a NBA $\mathcal{B} = (S, S_0, 2^{AP}, \delta, F)$, the product Büchi automaton (PBA) $\mathcal{P} = \mathcal{T}_c \times \mathcal{B} = (S_p, S_{0,p}, 2^{AP}, \delta_p, F_p)$, where*

- *$S_p = X \times S$,*
- *$S_{0,p} = X_0 \times S_0$,*
- *$\delta_p \subseteq S_p \times S_p$, defined by $((x, s), (x', s')) \in \delta_p$ if and only if $x' \in Post(x)$ and $s' \in \texttt{Post}(s)$,*
- *$F_p = (X \times F) \cap S_p$.*

2.5.2 Signal Temporal Logic

Signal temporal logic (STL) [18] is a predicate logic consisting of predicates μ, which are defined through a predicate function $g_\mu : \mathbb{R}^n \to \mathbb{R}$ as

$$\mu := \begin{cases} \top, & \text{if} \quad g_\mu(x) \geq 0 \\ \bot, & \text{if} \quad g_\mu(x) < 0. \end{cases}$$

The syntax of STL is given by

$$\varphi ::= \top \mid \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \mathsf{U}_\mathrm{I} \varphi_2, \tag{2.11}$$

where $\varphi, \varphi_1, \varphi_2$ are STL formulas and I is a closed or half-closed interval of $\mathbb{R}$ of the form $[a, b]$ or $[a, b)$ with $a, b \in \mathbb{R}_{\geq 0} \cup \infty$ and $a \leq b$.

The validity of a STL formula φ with respect to a discrete-time signal $\boldsymbol{x}$ at time t_k, is defined inductively as follows [26]:

$$\begin{aligned} (\boldsymbol{x}, t_k) \vDash \mu \quad &\Leftrightarrow g_\mu(\boldsymbol{x}(t_k)) \geq 0, \\ (\boldsymbol{x}, t_k) \vDash \neg\varphi \quad &\Leftrightarrow \neg((\boldsymbol{x}, t_k) \vDash \varphi), \\ (\boldsymbol{x}, t_k) \vDash \varphi_1 \wedge \varphi_2 \quad &\Leftrightarrow (\boldsymbol{x}, t_k) \vDash \varphi_1 \wedge (\boldsymbol{x}, t_k) \vDash \varphi_2, \\ (\boldsymbol{x}, t_k) \vDash \varphi_1 \mathsf{U}_{[a,b]} \varphi_2 \quad &\Leftrightarrow \exists t_{k'} \in [t_k + a, t_k + b] \text{ s.t. } (\boldsymbol{x}, t_{k'}) \\ &\qquad \vDash \varphi_2 \wedge \forall t_{k''} \in [t_k, t_{k'}], (\boldsymbol{x}, t_{k''}) \vDash \varphi_1. \end{aligned}$$

The signal $\boldsymbol{x} = x_0 x_1 \ldots$ satisfies φ, denoted by $\boldsymbol{x} \vDash \varphi$ if $(\boldsymbol{x}, t_0) \vDash \varphi$.

Definition 2.10. *[39] The time horizon $\|\phi\|$ of a STL formula ϕ is defined as*

$$\|\phi\| = \begin{cases} 0, & \textit{if } \phi = \mu \\ \|\phi_1\|, & \textit{if } \phi = \neg\phi_1 \\ \max\{\|\phi_1\|, \|\phi_2\|\}, & \textit{if } \phi = \phi_1 \wedge \phi_2 \\ b + \max\{\|\phi_1\|, \|\phi_2\|\}, & \textit{if } \phi = \phi_1 \mathsf{U}_{[a,b]} \phi_2. \end{cases}$$

Part II

Distributed Control of Multi-Agent Systems under Cooperative Tasks

Chapter 3

Distributed Event-Triggered Consensus

This chapter studies the distributed control of multi-agent consensus. An asynchronous event-triggered control (ETC) strategy is proposed for linear multi-agent systems (MAS) such that the rates of communication and controller update are reduced. It is shown that asymptotical consensus can be achieved as well as Zeno behavior is excluded.

3.1 Introduction

During the past decades, numerous contributions have been made in the research of distributed cooperative control of MAS due to its wide applications in various fields, for instance, robots formation and synchronization [5], [6], estimation over sensor networks [7], and distributed computing [8]. Consensus, as the benchmark problem, has been investigated intensively.

An important aspect in the implementation of distributed control is the design of communication and control strategy. At the early stage, control laws for multi-agent consensus require either continuous or high frequency interaction between neighboring agents, which is not resource efficient in terms of communication. This fact has resulted in a recent interest on ETC [14], [33], [40]–[50]. Different from sampled-data control, where the states of each agent are sampled and transmitted and the controller is updated at every sampling instant, the time instants in ETC (at which the state or/and control input is transmitted), is determined by a pre-defined triggering condition [14]. In this

way, a substantial reduction of rate of communication may be achieved. The possibility of reducing control costs and saving resources, makes ETC appealing for resource-limited control systems. Since the pioneering work reported in paper [14], ETC has been widely studied in networked control systems and MAS. In the context of MAS, it is found that the design of ETC strategy is more challenging compared to single-agent-based networked control systems.

Motivated by [40], where the centralized and distributed event-triggered consensus problem for single-integrator is studied, great efforts were devoted to the ETC of MAS with single- and double-integrator dynamics [40], [42], [43]. In [40], [43], the control input of each agent was required to be triggered at its own communication instants as well as all its neighbors', which will result in an very frequent control update as the number of neighbors is increasing. To cope with this problem, a combinational measurement approach to designing the event-triggered scheme is developed in [42]. However, continuous communication between neighboring agents is required. Recently, some researches have considered event-triggered consensus problem for general linear MAS [33], [45]–[47], [49]. In [45], [46], continuous communication of neighbors' states are required to check the triggering conditions. In [47], an additional constant is introduced in the threshold function to avoid continuous communication, whereas only bounded consensus can be achieved rather than complete consensus. Then, in [33], [49], the continuous requirement for communication is relaxed by introducing additional assumptions [49] or by using the matrix exponential function e^{At} [33], nevertheless the continuous controller update is still required.

Motivated by the above discussion, this chapter investigates the distributed ETC of linear MAS, where output feedback is considered. The contributions are summarized as follows. A distributed asynchronous event-triggered communication and control strategy is proposed, which is capable of reducing both the rates of communication and controller update. It is shown that consensus is achieved asymptotically. Furthermore, Zeno behavior is excluded.

3.2 Problem Formulation

Consider a MAS with N agents, and the dynamics of each agent is formulated by

$$\begin{aligned} \dot{x}_i(t) &= Ax_i(t) + Bu_i(t), \\ y_i(t) &= Cx_i(t), \quad i = 1, 2, \ldots, N, \end{aligned} \tag{3.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{R}^{l\times n}$ are constant matrices; $x_i \in \mathbb{R}^n$, $u_i \in \mathbb{R}^m$ and $y_i \in \mathbb{R}^l$ are agent i's state, control input and measurement output, respectively.

Assumption 3.1. *The communication graph $\mathcal{G}$ formed by the group of agents contains a directed spanning tree.*

Assumption 3.2. *[51] The matrix pair (A, B) is stabilizable. That is, the following algebraic Riccati equation (ARE)*

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{3.2}$$

has a unique solution $P = P^T \succ 0$ for any given matrices $R = R^T \succ 0$ and $Q = Q^T \succ 0$.

Assumption 3.3. *The matrix pair (A, C) is detectable.*

Definition 3.1. *The consensus of the MAS (3.1) is said to be achieved asymptotically, if and only if for any initial condition,*

$$\lim_{t\to\infty} \|x_i(t) - x_j(t)\| = 0, \quad \forall i, j, i \neq j.$$

An observer-based consensus protocol is proposed in [52], that is,

$$\dot{\hat{x}}_i(t) = A\hat{x}_i(t) + Bu_i(t) + F(y_i(t) - C\hat{x}_i(t)), \tag{3.3}$$

and

$$u_i(t) = -cK \sum_{j\in\mathcal{N}_i} \Big(\hat{x}_i(t) - \hat{x}_j(t)\Big), \tag{3.4}$$

where $\hat{x}_i \in \mathbb{R}^n$ is the observer state, $c > 0$ is the coupling gain, and $F \in \mathbb{R}^{n\times l}$ and $K \in \mathbb{R}^{m\times n}$ are the feedback gain matrices to be determined. Let the gain matrix $K = R^{-1}B^T P$, where P is the unique solution of ARE (3.2) for appropriately chosen $R = R^T \succ 0$ and $Q = Q^T \succ 0$.

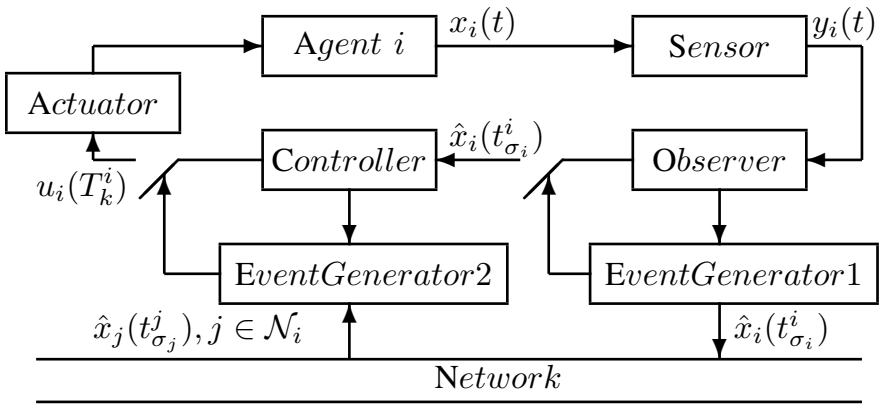

Figure 3.1: The structure of ETC for agent i.

It was proven in [52] that under Assumptions 3.1-3.3, consensus of the MAS (3.1) is achieved with the controller (3.4) if and only if the coupling gain $c > 1/(2\underline{\lambda}_R)$, where $\underline{\lambda}_R = \min_{2\leq i\leq N} \mathrm{Re}(\lambda_i)$ and λ_i is the i-th eigenvalue of the Laplacian matrix L.

However, in order to be implemented, the controller is required to access the observer state continuously and update continuously. Although different ETC strategies regarding consensus of linear MAS were proposed, either continuous communication [45], [46] or continuous controller update [33], [47] is required. In this chapter, the problem under consideration is formulated as follows.

Problem 3.1. *Consider the MAS (3.1) with the observer (3.3). Design communication and controller update strategies for each agent i such that i) the requirement of continuous communication and controller update is relaxed and ii) the consensus of the MAS (3.1) is achieved asymptotically.*

3.3 Main Results

In this section, a distributed ETC strategy is proposed for the MAS (3.1). The structure of the ETC for each agent i is shown in Figure 3.1, where event generators are implemented in both the communication side and the controller side. The proposed design procedure can be divided into two major stages.

In the first stage, an event-triggered communication scheme is designed. Under such a communication scheme, in the second stage, a distributed event-triggered controller that depends only on the transmitted information of agent itself and its neighbors is proposed. The following subsections detail this procedure.

3.3.1 Communication function

This subsection presents the design of the event-triggered communication scheme. For each agent i, let $t^i_{\sigma_i}, \sigma_i = 0, 1, 2, \ldots$ be the increasing sequence of communication time instants at which $\hat{x}_i$ is transmitted. Then, we define the communication measurement error for agent i as

$$e_i(t) = e^{A\left(t-t^i_{\sigma_i}\right)}\hat{x}_i(t^i_{\sigma_i}) - \hat{x}_i(t), \quad t \in [t^i_{\sigma_i}, t^i_{\sigma_i+1}),$$

where the communication time instant $t^i_{\sigma_i}$ is updated by

$$t^i_{\sigma_i+1} = \inf\left\{t > t^i_{\sigma_i} : f\left(t, e_i\left(t\right)\right) \geq 0\right\}, \tag{3.5}$$

where

$$f\left(t, e_i\left(t\right)\right) = \left\|e_i\left(t\right)\right\| - I_0 e^{-\alpha t} \tag{3.6}$$

with $I_0 > 0, \alpha > 0$ are constants to be determined. The condition $f(t, e_i(t)) \geq 0$ is called the *communication function*. Without loss of generality, we assume $t^i_0 = 0, \forall i$.

3.3.2 Control function

Before proceeding, the following notations are introduced. For each agent i, $\zeta_i(t)$ is defined as

$$\zeta_i(t) = e^{A\left(t-t^i_{\sigma_i}\right)}\hat{x}_i(t^i_{\sigma_i}), \quad t \in [t^i_{\sigma_i}, t^i_{\sigma_i+1})$$

and $\zeta^i_j(t)$ is defined as

$$\zeta^i_j(t) = e^{A\left(t-t^j_{\sigma_j}\right)}\hat{x}_j(t^j_{\sigma_j}), \quad t \in [t^j_{\sigma_j}, t^j_{\sigma_j+1}), j \in \mathcal{N}_i.$$

The event-triggered controller for each agent i is designed as

$$u_i(t) = -cK\sum_{j\in\mathcal{N}_i}\left(\zeta_i(T^i_k) - \zeta^i_j(T^i_k)\right), \quad t \in [T^i_k, T^i_{k+1}), \tag{3.7}$$

where $T_k^i, k \in \mathbb{N}$ is the controller update time sequence of agent i. From the above definitions, one has

$$\zeta_i(T_k^i) = e^{A\left(T_k^i - t_{\sigma_i}^i\right)}\hat{x}_i(t_{\sigma_i}^i), \quad \sigma_i \triangleq \arg\min_{l\in\mathbb{N}:T_k^i \geq t_l^i} \left\{T_k^i - t_l^i\right\},$$

$$\zeta_j^i(T_k^i) = e^{A\left(T_k^i - t_{\sigma_j}^j\right)}\hat{x}_j(t_{\sigma_j}^j), \quad \sigma_j \triangleq \arg\min_{l\in\mathbb{N}:T_k^i \geq t_l^j} \left\{T_k^i - t_l^j\right\}, \forall j \in \mathcal{N}_i,$$

where $t_{\sigma_i}^i, \hat{x}_i(t_{\sigma_i}^i)$ and $t_{\sigma_j}^j, \hat{x}_j(t_{\sigma_j}^j), j \in \mathcal{N}_i$ represent the latest communication time instant and the latest received information of agent i and its neighbors j before T_k^i, respectively.

Let $z_i(t) = \sum_{j\in\mathcal{N}_i}(\zeta_i(t) - \zeta_j^i(t))$. Define the controller measurement error of agent i as the combined state differences between the last triggering instant T_k^i and the current time, which is

$$\begin{aligned}\hat{e}_i(t) &= \sum_{j\in\mathcal{N}_i}\left(\zeta_i(T_k^i) - \zeta_j^i(T_k^i)\right) - \sum_{j\in\mathcal{N}_i}(\zeta_i(t) - \zeta_j^i(t)) \\ &= z_i(T_k^i) - z_i(t).\end{aligned} \tag{3.8}$$

Combining $e_i(t)$ and $\hat{e}_i(t)$, the controller (3.7) can be rewritten as

$$u_i(t) = -cK\left\{\sum_{j\in\mathcal{N}_i}(\hat{x}_i(t) + e_i(t) - (\hat{x}_j(t) + e_j(t))) + \hat{e}_i(t)\right\}. \tag{3.9}$$

Define the observation error of agent i as $\tilde{x}_i(t) = x_i(t) - \hat{x}_i(t)$, let $\tilde{x}(t) = (\tilde{x}_1(t), \ldots, \tilde{x}_N(t))$. Then the dynamics of the observation error system is

$$\dot{\tilde{x}}(t) = (I_N \otimes (A - FC))\,\tilde{x}(t),$$

which is globally asymptotically stable if and only if the matrix $A - FC$ is Hurwitz.

Let $\hat{x}, e, \hat{e}$ be the concatenated vectors of $\hat{x}_i, e_i, \hat{e}_i$, respectively. Then, the dynamics of the observer (3.3) can be rewritten as

$$\begin{aligned}\dot{\hat{x}}(t) = &(I_N \otimes A)\,\hat{x}(t) - \left(cL \otimes BR^{-1}B^TP\right)(\hat{x}(t) + e(t)) \\ &- \left(cI_N \otimes BR^{-1}B^TP\right)\hat{e}(t) + (I_N \otimes FC)\,\tilde{x}(t).\end{aligned} \tag{3.10}$$

Let $\xi_i(t) = \hat{x}_i(t) - \hat{x}_1(t), \forall i$ and $\xi(t) = (\xi_1(t), \xi_{2-N}(t))$, where $\xi_{2-N}(t) \triangleq (\xi_2(t), \ldots, \xi_N(t)) \in \mathbb{R}^{(N-1)n}$. It follows that $\xi_1(t) \equiv 0$, and

the vector $\xi_{2-N}(t)$ satisfies

$$\begin{aligned}\dot{\xi}_{2-N}(t) =&(I_{N-1}\otimes A - c\tilde{L}\otimes BR^{-1}B^TP)\xi_{2-N}(t)\\ &-\left(c\tilde{L}W\otimes BR^{-1}B^TP\right)e(t)\\ &-\left(cW\otimes BR^{-1}B^TP\right)\hat{e}(t)+(W\otimes FC)\,\tilde{x}(t)\\ =&\Pi\xi_{2-N}(t)-G_1e(t)-G_2\hat{e}(t)+G_3\tilde{x}(t),\end{aligned}\tag{3.11}$$

where

$$\begin{aligned}\Pi &= I_{N-1}\otimes A - c\tilde{L}\otimes BR^{-1}B^TP,\\ G_1 &= c\tilde{L}W\otimes BR^{-1}B^TP,\\ G_2 &= cW\otimes BR^{-1}B^TP,\\ G_3 &= W\otimes FC,\end{aligned}$$

$W=[-1_{N-1},I_{N-1}]\in\mathbb{R}^{(N-1)\times N}$, and $\tilde{L}$ is defined in (2.1). It can be seen that the observer (3.10) achieves consensus, if and only if $\lim_{t\to\infty}\xi_{2-N}(t)=0$.

Since Assumption 3.1 holds, it follows from Lemma 2.1 that $\tilde{L}\in\mathbb{R}^{(N-1)\times(N-1)}$ is a full-rank matrix. Moreover, the eigenvalues of $\tilde{L}$ have positive real parts. Choosing the coupling gain $c>1/(2\underline{\lambda}_R)$, it is proven in [52] that the matrix Π is Hurwitz. Thus, there exists a positive definite matrix $\bar{P}=\bar{P}^T$ satisfying the Lyapunov condition $\bar{P}\Pi+\Pi^T\bar{P}=-\bar{Q}$ for any given $\bar{Q}=\bar{Q}^T\succ 0$.

Now, we are ready to define the controller update time sequence. The controller update time instants T_k^i for each agent i are given by

$$T_{k+1}^i=\inf\left\{t>T_k^i : g\left(\hat{e}_i\left(t\right),z_i(t),t\right)\geq 0\right\},\tag{3.12}$$

where

$$g\left(\hat{e}_i\left(t\right),z_i\left(t\right),t\right)=\left\|\hat{e}_i(t)\right\|-\left(\theta\gamma\left\|z_i(t)\right\|+\eta e^{-\alpha t}\right)\tag{3.13}$$

with constants $0\leq\theta<1$, $\gamma=\lambda_{\min}\left(\bar{Q}\right)/(2(\hat{l}+N\sqrt{\hat{l}})\left\|\bar{P}G_2\right\|)$, $\eta>0$, $\hat{l}=\max_i\{d_{ii}\}\leq N-1$, and α defined in (3.6). The condition $g\left(\hat{e}_i\left(t\right),z_i(t),t\right)\geq 0$ is called the *control function*. Without loss of generality, we assume $T_0^i=0,\forall i$.

From the definitions of $z_i(t)$ and $\hat{e}_i(t)$, one can see that only the discrete communication time instants and the states transmitted at these communication time instants (determined by the communication function (3.6)) are required to implement the control function (3.13). When the controller measurement error $\hat{e}_i$ exceeds a certain threshold, that is, $g\left(\hat{e}_i(T_k^i),z_i(T_k^i),t\right)\geq 0$, an

event is triggered for agent i. Agent i updates its controller using the latest communication instants and the latest received states. Meanwhile, the controller measurement error $\hat{e}_i$ is reset to zero.

Remark 3.1. *The communication function $f(t, e_i(t)) \geq 0$ can be seen as a trigger for communication and the control function $g\left(\hat{e}_i\left(t\right), z_i(t), t\right) \geq 0$ can be seen as a trigger for controller update. They work asynchronously. Note that no communication is required at the controller update time instants for each agent, which is different from the previous ETC strategies, where each agent updates its controller and communicates with its neighbors at the same time.*

Remark 3.2. *It is worth to mention that except for the controller update time instants, the control function $g\left(\hat{e}_i\left(t\right), z_i(t), t\right)$ will also be reset at the communication instants of itself and its neighbors. Therefore, $g\left(\hat{e}_i\left(t\right), z_i(t), t\right)$ can be discontinuous within two successive controller update instants, thereby resulting in additional difficulties in proving the exclusion of Zeno behavior. Furthermore, different from the control functions proposed in [42], [47], an extra term $\eta e^{-\alpha t}$ is introduced in this chapter to exclude Zeno behavior.*

3.3.3 Convergence result

Now, we are in the position to give the following result.

Theorem 3.1. *Consider the MAS (3.1) with the observer (3.3), where the matrix F is chosen such that the matrix $(A-FC)$ is Hurwitz. Let the controller be given in (3.7), where the coupling gain satisfies $c > 1/(2\underline{\lambda}_R)$. Suppose Assumptions 3.1-3.3 hold and that the communication function (3.6) and the control function (3.13) are applied with*

$$0 < \alpha < \min\{((1-\theta)\lambda_{\min}(\bar{Q}) - a)/2\lambda_{\max}(\bar{P}), -\max_i Re\left(\lambda_i\left(A - FC\right)\right)\},$$

where $0 < a < (1-\theta)\lambda_{\min}\left(\bar{Q}\right)$. Then, consensus of the MAS (3.1) is achieved asymptotically. Furthermore, Zeno behavior is excluded.

Proof. Consider the following Lyapunov function candidate

$$V(t) = \xi_{2-N}(t)^T(t)\bar{P}\xi_{2-N}(t).$$

Differentiating $V(t)$ along the trajectories of (3.11), one has

$$\begin{aligned}\dot{V}(t) = &- \xi_{2-N}^T(t)\bar{Q}\xi_{2-N}(t) - 2\xi_{2-N}^T(t)\bar{P}G_1 e(t) \\ &- 2\xi_{2-N}^T(t)\bar{P}G_2\hat{e}(t) + 2\xi_{2-N}^T(t)\bar{P}G_3\tilde{x}(t).\end{aligned} \tag{3.14}$$

According to (3.5), one has $\|e_i(t)\| \leq I_0 e^{-\alpha t}, \forall i$, and thus $\|e(t)\| \leq \sqrt{N} I_0 e^{-\alpha t}$. Besides, for agent i, an event for controller update is triggered at T_{k+1}^i when $g\left(\hat{e}_i\left(t\right), z_i(t)\right) \geq 0$. Thus, one has $g\left(\hat{e}_i\left(t\right), z_i(t)\right) < 0$ for $t \in [T_k^i, T_{k+1}^i)$. According to the definition of $z_i(t)$, one has

$$\begin{aligned}\|z_i\left(t\right)\| =& \Big\| \sum_{j\in\mathcal{N}_i} \left(\zeta_i(t) - \zeta_j^i(t)\right)\Big\| \\ \leq& \Big\| \sum_{j\in\mathcal{N}_i} \left((\hat{x}_i(t) - \hat{x}_1(t)) - (\hat{x}_j(t) - \hat{x}_1(t))\right)\Big\| + \Big\| \sum_{j\in\mathcal{N}_i} \left(e_i(t) - e_j(t)\right)\Big\| \\ \leq& \sum_{j\in\mathcal{N}_i} \left(\|\xi_i(t)\| + \|\xi_j(t)\|\right) + \sum_{j\in\mathcal{N}_i} \left(\|e_i(t)\| + \|e_j(t)\|\right).\end{aligned} \tag{3.15}$$

Letting $\hat{\gamma} = \lambda_{\min}\left(\bar{Q}\right)/\left(2\left\|\bar{P}G_2\right\|\right)$, one further has

$$\begin{aligned}\|\hat{e}\left(t\right)\| \leq& \sum_{i=1}^{N} \left(\theta\gamma \|z_i(t)\| + \eta e^{-\alpha t}\right) \leq \theta\hat{\gamma}\left(\|\xi(t)\| + \|e(t)\|\right) + N\eta e^{-\alpha t} \\ \leq& \theta\hat{\gamma}\|\xi_{2-N}(t)\| + (\theta\hat{\gamma} I_0 + \eta) N e^{-\alpha t}.\end{aligned}$$

Using the inequality $2xy \leq ax^2 + y^2/a, \forall a > 0$ several times, (3.14) can then be rewritten as

$$\begin{aligned}\dot{V}(t) \leq& -\lambda_{\min}\left(\bar{Q}\right)\|\xi_{2-N}(t)\|^2 + 2\|\xi_{2-N}(t)\|\left\|\bar{P}G_1\right\|\|e\left(t\right)\| \\ &+ 2\|\xi_{2-N}(t)\|\left\|\bar{P}G_2\right\|\|\hat{e}\left(t\right)\| + 2\|\xi_{2-N}(t)\|\left\|\bar{P}G_3\right\|\|\tilde{x}\left(t\right)\| \\ \leq& -\left((1-\theta)\lambda_{\min}\left(\bar{Q}\right) - a\right)\|\xi_{2-N}(t)\|^2 + \frac{2}{a}\|PG_3\|^2\|\tilde{x}\left(t\right)\|^2 \\ &+ \frac{2}{a}\left(\left\|\bar{P}G_1\right\| I_0 + \left\|\bar{P}G_2\right\|\left(\theta\hat{\gamma} I_0 + \eta\right)\right)^2 N^2 e^{-2\alpha t} \\ \leq& -2\beta_1 V\left(t\right) + \beta_2 e^{-2\alpha t} + \beta_3\|\tilde{x}\left(t\right)\|^2,\end{aligned} \tag{3.16}$$

where

$$\begin{aligned}\beta_1 =& \left((1-\theta)\lambda_{\min}\left(\bar{Q}\right) - a\right)/2\lambda_{\max}\left(\bar{P}\right), \\ \beta_2 =& 2\left(\left\|\bar{P}G_1\right\| I_0 + \left\|\bar{P}G_2\right\|\left(\theta\hat{\gamma} I_0 + \eta\right)\right)^2 N^2/a, \\ \beta_3 =& 2\|PG_3\|^2/a.\end{aligned}$$

Choosing $a < (1-\theta)\lambda_{\min}\left(\bar{Q}\right)$, β_1 is positive. Based on the comparison theorem in [37] and (3.16), one can get that the solution of $V(t)$ satisfies

$$V\left(t\right) \leq e^{-2\beta_1 t} V\left(0\right) + \int_0^t e^{-2\beta_1(t-s)}(\beta_2 e^{-2\alpha s} + \beta_3\|\tilde{x}(s)\|^2) ds.$$

Define $\hat{C} = A - FC$ and let $P_{\hat{C}}$ and $P_{\hat{C}}^{-1}$ be the matrices such that $P_{\hat{C}}^{-1}\hat{C}P_{\hat{C}} = J_{\hat{C}}$, where $J_{\hat{C}}$ is the Jordan canonical form of the matrix $\hat{C}$. Then, it follows from Lemma 2.2 that for $0 \leq s \leq t$,

$$\begin{aligned}
&\left\| e^{-2\beta_1(t-s)}(\beta_2 e^{-2\alpha s} + \beta_3 \|\tilde{x}(s)\|^2) \right\| \\
\leq &\beta_2 e^{-2\beta_1(t-s)} e^{-2\alpha s} + \beta_3 \left(c_{\hat{C}} \left\| P_{\hat{C}} \right\| \left\| P_{\hat{C}}^{-1} \right\| \|\tilde{x}(0)\| \right)^2 e^{-2\beta_1(t-s)} e^{2a_{\hat{C}} s},
\end{aligned}$$

where $\max_i \mathrm{Re}(\lambda_i(\hat{C})) < a_{\hat{C}} < 0$ and $c_{\hat{C}}$ is a positive constant with respect to $\hat{C}$. Let

$$\begin{aligned}
a_1 &= V(0) + a_2 + a_3, \\
a_2 &= \beta_2 / \left| 2\alpha - 2\beta_1 \right|, \\
a_3 &= \beta_3 (c_{\hat{C}} \|P_{\hat{C}}\| \|P_{\hat{C}}^{-1}\| \|\tilde{x}(0)\|)^2 / \left| 2a_{\hat{C}} + 2\beta_1 \right|.
\end{aligned}$$

Then, one can further have

$$V(t) \leq a_1 e^{-2\beta_1 t} + a_2 e^{-2\alpha t} + a_3 e^{2a_{\hat{C}} t}.$$

Since $\beta_1 > 0, \alpha > 0$ and $a_{\hat{C}} < 0$, one has $\lim_{t\to\infty} V(t) = 0$. From the definition of V, one can see that $V(t) = 0$ if and only if $\|\xi_{2-N}(t)\| = 0$, which is equivalent to $\|\hat{x}_i(t) - \hat{x}_j(t)\| = 0, \forall (i,j) \in \mathcal{E}$. Besides, one has $\lim_{t\to\infty} \|\tilde{x}(t)\| = 0$, which is equivalent to $\lim_{t\to\infty} \|x_i(t) - \hat{x}_i(t)\| = 0, \forall i$. Therefore, consensus of the closed loop system (3.1) and (3.3) is achieved asymptotically.

In the following, we will show that Zeno behavior is excluded. Firstly, the communication function (3.6) is analyzed.

From the definition of $\xi(t)$, one has

$$\|\xi(t)\| = \|\xi_{2-N}(t)\| \leq \sqrt{V(t)/\lambda_{\min}(\bar{P})} = b_1 e^{-\beta_1 t} + b_2 e^{-\alpha t} + b_3 e^{a_{\hat{C}} t},$$

where $b_1 = \sqrt{a_1/\lambda_{\min}(\bar{P})}$, $b_2 = \sqrt{a_2/\lambda_{\min}(\bar{P})}$, and $b_3 = \sqrt{a_3/\lambda_{\min}(\bar{P})}$.

Let $u(t)$ be the column stack vector of $u_i(t)$. Then one has

$$\begin{aligned}
&\|(I_N \otimes B) u(t)\| \\
\leq &\left\| (cL \otimes BR^{-1}B^T P)(\hat{x}(t) + e(t)) \right\| + \left\| cI_N \otimes BR^{-1}B^T P \right\| \|\hat{e}(t)\| \\
\leq &c\|L\| \left\| BR^{-1}B^T P \right\| (\|\xi(t)\| + \|e(t)\|) + c \left\| BR^{-1}B^T P \right\| \|\hat{e}(t)\| \\
\leq &d_1 e^{-\beta_1 t} + d_2 e^{-\alpha t} + d_3 e^{a_{\hat{C}} t},
\end{aligned} \tag{3.17}$$

where

$$
\begin{aligned}
d_1 &= c(\|L\| + \theta\gamma)\left\|BR^{-1}B^TP\right\| b_1,\\
d_2 &= c\left(\|L\| I_0 + (\theta\gamma I_0 + \eta)\left\|BR^{-1}B^TP\right\|\right)\sqrt{N} + d_1 b_2/b_1,\\
d_3 &= c(\|L\| + \theta\gamma)\left\|BR^{-1}B^TP\right\| b_3.
\end{aligned}
$$

Furthermore, one has $\forall t \in [t^i_{\sigma_i}, t^i_{\sigma_i+1})$,

$$
\begin{aligned}
\dot{e}_i(t) =& Ae^{A(t-t^i_{\sigma_i})}\hat{x}_i(t^i_{\sigma_i}) - (A\hat{x}_i(t) + Bu_i(t) + F(y_i(t) - C\hat{x}_i(t)))\\
=& Ae_i(t) - Bu_i(t) - FC\tilde{x}_i(t)
\end{aligned}
$$

and $\|u_i(t)\| \le \|u(t)\|$. Thus, $\|\dot{e}_i(t)\| \le k_1 e^{-\beta_1 t} + k_2 e^{-\alpha t} + k_3 e^{a_{\hat{C}} t}$, where $k_1 = d_1$, $k_2 = d_2 + \|A\| I_0$, and $k_3 = d_3 + c_{\hat{C}}\|P_{\hat{C}}\|\|P^{-1}_{\hat{C}}\|\|FC\|\|\tilde{x}_i(0)\|$. Denote the latest communication time of agent i by $\hat{t}^*_i$, then the next communication time will not occur before $\|e_i(t)\| = I_0 e^{-\alpha t}$. Thus, a lower bound on the inter-communication time of the agent $i, \forall i$ is given by $\tau_i = t - \hat{t}^*_i$ that solves the equation $(k_1 e^{-\beta_1 \hat{t}^*_i} + k_2 e^{-\alpha \hat{t}^*_i} + k_3 e^{a_{\hat{C}} \hat{t}^*_i})\tau_i = I_0 e^{-\alpha t}$, which is equivalent to

$$
\left(k_1 e^{(-\beta_1+\alpha)\hat{t}^*_i} + k_2 + k_3 e^{(a_{\hat{C}}+\alpha)\hat{t}^*_i}\right)\tau_i = I_0 e^{-\alpha\tau_i}. \tag{3.18}
$$

Since $\beta_1 = ((1-\theta)\lambda_{\min}(\bar{Q}) - a)/2\lambda_{\max}(\bar{P})$, one has $\alpha < \min\{\beta_1, -\max_i \mathrm{Re}(\lambda_i(\hat{C}))\}$, then there exists a positive constant $\alpha < \beta_1$ and $-\max_i \mathrm{Re}(\lambda_i(\hat{C})) > -a_{\hat{C}} > \alpha > 0$. Thus, it is concluded that the solution τ_i of (3.18) is not smaller than τ^*, which is given by $(k_1 + k_2 + k_3)\tau^* = I_0 e^{-\alpha\tau^*}$ for all agent i, which is strictly positive. Therefore, Zeno behavior is excluded for the communication function (3.6).

Next, the control function (3.13) is analyzed. Note that the control function $g(\cdot,\cdot,\cdot)$ is not necessary continuous within two consecutive controller update instants (as stated in Remark 3.2). Define $\hat{t}^i_{\hat{\sigma}_i} = \cup_{j\in\mathcal{N}^+_i} t^j_{\sigma_j}, \hat{\sigma}_i \in \mathcal{Z}$ as the increasing sequence of communication time instants of agent i and its neighbors, *i.e.,* $0 = \hat{t}^i_0 < \hat{t}^i_1 < \hat{t}^i_2 < \cdots, \forall i$. Let $\hat{T}^i_{\hat{k}} = \hat{t}^i_{\hat{\sigma}_i} \cup T^i_k, \hat{k} \in \mathcal{Z}$, where $0 = \hat{T}^i_0 < \hat{T}^i_1 < \hat{T}^i_2 < \cdots, \forall i$. Then the set $\{\hat{T}^i_{\hat{k}}\}$ can be seen as the jump set of function $g(\hat{e}_i(t), z_i(t), t)$. Within two successive instants of $\hat{T}^i_{\hat{k}}$, one has $g(\hat{e}_i(t), z_i(t), t)$ is continuous. Define $\tau'_i = T^i_{k+1} - T^i_k$ as the time interval between two neighboring controller update instants of agent i, then for each $t \in [T^i_k, T^i_{k+1})$, one can get

i) if $T_k^i \in \{\hat{t}_{\hat{\sigma}_i}^i\}$ and $T_{k+1}^i \in \{\hat{t}_{\hat{\sigma}_i}^i\}$, one has $\tau_i' = T_{k+1}^i - T_k^i \geq \hat{t}_{\hat{\sigma}_i+1}^i - \hat{t}_{\hat{\sigma}_i}^i$. Since Zeno behavior is excluded for the communication function (3.5), one has that τ_i' is strictly positive in this case;

ii) if $T_k^i \in \{\hat{t}_{\hat{\sigma}_i}^i\}, T_{k+1}^i \notin \{\hat{t}_{\hat{\sigma}_i}^i\}$, there must exist a $\hat{\sigma}_i^* \in \mathcal{Z}$ such that $T_{k+1}^i \in (t_{\hat{\sigma}_i^*}^i, t_{\hat{\sigma}_i^*+1}^i)$. if $t_{\hat{\sigma}_i^*}^i \geq \hat{t}_{\hat{\sigma}_i+1}^i$, one has $\tau_i' \geq \hat{t}_{\hat{\sigma}_i+1}^i - \hat{t}_{\hat{\sigma}_i}^i$. Otherwise, $t_{\hat{\sigma}_i^*}^i = \hat{t}_{\hat{\sigma}_i}^i$, which means T_k^i and T_{k+1}^i are neighboring instants of $\hat{T}_{\hat{k}}^i$, thus $g\left(\hat{e}_i\left(t\right), z_i(t), t\right)$ is continuous for $t \in [T_k^i, T_{k+1}^i)$. Taking the derivative of $\hat{e}_i(t)$ on t, one has $\|\dot{\hat{e}}_i(t)\| \leq \|Az_i(t)\| \leq m_1 e^{-\beta_1 t} + m_2 e^{-\alpha t} + m_3 e^{a_{\hat{C}} t}$, where $m_1 = \|A\|\left(\hat{l}+N\sqrt{\hat{l}}\right)b_1$, $m_2 = \|A\|\left(\hat{l}+N\sqrt{\hat{l}}\right)(b_2+\sqrt{N}I_0)$, and $m_3 = \|A\|\left(\hat{l}+N\sqrt{\hat{l}}\right)b_3$. Besides, from (3.13), one observes that the next controller update time will not occur before $\|\hat{e}_i\left(t\right)\| = \eta e^{-\alpha t}$. Similar to the following analysis of the communication function (3.6), one can get that $\tau_i', \forall i$ is greater than or equal to the solution $\hat{\tau}^*$ of

$$\left(m_1 + m_2 + m_3\right)\hat{\tau}^* = \eta e^{-\alpha \hat{\tau}^*},$$

which is strictly positive;

iii) for the cases $T_k^i \notin \{\hat{t}_{\hat{\sigma}}^i\}, T_{k+1}^i \in \{\hat{t}_{\hat{\sigma}}^i\}$ and $T_k^i \notin \{\hat{t}_{\hat{\sigma}}^i\}, T_{k+1}^i \notin \{\hat{t}_{\hat{\sigma}}^i\}$, one can also get that $\tau_i', \forall i$ is strictly positive similar to the analysis of i) and ii).

Since there is a strictly positive lower bound on the neighboring controller update time instants in all cases, one can conclude that Zeno behavior is excluded for the control function (3.13). □

3.4 Example

In this section, a numerical example is given to verify the theoretical results. A network of 6 agents with communication graph $\mathcal{G}$ is shown in Figure 3.2. One can calculate that $\underline{\lambda}_R = 1$, then we choose $c = 1 > 1/(2\underline{\lambda}_R)$. The initial state $x_i(0)$ of each agent i is chosen randomly from the box $[-5, 5] \times [-5, 5]$, and the initial state of the observer $\hat{x}_i(0)$ of each agent i is chosen to be $[0, 0]^T, \forall i$.

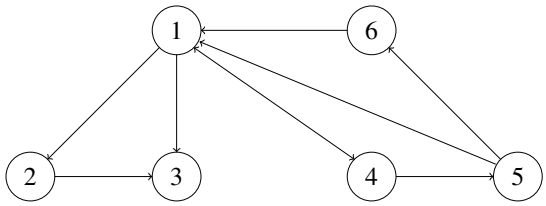

Figure 3.2: Communication graph for the MAS.

The system matrices are chosen as

$$A = \begin{bmatrix} -2 & 1 \\ 0.1 & 0.2 \end{bmatrix}, B = \begin{bmatrix} 0.8 \\ 0.5 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Given $Q = 5I_N$ and $R = 2I_N$, one can get $K = [0.6917, 1.8780]$ by solving the ARE (3.2). The feedback gain matrix F is chosen as $F = [-0.5, 2]^T$ such that $A - FC$ is Hurwitz. Given $\bar{Q} = 5I_{2N-2}$, then one can get $\lambda_{\max}(\bar{P}) = 6.7513$ by solving the Lyapunov function $\Pi^T\bar{P} + \bar{P}\Pi = -\bar{Q}$. Choosing $\theta = 0.4$ and $a = 0.001$, one has $\beta_1 = 0.444$. Then, we can choose $\alpha = 0.4 < \min\{\beta_1, -\max_i \mathrm{Re}\,(\lambda_i\,(A - FC))\}$.

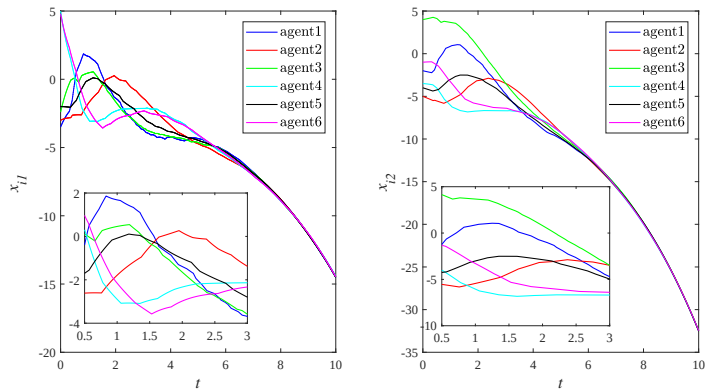

Figure 3.3: The evolution of x_{i1}, x_{i2} under controller (3.7).

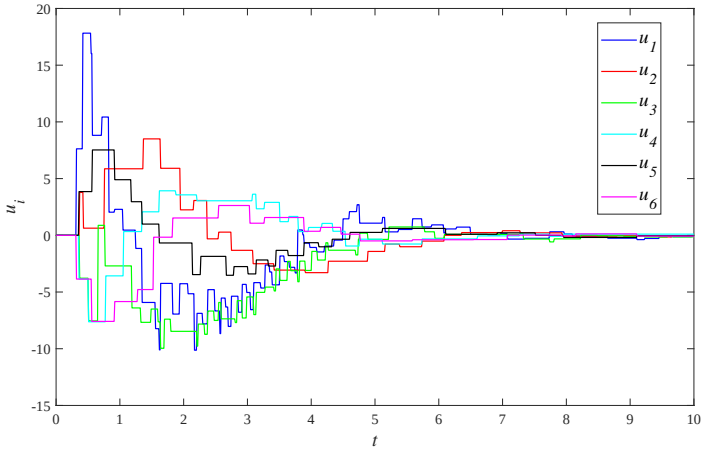

Figure 3.4: The evolution of controller (3.7).

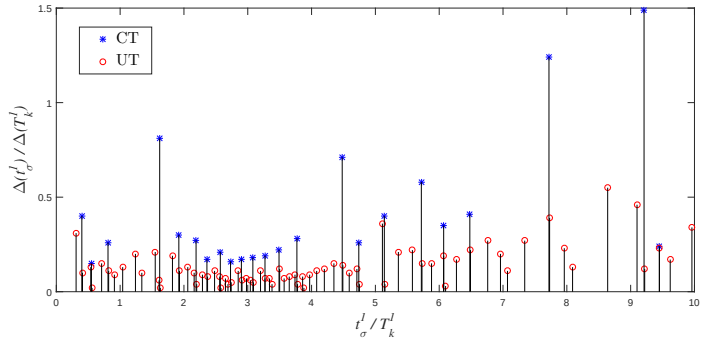

Figure 3.5: The communication/controller update time instants of agent 1.

The simulation results for the multi-agent systems (3.1), (3.3) with controller (3.7) are shown in Figures 3.3-3.5. The state trajectories are plotted in Figure 3.3, where x_{i1} and x_{i2} are the state components of agent i. The evolutions of controller (3.7) for each agent i are plotted in Figure 3.4. As an example, the communication/controller update time instants of agent 1 (labeled as t_σ^1/T_k^1) and the inter-communication/controller update interval of agent 1 (labeled as $\Delta(t_\sigma^1)/\Delta(T_k^1)$) is presented in Figure 3.5. One can see that Zeno behavior is excluded for both the communication and controller update. In

Figure 3.6, the state trajectories under the controller (2) proposed in [33] are depicted, and the evolutions of controller (2) are plotted in Figure 3.7.

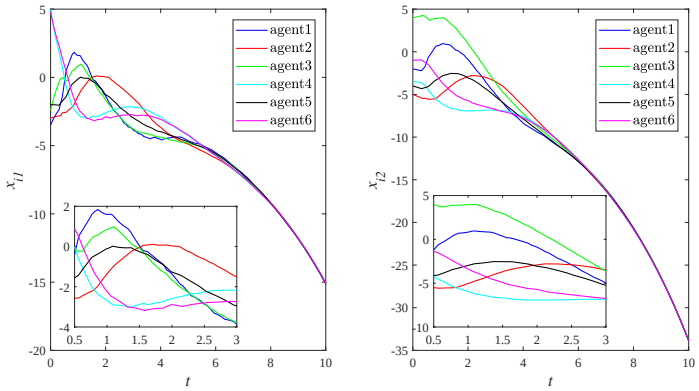

Figure 3.6: The evolution of x_{i1}, x_{i2} under controller (2) proposed in [33].

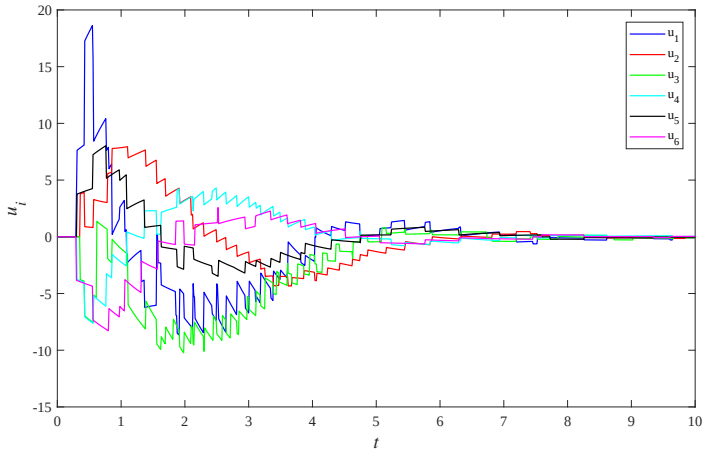

Figure 3.7: The evolution of controller (2) proposed in [33].

Table 3.1 summarises the simulation results for the controller (3.7) and the controller (2) of [33]. The amount of communication times and controller update times within the simulation time interval $[0, 30]$ are given for each agent

Table 3.1: The amount of communication times (CT) and controller update times (UT)

	CT	CT [33]	UT
Time Interval	[0, 30]	[0, 30]	[0, 30]
Agent 1	44	46	129
Agent 2	41	37	52
Agent 3	45	45	97
Agent 4	32	29	50
Agent 5	30	29	49
Agent 6	40	30	50
Overall CT	232	216	-

and the overall communication times are calculated. Since the controller proposed in [33] is updated continuously, the controller update times in [33] is ∞ for each agent, and thus the improvement in our case is straightforward. One can see that during the time interval $[0, 30]$, the overall communication times of controller (8) are slightly more than that of controller (2) proposed in [33]. Therefore, it is concluded that the proposed distributed ETC strategy reduces significantly the controller update times without significantly increasing the communication times.

3.5 Summary

In this chapter, distributed ETC of linear MAS was investigated. Firstly, in the communication side, an event-triggered communication scheme was proposed for each agent. Then, in the control side, a distributed event-triggered controller was implemented for each agent. It was proven that the consensus of the MAS is achieved asymptotically. It was also shown that Zeno behavior is excluded.

Chapter 4

Periodic Event-Triggered Consensus under Limited Data Rate

Chapter 3 proposed an event-triggered control (ETC) strategy for multi-agent consensus. In this chapter, a distributed periodic event-triggered control (PETC) strategy is developed for multi-agent systems (MAS), which further relaxes the requirement of continuous sensing and computation in ETC of Chapter 3. First, an approach on finding the maximum allowable sampling period (MASP) is proposed. Then, an asynchronous PETC strategy is proposed for the MAS. Finally, the constraint of limited data rate is taken into account. It is shown that exponential consensus can be achieved in all the cases.

4.1 Introduction

In recent years, ETC has been proposed as an alternative to sample-data control (which will be called time-triggered control (TTC) in the following), and various ETC strategies have been proposed for different kind of systems [14], [33], [40]–[50]. It should be pointed out that, although the literature on ETC is rich, limitations still remain and there are still some issues to be investigated. On one hand, most of the existing ETC strategies require the triggering condition to be monitored continuously[40]–[43], [47], [53] or partially continuously [49], [50], which may result in excessive use of sensing and computational resources. On the other hand, different from TTC, in which the devices (such as sensors, controllers and actuators) are activated only at the discrete sampling instants, in the ETC mechanism, it is necessary for all de-

vices to be activated all the time, which increases the energy consumption and thus reduces the lifespan of those devices. To address these problems, PETC[1] has been proposed as a solution [54]–[67].

In the area of PETC, most of the effort has been devoted to the stabilization of a single agent system [54]–[61], while the cooperation in the multi-agent case has not been considered to the same extent. Results on the design of PETC strategy for MAS with single-integrator dynamics are presented in [62]–[64]. For MAS with general linear dynamics or Lipchitz nonlinear dynamics, some recent results are reported in [65], [66] and [67], respectively. However, to the best of our knowledge, except for single-integrator MAS, the computation of MASP for MAS is typically not carried out [66] or given by solving a set of linear matrix inequalities [65], [67] without resulting in an explicit formula. We note that it is hard in general to find the explicit formula of MASP for PETC of general linear and nonlinear MAS. An overview of recent advances in ETC/PETC of MAS can be referred to the survey paper [68]. A comparison between TTC, ETC, and PETC is given in Figure 4.1.

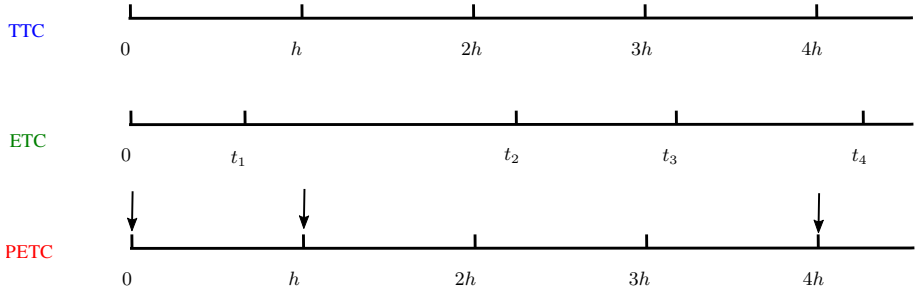

Figure 4.1: Differences between TTC, ETC and PETC, where the arrows in PETC represent the instants when the transmission actually happens.

Motivated by the above discussions, this chapter investigates PETC for MAS. The contributions are as follows.

(i) An approach on finding the explicit formula of the MASP is presented.

(ii) An asynchronous PETC strategy is formulated, where a communication function and a control function are designed for each agent to determine

[1]PETC combines the idea of TTC and ETC, in which the triggering condition is monitored periodically at the pre-defined discrete sampling time sequence, which allows to achieve a balance between TTC and ETC.

respectively whether or not the sampled state and control input should be transmitted at each sampling instant.

(iii) The constraint of limited data rate is considered. An observer-based encoder-decoder with finite-level quantization is designed for the sensor-controller communication and a finite-level quantizer with scaling is designed for the controller-actuator communication, such that certain constraint on the data rate is satisfied.

4.2 Problem Formulation

Consider a MAS with N agents, and the dynamics of each agent is given by

$$\dot{x}_i(t) = f(x_i(t)) + \hat{u}_i(t), \quad i = 1, 2, \ldots, N, \tag{4.1}$$

with

$$\hat{u}_i(t) = \sum_{j \in \mathcal{N}_i} \psi_{ij}(x_j(t_l) - x_i(t_l)), \quad t \in [t_l, t_{l+1}), \tag{4.2}$$

where $x_i \in \mathbb{R}^n$, $\hat{u}_i \in \mathbb{R}^n$ are respectively the state and the control input of the ith agent, f is a function representing the known nonlinearity of the system, ψ_{ij} is a control function to be designed, and $t_l = lh, l \in \mathbb{N}$ is the increasing sampling sequence. Here, $h > 0$ is the sampling period, which is common to all agents.

Assumption 4.1. *The communication graph $\mathcal{G}$ formed by the group of agents is undirected and connected.*

Assumption 4.2. *The function f is Lipschitz continuous with Lipschitz constant $\rho_1 > 0$ and $f(0) = 0$.*

Remark 4.1. *For the sake of simplicity, we analyze the nonlinear system (4.1). We note that it is straightforward to extend the results presented in this chapter to dynamical systems $\dot{x}_i(t) = f(x_i(t)) + Ax_i(t) + B\hat{u}_i(t)$ when the matrix pair (A, B) is stablizable.*

Definition 4.1. *The function $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is said to be in sector $[l_1, l_2]$ if for all $q \in \mathbb{R}^n$, one has*

$$(q^T\phi(q) - l_1 q^T q)(q^T\phi(q) - l_2 q^T q) \leq 0.$$

Assumption 4.3. *The functions $\psi_{ij} : \mathbb{R}^n \to \mathbb{R}^n, (i,j) \in \mathcal{E}$ are required to satisfy the following conditions:*

a) for any $x_i, x_j \in \mathbb{R}^n$, one has $\psi_{ij}(x_{ji}) = -\psi_{ji}(x_{ij})$, where $x_{ij} = x_i - x_j$;

b) there exists a constant $K > 0$ such that for any $x_{ij} \in \mathbb{R}^n$, one has $x_{ij}^T \psi_{ji}(x_{ij}) \geq K x_{ij}^T x_{ij}$; and

c) ψ_{ij} are globally Lipschitz continuous functions with Lipschitz constant $\rho_2 > 0$, that is, $\|\psi_{ij}(x') - \psi_{ij}(x'')\| \leq \rho_2 \|x' - x''\|$ for any $x', x'' \in \mathbb{R}^n$, and $\psi_{ij}(0) = 0, \forall (i,j) \in \mathcal{E}$.

According to item c), one has $\|\psi_{ji}(x_{ij})\| \leq \rho_2 \|x_{ij}\|$ and thus $x_{ij}^T \psi_{ji}(x_{ij}) \leq \rho_2 \|x_{ij}\|^2$. Combining with item b), one can further have $K\|x_{ij}\|^2 \leq x_{ij}^T \psi_{ji}(x_{ij}) \leq \rho_2 \|x_{ij}\|^2$. That is to say, the function ψ_{ji} is in sector $[K, \rho_2], \forall (j,i) \in \mathcal{E}$.

Let $\bar{x}(t) = \sum_{i=1}^N x_i(t)/N$ be the average state of all agents. Define the state error between agent i and the average as $\xi_i(t) = x_i(t) - \bar{x}(t), \forall i$ and the error vector as $\xi(t) = (\xi_1(t), \ldots, \xi_N(t))$.

Definition 4.2. *Consensus of the MAS (4.1) is said to be achieved exponentially, if there exist positive constants κ, μ such that the error vector satisfies*

$$\|\xi(t)\| \leq \kappa e^{-\mu t}, \forall t \geq 0.$$

The constant μ is called the convergence rate *and the constant κ is called the* convergence coefficient.

In this chapter, the problem under consideration is formulated as follows.

Problem 4.1. *Consider the MAS (4.1). Design PETC strategies for each agent i such that the consensus of the MAS (4.1) is achieved exponentially.*

4.3 Explicit Computation of MASP

In this section, an approach on finding the MASP is proposed. Before proceeding, the following notations are introduced.

Define

$$\begin{aligned} y_i(t_l) &= x_i(t_l), \\ y_i(t) &= y_i(t_l) + \int_{t_l}^{t} f(y_i(s))ds, \quad t \in (t_l, t_{l+1}), \end{aligned} \tag{4.3}$$

and $u_i(t) = \sum_{j\in\mathcal{N}_i} \psi_{ij}(y_j(t) - y_i(t))$. The sampling-induced errors of agent i are defined as

$$e_{x_i}(t) = y_i(t) - x_i(t), e_{u_i}(t) = \hat{u}_i(t) - u_i(t), \forall i. \tag{4.4}$$

Then, one has $\|e_{x_i}(t_l)\| = 0$ and $\|e_{u_i}(t_l)\| = 0$ for all t_l.

The control input (4.2) can then be rewritten as

$$\hat{u}_i(t) = \sum_{j\in\mathcal{N}_i} \psi_{ij}(x_{ji}(t) + e_{x_{ji}}(t)) + e_{u_i}(t),$$

where $x_{ji}(t) = x_j(t) - x_i(t)$ and $e_{x_{ji}}(t) = e_{x_j}(t) - e_{x_i}(t)$.

Since the graph $\mathcal{G}$ is undirected and $\psi_{ij}(x_{ji}) = -\psi_{ij}(x_{ij})$, one has

$$\dot{\xi}_i(t) = f(x_i(t)) - \frac{1}{N}\sum_{i=1}^{N} f(x_i(t)) + \hat{u}_i(t), \quad i = 1, 2, \ldots, N, \tag{4.5}$$

where $\hat{u}_i(t)$ can be equivalently rewritten as

$$\hat{u}_i(t) = \sum_{j\in\mathcal{N}_i} \psi_{ij}(\xi_{ji}(t) + e_{x_{ji}}(t)) + e_{u_i}(t), \tag{4.6}$$

where $\xi_{ji}(t) = \xi_j(t) - \xi_i(t)$.

One can see that the exponential consensus of the MAS (4.1) is achieved if and only if the stability of the error system (4.5) is achieved exponentially. Therefore, in the following, the stability of the error system (4.5) is investigated.

Let ξ, e_x, e_u be the concatenated vectors of ξ_i, e_{x_i}, e_{u_i}, respectively, where we dropped the time argument t for notation convenience. Define $\|z\|' := d\|z\|/dt, \forall z$. Then, we get the following propositions.

Proposition 4.1. *Let the function $V : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be $V(z) = z^2$, then the derivative of $V(\|\xi\|)$ along the trajectory of (4.5) satidfies*

$$\dot{V}(\|\xi\|) \leq -\hat{L}\|\xi\|^2 + \frac{1}{a_1}\gamma^2\|e_x\|^2 + \frac{1}{a_1}\|e_u\|^2, \tag{4.7}$$

where $\hat{L} = 2K\lambda_2(L) - 4\rho_1 - 2a_1,\ \gamma = \rho_2\sqrt{2\hat{l}^2 + 2N\hat{l}},\ \hat{l} = \max_i\{d_{ii}\} \leq N-1,\ a_1 > 0$, and $\lambda_2(L)$ is the algebraic connectivity of the Laplacian matrix L.

Proof. The proof is provided in Appendix. □

Proposition 4.2. *For all $t \in [t_l, t_{l+1})$, the following inequalities hold*

$$\|e_x\|' \leq 2\gamma\|\xi\| + (\rho_1 + 2\gamma)\|e_x\| + \sqrt{2}\|e_u\|, \tag{4.8}$$
$$\|e_u\|' \leq \sqrt{2}\rho_1\gamma\|\xi\| + \sqrt{2}\rho_1\gamma\|e_x\|. \tag{4.9}$$

Proof. The proof is provided in Appendix. □

Now, we provide an explicit computation of the MASP h^* for the MAS (4.1). Firstly, we introduce two auxiliary functions ϕ_1, ϕ_2. Let $\phi_1(\tau) : \mathbb{R}_{\geq 0} \to \mathbb{R}$ and $\phi_2(\tau) : \mathbb{R}_{\geq 0} \to \mathbb{R}$ be such that

$$\frac{d\phi_1(\tau)}{d\tau} = -m_1\phi_1^2(\tau) - m_2\phi_1(\tau) - m_3, \tag{4.10}$$
$$\frac{d\phi_2(\tau)}{d\tau} = -n_1\phi_2^2(\tau) - n_2\phi_2(\tau) - n_3, \tag{4.11}$$

where

$$\begin{aligned} m_1 &= \frac{4\gamma^3}{{a_1}^2} + a_2, \quad m_2 = 2\rho_1 + 4\gamma + (\hat{L} - 2a_1), \quad m_3 = \gamma + a_2, \\ n_1 &= \frac{\gamma^2\rho_1^2}{a_1^2} + \frac{2\gamma\rho_1^2}{a_2}, \quad n_2 = \hat{L} - 2a_1, \quad n_3 = \frac{2\gamma}{a_2} + 1 \end{aligned} \tag{4.12}$$

for some $a_2 > 0$. The constants $\hat{L}, a_1, \gamma$ in (4.12) are given in Proposition 4.1. To state our main results, we further introduce the following function:

$$\mathcal{T}(x, y, z) = \begin{cases} \frac{2}{r}\arctan\left(\frac{r}{y}\right) & \text{if } 4xz > y^2, \\ \frac{2}{y}, & \text{if } 4xz = y^2, \\ \frac{2}{r}\text{arctanh}\left(\frac{r}{y}\right), & \text{if} 4xz < y^2, \end{cases}$$

where

$$r := \sqrt{|4xz - y^2|}.$$

Then, the MASP h^* is selected as

$$h^* = \min\{\mathcal{T}(m_1, m_2, m_3), \mathcal{T}(n_1, n_2, n_3)\}. \tag{4.13}$$

Theorem 4.1. *Consider the MAS (4.1) with the control input (4.2), where the sampling period h is chosen from $h \in (0, h^*)$. Suppose Assumptions 4.1-4.3 hold with $K > 2\rho_1/\lambda_2(L)$. Then, consensus of the MAS (4.1) is achieved exponentially with convergence rate $K\lambda_2(L) - 2\rho_1 - 2a_1$ and convergence coefficient $\|\xi(0)\|$, where $a_1 \in (0, (K\lambda_2(L) - 2\rho_1)/2)$.*

Proof. The proof is provided in Appendix. □

Remark 4.2. *In this chapter, the communication graph is assumed to be undirected. It is worth to point out that the results obtained are applicable to the case of a directed graph with a spanning tree when a linear system model is considered.*

Remark 4.3. *Two constants, i.e., a_1, a_2 are introduced for the computation of MASP. It can be seen that both the MASP and the convergence rate are related to the constant a_1. Moreover, a small value of a_1 means faster convergence, however, the MASP will be smaller. Therefore, a_1 can be used as a trade-off between the rate of communication and the rate of convergence. In addition, from (4.10) and (4.11), one can see that a small value of a_2 means a bigger $\mathcal{T}(m_1, m_2, m_3)$ and a smaller $\mathcal{T}(n_1, n_2, n_3)$, while a big value of a_2 means a smaller $\mathcal{T}(m_1, m_2, m_3)$ and a bigger $\mathcal{T}(n_1, n_2, n_3)$. Therefore, a_2 can be used as a trade-off between $\mathcal{T}(m_1, m_2, m_3)$ and $\mathcal{T}(n_1, n_2, n_3)$ such that the maximum MASP can be achieved. A way of tuning parameters a_1, a_2 is given as follows: i) choose a_1 such that the requirement of convergence performance is satisfied; and ii) find a_2 such that $\mathcal{T}(m_1, m_2, m_3) = \mathcal{T}(n_1, n_2, n_3)$.*

4.4 PETC

In Section 4.3, an approach on finding the MASP is proposed for TTC. To further reduce the rates of communication and controller update, in this section, an asynchronous PETC strategy is developed.

4.4.1 Communication and control functions

For each agent i, let $t^i_{\sigma_i}, \sigma_i \in \mathbb{N}$ be the increasing sequence of communication time instants at which x_i is transmitted and $\{t^i_{\sigma_i}\}$ be the set of communication instants. On the sensor side, each agent implements an estimator of itself using

the most recently transmitted data $x_i(t^i_{\sigma_i})$, that is,

$$\begin{aligned} \dot{\hat{y}}_i(t) &= f(\hat{y}_i(t)), \quad t \in [t^i_{\sigma_i}, t^i_{\sigma_i+1}), \\ \hat{y}_i(t^i_{\sigma_i}) &= x_i(t^i_{\sigma_i}). \end{aligned} \tag{4.14}$$

For agent i, the communication error at the sampling instant t_l is defined as $e_{y_i}(t_l) = \hat{y}_i(t_l) - x_i(t_l)$, and the communication time sequence $t^i_{\sigma_i}$ is generated by

$$t^i_{\sigma_i+1} = \inf_{t_l} \left\{ t_l > t^i_{\sigma_i} : h_1\left(t_l, e_{y_i}\left(t_l\right)\right) \geq 0 \right\}, \tag{4.15}$$

where

$$h_1\left(t_l, e_{y_i}\left(t_l\right)\right) = \left\| e_{y_i}\left(t_l\right) \right\| - c_1 e^{-\alpha t_l} \tag{4.16}$$

with constants $c_1 > 0, \alpha > 0$. Without loss of generality, we assume $t^i_0 = 0, \forall i$.

From the definition of $e_{y_i}(t_l)$, one can see that only the sampled state $x_i(t_l)$ is required to implement the communication function (4.16) for each agent i. At each t_l, $\|e_{y_i}(t_l)\|$ is compared to a certain threshold, *i.e.,* $c_1 e^{-\alpha t_l}$, and the sampled state $x_i(t_l)$ is transmitted if and only if $\|e_{y_i}(t_l)\|$ is bigger than or equal to that threshold. Moreover, one can see from (4.15) that the communication function (4.16) is detected only at the sampling instants t_l. Thus, $\{t^i_{\sigma_i}\} \in \{t_l = lh\}, \forall i$.

On the control side, each agent implements estimators of itself $\hat{y}_i(t)$ as well as its neighbors $\hat{y}_j(t)$ based on the received states, that is,

$$\begin{aligned} \dot{\hat{y}}_j(t) &= f(\hat{y}_j(t)), \quad t \in [t^j_{\sigma_j}, t^j_{\sigma_j+1}), \\ \hat{y}_j(t^j_{\sigma_j}) &= x_j(t^j_{\sigma_j}), \quad j \in \mathcal{N}_i^+. \end{aligned} \tag{4.17}$$

The distributed event-triggered controller for agent i is designed as

$$\hat{u}_i(t) = \sum_{j \in \mathcal{N}_i} \psi_{ij}(\hat{y}_j(T^i_k) - \hat{y}_i(T^i_k)), \quad t \in [T^i_k, T^i_{k+1}), \tag{4.18}$$

where $T^i_k, k \in \mathbb{N}$ is the controller update time sequence.

Define $q_i(t) = \sum_{j \in \mathcal{N}_i} \psi_{ij}(\hat{y}_j(t) - \hat{y}_i(t))$. Then the control error at sampling instant t_l is defined as $\hat{e}^l_{u_i}(t_l) = q_i(T^i_k) - q_i(t_l)$. Let $\{T^i_k\}$ be the set of controller update instants, in which T^i_{k+1} is generated by

$$T^i_{k+1} = \inf_{t_l} \left\{ t_l > T^i_k : h_2\left(t_l, \hat{e}^l_{u_i}(t_l)\right) \geq 0 \right\}, \tag{4.19}$$

where

$$h_2\left(t_l, \hat{e}^l_{u_i}(t_l)\right) = \left\|\hat{e}^l_{u_i}(t_l)\right\| - c_2 e^{-\alpha t_l} \tag{4.20}$$

with constant $c_2 > 0$. Without loss of generality, we assume $T^i_0 = 0, \forall i$.

From the definition of $q_i(t)$, $e^l_{u_i}(t_l)$ and (4.17), one can see that only the transmitted states of agent i itself and its neighbors, *i.e.,* $x_j(t^j_{\sigma_j}), j \in \mathcal{N}^+_i$ are required to implement the control function (4.20) for each agent i. Moreover, one can see from (4.19) that the control function (4.20) is detected only at the sampling instants t_l. Thus, $\{T^i_k\} \in \{t_l = lh\}, \forall i$.

4.4.2 Convergence result

Define $\hat{e}_{u_i}(t) = q_i(t_l) - q_i(t), t \in [t_l, t_{l+1})$ and $e_{y_i}(t) = \hat{y}_i(t) - y_i(t)$, where y_i is defined in (4.3). Then, one has $e_{y_i}(t_l) = \hat{y}_i(t_l) - y_i(t_l) = \hat{y}_i(t_l) - x_i(t_l)$. For convenience, we extend the control error as piecewise constant signals as $\hat{e}^l_{u_i}(t) = \hat{e}^l_{u_i}(t_l), t \in [t_l, t_{l+1})$. Combining the definition of $e_{x_i}(t)$ given in (4.4) and $\hat{e}^l_{u_i}(t), \hat{e}_{u_i}(t), e_{y_i}(t)$, (4.18) can be rewritten as

$$\begin{aligned}\hat{u}_i &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(x_{ji} + e_{x_{ji}} + e_{y_{ji}}) + \hat{e}^l_{u_i} + \hat{e}_{u_i}, \\ &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(\xi_{ji} + e_{x_{ji}} + e_{y_{ji}}) + \hat{e}^l_{u_i} + \hat{e}_{u_i},\end{aligned} \tag{4.21}$$

where $e_{y_{ji}} = e_{y_j} - e_{y_i}$. Since the graph $\mathcal{G}$ is undirected and $\psi_{ij}(x_{ji}) = -\psi_{ij}(x_{ij})$, one has $\sum_{i=1}^N \hat{u}_i = \sum_{i=1}^N \hat{e}^l_{u_i}$. Then, one can further have

$$\dot{\xi}_i = f(x_i) - \frac{1}{N}\sum_{i=1}^N f(x_i) + \hat{u}_i - \frac{1}{N}\sum_{i=1}^N \hat{e}^l_{u_i}. \tag{4.22}$$

Let $\xi, e_x, e_y, \hat{e}^l_u, \hat{e}_u$ be the concatenated vectors of $\xi_i, e_{x_i}, e_{y_i}, \hat{e}^l_{u_i}, \hat{e}_{u_i}$, respectively. Then, we get the following proposition.

Proposition 4.3. *The derivative of $V(\|\xi\|)$ along the trajectories of (4.22) satisfies*

$$\dot{V}(\|\xi\|) \leq -\hat{L}\|\xi\|^2 + \frac{4}{a_1}\gamma^2\|e_x\|^2 + \frac{4}{a_1}\gamma^2\|e_y\|^2 + \frac{4}{a_1}\left\|\hat{e}^l_u\right\|^2 + \frac{2}{a_1}\|\hat{e}_u\|^2. \tag{4.23}$$

In addition, for all $t \in [t_l, t_{l+1})$, the following inequalities hold

$$\|e_x\|' \le 3\gamma\|\xi\| + (\rho_1 + 3\gamma)\|e_x\| + 3\gamma\|e_y\| + \sqrt{3}\|\hat{e}_u\| + \sqrt{3}\left\|\hat{e}_u^l\right\|,$$
$$\|e_u\|' \le \sqrt{3}\rho_1\gamma(\|\xi\| + \|e_x\| + \|e_y\|).$$

Proof. The proof is similar to Propositions 4.1 and 4.2, and hence omitted. □

Similar to Section 4.3, one can derive that the MASP for PETC is given by

$$\hat{h}^* = \min\{\mathcal{T}(\hat{m}_1, \hat{m}_2, \hat{m}_3), \mathcal{T}(\hat{n}_1, \hat{n}_2, \hat{n}_3)\}, \tag{4.24}$$

where

$$\begin{aligned}
&\hat{m}_1 = \frac{9\gamma^3 + 2a_1}{{a_1}^2} + a_2, \quad \hat{m}_2 = 2\rho_1 + 6\gamma + (\hat{L} - 2a_1), \quad \hat{m}_3 = 4\gamma + a_2,\\
&\hat{n}_1 = \frac{3\gamma^2\rho_1^2 + a_1}{a_1^2} + \frac{3\gamma\rho_1^2}{a_2}, \quad \hat{n}_2 = \hat{L} - 2a_1, \quad \hat{n}_3 = \frac{3\gamma}{a_2} + 2.
\end{aligned} \tag{4.25}$$

Then, we can get the following result.

Theorem 4.2. *Consider the MAS (4.1) with the control input (4.18), where the sampling period h is chosen from $h \in (0, \hat{h}^*)$. Suppose Assumptions 4.1-4.3 hold with $K > 2\rho_1/\lambda_2(L)$. The communication time sequence and the controller update time sequence are given respectively by (4.15) and (4.19) with $\alpha > K\lambda_2(L) - 2\rho_1 - 2a_1$. Then, consensus of the MAS (4.1) is achieved exponentially with convergence rate $K\lambda_2(L) - 2\rho_1 - 2a_1$ and convergence coefficient $\sqrt{d_1}$, where*

$$\begin{aligned}
a_1 &\in \Big(0, \frac{K\lambda_2(L) - 2\rho_1}{2}\Big),\\
d_1 &= \|\xi(0)\|^2 + \frac{\beta_1 + \beta_2}{2\alpha - 2K\lambda_2(L) + 4\rho_1 + 4a_1},\\
\beta_1 &= Nc_1^2\gamma^2\Big(\frac{4}{a_1} + 3(3\gamma + \rho_1^2)\Big)e^{2(\rho_1+\alpha)h},\\
\beta_2 &= Nc_2^2\Big(\frac{4}{a_1} + 3\gamma\Big)e^{2\alpha h}.
\end{aligned}$$

Proof. The proof is provided in Appendix. □

Remark 4.4. *In this chapter, the approach on finding the MASP is related to [69]. In [69], a sufficient condition is provided on the existence of an explicit formula for MASP that guarantees the stabilization of a networked control system. In this chapter, we consider the consensus of MAS in a networked control system framework. We note that the technical difficulty of our approach lies in the design of the asynchronous event-triggered communication and control strategy such that the sufficient condition provided in [69] is satisfied. This problem is non-trivial since we do not assume a passivity property on the nonlinear dynamics f. On the contrary, we overcome this problem by introducing estimators and suitably-designed communication and control functions for sensor-controller communication and controller-actuator communication, respectively.*

4.5 PETC under Limited Data Rate

The results obtained in Section 4.4 are based on perfect communication. However, any real network has limited channel capacity, which means that only a finite number of bits of information can be transmitted. Based on this observation, in this section, we further investigate PETC under limited data rate.

Before proceeding, the definition of a uniform quantizer is required. A finite-level uniform quantizer is a map $Q_{\Delta,M} : \mathbb{R} \to \mathbb{R}$ such that

$$Q_{\Delta,M}(y) = \begin{cases} \Delta \left\lfloor \frac{y}{\Delta} + \frac{1}{2} \right\rfloor, & \text{if } 0 \le y \le M\Delta, \\ M\Delta, & \text{if } y > M\Delta, \\ -Q_{\Delta,M}(-y), & \text{if } y < 0. \end{cases} \tag{4.26}$$

where $\Delta > 0$ is the quantization interval, and M is the number of quantization levels. Note that as long as the quantizer is not saturated ($|y| \le M\Delta$), the quantization error is always bounded by $\Delta/2$, namely, $|Q_{\Delta,M}(y) - y| \le \Delta/2$.

The above definition of the scalar-valued uniform quantizer can be easily extended to its vector-valued counterpart. For any $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, define the vector uniform quantizer $Q_{\Delta,M}(x) : \mathbb{R}^n \to \mathbb{R}^n$ to be $Q_{\Delta,M}(x) := (Q_{\Delta,M}(x_1), \ldots, Q_{\Delta,M}(x_n))$.

It is assumed that both sensor-controller communication and controller-actuator communication are conducted via a network. Therefore, both state quantization and input quantization are considered. The desired event-triggered communication and control structure of agent i is shown in Figure

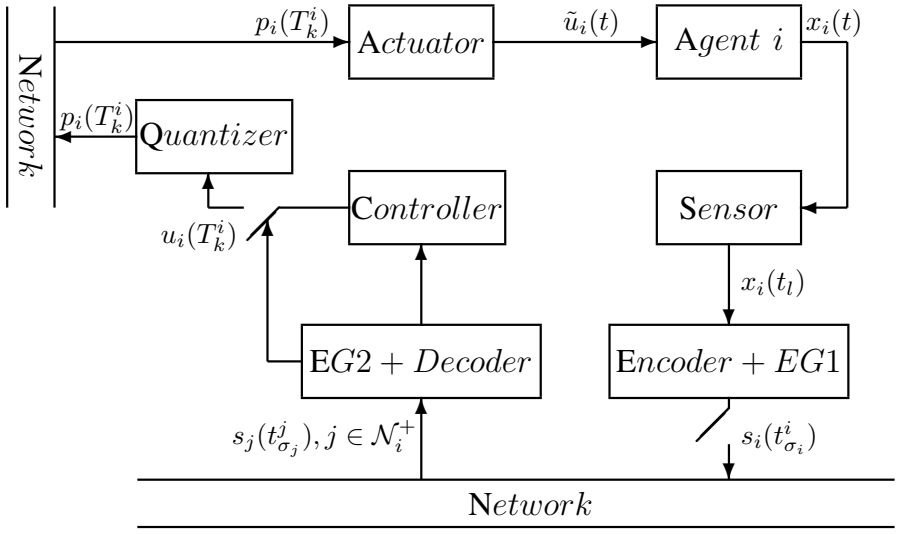

Figure 4.2: The structure of PETC for agent i.

4.2. Due to input quantization, the dynamics of agent i is written as:

$$\dot{x}_i(t) = f(x_i(t)) + \tilde{u}_i(t), \quad i = 1, 2, \ldots, N, \tag{4.27}$$

where $\tilde{u}_i$ is the quantized control input with scaling, which will be specified later.

4.5.1 Communication and control structure

Without loss of generality, we consider agent i and its neighbors $j \in \mathcal{N}_i$ to illustrate the idea, and only show how information flows within agent i (from sensor to controller and from controller to actuator) and how agent i obtains information from agent $j, j \in \mathcal{N}_i$.

Step 1: The sensor samples the state of agent i at the sampling instants t_l ($t_{l+1} - t_l \equiv h$ and h is the sampling period), and then the sampled state $x_i(t_l)$ is sent to the event generator 1 (EG1), which determines whether or not the sampled state should be transmitted. On the sensor side, agent i implements an estimator of itself, which is given by (4.14). The communication time instants $t^i_{\sigma_i}$ are determined by

$$t^i_{\sigma_i+1} = \inf_{t_l} \left\{ t_l > t^i_{\sigma_i} : h_1\left(t_l, e_{y_i}\left(t_l\right)\right) \geq 0 \cup t_l - t^i_{\sigma_i} \geq Rh \right\}, \tag{4.28}$$

where R is a positive integer, and $h_1(\cdot,\cdot)$ is defined in (4.16). Compared to (4.15), a constraint on the maximum time interval between two successive communications is enforced. It means that each agent can endure at most Rh units of time without communication. The reason for this constraint comes from the quantization error induced by the dynamic encoder-decoder, which is defined below. The good news is that, in general, R can be chosen arbitrarily large as long as it has a finite value.

Step 2: When the transmitted state $x_i(t^i_{\sigma_i})$ is generated, the encoder s_i of agent i is activated:

$$\begin{cases} \hat{x}_i(t^i_{-1}) =0, \\ s_i(t^i_{\sigma_i}) =Q_{\Delta,M_1^i}\left(\dfrac{x_i(t^i_{\sigma_i})-\hat{x}_i(t^i_{\sigma_i-1})-\int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds}{g_1(t^i_{\sigma_i})}\right), \\ \hat{x}_i(t^i_{\sigma_i}) =\hat{x}_i(t^i_{\sigma_i-1})+\int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds+g_1(t^i_{\sigma_i})s_i(t^i_{\sigma_i}), \end{cases} \tag{4.29}$$

where $x_i(t^i_{\sigma_i})$ and $s_i(t^i_{\sigma_i})$ are the input and the output of the encoder, respectively. The function $g_1(t)>0$ is a scaling function which will be defined later. In the above, $\hat{x}_i(t^i_{\sigma_i})$ is the internal state of the encoder, and $Q_{\Delta,M_1^i}(\cdot)$ is a finite-level uniform quantizer defined in (4.26) with quantization interval Δ and quantization level M_1^i.

Step 3: After the signal $s_j(t^j_{\sigma_j}), j\in\mathcal{N}_i^+$ of agent i itself or its neighbors $j\in\mathcal{N}_i$ is received on the control side, the decoder ς_j^i will be activated:

$$\begin{cases} \varsigma_j^i(t^j_{-1}) =0, \\ \varsigma_j^i(t^j_{\sigma_j}) =\varsigma_j^i(t^j_{\sigma_j-1})+\int_{t^j_{\sigma_j-1}}^{t^j_{\sigma_j}} f\left(\varsigma_j^i(s)\right)ds+g_1(t^j_{\sigma_j})s_j(t^j_{\sigma_j}), \quad j\in\mathcal{N}_i^+, \end{cases} \tag{4.30}$$

where $\varsigma_j^i(t^j_{\sigma_j})$ is the output of the decoder. From (4.29) and (4.30), one has $\hat{x}_i(t^i_{\sigma_i})=\varsigma_i^j(t^i_{\sigma_i}), \forall j\in\mathcal{N}_i^+$.

Step 4: The decoded states $\varsigma_j^i(t^j_{\sigma_j}), j\in\mathcal{N}_i^+$ are sent to event generator 2 (EG2) to determine whether or not the controller is updated. Similar to Section 4.3, each agent implements estimators of itself $\tilde{y}_i(t)$ as well as its neighbors

$\tilde{y}_j(t)$ based on decoded states, that is,

$$\begin{aligned}\dot{\tilde{y}}_j(t) &= f(\tilde{y}_j(t)), \quad t \in [t^j_{\sigma_j}, t^j_{\sigma_j+1}),\\ \tilde{y}_j(t^j_{\sigma_j}) &= \varsigma^i_j(t^j_{\sigma_j}), \quad j \in \mathcal{N}^+_i.\end{aligned} \tag{4.31}$$

Then, the controller for agent i is designed as

$$\hat{u}_i(t) = \sum_{j\in\mathcal{N}_i} \psi_{ij}(\tilde{y}_j(T^i_k) - \tilde{y}_i(T^i_k)), \quad t \in [T^i_k, T^i_{k+1}), \tag{4.32}$$

where T^i_k is the controller update time sequence. Let $\tilde{q}_i(t) = \sum_{j\in\mathcal{N}_i} \psi_{ij}(\tilde{y}_j(t) - \tilde{y}_i(t))$. Redefine in this section $\hat{e}^l_{u_i}(t_l) = \tilde{q}_i(T^i_k) - \tilde{q}_i(t_l)$ and $\hat{e}_{u_i}(t) = \tilde{q}_i(t_l) - \tilde{q}_i(t)$. Then, the controller update time sequence T^i_k is determined by

$$T^i_{k+1} = \inf_{t_l} \left\{ t_l > T^i_k : h_2\left(t_l, \hat{e}^l_{u_i}(t_l)\right) \geq 0 \cup t_l - t^i_{\sigma_i} \geq Rh \right\}, \tag{4.33}$$

where $h_2(\cdot,\cdot)$ is defined in (4.20). Similar to (4.28), an additional constraint on the maximum time interval between two successive controller updates is enforced.

Step 5: When the transmitted control input $\hat{u}_i(T^i_k)$ is generated, it is quantized with scaling. That is,

$$p_i(T^i_k) = Q_{\Delta,M_2}\left(\frac{\hat{u}_i(T^i_k)}{g_2(T^i_k)}\right). \tag{4.34}$$

The function $Q_{\Delta,M_2}(\cdot)$ is a finite-level uniform quantizer defined in (4.26) with quantization interval Δ and quantization level M_2, and $g_2(t) > 0$ is a scaling function which will be defined later.

Now, the quantized control input in (4.27) is given by

$$\tilde{u}_i(t) = g_2(T^i_k)p_i(T^i_k), \quad t \in [T^i_k, T^i_{k+1}). \tag{4.35}$$

4.5.2 Convergence result

Define the state quantization induced errors of agent i as $e^x_{q_i}(t^i_{\sigma_i}) = \varsigma^i_i(t^i_{\sigma_i}) - x_i(t^i_{\sigma_i})$. Let $e^y_{q_i}(t) = \tilde{y}_i(t) - \hat{y}_i(t)$. Then one can get $e^y_{q_i}(t^i_{\sigma_i}) = e^x_{q_i}(t^i_{\sigma_i})$ since $\tilde{y}_i(t^i_{\sigma_i}) = \varsigma^i_i(t^i_{\sigma_i})$ and $\hat{y}_i(t^i_{\sigma_i}) = x_i(t^i_{\sigma_i})$. Define the control quantization induced errors of agent i as $e^u_{q_i}(T^i_k) = \tilde{u}_i(T^i_k) - \hat{u}_i(T^i_k)$. For convenience, we

extend the control error and the control quantization induced error respectively as piecewise constant signals as $\hat{e}^l_{u_i}(t) = \hat{e}^l_{u_i}(t_l), t \in [t_l, t_{l+1})$ and $e^u_{q_i}(t) = \tilde{u}_i(T^i_k) - \hat{u}_i(T^i_k), t \in [T^i_k, T^i_{k+1})$.

Recall that $e_{x_i}(t) = y_i(t) - x_i(t), e_{y_i}(t) = \hat{y}_i(t) - y_i(t)$. Combining the definition of $e_{x_i}(t), e_{y_i}(t), \hat{e}_{u_i}(t), \hat{e}^l_{u_i}(t_l), e^y_{q_i}(t)$ and $e^u_{q_i}(t)$, (4.35) can be rewritten as

$$\begin{aligned}\tilde{u}_i(t) &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(\tilde{y}_j(T^i_k) - \tilde{y}_i(T^i_k)) + e^u_{q_i}(t)\\ &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(\hat{y}_j(T^i_k) + e^y_{q_j}(t) - \hat{y}_i(T^i_k) - e^y_{q_i}(t)) + e^u_{q_i}(t)\\ &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(\xi_{ji}(t) + e_{x_{ji}}(t) + e_{y_{ji}}(t) + e^y_{q_{ji}}(t)) + \hat{e}^l_{u_i}(t) + \hat{e}_{u_i}(t) + e^u_{q_i}(t).\end{aligned} \tag{4.36}$$

Since the graph $\mathcal{G}$ is undirected and $\psi_{ij}(x_{ji}) = -\psi_{ij}(x_{ij})$, one has $\sum_{i=1}^N \tilde{u}_i = \sum_{i=1}^N (\hat{e}^l_{u_i} + e^u_{q_i})$. Then, one can further have

$$\dot{\xi}_i = f(x_i) - \frac{1}{N}\sum_{i=1}^N f(x_i) + \tilde{u}_i - \frac{1}{N}\sum_{i=1}^N (\hat{e}^l_{u_i} + e^u_{q_i}). \tag{4.37}$$

Let e^y_q, e^u_q be the concatenated vectors of $e^y_{q_i}, e^u_{q_i}$, respectively. Then, we get the following proposition.

Proposition 4.4. *The derivative of $V(\|\xi\|)$ along the trajectories of (4.37) satisfies*

$$\begin{aligned}\dot{V}(\|\xi\|) \le &-\hat{L}\|\xi\|^2 + \frac{9}{a_1}\gamma^2\|e_x\|^2 + \frac{9}{a_1}\gamma^2\|e_y\|^2 + \frac{9}{a_1}\gamma^2\left\|e^y_q\right\|^2\\ &+ \frac{6}{a_1}\left\|\hat{e}^l_u\right\|^2 + \frac{3}{a_1}\|\hat{e}_u\|^2 + \frac{6}{a_1}\left\|e^u_q\right\|^2.\end{aligned} \tag{4.38}$$

In addition, for all $t \in [t_l, t_{l+1})$, the following inequalities hold

$$\begin{aligned}\|e_x\|' &\le 4\gamma\|\xi\| + (\rho_1 + 4\gamma)\|e_x\| + 4\gamma\|e_y\| + 4\gamma\left\|e^y_q\right\|\\ &\quad + 2\|\hat{e}_u\| + 2\left\|\hat{e}^l_u\right\| + 2\left\|e^u_q\right\|,\\ \|e_u\|' &\le 2\rho_1\gamma(\|\xi\| + \|e_x\| + \|e_y\| + \left\|e^y_q\right\|).\end{aligned}$$

Proof. The proof is similar to Propositions 4.1 and 4.2, and hence omitted. □

The MASP for PETC under limited data rate is given by

$$\tilde{h}^* = \min\{\mathcal{T}(\tilde{m}_1, \tilde{m}_2, \tilde{m}_3), \mathcal{T}(\tilde{n}_1, \tilde{n}_2, \tilde{n}_3)\}, \tag{4.39}$$

where

$$\begin{aligned}
&\tilde{m}_1 = \frac{16\gamma^3 + 4a_1}{{a_1}^2} + a_2, \quad \tilde{m}_2 = 2\rho_1 + 8\gamma + (\hat{L} - 2a_1), \quad \tilde{m}_3 = 9\gamma + a_2, \\
&\tilde{n}_1 = \frac{4\gamma^2\rho_1^2 + 2a_1}{a_1^2} + \frac{4\gamma\rho_1^2}{a_2}, \quad \tilde{n}_2 = \hat{L} - 2a_1, \quad \tilde{n}_3 = \frac{4\gamma}{a_2} + 3.
\end{aligned} \tag{4.40}$$

With the designed encoder-decoder (4.29), (4.30) and the control quantizer (4.34), the following result is obtained.

Theorem 4.3. *Consider the MAS (4.27) with the quantized control input (4.35), where the sampling period h is chosen from $h \in (0, \tilde{h}^*)$. Suppose Assumptions 4.1-4.3 hold with $K > 2\rho_1/\lambda_2(L)$. The communication time sequence and the controller update time sequence are given by (4.28) and (4.33) with $\alpha > 0$, respectively. Let the encoder-decoder be given by (4.29) and (4.30), and the scaling functions $g_1(t) = c_3 e^{-\eta t}$ and $g_2(t) = c_4 e^{-\eta t}$ with constants $c_3 > 0,\ c_4 > 0$, and $0 < \eta < \min\{K\lambda_2(L) - 2\rho_1 - 2a_1, \alpha\}$, where $a_1 \in (0, (K\lambda_2(L) - 2\rho_1)/2)$. The quantization levels M_1^i and M_2 satisfy*

$$M_1^i \geq \max\left\{ \frac{c_1 e^{(\rho_1+\alpha)h} + h(Ce^{\eta Rh} + c_2 + c_4\sqrt{n}\Delta/2)e^{\eta h}}{c_3\Delta} + \frac{\sqrt{n}}{2} e^{(\rho_1+\eta)Rh}, \frac{\|x_i(0)\|}{c_3\Delta} \right\}, \forall i \tag{4.41}$$

and

$$M_2 \geq \frac{C}{c_4\Delta}, \tag{4.42}$$

where $C = \gamma(\sqrt{d_2} + \sqrt{N}c_1 + \sqrt{Nn}c_3\Delta e^{(\eta+\rho_1)Rh}/2)$, and γ is defined in Proposition 4.1. Then, consensus of the MAS (4.27) is achieved exponentially with convergence rate η and convergence coefficient $\sqrt{d_2}$, where

$$\begin{aligned}
d_2 &= \xi(0)^2 + \frac{\beta_3 + \beta_4}{|2\alpha - 2K\lambda_2(L) + 4\rho_1 + 4a_1|} + \frac{k_1}{2K\lambda_2(L) - 4\rho_1 - 4a_1 - 2\eta}, \\
\beta_3 &= Nc_1^2\gamma^2\Big(\frac{9}{a_1} + 4(4\gamma + \rho_1^2)\Big)e^{2(\rho_1+\alpha)h},
\end{aligned}$$

$$\begin{aligned}\beta_4 &= Nc_2^2\Big(\frac{6}{a_1}+4\gamma\Big)e^{2\alpha h},\\ k_1 &= \frac{\beta_3 n c_3^2\Delta^2 e^{2(\eta+\rho_1)Rh}}{4c_1^2}+\frac{\beta_4 n c_4^2\Delta^2 e^{2\eta Rh}}{4c_2^2}.\end{aligned}$$

Proof. The proof is provided in Appendix. □

Remark 4.5. *There is literature considering ETC of MAS under limited data rate [70]–[73]. In [70], [71], MAS with single-integrator dynamics are considered, and ETC and self-triggered control are proposed, respectively. Discrete-time MAS with general linear dynamics are studied in [72], while continuous-time MAS with general linear dynamics are investigated in [73]. In both [72] and [73], synchronous ETC strategy is designed. However, In this chapter, nonlinear MAS with PETC strategy is instead considered. Moreover, to the best of our knowledge, an asynchronous PETC strategy under limited date rate is proposed for the first time.*

Remark 4.6. *In this chapter, three different communication and control strategies, i.e., TTC, PETC, and PETC under limited data rate are considered and different triggering conditions are proposed, respectively. In TTC, a universal clock is used to trigger both state and control input transmission. In PETC, despite of a universal clock, communication and control conditions are further designed to determine whether or not the state and the control input should be transmitted, respectively. When limited data rate is considered for PETC, an additional clock is required for both state transmission and controller update. The reason for introducing this additional clock is to ensure that the quantization error is limited.*

Remark 4.7. *For TTC, the rate of communication (for both state and control input) is determined by the sampling period h. However, for PETC, we note that the rate of communication is not mainly determined by the sampling period h, but the communication and control conditions (4.15) and (4.19). For PETC under limited data rate, the rate of communication is determined by both the sampling period h and the communication and control functions (4.28) and (4.33), since the period of the additional clock is a constant times of h.*

4.6 Example

In this section, a simulation example is provided to validate the effectiveness of the theoretical results.

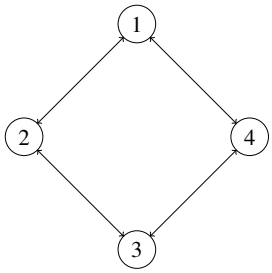

Figure 4.3: Communication graph for the MAS.

Consider a MAS consisting of 4 single-link robot arms, the communication graph among them is shown in Figure 4.3. The initial state $x_i(0)$ of each robot arm i is chosen randomly from the box $[-5;5]\times[-5;5]$. The dynamics of the robot arm i is described by (4.1), where

$$f(x_{i_1}(t),x_{i_2}(t))=\begin{pmatrix} x_{i_2} \\ -\sin(x_{i_1}) \end{pmatrix}.$$

and $x_i = (x_{i1},x_{i2})^T$. Clearly, $f(x_i)$ is Lipschitz with a Lipschitz constant $\rho_1 = 1$. The control input for agent i is designed as

$$\hat{u}_i(t)=2\sum_{j\in\mathcal{N}_i}(x_j(t_l)-x_i(t_l)),\quad t\in[t_l,t_{l+1}).$$

One can verify that Assumption 4.2 holds with $K = 2$ and $\rho_2 = 2$. According to Proposition 4.1, one can calculate $\hat{L}=4-2a_1,\gamma=4\sqrt{2}$. Choosing $a_1 = 0.8$ such that $\hat{L}-2a_1 = 4-4a_1 = 0.8 > 0$, then one can get $h^* = 3.18\times10^{-3}$. The sampling period h is chosen as $h = 3\times10^{-3}$. The simulation results for TTC are shown in Figure 4.4, where the evolution of the states x_{i1} and x_{i2} is plotted.

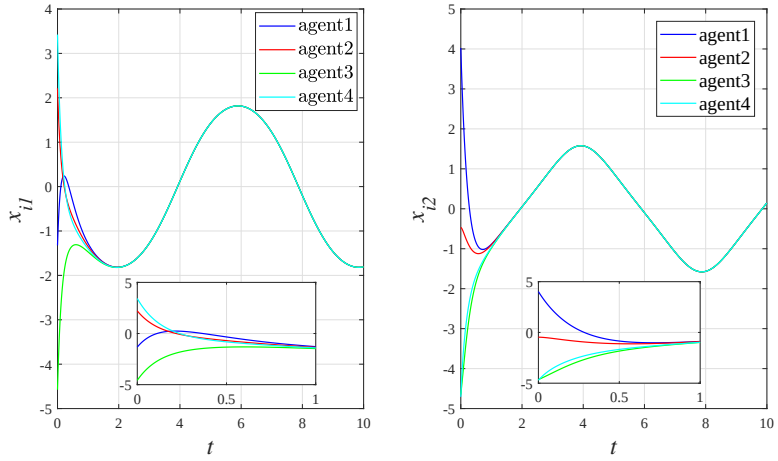

Figure 4.4: The evolution of x_{i1}, x_{i2} under TTC.

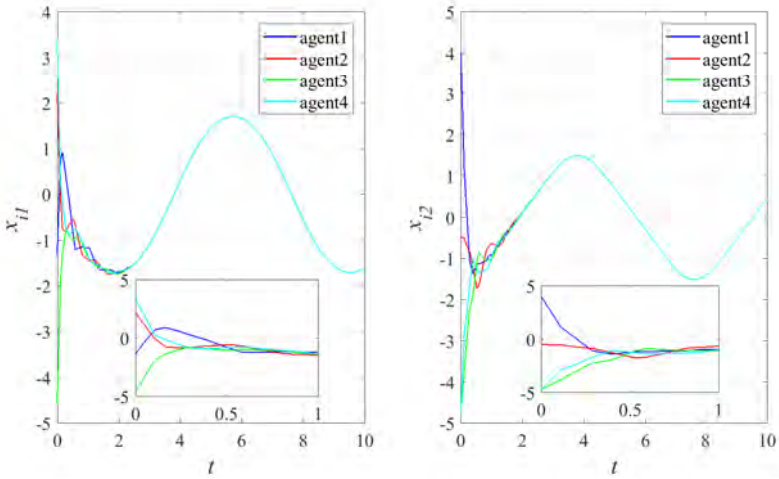

Figure 4.5: The evolution of x_{i1}, x_{i2} under PETC.

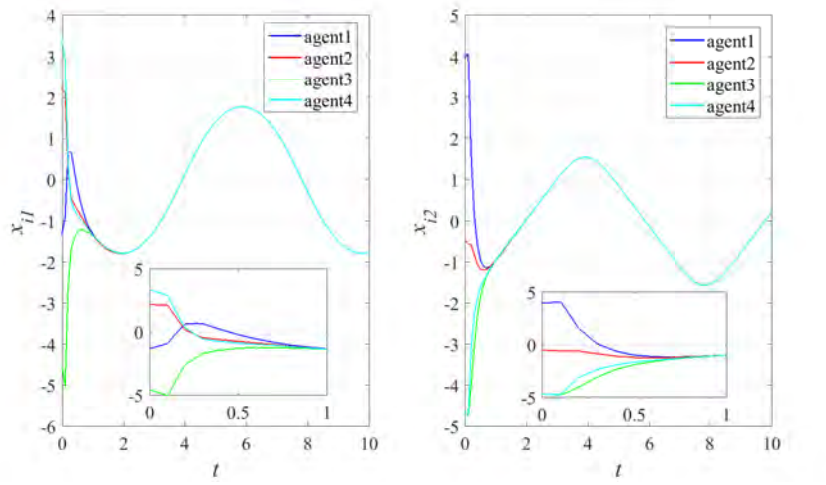

Figure 4.6: The evolution of x_{i1}, x_{i2} under PETC*.

For PETC and PETC under limited data rate (PETC*), the sampling period h is chosen as $h = 1 \times 10^{-3}$. Given $c_1 = c_2 = 4$, then one can chose $\alpha = 2.01$. The simulation results for PETC are shown in Figure 4.5. Choosing $R = 100$, and the parameters for the scaling functions g_1, g_2 as $c_3 = c_4 = 4$ and $\eta = 1.8 < \min\{K\lambda_2(L) - 2\rho_1 - 2a_1, \alpha\}$, the simulation results for PETC* are plotted in Figure 4.6.

Table 4.1 summarises the communication times and controller update times for each agent under the three cases. One can see that the introduction of communication and control functions significantly reduces the rates of communication and controller update. One can also see that consensus of the MAS can be achieved under data rate constraint.

Table 4.1: The amount of communication times (CT) and controller update times (UT)

CT/UT	TTC	PETC	PETC*
agent1	3333/3333	31/90	99/106
agent2	3333/3333	30/87	99/107
agent3	3333/3333	29/87	100/105
agent4	3333/3333	32/90	99/108

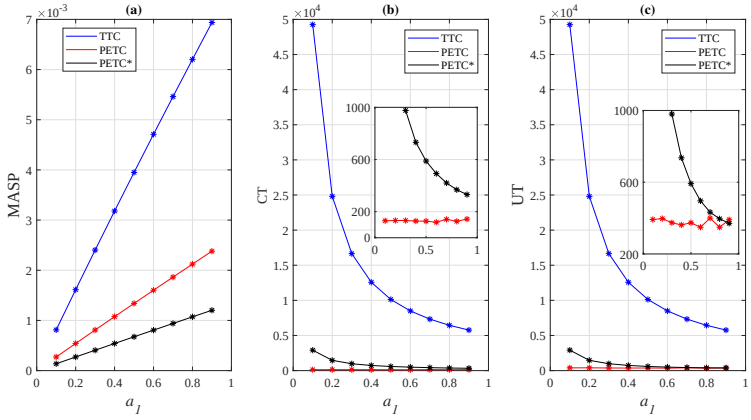

Figure 4.7: The effects of tuning a_1 on the MASP, the communication times and the controller update times.

Next, the effects of tuning parameter a_1 on the MASP and the total communication/controller update times are shown in Figure 4.7. From Figure 4.7(a), one can see that the MASP increases as a_1 increases for all the three cases and $h^* > \hat{h}^* > \tilde{h}^*$ when choosing the same a_1. From Figure 4.7(b) and (c), one can see that for both TTC and PETC*, the communication/controller update times are decreasing as the sampling period is increasing. However, the communication/controller update times for PETC remain almost the same no matter what the sampling period is. That is because the communication/controller update instants for PETC are mainly determined by the communication/control conditions (which remain the same for all simulations). We note that the relationship of the MASP for the three cases is $h^* > \hat{h}^* > \tilde{h}^*$. However, the relationship of the total communication/controller update times is PETC $<$ PETC* $<$ TTC, which again verifies that the introduction of communication and control functions reduces significantly the rates of communication and controller update.

4.7 Summary

In this chapter, we considered the PETC of MAS. First, TTC was considered and an approach on finding the MASP was established. Then, a PETC strategy was formulated. Finally, the PETC strategy was investigated under the con-

straint of limited data rate. It was proved that exponential consensus can be achieved in all cases.

Appendix

Proof of Proposition 4.1

From (4.5), one has

$$\begin{aligned}\|\xi_i\|' =&\frac{d\|\xi_i\|}{dt} = \frac{\xi_i^T\dot{\xi}_i}{\|\xi_i\|} = \frac{\xi_i^T\left(f(x_i) - \frac{1}{N}\sum\limits_{i=1}^{N} f(x_i) + \hat{u}_i\right)}{\|\xi_i\|}\\ =&\frac{\xi_i^T\left(f(x_i) - f(\bar{x}) + f(\bar{x}) - \frac{1}{N}\sum\limits_{i=1}^{N} f(x_i) + \hat{u}_i\right)}{\|\xi_i\|}\\ \leq&\rho_1\|\xi_i\| + \frac{1}{N}\sum_{i=1}^{N}\rho_1\|\xi_i\| + \frac{\xi_i^T\hat{u}_i}{\|\xi_i\|},\end{aligned} \tag{4.43}$$

and thus

$$\xi_i^T\dot{\xi}_i \leq \rho_1\|\xi_i\|^2 + \frac{\rho_1}{N}\|\xi_i\|\sum_{i=1}^{N}\|\xi_i\| + \xi_i^T\hat{u}_i.$$

Then,

$$\begin{aligned}\|\xi\|' = \frac{\sum\limits_{i=1}^{N}\xi_i^T(t)\dot{\xi}_i}{\|\xi\|} \leq& \frac{\sum\limits_{i=1}^{N}\rho_1\|\xi_i\|^2 + \frac{\rho_1}{N}\|\xi_i\|\sum\limits_{i=1}^{N}\|\xi_i\| + \xi_i^T\hat{u}_i}{\|\xi\|}\\ \leq& 2\rho_1\|\xi\| + \frac{\xi^T\hat{u}}{\|\xi\|}.\end{aligned} \tag{4.44}$$

Differentiating $V(\|\xi\|)$ along the trajectories of (4.44), one has

$$\begin{aligned}\dot{V}(\|\xi\|) =&2\|\xi\|\,\|\xi\|' \leq 2\|\xi\|\left(2\rho_1\|\xi\| + \frac{\xi^T\hat{u}}{\|\xi\|}\right) = 4\rho_1\|\xi\|^2 + 2\xi^T\hat{u}\\ =&4\rho_1\|\xi\|^2 + 2\sum_{i=1}^{N}\xi_i^T\left(\delta_i + \hat{\delta}_i + e_{u_i}\right),\end{aligned} \tag{4.45}$$

where $\delta_i = \sum_{j\in\mathcal{N}_i}\psi_{ij}(\xi_{ji})$ and $\hat{\delta}_i = \sum_{j\in\mathcal{N}_i}\psi_{ij}(\xi_{ji}+e_{x_{ji}}) - \delta_i$.

From Assumption 4.2, one has $x_{ij}^T\psi_{ji}(x_{ij}) \geq Kx_{ij}^Tx_{ij} = K\|x_{ij}\|^2, \forall x_{ij} \in \mathbb{R}^n$, then

$$\sum_{(i,j)\in\mathcal{E}} \xi_{ij}^T\psi_{ji}(\xi_{ij}) \geq K\sum_{(i,j)\in\mathcal{E}}\|\xi_{ij}\|^2 = 2K\xi^T\left(L\otimes I_n\right)\xi. \tag{4.46}$$

According to the definition of ξ, one has $1_{Nn}^T\xi = 0$, and if the graph $\mathcal{G}$ is connected, one has $\xi^T\left(L\otimes I_n\right)\xi \geq \lambda_2\left(L\right)\|\xi\|^2$ and $\lambda_2\left(L\right) > 0$ [74]. Then,

$$\begin{aligned}\sum_{i=1}^N \xi_i^T\delta_i =& = \frac{1}{2}\sum_{i=1}^N \xi_i^T\sum_{j\in\mathcal{N}_i}\psi_{ij}(\xi_{ji}) + \frac{1}{2}\sum_{j=1}^N \xi_j^T\sum_{i\in\mathcal{N}_j}\psi_{ji}(\xi_{ij})\\ &= -\frac{1}{2}\sum_{i=1}^N \xi_{ij}^T\sum_{j\in\mathcal{N}_i}\psi_{ji}(\xi_{ij}) \leq -K\lambda_2(L)\|\xi\|^2.\end{aligned}$$

In addition, the function $\psi_{ij}, \forall(i,j) \in \mathcal{E}$ is globally Lipschitz, and then one has

$$\begin{aligned}\sum_{i=1}^N \left\|\hat{\delta}_i\right\|^2 \leq& \sum_{i=1}^N \Big\|\sum_{j\in\mathcal{N}_i}\psi_{ij}(\xi_{ji}+e_{x_{ji}}) - \sum_{j\in\mathcal{N}_i}\psi_{ij}(\xi_{ji})\Big\|^2\\ \leq& \rho_2^2\sum_{i=1}^N\Big(\sum_{j\in\mathcal{N}_i}\left\|e_{x_{ji}}\right\|\Big)^2,\end{aligned} \tag{4.47}$$

where

$$\begin{aligned}\sum_{i=1}^N\Big(\sum_{j\in\mathcal{N}_i}\left\|e_{x_{ji}}\right\|\Big)^2 &\leq \sum_{i=1}^N\left(d_{ii}\|e_{x_i}\| + \sum_{j\in\mathcal{N}_i}\|e_{x_j}\|\right)^2\\ &\leq 2\hat{l}^2\sum_{i=1}^N\|e_{x_i}\|^2 + 2\sum_{i=1}^N\Big(\sum_{j\in\mathcal{N}_i}\|e_{x_j}\|\Big)^2\\ &\leq 2\hat{l}^2\|e_x\|^2 + 2\sum_{i=1}^N\hat{l}\|e_x\|^2 = (2\hat{l}^2+2N\hat{l})\|e_x\|^2.\end{aligned}$$

Using the inequality $2xy \leq ax^2 + 1/ay^2, \forall a > 0$, (4.45) can be rewritten as

$$\begin{aligned}\dot{V}(\|\xi\|) \leq& -\left(2K\lambda_2(L) - 4\rho_1 - 2a_1\right)\|\xi\|^2\\ &+ \frac{1}{a_1}\left(\rho_2^2(2\hat{l}^2+2N\hat{l})\|e_x\|^2 + \|e_u\|^2\right)\end{aligned} \tag{4.48}$$

for any $a_1 > 0$. □

Proof of Proposition 4.2

According to (4.4), one has $\dot{e}_{x_i}(t) = \dot{y}_i(t) - \dot{x}_i(t) = f(y_i(t)) - f(x_i(t)) - \hat{u}_i(t)$, which jumps at $t = t_l$. Thus, for $\forall t \in [t_l, t_{l+1})$, one has that e_{x_i} is continuous. Similar to Proposition 4.1, one can get

$$\|e_x(t)\|' = \frac{\sum_{i=1}^{N} e_{x_i}^T(t)\dot{e}_{x_i}(t)}{\|e(t)\|} \leq \rho_1 \|e_x(t)\| - \frac{e_x^T(t)\hat{u}(t)}{\|e_x(t)\|} \leq \rho_1 \|e_x(t)\| + \|\hat{u}(t)\|. \tag{4.49}$$

Since

$$\begin{aligned}
\|\hat{u}(t)\|^2 =& \sum_{i=1}^{N} \|\hat{u}_i(t)\|^2 = \sum_{i=1}^{N} \Big\| \sum_{j\in\mathcal{N}_i} \psi_{ij}(\xi_{ji}(t) + e_{x_{ji}}(t)) + e_{u_i}(t) \Big\|^2 \\
\leq& 2\sum_{i=1}^{N} \Big\| \sum_{j\in\mathcal{N}_i} \psi_{ij}(\xi_{ji}(t) + e_{x_{ji}}(t)) \Big\|^2 + 2\sum_{i=1}^{N} \|e_{u_i}(t)\|^2 \\
\leq& 2\sum_{i=1}^{N} \Big(\sum_{j\in\mathcal{N}_i} \rho_2 \|\xi_{ji}(t) + e_{x_{ji}}(t)\| \Big)^2 + 2\sum_{i=1}^{N} \|e_{u_i}(t)\|^2 \\
\leq& 4\rho_2^2 \sum_{i=1}^{N} \Big\{ \Big(\sum_{j\in\mathcal{N}_i} \|\xi_{ji}\| \Big)^2 + \Big(\sum_{j\in\mathcal{N}_i} \|e_{x_{ji}}\| \Big)^2 \Big\} + 2\sum_{i=1}^{N} \|e_{u_i}(t)\|^2 \\
\leq& 4\rho_2^2 (2\hat{l}^2 + 2N\hat{l}) \left(\|\xi(t)\|^2 + \|e_x(t)\|^2 \right) + 2\|e_u(t)\|^2,
\end{aligned}$$

then one has

$$\|\hat{u}(t)\| = \left(\|\hat{u}(t)\|^2 \right)^{\frac{1}{2}} \leq 2\gamma (\|\xi(t)\| + \|e_x(t)\|) + \sqrt{2}\|e_u(t)\|. \tag{4.50}$$

Substituting (4.50) into (4.49) yields (4.8).

Moreover, according to (4.4), one has $\nabla_t e_{u_i}(t) = \nabla_t \hat{u}_i(t) - \nabla_t u_i(t)$, where $\hat{u}_i(t)$ is a constant for all $t \in [t_l, t_{l+1})$ and $u_i(t)$ jumps at $t = t_l$. Thus, for all $t \in [t_l, t_{l+1})$, e_{u_i} is continuous and $\nabla_t e_{u_i}(t) = -\nabla_t u_i(t)$. Then,

$$\|e_u(t)\|' = \frac{\sum_{i=1}^{N} e_{u_i}^T(t)\nabla_t e_{u_i}(t)}{\|e_u(t)\|} = -\frac{\sum_{i=1}^{N} e_{u_i}^T(t)\nabla_t u_i(t)}{\|e_u(t)\|} \leq \|\nabla_t u(t)\|. \tag{4.51}$$

According to Assumption 4.2, the function ψ_{ij} is locally Lipschitz, and then one can further derive

$$\begin{aligned}
\|\nabla_t u(t)\| =& \Big(\sum_{i=1}^{N}\Big\|\nabla_t \sum_{j\in\mathcal{N}_i}\psi_{ij}(y_j(t)-y_i(t))\Big\|^2\Big)^{\frac{1}{2}}\\
=& \Big(\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}\nabla_{y_j-y_i}\psi_{ij}(y_j-y_i)^T(\dot{y}_j(t)-\dot{y}_i(t))\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \Big(\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}\rho_2 f(y_i(t))-f(y_j(t))\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \rho_2\rho_1\Big(\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}y_i(t)-y_j(t)\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \rho_2\rho_1\Big(\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}x_i(t)+e_{x_i}(t)-x_j(t)-e_{x_j}(t)\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \rho_2\rho_1\Big(\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}\xi_{ij}(t)+e_{x_{ij}}(t)\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \rho_2\rho_1\Big(2\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}\xi_{ij}(t)\Big\|^2+2\sum_{i=1}^{N}\Big\|\sum_{j\in\mathcal{N}_i}e_{x_{ij}}(t)\Big\|^2\Big)^{\frac{1}{2}}\\
\leq& \sqrt{2}\rho_1\gamma\left(\|\xi(t)\|+\|e_x(t)\|\right),
\end{aligned} \tag{4.52}$$

which corresponds to (4.9). □

Proof of Theorem 4.1

Define $\tau:\mathbb{R}_{\geq 0}\to[0,h]$ as

$$\tau(t)=t-kh,\quad t\in[kh,(k+1)h),\quad k=0,1,\ldots, \tag{4.53}$$

where $h\in(0,h^*)$.

Let $z:=(\xi,e_x,e_u)$. Define the Lyapunov function candidate as

$$R_1(z)=V(\|\xi\|)+\frac{1}{a_1}\gamma\phi_1(\tau)V(\|e_x\|)+\frac{1}{a_1}\phi_2(\tau)V(\|e_u\|). \tag{4.54}$$

Since $h < h^*$, by properly choosing $\phi_1(0), \phi_2(0) > 0$, one can conclude from Lemma 2.4 that $\phi_1(\tau) \geq 0$ and $\phi_2(\tau) \geq 0$ for all $\tau \in [0, h]$. Therefore, the Lyapunov function $R_1(z)$ is positive definite.

Taking the derivative of $R_1(z)$ on $\forall t \in [t_l, t_{l+1})$, one has

$$\begin{aligned}\dot{R}_1(z) =&\dot{V}(\|\xi\|) + \frac{1}{a_1}\gamma\dot{\phi}_1(\tau)\|e_x\|^2 + 2\frac{1}{a_1}\gamma\phi_1(\tau)\,\|e_x\|\,\|e_x\|' \\ &+ \frac{1}{a_1}\dot{\phi}_2(\tau)\|e_u\|^2 + 2\frac{1}{a_1}\phi_2(\tau)\,\|e_u\|\,\nabla_t\,\|e_u\|\,.\end{aligned} \tag{4.55}$$

Substituting (4.7), (4.8), (4.9) and (4.10), (4.11) into (4.55), one can further have

$$\begin{aligned}\dot{R}_1(z) \leq& -\hat{L}\|\xi\|^2 + \frac{1}{a_1}\gamma^2\|e_x\|^2 + \frac{1}{a_1}\|e_u\|^2 + 2\frac{1}{a_1}\gamma\phi_1(\tau)\|e_x\|(\gamma\|\xi\| + \|e_u\| \\ &+ (\rho_1+\gamma)\|e_x\| + \frac{1}{a_1}\gamma(-m_1\phi_1^2(\tau) - m_2\phi_1(\tau) - m_3)\|e_x\|^2 \\ &+ 2\frac{1}{a_1}\phi_2(\tau)\|e_u\|\,(\rho_1\gamma\|\xi\| + \rho_1\gamma\|e_x\|) \\ &+ \frac{1}{a_1}(-n_1\phi_2^2(\tau) - n_2\phi_2(\tau) - n_3)\|e_u\|^2 \\ =& -(\hat{L} - 2a_1)\|\xi\|^2 - \frac{1}{a_1}(\hat{L} - 2a_1)\gamma\phi_1(\tau)\|e_x\|^2 \\ &- \frac{1}{a_1}(\hat{L} - 2a_1)\phi_2(\tau)\|e_u\|^2 \\ \leq& -(\hat{L} - 2a_1)R_1(z).\end{aligned} \tag{4.56}$$

Since $a_1 \in (0, (K\lambda_2(L) - 2\rho_1)/2)$, one has $\hat{L} - 2a_1 > 0$, which implies $\dot{R}_1(z) < 0, \forall t \in [t_l, t_{l+1})$ when $\|\xi\| \neq 0$. Based on the comparison theorem in [37] and (4.56), one can get that the solution of $R_1(z)$ satisfies

$$R_1(z(t)) \leq e^{-(\hat{L}-2a_1)(t-t_l)}R_1(z(t_l)), \forall t \in [t_l, t_{l+1}).$$

Moreover, at the jump, *i.e.,* when $t = t_l^+$, one has $\|\xi(t_l^+)\| = \|\xi(t_l)\|$, $\|e_x(t_l^+)\| = 0 \leq \|e_x(t_l)\|$ and $\|e_u(t_l^+)\| = 0 \leq \|e_u(t_l)\|$. That is to say, $R_1(z)$ is non-increasing during the jump. Then, one can further have $R_1(z(t)) \leq e^{-(\hat{L}-2a_1)t}R_1(z(0)) = e^{-(\hat{L}-2a_1)t/2}\|\xi(0)\|^2, \forall t$ and thus $\|\xi(t)\| \leq \sqrt{R_1(z(t))} \leq \|\xi(0)\|e^{-(\hat{L}-2a_1)t/2} = \|\xi(0)\|e^{-(K\lambda_2(L)-2\rho_1-2a_1)t}, \forall t$. That is, consensus of the nonlinear MAS (4.1) is achieved exponentially with convergence rate $K\lambda_2(L) - 2\rho_1 - 2a_1$ and convergence coefficient $\|\xi(0)\|$. □

Proof of Theorem 4.2

Define $\hat{\tau} : \mathbb{R}_{\geq 0} \to [0, h]$ as

$$\hat{\tau}(t) = t - kh, \quad t \in [kh, (k+1)h), \quad k = 0, 1, \ldots, \tag{4.57}$$

where $h \in (0, \hat{h}^*)$.

Let $\hat{z} := (\xi, e_x, \hat{e}_u)$. Define the Lyapunov function candidate

$$R_2(\hat{z}) = V(\|\xi\|) + \frac{1}{a_1}\gamma\hat{\phi}_1(\hat{\tau})V(\|e_x\|) + \frac{1}{a_1}\hat{\phi}_2(\hat{\tau})V(\|\hat{e}_u\|), \tag{4.58}$$

where

$$\frac{d\hat{\phi}_1(\hat{\tau})}{d\hat{\tau}} = -\hat{m}_1\hat{\phi}_1^2(\hat{\tau}) - \hat{m}_2\hat{\phi}_1(\hat{\tau}) - \hat{m}_3, \tag{4.59a}$$

$$\frac{d\hat{\phi}_2(\hat{\tau})}{d\hat{\tau}} = -\hat{n}_1\hat{\phi}_2^2(\hat{\tau}) - \hat{n}_2\hat{\phi}_2(\hat{\tau}) - \hat{n}_3, \tag{4.59b}$$

and $\hat{m}_i, \hat{n}_i, i = \{1, 2, 3\}$ are given in (4.25). Similar to Theorem 4.1, one can guarantee that $R_2(\hat{z})$ is positive definite by properly choosing $\hat{\phi}_1(0), \hat{\phi}_2(0) > 0$. Taking the derivative of $R_2(\hat{z})$ on $t \in [t_l, t_{l+1})$, one has

$$\begin{aligned}
\dot{R}_2(\hat{z}) \leq & -(\hat{L} - 2a_1)R_2(\hat{z}) + \frac{4\gamma^2}{a_1}\|e_y\|^2 + \frac{4}{a_1}\left\|\hat{e}_u^l\right\|^2 - \frac{2}{a_1^2}\gamma\|e_x\|^2 \\
& + \frac{2}{a_1}\gamma\phi_x(\hat{\tau})\|e_x\|\left(3\gamma\|e_y\| + \sqrt{3}\left\|\hat{e}_u^l\right\|\right) - \frac{1}{a_1^2}\|\hat{e}_u\|^2 \\
& + \frac{2\sqrt{3}}{a_1}\rho_1\gamma\phi_u(\hat{\tau})\|\hat{e}_u\|\|e_y\| \\
\leq & -(\hat{L} - 2a_1)R_2(\hat{z}) + \left(\frac{4\gamma^2}{a_1} + 3\gamma^2(3\gamma + \rho_1^2)\right)\|e_y\|^2 \\
& + \left(\frac{4}{a_1} + 3\gamma\right)\|\hat{e}_u^l\|^2.
\end{aligned} \tag{4.60}$$

For $\forall t \in [t_l, t_{l+1})$ and $\forall i$, one has that i) if $t_l \in \{t_{\sigma_i}^i\}$, one has $\|e_{y_i}(t)\| = 0$, else ii) if $t_l \notin \{t_{\sigma_i}^i\}$, one has $\|e_{y_i}(t_l)\| < c_1 e^{-\alpha_1 t_l}$ and

$$\begin{aligned}
\|e_{y_i}(t)\| = & \|\hat{y}_i(t) - y_i(t)\| = \left\|\int_{t_{\sigma_i}^i}^t f\left(\hat{y}_i\left(s\right)\right)ds - \int_{t_l}^t f\left(y_i\left(s\right)\right)ds\right\| \\
\leq & e^{\rho_1\left(t - t_{\sigma_i}^i\right)}\|\hat{y}_i\left(t_{\sigma_i}^i\right) - e^{\rho_1(t-t_l)}y_i\left(t_l\right)\| \\
= & e^{\rho_1(t-t_l)}\|e_{y_i}\left(t_l\right)\|
\end{aligned} \tag{4.61}$$

for $t \in (t_l, t_{l+1})$. Thus, one has $\|e_{y_i}\| < c_1 e^{\rho_1(t-t_l)} e^{-\alpha t_l} < c_1 e^{(\rho_1+\alpha)h} e^{-\alpha t}$ and

$$\|e_y\| < \sqrt{N} c_1 e^{(\rho_1+\alpha)h} e^{-\alpha t}, \quad \forall t \geq 0. \tag{4.62}$$

Besides, for $\forall t \in [t_l, t_{l+1})$ and $\forall i$, one also has that i) if $t_l \in \{T_k^i\}$, one has $\|e_{u_i}(t_l)\| = 0$, else ii) if $t_l \notin \{T_k^i\}$, one has $\|e_{u_i}(t_l)\| < c_2 e^{-\alpha t_l} < c_2 e^{\alpha h} e^{-\alpha t}$. Then, one can get

$$\|e_u^l\| < \sqrt{N} c_2 e^{\alpha h} e^{-\alpha t}, \quad \forall t \geq 0. \tag{4.63}$$

Substituting (4.62) and (4.63) into (4.60) yields

$$\dot{R}_2(\hat{z}) < -(\hat{L} - 2a_1) R_2(\hat{z}) + (\beta_1 + \beta_2) e^{-2\alpha t}, \tag{4.64}$$

where β_1, β_2 are given in the statement of Theorem 4.2.

Furthermore, if one has $\alpha > K\lambda_2(L) - 2\rho_1 - 2a_1$, then based on the comparison theorem and (4.64), one can get that the solution of $R_2(\hat{z})$ satisfies

$$\begin{aligned} R_2(\hat{z}) \leq & e^{-(\hat{L}-2a_1)t} R_2(\hat{z}) + \int_0^t e^{-(\hat{L}-2a_1)(t-s)} ((\beta_1+\beta_2) e^{-2\alpha t}) ds \\ \leq & d_1 e^{-(\hat{L}-2a_1)t}, \end{aligned}$$

where $d_1 = \|\xi(0)\|^2 + (\beta_1+\beta_2)/(2\alpha - 2K\lambda_2(L) + 4\rho_1 + 4a_1)$. Then, one can further have $\|\xi(t)\| \leq \sqrt{d_1} e^{-(\hat{L}-2a_1)t/2} = \sqrt{d_1} e^{-(K\lambda_2(L)-2\rho_1-2a_1)t}$. That is, consensus of the nonlinear MAS (4.1) is achieved exponentially with convergence rate $K\lambda_2(L) - 2\rho_1 - 2a_1$ and convergence coefficient $\sqrt{d_1}$. □

Proof of Theorem 4.3

Firstly, we assume that the communication bandwidth is unlimited, that is, the quantizer (4.26) will never be saturated and thus we have $\|Q_{\Delta,M_1^i}(x(t)) - x(t)\| \leq \sqrt{n}\Delta/2$ and $\|Q_{\Delta,M_2}(u(t)) - u(t)\| \leq \sqrt{n}\Delta/2$ for all $x(t), u(t) \in \mathbb{R}^n$ and for all i.

Define $\tilde{\tau} : \mathbb{R}_{\geq 0} \to [0, h]$ as

$$\tilde{\tau}(t) = t - kh, \quad t \in [kh, (k+1)h), \quad k = 0, 1, \ldots, \tag{4.65}$$

where $h \in (0, \tilde{h}^*)$.

Consider the Lyapunov function candidate

$$R_3(\hat{z}) = V(\|\xi\|) + \frac{1}{a_1}\gamma\tilde{\phi}_1(\tilde{\tau})V(\|e_x\|) + \frac{1}{a_1}\tilde{\phi}_2(\tilde{\tau})V(\|\hat{e}_u\|), \tag{4.66}$$

where

$$\frac{d\tilde{\phi}_1(\tilde{\tau})}{d\tilde{\tau}} = -\tilde{m}_1\tilde{\phi}_1^2(\tilde{\tau}) - \tilde{m}_2\tilde{\phi}_1(\tilde{\tau}) - \tilde{m}_3, \tag{4.67a}$$

$$\frac{d\tilde{\phi}_2(\tilde{\tau})}{d\tilde{\tau}} = -\tilde{n}_1\tilde{\phi}_2^2(\tilde{\tau}) - \tilde{n}_2\tilde{\phi}_2(\tilde{\tau}) - \tilde{n}_3. \tag{4.67b}$$

Similar to Theorem 4.2, one can get that the derivative of $R_3(\hat{z})$ on $t \in [t_l, t_{l+1})$ satisfies

$$\begin{aligned}\dot{R}_3(\hat{z}) \leq & -(\hat{L} - 2a_1)R_3(\hat{z}) + \left(\frac{6}{a_1} + 4\gamma\right)(\|\hat{e}_u^l\|^2 + \|\hat{e}_q^u\|^2) \\ & + \left(\frac{9\gamma^2}{a_1} + 4\gamma^2(4\gamma + \rho_1^2)\right)(\|e_y\|^2 + \|e_q^y\|^2).\end{aligned} \tag{4.68}$$

According to (4.29), one has

$$\begin{aligned}&\left\|x_i(t^i_{\sigma_i}) - \hat{x}_i(t^i_{\sigma_i})\right\| \\ =&\left\|x_i(t^i_{\sigma_i}) - \hat{x}_i(t^i_{\sigma_i-1}) - \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds\right. \\ &\left. -g_1(t^i_{\sigma_i})Q_{\Delta,M_1^i}\left(\frac{x_i(t^i_{\sigma_i}) - \hat{x}_i(t^i_{\sigma_i-1}) - \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds}{g_1(t^i_{\sigma_i})}\right)\right\| \\ \leq&\left\|x_i(t^i_{\sigma_i}) - \hat{x}_i(t^i_{\sigma_i-1}) - \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds\right. \\ &\left. -g_1(t^i_{\sigma_i})\frac{x_i(t^i_{\sigma_i}) - \hat{x}_i(t^i_{\sigma_i-1}) - \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds}{g_1(t^i_{\sigma_i})}\right\| + \frac{\sqrt{n}\Delta g_1(t^i_{\sigma_i})}{2} \\ =&\frac{\sqrt{n}\Delta g_1(t^i_{\sigma_i})}{2}.\end{aligned}$$

Combining (4.29) and (4.30), one further has

$$e_{q_i}^x(t_{\sigma_i}^i) = \left\|\varsigma_i^i(t_{\sigma_i}^i) - x_i(t_{\sigma_i}^i)\right\| = \left\|\hat{x}_i(t_{\sigma_i}^i) - x_i(t_{\sigma_i}^i)\right\| \leq \frac{\sqrt{n}\Delta g_1(t_{\sigma_i}^i)}{2}.$$

Besides, one has $t - t_{\sigma_i}^i \leq Rh, \forall t \in [t_{\sigma_i}^i, t_{\sigma_i+1}^i)$, then according to (4.28), one can get

$$\left\|e_{q_i}^y(t)\right\| \leq e^{\rho_1 Rh}\left\|e_{q_i}^y(t_{\sigma_i}^i)\right\| = e^{\rho_1 Rh}\left\|e_{q_i}^x(t_{\sigma_i}^i)\right\| \leq \frac{e^{\rho_1 Rh}\sqrt{n}\Delta g_1(t_{\sigma_i}^i)}{2}.$$

From the definition of g_1, one has $g_1(t_{\sigma_i}^i) = c_3 e^{-\eta t_{\sigma_i}^i} = c_3 e^{\eta(t-t_{\sigma_i}^i)}e^{-\eta t} \leq c_3 e^{\eta Rh}e^{-\eta t}$. Then, one further has $\|e_{q_i}^y(t)\| \leq c_3\sqrt{n}\Delta e^{(\eta+\rho_1)Rh}e^{-\eta t}/2$ and thus $\|e_q^y(t)\| \leq \sqrt{N}c_3\sqrt{n}\Delta e^{(\eta+\rho_1)Rh}e^{-\eta t}/2, \forall t$. Similarly, one has

$$\|e_{q_i}^u(t)\| \leq \frac{\sqrt{n}\Delta g_2(T_k^i)}{2} \leq \frac{c_4\sqrt{n}\Delta e^{\eta Rh}e^{-\eta t}}{2},$$

and thus $\|e_q^u(t)\| \leq \sqrt{Nn}c_4\Delta e^{\eta Rh}e^{-\eta t}/2, \forall t$. Then, (4.68) can be rewritten as

$$\dot{R}_3(\hat{z}) < -(\hat{L} - 3a_1)R_2(\hat{z}) + (\beta_3 + \beta_4)e^{-2\alpha t} + k_1 e^{-2\eta t}, \tag{4.69}$$

where β_3, β_4, k_1 are given in the statement of Theorem 4.3. Moreover, one has $\eta < \min\{K\lambda_2(L) - 2\rho_1 - 4a_1/2, \alpha\}$, thus $R_3(\hat{z}) < d_2 e^{-2\eta t}$ and $\|\xi(t)\| < \sqrt{d_2}e^{-\eta t}$ for all $t \geq 0$. Therefore, one can conclude that consensus of the nonlinear MAS (4.27) is achieved exponentially with convergence rate η and convergence coefficient $\sqrt{d_2}$.

In the above, we assume that the communication bandwidth is unlimited so that the quantizer (4.26) will never be saturated. In the following, we will show that the state quantizer and the control quantizer will never be saturated in the case of finite quantization levels M_1^i and M_2, respectively, which implies that the above results hold for the finite-level quantizer case as well. For agent i, define

$$\chi_i^x(\sigma_i) = \frac{x_i(t_{\sigma_i}^i) - \hat{x}_i(t_{\sigma_i-1}^i) - \int_{t_{\sigma_i-1}^i}^{t_{\sigma_i}^i} f\left(\hat{x}_i(s)\right)ds}{g_1(t_{\sigma_i}^i)}, \sigma_i = 0, 1, \ldots$$

and

$$\chi_i^u(k) = \frac{\hat{u}_i(T_k^i)}{g_2(T_k^i)}, k = 0, 1, \ldots.$$

Firstly, the state quantizer is analyzed. if $\sigma_i = 0$, one has

$$\|\chi_i^x(0)\| = \left\|\frac{x_i(0)}{g_1(0)}\right\| \leq \left\|\frac{x_i(0)}{c_3}\right\| \leq \Delta M_1^i,$$

where M_1^i is given in (4.41).

if $\sigma_i \geq 1$, define $\varsigma_i(t^i_{\sigma_i}) = \hat{x}_i(t^i_{\sigma_i-1}) + \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f(\hat{x}_i(s))ds$, then one has

$$\|\chi_i^x(\sigma_i)\| = \left\|\frac{x_i(t^i_{\sigma_i}) - \varsigma_i(t^i_{\sigma_i})}{g_1(t^i_{\sigma_i})}\right\| = \left\|\frac{x_i(t^i_{\sigma_i}) - \hat{y}_i(t^{i-}_{\sigma_i}) + \hat{y}_i(t^{i-}_{\sigma_i}) - \varsigma_i(t^i_{\sigma_i})}{g_1(t^i_{\sigma_i})}\right\|,$$

where $\hat{y}_i(t^{i-}_{\sigma_i})$ is the value of $\hat{y}_i$ before jumping, and it is used to distinguish with $\hat{y}_i(t^i_{\sigma_i})$. According to the definition of $\hat{y}_i(t)$, one has

$$\begin{aligned} x_i(t^i_{\sigma_i}) - \hat{y}_i(t^{i-}_{\sigma_i}) =& x_i(t^i_{\sigma_i} - h) + \int_{t^i_{\sigma_i}-h}^{t^i_{\sigma_i}} f(x_i(s)) + \tilde{u}_i(s)ds \\ & \hat{y}_i(t^i_{\sigma_i} - h) - \int_{t^i_{\sigma_i}-h}^{t^i_{\sigma_i}} f(\hat{y}_i(s))\, ds. \end{aligned} \tag{4.70}$$

According to the definition of $e_{y_i}(t_l)$, one has $\|x_i(t^i_{\sigma_i} - h) - \hat{y}_i(t^i_{\sigma_i} - h)\| = \|e_{y_i}(t^i_{\sigma_i} - h)\|$. if $t^i_{\sigma_i} - h \in \{t^i_{\sigma_i}\}$, $\|e_{y_i}(t^i_{\sigma_i} - h)\| = 0$, otherwise, $\|e_{y_i}(t^i_{\sigma_i} - h)\| \leq c_1 e^{-\alpha(t^i_{\sigma_i} - h)}$. Thus,

$$\begin{aligned} & \left\| x_i(t^i_{\sigma_i} - h) - \hat{y}_i(t^i_{\sigma_i} - h) + \int_{t^i_{\sigma_i}-h}^{t^i_{\sigma_i}} f(x_i(s)) - f(\hat{y}_i(s))\, ds \right\| \\ \leq & e^{\rho_1 h} \left\| x_i(t^i_{\sigma_i} - h) - \hat{y}_i(t^i_{\sigma_i} - h) \right\| \\ \leq & c_1 e^{(\rho_1+\alpha)h} e^{-\alpha t^i_{\sigma_i}}. \end{aligned}$$

Moreover, $\tilde{u}(t) = \tilde{u}(t^i_{\sigma_i} - h)$ is a constant vector for $t \in [t^i_{\sigma_i} - h, t^i_{\sigma_i})$. Therefore,

$$\begin{aligned} \left\| x_i(t^i_{\sigma_i}) - \hat{y}_i(t^{i-}_{\sigma_i}) \right\| \leq & c_1 e^{(\rho_1+\alpha)h} e^{-\alpha t^i_{\sigma_i}} + h \left\| \tilde{u}_i(t^i_{\sigma_i} - h) \right\| \\ \leq & c_1 e^{(\rho_1+\alpha)h} e^{-\alpha t^i_{\sigma_i}} + h(\left\| \hat{u}_i(t^i_{\sigma_i} - h) \right\| + \left\| e^u_{q_i}(t^i_{\sigma_i} - h) \right\|). \end{aligned} \tag{4.71}$$

a) if $t^i_{\sigma_i} - h \in \{T^i_k\}$, then one has

$$\begin{aligned}\hat{u}_i(t^i_{\sigma_i} - h) &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(\tilde{y}_j(t^i_{\sigma_i} - h) - \tilde{y}_i(t^i_{\sigma_i} - h)) \\ &= \sum_{j\in\mathcal{N}_i} \psi_{ij}(x_j(t^i_{\sigma_i} - h) - x_i(t^i_{\sigma_i} - h) + e_{y_j}(t^i_{\sigma_i} - h) \\ &\quad - e_{y_i}(t^i_{\sigma_i} - h) + e^y_{q_j}(t^i_{\sigma_i} - h) - e^y_{q_i}(t^i_{\sigma_i} - h)).\end{aligned}$$

Let $\hat{u}(t)$ be the column stack vector of $\hat{u}_i(t)$. Then one has

$$\begin{aligned}\left\|\hat{u}_i(t^i_{\sigma_i} - h)\right\| \leq& \gamma\left(\left\|\xi(t^i_{\sigma_i} - h)\right\| + \left\|e_y(t^i_{\sigma_i} - h)\right\| + \left\|e^y_q(t^i_{\sigma_i} - h)\right\|\right) \\ \leq& \gamma(\sqrt{d_2} + \sqrt{N}c_1 + \frac{\sqrt{Nn}c_3\Delta e^{(\eta+\rho_1)Rh}}{2})e^{-\eta(t^i_{\sigma_i}-h)} \\ =& Ce^{-\eta(t^i_{\sigma_i}-h)},\end{aligned} \tag{4.72}$$

where C is given in the statement of Theorem 4.3.

Substituting (4.72) into (4.71), one can get

$$\|x_i(t^i_{\sigma_i}) - \hat{y}_i(t^{i-}_{\sigma_i})\| \leq c_1 e^{(\rho_1+\alpha)h}e^{-\alpha t^i_{\sigma_i}} + h\left(C + \frac{c_4\sqrt{n}\Delta}{2}\right)e^{-\eta(t^i_{\sigma_i}-h)}. \tag{4.73}$$

b) if $t^i_{\sigma_i} - h \notin \{T^i_k\}$, then one has $\left\|\hat{u}_i(t^i_{\sigma_i} - h)\right\| \leq \left\|\hat{u}_i(T^i_{k'})\right\| + c_2 e^{-\alpha(t^i_{\sigma_i}-h)}$, where $k' = \arg\min_{l\in\mathbb{N}:T^i_l<t^i_{\sigma_i}-h}\{t^i_{\sigma_i} - h - T^i_l\}$. According to a), one can get

$$\left\|\hat{u}_i(T^i_{k'})\right\| \leq Ce^{-\eta T^i_{k'}} \leq Ce^{\eta Rh}e^{-\eta(t^i_{\sigma_i}-h)}.$$

Thus,

$$\left\|\hat{u}_i(t^i_{\sigma_i} - h)\right\| \leq Ce^{\eta Rh}e^{-\eta(t^i_{\sigma_i}-h)} + c_2 e^{-\alpha(t^i_{\sigma_i}-h)}. \tag{4.74}$$

Substituting (4.74) into (4.71), one can get

$$\begin{aligned}\|x_i(t^i_{\sigma_i}) - \hat{y}_i(t^{i-}_{\sigma_i})\| \leq& c_1 e^{(\rho_1+\alpha)h}e^{-\alpha t^i_{\sigma_i}} \\ &+ h\left(Ce^{\eta Rh} + c_2 + \frac{c_4\sqrt{n}\Delta}{2}\right)e^{-\eta(t^i_{\sigma_i}-h)}.\end{aligned} \tag{4.75}$$

In addition,

$$\begin{aligned}
&\|\hat{y}_i(t^{i-}_{\sigma_i}) - \varsigma_i(t^i_{\sigma_i})\| \\
=&\left\|\hat{y}_i(t^i_{\sigma_i-1}) + \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(x_i(s)\right)ds - \hat{x}_i(t^i_{\sigma_i-1}) - \int_{t^i_{\sigma_i-1}}^{t^i_{\sigma_i}} f\left(\hat{x}_i(s)\right)ds\right\| \\
\leq& e^{\rho_1(t^i_{\sigma_i}-t^i_{\sigma_i-1})}\|x_i(t^i_{\sigma_i-1}) - \hat{x}_i(t^i_{\sigma_i-1})\|
\end{aligned}$$

and $\|x_i(t^i_{\sigma_i-1}) - \hat{x}_i(t^i_{\sigma_i-1})\| \leq g_1(t^i_{\sigma_i-1})\sqrt{n}\Delta/2$. Thus,

$$\|y_i(t^{i-}_{\sigma_i}) - \varsigma_i(t^i_{\sigma_i})\| \leq e^{\rho_1 Rh}c_3 e^{-\eta(t^i_{\sigma_i-1})}\sqrt{n}\Delta/2. \tag{4.76}$$

Combining (4.73), (4.75), and (4.76), one has

$$\begin{aligned}
\|\chi^x_i(\sigma_i)\| \leq& \frac{c_1 e^{(\rho_1+\alpha)h}e^{-\alpha t^i_{\sigma_i}} + h\left(Ce^{\eta Rh} + c_2 + c_4\sqrt{n}\Delta/2\right)e^{-\eta(t^i_{\sigma_i}-h)}}{c_3 e^{-\eta t^i_{\sigma_i}}} \\
&+ \frac{e^{\rho_1 Rh}c_3 e^{-\eta(t^i_{\sigma_i-1})}\sqrt{n}\Delta/2}{c_3 e^{-\eta t^i_{\sigma_i}}} \\
\leq& \frac{c_1 e^{(\rho_1+\alpha)h} + h\left(Ce^{\eta Rh} + c_2 + c_4\sqrt{n}\Delta/2\right)e^{\eta h}}{c_3} \\
&+ \frac{\sqrt{n}\Delta}{2}e^{(\rho_1+\eta)Rh} \\
\leq& \Delta M_1^i.
\end{aligned}$$

Next, the control quantizer is analyzed. For all $k \geq 0$, one has

$$\|\chi^u_i(k)\| = \left\|\frac{\hat{u}_i(T^i_k)}{g_2(T^i_k)}\right\| \leq \left\|\frac{\hat{u}(T^i_k)}{g_2(T^i_k)}\right\| \leq \frac{Ce^{-\eta T^i_k}}{g_2(T^i_k)} = \frac{C}{c_4} \leq \Delta M_2,$$

where M_2 is given in (4.42). □

Chapter 5

Leader-Follower Target Tracking under Time Constraint

Chapters 3 and 4 considered multi-agent control under a single cooperative task, *i.e.,* consensus, and the task only needs to be completed asymptotically (there is no deadline). However, for many practical applications (*e.g.,* robotics), a group of agents are required to achieve a sequence of tasks. Moreover, each task may be required to be completed within a deadline. Due to these considerations, in this chapter, we investigate the leader-follower target tracking problem for multi-agent systems (MAS). We assume that the leader-follower MAS is subject to a sequence of tasks, each of which is activated dynamically and has to be completed within a deadline. In addition, the reward and cost of satisfying the sequence of tasks are taken into account. The objective is to jointly schedule the dynamically activated tasks and design distributed control laws such that the pre-defined objective function is maximized.

5.1 Introduction

Over the past decades, the research on multi-agent cooperative control has been usually focused on achieving one single task, such as consensus [9], in an asymptotic manner. In practice, a group of agents (robots) may encounter the request of a sequence of tasks which are activated dynamically. Moreover, deadline constraints on the completion of each task is a reasonable requirement, *e.g.,* "visiting region A within 10 time units". Therefore, jointly scheduling of deadline-constrained dynamically activated task sequences and design-

ing distributed controllers for a group of agents is challenging.

In real-time systems, a task is usually characterized by a specific deadline. For such systems, many dynamic scheduling algorithms rely on Earliest Deadline First (EDF) policies [75], [76] and their extensions [77], [78]. The objective of these algorithms is to execute a set of tasks in order to meet the most possible number of deadlines. Reward-based scheduling [79] represents another class of dynamic scheduling algorithms, in which, the reward of a task is associated with its execution time rather than the deadline. For example, increasing reward with increasing service models [80] fall within the scope of this framework. Different from EDF, reward-based scheduling is capable of modeling tasks that are characterized not only by a deadline, but also by their 'importance'. The performance of the algorithm is then evaluated by computing the cumulative reward gained on a task set.

In this chapter, we address the dynamic scheduling problem under the framework of reward-based scheduling. In the increasing reward with increasing service models, the reward of a task increases with the residual time[1] [80]. In our task model, we consider that a task can be completed at several Quality-of-Satisfaction (QoSa) levels, and each QoSa level corresponds to a time interval within which the task should be completed. However, the reward of a task does not necessarily increase with the residual time. Apart from reward, the cost of satisfying a set of tasks is further considered, which is defined as the estimated total distance travelled by the group of agents. Based on the above setting, a dynamic scheduling strategy is proposed to combine the ideas of EDF and reward-based scheduling.

On the other hand, to guarantee that the desired performance (in terms of reward and cost) can be achieved based on the dynamic scheduling strategy, one has to ensure that each task is completed at the desired QoSa level. This further requires that each task is completed at a specific time interval, *e.g.,* "visiting region A within $[6, 8]$ time units", and brings additional difficulties to the control synthesis. To the best of our knowledge, this type of control design problem under specific time constraints is rarely considered in the context of MAS. In our previous work [81], multi-agent control under specific deadline constraints was studied and a linear feedback controller was designed. Nevertheless, the control design methods are not applicable when specific interval constraints are presented. In [82], [83], multi-agent control under signal temporal logic tasks was investigated and a barrier function based controller

[1]The amount of time between the completion time and the absolute deadline

was proposed for each agent. However, the control policy requires the knowledge of each agent of the task plan and the target regions. In this chapter, we consider a leader-follower MAS, in which, only the leader agents know the information of the target regions. Therefore, it is necessary that the group of agents collaborate for the task completion. In this case, the fully distributed control synthesis (each agent can only communicate with neighboring agents) is challenging, particular under an explicit time interval constraint.

This chapter investigates the jointly design of task scheduling and distributed control law for leader-follower MAS subject to a sequence of dynamically activated tasks. More specifically, each task is associated with a region of interest, where the group of agents need to visit together within a deadline. The main contributions are twofold.

(i) The notion of QoSa levels is introduced for completing each task. By associating the reward of completing a set of tasks to the QoSa levels and the cost to the estimated total travelling distance, a new task scheduling problem is formulated and a dynamic scheduling strategy is proposed.

(ii) Under the condition that the information of the target regions is available only to the leader agents, distributed control laws are designed respectively for the leader and follower agents. It is shown that the designed control laws can guarantee the satisfaction of each task at the desired QoSa level.

We note that the control design strategy is motivated by the funnel control [84] and the prescribed performance control [85] approaches, which have been originally used to solve control problems with transient performance constraints [84], [85] and recently applied to the multi-agent setting [86]–[88]. In this chapter, we propose for the first time a way to transform time interval constraints into transient performance constraints under the leader-follower MAS framework, and hence, funnel control and prescribed performance control can be applied.

5.2 Problem Formulation

5.2.1 Leader-follower MAS

Consider a group of N agents, each of which obeys the following second-order dynamics

$$\begin{aligned} \dot{x}_i(t) &= v_i(t), \\ \dot{v}_i(t) &= u_i(t), \quad i = 1, 2, \ldots, N, \end{aligned} \tag{5.1}$$

where $x_i \in \mathbb{R}^n, v_i \in \mathbb{R}^n$ and $u_i \in U_i \subseteq \mathbb{R}^n$ are the position, velocity and control input of agent i, respectively. Let $x = (x_1, \ldots, x_N), v = (v_1, \ldots, v_N), u = (u_1, \ldots, u_N)$ be the stack vector of positions, velocities, and inputs, respectively. The input of agent i is constrained to a set U_i. Without loss of generality, we suppose that the first $n_l, n_l \leq N$ agents are leaders. Then, one can define the leader set $\mathcal{V}_L := \{1, 2, \ldots, n_l\}$ and the follower set $\mathcal{V}_F = \{n_l + 1, \ldots, N\}$.

The input of each agent is subject to the following constraint $\|u_i\| \leq u_i^{\max}, i \in \mathcal{V}$, where $u_i^{\max} > 0$. Then, the input set U_i is given by

$$U_i = \{u \in \mathbb{R}^n : \|u\| \leq u_i^{\max}\}, i \in \mathcal{V}. \tag{5.2}$$

It is assumed that the graph $\mathcal{G}$ formed by the leader-follower MAS is undirected and connected. Denote by $\bar{x} = (\bar{x}_1, \ldots, \bar{x}_p)$ the p-dimensional stack vector of relative positions of pairs of agents that form an edge in $\mathcal{G}$, where p is the number of edges. The kth element of vector $\bar{x}$, denoted by $\bar{x}_k := x_{ij} = x_i - x_j, k \in \{1, 2, \ldots, p\}$, corresponds to an edge $e_k = (i, j)$ in the graph.

5.2.2 Task specification

For the group of agents, we assume the existence of a set of $M \in \mathbb{N}$ tasks, denoted by $\phi := \{\phi_1, \phi_2, \ldots, \phi_M\}$. To be more specific, each task is associated with a target region that the MAS has to visit within a deadline. To model the dynamic task generation process, we need the notion of a task generation signal. Let the set

$$T_g := \{T_0, T_1, \ldots\} \tag{5.3}$$

be the sequence of *task generation times*, where $T_0 < T_1 < \cdots$. Without loss of generality, we assume $T_0 = 0$. Then, we define the function $\delta : T_g \to 2^\phi$

as the *task generation signal*, which maps each task generation time T_k to a subset of tasks that are activated at T_k.

The task set ϕ is assumed to have the following features:

1. Tasks are preemptable[2] and activated dynamically;
2. Each task can be activated multiple times. However, a task can not be activated again if the previous activated one is not completed.

Each task $\phi_l, l \in \{1, \ldots, M\}$ is defined as a tuple $\phi_l := (\mathbb{X}_l, a_l, d_l, D_l, A_l, k_l, \mathrm{I}_l, R_l)$, where

- $\mathbb{X}_l \subset \mathbb{R}^n$: the target region associated with task ϕ_l;
- a_l: the set of activation times $a_l := \{a_{l,1}, a_{l,2}, \ldots\} \subseteq T_g$, *i.e.,* the subset of task generation times at which the task ϕ_l is activated and becomes ready to execute;
- d_l: the relative deadline, which is a constant value for each task;
- D_l: the absolute deadline $D_l := \{D_{l,1}, D_{l,2}, \ldots\}$. For task ϕ_l that arrives at $a_{l,k} \in a_l$, $D_{l,k}$ is computed as $D_{l,k} = a_{l,k} + d_l$;
- A_l: the (actual) completion times $A_l := \{A_{l,1}, A_{l,2}, \ldots\}$, where $A_{l,k}$ corresponds to $a_{l,k}$. We note that each $A_{l,k}$ is determined online;
- $k_l \in \mathbb{N}$: the number of QoSa levels for ϕ_l, which satisfies $k_l \geq 2$;
- I_l: the interval set $\mathrm{I}_l := \{\mathrm{I}_l[k_l - 1], \ldots, \mathrm{I}_l[1], \mathrm{I}_l[0]\}$, where $\mathrm{I}_l[\hat{k}], \hat{k} \in \{0, 1, \ldots, k_l - 1\}$ is the time interval corresponding to task ϕ_l and QoSa level $\hat{k}$. In addition, I_l satisfies the following properties: i) $\mathrm{I}_l[0] = (d_l, \infty)$; ii) $\cup_{m=1}^{k_l-1} \mathrm{I}_l[m] = [0, d_l]$; iii) $\mathrm{I}_l[m_1] \cap \mathrm{I}_l[m_2] = \emptyset, \forall m_1 \neq m_2$; and iv) $\forall t_1 \in \mathrm{I}_l[m_1], t_2 \in \mathrm{I}_l[m_2]$, $m_1 > m_2$ implies $t_1 < t_2$;
- R_l : the reward set $R_l := \{R_l[k_l - 1], \ldots, R_l[1], R_l[0]\}$, where $R_l[\hat{k}], \hat{k} \in \{0, 1, \ldots, k_l - 1\}$ represents the reward that task ϕ_l contributes if it is completed at QoSa level $\hat{k}$.

[2]Preemptable means that the execution of a uncompleted task can be temporarily halted and later resumed.

For the sake of notation simplicity, in the following, we use a_l, D_l, A_l to represent any $a_{l,k} \in a_l, D_{l,k} \in D_l, A_{l,k} \in A_l$, respectively, when there is no ambiguity. Note that when a_l corresponds to $a_{l,k}$, D_l, A_l correspond to $D_{l,k}, A_{l,k}$, respectively. Then, we have the following definition.

Definition 5.1. *Given the task ϕ_l and the activation time a_l, we say that ϕ_l* is completed at time A_l *if*

$$x_i(A_l) \in \mathbb{X}_l, \forall i \in \mathcal{V}.$$

In addition, we say that ϕ_l is completed at QoSa level $\hat{k} \in \{0, 1, \ldots, k_l - 1\}$ *if the (actual) completion time*

$$A_l \in \{a_l + t | t \in I_l[\hat{k}]\}.$$

Remark 5.1. *Each task ϕ_l has at least two QoSa levels. From the properties* i-iv) *of I_l and Definition 5.1, one can see that a QoSa level of bigger than or equal to 1 means that the task is completed before the deadline, while $QoSa$ level 0 means that the deadline of the task is violated. In addition, a higher QoSa level means that the task is completed faster. However, we note that the reward is not necessarily higher for a higher QoSa level. It is assumed here that for each task ϕ_l, $I_l[\hat{k}], R_l[\hat{k}], \forall \hat{k} \in \{0, \ldots, k_l - 1\}$ are given a priori and are known to each agent.*

In the following, each target set $\mathbb{X}_l$ will be over-approximated by its largest inscribed ball, denote by $\overline{\mathbb{X}}_l$. Let c_l be the Chebyshev center of $\mathbb{X}_l$. Then, $\overline{\mathbb{X}}_l$ can be represented by

$$\overline{\mathbb{X}}_l := \mathcal{B}(c_l, \bar{r}_l) = \{y \in \mathbb{R}^n : \|y - c_l\| \leq \bar{r}_l\}, \tag{5.4}$$

where $\bar{r}_l$ is the radius of $\overline{\mathbb{X}}_l$. It is assumed that all target regions are compact and have non-empty interiors, *i.e.,* there exist $r_{\min}, r_{\max} > 0$ such that $r_{\min} \leq \bar{r}_l \leq r_{\max}, \forall l$. In addition, the target regions are disjoint, *i.e.,* $\mathbb{X}_{k_1} \cap \mathbb{X}_{k_2} = \emptyset, \forall k_1, k_2 \in \{1, \ldots, M\}, k_1 \neq k_2$. Note that the problem of finding the maximum inscribed ball can be solved offline by several iterative algorithms, such as the one proposed in [89].

Remark 5.2. *In this chapter, we consider that the task ϕ_l is completed once all agents lie inside the target region $\mathbb{X}_l$. However, in practice, there might be a (set of) service(s) associated with ϕ_l, for which the group of agents can provide only when being present in the region of interest $\mathbb{X}_l$. Hence, upon the visit to $\mathbb{X}_l$, the*

group of agents may still need to accomplish services, such as loading or releasing a cargo, etc. Note that in this chapter, we do not focus on how these services are being provided at the target region, but rather aim at guaranteeing the necessary preconditions for providing these services, i.e., being present at the target region.

5.2.3 Reward and cost of a given task set

In this chapter, we are interested in the quantitative reward and cost of satisfying the dynamically activated tasks. Let $\eta \in 2^{\phi} \setminus \emptyset$ be any subset of ϕ except $\emptyset$. The index set associated with η is denoted by $\mathbb{I}_{\eta}$. Let Π_{η} be the set containing all permutations of $\mathbb{I}_{\eta}$, and $\pi \in \Pi_{\eta}$ be $\pi := (\pi[1], \ldots, \pi[|\mathbb{I}_{\eta}|])$.

Example 5.1. *Let* $\eta = \{\phi_1, \phi_5, \phi_7\}$*, then* $\mathbb{I}_{\eta} = \{1, 5, 7\}$ *and* $\Pi_{\eta} = \{(1,5,7), (1,7,5), (5,1,7), (5,7,1), (7,1,5), (7,5,1)\}$*. if* $\pi = (5,7,1) \in \Pi_{\eta}$*, then* $\pi[1] = 5, \pi[2] = 7, \pi[3] = 1$*.*

Denote by $\mathbf{P}_{\eta}(\pi[0]\pi) = \mathbf{P}_{\eta}(\pi[0]\pi[1] \ldots \pi[|\mathbb{I}_{\eta}|]) = \mathbb{X}_{\pi[0]} \ldots \mathbb{X}_{\pi[|\mathbb{I}_{\eta}|]}$ a *path* generated by the task set η with the order of completion given by π, where $\pi[0] := 0$, $\mathbb{X}_{\pi[0]}$ is a collection of the agents' initial states, which thus contains finite number of elements and will be formally defined later.

1) *Reward*: The reward of the path $\mathbf{P}_{\eta}(\pi[0]\pi)$ is given by

$$R(\mathbf{P}_{\eta}(\pi[0]\pi)) = \sum_{l=1}^{|\mathbb{I}_{\eta}|} R_{\pi[l]}[\hat{k}_{\pi[l]}], \tag{5.5}$$

where

$$\hat{k}_{\pi[l]} \in \{0, \ldots, k_{\pi[l]} - 1\}, l = 1, \ldots, |\mathbb{I}_{\eta}|,$$

represents the QoSa level of task $\phi_{\pi[l]}$. Note that the QoSa levels are defined in terms of the (actual) completion time and each task can be completed at different QoSa levels (and thus return different rewards). Therefore, $R(\mathbf{P}_{\eta}(\pi[0]\pi))$ is dependent on the time needed to complete each task.

2) *Cost*: Motivated by [90], the cost of the path $\mathbf{P}_{\eta}(\pi[0]\pi)$ is defined as the (estimated) distance travelled by the group of agents. Let $W(\mathbb{X}_k, \mathbb{X}_j) \in \mathbb{R}_{\geq 0}$ be the transition cost from set $\mathbb{X}_k$ to $\mathbb{X}_j$, which is given by

$$W(\mathbb{X}_k, \mathbb{X}_j) := \begin{cases} \sum_{x \in \mathbb{X}_k} \|x - c_j\|, & \text{if } k = 0, j \in \{1, \ldots, M\} \\ N(\|c_k - c_j\|), & \text{if } k, j \in \{1, \ldots, M\}, k \neq j. \end{cases}$$

Then, the cost of $\mathbf{P}_\eta(\pi[0]\pi)$ is defined as:

$$C(\mathbf{P}_\eta(\pi[0]\pi)) = \sum_{k=0}^{|\mathbb{I}_\eta|-1} W(\mathbb{X}_{\pi[k]}, \mathbb{X}_{\pi[k+1]}). \tag{5.6}$$

It is obvious that the completion of the tasks in η can be scheduled in different orders, which corresponds to different paths. Let $\mathbb{P}_\eta = \cup_{\pi\in\Pi_\eta, \hat{k}_{\pi[l]}\in\{0,\ldots,k_{\pi[l]}-1\}, l\in 1,\ldots,|\mathbb{I}_\eta|}\{\mathbf{P}_\eta(\pi[0]\pi)\}$ be the set of all paths. In this chapter, we want to maximize the reward as well as minimizing the cost. Therefore, we propose the following objective function

$$\mathbf{J}(\mathbb{P}_\eta) \triangleq \max_{\substack{\pi\in\Pi_\eta,\\ \hat{k}_{\pi[l]}\in\{0,\ldots,k_{\pi[l]}-1\},\\ l\in\{1,\ldots,|\mathbb{I}_\eta|\}}} \{\varsigma R(\mathbf{P}_\eta(\pi[0]\pi)) - (1-\varsigma)C(\mathbf{P}_\eta(\pi[0]\pi))\}, \tag{5.7}$$

where $\varsigma \in [0,1]$ is a parameter used to balance between the reward and the cost.

Example 5.2. *Following Example 5.1, we assume that the number of QoSa levels $k_1 = 3, k_5 = 4, k_7 = 2$. Then $\hat{k}_1 \in \{0,1,2\}, \hat{k}_5 \in \{0,1,2,3\}$ and $\hat{k}_7 \in \{0,1\}$. Given one schedule $\pi = (5,7,1)$, we have $R(\boldsymbol{P}_\eta(\pi[0]\pi)) = R_5[\hat{k}_5] + R_7[\hat{k}_7] + R_1[\hat{k}_1]$ and $C(\boldsymbol{P}_\eta(\pi[0]\pi)) = \sum_{x\in\mathbb{X}_0}\|x - c_5\| + N\|c_5 - c_7\| + N\|c_7 - c_1\|$, where N is the total number of agents, $\mathbb{X}_0 := \{x_1(0),\ldots,x_N(0)\}$. Note that the reward $R(\boldsymbol{P}_\eta(\pi[0]\pi))$ is further determined by the chosen QoSa level for each task.*

5.2.4 Problem

In this chapter, we consider that only the leader agents know the information of the target regions. Moreover, it is further considered that each agent does not know the identity (leader or follower) of its neighbors. Therefore, it is necessary for the group of agents (both leader and follower agents) to coordinate to complete the tasks.

Let $\eta(t) \in 2^\phi$ be a set of tasks that are active (being activated) at time t. The purpose of this chapter is to maximize, in an online fashion, the objective function (5.7) for the set of active tasks $\eta(t)$. Formally, the problem is stated below.

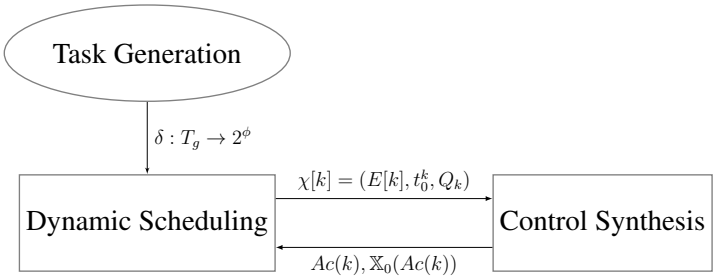

Figure 5.1: The relation between dynamic scheduling and control synthesis.

Problem 5.1. *Given the leader-follower MAS (5.1) and the task specifications in Section 5.2.2, jointly schedule the dynamically activated tasks and design distributed (with only measurements of states of neighbors) control law u_i for each leader or follower agent i, such that $\boldsymbol{J}(\mathbb{P}_{\eta(t)})$ is maximized.*

5.3 Solution

The proposed solution consists of two layers: i) an online dynamic scheduling layer and ii) a (distributed) control synthesis layer which guarantees that the group of agents (both leader and follower) arrive at their progressive target regions at the corresponding QoSa levels at any time.

The relation between the two layers is depicted in Figure 5.1. The dynamic scheduling layer determines when and which task should be executed at which QoSa level. Let χ be the execution task sequence generated by the scheduling layer, which records the sequence of tasks being executed. The kth element of χ, denoted by $\chi[k]$, contains the information of the kth execution task. It is represented by a triple $\chi[k] = (E[k], t_0^k, Q_k)$, where $E[k] \in \phi$ represents the kth execution task, $t_0^k \in \mathbb{R}_{\geq 0}$ is the time that $E[k]$ starts execution, and Q_k is the desired QoSa level of $E[k]$, respectively. $\chi[k]$ will be used for the control synthesis. The (actual) completion time of the task $E[k]$, denoted by $Ac(k)$, is determined online by the synthesized controller. Define $\mathbb{X}_0(Ac(k)) := \{x_1(Ac(k)), \ldots, x_N(Ac(k))\}$ as the set which contains the position information of the MAS at time $Ac(k)$. As shown in Figure 5.1, once $E[k]$ is completed, the information $Ac(k), \mathbb{X}_0(Ac(k))$ will be sent to the dynamic scheduling layer for generating the next execution task.

The task plan is discrete while the state evolution of the MAS is contin-

uous. Therefore, in the following, a hybrid system structure is proposed to model the whole process.

5.3.1 Hybrid structure

As stated previously, the information of the target regions is known to only the leader agents. Therefore, different control laws have to be synthesized for the leader and follower agents, respectively. In addition, when there is no task left to be executed, *i.e.,* $\chi[k] = \emptyset$, all agents will switch to idle mode. To cope with these variations, we propose three different control modes for each agent i:

i) the leader mode: $u_i = u_i^{\text{lead}}$;
ii) the follower mode: $u_i = u_i^{\text{fol}}$;
iii) the idle mode: $u_i = u_i^{\text{idle}}$,

where $u_i^{\text{lead}}, u_i^{\text{fol}}$ and u_i^{idle} will be defined later.

The execution task sequence χ is updated either at the task generation time $t = T_l, T_l \in T_g$ or at the (actual) completion time $Ac(k)$ of the current execution task $\chi[k]$ (i.e., $x_i(Ac(k)) \in \mathbb{X}_{\mathbb{I}(E[k])}, \forall i$). Here, $\mathbb{I}(E[k])$ returns the index of task $E[k]$. If at $Ac(k)$, there is no task to be executed next (*i.e.,* $E[k+1] = \emptyset$), all agents will switch to idle mode. We note that $\chi[k] = \emptyset$ if $E[k] = \emptyset$. Each $\chi[k]$ determines uniquely a control law u_i for each agent i, which is denoted by $u_i^{\chi[k]}$.

To model the discrete behavior of the scheduling and the control law switching, in the following, the hybrid system framework proposed in [91], [92] is applied. An integer variable $k \in \mathbb{N}$ is introduced, which is initialized to be 0. It remains constant during flows and it is incremented by 1 once a new execution task is generated. In other words,

$$F : \begin{cases} \dot{x} = v \\ \dot{v} = u^{\chi[k]} \\ \dot{t} = 1 \\ \dot{k} = 0 \end{cases} \qquad G : \begin{cases} x^+ = x \\ v^+ = v \\ t^+ = t \\ k^+ = k + 1 \end{cases} \tag{5.8}$$

where $u^{\chi[k]} = (u_1^{\chi[k]}, \ldots, u_N^{\chi[k]})$, and

$$u_i^{\chi[k]} = \begin{cases} u_i^{\text{idle}}, \text{ if } \chi[k] = \emptyset, \\ u_i^{\text{lead}}, \text{ if } \chi[k] \neq \emptyset \text{ and } i \in \mathcal{V}_L, \\ u_i^{\text{fol}}, \text{ if } \chi[k] \neq \emptyset \text{ and } i \in \mathcal{V}_F. \end{cases} \tag{5.9}$$

The jump set $\mathcal{D}$ is given by two kinds of events, that is

$$\mathcal{D} := \{ \underbrace{\exists T_l \in T_g \text{ s.t. } t = T_l}_{\text{new tasks activated}} \vee \underbrace{x_i(t) \in \mathbb{X}_{\mathbb{I}(E[k])}, \forall i}_{\text{current task completed}} \}, \tag{5.10}$$

and the flow set $\mathcal{C}$ is given by

$$\mathcal{C} := \mathbb{R}_{\geq 0} \times \mathbb{R}^{nN} \setminus \mathcal{D}. \tag{5.11}$$

At jumps, the control input for each agent will be updated according to the next execution task. The design strategy will be given in the later subsection.

Remark 5.3. *The hybrid system (5.8) is introduced to model the discrete task evolution and the continuous state evolution of the MAS (5.1). It will become clear later in the control synthesis section that the control design for each agent and the analysis are conducted on a single task $\chi[k]$. We note that there is no jump during the execution of $\chi[k]$. Therefore, the hybrid time domain (t, j) and the solutions of hybrid systems on (t, j) are not introduced.*

5.3.2 Dynamic scheduling algorithm

The dynamic scheduling process is shown in Figure 5.2, which consists of two algorithms: *Arrival* and *Completion*.

Let W and R be the ordered wait and (temporarily) reject task set, respectively. Define $W[l] \in \phi$ and $R[l] \in \phi$ as the lth element of W and R, respectively. The index set associated with W is denoted by $\mathbb{I}_W$ and the set of permutations of $\mathbb{I}_W$ is denoted by Π_W. In this chapter, it is assumed that the cardinality (length) of the wait task set W is limited (for computation efficiency), given by $\bar{m}$. In particular, there are at most $\bar{m}$ tasks maintained within it. However, we note that there are extra positions in W, which can be occupied instantaneously.

At the task generation time instant T_l, the set of tasks $\delta(T_l)$ is activated. Then, the algorithm *Arrival, i.e.,* Algorithm 5.1, will be activated to determine (among all active tasks) which task should be executed. Due to the fact that the cardinality of the wait task set W is limited, one needs to check whether there are enough spaces in W before scheduling. Otherwise, we first schedule all the active tasks (including the current executing task $E[k]$, the set of newly activated tasks $\delta(T_l)$ and all the tasks in W) according to the EDF policy. Then, the first $\bar{m}$ tasks are kept in W while the remaining tasks are put into the

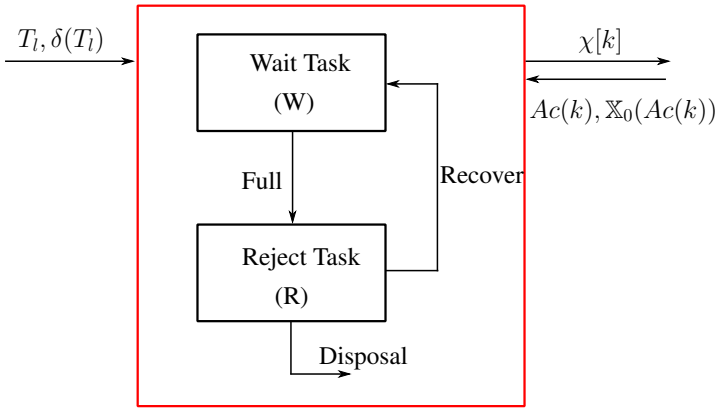

Figure 5.2: Dynamic scheduling process.

(temporarily) reject list R (lines 1-5). After this process, we (re)schedule the tasks in W using the optimization program $\mathcal{P}(W, \mathbb{X}_0(T_l), T_l)$. The first task in line, *i.e.,* $\pi^*[1]$, is chosen as the next executing task, and the corresponding desired QoSa level, is given by $\hat{k}_{\pi^*[1]}$ (lines 6-8).

Algorithm 5.1 *Arrival*

Input: $E[k], W, R, T_l, \delta(T_l)$ and $\mathbb{X}_0(T_l) \leftarrow \{x_1(T_l), \ldots, x_N(T_l)\}$.
Output: $E[k+1], t_0^{k+1}, Q_{k+1}$.

1: $W \leftarrow E[k] \cup W \cup \delta(T_l)$,
2: **if** $|W| \geq \bar{m} + 1$ **then**,
3: Schedule W according to the EDF policy,
4: $W \leftarrow W \backslash \{W[\bar{m}+1], \ldots W[|W|]\}, T \leftarrow T \cup \{W[\bar{m}+1], \ldots, W[|W|]\}$,
5: **end if**
6: Solve the optimization program $\mathcal{P}(W, \mathbb{X}_0(T_l), T_l)$, which returns $\pi^*, \hat{k}_{\pi^*}$ (see (5.12)),
7: $E[k+1] \leftarrow \pi^*[1], t_0^{k+1} \leftarrow T_l, Q_{k+1} \leftarrow \hat{k}_{\pi^*[1]}$,
8: $W[i] \leftarrow \pi^*[i+1], i = 1, \ldots, |W| - 1$.

In line 6, the optimization program $\mathcal{P}(W, \mathbb{X}_0, t)$ is given by:

$$\max_{\pi \in \Pi_W,\, \hat{k}_\pi} \varsigma \sum_{l=1}^{|\mathbb{I}_W|} R_{\pi[l]} - (1-\varsigma) \sum_{l'=0}^{|\mathbb{I}_W|-1} W(\mathbb{X}_{\pi[l']}, \mathbb{X}_{\pi[l'+1]}) \tag{5.12a}$$

subject to

$$\hat{k}_{\pi[l]} \in \{0, 1, \ldots, k_{\pi[l]} - 1\}, \tag{5.12b}$$

$$R_{\pi[l]} := R_{\pi(l)}[\hat{k}_{\pi[l]}], \tag{5.12c}$$

$$E_{\pi[l]}[0] = a_{\pi[l]} + d_{\pi[l]} + \epsilon, \epsilon > 0, \tag{5.12d}$$

$$E_{\pi[l]}[\hat{k}_{\pi[l]}] = a_{\pi[l]} + \sup\{\mathrm{I}_{\pi[l]}[\hat{k}_{\pi[l]}]\}, \quad \hat{k}_{\pi[l]} \geq 1, \tag{5.12e}$$

$$E_{\pi[1]}[\hat{k}_{\pi[1]}] > t + t_{\min}(\mathbb{X}_0, \mathbb{X}_{\pi[1]}), \tag{5.12f}$$

$$E_{\pi[l+1]}[\hat{k}_{\pi[l+1]}] > E_{\pi[l]}[\hat{k}_{\pi[l]}] + t_{\min}(\mathbb{X}_{\pi[l]}, \mathbb{X}_{\pi[l+1]}), \quad l = 1, \ldots, |\mathbb{I}_W| - 1, \tag{5.12g}$$

where $\hat{k}_\pi := \{\hat{k}_{\pi[1]}, \hat{k}_{\pi[2]}, \ldots, \hat{k}_{\pi[|\mathbb{I}_W|]}\}$. In (5.12f) and (5.12g),

$$t_{\min}(\mathbb{X}_{\pi[l]}, \mathbb{X}_{\pi[l+1]}) := \max_{i \in \mathcal{V}} \min_{u_i \in U_i} T_i^f \tag{5.13a}$$

subject to

$$x_i(0) = \mathbb{X}_{\pi[l]}[i], v_i(0) = 0, \quad l = 0, \tag{5.13b}$$

$$x_i(0) = c_{\pi[l]}, v_i(0) = 0, \quad l \geq 1, \tag{5.13c}$$

$$\dot{x}_i = v_i, \dot{v}_i = u_i, \tag{5.13d}$$

$$x_i(T_i^f) \in \mathbb{X}_{\pi[l+1]}, \tag{5.13e}$$

representing the (estimated) minimal time required to drive the MAS from set $\mathbb{X}_{\pi[l]}$ to set $\mathbb{X}_{\pi[l+1]}$ under the input constraints. Note that in (5.13b), $\pi[0] = 0$ and $\mathbb{X}_{\pi[0]}[i] = \mathbb{X}_0[i]$ is the ith element of set $\mathbb{X}_0$, where $\mathbb{X}_0$ contains the initial state of the group of agents. The optimal solution of $\mathcal{P}(W, \mathbb{X}_0, t)$ is given by $\pi^* = \{\pi^*[1], \ldots, \pi^*[|\mathbb{I}_W|]\}$ and $\hat{k}_{\pi^*} = \{\hat{k}_{\pi^*[1]}, \ldots, \hat{k}_{\pi^*[|\mathbb{I}_W|]}\}$. In (5.12d)-(5.12g), the notation $E_{\pi[l]}[\hat{k}_{\pi[l]}]$ represents the estimated completion time of task $\phi_{\pi[l]}$ at QoSa level $\hat{k}_{\pi[l]}$. It is defined as $a_{\pi[l]} + d_{\pi[l]} + \epsilon$ for QoSa level 0 and $a_{\pi[l]} + \sup\{\mathrm{I}_l[\hat{k}_{\pi[l]}]\}$ for QoSa level $\hat{k}_{\pi[l]} \geq 1$, as seen in (5.12d) and (5.12e) respectively. The constant ϵ is used to distinguish between $E_{\pi[l]}[1]$ and $E_{\pi[l]}[0]$. Conditions (5.12f) and (5.12g) mean that the estimated completed time of the $l+1$th task in line must be bigger than the estimated completed time of the lth task plus the minimal time required to move from $\mathbb{X}_{\pi[l]}$ to $\mathbb{X}_{\pi[l+1]}$.

When the kth execution task is completed at time $Ac(k)$, algorithm *Completion* (Algorithm 5.2), is activated to determine the next execution task. We note that if the activating time and the completion time coincide, algorithm *Arrival* will run first and then algorithm *Completion* is executed. As one can

see in (5.12), the optimization program $\mathcal{P}(W, \mathbb{X}_0, t)$ admits tasks with QoSa level 0 as feasible solutions. Therefore, before rescheduling the wait list, one needs to further inspect to ensure that tasks in the wait list W and the temporarily reject list T are feasible at the current time $Ac(k)$. All infeasible tasks, *i.e.,* $\{\phi_l : \phi_l \in W \cup R, D_l \leq Ac(k)\}$ will be removed forever (line 1). After the inspection, if there are empty positions in W, then tasks in R will be recovered (lines 2-5). At this stage, if $W = \emptyset$, it means that there is no feasible task to be executed next, and then all agents will switch to idle mode (lines 6-9). The idle mode of an agent will be defined later. Otherwise, the wait list W will be rescheduled according to (5.12), and the first task in W will be chosen as the next execution task and removed from W (lines 10-13).

Algorithm 5.2 *Completion*

Input: W, R, $Ac(k)$ and $\mathbb{X}_0(Ac(k)) \leftarrow \{x_1(Ac(k)), \ldots, x_N(Ac(k))\}$.
Output: $E[k+1], t_0^{k+1}, Q_{k+1}$.
1: Remove all infeasible tasks in W and R,
2: **if** $R \neq \emptyset$, **then**
3: Schedule R according to the EDF policy,
4: Move tasks from R to W, until either W is full or R is empty,
5: **end if**
6: **if** $W = \emptyset$, **then**
7: $E[k+1] \leftarrow \emptyset, t_0^{k+1} \leftarrow \emptyset, Q_{k+1} \leftarrow \emptyset$,
8: All agents switch to idle mode,
9: **else**
10: Solve program $\mathcal{P}(W, \mathbb{X}_0(Ac(k)), Ac(k))$, which returns $\pi^*, \hat{k}_{\pi^*}$,
11: $E[k+1] \leftarrow \pi^*[1], t_0^{k+1} \leftarrow Ac(k), Q_{k+1} \leftarrow \hat{k}_{\pi^*[1]}$,
12: $W[l] \leftarrow \pi^*[l+1], l = 1, \ldots, |W| - 1$.
13: **end if**

Remark 5.4. *In the optimization program (5.12), tasks with QoSa level 0 are considered as feasible and kept in W or R for robustness considerations. The reason is that in real-time execution, a task may be completed before its estimated completion time, and in this case, the previous infeasible tasks (task with QoSa level 0) may become feasible again.*

Remark 5.5. *The optimization program $\mathcal{P}(W, \mathbb{X}_0, t)$ can be viewed as a two-layer optimization problem. The first (outer) layer is a combinatorial optimization problem, where the set of permutations of $\mathbb{I}_W$, i.e., Π_W needs to*

be computed. The second (inner) layer is a mixed integer linear program. This is because for each $\pi \in \Pi_W$*, the cost function (only nonlinear term)* $\sum_{l'=0}^{|\mathbb{I}_W|-1} W(\mathbb{X}_{\pi[l']}, \mathbb{X}_{\pi[l'+1]})$ *is a constant. In general, finding a solution can be difficult when the cardinality of the wait task set* W *is large. To cope with this issue, we assume that the cardinality of* W *is upper bounded by* $\bar{m}$*. Moreover, we note that the only integer constraint is (5.12b). Nevertheless, since the number of QoSa levels, given by* $k_{\pi[l]}$*, is finite for each task, each integer parameter* $\hat{k}_{\pi[l]}$ *has only a limited number of choices. Therefore, we conclude that the optimization program* $\mathcal{P}(W, \mathbb{X}_0, t)$ *can be solved efficiently since it is equivalent to solving a finite number (*$\leq \bar{m}! \sum_{l \in \mathbb{I}_W} k_l$*) of linear programs.*

Remark 5.6. *The purpose of this chapter is to maximize the objective function* $\boldsymbol{J}(\mathbb{P}_{\eta(t)})$*. Therefore, fulfillment of all tasks is a soft constraint in view of the aforementioned maximization. However, we note that by setting the absolute value of the rejection penalty* $|R_l[0]|, \forall l$*, to be high enough (e.g., higher than the sum of the highest rewards of all tasks being activated), one can verify that the proposed dynamic scheduling algorithm is optimal in the sense that the* miss ratio[3] *is minimized.*

5.3.3 Control law synthesis

Given the information $\chi[k] = (E[k], t_0^k, Q_k)$, the next step is to do control synthesis. if $\chi[k] = \emptyset$, all agents will switch to the idle mode, and the control law is given by

$$u_i^{\text{idle}} = -\sum_{j \in \mathcal{N}_i} (x_i - x_j) - v_i, \forall i \in \mathcal{V}.$$

Otherwise, we assume without loss of generality that

$$E[k] = \phi_l, \ \mathrm{I}_l[Q_k] = (\underline{t}, \bar{t}], \tag{5.14}$$

which means that the kth task being executed is ϕ_l and the time interval corresponding to QoSa level Q_k of task ϕ_l is $(\underline{t}, \bar{t}]$. Then, we have the following proposition.

Proposition 5.1. *Given the task* ϕ_l *and the starting time* t_0^k*,* ϕ_l *is completed at QoSa level* Q_k *if the conditions*

[3] For a sequence of dynamically activated tasks, the *miss ratio* is defined as the number of reject tasks divided by the total number of activated tasks [93].

C1. $\forall t \in [t_0^k, a_l + \underline{t}]$, $\exists i \in \mathcal{V}_L$ *such that* $x_i(t) \notin \mathbb{X}_l$*; and*

C2. $\exists t \in (a_l + \underline{t}, a_l + \bar{t}]$, $\forall i \in \mathcal{V}$ *such that* $x_i(t) \in \mathbb{X}_l$

hold simultaneously, where a_l *is the activation time of* ϕ_l*.*

Before presenting the control synthesis, the following assumption is needed.

Assumption 5.1. *Let there be given the leader-follower MAS (5.1) with the leader set* $\mathcal{V}_L$ *and the target region* $\mathbb{X}_l, l \in \{1, \ldots, M\}$*, with its Chebyshev center* c_l*. There exist constants* $\sigma, d_s > 0$ *such that*

- $\mathcal{B}(c_l, \sigma) \subseteq \mathbb{X}_l$ *and*
- $x_i \in \mathcal{B}(c_l, \sigma), \forall i \in \mathcal{V}_L$ *and* $\|x_i - x_j\| \leq d_s, \forall (i,j) \in \mathcal{E}$*, imply* $x_i \in \mathbb{X}_l, \forall i \in \mathcal{V}$*.*

Remark 5.7. *Assumption 5.1 is not conservative because i) the constants* σ, d_s *can be different for different target regions and ii) one feasible choice of* σ, d_s *for* $\mathbb{X}_l$ *is* $\sigma = \underline{r}_l/2, d_s = \underline{r}_l/(2(N-1))$*, where* $\underline{r}_l$ *is the radius of the minimal enclosing ball of* $\mathbb{X}_l$ *centered at* c_l*.*

In addition, the following definitions are introduced.

Definition 5.2. *[85] A function* $\rho : \mathbb{R}_{\geq 0} \to \mathbb{R}_{>0}$ *will be called a* performance function *if* ρ *is bounded, nonnegative and non-increasing.*

Definition 5.3. *[85] A function* $S : \mathbb{R} \to \mathbb{R}$ *will be called a* transformation function *if* S *is strictly increasing, injective and admitting an inverse.*

In particular, we define two transformation functions S_1 and S_2 as

$$S_1(z) = \ln(\frac{1}{1-z}), S_2(z) = \ln(\frac{1+z}{1-z}).$$

For leader agent $i \in \mathcal{V}_L$, we prescribe the norm of the tracking error $x_i(t) - c_l$ within the following bounds,

$$\alpha_i(t) < \|x_i(t) - c_l\| < \beta_i(t), \quad i \in \mathcal{V}_L. \tag{5.15}$$

However, since the follower agents do not know where the target region is, we prescribe the norm of the relative distance between neighboring agents within the following bounds,

$$\|x_{ij}(t)\| < \gamma_i(t), \quad i \in \mathcal{V}, (i,j) \in \mathcal{E}, \tag{5.16}$$

where $\alpha_i(t), \beta_i(t), \gamma_i(t)$ are performance functions to be defined.

The dynamic scheduling algorithm in Section IV-A ensures that $a_l + \bar{t} > t_0^k$. However, it is possible that $a_l + \underline{t} \leq t_0^k$. In this case, C1 of Proposition 5.1 is meaningless, and one only needs to design performance functions such that C2 of Proposition 5.1 is satisfied. Nevertheless, in the other case, *i.e.,* $a_l + \underline{t} > t_0^k$, it is more complicated to design the performance functions since one needs to ensure that both C1 and C2 are satisfied. Therefore, in the following, we will present the design of the performance functions and the control synthesis according to the two different cases, respectively.

Remark 5.8. *Note that if at time t_0^k, one has $x_i(t_0^k) \in \mathbb{X}_l, \forall i \in \mathcal{V}$, i.e., all agents are already in the target region $\mathbb{X}_l$ at t_0^k, then according to the proposed dynamic scheduling strategy, the algorithm Completion will be activated, and the group of agents will proceed to the next task (no control action is needed).*

Case I: $a_l + \underline{t} \leq t_0^k$

Define

$$\alpha_i(t) = 0, \tag{5.17}$$

$$\beta_i(t) = \beta_{i0} e^{-\kappa_{i,1}(t - t_0^k)}, \tag{5.18}$$

$$\gamma_i(t) = \gamma_{i0} e^{-\mu_{i,1}(t - t_0^k)}, \tag{5.19}$$

for $t \geq t_0^k$, where $\beta_{i0} > \max\{\|x_i(t_0^k) - c_l\|, \sigma\}$, $\gamma_{i0} > \max\{\max_{j \in \mathcal{N}_i}\{\|x_{ij}(t_0^k)\|\}, d_s\}$, and σ, d_s satisfy Assumption 5.1. In addition,

$$\kappa_{i,1} = \frac{1}{(a_l + \bar{t} - t_0^k)} \ln \frac{\beta_{i0}}{\sigma}, \tag{5.20}$$

$$\mu_{i,1} = \frac{1}{(a_l + \bar{t} - t_0^k)} \ln \frac{\gamma_{i0}}{d_s}. \tag{5.21}$$

Remark 5.9. *The definitions of β_{i0}, γ_{i0} guarantee that the performance bounds (5.15) and (5.16) are satisfied at the starting time t_0^k. In addition, from (5.18) and (5.20) one can get*

$$\|x_i(a_l + \bar{t}) - c_l\| < \beta_i(a_l + \bar{t}) = \beta_{i0} e^{-\kappa_{i,1}(a_l + \bar{t} - t_0^k)} = \sigma, \forall i \in \mathcal{V}_L. \quad (5.22)$$

Moreover, from (5.19) and (5.21) one can get

$$\|x_{ij}(a_l + \bar{t})\| < \gamma_i(a_l + \bar{t}) = \gamma_{i0} e^{-\mu_{i,1}(a_l + \bar{t} - t_0^k)} = d_s, \forall (i,j) \in \mathcal{E}. \quad (5.23)$$

According to Assumption 5.1, (5.22) and (5.23) together imply $x_i(a_l + \bar{t}) \in \mathbb{X}_l, \forall i \in \mathcal{V}$. Therefore, C2 is satisfied.

Based on Remark 5.9, one can conclude that if for task ϕ_l,

i) the tracking error $\|x_i - c_l\|, \forall i \in \mathcal{V}_L$ is evolving within the prescribed performance bound (5.15), and

ii) the relative distance $\|x_{ij}\|, \forall (i,j) \in \mathcal{E}$ is evolving within the prescribed performance bound (5.16) for $t \geq t_0^k$,

then the task ϕ_l will be completed at the desired QoSa level Q_k.

Define the normalized errors as

$$\xi_i(t) := \frac{\|x_i(t) - c_l\|}{\beta_i(t)}, \quad \xi_{ij}(t) := \frac{\|x_{ij}(t)\|}{\gamma_i(t)}, \quad (5.24)$$

respectively. Then, the corresponding sets

$$D_{\xi_i} := \{\xi_i(t) : \xi_i(t) \in [0,1)\} \quad (5.25)$$

$$D_{\xi_{ij}} := \{\xi_{ij}(t) : \xi_{ij}(t) \in [0,1)\} \quad (5.26)$$

are equivalent to (5.15) and (5.16), respectively. The normalized errors ξ_i and ξ_{ij} are transformed through the transformation function S_1. We denote the transformed error $\zeta_i(\xi_i)$ and $\varepsilon_{ij}(\xi_{ij})$ by

$$\zeta_i(\xi_i) := S_1(\xi_i), \quad \varepsilon_{ij}(\xi_{ij}) := S_1(\xi_{ij}), \quad (5.27)$$

respectively, where we drop the time argument t for notation convenience.

Let $\nabla S_1(\xi_i) = \partial S_1(\xi_i)/\partial \xi_i, \nabla S_1(\xi_{ij}) = \partial S_1(\xi_{ij})/\partial \xi_{ij}$. The control laws for the leader and follower agents are proposed respectively as:

$$\begin{aligned} u_i^{\text{lead}} = &- \sum_{j\in\mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} \\ &- \frac{1}{\beta_i} \nabla S_1(\xi_i) \zeta_i(\xi_i) \mathbf{n}_i - K_i v_i, \end{aligned} \tag{5.28}$$

and

$$u_i^{\text{fol}} = - \sum_{j\in\mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} - K_i v_i, \tag{5.29}$$

where $\mathbf{n}_i = (x_i - c_l)/\|x_i - c_l\|, \mathbf{n}_{ij} = (x_i - x_j)/\|x_{ij}\|$, and K_i is a positive control gain to be determined later.

Let $\bar{\xi}_k = \xi_{ij}, \bar{\varepsilon}_k = \varepsilon_{ij}, y_i = x_i - c_l$. Let also $\bar{\varepsilon} = (\bar{\varepsilon}_1(\bar{\xi}_1), \ldots, \bar{\varepsilon}_p(\bar{\xi}_p)), \bar{\zeta} = (\zeta_1(\xi_1), \ldots, \zeta_N(\xi_N))$ be the stack vector of the transformed errors and $y = (y_1, \ldots, y_N)$. Define the following function

$$\begin{aligned} V_1(y, v, \bar{\varepsilon}, \bar{\zeta}) = &\frac{1}{2} \begin{bmatrix} y & v \end{bmatrix} \left\{ \begin{bmatrix} K\theta & \theta \\ \theta & I_N \end{bmatrix} \otimes I_n \right\} \begin{bmatrix} y \\ v \end{bmatrix} \\ &+ \frac{1}{2} \bar{\varepsilon}^T \bar{\varepsilon} + \frac{1}{2} \bar{\zeta}^T H \bar{\zeta}, \end{aligned} \tag{5.30}$$

where $H \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries h_i ($h_i = 0$ if $i \in \mathcal{V}_L$ and $h_i = 0$ otherwise), $K \in \mathbb{R}^{N\times N}$ is a diagonal matrix with entries K_i, and θ is a diagonal matrix with entries $\theta_i = \max\{\mu_{i,1}, \kappa_{i,1}\}$.

Let $V_1^0 := V_1(y(t_0^k), v(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k)$ and $\Theta := S_1^{-1}(\sqrt{2V_1^0})$. Then, the following holds.

Theorem 5.1. *Consider the leader-follower MAS (5.1) and the kth execution task $\chi[k]$ given in (5.14). The control inputs for the leader and follower agents are given by (5.28) and (5.29), respectively. Suppose Assumption 5.1 holds and $a_l + \underline{t} \leq t_0^k$. Assume that the control gain $K_i > \max\{\mu_{i,1}, \kappa_{i,1}\}, \forall i$, and the input constraints for leader and follower agents satisfy*

$$u_i^{\max} \geq \left(\frac{|\mathcal{N}_i|}{d_s(1-\Theta)} + \frac{1}{\sigma(1-\Theta)} \right) \sqrt{2V_1^0} + K_i(\sqrt{2V_1^0} + \theta_i \beta_{i0}), \quad \forall i \in \mathcal{V}_L,$$

and

$$u_i^{\max} \geq \frac{|\mathcal{N}_i|}{d_s(1-\Theta)} \ln \frac{1}{(1-\Theta)} + K_i(\sqrt{2V_1^0} + \theta_i \beta_{i0}), \forall i \in \mathcal{V}_F,$$

respectively. Then, i) the tracking error $\|x_i - c_l\|, \forall i \in \mathcal{V}_L$ *and the relative distance* $\|x_{ij}\|, (i,j) \in \mathcal{E}$ *will evolve respectively within the prescribed performance bounds (5.15) and (5.16) for all* $t \geq t_0^k$ *and ii) the input constraint for each agent* i*, i.e.,* $u_i(t) \in U_i, \forall i$*, will be satisfied for all* $t \in [t_0^k, a_l + \bar{t}]$.

Proof. Since $K_i > \max\{\mu_{i,1}, \kappa_{i,1}\} = \theta_i, \forall i$, one can derive that $V(y, v, \bar{\varepsilon}, \bar{\zeta})$ is nonnegative for all $y, v, \bar{\varepsilon}, \bar{\zeta}$.

Differentiating (5.30) along the trajectories of (5.1), one has

$$\begin{aligned}\dot{V}_1(y, v, \bar{\varepsilon}, \bar{\zeta}) =& y^T K\theta v + y^T \theta u^{\chi[k]} + v^T \theta v + v^T u^{\chi[k]} \\ &+ \bar{\varepsilon}^T \dot{\bar{\varepsilon}} + \bar{\zeta}^T H \dot{\bar{\zeta}},\end{aligned} \tag{5.31}$$

where

$$u_i^{\chi[k]} = \begin{cases} u_i^{\text{lead}}, & \text{if } i \in \mathcal{V}_L \\ u_i^{\text{fol}}, & \text{if } i \in \mathcal{V}_F. \end{cases}$$

Substituting (5.28) and (5.29) into (5.31), we obtain

$$\begin{aligned}\dot{V}_1(y, v, \bar{\varepsilon}, \bar{\zeta}) =& -\sum_{i=1}^{N} \theta_i y_i^T \sum_{j \in \mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} \\ &- \sum_{i=1}^{N} \theta_i y_i^T \frac{h_i}{\beta_i} \nabla S_1(\xi_i) \zeta_i(\xi_i) \mathbf{n}_i \\ &- \sum_{i=1}^{N} v_i^T \sum_{j \in \mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} \\ &- \sum_{i=1}^{N} v_i^T \frac{h_i}{\beta_i} \nabla S_1(\xi_i) \zeta_i(\xi_i) \mathbf{n}_i \\ &- \sum_{i=1}^{N} (K_i - \theta_i) v_i^T v_i + \sum_{i=1}^{p} \varepsilon_{ij}(\xi_{ij}) \nabla S_1(\xi_{ij}) \dot{\xi}_{ij} \\ &+ \sum_{i=1}^{N} h_i \zeta_i(\xi_i) \nabla S_1(\xi_i) \dot{\xi}_i.\end{aligned} \tag{5.32}$$

According to (5.24), one can get

$$\dot{\xi}_{ij} = \frac{1}{\gamma_i}\frac{\|x_{ij}\|'\gamma_i - \|x_{ij}\|\dot{\gamma}_i}{\gamma_i} = \frac{1}{\gamma_i}\Big(\|x_{ij}\|' + \mu_{i,1}\|x_{ij}\|\Big),$$
$$\dot{\xi}_i = \frac{1}{\beta_i}\frac{\|x_i - c_l\|'\beta_i - \|x_i - c_l\|\dot{\beta}_i}{\beta_i} = \frac{1}{\beta_i}\Big(\|y_i\|' + \kappa_{i,1}\|y_i\|\Big),$$

where $-\dot{\beta}_i/\beta_i \equiv \kappa_{i,1}$ and $-\dot{\gamma}_i/\gamma_i \equiv \mu_{i,1}$.

Due to symmetry, one has

$$\frac{\partial\|x_{ij}\|}{\partial x_{ij}} = \frac{\partial\|x_{ij}\|}{\partial x_i} = -\frac{\partial\|x_{ij}\|}{\partial x_j},$$

and from (5.1), we get

$$\begin{aligned}\sum_{(i,j)\in\mathcal{E}} \nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|' =& \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\frac{\partial\|x_{ij}\|}{\partial x_{ij}}\dot{x}_{ij}\\ =& \sum_{i=1}^{N} v_i^T\sum_{j\in\mathcal{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij}.\end{aligned}$$

In addition,

$$\begin{aligned}\sum_{i=1}^{N} y_i^T\sum_{j\in\mathcal{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij} =& \sum_{i=1}^{N} y_i^T\sum_{j\in\mathcal{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\frac{y_i - y_j}{\|y_i - y_j\|}\\ =& \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|.\end{aligned}$$

Then, (5.32) can be rewritten as

$$\begin{aligned}\dot{V}_1(y, v, \bar{\varepsilon}, \bar{\zeta}) =& -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\frac{\theta_i - \mu_{i,1}}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\|\\ &-\sum_{i=1}^{N}\frac{h_i(\theta_i - \kappa_{i,1})}{\beta_i}\zeta_i(\xi_i)\nabla S_1(\xi_i)\|y_i\| - (K-\theta)v^T v.\end{aligned} \tag{5.33}$$

According to the definition of S_1, one can derive that $\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \geq 0$ and $\zeta_i(\xi_i)\nabla S_1(\xi_i)\|y_i\| \geq 0$. In addition, $\theta_i - \mu_{i,1} \geq 0$ and $\theta_i - \kappa_{i,1} \geq 0$

for all i. Therefore, one derives that $\dot{V}_1(y, v, \bar{\varepsilon}, \bar{\zeta}) \leq 0$, which in turn implies $V_1(y, v, \bar{\varepsilon}, \bar{\zeta}) \leq V_1(y(t_0^k), v(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k) := V_1^0$ and thus

$$|S_1(\xi_i)| = |\zeta_i(\xi_i)| \leq |\bar{\zeta}| \leq \sqrt{2V_1^0}, \forall i \in \mathcal{V}_L,$$
$$|S_1(\xi_{ij})| = |\varepsilon_{ij}(\xi_{ij})| \leq |\bar{\varepsilon}| \leq \sqrt{2V_1^0}, \forall (i,j) \in \mathcal{E},$$

for all $t \geq t_0^k$. Moreover, $\xi_i(t_0^k), \forall i \in \mathcal{V}_L$ and $\xi_{ij}(t_0^k), \forall (i,j) \in \mathcal{E}$ are within the regions (5.25) and (5.26), respectively. By using the inverse of S_1, we can bound $0 \leq \xi_i(t) \leq S_1^{-1}\left(\sqrt{2V_1^0}\right) < 1$ and $0 \leq \xi_{ij}(t) \leq S_1^{-1}\left(\sqrt{2V_1^0}\right) < 1$ for all $t > t_0^k$. In particular, $\xi_i(t), \forall i \in \mathcal{V}_L$ and $\xi_{ij}(t), \forall (i,j) \in \mathcal{E}$ will evolve within the regions (5.25) and (5.26) for all $t \geq t_0^k$. Since (5.25) and (5.26) are equivalent to the prescribed performance bounds (5.15) and (5.16), respectively, item i) of Theorem 5.1 holds.

Since $\xi_i(t), \xi_{ij}(t) \in [0, \Theta)$ and the functions β_i, γ_i are monotonically decreasing, one has from (5.28) that

$$\begin{aligned}\|u_i^{\text{lead}}(t)\| \leq & \left| \sum_{j \in \mathcal{N}_i} \frac{1}{\gamma_i(t)} \nabla S_1(\xi_{ij}(t)) \varepsilon_{ij}(\xi_{ij}(t)) \right| + \left| \frac{1}{\beta_i(t)} \nabla S_1(\xi_i(t)) \zeta_i(\xi_i(t)) \right| \\ & + \|K_i v_i(t)\| \\ \leq & \left(\frac{|\mathcal{N}_i|}{\gamma_i(t_0^k + \bar{t})(1-\Theta)} + \frac{1}{\beta_i(t_0^k + \bar{t})(1-\Theta)} \right) \sqrt{2V_1^0} + \|K_i v_i(t)\| \end{aligned} \tag{5.34}$$

for $t \in [t_0^k, a_l + \bar{t}]$. In addition,

$$\begin{aligned} 2V_1^0 \geq & [y \quad v] \left\{ \begin{bmatrix} K\theta & \theta \\ \theta & I_N \end{bmatrix} \otimes I_n \right\} \begin{bmatrix} y \\ v \end{bmatrix} \\ \geq & \sum_{i=1}^{N} (\|v_i\| - \theta_i \|y_i\|)^2 \\ = & \sum_{i=1}^{N} (\|v_i\| - \theta_i \|x_i - c_l\|)^2 \\ \geq & \sum_{i=1}^{N} (\|v_i\| - \theta_i \beta_{i0})^2. \end{aligned}$$

Then, one has

$$\|v_i(t)\| \leq \sqrt{2V_1^0} + \theta_i\beta_{i0}, t \in [t_0^k, a_l + \bar{t}]. \tag{5.35}$$

Combining (5.34), (5.35) and $\gamma_i(a_l + \bar{t}) = d_s, \beta_i(a_l + \bar{t}) = \sigma$, one can further get

$$\begin{aligned}\|u_i^{\text{lead}}(t)\| \leq & \left(\frac{|\mathcal{N}_i|}{d_s(1-\Theta)} + \frac{1}{\sigma(1-\Theta)}\right)\sqrt{2V_1^0} + K_i(\sqrt{2V_1^0} + \theta_i\beta_{i0}) \\ \leq & u_i^{\max}, \forall i \in \mathcal{V}_L, \forall t \in [t_0^k, a_l + \bar{t}].\end{aligned}$$

Similarly, one can also derive that $\|u_i^{\text{fol}}(t)\| \leq u_i^{\max}, \forall i \in \mathcal{V}_F, \forall t \in [t_0^k, a_l + \bar{t}]$. Item ii) of Theorem 5.1 holds. □

Case II: $a_l + \underline{t} > t_0^k$

For a given set $\mathbb{X}$, the indicator function is defined as

$$\mathbb{1}_{\mathbb{X}}(x) = \begin{cases} 1, & \text{if} \quad x \in \mathbb{X} \\ 0, & \text{if} \quad x \notin \mathbb{X}. \end{cases}$$

Then, the performance functions $\alpha_i, \beta_i, \gamma_i$ are defined as

$$\alpha_i(t) = \begin{cases} 0, & t \geq t_0^k, \quad \text{if } \mathbb{1}_{\bar{\mathbb{X}}_l}(x_i(t_0^k)) = 1, \\ 0, & t > a_l + \underline{t}, \quad \text{if } \mathbb{1}_{\bar{\mathbb{X}}_l}(x_i(t_0^k)) = 0, \\ \alpha_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, t \in [t_0^k, a_l + \underline{t}], & \text{if } \mathbb{1}_{\bar{\mathbb{X}}_l}(x_i(t_0^k)) = 0 \end{cases} \tag{5.36}$$

$$\beta_i(t) = \begin{cases} \beta_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, & t \in [t_0^k, a_l + \underline{t}] \\ \beta_i(\underline{t})e^{-\kappa_{i,3}(t-a_l-\underline{t})}, & t > a_l + \underline{t}, \end{cases} \tag{5.37}$$

$$\gamma_i(t) = \begin{cases} \gamma_{i0}e^{-\kappa_{i,2}(t-t_0^k)}, & t \in [t_0^k, a_l + \underline{t}] \\ \gamma_i(\underline{t})e^{-\mu_{i,2}(t-a_l-\underline{t})}, & t > a_l + \underline{t}, \end{cases} \tag{5.38}$$

where

$$\alpha_{i0} = \|x_i(t_0^k) - c_l\| - \Delta_i, \tag{5.39a}$$

$$\beta_{i0} = \|x_i(t_0^k) - c_l\| + \Delta_i, \tag{5.39b}$$

$$\gamma_{i0} > \max\left\{\max_{j \in \mathcal{N}_i}\{\|x_{ij}(t_0^k)\|\}, d_s\right\}, \tag{5.39c}$$

and $0 < \Delta_i < \|x_i(t_0^k) - c_l\| - \bar{r}_l, \forall i \in \mathcal{V}_L$. In addition,

$$\kappa_{i,2} = \frac{1}{(a_l + \underline{t} - t_0^k)} \ln \frac{\alpha_{i0}}{\bar{r}_l}, \tag{5.40a}$$

$$\kappa_{i,3} = \frac{1}{(\bar{t} - \underline{t})} \ln \frac{\beta_{i0}\bar{r}_l}{\sigma\alpha_{i0}}, \tag{5.40b}$$

$$\mu_{i,2} = \frac{1}{(\bar{t} - \underline{t})} \ln \frac{\gamma_{i0}\bar{r}_l}{d_s\alpha_{i0}}, \tag{5.40c}$$

where $\bar{r}_l$ defined in (5.4) is the radius of $\overline{\mathbb{X}}_l$.

One can verify that the functions $\alpha_i, \beta_i, \gamma_i$ satisfy Definition 5.2. The function α_i is designed to ensure condition C1. The functions β_i, γ_i are designed to ensure condition C2. However, for the special case $x_i(t_0^k) \in \overline{\mathbb{X}}_l, \forall i \in \mathcal{V}_L$, *i.e.,* all leader agents are already in the region $\overline{\mathbb{X}}_l$ at the starting time t_0^k, it is not always possible to satisfy C1.

Corollary 5.1. *Suppose Assumption 5.1 holds and $\exists i \in \mathcal{V}_L, x_i(t_0^k) \notin \overline{\mathbb{X}}_l$. The performance functions α_i, β_i and γ_i defined in (5.36), (5.37) and (5.38) guarantee that the conditions C1 and C2 are satisfied simultaneously.*

Proof. Firstly, the definitions of α_{i0}, β_{i0} and γ_{i0} guarantee that the performance bounds (5.15) and (5.16) are satisfied at the starting time t_0^k. From (5.36) and (5.40a), one can derive that

$$\begin{aligned}\|x_i(a_l + \underline{t}) - c_l\| > \alpha_i(a_l + \underline{t}) &= \alpha_{i0} e^{-\kappa_{i,2}(a_l + \underline{t} - t_0^k)} \\ &= \bar{r}_l,\ \forall i \in \mathcal{V}_L \wedge \mathbb{1}_{\overline{\mathbb{X}}_l}(x_i(t_0^k)) = 0.\end{aligned}$$

Hence, C1 is satisfied.

From (5.37), (5.40b) and (5.38), (5.40c), one can derive that

$$\|x_i(a_l + \bar{t}) - c_l\| < \beta_i(a_l + \bar{t}) = \beta_i(\underline{t}) e^{-\kappa_{i,3}(\bar{t} - \underline{t})} = \sigma,\ \forall i \in \mathcal{V}_L,$$

and

$$\|x_{ij}(a_l + \bar{t})\| < \gamma_i(a_l + \bar{t}) = \gamma_i(\underline{t}) e^{-\mu_{i,2}(\bar{t} - \underline{t})} = d_s,\ \forall (i,j) \in \mathcal{E},$$

respectively. According to Assumption 5.1, one can conclude that $x_i(a_l + \bar{t}) \in \mathbb{X}_l, \forall i \in \mathcal{V}$. Hence, C2 is satisfied. □

Let $\rho_i(t) := (\beta_i(t) + \alpha_i(t))/2$ and $\delta_i(t) := (\beta_i(t) - \alpha_i(t))/2$. Then, (5.15) can be rewritten as

$$-\delta_i(t) + \rho_i(t) < \|x_i(t) - c_l\| < \rho_i(t) + \delta_i(t). \tag{5.41}$$

Define in this case the normalized error $\xi_i(t)$ as

$$\xi_i(t) := \frac{\|x_i(t) - c_l\| - \rho_i(t)}{\delta_i(t)},$$

and ξ_{ij} is defined the same as in (5.24). Then, the set

$$\hat{D}_{\xi_i} := \{\xi_i(t) : \xi_i(t) \in (-1, 1)\} \tag{5.42}$$

is equivalent to (5.15) in this case. The normalized error ξ_i is transformed through the transformation function $S2$. We denote the transformed errors $\zeta_i(\xi_i)$ and $\varepsilon_{ij}(\xi_{ij})$ by

$$\zeta_i(\xi_i) := S_2(\xi_i), \quad \varepsilon_{ij}(\xi_{jk}) := S_1(\xi_{ij}), \tag{5.43}$$

respectively.

Let $\nabla S_2(\xi_i) = \partial S_2(\xi_i)/\partial \xi_i, \nabla S_1(\xi_{ij}) = \partial S_1(\xi_{ij})/\partial \xi_{ij}$. The control laws for the leader and follower agents are proposed respectively as:

$$\begin{aligned} u_i^{\text{lead}} = &- \sum_{j \in \mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} \\ &- \frac{1}{\delta_i} \nabla S_2(\xi_i) \zeta_i(\xi_i) \mathbf{n}_i - K_i v_i, \end{aligned} \tag{5.44}$$

and

$$u_i^{\text{fol}} = - \sum_{j \in \mathcal{N}_i} \frac{1}{\gamma_i} \nabla S_1(\xi_{ij}) \varepsilon_{ij}(\xi_{ij}) \mathbf{n}_{ij} - K_i v_i, \tag{5.45}$$

where K_i is a positive control gain to be determined.

Let $z = (y, v)^T$. Define the following function

$$V_2(z, \bar{\varepsilon}, \bar{\zeta}) = \begin{cases} \frac{1}{2}(z^T G_1 z + + \bar{\varepsilon}^T \bar{\varepsilon} + \bar{\zeta}^T H \bar{\zeta}), \ t \in [t_0^k, a_l + \underline{t}) \\ \frac{1}{2}(z^T G_2 z + + \bar{\varepsilon}^T \bar{\varepsilon} + \bar{\zeta}^T H \bar{\zeta}), \ t \geq a_l + \underline{t} \end{cases} \tag{5.46}$$

where

$$G_1 = \begin{bmatrix} K\kappa_2 & \kappa_2 \\ \kappa_2 & I_N \end{bmatrix} \otimes I_n, G_2 = \begin{bmatrix} K\kappa_3 & \kappa_3 \\ \kappa_3 & I_N \end{bmatrix} \otimes I_n, \tag{5.47}$$

and $\kappa_2, \kappa_3 \in \mathbb{R}^{N\times N}$ are diagonal matrices with entries $\kappa_{i,2}$ and $\kappa_{i,3}$, respectively. The matrices H, K are defined in (5.30).

Denote $V_2^0 := V_2(z(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k))$. Let $V_2^* = V_2^0 + \sum_{i=1}^N \Big(K_i|\kappa_{i,2} - \kappa_{i,3}|\beta_{i0}^2 + 2|\kappa_{i,2} - \kappa_{i,3}|\beta_{i0}(\sqrt{2V_2^0} + \kappa_{i,2}\beta_{i0})\Big)$ and $\Theta_1 := S_1^{-1}(\sqrt{2V_2^*}), \Theta_2 := S_2^{-1}(\sqrt{2V_2^*})$. Then, the following holds.

Theorem 5.2. *Consider the leader-follower MAS (5.1) and the kth execution task $\chi[k]$ given in (5.14). The control inputs for the leader and follower agents are given by (5.44) and (5.45), respectively. Suppose Assumption 5.1 holds, $\underline{t} > a_l + t_0^k$, and $\exists i \in \mathcal{V}_L, x_i(t_0^k) \notin \overline{\mathbb{X}}_l$. Assume that the control gain $K_i > \max\{\kappa_{i,2}, \kappa_{i,3}\}, \forall i$, the constant γ_{i0} in (5.39c) satisfies $\gamma_{i0} < d_s\alpha_{i0}e^{\kappa_{i,3}(\underline{t}-\bar{t})}/\bar{r}_l, \forall i$, and the input constraints for leader and follower agents satisfy*

$$u_i^{\max} \geq \left(\frac{|\mathcal{N}_i|}{d_s(1-\Theta_1)} + \frac{4}{\sigma(1-\Theta_2^2)}\right)\sqrt{2V_2^*} + K_i(\sqrt{2V_2^*} + K_i\beta_{i0}), \forall i \in \mathcal{V}_L,$$

and

$$u_i^{\max} \geq \frac{|\mathcal{N}_i|}{d_s(1-\Theta_1)}\sqrt{2V_2^*} + K_i(\sqrt{2V_2^*} + K_i\beta_{i0}), \forall i \in \mathcal{V}_F,$$

respectively. Then, i) the tracking error $\|x_i - c_l\|, \forall i \in \mathcal{V}_L$ and the relative distance $\|x_{ij}\|, (i,j) \in \mathcal{E}$ will evolve respectively within the prescribed performance bounds (5.15) and (5.16) for all $t \geq t_0^k$ and ii) the input constraint for each agent i, i.e., $u_i(t) \in U_i, \forall i$, will be satisfied for all $t \in [t_0^k, a_l + \bar{t}]$.

Proof. Since $K_i > \max\{\kappa_{i,2}, \kappa_{i,3}\}, \forall i$, one can derive $G_1 \succ 0, G_2 \succ 0$. Therefore, $V_2(z, \bar{\varepsilon}, \bar{\zeta})$ is nonnegative for all $z, \bar{\varepsilon}, \bar{\zeta}$.

For $t \in [t_0^k, \underline{t}]$, differentiating (5.46) along the trajectories of (5.1) and sub-

stituting (5.44) and (5.45), one has

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) =& -\sum_{i=1}^{N}\kappa_{i,2}y_i^T\sum_{j\in\mathcal{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij}\\ &-\sum_{i=1}^{N}\kappa_{i,2}y_i^T\frac{h_i}{\delta_i}\nabla S_2(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i\\ &-\sum_{i=1}^{N}v_i^T\sum_{j\in\mathcal{N}_i}\frac{1}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\mathbf{n}_{ij}\\ &-\sum_{i=1}^{N}v_i^T\frac{h_i}{\delta_i}\nabla S_2(\xi_i)\zeta_i(\xi_i)\mathbf{n}_i\\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,2})v_i^Tv_i+\sum_{i=1}^{p}\varepsilon_{ij}(\xi_{ij})\nabla S_1(\xi_{ij})\dot{\xi}_{ij}\\ &+\sum_{i=1}^{n}h_i\zeta_i(\xi_i)\nabla S_2(\xi_i)\dot{\xi}_i,\end{aligned}\tag{5.48}$$

where

$$\begin{aligned}\dot{\xi}_{ij} =& \frac{1}{\gamma_i}\frac{\|x_{ij}\|'\gamma_i-\|x_{ij}\|\dot{\gamma}_i}{\gamma_i},\\ \dot{\xi}_i =& \frac{1}{\delta_i}\frac{(\|x_i-c_l\|'-\dot{\rho}_i)\delta_i-(\|x_i-c_l\|-\rho_i)\dot{\delta}_i}{\delta_i}\\ =& \frac{1}{\delta_i}\frac{(\|y_i\|'-\dot{\rho}_i)\delta_i-(\|y_i\|-\rho_i)\dot{\delta}_i}{\delta_i}.\end{aligned}$$

Similar to the proof of Theorem 5.1, one can further get

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) = &-\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\frac{\kappa_{i,2}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \\ &-\sum_{i=1}^{N}\frac{h_i(\kappa_{i,2}-\hat{\delta}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)(\|y_i\|-\rho_i) \\ &-\sum_{i=1}^{N}\frac{h_i\rho_i(\kappa_{i,2}-\hat{\rho}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i) \\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,2})v_i^T v_i,\end{aligned}$$

where $\hat{\gamma}_i = -\dot{\gamma}_i/\gamma_i$, $\hat{\delta}_i = -\dot{\delta}_i/\delta_i$ and $\hat{\rho}_i = -\dot{\rho}_i/\rho_i$. In addition, according to the definition of $\gamma_i(t), \delta_i(t)$ and $\rho_i(t)$, one can derive that $\hat{\gamma}_i(t) = \hat{\delta}_i(t) = \hat{\rho}_i(t) \equiv \kappa_{i,2}$ for all $t \in [t_0^k, \underline{t}]$. Therefore,

$$\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) \leq -\sum_{i=1}^{N}(K_i-\kappa_{i,2})v_i^T v_i \leq 0, \quad \forall[t_0^k,\underline{t}]. \tag{5.49}$$

For $t > \underline{t}$, differentiating (5.46) along the trajectories of (5.1) and substituting (5.44) and (5.45), one has

$$\begin{aligned}\dot{V}_2(z,\bar{\varepsilon},\bar{\zeta}) = &-\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\frac{\kappa_{i,3}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \\ &-\sum_{i=1}^{N}\frac{h_i(\kappa_{i,3}-\hat{\delta}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i)(\|y_i\|-\rho_i) \\ &-\sum_{i=1}^{N}\frac{h_i\rho_i(\kappa_{i,3}-\hat{\rho}_i)}{\delta_i}\zeta_i(\xi_i)\nabla S_2(\xi_i) \\ &-\sum_{i=1}^{N}(K_i-\kappa_{i,3})v_i^T v_i,\end{aligned} \tag{5.50}$$

where $\hat{\delta}_i(t) = \hat{\rho}_i(t) \equiv \kappa_{i,3}$ for all $t > \underline{t}$. Then, one can further have

$$\dot{V}_2(z, \bar{\varepsilon}, \bar{\zeta}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\frac{\kappa_{i,3}-\hat{\gamma}_i}{\gamma_i}\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| - \sum_{i=1}^{N}(K_i-\kappa_{i,3})v_i^T v_i.$$

If one chooses $\gamma_{i0} < d_s\alpha_{i0}e^{\kappa_{i,3}(\underline{t}-\bar{t})}/\bar{r}_l, \forall i$, one can verify that $\hat{\gamma}_i < \kappa_{i,3}, \forall i$. In addition, one has $\nabla S_1(\xi_{ij})\varepsilon_{ij}(\xi_{ij})\|x_{ij}\| \geq 0, \forall (i,j) \in \mathcal{E}$. Therefore,

$$\dot{V}_2(z, \bar{\varepsilon}, \bar{\zeta}) \leq 0, \quad \forall t > \underline{t}. \tag{5.51}$$

Combining (5.49) and (5.51), one can get that

$$V_2(z, \bar{\varepsilon}, \bar{\zeta}) \leq \max\{V_2(z(t_0^k), \bar{\varepsilon}(t_0^k), \bar{\zeta}(t_0^k)), V_2(z(\underline{t}), \bar{\varepsilon}(\underline{t}), \bar{\zeta}(\underline{t}))\} \leq V_2^*.$$

Thus

$$|\zeta_i(\xi_i)| \leq |\bar{\zeta}| \leq \sqrt{2V_2^*}, \forall i \in \mathcal{V}_L$$

and

$$|\varepsilon_{ij}(\xi_{ij})| \leq |\bar{\varepsilon}| \leq \sqrt{2V_2^*}, \forall (i,j) \in \mathcal{E},$$

for all $t \geq t_0^k$. Moreover, $\xi_i(t_0^k), \forall i \in \mathcal{V}_L$ and $\xi_{ij}(t_0^k), \forall (i,j) \in \mathcal{E}$ are within the regions (5.42) and (5.26), respectively. By using the inverse of S_1 and S_2, we can bound $-1 < S_2^{-1}(-\sqrt{2V_2^*}) \leq \xi_i(t) \leq S_2^{-1}(\sqrt{2V_2^*}) < 1, \forall i \in \mathcal{V}_L$ and $0 \leq \xi_{ij}(t) \leq S_1^{-1}(\sqrt{2V_2^*}) < 1, \forall (i,j) \in \mathcal{E}$ for $t > t_0^k$. In particular, $\xi_i(t), \forall i \in \mathcal{V}_L$ and $\xi_{ij}(t), \forall (i,j) \in \mathcal{E}$ will evolve within the regions (5.42) and (5.26) for all $t \geq t_0^k$.

The remainder of the proof is similar to that of Theorem 5.1 and hence omitted. □

Remark 5.10. *In the optimization program (5.12), we choose the estimated completion time for each QoSa level (except level 0) as the upper bound of the corresponding time interval (see constraint (5.12e)), which guarantees the feasibility of the plan. However, in real-time execution, a task may be completed before its estimated completion time, allowing for other possibilities of execution. Due to this reason, the obtained plan from the optimization program (5.12) may not be optimal. We note that this is unavoidable since the (actual) completion time of each task can not be foreseen at the scheduling time. Another reason that affects*

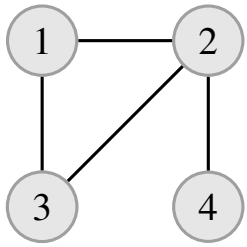

Figure 5.3: Communication graph for the MAS.

the optimality of the plan is the length of the waiting task set W, given by $\bar{m}$. It is pointed out in Remark 5.9 that the computational complexity of (5.12) grows exponentially with respect to the length of W. Therefore, we aim to limit the size of waiting task set W for computational efficiency.

5.4 Example

In this section, a numerical example illustrates the theoretical results. Consider a group of 4 agents with $n = 2$, where communication graph $\mathcal{G}$ is shown in Figure 5.3. The initial position $x_i(0)$ of each agent i is chosen randomly from the box $[0, 2] \times [0, 2]$, and the initial velocity $v_i(0)$ of each agent i is $[0, 0]^T$.

The task set ϕ consists of 6 tasks, each of which is associated with a target region. For the sake of simplicity, it is assumed that each target region $\mathbb{X}_l, l \in \{1, \ldots, 6\}$ has the shape of a ball, denoted by $\mathcal{B}(c_l, r_l)$, where c_l, r_l are the center and radius of the ball, respectively. For each task, the corresponding parameters (QoSa level and the corresponding time interval, reward, target set, activation time set) are summarized in TABLE 5.1. In addition, we further consider the general case that the group of leader and follower agents can be different for different tasks. For each task, the set of leader agents is also given in TABLE 5.1. One can see that 5 tasks ($\{\phi_1, \phi_2, \phi_3, \phi_4, \phi_6\}$) are activated initially. In total, 11 tasks are activated.

Table 5.1: Task specifications.

Task	Time Interval				Target set	Leader set
	Reward					
QoSa	3	2	1	0		
ϕ_1	(0,6]	(6,10]	(10,20]	(20,∞)	$\mathcal{B}((10,8),1)$	$\{1,4\}$
	8	20	5	-20		
ϕ_2	-	-	(0,12]	(12,∞)	$\mathcal{B}((8,2),1)$	$\{1,4\}$
	-	-	10	-20		
ϕ_3	-	(0,8]	(8,14]	(14,∞)	$\mathcal{B}((5,5),0.5)$	$\{1,3\}$
	-	10	20	-20		
ϕ_4	(0,6]	(6,10]	(10,23]	(23,∞)	$\mathcal{B}((15,9),0.8)$	$\{1,2,3,4\}$
	25	5	20	-20		
ϕ_5	-	-	(0,10]	(10,∞)	$\mathcal{B}((5,10),1)$	$\{1,4\}$
	-	-	10	-20		
ϕ_6	-	-	(0,15]	(15,∞)	$\mathcal{B}((20,15),1)$	$\{1,2,4\}$
	-	-	5	-20		

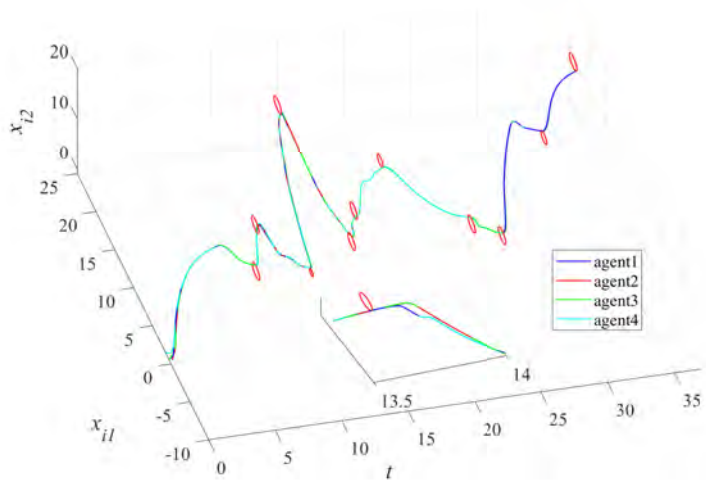

Figure 5.4: The evolution of positions for each agent under the dynamic scheduling strategy.

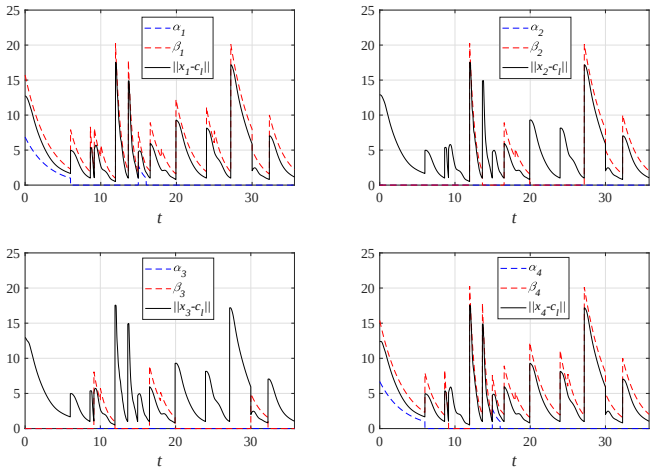

Figure 5.5: The evolution of the tracking error and performance bounds α_i, β_i for each agent. Note that the performance bounds are presented only for the leader agents in each task.

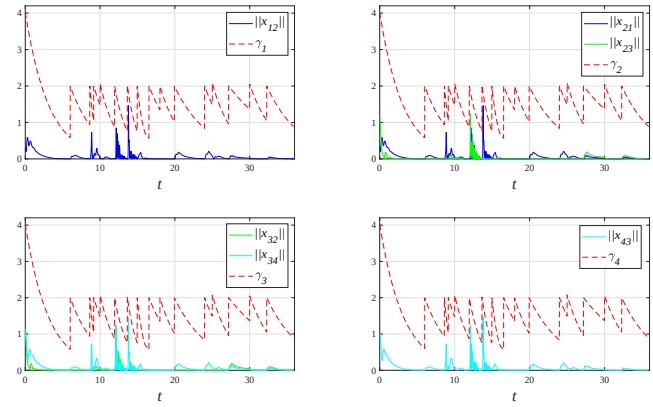

Figure 5.6: The evolution of the relative distances between neighboring agents and performance bounds γ_i for each agent.

In the optimization program (5.12), the parameter ς is chosen as $\varsigma = 1$ (the reason for this choice is to make a fair comparison with the EDF policy in the

following). Firstly, the dynamic scheduling strategy proposed In this chapter is considered, where the cardinality of the waiting task set W is 4. The simulation results are given in Figs. 5.4-5.6. Figure 5.4 shows the evolution of positions for each agent with respect to time t, where x_{i1} and x_{i2} are position components. The 11 red circles (from left to right) represent the 11 target regions that being reached, respectively. In Figure 5.5, the evolution of the tracking error $\|x_i - c_l\|$ for each agent and the corresponding performance bounds $\alpha_i, \beta_i, i \in \mathcal{V}_L$, are depicted. In addition, the evolution of the relative distances $\|x_{ij}\|$ between neighboring agents and the corresponding performance bounds γ_i are plotted in Figure 5.6. One can see that the performance bounds are satisfied at any time.

Secondly, we consider the dynamic scheduling strategy, but for the case that there is no limit on $|W|$. In this way, the true optimal schedule (i.e. considering all possible permutations of the waiting task set) can be obtained. Finally, the EDF policy is used to schedule the same set of tasks. The simulation results for the three different policies are summarized in TABLE 5.2. One can see that the best total reward is obtained when the dynamic scheduling strategy with no limit on $|W|$ is implemented. However, we note that this strategy will result in a computational complexity problem when $|W|$ is large. The dynamic scheduling strategy (with limit on $|W|$) can be seen as a compromise between the reward and the computational efficiency. When the EDF policy is implemented, the total reward is 120, which is the worst among the three policies. This makes sense since the EDF policy involves only the deadline of each task.

Table 5.2: Results for dynamic scheduling strategy (DSS), dynamic scheduling strategy with no limit on $|W|$ (DSS*), and EDF policy.

Strategy	Task completion order	Total reward
DSS	2-1-3-6-5-1-4-5-2-4-6	155
DSS*	4-1-2-3-6-5-1-2-5-6-4	160
EDF	2-3-6-5-1-4-5-1-2-6-4	120

5.5 Summary

We proposed a dynamic task scheduling and distributed control synthesis strategy for a leader-follower MAS whose goal was to complete a set of dynamically activated tasks. Each task is associated with a relative deadline and

can be completed in several QoSa levels. By utilizing ideas from prescribed performance control, we developed distributed control laws respectively for the leader and the follower agents such that the satisfaction of tasks under explicit deadline or time interval constraint is guaranteed.

Part III

Control and Coordination under Temporal Logic Specifications

Chapter 6

Symbolic Control of Continuous-Time Uncertain Nonlinear Systems

Discrete abstractions have become a standard approach to assist control synthesis under complex specifications, *e.g.,* linear temporal logic (LTL) specifications. To ensure the correctness of the solutions obtained from the abstract system for the original (infinite) system, it is important to establish an equivalence or inclusion relation between the abstract system and the original system [94]. This chapter is concerned with the construction of symbolic models for continuous-time uncertain nonlinear systems.

6.1 Introduction

In recent years, discrete abstractions have become one of the standard approaches for control synthesis in the context of complex dynamical systems and specifications [95]. It allows one to leverage computational tools developed for discrete-event systems [96]–[98] and games on automata [99], [100] to assist control synthesis for specifications difficult to enforce with conventional control design methods, such as LTL specifications [17]. Moreover, if the behaviors of the original system (referred to as the concrete system) and the abstract system (obtained by, *e.g.,* discretizing the state-space) can be formally related by an inclusion or equivalence relation, the synthesized controller is known to be correct by design [101].

For a long time, (bi)simulation relations were a central notion to deal with complexity reduction [102], [103]. It was later pointed out in [104] that this

kind of equivalence relation is often too strong. To this end, a new notion called approximate (bi)simulation, which only asks for the closeness of observed behaviors, was introduced in [94]. Based on the notion of incremental (input-to-state) stability [35], approximately bisimilar symbolic models were built and extended to various systems [105], [106]. However, incremental (input-to-state) stability is a strong property for dynamical control systems, which makes its applicability restrictive. In [36], the authors relax the incremental (input-to-state) stability requirement by only assuming Lipschitz continuous and incremental forward completeness, and an approximate alternating simulation relation is established by over-approximating the behavior of the concrete system. However, as recently pointed out in [107], this approach may result in a refinement complexity issue. To this end, a new simulation relation, called feedback refinement relation is proposed in [107].

Although continuous-time systems are extensively studied and various abstraction techniques are proposed in the existing literature, most techniques for the construction of symbolic models require time-space discretization of the continuous-time system, which constitute property satisfaction non-trivial since closeness of the observed behaviors between the concrete system and its abstraction is not guaranteed within neighboring discrete time instants. Recently, different approaches have been proposed in the literature to deal with this [108]–[110]. In [108], a disturbance simulation relation is introduced for incrementally input-to-state stable nonlinear systems. In [109], [110], symbolic control approaches are proposed for a class of sample-data nonlinear systems, where property satisfaction of the continuous-time systems is guaranteed by equipping the finite abstractions with certain robustness margins [109] or assume-guarantee contracts [110]. While almost all the results are providing behavioral relationships between a time discretized version of the original system and its symbolic model, in this chapter, we provide for the first time a behavioral relationship between the original continuous-time system and its symbolic model.

This chapter investigates the construction of symbolic models for continuous-time uncertain nonlinear systems. It improves upon most of the existing results in two aspects: 1) by not requiring time-space discretization of the concrete system and 2) by being applicable to more general uncertain nonlinear systems under input constraint. The main contributions are as follows.

(i) We propose a novel stability notion, called η-approximate controlled

globally practically stable with respect to a given set Ω. This is a property defined on the augmented system (composed of the concrete system and the abstract system) via an admissible control interface. We show that the abstract system can be constructed without time-space discretization. This is crucial for safety-critical applications, in which it is necessary that the trajectories of the concrete system and the abstract system are close enough at all time instants.

(ii) We define a notion of robust approximate (bi)simulation relation. It is shown that for an uncertain concrete system, the abstract system can be constructed such that the concrete system robustly approximately simulates the abstraction.

(iii) For the class of incrementally quadratic nonlinear systems, the systematic construction of the admissible control interfaces and robust approximately symbolic models under a bounded input set is provided.

The introduction of the control interface is inspired by the hierarchical control framework [111]–[114], in which an interface is built between a high dimensional concrete system and a simplified low dimensional abstraction of it. Both the concrete system and the abstract system are continuous (in state-space) in [111]–[114]. In contrast, in this chapter, we propose to build a control interface between the continuous-time concrete system and its discrete state-space abstraction. Moreover, In this chapter we consider a bounded input set (the input set considered in [111]–[114] is unbounded), which results in additional difficulty in constructing the interface.

6.2 Problem Formulation

Consider a continuous-time uncertain nonlinear system of the form

$$\Sigma : \begin{cases} \dot{x}_1(t) = f(t, x_1(t), u(t)) + w(t), \\ y_1(t) = h(x_1(t)), \end{cases} \tag{6.1}$$

where $x_1(t) \in \mathbb{R}^n, y_1(t) \in \mathbb{R}^l, u(t) \in U \subseteq \mathbb{R}^m, w(t) \in W \subset \mathbb{R}^n$ are the state, output, control input, and external disturbance at time t, respectively. The input and disturbance are constrained to sets U and W, respectively. We assume that $f : [0, \infty) \times \mathbb{R}^n \times U \to \mathbb{R}^n$ is piecewise continuous in t, continuous in x_1 and u, and the vector field f is such that for any input in U and any

initial condition $x_1(0) \in \mathbb{R}^n$, this differential equation has a unique solution. Throughout the chapter, we will refer to Σ as the concrete system, that is the system that we actually want to control.

Let $\mathcal{U}$ be the set of all measurable functions that take their values in U and are defined on $\mathbb{R}_{\geq 0}$. Similarly, one can define $\mathcal{W}$ as the set of all measurable functions that take their values in W and are defined on $\mathbb{R}_{\geq 0}$.

A curve $\xi : [0, \tau[\to \mathbb{R}^n$ is said to be a trajectory of Σ if there exists an input signal $u \in \mathcal{U}$ and a disturbance signal $w \in \mathcal{W}$ satisfying $\dot{\xi}(t) = f(t, \xi(t), u(t)) + w(t)$ for almost all $t \in [0, \tau[$. A curve $\zeta : [0, \tau[\to \mathbb{R}^l$ is said to be an output trajectory of Σ if $\zeta(t) = h(\xi(t))$ for almost all $t \in [0, \tau[$, where ξ is a trajectory of Σ. We use $\xi(\xi_0, u, w, t)$ to denote the trajectory point reached at time t under the input signal $u \in \mathcal{U}$ and the disturbance signal $w \in \mathcal{W}$ from initial state ξ_0.

The deterministic system is defined as

$$\Sigma_d : \begin{cases} \dot{x}_1(t) = f(t, x_1(t), u(t)), \\ y_1(t) = h(x_1(t)), \end{cases} \tag{6.2}$$

and $\xi(\xi_0, u, t) := \xi(\xi_0, u, 0, t)$ denotes the trajectory of the deterministic system (6.2). Note that (6.2) is the deterministic version of (6.1) with $w(t) \equiv \mathbf{0}$.

By a minor modification of the statement of Definitions 2.3 and 2.4, one can define forward complete (FC) and incrementally globally practically stable (δ-GPS) for uncertain systems.

Definition 6.1. *The uncertain system (6.1) is called FC if for every initial condition $x_0 \in \mathbb{R}^n$, every input signal $u \in \mathcal{U}$, and every disturbance signal $w \in \mathcal{W}$, the corresponding solution is defined for all $t \in [0, \infty)$.*

Definition 6.2. *The uncertain system (6.1) is called δ-GPS if it is FC and there exist a class $\mathcal{KL}$ function β, and a $\mathcal{K}_\infty$ function κ such that for any $t \in \mathbb{R}_{\geq 0}$, any $x_0, x_0' \in \mathbb{R}^n$, any disturbance signals $w, w' \in \mathcal{W}$, and any $u \in \mathcal{U}$,*

$$\|\xi(x_0, u, w, t) - \xi(x_0', u, w', t)\| \leq \beta(\|x_0 - x_0'\|, t) + \kappa(\|w - w'\|_\infty).$$

In this chapter, the problem under consideration is formulated as follows.

Problem 6.1. *For what kind of continuous-time uncertain nonlinear system (6.1), can one build a discrete state-space abstraction such that the original system and the abstract system are formally related by a simulation relation? In addition, how to build such an abstraction?*

6.3 Abstraction and Stability Notion

In this chapter, the abstraction technique developed in [105] is applied, in which the state-space $\mathbb{R}^n$ is approximated by the lattice

$$[\mathbb{R}^n]_\eta = \left\{ q \in \mathbb{R}^n | q_i = k_i \frac{2\eta}{\sqrt{n}}, k_i \in \mathbb{Z}, i = 1, \ldots, n \right\}, \tag{6.3}$$

where $\eta \in \mathbb{R}_{\geq 0}$ is a state-space discretization parameter. Define the associated quantizer $Q_\eta : \mathbb{R}^n \to [\mathbb{R}^n]_\eta$ as $Q_\eta(x) = q$ if and only if $|x_i - q_i| \leq \eta/\sqrt{n}, \forall i = 1, \ldots n$. Then, one has $\|x - Q_\eta(x)\| \leq \eta, \forall x \in \mathbb{R}^n$.

The abstract system is obtained by applying the state abstraction (6.3) to the deterministic system (6.2), which is given by

$$\Sigma' : \begin{cases} x_2(t) = Q_\eta(\xi(Q_\eta(x_1(0)), v, t)), \\ y_2(t) = h(x_2(t)), \end{cases} \tag{6.4}$$

where $x_2(t) \in [\mathbb{R}^n]_\eta, y_2(t) \in \mathbb{R}^l$, and $v(t) \in U'(t)$ represent the state, output and control input of the abstract system, respectively. Note that the state variable $x_2(t)$ is neither continuous nor differentiable due to the state-space discretization. In addition, the map $U' : \mathbb{R}_{\geq 0} \to 2^{\mathbb{R}^m}$ denotes the possibly time-varying input constraint for the abstract system, which is a design parameter that will be specified later.

Let $R_{U'} := \{U'(t) : t \in \mathbb{R}_{\geq 0}\}$ be the range of the set-valued function U'. Then, we define

$$\mathcal{U}' = R_{U'}^{[0,\infty)} \tag{6.5}$$

as the set of all functions of time from interval $[0, \infty)$, such that the value of the function at a particular time instant t, is an element of $U'(t)$.

A (hybrid) curve $\xi' : [0, \infty) \to [\mathbb{R}^n]_\eta$ is said to be a trajectory of Σ' if there exists $v \in \mathcal{U}'$ satisfying:

$$\xi'(t) = Q_\eta(\xi(t)), \forall t \in [0, \infty),$$

where $\dot{\xi}(t) = f(t, \xi(t), v(t))$ and $\xi(0) = \xi'(0)$. A curve $\zeta' : [0, \infty) \to \mathbb{R}^l$ is said to be an output trajectory of Σ' if $\zeta'(t) = h(\xi'(t))$, for almost all $t \in [0, \infty)$, where ξ' is a trajectory of Σ'. With a little abuse of notation, we use $\xi'(\xi'_0, v, t)$ to denote the trajectory point of Σ' reached at time t under the input signal $v \in \mathcal{U}'$ from an initial condition $\xi'_0 \in [\mathbb{R}^n]_\eta$.

The control input $u(t)$ of the concrete system (6.1) will be synthesized hierarchically via the abstract system (6.4) with a control interface $u_v : \mathbb{R}_{\geq 0} \times \mathbb{R}^m \times \mathbb{R}^n \times [\mathbb{R}^n]_\eta \to \mathbb{R}^m$, which is given by

$$u(t) = u_v(t, v(t), x_1(t), x_2(t)). \tag{6.6}$$

Then, the augmented control system $\hat{\Sigma}$ is defined as

$$\hat{\Sigma} : \begin{cases} \dot{x}_1(t) = f(t, x_1(t), u_v(t, v(t), x_1(t), x_2(t))) + w(t), \\ x_2(t) = Q_\eta(\xi(Q_\eta(x_1(0)), v, t)), \\ y_1(t) = h(x_1(t)), \\ y_2(t) = h(x_2(t)), \end{cases} \tag{6.7}$$

and we denote by

$$\begin{aligned} x(t) &= (x_1(t), x_2(t)) \in \hat{X} := \{(z, z') : z \in \mathbb{R}^n, z' \in [\mathbb{R}^n]_\eta\}, \\ y(t) &= (y_1(t), y_2(t)) \in \hat{Y} := \{(z, z') : z \in \mathbb{R}^l, z' \in \mathbb{R}^l\}, \\ \hat{u}(t) &= (u_v(t), v(t)) \in \hat{U}(t) := \{(z, z') : z \in U, z' \in U'(t)\}, \\ \hat{w}(t) &= (w(t), \mathbf{0}) \in \hat{W} := \{(z, z') : z \in W, z' = \mathbf{0}\}. \end{aligned} \tag{6.8}$$

Note that $x_2(0) = Q_\eta(x_1(0))$, and then one has $\|x_1(0) - x_2(0)\| = \|x_1(0) - Q_\eta(x_1(0))\| \leq \eta, \forall x_1(0) \in \mathbb{R}^n$. Therefore, one can define

$$\hat{X}_0 := \{(x_1, x_2) \in \hat{X} | \|x_1 - x_2\| \leq \eta\} \tag{6.9}$$

as the set of initial states for $\hat{\Sigma}$.

To guarantee that the synthesized controller $u(t)$ is applicable to the concrete system (6.1), it is necessary that $u(t) = u_v(t, v(t), x_1(t), x_2(t)) \in U, \forall t \in [0, \infty)$. Therefore, we propose the following definition.

Definition 6.3. *The control interface $u_v : \mathbb{R}_{\geq 0} \times \mathbb{R}^m \times \mathbb{R}^n \times [\mathbb{R}^n]_\eta \to \mathbb{R}^m$ is called* admissible *if there exists an input map $U' : \mathbb{R}_{\geq 0} \to 2^{\mathbb{R}^m}$ that satisfies*

i) $U'(t) \neq \emptyset, \forall t \in [0, \infty)$, and

ii) $u(t) = u_v(t, v(t), \xi(\xi_0, u_v, w, t), \xi'(\xi'_0, v, t)) \in U, \forall t \in [0, \infty), \forall (\xi_0, \xi'_0) \in \hat{X}_0, \forall v \in \mathcal{U}', \forall w \in \mathcal{W}$.

In this case, the input map U' is called admissible *to u_v.*

The trajectory of $\hat{\Sigma}$ will be denoted by $\hat{\xi}$. Moreover, we use $\hat{\xi}(\hat{\xi}_0, \hat{u}, \hat{w}, t)$ to denote the trajectory point of $\hat{\Sigma}$ reached at time t under the input and disturbance signals $\hat{u}, \hat{w}$ from initial state $\hat{\xi}_0$, where $\hat{u}, \hat{w}$ are defined in (6.8). The diagonal set $\Omega \subseteq \mathbb{R}^{2n}$ is defined as:

$$\Omega = \{z \in \mathbb{R}^{2n} | \exists x \in \mathbb{R}^n : z = (x, x)\}. \tag{6.10}$$

Then, we introduce the following definition which is inspired by [115].

Definition 6.4. *The augmented control system $\hat{\Sigma}$ is called* η-approximate controlled globally practically stable with respect to the set Ω (η-C-Ω-GPS) *if it is FC and there exist an admissible control interface u_v, a $\mathcal{KL}$ function β, and $\mathcal{K}_\infty$ functions γ_1, γ_2 such that $\forall t \in [0, \infty)$, $\forall \hat{\xi}_0 \in \hat{X}_0, \forall v(t) \in U'(t), \forall \hat{w}(t) \in \hat{W}$,*

$$\mathrm{d}(\hat{\xi}(\hat{\xi}_0, \hat{u}, \hat{w}, t), \Omega) \leq \beta(\mathrm{d}(\hat{\xi}_0, \Omega), t) + \gamma_1(\eta) + \gamma_2(\|\hat{w}\|_\infty),$$

where $\hat{u} = (u_v, v)$ and u_v is given by the admissible control interface. Moreover, u_v is called an interface *for $\hat{\Sigma}$, associated to the* η-C-Ω-GPS *property.*

Remark 6.1. *According to Definitions 6.3-6.4, a general idea on determining the admissible control interface and the associated input map U' can be provided as follows: firstly, ignore the input constraint for the concrete system (6.1) by assuming that $U = \mathbb{R}^m$ (in this way, any control interface that maps to $\mathbb{R}^m$ is admissible), and find one or several control interfaces u_v such that $\hat{\Sigma}$ is* η-C-Ω-GPS. *Secondly, taking the real input set U into account, refine the control interfaces obtained in the previous step in a way that the admissible ones and the associated input maps are kept.*

Remark 6.2. *We note that the notion of η-C-Ω-GPS defined in Definition 6.4 is essentially different from the notion of δ-ISS given in Definition 2.5 or δ-FC given in Definition 2.6. Both δ-ISS and δ-FC are properties defined on the concrete system Σ while η-C-Ω-GPS is a property defined on the augmented control system $\hat{\Sigma}$. Moreover, for concrete systems that are not δ-ISS or δ-FC, the η-C-Ω-GPS property can still hold for the corresponding augmented systems (as shown later in Section 6.5).*

Proposition 6.1. *Consider the augmented control system $\hat{\Sigma}$. If $\hat{\Sigma}$ is* η-C-Ω-GPS, *then one has that $\forall t \in [0, \infty)$, $\forall (x_0, x_0') \in \hat{X}_0, \forall w \in \mathcal{W}$,*

$$\|\xi(x_0, u_v, w, t) - \xi'(x_0', v, t)\| \leq \beta(\|x_0 - x_0'\|, t) + \gamma_1(\eta) + \gamma_2(\|w\|_\infty),$$

where $\beta, \gamma_1, \gamma_2$ are defined in Definition 6.4.

Proof. The proof follows from the fact that $\mathrm{d}(x, \Omega) = \|x_1 - x_2\|, \forall x_1, x_2$, where $x = (x_1, x_2)$ [115]. □

Remark 6.3. *Another difference between δ-FC in [36] and η-C-Ω-GPS is that the β function belongs to class $\mathcal{K}_\infty$ in the Definition of δ-FC while class $\mathcal{KL}$ in Definition 6.4. In [36], the state error $\|\xi(x_0, u, t) - \xi(x_0', u', t)\|$ is not bounded by the initial state error $\|x_0 - x_0'\|$ because $\beta(\|x_0 - x_0'\|, t)$ can go to infinity as time goes to infinity. This causes the refinement issues. However, it is shown in Definition 6.4 and Proposition 6.1 that by properly designing the admissible control interface u_v, one can upper bound the state error $\|\xi(x_0, u_v, t) - \xi(x_0', v, t)\|$ by a $\mathcal{KL}$ function $\beta(\|x_0 - x_0'\|, t)$ (which vanishes as the time goes to infinity) and a constant $\gamma_1(\eta)$ (we consider the deterministic system here for comparison, i.e., $w(t) \equiv \mathbf{0}$). Therefore, our approach has no refinement issues.*

In the following, the Lyapunov function characterization of the stability notion η-C-Ω-GPS is proposed, which is motivated by [111].

Definition 6.5. *Consider the augmented control system $\hat{\Sigma}$, a smooth function $V : [0, \infty) \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ and a control interface u_v. Function V is called a* η-C-Ω-GPS Lyapunov function *for $\hat{\Sigma}$ and u_v is the associated control interface if there exist $\mathcal{K}_\infty$ functions $\underline{\alpha}, \bar{\alpha}, \sigma_1, \sigma_2$, and a constant $\mu > 0$ such that*

i) $\forall x, x' \in \mathbb{R}^n$,

$$\underline{\alpha}(\|x(t) - x'(t)\|) \leq V(t, x(t), x'(t)) \leq \bar{\alpha}(\|x(t) - x'(t)\|), \tag{6.11}$$

ii) $\forall x, x' \in \mathbb{R}^n, \forall v(t) \in U'(t)$, *and* $\forall w \in \mathcal{W}$,

$$\begin{aligned} &\frac{\partial V}{\partial x}\left\{f(t, x(t), u_v(t, v(t), x(t), Q_\eta(x'(t)))) + w(t)\right\} \\ &\qquad + \frac{\partial V}{\partial x'} f(t, x'(t), v(t)) + \frac{\partial V}{\partial t} \\ &\leq -\mu V(t, x(t), x'(t)) + \sigma_1(\eta) + \sigma_2(\|\omega\|_\infty). \end{aligned} \tag{6.12}$$

Then, we can get the following theorem.

Theorem 6.1. *Consider the augmented control system $\hat{\Sigma}$ and the diagonal set Ω. If i) $\hat{\Sigma}$ is FC, ii) there exists a* η-C-Ω-GPS Lyapunov function *for $\hat{\Sigma}$ and with u_v being the associated control interface, and iii) u_v is* admissible*, then, $\hat{\Sigma}$ is* η-C-Ω-GPS *and u_v is the* interface *for $\hat{\Sigma}$, associated to the* η-C-Ω-GPS *property.*

Proof. Let $\xi(t) := \xi(Q_\eta(x_1(0)), v, t)$ for short. Then, one has $\dot{\xi}(t) = f(t, \xi(t), v(t))$ and $\xi(0) = Q_\eta(x_1(0))$. In addition, one can get from (6.7) that

$$x_2(t) = Q_\eta(\xi(t)), \forall t \in [0, \infty). \tag{6.13}$$

Let V be the η-C-Ω-GPS Lyapunov function for $\hat{\Sigma}$ and u_v the associated control interface. Then, one has (6.12) holds and thus

$$\begin{aligned}
&\frac{\partial V}{\partial x_1}\{f(t, x_1(t), u_v(t, v(t), x_1(t), Q_\eta(\xi(t)))) + w(t)\} \\
&\qquad + \frac{\partial V}{\partial \xi} f(t, \xi(t), v(t)) + \frac{\partial V}{\partial t} \\
\leq & -\mu V(t, x_1(t), \xi(t)) + \sigma_1(\eta) + \sigma_2(\|\omega\|_\infty) \\
\leq & -\frac{\mu}{2} V(t, x_1(t), \xi(t))
\end{aligned}$$

for all $\|x_1(t) - \xi(t)\| \geq \underline{\alpha}^{-1}\left(2\sigma_1(\eta) + 2\sigma_2(\|\omega\|_\infty)\right)/\mu)$. According to Lemma 2.3, one can further have

$$\begin{aligned}
\|x_1(t) - \xi(t)\| \leq & \underline{\alpha}^{-1}(e^{-\frac{\mu}{2}t}\bar{\alpha}(\|x_1(0) - \xi(0)\|)) \\
& + \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}\left(2\sigma_1(\eta) + 2\sigma_2(\|\omega\|_\infty)\right)/\mu))).
\end{aligned}$$

Moreover, one has from (6.13) that $\|x_2(t) - \xi(t)\| = \|Q_\eta(\xi(t)) - \xi(t)\| \leq \eta$. Thus,

$$\begin{aligned}
\|x_1(t) - x_2(t)\| \leq & \|x_1(t) - \xi(t)\| + \|\xi(t) - x_2(t)\| \\
\leq & \underline{\alpha}^{-1}(e^{-\frac{\mu}{2}t}\bar{\alpha}(\|x_1(0) - x_2(0)\|)) \\
& + \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}\left(2\sigma_1(\eta) + 2\sigma_2(\|\omega\|_\infty)\right)/\mu))) + \eta \\
\leq & \underline{\alpha}^{-1}(e^{-\frac{\mu}{2}t}\bar{\alpha}(\|x_1(0) - x_2(0)\|)) \\
& + \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}\left(4\sigma_1(\eta)\right))) + \eta \\
& + \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}\left(4\sigma_2(\|\omega\|_\infty)\right)/\mu))).
\end{aligned}$$

Combining the fact that u_v is admissible, one can conclude that $\hat{\Sigma}$ is η-C-Ω-GPS and u_v is the interface for $\hat{\Sigma}$, associated to the η-C-Ω-GPS property. □

6.4 Robust Approximate Simulation Relation

In this section, the construction of symbolic models for the concrete system (6.1) is considered. Firstly, the notion of robust approximate (bi)simulation relation is proposed.

Definition 6.6. *Consider the concrete system Σ in (6.1) and the abstract system Σ' in (6.4). Let $\varepsilon > 0$ be a given precision and $\tilde{\varepsilon} \geq 0$. We say that Σ* robustly approximately simulates *Σ' with parameters $(\varepsilon, \tilde{\varepsilon})$, denoted by $\Sigma' \preceq_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma$, if*

i) $\forall x_0' \in [\mathbb{R}^n]_\eta, \exists x_0 \in \mathbb{R}^n$ *such that* $(x_0, x_0') \in \hat{X}_0$, *and*

ii) $\forall (x_0, x_0') \in \hat{X}_0, \forall v \in \mathcal{U}', \exists u \in \mathcal{U}$ *such that* $\forall t \geq 0$,

$$\|h(\xi(x_0, u, w, t)) - h(\xi'(x_0', v, t))\| \leq \varepsilon, \forall w : \|w\|_\infty < \tilde{\varepsilon},$$

where $\mathcal{U}'$ and $\hat{X}_0$ are defined in (6.5) and (6.9), respectively.

The systems Σ and Σ' are said to be robust approximate bisimilar with parameters $(\varepsilon, \tilde{\varepsilon})$, denoted by $\Sigma \cong_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma'$, *if $\Sigma \preceq_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma'$ and $\Sigma' \preceq_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma$.*

Remark 6.4. *We note that the robust approximate (bi)simulation relation defined in Definition 6.6 resembles the notion of disturbance (bi)simulation relation given in [108], Definition 2. The difference lies in that our relation is defined for continuous-time systems while in [108], the relation is defined for discrete-time systems.*

Before proceeding, we need the following additional assumption.

Assumption 6.1. *The output function $h : \mathbb{R}^n \to \mathbb{R}^l$ is globally Lipschitz continuous with Lipschitz constant ρ on the set X_ε. That is,*

$$\|h(x_1) - h(x_2)\| \leq \rho \|x_1 - x_2\|, \forall (x_1, x_2) \in X_\varepsilon,$$

where $X_\varepsilon := \{(x_1, x_2) : \|x_1 - x_2\| \leq \underline{\alpha}^{-1}(\bar{\alpha}(\varepsilon)) + \underline{\alpha}^{-1}\big((\sigma_1(\varepsilon) + \max_{w \in \mathcal{W}}\{\sigma_2(\|w\|_\infty)\})/\mu\big) + \varepsilon\}$, $\underline{\alpha}, \bar{\alpha}, \sigma_1, \sigma_2, \mu$ are defined in Definition 6.5, $\mathcal{W}$ is the set of disturbance signals, and ε is the desired precision.

Assumption 6.1 is not conservative since it only requires Lipschitz continuity within a neighborhood of x_1, the radius of which is determined by the desired precision ε. Note that the Lipschitz constant ρ is independent of ε. Then, we can get the following result.

Theorem 6.2. *Consider the concrete system Σ in (6.1) and the abstract system Σ' in (6.4). Let $\varepsilon > 0$ be a desired precision. Suppose Assumption 6.1 holds. Assume that there exists a η-C-Ω-GPS* Lyapunov function *V for the augmented*

system $\hat{\Sigma}$ *and let* u_v *be the associated control interface that is* admissible. *If furthermore, one has that* $\|w\|_\infty < \tilde{\varepsilon} := \sigma_2^{-1}(\mu\underline{\alpha}(\bar{\alpha}^{-1}(\underline{\alpha}(\varepsilon/\rho)))/4), \forall w \in \mathcal{W}$; *then,* $\Sigma' \preceq_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma$ *if*

$$\begin{aligned} &\underline{\alpha}^{-1}(\bar{\alpha}(\eta)) + \eta + \underline{\alpha}^{-1}\left(\bar{\alpha}\left(\underline{\alpha}^{-1}\left(\frac{4\sigma_1(\eta)}{\mu}\right)\right)\right) \\ &\quad < \frac{\varepsilon}{\rho} - \underline{\alpha}^{-1}\left(\bar{\alpha}\left(\underline{\alpha}^{-1}\left(\frac{4\sigma_2(\|w\|_\infty)}{\mu}\right)\right)\right). \end{aligned} \tag{6.14}$$

Proof. Item *i)* of Definition 6.6 holds trivially. In the following, we will prove item *ii)*.

Let there be given $(x_0, x_0') \in \hat{X}_0$ and an input signal $v \in \mathcal{U}'$. Since the control interface u_v is admissible, then one has $u(t) = u_v(t, v(t), \xi(x_0, u_v, w, t), \xi'(x_0', v, t)) \in U, \forall t \in [0, \infty)$. Thus, $u \in \mathcal{U}$. Let $q(t) = \xi(x_0', v, t))$. Then, one has $x_2(t) = \xi'(x_0', v, t) = Q_\eta(q(t)), \forall t \in [0, \infty)$. Let also $x_1(t) = \xi(x_0, u_v, w, t))$, where u_v is the admissible control interface. To prove item *ii)* of Definition 6.6, it is sufficient to prove that $\|h(x_1(t)) - h(x_2(t)\| \leq \varepsilon, \forall t \in [0, \infty)$.

Since V is a η-C-Ω-GPS Lyapunov function for $\hat{\Sigma}$, then (6.12) of Definition 6.5 holds. One has from Theorem 6.1 that

$$\begin{aligned} \|x_1(t) - q(t)\| \leq &\underline{\alpha}^{-1}(e^{-\frac{\mu}{2}t}\bar{\alpha}(\|x_1(0) - q(0)\|)) \\ &+ \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}(4\sigma_1(\eta)/\mu))) \\ &+ \underline{\alpha}^{-1}(\bar{\alpha}(\underline{\alpha}^{-1}(4\sigma_2(\|\omega\|_\infty))/\mu))). \end{aligned}$$

In addition, $\|x_1(0) - q(0)\| = \|\xi(0) - \xi'(0)\| = \|x_0 - x_0'\| \leq \eta$. Using (6.14), one can further get

$$\begin{aligned} \|x_1(t) - x_2(t)\| &\leq \|x_1(t) - q(t)\| + \|q(t) - x_2(t)\| \\ &= \|x_1(t) - q(t)\| + \|q(t) - Q_\eta(q(t))\| \\ &\leq \varepsilon/\rho, \end{aligned}$$

and thus $\|h(x_1(t)) - h(x_2(t))\| \leq \rho\|x_1(t) - x_2(t)\| \leq \varepsilon$. Item *ii)* of Definition 6.6 holds and thus $\Sigma' \preceq_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma$. □

Corollary 6.1. *Consider the concrete system* Σ *in (6.1), the abstract system* Σ' *in (6.4), and the desired precision* $\varepsilon > 0$. *Assume that the concrete system* Σ *is* δ*-GPS. Let* $u_v(t, v(t), x_1(t), x_2(t)) = v(t), \forall t \geq 0$ *be the admissible control*

interface with $U'(t) \equiv U$. Suppose Assumption 6.1 holds. If furthermore, one has that $\|w\|_\infty < \tilde{\varepsilon} := \kappa^{-1}(\varepsilon/\rho), \forall w \in \mathcal{W}$; then, $\Sigma' \cong_{\mathcal{S}}^{(\varepsilon,\tilde{\varepsilon})} \Sigma$ if

$$\beta(\eta, 0) + \eta < \frac{\varepsilon}{\rho} - \kappa(\|w\|_\infty),$$

where β, κ are given in Definition 6.2.

Remark 6.5. *The construction of symbolic models and the implementation of the admissible control interface rely on the computation of the state-space abstraction and the abstract controller. For different systems, computational tools have been developed for this purpose, e.g., PESSOA [116], SCOTS [117], and LTLCon [118].*

Remark 6.6. *One key step for the construction of symbolic models is to find an admissible control interface. From Definition 6.3, one can see that for a control interface u_v to be admissible, the key factor is to find an input map U' admissible to u_v. When the input set for the concrete system is unbounded, i.e., $U = \mathbb{R}^m$, any control interface that maps to $\mathbb{R}^m$ is admissible. However, in practical applications, input saturations are common constraints. We note that when the input set for the concrete system is bounded, it is not always possible to find an admissible control interface. The good news is that, for a certain class of incrementally quadratic nonlinear systems, we show in the next section that it is possible to construct an admissible control interface u_v, such that Σ robustly approximately simulates Σ'.*

6.5 Incrementally Quadratic Nonlinear Systems

In this section, we consider a class of perturbed incrementally quadratic nonlinear systems [119], for which the systematic construction of the admissible control interface and robust approximately symbolic models is possible. This kind of nonlinear systems are very useful and include many commonly encountered nonlinearities, such as the global Lipschitz nonlinearity, as special cases.

Consider the nonlinear time-varying system described by

$$\Sigma_1 : \begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + Ep(t, C_q x + D_q p) + w(t), \\ y(t) = Cx(t). \end{cases} \tag{6.15}$$

where $x \in \mathbb{R}^n, y \in \mathbb{R}^l$, $u \in U \subseteq \mathbb{R}^m$, and $w \in W \subset \mathbb{R}^n$ are the state, output, control input, and external disturbance, respectively, $p : \mathbb{R}_{\geq 0} \times \mathbb{R}^{l_p} \to \mathbb{R}^{l_e}$ represents the known continuous nonlinearity of the system, and A, B, C, E, C_q, D_q are constants matrices of appropriate dimensions.

Definition 6.7. *[120] Given a function $p : \mathbb{R}_{\geq 0} \times \mathbb{R}^{l_p} \to \mathbb{R}^{l_e}$, a symmetric matrix $M \in \mathbb{R}^{(l_p+l_e)\times(l_p+l_e)}$ is called an incremental multiplier matrix for p if it satisfies the following incremental quadratic constraint for any $q_1, q_2 \in \mathbb{R}^{l_p}$:*

$$\begin{bmatrix} q_2 - q_1 \\ p(t, q_2) - p(t, q_1) \end{bmatrix}^T M \begin{bmatrix} q_2 - q_1 \\ p(t, q_2) - p(t, q_1) \end{bmatrix} \geq 0. \tag{6.16}$$

Remark 6.7. *The incremental quadratic constraint (6.16) includes a broad class of nonlinearities as special cases. For instance, the globally Lipschitz condition, the sector bounded nonlinearity, and the positive real nonlinearity $p^T S q \geq 0$ for some symmetric, invertible matrix S. Some other nonlinearities that can be expressed using the δ-QC were discussed in [119], [120], such as the case when the Jacobian of p with respect to q is confined in a polytope or a cone.*

Assumption 6.2. *There exist matrices $P = P^T \succ 0, L$ and a scalar $\alpha > 0$ such that the following matrix inequality*

$$\begin{aligned} &\begin{bmatrix} P(A + BL) + (A + BL)^T P + 2\alpha P & PE \\ E^T P & 0 \end{bmatrix} \\ &\qquad + \begin{bmatrix} C_q & D_q \\ 0 & I \end{bmatrix}^T M \begin{bmatrix} C_q & D_q \\ 0 & I \end{bmatrix} \leq 0 \end{aligned} \tag{6.17}$$

is satisfied, where $M = M^T$ is an incremental multiplier matrix for function p.

Remark 6.8. *The matrix inequality (6.17) is not a linear matrix inequality. Hence, one can not solve for P, L reliably via, e.g., the interior point method algorithms. However, we note that parameterization methods, such as block diagonal parameterization [120] can be utilized to transform (6.17) into Ricatti equations and/or linear matrix inequalities under certain conditions. Moreover, we note that several necessary and/or sufficient conditions have been provided in [119], [120] to guarantee the existence of solutions to (6.17).*

The abstract system obtained by applying the state-space discretization (6.3) is given by

$$\Sigma_1' : \begin{cases} \xi(t) = Q_\eta(\hat{x}(Q_\eta(x(0)), v, t)), \\ \zeta(t) = C\xi(t), \end{cases} \tag{6.18}$$

where $\dot{\hat{x}}(t) = A\hat{x} + Bv(t) + Ep(t, C_q\hat{x} + D_qp)$ and $v \in U'(t)$.

According to Remark 6.1, we first ignore the input constraint for the concrete system (6.15) by assuming that $U = \mathbb{R}^m$. The control interface $u_v : \mathbb{R}_{\geq 0} \times 2^{\mathbb{R}^m} \times \mathbb{R}^n \times [\mathbb{R}^n]_\eta \to \mathbb{R}^m$ is then designed as

$$u_v(t, v(t), x(t), \xi(t)) = v(t) + L(x(t) - \xi(t)), \tag{6.19}$$

where L is the solution of (6.17). One can verify that u_v is admissible by letting $U'(t) = \mathbb{R}^m, \forall t \in [0, \infty)$. Then, we get the following result.

Theorem 6.3. *Consider the concrete system (6.15) with the input set $U = \mathbb{R}^m$ and the abstract system (6.18). The input $u(t)$ of (6.15) is synthesized by the control interface (6.19). Suppose that Assumption 6.2 holds and the disturbance set W satisfies $\|w\|_\infty < \tilde{\varepsilon} := \alpha\varepsilon\sqrt{\lambda_{\min}(P)}/(2\|c\|\sqrt{\lambda_{\max}(P)}), \forall w \in \mathcal{W}$; then, $\Sigma_1' \preceq_S^{(\varepsilon,\tilde{\varepsilon})} \Sigma_1$ if the state-space discretization parameter η satisfies*

$$\eta \leq \left(\frac{\varepsilon}{\|C\|} - \frac{2\sqrt{\lambda_{\max}(P)}\|w\|_\infty}{\alpha\sqrt{\lambda_{\min}(P)}}\right) \frac{\alpha\sqrt{\lambda_{\min}(P)}}{\alpha\sqrt{\lambda_{\min}(P)} + \sqrt{\alpha^2\lambda_{\max}(P) + 2\|\hat{L}\|}},$$

where $\hat{L} = L^T B^T PBL$ and P, L, α are the solution to (6.17).

Proof. Let $\hat{x}(t) = \hat{x}(Q_\eta(x(0)), v, t)$ and $e(t) = \xi(t) - \hat{x}(t)$, then one has $\|e(t)\| \leq \eta, \forall t$. Define $\delta(t) = x(t) - \hat{x}(t)$. Then, from (6.15) and (6.18) one has

$$\begin{aligned}\dot{\delta}(t) =& A\delta(t) + BL(\delta(t) + e(t)) \\ &+ E(p(t, C_qx + D_qp) - p(t, C_q\hat{x} + D_qp)) + w(t) \\ =& A_c\delta(t) + BLe(t) + E\Phi_p(t, x, \hat{x}) + w(t),\end{aligned}$$

where $A_c = A + BL$ and

$$\Phi_p(t, x, \hat{x}) = p(t, C_qx + D_qp) - p(t, C_q\hat{x} + D_qp).$$

Post and pre multiplying both sides of inequality (6.17) by $(\delta(t), \Phi_p(t, x, \hat{x}))$ and its transpose and using condition (6.16) we obtain

$$\delta^T P\dot{\delta} \leq -\alpha\delta^T P\delta + \delta^T PBLe + \delta^T Pw.$$

Consider the following Lyapunov function candidate

$$V(t, x, \hat{x}) = (x - \hat{x})^T P(x - \hat{x}).$$

Then, one has $\lambda_{\min}(P)\|x-\hat{x}\|^2 \leq V(t,x,\hat{x}) \leq \lambda_{\max}(P)\|x-\hat{x}\|^2$. Taking the derivative of V on t, one has

$$\begin{aligned}\dot{V}(t,x,\hat{x}) &= 2\delta^T P\dot{\delta}\\ &\leq -2\alpha\delta^T P\delta + 2\delta^T PBLe + 2\delta^T Pw\\ &\leq -\alpha V(t,x,\hat{x}) + \frac{2}{\alpha}\|\hat{L}\|\|e\|^2 + \frac{2}{\alpha}\|P\|\|w\|^2\\ &\leq -\alpha V(t,x,\hat{x}) + \frac{2}{\alpha}\|\hat{L}\|\eta^2 + \frac{2}{\alpha}\|P\|\|w\|^2.\end{aligned} \tag{6.20}$$

Therefore, $V(t,x,\hat{x})$ is a valid η-C-Ω-GPS Lyapunov function for $\hat{\Sigma}_1 := (\Sigma_1, \Sigma_1')$, where $\underline{\alpha}(x) = \lambda_{\min}(P)x^2$, $\bar{\alpha}(x) = \lambda_{\max}(P)x^2$, $\sigma_1(\eta) = 2\|\hat{L}\|\eta^2/\alpha$ and $\sigma_2(\|w\|_\infty) = 2\|P\|\|w\|_\infty^2/\alpha$. In addition, one can verify that Assumption 6.1 holds with $\rho = \|C\|$. Then, the conclusion follows from Theorem 6.2. □

Next, we will show how to find an input map U' admissible to u_v when the real input set U is considered.

From Theorem 6.3, we have (6.20) holds. Then, using the comparison principle, we can further get

$$\begin{aligned}&V(t,x(t),\hat{x}(t))\\ \leq& e^{-\alpha t}V(t,x(0),\hat{x}(0)) + \frac{2\|\hat{L}\|\eta^2 + 2\|P\|\|w\|^2}{\alpha^2}(1-e^{-\alpha t})\\ \leq& \lambda_{\max}(P)e^{-\alpha t}\|x(0)-\hat{x}(0)\|^2 + \frac{2\|\hat{L}\|\eta^2 + 2\|P\|\|w\|^2}{\alpha^2}\\ \leq& \lambda_{\max}(P)\eta^2 + \frac{2\|\hat{L}\|\eta^2 + 2\|P\|\|w\|^2}{\alpha^2}.\end{aligned}$$

Then, one can further have

$$\|x(t)-\hat{x}(t)\| \leq \sqrt{\frac{V(t,x(t),\xi(t))}{\lambda_{\min}(P)}} \leq K_1\eta + K_2\bar{w},$$

where $K_1 = \sqrt{\lambda_{\max}(P)/\lambda_{\min}(P) + 2\|\hat{L}\|/(\alpha^2\lambda_{\min}(P))}$,
$K_2 = \sqrt{2\lambda_{\max}(P)/(\alpha^2\lambda_{\min}(P))}$, $\bar{w} = \max_{w\in\mathcal{W}}\{\|w\|_\infty\}$, and

$$\|x(t)-\xi(t)\| \leq \|x(t)-\hat{x}(t)\| + \|\hat{x}(t)-\xi(t)\| \leq (K_1+1)\eta + K_2\bar{w}.$$

Define $e_u(t) = u(t) - v(t)$. Then, one has

$$\|e_u(t)\| = \|L(x(t) - \xi(t))\| \leq \|L\|((K_1 + 1)\eta + K_2\bar{w}). \tag{6.21}$$

From (6.21), one can see that the norm of the relative error between $u(t)$ and $v(t)$, *i.e.,* $\|e_u(t)\|$ is upper bounded, and the radius of the upper bound is determined by η and $\bar{w}$(due to the special form of control interface that was designed in (6.19)). Let

$$\tilde{U} = \big\{z \in U | \mathrm{dist}(z, F_r(U)) < \|L\|((K_1 + 1)\eta + K_2\bar{w})\big\}, \tag{6.22}$$

be the set of points in U, whose distance to the boundary of U is less than $\|L\|((K_1 + 1)\eta + K_2\bar{w})$. Then, by choosing

$$U'(t) = U \setminus \tilde{U}, \forall t \in [0, \infty),$$

one can guarantee that $u(t) \in U, \forall v(t) \in U'(t), \forall t \in [0, \infty)$. Moreover, we note that when Σ_1 is deterministic, *i.e.,* $w(t) \equiv 0$, one can always find $U'(t) \neq \emptyset, \forall t \in [0, \infty)$ for all $\mathrm{int(U)} \neq \emptyset$ since $U'(t) \to U$ when $\eta \to 0$.

Note that when Σ_1 is deterministic, *i.e.,* $w(t) \equiv 0$, one can always find by letting η be small enough since $U'(t) \to U$ when $\eta \to 0$. That is to say, the control interface u_v given by (6.19) is admissible for any $U, \mathrm{int(U)} \neq \emptyset$.

6.6 Examples

In this section, two simulation examples are provided to validate the effectiveness of the theoretical results.

6.6.1 Example 1

Consider the (undisturbed) time-varying nonlinear system Σ given by

$$\Sigma : \begin{cases} \dot{x}_1(t) = Ax_1(t) + \dfrac{1}{t+1}\sin(x_1(t)) + u(t), \\ y_1(t) = x_1(t), \end{cases} \tag{6.23}$$

where $x_1, y_1, u \in \mathbb{R}^2$, $A = [0.15, 0; 0, 0.05]$ is a constant matrix and the sinusoidal function $\sin(\cdot)$ is defined element-wise. One can verify that (6.23) is not δ-ISS.

Applying the state abstraction (6.3), then the abstract system can be written as

$$\Sigma' : \begin{cases} x_2(t) = Q_\eta(\varphi(Q_\eta(x_1(0)), v, t)), \\ y_2(t) = x_2(t), \end{cases} \tag{6.24}$$

where $\dot{\varphi}(t) = A\varphi(t) + \sin(\varphi(t))/(t+1) + v(t)$ and $v(t) \in U'(t)$. Since $U = \mathbb{R}^2$, one can choose the input map U' as $U'(t) = \mathbb{R}^2, \forall t \geq 0$, which is admissible to any u_v.

The control interface $u_v : \mathbb{R}_{\geq 0} \times \mathbb{R}^n \times \mathbb{R}^n \times [\mathbb{R}^n]_\eta \to \mathbb{R}^n$ is designed as

$$u_v(t, v(t), x_1(t), x_2(t)) = v(t) + P^{-1}R(x_1(t) - x_2(t)), \tag{6.25}$$

where $P = I_2, R = -5.4I_2$ are the solution to the following linear matrix inequality

$$\begin{bmatrix} A^TP + PA + 2R + 2\alpha P & P \\ P & 0 \end{bmatrix} + \begin{bmatrix} nI_n & 0_n \\ 0_n & -I_n \end{bmatrix} \leq 0$$

with a scalar $\alpha = 3.7$. Let $\varepsilon = 0.5$ be the desired precision. According to Theorem 6.3, the desired precision $\varepsilon = 0.5$ can be achieved by choosing the state-space discretization parameter $\eta = 0.18$. The goal is to stabilize the system Σ to a unit ball around the origin.

The simulation results are shown in Figs. 6.1-6.2. The trajectory x_2 of Σ' is obtained by applying a stabilization controller $v(t)$, and it is represented by the red line in Figure 6.1 ($x_{2,1}, x_{2,2}$ are the two state components of x_2). The trajectory x_1 of Σ is obtained via the control interface (6.25), and it is represented by the blue line in Figure 6.1 ($x_{1,1}, x_{1,2}$ are the two state components of x_1). The evolution of the output error $\|y_1 - y_2\|$ is depicted in Figure 6.2, and one can see that the desired precision 0.5 is satisfied at any time. The evolution of the input components v_1, v_2 for the abstract system Σ' and the input components u_1, u_2 for the concrete system Σ is plotted in Figure 6.3, respectively.

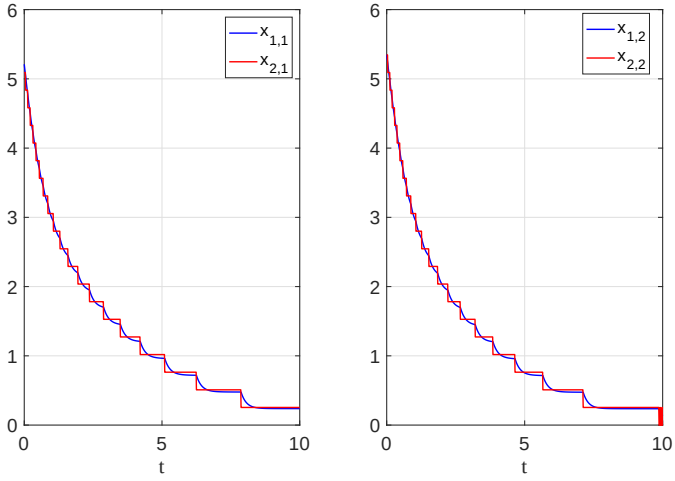

Figure 6.1: Output trajectory of the concrete system Σ_2 (blue line) and output trajectory of the abstract system Σ_2' (red line).

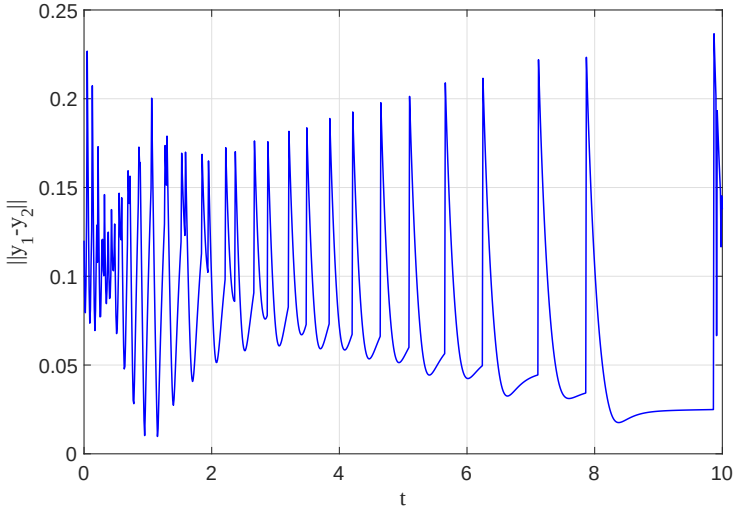

Figure 6.2: The evolution of $\|y - \zeta\|$.

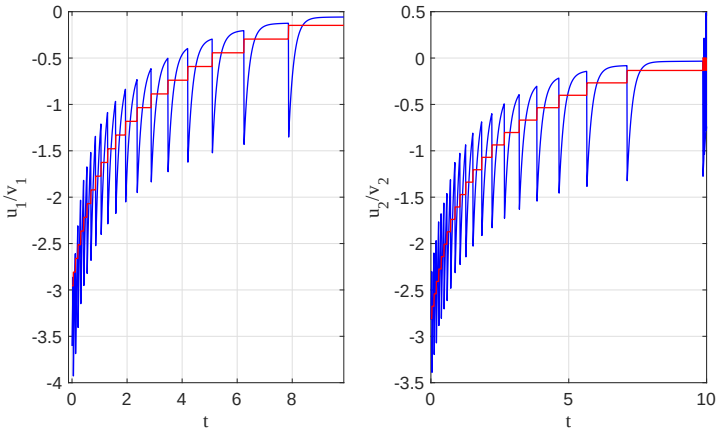

Figure 6.3: The evolution of the inputs u and v.

In this example, the desired precision is $\varepsilon = 0.5$ while the simulation result in Figure 6.2 shows that the output error $\|y_1 - y_2\|$ is at most 0.25. This means that the theoretical bound of η obtained using Theorem 6.3 can be conservative (due to the use of Lyapunov-like function).

6.6.2 Example 2

Consider a mobile robot moving in $\mathbb{R}^2$, the dynamics of which is given by

$$\Sigma_2 : \begin{cases} \dot{x}_1 = Ax_1 + Bu + w, \\ y_1 = x_1, \end{cases} \tag{6.26}$$

where

$$A = \begin{bmatrix} 0.2 & 0.3 \\ 0.5 & -0.5 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

The input set $U = [-5, 5] \times [-5, 5]$ and the disturbance set $W = [-0.05, 0.05] \times [-0.05, 0.05]$. It is readily seen that Σ_2 is not δ-ISS.

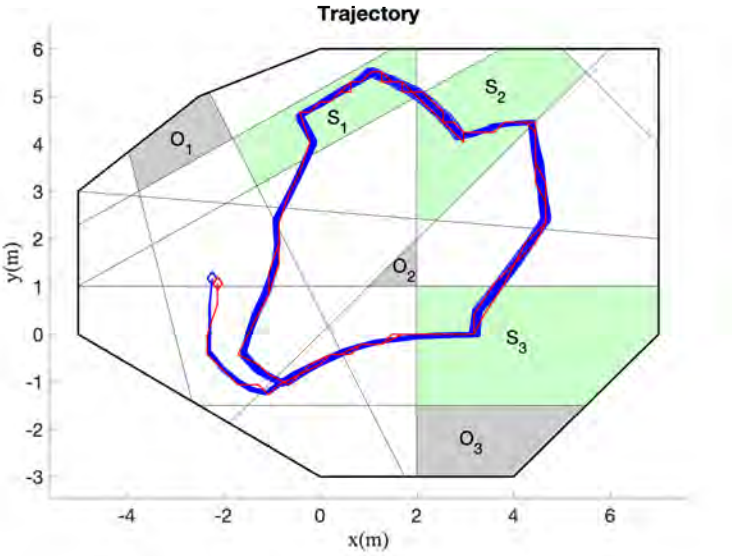

Figure 6.4: Output trajectories of the concrete system Σ_2 (blue lines) for 100 realizations of disturbance signals and output trajectory of the abstract system Σ_2' (red line).

The problem is to drive the robot in the bounded workspace $\mathbb{W}$ shown in Figure 6.4, where the three grey solid polygons O_1, O_2, O_3 represent obstacles and the three green solid polygons S_1, S_2, S_3 represent target regions. The goal of the motion planning problem consists in visiting all the three target regions S_1, S_2, S_3 infinitely many times while avoiding collision with the obstacles. This specification can be represented by an LTL [17] formula $\phi = \mathsf{G}\mathbb{W} \wedge \mathsf{G}(\neg(O_1 \vee O_2 \vee O_3)) \wedge \mathsf{GF}(S_1 \wedge S_2 \wedge S_3)$.

Let the desired precision be $\varepsilon = 1$. The control interface is designed as

$$u_v(t, v(t), x(t), \xi(t)) = v(t) - \frac{1}{2} B^T P(x(t) - \xi(t)), \tag{6.27}$$

where P is the solution to the ARE

$$A^T P + PA - PBB^T P + I_2 = 0.$$

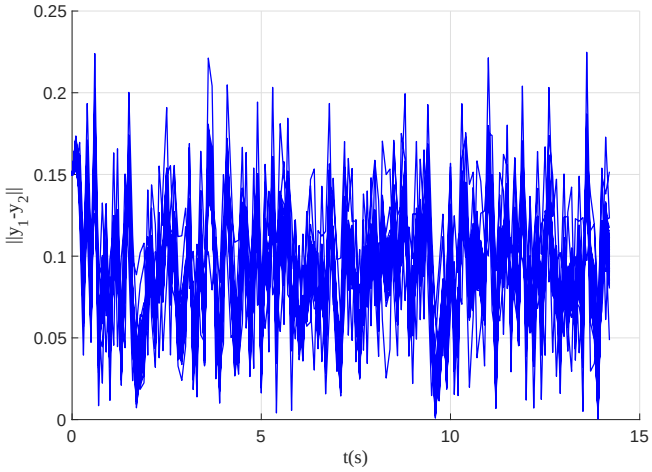

Figure 6.5: The evolution of $\|y_1 - y_2\|$ for 100 realizations of disturbance signals.

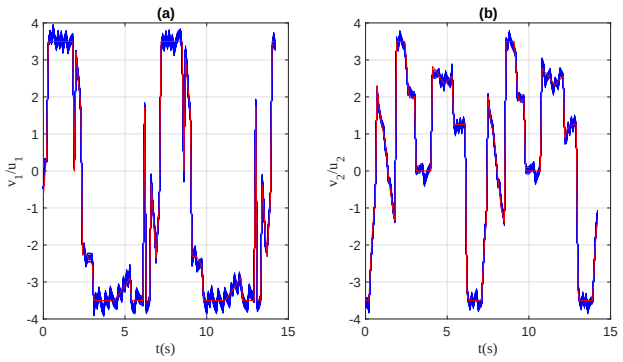

Figure 6.6: The evolution of the inputs u (blue lines) for 100 realizations of disturbance signals and v (red line).

According to Theorem 6.3, the desired precision ε can be achieved by choosing the state-space discretization parameter $\eta = 0.15$. Then, by further choosing $U'(t) = [-3.5, 3.5] \times [-3.5, 3.5], \forall t \geq 0$, one can guarantee that the control interface (6.27) is admissible. The abstract system (obtained by applying the state-space abstraction (6.3)) is denoted by Σ_2' and the output of Σ_2' is

denoted by y_2.

Using the LTL control synthesis toolbox LTLCon [118], we first synthesize a trajectory and the associated control policy for the abstract system Σ_2', which is shown by the red solid line in Figure 6.4. One can see that any trajectory remaining within the distance 1 from this trajectory satisfies the problem specification.

The output trajectory y_1 of Σ_2 is obtained by applying the synthesized input for the abstract system Σ_2' via the control interface (6.27). Furthermore, in order to validate robustness, we run 100 realizations of the disturbance trajectories. The resulting trajectories for these 100 realizations are shown (by the solid blue line) in Figure 6.4. One can see that all the trajectories satisfy the goal of the motion planning problem. The evolution of the output error $\|y_1 - y_2\|$ for the 100 realizations is depicted in Figure 6.5, and one can see that the desired precision is preserved at any time. In addition, the evolution of the input components v_1, v_2 for the abstract system Σ_2' and the input components u_1, u_2 for the concrete system Σ_2 are plotted in Figure 6.6, respectively. One can see that $u \in U$ (*i.e.,* the input constraint is satisfied) at any time.

Similar to Example 1, the desired precision is $\varepsilon = 1$ in this example while the simulation result in Figure 6.5 shows that the output error $\|y_1 - y_2\|$ is at most 0.25. This again means that the theoretical bound of η can be conservative.

6.7 Summary

This chapter involved the construction of discrete state-space symbolic models for continuous-time uncertain nonlinear systems. Firstly, a stability notion called η-C-Ω-GPS and its Lyapunov function characterizations were proposed. After that, a notion of robust approximate (bi)simulation relation was further introduced. It was shown that every continuous-time uncertain concrete system, under the condition that there exists an admissible control interface such that the augmented system can be made η-C-Ω-GPS, robustly approximately simulates its discrete state-space abstraction.

Chapter 7

Robust Satisfiability Check and Control Synthesis under STL Specifications

Linear temporal logic (LTL) formulae are useful for expressing complex specifications for dynamical systems. For finite transition systems, automated-based approaches are well-established for the model checking and control synthesis under LTL specifications. When infinite systems is considered, we showed in Chapter 6 that symbolic models can be constructed to assist control synthesis. Compared to LTL, signal temporal logic (STL) further allows to specify desired quantitative temporal and spatial properties on the system, which is beneficial for cyber-physical systems [121]. Nevertheless, the automated control synthesis under STL specifications has not been investigated to the same extent. In this chapter, we study the online control synthesis of uncertain systems under STL specifications, in which a new approach based on STL, reachability analysis, and the notion of tube-based temporal logic trees is proposed. In addition to the control synthesis problem, we also study the robust satisfiability check problem, *i.e.,* check whether or not there exists a control policy such that the resulting trajectory of the underlying system satisfies properties specified by the STL formula under all possible uncertainties.

7.1 Introduction

Rapid growth of robotic applications, such as autonomous vehicles and service robots, has stimulated the need of new control synthesis approaches to safely

accomplish more complex objectives such as nondeterministic, periodic, or sequential tasks. Temporal logics, such as linear temporal logic (LTL) [17] and signal temporal logic (STL) [18], have shown capability in expressing such objectives for dynamical systems in the last decade. Various control approaches have been developed accordingly.

LTL focuses on the Boolean satisfaction of properties by given signals. Existing control approaches that use LTL mainly rely on a finite abstraction of the system dynamics and a language equivalent automata [19] representation of the LTL specification. The controller is synthesized by solving a game over the product automata [20], [21]. Other control approaches include optimization-based [122], [123] and sampling-based methods [124], [125]. STL is a more recently developed temporal logic, which allows the specification of properties over dense-time. Due to a number of advantages, such as explicitly treating real-valued signals [18], and admitting qualitative semantics [23], control synthesis under STL specifications has gained popularity in the last few years.

Existing approaches that deal with control synthesis under STL specifications include barrier function [24], [25] and optimization methods [26]–[28]. Barrier function methods are mainly used for continuous-time systems. The idea is to transfer the STL formula into one or several (time-varying) control barrier functions, and then obtain feedback control laws by solving quadratic programs [24], [25]. This method is computationally efficient. However, as the existence and design of barrier functions are still open problems, it currently mainly applies to deterministic affine systems. Optimization methods are mainly used for discrete-time systems. The idea is to encode STL formulas as mixed-integer constraints, and then the satisfying controller can be obtained by solving a series of optimization problems [26], [27]. An extension of the mixed-integer formulation is investigated for linear systems with additive bounded disturbances in [28], where the controller is obtained by solving the optimization problem at each time step in a receding horizon fashion. The mixed-integer programming approach is sound but not complete for uncertain systems. Moreover, it deals with only bounded STL specifications. Other control synthesis approaches include sampling-based [29] and learning-based methods [30], [31].

We note that although various methods exist for the control synthesis under STL specifications, guaranteeing robustness under uncertainties is still a challenging problem. One core contribution of this chapter is on robust control synthesis for uncertain systems under STL specifications. In addition, we

also study the satisfiability check problem, *i.e.,* the problem to check whether or not there exists a control policy such that the resulting trajectory of the underlying system satisfies properties specified by the temporal logic formula. To the best of our knowledge, the satisfiability check problem and its robust variant for uncertain systems (*i.e.,* check the existence of a control policy under all possible uncertainties) under STL specifications have not been solved so far.

Motivated by the above considerations, this chapter considers the robust satisfiability check and online control synthesis problems for uncertain discrete-time systems under STL specifications. It is inspired by [126], where relationships between temporal operators and reachable sets are developed, and [127], where the notion of temporal logic tree (TLT) is proposed for LTL. However, we note that it is far from straightforward to extend these results to general STL formulas. The contributions of this chapter are summarized as follows:

(i) A real-time version of satisfaction relation and a tube-based temporal logic tree (tTLT) are proposed for STL formulas. A correspondence between STL formulas and tTLT is established via reachability analysis on the underlying systems. An algorithm is proposed for the automated construction of tTLT. Note that the tTLTs in this chapter are different from the TLTs defined for LTL formulas in [127], due to the time constraints encoded in the STL formulas.

(ii) We use the tTLT to address the STL robust satisfiability check problem. A sufficient condition is obtained for uncertain systems. That is, the STL robust satisfiability check can be replaced by a tTLT robust satisfiability check. Moreover, when the underlying system is deterministic, a necessary and sufficient condition is obtained for the satisfiability check.

(iii) We solve the STL control synthesis problem for uncertain systems. An online control synthesis algorithm is proposed based on the constructed tTLT from the STL formula. When the STL formula is robustly satisfiable and the initial state of the system belongs to the initial root node of the tTLT, it is proven that the trajectory generated by the proposed online control synthesis algorithm satisfies the STL formula.

7.2 Problem Formulation

7.2.1 Model description

Consider an uncertain discrete-time control system of the form

$$x_{k+1} = f(x_k, u_k, w_k), \tag{7.1}$$

where $x_k := x(t_k) \in \mathbb{R}^n, u_k := u(t_k) \in U, w_k := w(t_k) \in W, k \in \mathbb{N}$ are the state, control input, and disturbance at time t_k, respectively. The time sequence $\{t_k\}$ can be seen as a sequence of sampling instants, which satisfy $t_0 < t_1 < \cdots$. The control input is constrained to a compact set $U \subset \mathbb{R}^m$ and the disturbance is constrained to a compact set $W \subset \mathbb{R}^l$.

Definition 7.1. *A* control policy $\boldsymbol{\nu} = \nu_0\nu_1 \ldots \nu_k \ldots$ *is a sequence of maps* $\nu_k : \mathbb{R}^n \to U, \forall k \in \mathbb{N}$. *Denote by* $\mathcal{U}_{\geq k}$ *the set of all control policies that start from time* t_k.

Definition 7.2. *A disturbance signal* $\boldsymbol{w} = w_0w_1 \ldots w_k \ldots$ *is called* admissible *if* $w_k \in W, \forall k \in \mathbb{N}$. *Denote by* $\mathcal{W}_{\geq k}$ *the set of all admissible disturbance signals that start from time* t_k.

The solution of (7.1) is defined as a discrete-time signal $\boldsymbol{x} := x_0x_1 \ldots$. We call $\boldsymbol{x}$ a *trajectory* of (7.1) if there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq 0}$ and a disturbance signal $\boldsymbol{w} \in \mathcal{W}_{\geq 0}$ satisfying (7.1), *i.e.,*

$$x_{k+1} = f(x_k, \nu_k(x_k), w_k), \forall k.$$

We use $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}}(t_k)$ to denote the trajectory point reached at time t_k under the control policy $\boldsymbol{\nu}$ and the disturbance $\boldsymbol{w}$ from initial state x_0 at time t_0.

The deterministic system is defined by

$$x_{k+1} = f_d(x_k, u_k) \tag{7.2}$$

and $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu}}(t_k)$ denotes the solution at time t_k of the deterministic system when the control policy is $\boldsymbol{\nu}$ and the initial state is x_0 at time t_0.

7.2.2 Reachability operators

The definitions of maximal and minimal reachable tube are given as follows.

Definition 7.3. *Consider the system (7.1), three sets $\Omega_1, \Omega_2, \mathcal{C} \subseteq \mathbb{R}^n$, and a time interval $[a, b]$. The maximal reachable tube from Ω_1 to Ω_2 is defined as*

$$\begin{aligned}\mathcal{R}^M(\Omega_1, \Omega_2, \mathcal{C}, [a,b], k) = \Big\{ & x_k \in \Omega_1 \mid \exists \boldsymbol{\nu} \in \mathcal{U}_{\geq k}, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}, \\ & \text{s.t. } \exists t_{k'} \in [\max\{a, t_k\}, b], \boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}(t_{k'}) \in \Omega_2, \\ & \forall t_{k''} \in [t_k, t_{k'}], \boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}(t_{k''}) \in \mathcal{C} \Big\}, t_k \in [0, b].\end{aligned}$$

The set $\mathcal{R}^M(\Omega_1, \Omega_2, \mathcal{C}, [a, b], k)$ collects all states in Ω_1 at time t_k from which there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ that, despite the worst disturbance signals, drives the system to the target set Ω_2 at some time instant $t_{k'} \in [\max\{a, t_k\}, b]$ while satisfying constraints defined by $\mathcal{C}$ prior to reaching the target.

Definition 7.4. *Consider the system (7.1), two sets $\Omega_1, \Omega_2 \subseteq \mathbb{R}^n$, and a time interval $[a, b]$. The minimal reachable tube from Ω_1 to Ω_2 is defined as*

$$\begin{aligned}\mathcal{R}^m(\Omega_1, \Omega_2, [a,b], k) = \Big\{ & x_k \in \Omega_1 \mid \forall \boldsymbol{\nu} \in \mathcal{U}_{\geq k}, \exists \boldsymbol{w} \in \mathcal{W}_{\geq k}, \\ & \text{s.t. } \exists t_{k'} \in [\max\{a, t_k\}, b], \boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}(t_{k'}) \in \Omega_2 \Big\}, t_k \in [0, b].\end{aligned}$$

The set $\mathcal{R}^m(\Omega_1, \Omega_2, [a, b], k)$ collects all states in Ω_1 at time t_k from which no matter what control policy $\boldsymbol{\nu}$ is applied, there exists a disturbance signal that drives the system to the target set Ω_2 at some time instant $t_{k'} \in [\max\{a, t_k\}, b]$. In this definition, the constraint set $\mathcal{C}$ is redundant.

7.2.3 Problems

Before proceeding, the following definitions are required.

Definition 7.5. *(Satisfiability) Consider the deterministic system (7.2) and the STL formula φ. We say φ* is satisfiable from the initial state x_0 *if there exists a control policy $\boldsymbol{\nu}$ such that*

$$\boldsymbol{x}_{x_0}^{\boldsymbol{\nu}} \vDash \varphi. \tag{7.3}$$

Definition 7.6. *(Robust satisfiability) Consider the uncertain system (7.1) and the STL formula φ. We say φ* is robustly satisfiable from the initial state x_0 *if there exists a control policy $\boldsymbol{\nu}$ such that*

$$\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0}. \tag{7.4}$$

Given an STL formula φ, let

$$\mathbb{S}_\varphi := \{x_0 \in \mathbb{R}^n | \varphi \text{ is (robustly) satisfiable from } x_0\} \tag{7.5}$$

denote the set of initial states from which φ is (robustly) satisfiable. Then, one has the following Lemma.

Lemma 7.1. *[126] Consider the system (7.1) and the STL formulas φ, φ_1, and φ_2. Then, one has*

i) until: $\mathbb{S}_{\varphi_1 \mathsf{U}_{[a,b]} \varphi_2} = \mathcal{R}^M(\mathbb{R}^n, \mathbb{S}_{\varphi_2}, \mathbb{S}_{\varphi_1}, [a,b], 0)$;

ii) always: $\mathbb{S}_{\mathsf{G}_{[a,b]}\varphi} = \overline{\mathcal{R}^m(\mathbb{R}^n, \overline{\mathbb{S}_\varphi}, [a,b], 0)}$.

The problems under consideration are now formulated as follows.

Problem 7.1 (Robust satisfiability check)**.** *Consider the system (7.1) and a task specification expressed as an STL formula φ. For an initial state x_0, determine whether the task specification φ is robustly satisfiable or not.*

Problem 7.2 (Online control synthesis)**.** *Consider the system (7.1) and a task specification expressed as an STL formula φ. For an initial state x_0, find, if there exists, a control policy $\boldsymbol{\nu} = \nu_0\nu_1 \ldots \nu_k \ldots$ such that the resulting trajectory $\boldsymbol{x} = x_0x_1 \ldots x_k \ldots$ satisfies φ.*

7.3 Real-Time STL Semantics

In this section, a real-time version of STL semantics is proposed.

It has been proven in [28] that each STL formula has an equivalent STL formula in positive normal form, *i.e.,* negations only occur adjacent to predicates. The syntax of the positive normal form STL is given by

$$\varphi ::= \top \mid \mu \mid \neg\mu \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathsf{U}_\mathrm{I} \varphi_2 \mid \mathsf{G}_\mathrm{I}\varphi. \tag{7.6}$$

Before proceeding, the following definition is required.

Definition 7.7 (Suffix and Completions)**.** *Given a discrete-time signal $\boldsymbol{x} = x_0x_1 \ldots$, we say that a partial signal $\boldsymbol{s} = s_l s_{l+1} \ldots, l \in \mathbb{N}$, is a* suffix *of the signal $\boldsymbol{x}$ if $\forall k' \geq l, s_{k'} = x_{k'}$. The set of* completions *of a partial signal $\boldsymbol{s}$, denoted by $C(\boldsymbol{s})$, is given by*

$$C(\boldsymbol{s}) := \{\boldsymbol{x} : \boldsymbol{s} \text{ is a suffix of } \boldsymbol{x}\}.$$

Given a time instant t_k and a time interval $[a, b]$, denote by

$$t_k + [a, b] := [t_k + a, t_k + b].$$

The satisfaction relation of a partial signal $\boldsymbol{s} = s_l s_{l+1} \dots$ starting from time instant t_l is defined.

Definition 7.8. *The real-time STL semantics are recursively defined by*

$$\begin{aligned}
(\boldsymbol{s}, t_l) \models_{\asymp} \mu &\Leftrightarrow g_\mu(\boldsymbol{s}(t_l)) \geq 0, t_l = t_0, \\
(\boldsymbol{s}, t_l) \models_{\asymp} \neg\mu &\Leftrightarrow \neg((\boldsymbol{s}, t_l) \models_{\asymp} \mu), t_l = t_0, \\
(\boldsymbol{s}, t_l) \models_{\asymp} \varphi_1 \wedge \varphi_2 &\Leftrightarrow (\boldsymbol{s}, t_l) \models_{\asymp} \varphi_1 \wedge (\boldsymbol{s}, t_l) \models_{\asymp} \varphi_2, \\
&\quad t_l \in t_0 + [0, ||\varphi_1 \wedge \varphi_2||], \\
(\boldsymbol{s}, t_l) \models_{\asymp} \varphi_1 \vee \varphi_2 &\Leftrightarrow (\boldsymbol{s}, t_l) \models_{\asymp} \varphi_1 \vee (\boldsymbol{s}, t_l) \models_{\asymp} \varphi_2, \\
&\quad t_k \in t_0 + [0, ||\varphi_1 \vee \varphi_2||], \\
(\boldsymbol{s}, t_l) \models_{\asymp} \varphi_1 \mathsf{U}_{[a,b]} \varphi_2 &\Leftrightarrow \exists t_{k'} \in [\max\{t_0 + a, t_l\}, t_0 + b] \\
&\quad \textit{s.t.}\ (\boldsymbol{s}, t_{k'}) \models_{\asymp} \varphi_2 \ \wedge\ \forall t_{k''} \in [t_l, \\
&\quad t_{k'}], (\boldsymbol{s}, t_{k''}) \models_{\asymp} \varphi_1, t_l \in t_0 + [0, b], \\
(\boldsymbol{s}, t_l) \models_{\asymp} \mathsf{G}_{[a,b]} \varphi_1 &\Leftrightarrow \forall t_{k'} \in [\max\{t_0 + a, t_l\}, t_0 + b] \\
&\quad \textit{s.t.}\ (\boldsymbol{s}, t_{k'}) \models_{\asymp} \varphi_1, t_l \in t_0 + [0, b].
\end{aligned}$$

The real-time satisfaction relation $(\boldsymbol{s}, t_l) \models_{\asymp} \varphi$ denotes that the partial signal $\boldsymbol{s}$ is the suffix of a satisfying trajectory that starts from t_0, *i.e.,*

$$(\boldsymbol{s}, t_l) \models_{\asymp} \varphi \Leftarrow \exists \boldsymbol{x} \in C(\boldsymbol{s}), (\boldsymbol{x}, t_0) \models \varphi.$$

Note that when $t_l = t_0$, the satisfaction relation $(\boldsymbol{s}, t_l) \models_{\asymp} \varphi$ degenerates to $(\boldsymbol{s}, t_l) \models \varphi$.

Definition 7.9. *Consider the deterministic system (7.2) and the STL formula φ. We say φ* is satisfiable from the state x_k at time t_k *if there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ such that*

$$(\boldsymbol{x}^{\boldsymbol{\nu}}_{x_k}, t_k) \models_{\asymp} \varphi.$$

Definition 7.10. *Consider the uncertain system (7.1) and the STL formula φ. We say φ* is robustly satisfiable from the state x_k at time t_k *if there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ such that*

$$(\boldsymbol{x}^{\boldsymbol{\nu}, \boldsymbol{w}}_{x_k}, t_k) \models_{\asymp} \varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}.$$

Note that when $t_k = t_0$, Definitions 7.10 and 7.9 degenerate to Definitions 7.6 and 7.5, respectively. Given an STL formula φ, let

$$\mathbb{S}_{\varphi}(t_k) := \{x_k \in \mathbb{R}^n | \varphi \text{ is (robustly) satisfiable from } x_k \text{ at } t_k\} \tag{7.7}$$

denote the set of states from which φ is robustly satisfiable at t_k. Then, we have the following result.

Theorem 7.1. *Consider the system (7.1), a predicate μ, and STL formulas φ_1, and φ_2. Then,*

i) negation: $\mathbb{S}_{\neg\mu}(t_k) = \overline{\mathbb{S}_{\mu}(t_k)}$;

ii) conjunction: $\mathbb{S}_{\varphi_1 \wedge \varphi_2}(t_k) \subseteq \mathbb{S}_{\varphi_1}(t_k) \cap \mathbb{S}_{\varphi_2}(t_k)$;

iii) disjunction: $\mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k) \supseteq \mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k)$;

iv) until: $\mathbb{S}_{\varphi_1 \mathsf{U}_{[a,b]} \varphi_2}(t_k) = \mathcal{R}^M(\mathbb{R}^n, \mathbb{S}_{\varphi_2}, \mathbb{S}_{\varphi_1}, [a,b], k)$;

v) always: $\mathbb{S}_{\mathsf{G}_{[a,b]}\varphi_1}(t_k) = \overline{\mathcal{R}^m(\mathbb{R}^n, \overline{\mathbb{S}_{\varphi_1}}, [a,b], k)}$,

where $\mathbb{S}_{\varphi_1}$ and $\mathbb{S}_{\varphi_2}$ are defined in (7.5).

Proof. Item i) is trivial. The proofs of items iv) and v) follow from Lemma 7.1 and Definitions 7.3, 7.4, and 7.8. In the following, we will prove items ii) and iii).

Assume that $x_k \in \mathbb{S}_{\varphi_1 \wedge \varphi_2}(t_k)$. According to Definition 7.8 and (7.7), one has that there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ such that

$$(\boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}, t_k) \mid\asymp \varphi_1, , \forall \boldsymbol{w} \in \mathcal{W}_{\geq k} \ \wedge \ (\boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}, t_k) \mid\asymp \varphi_2, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}.$$

That is, $x_k \in \mathbb{S}_{\varphi_1}(t_k), x_k \in \mathbb{S}_{\varphi_2}(t_k)$. Thus, $x_k \in \mathbb{S}_{\varphi_1 \wedge \varphi_2}(t_k) \Rightarrow x_k \in \mathbb{S}_{\varphi_1}(t_k) \cap \mathbb{S}_{\varphi_2}(t_k)$. The other direction may not hold because it could happen that for a state x_k, there exist two control policies $\boldsymbol{\nu}_1, \boldsymbol{\nu}_2 \in \mathcal{U}_{\geq k}$ such that $(\boldsymbol{x}_{x_k}^{\boldsymbol{\nu}_1,\boldsymbol{w}}, t_k) \mid\asymp \varphi_1, (\boldsymbol{x}_{x_k}^{\boldsymbol{\nu}_2,\boldsymbol{w}}, t_k) \mid\asymp \varphi_2, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}$ (*i.e.,* $x_k \in \mathbb{S}_{\varphi_1}(t_k) \cap \mathbb{S}_{\varphi_2}(t_k)$). However, there is no control policy which ensures the robust satisfaction of $\varphi_1 \wedge \varphi_2$ at t_k.

Assume now that $x_k \in \mathbb{S}_{\varphi_1}(t_k)$, then one has that there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ such that $(\boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}, t_k) \mid\asymp \varphi_1, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}$. Moreover, according to STL syntax, one further has $(\boldsymbol{x}_{x_k}^{\boldsymbol{\nu},\boldsymbol{w}}, t_k) \mid\asymp \varphi_1 \vee \varphi_2, \forall \boldsymbol{w} \in \mathcal{W}_{\geq k}$. That is, $x_k \in \mathbb{S}_{\varphi_1}(t_k) \Rightarrow x_k \in \mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k)$. Similarly, one can also get $x_k \in \mathbb{S}_{\varphi_2}(t_k) \Rightarrow$

$x_k \in \mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k)$. Therefore, $x_k \in \mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k) \Rightarrow x_k \in \mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k)$. The other direction may not hold because it could happen that there exists no state such that either φ_1 or φ_2 is robustly satisfiable from at t_k, *i.e.,* $\mathbb{S}_{\varphi_1}(t_k) = \emptyset$, $\mathbb{S}_{\varphi_2}(t_k) = \emptyset$ and thus $\mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k) = \emptyset$. However, there exists a state x_k^* from which there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq k}$ such that

$$(\boldsymbol{x}_{x_k^*}^{\boldsymbol{\nu},\boldsymbol{w}_1}, t_k) \mid\asymp \varphi_1, \forall \boldsymbol{w}_1 \in \mathcal{W}_1 \wedge\ (\boldsymbol{x}_{x_k^*}^{\boldsymbol{\nu},\boldsymbol{w}_2}, t_k) \mid\asymp \varphi_2, \forall \boldsymbol{w}_2 \in \mathcal{W}_{\geq k} \setminus \mathcal{W}_1,$$

where $\mathcal{W}_1 \subset \mathcal{W}_{\geq k}$. In this case, one has $x_k^* \in \mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k)$. □

Note that in item iii), $x_k \in \mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k) \Rightarrow x_k \in \mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k)$ does not hold due to the uncertainty caused by the disturbances. For deterministic systems (7.2), we have the following result.

Corollary 7.1. *Consider the deterministic system (7.2) and the STL formulas φ_1, φ_2. Then, item i)-ii), iv)-v) of Theorem 7.1 hold and*

$$\mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k) = \mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k).$$

7.4 Tube-Based Temporal Logic Tree

In our previous work [127], a notion of TLT is proposed for LTL formula. In this chapter, STL formulas are considered and a tTLT is introduced.

Definition 7.11. *A tTLT is a tree for which*

- *each node is either a* tube *node that maps from the nonnegative time axis, i.e., $\mathbb{R}_{\geq 0}$, to the subset of $\mathbb{R}^n$, or an* operator *node that belongs to $\{\wedge, \vee, \mathsf{U}_I, \mathsf{G}_I\}$;*
- *the root node and the leaf nodes are* tube *nodes;*
- *if a* tube *node is not a leaf node, its unique child is an* operator *node;*
- *the children of any* operator *node are* tube *nodes.*

Definition 7.12. *A* complete path *of a tTLT is a path that starts from the root node and ends at a leaf node. Any subsequence of a complete path is called a* fragment *of the complete path.*

Then, the following result is obtained.

Theorem 7.2. *For the system (7.1) and every STL formula φ in positive normal form (7.6), a tTLT, denoted by $\mathcal{T}_\varphi$, can be constructed from φ through the reachability operators $\mathcal{R}^M$ and $\mathcal{R}^m$.*

Proof. Firstly, we show that each predicate μ and its negation $\neg\mu$ have a corresponding tTLT. Given a predicate μ, one has $\mathbb{S}_\mu = \{x_0 : g_\mu(x_0) \geq 0\}$. Then, the tTLT $\mathcal{T}_\mu$ (or $\mathcal{T}_{\neg\mu}$) has only one root node, which is given by $\mathbb{S}_\mu$ (or $\overline{\mathbb{S}_\mu}$). Following the similar idea, we can show that $\top$ (or $\bot$) has a corresponding tTLT that only has a root node, which is given by $\mathbb{R}^n$ (or $\emptyset$).

Then, we will prove that if the STL formulas φ_1 and φ_2 have corresponding tTLTs, respectively, then the STL formulas $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$, $\varphi_1 \mathsf{U}_{[a,b]} \varphi_2$, and $\mathsf{G}_{[a,b]}\varphi_1$ have their corresponding tTLTs, respectively.

Case 1: Boolean operators $\wedge$ and $\vee$. Consider two STL formulas φ_1, φ_2 and their corresponding tTLTs $\mathcal{T}_{\varphi_1}, \mathcal{T}_{\varphi_2}$. The root nodes of $\mathcal{T}_{\varphi_1}, \mathcal{T}_{\varphi_2}$ are denoted by $\mathbb{X}_{\varphi_1}(t_k)$ and $\mathbb{X}_{\varphi_2}(t_k)$, respectively. The tTLT $\mathcal{T}_{\varphi_1 \wedge \varphi_2}$ ($\mathcal{T}_{\varphi_1 \vee \varphi_2}$) can be constructed by connecting $\mathbb{X}_{\varphi_1}(t_k)$ and $\mathbb{X}_{\varphi_2}(t_k)$ through the operator node $\wedge$ ($\vee$) and taking the intersection (or union) of the two root nodes, *i.e.,* $\mathbb{X}_{\varphi_1}(t_k) \cap \mathbb{X}_{\varphi_2}(t_k)$ ($\mathbb{X}_{\varphi_1}(t_k) \cup \mathbb{X}_{\varphi_2}(t_k)$), to be the root node. An illustrative diagram for $\varphi_1 \wedge \varphi_2$ is given in Figure 7.1.

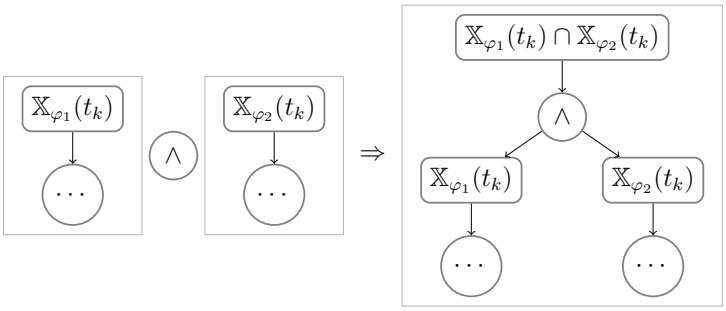

Figure 7.1: Illustrative diagram of construction tTLT for $\varphi_1 \wedge \varphi_2$.

Case 2: Until operator $\mathsf{U}_{[a,b]}$. Consider two STL formulas φ_1, φ_2 and their corresponding tTLTs $\mathcal{T}_{\varphi_1}, \mathcal{T}_{\varphi_2}$. The root nodes of $\mathcal{T}_{\varphi_1}, \mathcal{T}_{\varphi_2}$ are denoted by $\mathbb{X}_{\varphi_1}(t_k)$ and $\mathbb{X}_{\varphi_2}(t_k)$, respectively. In addition, the leaf nodes of $\mathcal{T}_{\varphi_1}$ are denoted by $\mathbb{Y}^1_{\varphi_1}(t_k), \cdots, \mathbb{Y}^N_{\varphi_1}(t_k)$, where N is the total number of leaf nodes of $\mathcal{T}_{\varphi_1}$. The tTLT $\mathcal{T}_{\varphi_1 \mathsf{U}_{[a,b]} \varphi_2}$ can be constructed by the following steps: 1) replace each leaf node $\mathbb{Y}^i_{\varphi_1}(t_k)$ by $\mathcal{R}^M(\mathbb{R}^n, \mathbb{X}_{\varphi_2}(t_0), \mathbb{Y}^i_{\varphi_1}(t_0), [a,b], k)$; 2) update $\mathcal{T}_{\varphi_1}$

from the leaf nodes to the root node with the new leaf nodes; and 3) connect each leaf node of the updated $\mathcal{T}_{\varphi_1}$ and the root node of $\mathcal{T}_{\varphi_2}$, *i.e.*, $\mathbb{X}_{\varphi_2}(t_k)$, with the operator node $\mathsf{U}_{[a,b]}$. One illustrative diagram for $\mathsf{U}_{[a,b]}$ is given in Figure 7.2.

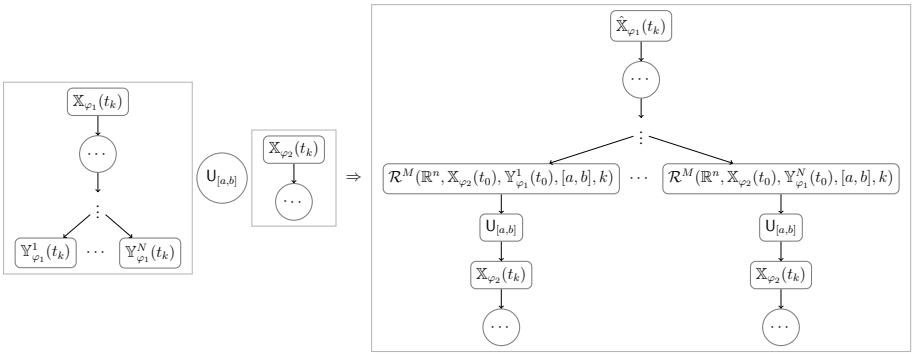

Figure 7.2: Illustrative diagram of construction tTLT for $\varphi_1 \mathsf{U}_{[a,b]} \varphi_2$.

Case 3: Always operator $\mathsf{G}_{[a,b]}$. Consider an STL formula φ_1 and its corresponding tTLT $\mathcal{T}_{\varphi_1}$. The root node of $\mathcal{T}_{\varphi_1}$ is given by $\mathbb{X}_{\varphi_1}(t_k)$. The tTLT $\mathcal{T}_{\mathsf{G}_{[a,b]}\varphi_1}$ can be constructed by connecting $\mathbb{X}_{\varphi_1}(t_k)$ through the operator $\mathsf{G}_{[a,b]}$ and making the tube $\overline{\mathcal{R}^m(\mathbb{R}^n, \overline{\mathbb{X}_{\varphi_1}(t_0)}, [a,b], k)}$ the root node. An illustrative diagram for $\mathsf{G}_{[a,b]}$ is given in Figure 7.3.

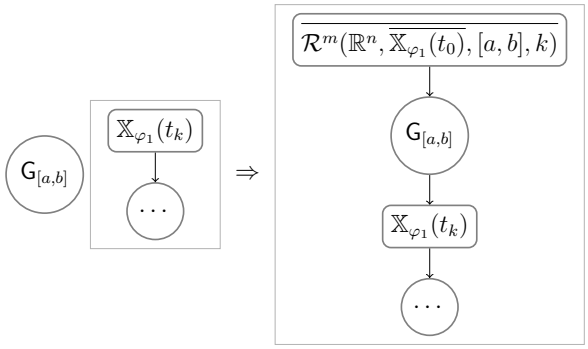

Figure 7.3: Illustrative diagram of construction tTLT for $\mathsf{G}_{[a,b]} \varphi_1$.

□

Based on Theorem 7.2, Algorithm 7.1 is designed for the construction of tTLT $\mathcal{T}_\varphi$. It takes the syntax tree[1] of the STL formula φ as input. For an STL formula, the nodes of its syntax tree are either predicate or operator nodes. In addition, all the leaf nodes are predicates and all other nodes are operators.

Algorithm 7.1 *tTLTConstruction*

Input: the syntax tree of STL formula φ.
Output: the tTLT $\mathcal{T}_\varphi$.

1: **for** each leaf node μ (or $\neg\mu$) of the syntax tree **do**,
2: Replace μ (or $\neg\mu$) by $\mathbb{S}_\mu$ (or $\mathbb{S}_{\neg\mu}$),
3: **end for**
4: **for** each operator node of the syntax tree through a bottom-up traversal, **do**
5: Construct $\mathcal{T}_\varphi$ according to Theorem 7.2,
6: **end for**

Example 7.1. *Consider the formula* $\varphi = \mathsf{F}_{[a_1,b_1]}\mathsf{G}_{[a_2,b_2]}\mu_1 \wedge \mu_2 \mathsf{U}_{[a_3,b_3]}\mu_3$, *where* $\mu_i, i = \{1,2,3\}$ *are predicates. The syntax tree of* φ *is shown on the left-hand side of Figure 7.4. The corresponding TLT for* φ *(constructed using Algorithm 7.1) is shown on the right-hand side of Figure 7.4, where*

$$\begin{aligned}
\mathbb{X}_4(t_k) &= \overline{\mathcal{R}^m(\mathbb{R}^n, \overline{\mathbb{S}_{\mu_1}}, [a_2, b_2], k)}, \\
\mathbb{X}_3(t_k) &= \mathcal{R}^M(\mathbb{R}^n, \mathbb{S}_{\mu_3}, \mathbb{S}_{\mu_2}, [a_3, b_3], k), \\
\mathbb{X}_2(t_k) &= \mathcal{R}^M(\mathbb{R}^n, \mathbb{X}_4(t_0), \mathbb{R}^n, [a_1, b_1], k), \\
\mathbb{X}_1(t_k) &= \mathbb{X}_2(t_k) \cap \mathbb{X}_3(t_k).
\end{aligned}$$

Remark 7.1. *Given an STL formula* φ *in positive normal form, let* N *denote the number of Boolean operators and* M *the number of temporal operators contained in* φ. *Let* $\mathcal{T}_\varphi$ *be the tTLT corresponds to* φ. *Then,* $\mathcal{T}_\varphi$ *has at most* $2N$ *number of complete paths. In addition, each complete path has at most* $2(N+M)+1$ *number of nodes, out of which at most* $N+M$ *are non-root tube nodes. Thus, one can conclude that* $\mathcal{T}_\varphi$ *contains at most* $4N(N+M)+1$ *number of nodes, out of which at most* $2N(N+M)+1$ *number of tube nodes.*

[1] A syntax tree is a tree representation of the syntactic structure of the source code [128].

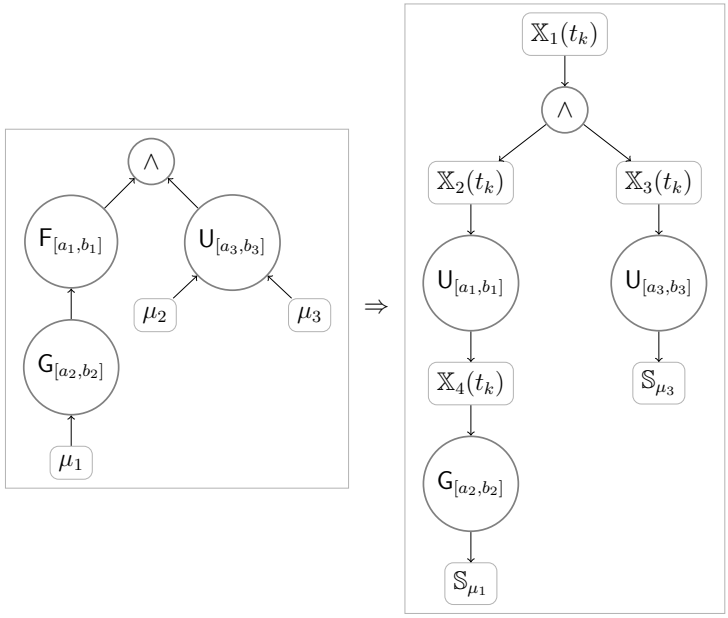

Figure 7.4: Example 7.1: syntax tree (left) and tTLT (right) for $\varphi = \mathsf{F}_{[a_1,b_1]}\mathsf{G}_{[a_2,b_2]}\mu_1 \wedge \mu_2\mathsf{U}_{[a_3,b_3]}\mu_3$. Recall that $\mathsf{F}_{[a,b]}\varphi = \top\mathsf{U}_{[a,b]}\varphi$.

7.5 Robust Satisfiability Check

This section addresses robust satisfiability check as defined by Problem 7.1. Before that, we need to define the satisfaction relation between a trajectory and a tTLT.

7.5.1 Definitions

We first define the *maximal temporal fragment (MTF)* for a tTLT, which plays an important role when simplifying the tTLT.

Definition 7.13. *A MTF of a complete path of the tTLT is one of the following types of fragment:*

1) *a fragment from the root node to the parent of the first Boolean operator node ($\wedge$ or $\vee$);*

2) *a fragment from one child of one Boolean operator node to the parent of the next Boolean operator node;*

3) *a fragment from one child of the last Boolean operator node to the leaf node.*

One can conclude from Definition 7.13 that any MTF starts and ends with a tube node and contains no Boolean operator nodes.

Definition 7.14. *A time coding of (a complete path of) the tTLT is an assignment of an activation time instant $t_{\kappa_i}, \kappa_i \in \mathbb{N}$ for each tube node $\mathbb{X}_i$ of (the complete path of) the tTLT.*

Now, we further define the satisfaction relation between a trajectory $\boldsymbol{x}$ and a complete path of the tTLT.

Definition 7.15. *Consider a trajectory $\boldsymbol{x} := x_0 x_1 \dots$ and a complete path $\boldsymbol{p}$ of a tTLT encoded in the form of $\boldsymbol{p} = \mathbb{X}_0 \Theta_1 \mathbb{X}_1 \Theta_2 \dots \Theta_{N_f} \mathbb{X}_{N_f}$, where N_f is the number of operator nodes contained in the complete path, $\mathbb{X}_i : \mathbb{R}_{\geq 0} \to 2^{\mathbb{R}^n}, \forall i \in \{0, 1, \dots, N_f\}$ represent tube nodes, and $\Theta_j, \forall j \in \{1, \dots, N_f\}$ represent operator nodes. We say $\boldsymbol{x}$* satisfies *$\boldsymbol{p}$, denoted by $\boldsymbol{x} \cong \boldsymbol{p}$, if there exists a time coding for $\boldsymbol{p}$ such that*

i) if $\Theta_i \in \{\wedge, \vee\}$, then $t_{\kappa_i} = t_{\kappa_{i-1}}$;

ii) if $\Theta_i = \mathsf{U}_I$, then $t_{\kappa_i} \in t_{\kappa_{i-1}} + I$;

iii) if $\Theta_i = \mathsf{G}_I$, then $t_{\kappa_i} = \operatorname{argmax}_{t_k} \{t_k \in t_{\kappa_{i-1}} + I\}$.

and

iv) $x_k \in \mathbb{X}_i(t_{k-\kappa_i}), \forall k \in [\kappa_i, \kappa_{i+1}], i = 0, \dots, N_f - 1$;

v) $x_{\kappa_{N_f}} \in \mathbb{X}_{N_f}(t_0)$.

Remark 7.2. *From items i)-iii) of Definition 7.15, one has that $t_{\kappa_0} \leq t_{\kappa_1} \leq \dots \leq t_{\kappa_{N_f}}$. This means that if a trajectory $\boldsymbol{x} \cong \boldsymbol{p}$, it must visit each tube node $\mathbb{X}_i$ of the complete path $\boldsymbol{p}$ sequentially. In addition, we can further conclude from items i)-v) that the trajectory $\boldsymbol{x}$ has to stay in each tube node $\mathbb{X}_i$ for sufficiently long time steps.*

With Definition 7.15, the satisfaction relation between a trajectory $\boldsymbol{x}$ and a tTLT can be defined as follows.

Definition 7.16. *Let there be given a trajectory $\boldsymbol{x}$ and a tTLT $\mathcal{T}_\varphi$. We say $\boldsymbol{x}$* satisfies *$\mathcal{T}_\varphi$, denoted by $\boldsymbol{x} \cong \mathcal{T}_\varphi$, if the output of Algorithm 7.2 is* true.

Definition 7.17. *(Robust satisfiable tTLT) The tTLT $\mathcal{T}_\varphi$ is called* robust satisfiable *for the system (7.1) with initial state x_0 if there exists a control policy $\boldsymbol{\nu} \in \mathcal{U}_{\geq 0}$ such that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0}$.*

Algorithm 7.2 *tTLTSatisfaction*

Input: a trajectory $\boldsymbol{x}$ and a tTLT $\mathcal{T}_\varphi$.
Output: true or false.

```
1: T_φ^c ← Compression(T_φ),
2: set all tube nodes in T_φ^c with false,
3: for each complete path of T_φ, do
4:     if x satisfies the complete path then
5:         set the leaf node of the corresponding complete path in T_φ^c with
           true,
6:     else
7:         set the leaf node of the corresponding complete path in T_φ^c with
           false,
8:     end if
9: end for
10: Backtracking(T_φ^c).
```

Algorithm 7.3 *Compression*

Input: a tTLT $\mathcal{T}_\varphi$.
Output: the compressed tree $\mathcal{T}_\varphi^c$.

```
1: for each complete path of T_φ, do
2:     for each MTF, do
3:         encode the MTF in the form of 𝕏_1 Θ_1 ... Θ_{N_f−1} 𝕏_{N_f},
4:         replace the MTF with one tube node ∪_{i=1}^{N_f} 𝕏_i,
5:     end for
6: end for
```

We further detail the *Compression* algorithm (Algorithm 7.3) and the *Backtracking* algorithm (Algorithm 7.4) in the following. Algorithm 7.3 aims at obtaining a simplified tree with Boolean operator nodes only. To do so, we first

encode each MTF in the form of $\mathbb{X}_1\Theta_1 \ldots \Theta_{N_f-1}\mathbb{X}_{N_f}$ (line 3), and then replace it with one tube node (line 4). Algorithm 7.4 takes the compressed tree $\mathcal{T}_\varphi^c$ as an input, and the output is the updated root node. This is done by updating the parent of each Boolean operator node through a bottom-up traversal. In Algorithm 7.4, $\mathrm{PA}(\Theta)$ and $\mathrm{CH}_1(\Theta), \mathrm{CH}_2(\Theta)$ represent the parent node and the two children of $\Theta \in \{\wedge, \vee\}$, respectively.

Algorithm 7.4 *Backtracking*

Input: a compressed tree $\mathcal{T}_\varphi^c$.
Output: the root node of $\mathcal{T}_\varphi^c$.

1: **for** each Boolean operator node Θ of $\mathcal{T}_\varphi^c$ through a bottom-up traversal, **do**
2: **if** $\Theta = \wedge$, **then**
3: $\mathrm{PA}(\Theta) \leftarrow \mathrm{PA}(\Theta) \vee (\mathrm{CH}_1(\Theta) \wedge \mathrm{CH}_2(\Theta))$,
4: **else**
5: $\mathrm{PA}(\Theta) \leftarrow \mathrm{PA}(\Theta) \vee (\mathrm{CH}_1(\Theta) \vee \mathrm{CH}_2(\Theta))$,
6: **end if**
7: **end for**

Example 7.2. *Let us continue with Example 7.1. The tTLT $\mathcal{T}_\varphi$ (right of Fig. 7.4) contains 2 complete paths, i.e.,*

$$\boldsymbol{p}_1 := \mathbb{X}_1 \wedge \mathbb{X}_2\mathsf{U}_{[a_1,b_1]}\mathbb{X}_4\mathsf{G}_{[a_2,b_2]}\mathbb{S}_{\mu_1}$$

and

$$\boldsymbol{p}_2 := \mathbb{X}_1 \wedge \mathbb{X}_3\mathsf{U}_{[a_3,b_3]}\mathbb{S}_{\mu_3},$$

and 3 MTFs, i.e., $\mathbb{X}_1$, $\mathbb{X}_2\mathsf{U}_{[a_1,b_1]}\mathbb{X}_4\mathsf{G}_{[a_2,b_2]}\mathbb{S}_{\mu_1}$, and $\mathbb{X}_3\mathsf{U}_{[a_3,b_3]}\mathbb{S}_{\mu_3}$. Let

$$\{t_{\kappa_1}, t_{\kappa_2}, t_{\kappa_4}, t_{\kappa_5}\}$$

be the time coding of the complete path $\boldsymbol{p}_1$, where $t_{\kappa_1}, t_{\kappa_2}, t_{\kappa_4}$, and t_{κ_5} are the activation time instants of the tube nodes $\mathbb{X}_1, \mathbb{X}_2, \mathbb{X}_4$, and $\mathbb{X}_5 := \mathbb{S}_{\mu_1}$, respectively. Then, we have according to Definition 7.15 that a trajectory $\boldsymbol{x} \cong \boldsymbol{p}_1$ if i) $t_{\kappa_1} = t_{\kappa_2}$; ii) $t_{\kappa_3} \in t_{\kappa_2} + [a_1, b_1]$; iii) $t_{\kappa_4} = \mathrm{argmax}_{t_k}\{t_k \in t_{\kappa_3} + [a_2, b_2]\}$; iv) $x_0 \in \mathbb{X}_1(t_0)$, $x_k \in \mathbb{X}_2(t_{k-\kappa_2}), \forall k \in [\kappa_2, \kappa_3]$, $x_k \in \mathbb{X}_4(t_{k-\kappa_3}), \forall k \in [\kappa_3, \kappa_4]$, and v) $x_{\kappa_4} \in \mathbb{X}_5$.

The compressed tree $\mathcal{T}_\varphi^c$ is shown in Fig. 7.5. If a trajectory $\boldsymbol{x}$ satisfies both of the complete paths $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$, the output of Algorithm 7.2 is true*, otherwise, the output is* false. □

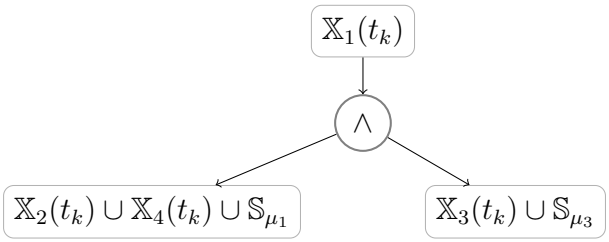

Figure 7.5: Example 7.2: compressed tree $\mathcal{T}_\varphi^c$, where $\mathcal{T}_\varphi$ is plotted in Figure 7.4.

7.5.2 Robust satisfiability check

Firstly, a sufficient condition is obtained for the robust satisfiability check of the uncertain system (7.1).

Theorem 7.3. *Let there be given the uncertain system (7.1) with initial state x_0 and an STL formula φ. Let $\mathcal{T}_\varphi$ be the tTLT corresponding to φ. Then, one has that φ is robustly satisfiable for (7.1) if the tTLT $\mathcal{T}_\varphi$ is robustly satisfiable for (7.1).*

Proof. From Definitions 7.6 and 7.17, one has that to prove Theorem 7.3, it is equivalent to prove $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0}$. Given one instance of disturbance signal $\boldsymbol{w}$, if one has $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$, then it implies $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu}} \cong \mathcal{T}_\varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi, \forall \boldsymbol{w} \in \mathcal{W}_{\geq 0}$. Therefore, it is sufficient to prove $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$.

In the following, we will first prove

i) $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Leftrightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for $\top$, predicates $\mu, \neg\mu$, and STL formulas $\mu_1 \wedge \mu_2, \mu_1 \vee \mu_2, \mu_1 \mathsf{U}_{[a,b]} \mu_2, \mathsf{G}_{[a,b]} \mu_1, \varphi_1 \wedge \varphi_2$;

ii) $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for STL formula $\varphi_1 \vee \varphi_2$;

where the sub-formulas φ_1, φ_2 are assumed to contain no Boolean operators.

Case 1: For $\top$, predicates $\mu, \neg\mu$, and $\mu_1 \wedge \mu_2, \mu_1 \vee \mu_2$, it is easy to verify that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Leftrightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$.

Case 2: $\varphi = \mu_1 \mathsf{U}_{[a,b]} \mu_2$ and $\varphi = \mathsf{G}_{[a,b]} \mu_1$. We note that the proofs of the two are similar, therefore, in the following, we only consider the case $\varphi = \mu_1 \mathsf{U}_{[a,b]} \mu_2$. The tTLT $\mathcal{T}_\varphi$ can be constructed via Algorithm 7.1, which is shown in Figure 7.6.

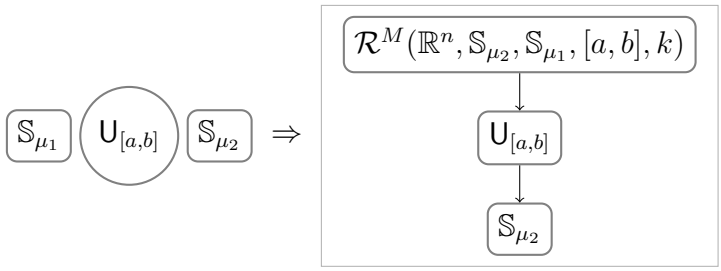

Figure 7.6: tTLTs $\mathcal{T}_{\mu_1}$, $\mathcal{T}_{\mu_2}$ and $\mathcal{T}_{\varphi}$.

Assume that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi}$, then one has from Definition 7.15 that $\exists t_{\kappa_1} \in t_0+[a,b], x_{\kappa_1} \in \mathbb{S}_{\mu_2}$ and $\forall k \in [0,\kappa_1], x_k \in \mathcal{R}^M(\mathbb{R}^n, \mathbb{S}_{\mu_2}, \mathbb{S}_{\mu_1}, [a,b], k) \subseteq \mathbb{S}_{\mu_1}$, which implies $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$. That is, $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$. Assume now that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$. Then, one has from STL semantics that i) $\exists t_{k'} \in t_0 + [a,b], x_{k'} \in \mathbb{S}_{\varphi_2}$ and ii) $\forall t_{k''} \in [t_0, t_{k'}], x_{k''} \in \mathbb{S}_{\varphi_1}$. Moreover, from Definition 7.3, one has that i) and ii) together implies $\forall t_{k''} \in [t_0, t_{k'}], x_{k''} \in \mathcal{R}^M(\mathbb{R}^n, \mathbb{S}_{\mu_2}, \mathbb{S}_{\mu_1}, [a,b], k'')$. Therefore, $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu}} \cong \mathcal{T}_{\varphi}$.

Case 3: $\varphi = \varphi_1 \wedge \varphi_2$. Assume that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi}$, then one has from Definition 7.15 that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi_1}$ and $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi_2}$. Moreover, since φ_1, φ_2 contain no Boolean operators, then one can conclude from *Case 2* that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi_i} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi_i, i = \{1,2\}$, which implies $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi_1 \wedge \varphi_2$. That is, $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$. The proof of the other direction is similar and hence omitted.

Case 4: $\varphi = \varphi_1 \vee \varphi_2$. The proof of $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ is similar to *Case 3*. The other direction does not hold because for an uncertain system, it is possible that there exists a trajectory $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}}$ such that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$, however, the initial state $x_0 \notin \mathbb{X}_{\text{root}}^{\varphi}(t_0)$ (due to item iii) of Theorem 7.1), where $\mathbb{X}_{\text{root}}^{\varphi}$ denotes the root node of $\mathcal{T}_{\varphi}$. In this case, $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}}$ does not satisfy $\mathcal{T}_{\varphi}$.

The proof of $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi} \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for STL formulas $\varphi_1 \mathsf{U}_{[a,b]} \varphi_2$, $\mathsf{G}_{[a,b]}\varphi_1$, $\varphi_1 \wedge \varphi_2$, and $\varphi_1 \vee \varphi_2$ (here, no assumption on φ_1, φ_2) can be completed inductively by combining *Cases 1, 2, 3* and *4*. Therefore, the conclusion follows. □

When the deterministic system (7.2) is considered, the above condition becomes a necessary and sufficient condition, as shown in the following corollary.

Corollary 7.2. *Let there be given the deterministic system (7.2) with initial state x_0 and an STL formula φ. Let $\mathcal{T}_{\varphi}$ be the tTLT corresponding to φ. Then, one has*

that φ is satisfiable for (7.2) if and only if the tTLT $\mathcal{T}_\varphi$ is satisfiable for (7.2).

Proof. For the deterministic system (7.2), one has from the proof of Theorem 7.3 that

i) $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Leftrightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for $\top$, predicates $\mu, \neg\mu$, and STL formulas $\mu_1 \wedge \mu_2, \mu_1 \vee \mu_2, \mu_1 \mathsf{U}_{[a,b]} \mu_2, \mathsf{G}_{[a,b]}\mu_1, \varphi_1 \wedge \varphi_2$

holds. Next, we will prove

iii) $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Leftrightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for STL formula $\varphi_1 \vee \varphi_2$,

where the sub-formulas φ_1, φ_2 are assumed to contain no Boolean operators.

Let $\varphi = \varphi_1 \vee \varphi_2$. The proof of $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ is given in Theorem 7.3. Assume now that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$, then one has from STL semantics that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi_1$ or $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi_2$. Since φ_1, φ_2 contain no Boolean operators, then one can conclude from item i) that $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi_i \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_{\varphi_i}, i = \{1, 2\}$. Moreover, one has from Corollary 7.1 that $\mathbb{S}_{\varphi_1 \vee \varphi_2}(t_k) = \mathbb{S}_{\varphi_1}(t_k) \cup \mathbb{S}_{\varphi_2}(t_k)$. Therefore, $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi \Rightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi$.

The proof of $\boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \cong \mathcal{T}_\varphi \Leftrightarrow \boldsymbol{x}_{x_0}^{\boldsymbol{\nu},\boldsymbol{w}} \vDash \varphi$ for STL formulas $\varphi_1 \mathsf{U}_{[a,b]} \varphi_2$, $\mathsf{G}_{[a,b]}\varphi_1$, $\varphi_1 \wedge \varphi_2$, and $\varphi_1 \vee \varphi_2$ (here, no assumption on φ_1, φ_2) can be completed inductively by combining items i) and iii). Therefore, the conclusion follows. □

Given a tTLT $\mathcal{T}_\varphi$, denote by $\mathbb{X}_{\text{root}}^{\varphi}$ the root node of $\mathcal{T}_\varphi$. The conditions given in Theorems 7.3 and Corollary 7.2 are in general difficult to check, therefore, two necessary conditions are given in the following propositions.

Proposition 7.1. *Let there be given the system (7.1) with initial state x_0 and an STL formula φ. Let $\mathcal{T}_\varphi$ be the tTLT corresponding to φ. Then, $\mathcal{T}_\varphi$ is robustly satisfiable for (7.1) only if $x_0 \in \mathbb{X}_{root}^{\varphi}(t_0)$.*

Proof. It follows from Definitions 7.15 and 7.17 that if $\boldsymbol{x} \cong \mathcal{T}_\varphi$, it must have $x_0 \in \mathbb{X}_{\text{root}}^{\varphi}(t_0)$. □

Similar necessary condition also holds for the deterministic systems.

Proposition 7.2. *Let there be given the deterministic system (7.2) with initial state x_0 and an STL formula φ. Let $\mathcal{T}_\varphi$ be the tTLT corresponding to φ. Then, φ is satisfiable for (7.2) only if $x_0 \in \mathbb{X}_{root}^{\varphi}(t_0)$.*

Proof. Consider a predicate μ and STL formulas φ_1, φ_2. Define

- $\hat{\mathbb{S}}_{\mu} = \{x \in \mathbb{R}^n : g_{\mu}(x) \geq 0\}$,
- $\hat{\mathbb{S}}_{\neg\mu} = \overline{\hat{\mathbb{S}}_{\mu}}$,
- $\hat{\mathbb{S}}_{\varphi_1 \wedge \varphi_2}(t_k) = \hat{\mathbb{S}}_{\varphi_1}(t_k) \cap \hat{\mathbb{S}}_{\varphi_2}(t_k)$,
- $\hat{\mathbb{S}}_{\varphi_1 \vee \varphi_2}(t_k) = \hat{\mathbb{S}}_{\varphi_1}(t_k) \cup \hat{\mathbb{S}}_{\varphi_2}(t_k)$,
- $\hat{\mathbb{S}}_{\varphi_1 \mathsf{U}_{[a,b]} \varphi_2}(t_k) = \mathcal{R}^M(\mathbb{R}^n, \hat{\mathbb{S}}_{\varphi_2}(t_0), \hat{\mathbb{S}}_{\varphi_1}(t_0), [a,b], k)$, and
- $\hat{\mathbb{S}}_{\mathsf{G}_{[a,b]}\varphi}(t_k) = \overline{\mathcal{R}^m(\mathbb{R}^n, \overline{\hat{\mathbb{S}}_{\varphi}(t_0)}, [a,b], k)}$.

From the construction of tTLT (Algorithm 7.1), one has that $\hat{\mathbb{S}}_{\varphi}(t_0) = \mathbb{X}^{\varphi}_{\text{root}}(t_0)$. In addition, one can conclude from Corollary 7.1 that $\mathbb{S}_{\varphi} \subseteq \hat{\mathbb{S}}_{\varphi}(t_0)$, where $\mathbb{S}_{\varphi}$ defined in (7.5) is the set of initial states from which φ is satisfiable. Therefore, $\mathbb{S}_{\varphi} \subseteq \mathbb{X}^{\varphi}_{\text{root}}(t_0)$. The conclusion follows. □

7.6 Online Control Synthesis

This section concerns online control synthesis as defined by Problem 7.2. From Theorems 7.3 (Corollary 7.2), one can see that to guarantee the satisfaction of the STL formula φ, it is sufficient to find a control policy $\boldsymbol{\nu}$ that guarantees the (robust) satisfaction of the corresponding tTLT $\mathcal{T}_{\varphi}$. In the following, the control synthesis algorithms are designed such that the tTLT $\mathcal{T}_{\varphi}$ is satisfied based on Definitions 7.15 and 7.16.

7.6.1 Definitions and notations

Before proceeding, the following definitions and notations are needed.

Definition 7.18. *The time horizon $|\Theta|$ of an STL operator Θ is defined as*

$$|\Theta| = \begin{cases} 0, & \text{if } \Theta = \{\wedge, \vee\}, \\ \hat{b}, & \text{if } \Theta \in \{\mathsf{U}_{[a,b]}, \mathsf{G}_{[a,b]}\}, \end{cases} \tag{7.8}$$

where $\hat{b} = \operatorname{argmax}_{t_k}\{a \leq t_k \leq b\}$.

Definition 7.19. *A fragment of the complete path of a tTLT is called* a Boolean fragment *if it starts and ends with a tube node and contains only Boolean operator nodes. We say a tube node $\mathbb{X}_j$ is* reachable from $\mathbb{X}_i$ by a Boolean fragment *if there exists a Boolean fragment that starts with $\mathbb{X}_i$ and ends with $\mathbb{X}_j$.*

Definition 7.20. *If each node of a tree is either a set node that is a subset of U or an operator node that belongs to $\{\wedge, \vee, \mathsf{U}_I, \mathsf{G}_I\}$, then the tree is called a* control tree.

Each tube node $\mathbb{X}_i$ of the tTLT $\mathcal{T}_\varphi$ is characterized by the following two parameters:

- $t_a(\mathbb{X}_i)$: the activation time of $\mathbb{X}_i$,
- $t_h(\mathbb{X}_i)$: the time horizon of $\mathbb{X}_i$, *i.e.,* the time that $\mathbb{X}_i$ is deactivated.

Denote by $\mathcal{T}_\varphi(t_k)$ the resulting tree of $\mathcal{T}_\varphi$ at time instant t_k. It is obtained by fixing the value of each tube node $\mathbb{X}_i$ according to the activation time $t_a(\mathbb{X}_i)$ (*i.e.,* $\mathcal{T}_\varphi(t_k)$ contains either set nodes or operator nodes). Let $S_i(t_k)$ be the i-th set node of $\mathcal{T}_\varphi(t_k)$, where $S_i(t_k)$ corresponds to the tube node $\mathbb{X}_i$. The relationship between $S_i(t_k)$ and $\mathbb{X}_i$ can be described as follows:

$$S_i(t_k) = \begin{cases} \mathbb{X}_i(t_0), & \text{if } t_k \leq t_a(\mathbb{X}_i), \\ \mathbb{X}_i(t_k - t_a(\mathbb{X}_i)), & \text{if } t_k > t_a(\mathbb{X}_i). \end{cases} \tag{7.9}$$

Moreover, one has that

$$t_a(S_i(t_k)) = t_a(\mathbb{X}_i), t_h(S_i(t_k)) = t_h(\mathbb{X}_i), \forall k \geq 0.$$

At each time instant t_k, $\mathcal{T}_\varphi(t_k)$ is characterized by

- $P(t_k)$: the set which collects all the set nodes of $\mathcal{T}_\varphi(t_k)$, *i.e.,* $P(t_k) = \cup_i S_i(t_k)$,
- Θ: the set which collects all the operator nodes of $\mathcal{T}_\varphi(t_k)$, which is time invariant.

For a node $N_i(t_k) \in P(t_k) \cup \Theta$, define

- $\mathrm{CH}(N_i(t_k))$: the set of children of node $N_i(t_k)$,
- $\mathrm{PA}(N_i(t_k))$: the set of parents of node $N_i(t_k)$,

- $\mathsf{Post}(N_i(t_k)) := \mathrm{CH}(\mathrm{CH}(N_i(t_k)))$,
- $\mathsf{Pre}(N_i(t_k)) := \mathrm{PA}(\mathrm{PA}(N_i(t_k)))$.

Given a state-time pair (x_k, t_k), define $L : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to 2^{P(t_k)}$ as the labelling function, given by

$$L(x_k, t_k) = \{S_i(t_k) \in P(t_k) : x_k \in S_i(t_k), t_k \leq t_h(S_i(t_k))\}, \tag{7.10}$$

which maps (x_k, t_k) to a subset of $P(t_k)$. Moreover, define the function $B : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to 2^{P(t_k)}$, which maps (x_k, t_k) to a set of valid set nodes in $P(t_k)$. Function $L(x_k, t_k)$ computes the subset of set nodes of $P(t_k)$ that contains x_k at time t_k (without the consideration of history trajectory) while function $B(x_k, t_k)$ is further introduced to capture the fact that given the history trajectory, not all set nodes in $L(x_k, t_k)$ are valid at time t_k. A rule for determining $B(x_k, t_k)$ given $L(x_k, t_k)$ is detailed in Algorithm 7.7 in the next subsection.

7.6.2 Online control synthesis

In the following, we will first present the online control synthesis algorithm (and its sub-algorithms), and then an example is given to further explain how each sub-algorithm works.

The online control synthesis algorithm is outlined in Algorithm 7.5. Before implementation, an initialization process (line 1) is required, which is outlined in Algorithm 7.6. Here, t_a and t_h are two functions that map each tube node $\mathbb{X}_i$ to its activation time and time horizon, respectively. if $t_a(\mathbb{X}_i)$ or $t_h(\mathbb{X}_i)$ is unknown for $\mathbb{X}_i$, its value will be set as $\bowtie$. Then, at each time instant t_k, a feasible control set $\mathbb{U}(x_k, t_k)$ is synthesized (lines 2-11). This process contains the following steps: 1) find the subset of set nodes in $P(t_k)$ that are valid at time t_k, *i.e.,* $B(x_k, t_k)$, via Algorithm 7.7 (line 2); 2) determine the activation time of $\mathbb{X}_i$, whose corresponding set node $S_i(t_k) \in B(x_k, t_k)$ (if $t_a(\mathbb{X}_i)$ is unknown, *i.e.,* being visited for the first time, it is set as t_k; otherwise, *i.e.,* being visited before, it is unchanged) (lines 3-7); 3) calculate $\mathcal{T}_\varphi(t_{k+1})$ via Algorithm 7.8 (line 8); 4) build a control tree $\mathcal{T}_u(t_k)$ (Definition 7.20) via Algorithm 7.9 (line 9), compress it via Algorithm 7.3 (line 10), and then the feasible control set $\mathbb{U}(x_k, t_k)$ is given by backtracking the compressed control tree $\mathcal{T}_u^c(t_k)$ via Algorithm 7.10 (line 11). If the obtained feasible control set $\mathbb{U}(x_k, t_k) = \emptyset$, the control synthesis process stops and returns NExis (lines 12-13); otherwise, the

Algorithm 7.5 *onlineControlSynthesis*

Input: The tTLT $\mathcal{T}_\varphi$ and (x_0, t_0).
Output: NExis or $(\boldsymbol{\nu}, \boldsymbol{x})$ with $\boldsymbol{\nu} = \nu_0\nu_1 \ldots \nu_k \ldots$ and $\boldsymbol{x} = x_0x_1 \ldots x_k \ldots$.

1: $(t_a, t_h, \texttt{Post}(B(x_{-1}, t_{-1}))) \leftarrow$ *initialization*$(\mathcal{T}_\varphi)$,
2: $B(x_k, t_k) \leftarrow$ *trackingSetNode*$(\texttt{Post}(B(x_{k-1}, t_{k-1})))$,
3: **for** each $S_i(t_k) \in B(x_k, t_k)$, **do**
4: **if** $t_a(S_i(t_k)) = \bowtie$, **then**
5: $t_a(\mathbb{X}_i) \leftarrow t_k$,
6: **end if**
7: **end for**
8: $\mathcal{T}_\varphi(t_{k+1}) \leftarrow$ *updatetTLT*$(\mathcal{T}_\varphi(t_k), t_a, B(x_k, t_k))$,
9: $\mathcal{T}_u(t_k) \leftarrow$ *buildControlTree*$(\mathcal{T}_\varphi(t_k), B(x_k, t_k), \mathcal{T}_\varphi(t_{k+1}))$,
10: $\mathcal{T}_u^c(t_k) \leftarrow$ *Compression*$(\mathcal{T}_u(t_k))$,
11: $\mathbb{U}(x_k, t_k) \leftarrow$ *Backtracking*$^*(\mathcal{T}_u^c)$,
12: **if** $\mathbb{U}(x_k, t_k) = \emptyset$, **then**
13: stop and return NExis,
14: **else**
15: choose $\nu_k \in \mathbb{U}(x_k, t_k)$,
16: implement ν_k and measure x_{k+1},
17: $\texttt{Post}(B(x_k, t_k)) \leftarrow$ *postSet*$(B(x_k, t_k), t_a, \mathcal{T}_\varphi(t_{k+1}))$,
18: update $k = k + 1$ and go to line 2.
19: **end if**

control input ν_k can be chosen as any element of $\mathbb{U}(x_k, t_k)$ (one example is to choose ν_k as $\min_{\nu_k \in \mathbb{U}(x_k, t_k)}\{\|\nu_k\|\}$) (line 15). Then, we implement the chosen ν_k, measure x_{k+1} (line 16), and finally compute the subset of set nodes that are possibly available at the next time instant t_{k+1}, *i.e.,* $\texttt{Post}(B(x_k, t_k))$, via Algorithm 7.11 (line 17).

Algorithm 7.6 *initialization*

Input: The tTLT $\mathcal{T}_\varphi$.
Output: $t_a, t_h, \texttt{Post}(B(x_{-1}, t_{-1}))$.

$t_a(\mathbb{X}^\varphi_{\text{root}}) \leftarrow t_0, t_h(\mathbb{X}^\varphi_{\text{root}}) \leftarrow t_0 + |\text{CH}(\mathbb{X}^\varphi_{\text{root}})|$,
for each non-root and non-leaf tube node $\mathbb{X}_i$ through a top-down traversal, **do**
    $t_a(\mathbb{X}_i) \leftarrow \bowtie, t_h(\mathbb{X}_i) \leftarrow t_h(\mathsf{Pre}(\mathbb{X}_i) + |\text{CH}(\mathbb{X}_i)|$,
end for
for each leaf node $\mathbb{X}_i$, **do**
    $t_a(\mathbb{X}_i) \leftarrow \bowtie, t_h(\mathbb{X}_i) \leftarrow \infty$,
end for
$\texttt{Post}(B(x_{-1}, t_{-1})) \leftarrow \mathbb{X}^\varphi_{\text{root}}(t_0)$,
for each $\mathbb{X}_j$ that is reachable from $\mathbb{X}^\varphi_{\text{root}}$ by a Boolean fragment (see Definition 7.19), **do**
    $\texttt{Post}(B(x_{-1}, t_{-1})) \leftarrow \texttt{Post}(B(x_{-1}, t_{-1})) \cup \mathbb{X}_j(t_0)$,
    $t_a(\mathbb{X}_j) \leftarrow t_0$,
end for

Algorithm 7.7 *trackingSetNode*

Input: $\texttt{Post}(B(x_{k-1}, t_{k-1}))$.
Output: $B(x_k, t_k)$.

Compute $L(x_k, t_k)$ according to (7.10),
$B(x_k, t_k) \leftarrow L(x_k, t_k) \cap \texttt{Post}(B(x_{k-1}, t_{k-1}))$,
for each $S_i(t_k) \in B(x_k, t_k)$ **do**,
    if $\exists S_j(t_k) \in B(x_k, t_k)$ s.t. $S_j(t_k) = \mathsf{Post}(S_i(t_k))$, **then**
        $B(x_k, t_k) \leftarrow B(x_k, t_k) \setminus S_i(t_k)$,
    end if
end for

Algorithm 7.8 *updatetTLT*

Input: $\mathcal{T}_\varphi(t_k)$, t_a and $B(x_k, t_k)$.
Output: $\mathcal{T}_\varphi(t_{k+1})$.
1: **for** each set node $S_i(t_k)$ of $\mathcal{T}_\varphi(t_k)$, **do**
2: **if** $S_i(t_k) \in B(x_k, t_k) \wedge t_a(S_i(t_k)) + |\mathrm{CH}(S_i(t_k))| \geq t_{k+1}$, **then**
3: $S_i(t_{k+1}) \leftarrow \mathbb{X}_i(t_{k+1} - t_a(S_i(t_k)))$,
4: **else**
5: $S_i(t_{k+1}) \leftarrow S_i(t_k)$,
6: **end if**
7: **end for**

We further detail the Algorithms 7.6-7.11 in the following.

- Algorithm 7.6 calculates the functions t_a and t_h (lines 1-7) and $\texttt{Post}(B(x_{-1}, t_{-1}))$ (lines 8-12).

- Algorithm 7.7 outlines the procedure of finding the subset of set nodes in $P(t_k)$ that are valid at time t_k, *i.e.,* $B(x_k, t_k)$. This is the most important step of the control synthesis, and it relates to Algorithm 7.11 *postSet*. Firstly, one needs to compute the subset of set nodes of $P(t_k)$ that contains x_k at time t_k, *i.e.,* $L(x_k, t_k)$ (line 1). Then, one has from Definition 7.15 that if a trajectory $\boldsymbol{x}$ satisfies one complete path of the tTLT, it must i) visit each tube node of the complete path sequentially and ii) stay in each tube node for sufficiently long time steps (Remark 7.2). Based on these two requirements, Algorithm 7.11 is designed to predict the subset of set nodes that are possibly available at the next time instant, *i.e.,* $\texttt{Post}(B(x_{k-1}, t_{k-1}))$. $B(x_k, t_k)$ must belong to $L(x_k, t_k)$ and $\texttt{Post}(B(x_{k-1}, t_{k-1}))$ at the same time. Therefore, we let $B(x_k, t_k) \leftarrow L(x_k, t_k) \cap \texttt{Post}(B(x_{k-1}, t_{k-1}))$ (line 2). The rest of Algorithm 7.7 (lines 3-7) is to guarantee that $B(x_k, t_k)$ contains at most one set node for each complete path of $\mathcal{T}_\varphi(t_k)$.

- Algorithm 7.8 outlines the procedure of calculating $\mathcal{T}_\varphi(t_{k+1})$, given $\mathcal{T}_\varphi(t_k)$, t_a and $B(x_k, t_k)$. It is designed based on (7.9).

- Algorithm 7.9 outlines the procedure of building a control tree $\mathcal{T}_u(t_k)$, which is then used for control set synthesis. It is initialized as $\mathcal{T}_\varphi(t_k)$ (line 1). Then, for those set nodes $S_i(t_k)$ that belongs to $B(x_k, t_k)$, it is

replaced with the feasible control set (lines 2-8), otherwise, it is replaced with $\emptyset$ (lines 9-11).

- Algorithm 7.10 is similar to Algorithm 7.4, which outlines the procedure of backtracking a compressed tree.
- Algorithm 7.11 outlines the procedure of finding the subset of set nodes that are possibly available at the next time instant t_{k+1} given $B(x_k, t_k)$, t_a and $\mathcal{T}_\varphi(t_{k+1})$. It is designed based on Definition 7.15, where the three cases (lines 4-8, 9-12, 13-16) correspond to items i)-iii) of Definition 7.15, respectively. It guarantees that the resulting trajectory visits each tube node of $\mathcal{T}_\varphi$ sequentially and stays in each tube node for sufficiently long time steps (as we discussed in Algorithm 7.7).

Algorithm 7.9 *buildControlTree*

Input: $\mathcal{T}_\varphi(t_k)$, $B(x_k, t_k)$, and $\mathcal{T}_\varphi(t_{k+1})$.
Output: A control tree $\mathcal{T}_u(t_k)$.

1: Initialize $\mathcal{T}_u(t_k)$ as $\mathcal{T}_\varphi(t_k)$,
2: **for** each $S_i(t_k) \in B(x_k, t_k)$ **do**
3: **if** $S_i(t_k)$ is a leaf node **then**,
4: $S_i(t_k) \leftarrow \mathbb{U}(S_i(t_k)) := U$,
5: **else**
6: $S_i(t_k) \leftarrow \mathbb{U}(S_i(t_k)) := \{u_k \in U : f_k(x_k, u_k, w_k) \in S_i(t_{k+1}), \forall w_k \in W\}$,
7: **end if**
8: **end for**
9: **for** each $S_i(t_k) \notin B(x_k, t_k)$ **do**
10: $S_i(t_k) \leftarrow \emptyset$,
11: **end for**

Algorithm 7.10 *Backtracking**

Input: a compressed tree $\mathcal{T}_u^c(t_k)$.
Output: the root node of $\mathcal{T}_u^c(t_k)$.

for each Boolean operator node Θ of $\mathcal{T}_u^c(t_k)$ through a bottom-up traversal, **do**
  if $\Theta = \wedge$, **then**
    $\mathrm{PA}(\Theta) \leftarrow \mathrm{PA}(\Theta) \cup (\mathrm{CH}_1(\Theta) \cap \mathrm{CH}_2(\Theta))$,
  else
    $\mathrm{PA}(\Theta) \leftarrow \mathrm{PA}(\Theta) \cup (\mathrm{CH}_1(\Theta) \cup \mathrm{CH}_2(\Theta))$,
  end if
end for

Algorithm 7.11 *postSet*

Input: $B(x_k, t_k), t_a$ and $\mathcal{T}_\varphi(t_{k+1})$.
Output: $\mathtt{Post}(B(x_k, t_k))$.

Initialize $\mathtt{Post}(S_i(t_k)) = \emptyset, \forall S_i(t_k) \in B(x_k, t_k)$.
for each $S_i(t_k) \in B(x_k, t_k)$, **do**
  switch the children of $S_i(t_k)$ **do**
    case $\mathrm{CH}(S_i(t_k)) \in \{\wedge, \vee\}$,
      $\mathtt{Post}(S_i(t_k)) \leftarrow S_i(t_{k+1})$,
      for each $S_j(t_k)$ that is reachable from $S_i(t_k)$ by a Boolean fragment, **do**
        $\mathtt{Post}(S_i(t_k)) \leftarrow \mathtt{Post}(S_i(t_k)) \cup S_j(t_{k+1})$,
      end for
    case $\mathrm{CH}(S_i(t_k)) \in \{\mathsf{U}_{[a,b]}\}$,
      if $t_k > t_a(\mathsf{Pre}(S_i(t_k)) + a\}$, **then**
        $\mathtt{Post}(S_i(t_k)) \leftarrow S_i(t_{k+1}) \cup \mathsf{Post}(S_i(t_{k+1}))$,
      end if
    case $\mathrm{CH}(S_i(t_k)) \in \{\mathsf{G}_{[a,b]}\}$,
      if $t_k > t_a(\mathsf{Pre}(S_i(t_k)) + b\}$, **then**
        $\mathtt{Post}(S_i(t_k)) \leftarrow S_i(t_{k+1}) \cup \mathsf{Post}(S_i(t_{k+1}))$,
      end if
end for

Next, an example is given to illustrate one iteration of the control synthesis algorithm (Algorithm 7.5).

Example 7.3. *Consider the single integrator model $\dot{x} = u + w$ with a sampling period of 1 second, then the resulting discrete-time system is given by*

$$x_{k+1} = x_k + u_k + w_k,$$

where $x_k \in \mathbb{R}^2, u_k \in U := \{u : ||u|| \leq 1\} \subset \mathbb{R}^2, w_k \in W := \{w : ||w|| \leq 0.1\} \subset \mathbb{R}^2, \forall k \in \mathbb{N}$. The task specification φ is given in Example 7.1, i.e., $\varphi = \mathsf{F}_{[a_1,b_1]}\mathsf{G}_{[a_2,b_2]}\mu_1 \wedge \mu_2\mathsf{U}_{[a_3,b_3]}\mu_3$, where $[a_1, b_1] = [5, 10],\ [a_2, b_2] = [0, 10],\ [a_3, b_3] = [0, 8]$, $g_{\mu_1}(x) = 1 - \|x\|,\ g_{\mu_2}(x) = 5 - \|x - [4, 4]^T\|$ and $g_{\mu_3}(x) = 1 - \|x - [3, 5]^T\|$. Then, one has

$$\begin{aligned}
&\mathbb{S}_{\mu_1} = \{x_0 : \|x_0\| \leq 1\},\\
&\mathbb{S}_{\mu_2} = \{x_0 : \|x_0 - [4, 4]^T\| \leq 5\},\ \textit{and}\\
&\mathbb{S}_{\mu_3} = \{x_0 : \|x_0 - [3, 5]^T\| \leq 1\}.
\end{aligned}$$

The tTLT that corresponds to φ is plotted in Figure 7.4. Using Definitions 7.3 and 7.4, one can calculate that

$$\begin{aligned}
&\mathbb{X}_4(t_k) = \{x_k : \|x_k\| \leq 0.9\},\\
&\mathbb{X}_3(t_k) = \{x_k : \|x_k - [3, 5]^T\| \leq 8.1 - k \wedge \|x_k - [4, 4]^T\| \leq 5\},\\
&\mathbb{X}_2(t_k) = \{x_k : \|x_k\| \leq 9.9 - k\},\ \textit{and}\\
&\mathbb{X}_1(t_k) = \mathbb{X}_2(t_k) \cap \mathbb{X}_3(t_k).
\end{aligned}$$

The initial state $x_0 = [0.5, 0.8]^T$, for which $x_0 \in \mathbb{X}^{\varphi}_{root}(t_0)$. Firstly, an initialization process is required, and one can get from Algorithm 7.6 that

$$\begin{aligned}
&t_h(\mathbb{X}_1) = 0, t_h(\mathbb{X}_2) = 10, t_h(\mathbb{X}_3) = 8,\\
&t_h(\mathbb{X}_4) = 20, t_h(\mathbb{S}_{\mu_1}) = \infty, t_h(\mathbb{S}_{\mu_3}) = \infty,
\end{aligned}$$

and

$$\texttt{Post}(B(x_{-1}, t_{-1})) = \{\mathbb{X}_1(t_0), \mathbb{X}_2(t_0), \mathbb{X}_3(t_0)\}.$$

Now, let us see how the feasible control set $\mathbb{U}(x_0, t_0)$ is synthesized at time instant t_0.

1) Find $B(x_0, t_0)$ via Algorithm 7.7. First, $L(x_0, t_0)$ is computed according to (7.10),

$$L(x_0, t_0) = \{\mathbb{X}_1(t_0), \mathbb{X}_2(t_0), \mathbb{X}_3(t_0), \mathbb{X}_4(t_0), \mathbb{S}_{\mu_1}\}.$$

Then, after running lines 2-7, one has

$$B(x_0, t_0) = \{S_2(t_0), S_3(t_0)\}.$$

2) Determine the activation time. Initially, both $t_a(\mathbb{X}_2)$ *and* $t_a(\mathbb{X}_3)$ *are unknown, therefore,* $t_a(\mathbb{X}_2) = t_a(\mathbb{X}_3) = t_0$.

3) Update the TLT (thus obtain $\mathcal{T}_\varphi(t_1)$*) via Algorithm 7.8. The output* $\mathcal{T}_\varphi(t_1)$ *is given by*

$$S_1(t_1) = \mathbb{X}_1(t_0), S_2(t_1) = \mathbb{X}_2(t_1),$$
$$S_3(t_1) = \mathbb{X}_3(t_1), S_4(t_1) = \mathbb{X}_4(t_0),$$

and the leaf nodes $\mathbb{S}_{\mu_1}$ *and* $\mathbb{S}_{\mu_3}$ *are unchanged.*

4) Build the control tree $\mathcal{T}_u(t_0)$*, compress it to obtain* $\mathcal{T}_u^c(t_0)$*, and then get* $\mathbb{U}(x_0, t_0)$*. This process is illustrated in Figure 7.7, and* $\mathbb{U}(x_0, t_0) = \mathbb{U}(S_2(t_0)) \cap \mathbb{U}(S_3(t_0))$.

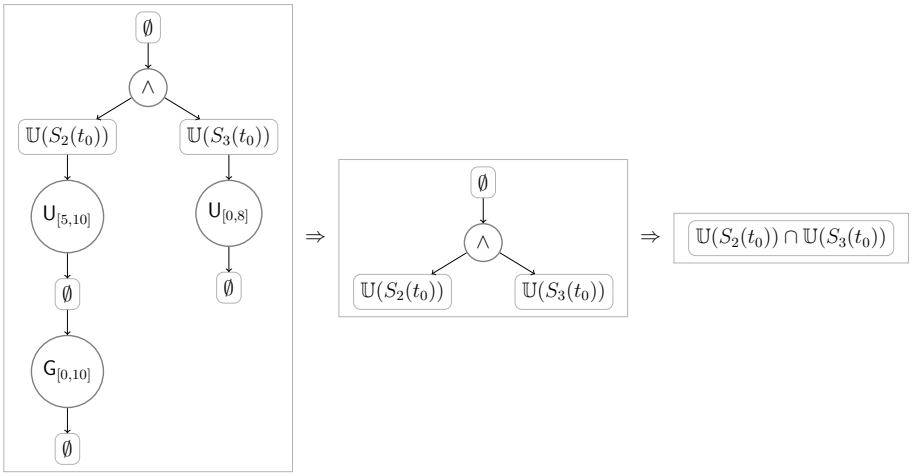

Figure 7.7: Left: $\mathcal{T}_u(t_0)$, Middle: $\mathcal{T}_u^c(t_0)$, Right: root node of $\mathcal{T}_u^c(t_0)$ after implementing Algorithm 7.10, where $\mathbb{U}(S_2(t_0)) = U = \{u : ||u|| \leq 1\}$ and $\mathbb{U}(S_3(t_0)) = U \cap \{u : \|u - [3.4, 3.1]^T\| \leq 5\}$.

Since $\mathbb{U}(x_0, t_0) \neq \emptyset$*, the online control synthesis continues, and we can further compute* $\texttt{Post}(B(x_0, t_0))$ *via Algorithm 7.11, which gives*

$$\texttt{Post}(B(x_0, t_0)) = \{S_2(t_1), S_3(t_1), \mathbb{S}_{\mu_3}\}.$$

The following theorem and corollary show the applicability and correctness of Algorithm 7.5.

Theorem 7.4. *Consider the system (7.1) with initial state x_0 and an STL formula φ. Assume that φ is robustly satisfiable for (7.1) and $x_0 \in \mathcal{T}^{\varphi}_{root}(t_0)$. Then, by implementing the online control synthesis algorithm (Algorithm 7.5), one can guarantee that*

i) the control set $\mathbb{U}(x_k, t_k)$ is nonempty for all $k \in \mathbb{N}$, and

ii) the resulting trajectory $\boldsymbol{x} \vDash \varphi$.

Proof. The proof follows from the construction of tTLT and Algorithms 7.5-7.11. The existence of a controller ν_k at each time step t_k, is guaranteed by the definition of maximal and minimal reachable sets (Definitions 7.3 and 7.4), and the construction of tTLT (Lemma 7.1, Theorem 7.1 and Algorithm 7.1). Moreover, the design of Algorithms 7.5-7.11 guarantees that the resulting trajectory $\boldsymbol{x}$ satisfies the tTLT $\mathcal{T}_{\varphi}$, *i.e.,* $\boldsymbol{x} \cong \mathcal{T}_{\varphi}$, which implies $\boldsymbol{x} \vDash \varphi$ as proven in Theorem 7.3. □

Corollary 7.3. *Consider the deterministic system (7.2) with initial state x_0 and an STL formula φ. Assume that φ is satisfiable for (7.2). Then, by implementing the online control synthesis algorithm (Algorithm 7.5), one can guarantee that*

i) the control set $\mathbb{U}(x_k, t_k)$ is nonempty for all $k \in \mathbb{N}$, and

ii) the resulting trajectory $\boldsymbol{x} \vDash \varphi$.

Remark 7.3. *It can be concluded from Theorem 7.4 and Corollary 7.3 that the online control synthesis algorithm (Algorithm 7.5) is sound for uncertain systems, and both sound and complete for deterministic systems.*

Remark 7.4. *The construction of tTLT relies on the computation of backward reachable tubes. It can be performed offline in many applications and has been widely studied in the existing literature [129], [130]. In addition, computational tools have also been developed for different kinds of systems, e.g., the Hamilton-Jacobi toolbox [131]. On the other hand, although the exact computation of backward reachable set/tube is in general nontrivial for high-dimensional nonlinear systems, efficient algorithms exist for linear systems with polygonal input and disturbance sets [129].*

Remark 7.5. *The online control synthesis algorithm (Algorithm 7.5) contains 7 sub-algorithms, i.e., Algorithm 7.3 and Algorithms 7.6-7.11. The computational complexity is determined by Algorithm 7.9, in which one-step feasible control sets*

need to be computed. The computational complexity of Algorithms 7.3, 7.6, 7.7, 7.8, 7.10, 7.11 is $\mathcal{O}(1)$. Note that in Algorithm 7.8, the computation of reachable sets, which is required for set node update, is done offline when constructing the tTLT.

Remark 7.6. *Different from the mixed-integer programming formulation for STL control synthesis [26], [27], where an entire control policy has to be synthesized at each time step, the control synthesis in our work is reactive in the sense that only the control input at the current time step is generated at each time step. In [28], a robust model predictive approach is proposed to control discrete-time linear systems with additive bounded disturbances. In our work, we consider general uncertain discrete-time systems and the recursive feasibility is guaranteed when the STL formula φ is robustly satisfiable and $x_0 \in \mathcal{T}^{\varphi}_{root}(t_0)$. Moreover, the framework proposed is also capable of dealing with unbounded STL formulas (as opposed to [26]–[28]).*

7.7 Example

In this section, a simulation example illustrating the theoretical results is provided. This example will specify an overtaking task as an STL formula and then show how to synthesize overtaking controller with safety guarantee. As shown in Figure 7.8, we consider a scenario where an automated vehicle Veh_1 plans to move to a target set $\mathbb{S}_{\mu_1}$ within 80 seconds. Since there is a broken vehicle Veh_2 in front of Veh_1 and there is another vehicle Veh_3 that moves in an opposite direction in the other lane, Veh_1 must overtake Veh_2 for reaching $\mathbb{S}_{\mu_1}$ and avoid Veh_3 for safety.

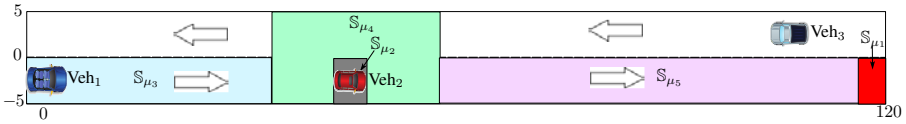

Figure 7.8: Scenario illustration: an automated vehicle plans to reach a target set $\mathbb{S}_{\mu_1}$ while overtaking a broken vehicle Veh_2 in front of it in the same lane and avoiding Veh_3 moving in an opposite direction in the other lane.

We describe the dynamics of the vehicle Veh_1 as in [132]:

$$x_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 & \delta \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{A} x_k + \underbrace{\begin{bmatrix} 0 & 0 \\ \delta & 0 \\ 0 & \delta \end{bmatrix}}_{B} u_k + w_k, \tag{7.11}$$

where $x_k = [p^x(k), p^y(k), v^x(k)]^T$, $u_k = [v^y(k), a^x(k)]^T$, and δ is the sampling period. The working space is $X = \{z \in \mathbb{R}^3 \mid [0, -5, -3]^T \leq z \leq [120, 5, 3]^T\}$, the control constraint set is $U = \{z \in \mathbb{R}^2 \mid [-1, -1]^T \leq z \leq [1, 1]^T\}$, the disturbance set is $W = \{z \in \mathbb{R}^3 \mid [-0.05, -0.05, -0.05]^T \leq z \leq [0.05, 0.05, 0.05]^T\}$, and the target region is $\mathbb{S}_{\mu_1} = \{z \in \mathbb{R}^2 \mid [115, -5, 0.5]^T \leq z \leq [120, 0, 0.5]^T\}$.

We use $\mathbb{S}_{\mu_2} = \{z \in \mathbb{R}^3 \mid [45, -5, -\infty]^T \leq z \leq [50, 0, \infty]^T\}$ to denote the state set that contains the occupancy of Veh_2. We describe the dynamics of the vehicle Veh_3 as

$$\bar{x}_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\bar{A}} x_k + \underbrace{\begin{bmatrix} \delta & 0 \\ 0 & \delta \end{bmatrix}}_{\bar{B}} \bar{u}_k, \tag{7.12}$$

where $x_k = [\bar{p}^x(k), \bar{p}^y(k)]^T$, $\bar{u}_k = [\bar{v}^x(k), \bar{v}^y(k)]^T$, We assume that it moves at a constant velocity $\bar{u}_k = [\bar{v}^x, 0]^T$. The initial state of $\mathcal{V}_3$ is $\bar{x}_0 = [\bar{p}^x_{ini}, 2.5]^T$. Then, we have that its position of x-axis is $\bar{p}^x_k = \bar{p}^x_{ini} + \delta \times (k-1) \times \bar{v}^x$. To formulate the overtaking task, we define the following three sets as shown in Figure 7.8: $\mathbb{S}_{\mu_3} = \{z \in \mathbb{R}^3 \mid [0, -5, -3]^T \leq z \leq [35, 0, 3]^T\}$, $\mathbb{S}_{\mu_4} = \{z \in \mathbb{R}^3 \mid [35, -5, -3]^T \leq z \leq [60, 5, 3]^T\}$, and $\mathbb{S}_{\mu_5} = \{z \in \mathbb{R}^3 \mid [60, -5, -3]^T \leq z \leq [120, 0, 3]^T\}$.

Let us choose the sampling period as $\delta = 0.2\text{s}$. To respect the time constraint and the input constraint for Veh_1, we consider two possible solutions to the previous reachability problem: (1) quick overtaking: overtake Veh_2 before Veh_3 passes Veh_2; (2) slow overtaking: wait until Veh_3 passes Veh_2 and then overtake Veh_2. The quick overtaking can be encoded into an STL formula:

$$\varphi_1 = (\mu_3 \mathsf{U}_{[0,16]}(\mu_4 \wedge \neg\mu_2)\mathsf{U}_{[0,15]}\mu_5 \mathsf{U}_{[0,30]}\mathsf{G}_{[0,2]}\mu_1) \wedge \mathsf{G}_{[0,80]}\neg\mu_6, \tag{7.13}$$

where $\mathbb{S}_{\mu_6} = \{z \in \mathbb{R}^6 \mid [\bar{p}^x(16), 0, -\infty]^T \leq z \leq [\bar{p}^x(0), 5, \infty]^T\}$. Note that $\mathbb{S}_{\mu_6}$ denotes the reachable set for the vehicle Veh_3 within the time interval

$[0, 16]$ seconds and 16 (that corresponds to the sampling index $k = 80$) is the maximal time instant that the vehicle Veh_1 can reach the set $\mathbb{S}_{\mu_5}$ in the sprit of φ_1. The slow overtaking can be encoded into an STL formula

$$\varphi_2 = (\mu_3 \mathsf{U}_{[16,32]}(\mu_4 \wedge \neg\mu_2)\mathsf{U}_{[0,15]}\mu_5\mathsf{U}_{[0,30]}\mathsf{G}_{[0,2]}\mu_1) \wedge \mathsf{G}_{[0,80]}\neg\mu_7, \quad (7.14)$$

where $\mathbb{S}_{\mu_7} = \{z \in \mathbb{R}^2 \mid [-\infty, 0, -\infty]^T \leq z \leq [\bar{p}^x(16), 5, \infty]^T\}$. Note that $\mathbb{S}_{\mu_7}$ denotes the reachable set for the vehicle Veh_3 within the time interval $[16, +\infty)$ and 16 (that corresponds to the sampling index $k = 80$) is the minimal time instant that the vehicle Veh_1 can reach the set $\mathbb{S}_{\mu_4}$ in the sprit of φ_2. The overall specification is written as $\varphi = \varphi_1 \vee \varphi_2$.

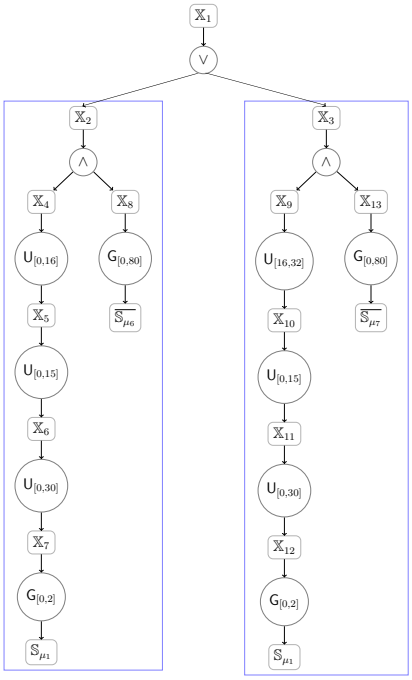

Figure 7.9: The constructed tTLT $\mathcal{T}_\varphi$, where the left and right blue boxes are the tTLTs $\mathcal{T}_{\varphi_1}$ and $\mathcal{T}_{\varphi_2}$, respectively.

Using Algorithm 7.1, one can construct the tTLT for φ (see Figure 7.9), where

$$\mathbb{X}_7(t_k) = \mathbb{X}_{12}(t_k) = \overline{\mathcal{R}^m(X, \overline{\mathbb{S}_{\mu_1}}, [0, 2], k)},$$

$$
\begin{aligned}
&\mathbb{X}_6(t_k) = \mathcal{R}^M(X, \mathbb{X}_7(t_0), \mathbb{S}_{\mu_5}, [0, 30], k),\\
&\mathbb{X}_{11}(t_k) = \mathcal{R}^M(X, \mathbb{X}_{12}(t_0), \mathbb{S}_{\mu_5}, [0, 30], k),\\
&\mathbb{X}_5(t_k) = \mathcal{R}^M(X, \mathbb{X}_6(t_0), \mathbb{S}_{\mu_4} \cap \overline{\mathbb{S}_{\mu_2}}, [0, 15], k),\\
&\mathbb{X}_{10}(t_k) = \mathcal{R}^M(X, \mathbb{X}_{11}(t_0), \mathbb{S}_{\mu_4} \cap \overline{\mathbb{S}_{\mu_2}}, [0, 15], k),\\
&\mathbb{X}_4(t_k) = \mathcal{R}^M(X, \mathbb{X}_5(t_0), \mathbb{S}_{\mu_3}, [0, 16], k),\\
&\mathbb{X}_9(t_k) = \mathcal{R}^M(X, \mathbb{X}_{10}(t_0), \mathbb{S}_{\mu_3}, [16, 32], k),\\
&\mathbb{X}_8(t_k) = \overline{\mathcal{R}^m(X, \mathbb{S}_{\mu_6}, [0, 80], k)},\\
&\mathbb{X}_{13}(t_k) = \overline{\mathcal{R}^m(X, \mathbb{S}_{\mu_7}, [0, 80], k)},\\
&\mathbb{X}_2(t_k) = \mathbb{X}_4(t_k) \cap \mathbb{X}_8(t_k),\\
&\mathbb{X}_3(t_k) = \mathbb{X}_9(t_k) \cap \mathbb{X}_{13}(t_k), \text{ and}\\
&\mathbb{X}_1(t_k) = \mathbb{X}_2(t_k) \cup \mathbb{X}_3(t_k).
\end{aligned}
$$

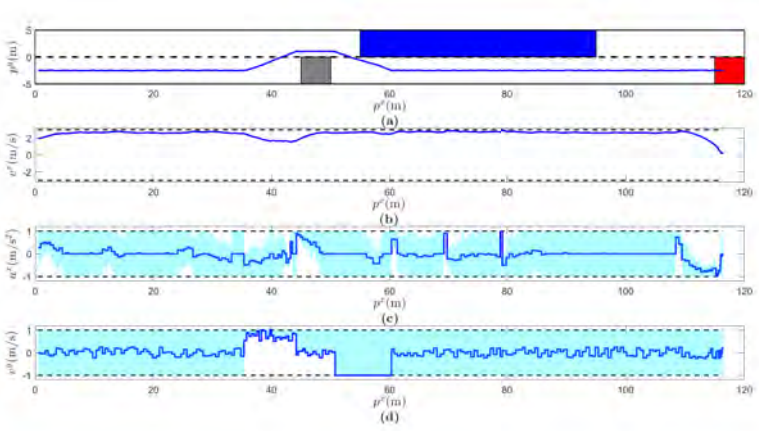

Figure 7.10: Trajectories for one realization of disturbance signal in the fast overtaking: (a) position trajectory; (b) velocity trajectory of x-axis; (c) control trajectory of x-axis; (d) control trajectory of y-axis.

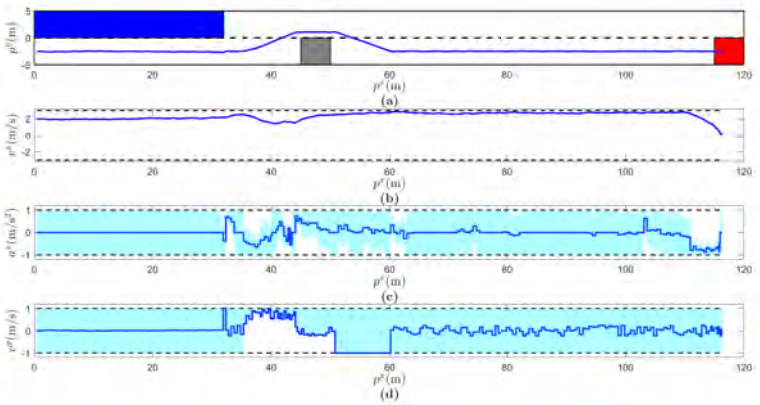

Figure 7.11: Trajectories for one realization of disturbance signal in the slow overtaking: (a) position trajectory; (b) velocity trajectory of x-axis; (c) control trajectory of x-axis; (d) control trajectory of y-axis.

In the following, two simulation cases are considered and the online control synthesis algorithm is implemented.

In the fast overtaking, we choose the initial position $\bar{p}^x_{ini} = 95$ and the moving velocity $\bar{v}^x = -2$ for the vehicle Veh_3 and the initial position $x_0 = [0.5, -2.5, 2]^T$ for Veh_1. By using the results for the satisfiability check, the specification φ_1 is robustly satisfiable while φ_2 is infeasible. Figure 7.10 (a) shows the position trajectories, from which we can see that the whole specification is completed. The blue region denotes the set $\mathbb{S}_{\mu_6}$. Figure 7.10 (b) shows the velocity trajectory of v^x and Figure 7.10 (c)–(d) show the corresponding control inputs, where the dashed lines denote the control bounds. The cyan regions represent the synthesized control sets and the blue lines are the control trajectories.

In the slow overtaking, we choose the initial position $\bar{p}^x_{ini} = 80$ and the moving velocity $\bar{v}^x = -3$ for the vehicle Veh_3 and the same initial position $x_0 = [0.5, -2.5]^T$ for Veh_1. In contrary to the fast overtaking, the specification φ_2 is robustly satisfiable while φ_1 is infeasible. Figure 7.11 (a) shows the position trajectories, from which we can see that the whole specification is completed. The blue region denotes the intersection between the set X and the set $\mathbb{S}_{\mu_7}$. Figure 7.11 (b) shows the velocity trajectory of v^x and Figure 7.11 (c)–(d) show the corresponding control input trajectories of a^x and v^y.

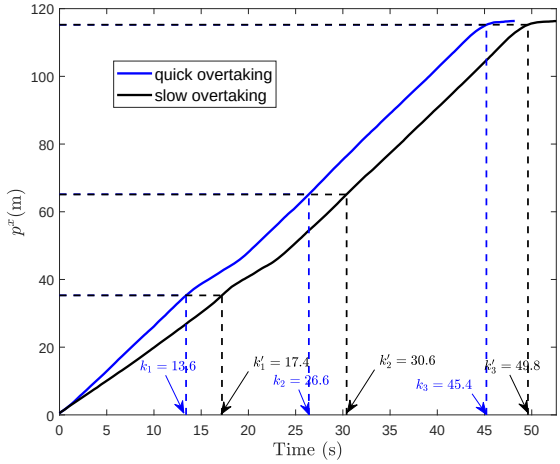

Figure 7.12: The position trajectories of x-axis along the time for both type of overtaking STL tasks as defined in (7.13) and (7.14).

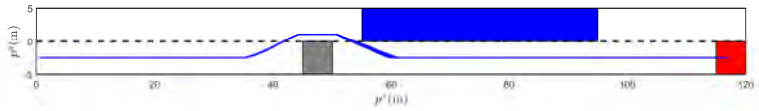

Figure 7.13: State trajectories for 100 realizations of disturbance signals in the fast overtaking.

Although the position trajectories in the two cases are similar as shown in Figs. 7.10(a)–7.11(a), we highlight their difference through the evolution of the position of x-axis along the time in Figure 7.12. We use k_1, k_2, and k_3 (or k_1', k_2', and k_3') to denote the minimal time instants that Veh_1 reaches the sets $\mathbb{S}_{\mu_4}$, $\mathbb{S}_{\mu_5}$, and $\mathbb{S}_{\mu_1}$ in the fast overtaking (or the slow overtaking), respectively. We can see that these two position trajectories satisfy the time intervals encoded in the φ_1 and φ_2, respectively. Furthermore, in order to show the robustness, we run 100 realizations of the disturbance trajectories in the fast overtaking. The position trajectories for such 100 realizations are shown in Figure 7.13.

Finally, we report the computation time of this example, which was run in Matlab R2016a with MPT toolbox [133] on a Dell laptop with Windows 7, Intel i7-6600U CPU2.80 GHz and 16.0 GB RAM. We perform reachability analysis

for constructing the tTLT offline, which takes 59.10 seconds. For online control synthesis, the minimal computation time at a single time step over 100 realizations is 0.23 seconds, while the maximal computation time is 1.07 seconds. The average time of each time step is 0.31 seconds. We remark that the mixed-integer formulation is difficult to implement in this example. This is because the computational complexity of mixed-integer programming grows exponentially with the horizon of the STL formula, which in this example reaches up to 400 sampling instants, much longer than the horizons considered in the simulation examples of [26]–[28].

7.8 Summary

In this chapter, an approach for the robust satisfiability check and online control synthesis of uncertain discrete-time systems under STL specifications was proposed. Using the notion of tTLT, a sufficient condition was obtained for the robust satisfiability check of the uncertain systems. Moreover, when the underlying system is deterministic, a necessary and sufficient condition was further obtained for the satisfiability check problem. An online control synthesis algorithm was proposed and shown to be sound for uncertain systems, and both sound and complete for deterministic systems.

Chapter 8

Distributed Motion Coordination under LTL Specifications

Chapters 6 and 7 studied the robust control of uncertain systems under temporal logic specifications. In this chapter, we further investigate the motion coordination of multi-agent systems (MAS) under temporal logic specifications. We consider a group of agents working in a shared workspace, where each agent is assigned a linear temporal logic (LTL) specification. The objective is to design a fully distributed motion coordination strategy, such that the specification of each agent is satisfied on the premise that safety (no collision with obstacles in the workspace and no inter-agent collision) is guaranteed.

8.1 Introduction

One challenge for MAS is the design of coordination strategies between agents that enable them to perform operations safely and efficiently in a shared workspace while achieving individual/group motion objectives. In recent years, the attention paid to this problem has grown significantly due to the emergence of new applications, such as smart transportation and service robotics. The existing literature can be divided into two categories: *path coordination* and *motion coordination*. The former category plans and coordinates the entire paths of all the agents in advance (offline), while the latter category focuses on decentralized and online approaches that allow agents to resolve

conflicts online when one occurs[1] [134]. This chapter aims at developing a fully distributed strategy for multi-agent motion coordination (MAMC) with safety guarantees.

Depending on how the controller is synthesized for each agent, the literature concerning MAMC can be further classified into two types: the reactive approach and the planner-based approach. Typical methods that generate reactive controllers consist of the potential-field approach [135], [136], sliding mode control [137], [138] and control barrier functions [139], [140]. These reactive-style methods are fast and operate well in real-time and unconstrained situations. However, these methods are sensitive to deadlocks that are caused by local minima. Moreover, procedures for setting control parameters is not analyzed formally when explicit constraints on the system states and/or inputs are presented [134]. Apart from the above, other reactive methods include the generalized roundabout policy [141] and a family of biologically inspired methods [142].

An early example of the planner-based method is the work of Azarm and Schmidt [143], where a framework for online coordination of multiple mobile agents was proposed. In this framework, MAMC was solved as a sequential trajectory planning problem, where priorities are assigned to agents when conflicts are detected, and then a motion planning algorithm is implemented to generate conflict-free paths. Based on this framework, various applications and different motion planning algorithms are investigated. Guo and Parker [144] proposed a MAMC strategy based on the D* algorithm. In this work, each agent has an independent goal position to reach and know all path information. In [145], a distributed bidding algorithm was designed to coordinate the movement of multiple agents, which focuses on area exploration. In the work of Liu [146], conflict resolution at intersections was considered for connected autonomous vehicles, where each vehicle is required to move along a pre-planned path. A literature review on MAMC can be found in [147].

No matter which type of controllers is implemented, safety has always been a crucial issue for MAMC. In addition, most of the above mentioned literature concerning MAMC considers relatively simple tasks for each agent (*e.g.,* an arrival task from initial state to goal state). However, as agents become more capable, a recent trend in the area of agent motion planning is to assign agents more complex, high-level tasks, such as temporal logic specifica-

[1] In some literature these two terms are used interchangeably. In this chapter, we try to distinguish between the two as explained above.

tions. In the last few years, multi-agent coordination under LTL specifications has been considered [148]–[150]. In [148], [149], offline path coordination instead of online motion coordination is investigated. In [150], online motion coordination is studied for agents with single-integrator dynamics. However, practical state and input constraints are not considered.

Motivated by the above observations, this chapter investigates the MAMC problem for a group of mobile agents moving in a shared workspace, each of which is assigned an LTL specification. Agents are assumed to have limited sensing capabilities and constraints on both state and input are considered. Conflicts are assumed to be local and can occur at arbitrary locations in the workspace. To cope with these setups, a fully distributed MAMC strategy is proposed. The contributions of this work can be summarized as follows.

(i) A framework for distributed MAMC under LTL specifications is proposed for each agent, which consists of three layers: an offline precomputation layer, an initialization layer, and an online coordination layer. The online coordination layer is further decomposed into three steps. Firstly, conflicts are detected within the sensing area of each agent. Once conflicts are detected, a rule is applied to assign dynamically a planning order to each pair of neighboring agents. Finally, a sampling-based algorithm is implemented for each agent that generates a local collision-free trajectory which at the same time satisfies the LTL specification.

(ii) Safety is established by combining the planner-based controller with an emergency braking controller.

(iii) As the motion coordination strategy is designed to be fully distributed and each agent considers only local information of neighboring agents, it is totally scalable in the sense that the computational complexity of the strategy does not increase with the number of agents in the workspace and the size of the workspace.

8.2 Problem Formulation

8.2.1 Model description

Consider a MAS moving in a bounded workspace $\mathbb{W} \subset \mathbb{R}^2$, the dynamics of each agent is given by

$$\dot{\xi}_i(t) = F_i(\xi_i(t), u_i(t)), i \in \mathcal{V} \tag{8.1}$$

where $\xi_i(t) := (x_i(t), \theta_i(t), v_i(t)) \in \mathbb{R}^4$ represents the state of agent i, which contains its position $x_i(t)$ in the workspace $\mathbb{W}$, orientation $\theta_i(t) \in \mathbb{R}$ and velocity $v_i(t) \in \mathbb{R}$ at time t. The function $F_i : \mathbb{R}^4 \times \mathbb{R}^2 \to \mathbb{R}^4$ which describes the state evolution of agent i is given by

$$F_i(\xi_i(t), u_i(t)) = (\cos(\theta_i(t))v_i(t), \sin(\theta_i(t))v_i(t), \omega_i(t), a_i(t))$$

and the input $u_i(t)$ is given by $u_i(t) := (\omega_i(t), a_i(t))$, where $\omega_i(t)$ is the turning rate and $a_i(t)$ is the acceleration of agent i at time t.

The velocity and input of agent i are subject to the hard constraints

$$|v_i(t)| \leq v_{i,\max}, |\omega_i(t)| \leq \omega_{i,\max}, |a_i(t)| \leq a_{i,\max}, \forall t \tag{8.2}$$

where $v_{i,\max}, \omega_{i,\max}, a_{i,\max} > 0$. Let

$$U_i := \{(\omega_i, a_i) : |\omega_i| \leq \omega_{i,\max}, |a_i| \leq a_{i,\max}\} \tag{8.3}$$

$$X_i := \{(x_i, \theta_i, v_i) : x_i \in \mathbb{W}, \theta_i \in [-\pi, \pi], |v_i| \leq v_{i,\max}\}. \tag{8.4}$$

Denote by $\xi_i : [0, \infty) \to \mathbb{R}^4$ the trajectory of agent i with dynamics given by (8.1). Then, we further define $x_i : [0, \infty) \to \mathbb{R}^2$ as the position trajectory of agent i, where $x_i(t) = \text{proj}_2(\xi_i(t)), \forall t \in \text{dom}(\xi_i)$. Given a time interval $[t_1, t_2], t_1 < t_2$, the corresponding trajectory and position trajectory are denoted by $\xi_i([t_1, t_2])$ and $x_i([t_1, t_2])$, respectively. Denote by $\xi_i([t, \infty))$ and $x_i([t, \infty))$ the trajectory and the position trajectory of agent i from time t onwards, respectively.

Supposing that the sensing radius of each agent is the same, denoted by $R > 0$, then the communication graph formed by the group of agents is undirected. The neighboring set of agent i at time t is given by $\mathcal{N}_i(t) = \{j \in \mathcal{V} : \|x_i(t) - x_j(t)\| \leq R, j \neq i\}$, so that $j \in \mathcal{N}_i(t) \Leftrightarrow i \in \mathcal{N}_j(t), \forall i \neq j, \forall t$.

8.2.2 Problem

The group of agents are working in a common workspace $\mathbb{W} \subset \mathbb{R}^2$, which is populated with a set of closed sets O_i, corresponding to obstacles. Let $\mathbb{O} = \cup_i O_i$, then the free space $\mathbb{F}$ is denoted by $\mathbb{F} := \mathbb{W} \setminus \mathbb{O}$.

Each agent i is subject to its own specification φ_i, which is in the form of an LTL$_{-\mathsf{X}}$ formula that is defined over the workspace $\mathbb{W}$. LTL$_{-\mathsf{X}}$ [151] is a known fragment of LTL, in which the X ("next") operator is not allowed. The choice of LTL$_{-\mathsf{X}}$ over LTL is motivated by the fact that LTL (given in (2.8)) increases expressivity (over LTL$_{-\mathsf{X}}$) only over words with a finite number of repetitions of a symbol, and a word corresponding to a continuous signal will never have a finite number of successive repetitions of a symbol.

Given a trajectory ξ_i, the notation $\xi_i \models \varphi_i$ means that the trajectory ξ_i satisfies the specification φ_i. Given the position x_i of agent i, we refer to its *footprint* $\phi_i(x_i)$ as the set of points in $\mathbb{W}$ that are occupied by agent i in this position. The objective of this chapter is to find, for each agent i, a trajectory ξ_i such that $\xi_i \models \varphi_i$ on the premise that safety (no collisions with static obstacles and no inter-agent collisions) is guaranteed. Let t_0 be the task activation time of agent i. Then, the centralized and offline version of the MAMC problem is formulated below:

$$\text{find} \quad \{\xi_i([t_0, \infty))\}_{i \in \mathcal{V}} \tag{8.5a}$$

subject to

$$(8.1) \text{ and } (8.2), \forall i \in \mathcal{V}, \tag{8.5b}$$

$$\xi_i([t_0, \infty)) \models \varphi_i, \forall i \in \mathcal{V}, \tag{8.5c}$$

$$\phi_i(x_i([t_0, \infty)) \subset \mathbb{F}, \forall i \in \mathcal{V}, \tag{8.5d}$$

$$\phi_i(x_i(t)) \cap \phi_j(x_j(t)) = \emptyset, \forall i, j \in \mathcal{V}, i \neq j, \forall t. \tag{8.5e}$$

Constraint (8.5d) means that the footprint of each agent will not collide with the static obstacles at any time. Constraint (8.5e) means that the footprint of two different agents can not intersect at any time, thus guaranteeing no inter-agent collision occurs. Note that in this chapter, it is assumed that each agent is not aware of the existence of other agents. Therefore, centralized motion coordination can not be conducted. Moreover, each agent has only local view and local information, *i.e.,* each agent considers only agents in its neighborhood $\mathcal{N}_i(t)$ at each time t and can have only local information of its neighbors. Under these settings, the MAMC problem (8.5) is broken into local distributed

motion coordination problems and solved online for individual agents. Let $x_j([t, t_j^*(t)])$ be the local position trajectory of agent j that is available to agent i at time t, where $t_j^*(t) := \min_{t'>t}\{x_j(t') \notin \mathcal{B}(x_i(t), R)\}$. Then, the (online) motion coordination problem for agent i is formulated as

$$\text{find} \quad \xi_i([t, \infty)) \tag{8.6a}$$

subject to

$$(8.1) \text{ and } (8.2), \tag{8.6b}$$

$$\xi_i([t_0, t] \cup [t, \infty)) \models \varphi_i, \tag{8.6c}$$

$$\phi_i(x_i([t, \infty))) \subset \mathbb{F}, \tag{8.6d}$$

$$\phi_i(x_i(t)) \cap \phi_j(x_j(t)) = \emptyset, \forall j \in \mathcal{N}_i(t), \forall t' \in [t, t_j^*(t)], \tag{8.6e}$$

where $\xi_i([t_0, t])$ is the history trajectory.

8.3 Solution

The proposed solution to the motion coordination problem (8.6) consists of three layers: 1) an offline pre-computation layer, 2) an initialization layer, and 3) an online coordination layer.

8.3.1 Structure of each agent

Before explaining the solution, the structure of each agent is presented (see Figure 8.1). Each agent i is equipped with five modules, the conflict detection module, the planning order assignment module, the trajectory planning module, the control module and the communication module. The first four modules work sequentially while the communication module works in parallel with the first four.

During online execution, agent i tries to satisfy its specification safely by resolving conflicts with other agents. This is done by following some mode switching rules encoded into a finite state machine, see Figure 8.2. Each finite state machine has the following three modes:

- **Free**: agent i moves as planned. This is the normal mode.
- **Busy**: agent i enters this mode when conflicts are detected. In this mode, the planning order assignment module and the trajectory planning module are activated.

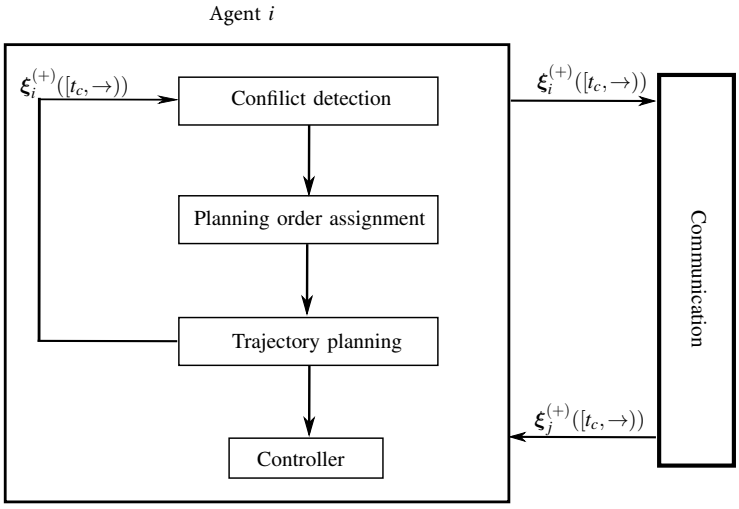

Figure 8.1: The structure of agent i.

- **Emerg**: agent i starts an emergency stop process.

In Figure 8.2, the transitions between different modes of the finite state machine are depicted. Initially, agent i is in **Free** mode. The conflict detection module is activated when the online execution starts. Once conflict neighbors (will be defined later) are detected, agent i enters to **Busy** mode and the planning order assignment and the trajectory planning modules are activated to solve the conflicts, otherwise, agent i stays in **Free** mode. When agent i is in **Busy** mode, it switches back to **Free** mode if the trajectory planning module returns a feasible solution, and the solution will be broadcasted to the agent's

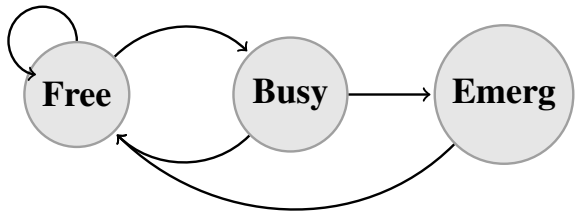

Figure 8.2: The three modes of agent i and the transitions among them.

neighboring area as well as sent to the controller for execution, otherwise (*e.g.,* no feasible plan is found), agent i switches to **Emerg** mode and a braking controller (defined later) is applied. Note that when agent i switches to **Emerg** mode, it will come to a stop but with power-on. This means that agent i will continue monitoring the environment and restart (switches back to **Free** mode) when it is possible.

8.3.2 Offline pre-computation

Braking controller

As stated in the previous subsection, when agent i enters **Emerg** mode, it starts an emergency stop process. In this chapter, we consider bounded deceleration for the agents. Due to this, the notions of *braking (position) trajectory* and *braking area* are needed for each agent. When agent i switches to **Emerg** mode, the maximum deceleration $-a_{i,\max}$ is applied to decelerate agent i to zero velocity. The braking controller is designed as

$$u_i^{\text{br}}(t) = \begin{cases} \left(-\omega_{i,\max}, -a_{i,\max}\dfrac{v_i(t)}{|v_i(t)|}\right), & \text{if } |v_i(t)| \neq 0, \\ \mathbf{0} \qquad\qquad\qquad\quad , & \text{if } |v_i(t)| = 0. \end{cases} \tag{8.7}$$

Supposing that agent i starts to apply the braking controller u_i^{br} at time t_0 and the velocity of agent i is $v_i(t_0)$. Then, the time needed for agent i to decelerate to zero velocity is given by

$$t^*(v_i(t_0)) := \frac{|v_i(t_0)|}{a_{i,\max}}. \tag{8.8}$$

Given initial state $\xi_i(t_0)$, denote by $\xi_i^{\text{br}}([t_0, t_0 + t^*(v_i(t_0))])$ the *braking trajectory* of agent i, and then we have

$$\xi_i^{\text{br}}(t) = \xi_i(\xi_i(t_0), u_i^{\text{br}}, t), t \in [t_0, t_0 + t^*(v_i(t_0))]. \tag{8.9}$$

The *braking position trajectory* of agent i, denoted by $x_i^{\text{br}}([t_0, t_0 + t^*(v_i(t_0))])$, is then given by

$$x_i^{\text{br}}([t_0, t_0 + t^*(v_i(t_0))]) := \text{proj}_2(\xi_i^{\text{br}}([t_0, t_0 + t^*(v_i(t_0))])). \tag{8.10}$$

Proposition 8.1. *The braking position trajectory of agent i starting at position $x_i, \forall x_i \in \mathbb{R}^2$ can be over-approximated by the area*

$$\Pi_i(x_i) := \mathcal{B}(x_i, D_i^{sf}),$$

where

$$D_i^{sf} = \begin{cases} \frac{2v_{i,\max} - a_{i,\max}\pi/\omega_{i,\max}}{\sqrt{a_{i,\max}^2 + \omega_{i,\max}^2}}, & \text{if } \frac{v_{i,\max}}{a_{i,\max}} \geq \frac{\pi}{\omega_{i,\max}} \\ \frac{v_{i,\max}}{\sqrt{a_{i,\max}^2 + \omega_{i,\max}^2}}, & \text{if } \frac{v_{i,\max}}{a_{i,\max}} < \frac{\pi}{\omega_{i,\max}}. \end{cases} \tag{8.11}$$

That is, $x_i^{\text{br}}([t_0, t_0 + t^*(v_i)]) \subset \Pi_i(x_i), \forall \xi_i(t_0) \in \{\xi_i \in \mathbb{R}^4 : \text{proj}_2(\xi_i) = x_i\}$.

Proof. Given an initial state $\xi_i(t_0) = (x_i(t_0), \theta_i(t_0), v_i(t_0))$ and the braking controller (8.7), the braking position trajectory $x_i^{\text{br}}([t_0, t_0 + t^*(v_i(t_0))])$ is given by

$$\begin{aligned} & x_i^{\text{br}}(t_0 + t) \\ =& x_i(t_0) + \int_0^t \begin{pmatrix} \cos(\theta_i(s)) v_i(s) \\ \sin(\theta_i(s)) v_i(s) \end{pmatrix} ds \\ =& x_i(t_0) + \int_0^t \begin{pmatrix} \cos(\theta_i(t_0) - \omega_{i,\max} s)\Big(v_i(t_0) - \dfrac{a_{i,\max} v_i(t_0) s}{|v_i(t_0)|}\Big) \\ \sin(\theta_i(t_0) - \omega_{i,\max} s)\Big(v_i(t_0) - \dfrac{a_{i,\max} v_i(t_0) s}{|v_i(t_0)|}\Big) \end{pmatrix} ds \\ =& x_i(t_0) + \frac{1}{a_{i,\max}^2 + \omega_{i,\max}^2} \\ & \begin{pmatrix} -a_{i,\max} \text{sgn}(v_i(t_0)) k_2(t_0, t) - \omega_{i,\max} k_1(t_0, t) \\ -a_{i,\max} \text{sgn}(v_i(t_0)) k_1(t_0, t) + \omega_{i,\max} k_2(t_0, t) \end{pmatrix}, \end{aligned}$$

where

$$\begin{aligned} k_1(t_0, t) =& \sin(\theta_i(t_0) - \omega_{i,\max} t)(v_i(t_0) - \text{sgn}(v_i(t_0)) a_{i,\max} t) \\ & - v_i(t_0) \sin(\theta_i(t_0)), \\ k_2(t_0, t) =& \cos(\theta_i(t_0) - \omega_{i,\max} t)(v_i(t_0) - \text{sgn}(v_i(t_0)) a_{i,\max} t) \\ & - v_i(t_0) \cos(\theta_i(t_0)), t \in [0, t^*(v_i(t_0))] \end{aligned}$$

and $\text{sgn}(\cdot)$ is a sign function.

Let $D := \max_{v_i \in \mathbb{V}_i, t \in [t_0, t_0 + t^*(v_i))]} \left\{ \| x_i^{\text{br}}(t) - x_i(t_0) \| \right\}$ be the maximum distance between $x_i(t_0)$ and points belong to the braking position trajectory x_i^{br}. Then, one can derive that $D = D_i^{\text{sf}}$. Thus, the conclusion follows. □

Workspace discretization

Suppose that a cell decomposition is given over the workspace $\mathbb{W}$. The cell decomposition is a partition of $\mathbb{W}$ into finite disjoint convex regions $\Phi := \{X_1, \ldots, X_{M_1}\}$ with $\mathbb{W} = \cup_{l=1}^{M_1} X_l$. Given a point $x \in \mathbb{R}^2$, define the map $Q : \mathbb{R}^2 \to 2^{\Phi}$ as

$$Q(x) := \{X_l \in \Phi : x \in X_l\}.$$

Let AP be the set of atomic propositions defined on $\mathbb{W}$, then the cell decomposition is required to satisfy

$$L_c(x) = L_c(x'), \forall Q(x) = Q(x').$$

That is, for all points that are contained in the same cell, the subset of AP which is true at these points is the same. Let AP_{φ_i} be the set of atomic propositions specified by φ_i, where φ_i is the task specification of agent i. Then, one has $AP_{\varphi_i} \subseteq AP, \forall i \in \mathcal{V}$. We note that the required cell decomposition can be computed exactly or approximately using many existing approaches (Chapters 4-5 [152]).

Given a set $S \subset \mathbb{R}^2$, let

$$Q(S) = \cup_{x \in S} Q(x) = \{X_l \in \Phi : X_l \cap S \neq \emptyset\}, \tag{8.12}$$

which represents the set of cells in Φ that intersects with S.

Braking area

For each agent i whose position is given by $x_i \in \mathbb{W}$, define the *braking area* of agent i at x_i as

$$\psi_i(x_i) := \mathcal{B}(\phi_i(x_i), D_i^{\text{sf}}), \tag{8.13}$$

where $\phi_i(x_i)$ is the footprint of agent i at position x_i. Then, for each cell $X_l \in \Phi$, define the *braking area* of agent i at cell X_l as

$$\psi_i(X_l) := \cup_{x_i \in X_l} \psi_i(x_i) = \cup_{x_i \in X_l} \mathcal{B}(\phi_i(x_i), D_i^{\text{sf}}).$$

Each agent i will compute offline a map $M_i : \Phi \to 2^{\Phi}$, where

$$M_i(X_l) := Q\left(\psi_i(X_l)\right), \forall X_l \in \Phi. \tag{8.14}$$

Intuitively, M_i projects a cell X_l into a set of cells that might be traversed by the braking position trajectory of agent i if agent i starts an emergency stop process inside X_l.

Product Büchi automaton (PBA) and potential function

Before proceeding, the following definitions and notations are required.

Given a PBA $\mathcal{P} = \mathcal{T}_c \times \mathcal{B} = (S_p, S_{0,p}, 2^{AP}, \delta_p, F_p)$ and a state $p \in S_p$, define the projection operator $\mathrm{pj}_X(p) : S_p \to X$ as a map from p to its first component $x \in X$. Given a state $x \in X$, define the function $\beta_{\mathcal{P}} : X \to 2^S$, given by

$$\beta_{\mathcal{P}}(x) := \{s \in S : (x, s) \in S_p\}. \tag{8.15}$$

as a map from x to the subset of Büchi states S that correspond to x. Denote by $D(p, p')$ the set of all finite runs between state $p \in S_p$ and $p' \in S_p$, *i.e.,* $D(p, p') := \{p_1 p_2 \dots p_n | p_1 = p, p' = p_n; (p_k, p_{k+1}) \in \delta_p, \forall k = 1, \cdots, n-1; \forall n \geq 2\}$. The state p' is said to be reachable from p if $D(p, p') \neq \emptyset$. The length of a finite run $\mathbf{p} = p_1 p_2 \dots p_n$ in $\mathcal{P}$, denoted by $Lg(\mathbf{p})$, is defined as

$$Lg(\mathbf{p}) := \sum_{i=1}^{n-1} \|\mathrm{pj}_X(p_{i+1}) - \mathrm{pj}_X(p_i)\|.$$

For all $p, p' \in S_p$, the distance between p and p' is defined as follows:

$$d(p, p') = \begin{cases} \min_{\mathbf{p} \in D(p,p')} Lg(\mathbf{p}), & \text{if } D(p, p') \neq \emptyset \\ \infty, & \text{otherwise.} \end{cases} \tag{8.16}$$

The following definitions of self-reachable set and potential functions are given in [153].

Definition 8.1. *A set $A \subseteq S_p$ is called* self-reachable *if and only if all states in A can reach a state in A, i.e., $\forall p \in A, \exists p' \in A$ such that $D(p, p') \neq \emptyset$.*

Definition 8.2. *For a set $B \subseteq S_p$, a set $C \subseteq B$ is called the* maximal self-reachable set *of B if each self-reachable set $A \subseteq B$ satisfies $A \subseteq C$.*

Definition 8.3 (Potential function of states in $\mathcal{P}$)**.** *The potential function of a state $p \in S_p$, denoted by $V_{\mathcal{P}}(p)$ is defined as:*

$$V_{\mathcal{P}}(p) = \begin{cases} \min_{p' \in F_p^*}\{d(p,p')\}, & \text{if } p \notin F_p^* \\ 0, & \text{otherwise,} \end{cases}$$

where F_p^* is the maximal self-reachable set of the set of accepting states F_p in $\mathcal{P}$ and $d(p,p')$ is defined in (8.16).

Definition 8.4 (Potential function of states in $\mathcal{T}_c$)**.** *Let a state $x \in X$ and a set $M \subseteq \beta_{\mathcal{P}}(x)$, where $\beta_{\mathcal{P}}(x)$ is defined in (8.15). The potential function of x with respect to M, denoted by $V_{\mathcal{T}_c}(x, M)$ is defined as*

$$V_{\mathcal{T}_c}(x, M) = \min_{s \in M}\{V_{\mathcal{P}}((x,s))\}.$$

Remark 8.1. *if $V_{\mathcal{T}_c}(x, M) < \infty$, it means that $\exists s \in M$ such that starting from (x, s), there exists a run that reaches a self-reachable accepting state of $\mathcal{P}$.*

Due to the continuity of the dynamics, the state and the input spaces, the transition system that represents (8.1) is infinite for each agent i. To this end, a probabilistically complete sampling-based algorithm is proposed in [154] to approximate (8.1) by a finite transition system. Given a sampling interval τ_s, the finite transition system that represents (8.1) is denoted by $\mathcal{T}_i := (\hat{X}_i, \hat{X}_i^0, U_i, \rightarrow_{c,i}, F_i, \mathbb{W}, h)$, where

- $\hat{X}_i$ collects all sampling points in X_i that are safe (with respect to the static obstacles), *i.e.,* $\psi_i(\mathrm{proj}_2(x_i)) \subset \mathbb{F}, \forall x_i \in \hat{X}_i$),
- $\hat{X}_i^0 \subseteq \hat{X}_i$,
- $\rightarrow_{c,i} \subseteq \hat{X}_i \times U_i \times \hat{X}_i$,

F_i is given in (8.1), $\mathbb{W}$ is the workspace as well as the observation space, and $h(\cdot) = \mathrm{proj}_2(\cdot)$ is the observation map. Here, X_i, U_i are the set of states and inputs, which are defined in (8.3) and (8.4), respectively. The transition relation $(x_i, u_i, x_i') \in \rightarrow_{c,i}$ holds if and only if $x_i' = \xi_i(x_i, u_i, \tau_s)$. Similarly to (2.10), define

$$Post(x_i) := \{x_i' \in \hat{X}_i : \exists u_i \in U_i, x_i \xrightarrow{u_i} x_i'\}. \tag{8.17}$$

Once $\mathcal{T}_i$ is obtained, one can construct the nondeterministic Büchi automaton (NBA) $\mathcal{B}_i := (S_i, S_i^0, 2^{AP_{\varphi_i}}, \delta_i, F_i)$ for the specification φ_i (Definition 2.7),

the controlled transition system (CTS) $\mathcal{T}_{c,i} := (\hat{X}_i, \hat{X}_i^0, AP_{\varphi_i}, \rightarrow_{c,i}, L_{c,i})$ (Definition 2.8), and then form the PBA $\mathcal{P}_i = \mathcal{T}_{c,i} \times \mathcal{B}_i$ (Definition 2.9). After that, the potential function for each state of $\mathcal{P}_i$ can be computed according to Definition 8.3.

Remark 8.2. *The cell decomposition of the workspace satisfies* $L_c(x) = L_c(x'), \forall Q(x) = Q(x')$. *In addition,* $AP_{\varphi_i} \subseteq AP, \forall i \in \mathcal{V}$. *Therefore, one has* $L_{c,i}(x) = L_{c,i}(x'), \forall Q(x) = Q(x'), \forall i \in \mathcal{V}$ *and thus* $\beta_{\mathcal{P}_i}(x) = \beta_{\mathcal{P}_i}(x'), \forall Q(x) = Q(x'), \forall i \in \mathcal{V}$. *Then, according to Definition 8.3, one can get that if* $V_{\mathcal{P}_i}((x, \beta_{\mathcal{P}_i}(x))) < \infty$, *then* $V_{\mathcal{P}_i}((x', \beta_{\mathcal{P}_i}(x'))) < \infty, \forall x' \in Q(x)$.

8.3.3 Initialization

Before proceeding, the following definition is required.

Definition 8.5. *We call that a trajectory* ξ_i *of (8.1)* safely satisfies *an LTL formula* φ_i *if i)* $\xi_i \models \varphi_i$ *and ii)* $\psi_i(x_i) \subset \mathbb{F}, \forall x_i \in \mathrm{proj}_2(\xi_i)$.

Denote by t_0 the task activation time of agent i. At t_0, agent i first finds a trajectory ξ_i^0 that safely satisfy φ_i. The trajectory planning problem for a single agent can be solved by many existing methods, such as the search/sampling based method [155], [156], automata-based method [157], and optimization-based method [158], [159]. We note that the initial trajectory planning is not the focus of this work. For details about this process, we refer to interested readers to corresponding literatures and the references therein.

The following assumption is needed to guarantee the feasibility of each task specification φ_i.

Assumption 8.1. *At time* t_0, *there exists a trajectory* ξ_i^0 *that safely satisfy* φ_i *for each agent* i.

8.3.4 Online motion coordination

The initially planned trajectory of each agent does not consider the motion of other agents. Moreover, each agent has only local view and local information (about other agents). Therefore, motion coordination is required during online implementation. Based on the sensing information (about the workspace) and broadcasted information (from neighboring agents), each agent can detect conflicts within its neighborhood and then conduct motion replanning such

that conflicts are avoided. In this chapter, we consider that the conflict detection is conducted at a sequence of sampling instants $\{t_k\}_{k\in\mathbb{N}}$ for each agent i, where $t_{k+1} - t_k \equiv \Delta$.

Conflict detection

Before proceeding, the following notation is introduced. Given a position trajectory $x_i([t_1, t_2])$ and a cell $X_l \in \Phi$, the function $\Gamma : \mathcal{F}(\mathbb{R}_{\geq 0}, \mathbb{R}^2) \times \Phi \to 2^{\mathbb{R}_{\geq 0}}$, defined as

$$\Gamma(x_i([t_1, t_2]), X_l) := \{t \in [t_1, t_2] : x_i(t) \in X_l\}, \tag{8.18}$$

gives the time interval that the position trajectory $x_i([t_1, t_2])$ occupies the cell X_l. $\mathcal{B}(x_i(t_k), R)$ represents the sensing area of agent i at time t_k and $Q(\mathcal{B}(x_i(t_k), R))$ represents the set of cells that intersect with $\mathcal{B}(x_i(t_k), R)$. Let

$$t_i^{fl}(t_k) := \min_{t>t_k}\{x_i(t) \notin \mathcal{B}(x_i(t_k), R)\}$$

be the first time that agent i leaves its sensing area $\mathcal{B}(x_i(t_k), R)$. Then, denote by

$$S_i(t_k) := Q\big(x_i([t_k, t_i^{fl}(t_k)])\big) \tag{8.19}$$

the set of cells traversed by agent i within $Q(\mathcal{B}(x_i(t_k), R))$ until it leaves it at $t_i^{fl}(t_k)$. Moreover, for each $X_l \in S_i(t_k)$, the braking area of agent i is given by $\psi_i(X_l)$. Then, we define

$$Res_i(X_l) := M_i(X_l) \tag{8.20}$$

as the set of reserved cells by agent i in order to safely brake when agent i is within X_l. According to (8.18), the time interval that agent i occupies the cell $X_l \in S_i(t_k)$ is given by $\Gamma(x_i([t_k, t_i^{fl}(t_k)]), X_l)$. In addition, by the continuity of $x_i([t_k, \infty))$, one can conclude that $\Gamma(x_i([t_k, t_i^{fl}(t_k)]), X_l)$ is given by one or several disjoint time interval(s) of the form $[a, b), a < b$. Supposing that

$$\Gamma(x_i([t_k, t_i^{fl}(t_k)]), X_l) = \cup_{l=1}^{m}[a_l, b_l), \tag{8.21}$$

where m is the number of disjoint intervals in $\Gamma(x_i([t_k, t_i^{fl}(t_k)]), X_l)$. Denote by $\mathcal{T}_i(X_l)$ the time interval that agent i reserves the area $Res_i(X_l)$. Then, it can be over-approximated by

$$\mathcal{T}_i(X_l) := \cup_{l=1}^{m}[a_l, b_l + t^*(v_{i,\max})), \tag{8.22}$$

where $t^*(v_{i,\max})$ defined in (8.8) is the time needed for agent i to decelerate from the maximum allowed velocity $v_{i,\max}$ to zero velocity.

Then, we have the following definitions.

Definition 8.6. *We say that there is a* spatial-temporal conflict *between agent i and j at time t_k if $\exists X_l \in S_i(t_k), X_{l'} \in S_j(t_k)$ such that $Res_i(X_l) \cap Res_j(X_{l'}) \neq \emptyset$ and $\mathcal{T}_i(X_l) \cap \mathcal{T}_j(X_{l'}) \neq \emptyset$.*

Based on Definition 8.6, define the set of conflict neighbors of agent i at time t_k, denoted by $\tilde{\mathcal{N}}_i(t_k)$, as

$$\begin{aligned}\tilde{\mathcal{N}}_i(t_k) :=&\{j \in \mathcal{N}_i(t_k) : \exists X_l \in S_i(t_k), X_{l'} \in S_j(t_k) \text{ s.t.} \\ &Res_i(X_l) \cap Res_j(X_{l'}) \neq \emptyset \wedge \mathcal{T}_i(X_l) \cap \mathcal{T}_j(X_{l'}) \neq \emptyset\}.\end{aligned} \tag{8.23}$$

Then, we have the following Proposition.

Proposition 8.2. *For agent i, if $\tilde{\mathcal{N}}_i(t_k) = \emptyset$, then one has*

i) $\psi_i(x_i(t)) \cap \psi_j(x_j(t)) = \emptyset, \forall j \in \mathcal{N}_i(t_k), \forall t \in [t_k, t_i^{fl}(t_k)]$;

ii) $\phi_i(x_i(t)) \cap \phi_j(x_j(t)) = \emptyset, \forall j \in \mathcal{N}_i(t_k), \forall t \in [t_k, t_i^{fl}(t_k)]$.

Proof. From (8.8), one can see that the braking time $t^*(v_i)$ increases monotonically with the absolute value of v_i. Therefore, $t^*(v_{i,\max})$ is the maximum time required to stop for all possible velocities. In addition, according to (8.14), (8.21) and (8.22), one has $\psi_i(x_i(t)) \subset Res_i(X_l), \forall x_i(t) \in X_l$ and $\Gamma(x_i^{\text{br}}([t, t^*(v_i(t))]), X_l) \subset \mathcal{T}_i(X_l), \forall x_i(t) \in X_l$. That is to say, $\tilde{\mathcal{N}}_i(t_k) = \emptyset$ implies i).

Let $\phi_i(X_l) = \cup_{x_i \in X_l} \phi_i(x_i)$. Then, one has $\phi_i(X_l) \subset \psi_i(X_l)$. The time interval that the footprint of agent i occupies X_l is given by $\Gamma(x_i([t, t_i^{fl}(t_k)]), X_l)$ and $\Gamma(x_i([t, t_i^{fl}(t_k)]), X_l) \subset \mathcal{T}_i(X_l)$. Thus, one can further get that i) implies ii). □

Agent i switches to **Busy** mode if and only if the set of conflict neighbors is non-empty (*i.e.,* $\tilde{\mathcal{N}}_i(t_k) \neq \emptyset$). The conflict detection process is outlined in Algorithm 8.1.

Algorithm 8.1 *conflictDetection*

Input: $S_j(t_k), \Gamma(x_j([t_k, t_j^{fl}(t_k)]), X_l), \forall X_l \in S_j(t_k)$ for each $j \in \mathcal{N}_i^+(t_k)$.
Output: Set of conflict neighbors $\tilde{\mathcal{N}}_i(t_k)$.
1: Initialize $\tilde{\mathcal{N}}_i(t_k) = \emptyset$.
2: Compute $Res_j(X_l), \mathcal{T}_j(X_l)$ for $j \in \mathcal{N}_i^+(t_k), X_l \in S_j(t_k)$,
3: **for** $j \in \mathcal{N}_i(t_k)$ **do**
4: **if** $\exists X_l \in S_i(t_k), X_{l'} \in S_j(t_k)$ s.t. $Res_i(X_l) \cap Res_j(X_{l'}) \neq \emptyset \wedge \mathcal{T}_i(X_l) \cap \mathcal{T}_i(X_{l'}) \neq \emptyset$ **then**
5: $\tilde{\mathcal{N}}_i(t_k) = \tilde{\mathcal{N}}_i(t_k) \cup \{j\}$,
6: **end if**
7: **end for**

Remark 8.3. *To implement Algorithm 8.1, each agent i needs only to broadcast to its neighboring area local information about its plan. To be more specific, which cell (e.g., X_l) within the sensing area $\mathcal{B}(x_i(t_k), R)$ is occupied by agent i and when that happens (i.e., $\Gamma(x_i([t, t_i^{fl}(t_k)]), X_l)$). Note that the orientation and velocity information of each agent are not required to be broadcasted to the neighbors.*

Determine planning order

Based on the neighboring relation and conflict relation, the graph $\mathcal{G}(t_k) = \{\mathcal{V}, \mathcal{E}(t_k)\}$ formed by the group of agents is naturally divided into one or multiple connected subgraphs, and the motion planning is conducted in parallel within each subgraph in a sequential manner. In order to do that, a planning order needs to be decided for each connected subgraph. In this chapter, we propose a simple rule to assign priorities between each pair of neighbors.

The number of neighbors and conflict neighbors of agent i at time t_k are given by $|\mathcal{N}_i(t_k)|$ and $|\tilde{\mathcal{N}}_i(t_k)|$, respectively. Then, we have the following definition.

Definition 8.7. *We say that agent i has* advantage *over agent j at time t_k if $\tilde{\mathcal{N}}_j(t_k) \neq \emptyset$ and*
1) $|\mathcal{N}_i(t_k)| > |\mathcal{N}_j(t_k)|$; OR
2) $|\mathcal{N}_i(t_k)| = |\mathcal{N}_j(t_k)|$ and $|\tilde{\mathcal{N}}_i(t_k)| > |\tilde{\mathcal{N}}_j(t_k)|$.

Let $\mathcal{Y}_i(t_k)$ be the set of neighbors that have higher priority than agent i at time t_k. The planning order assignment process is outlined in Algorithm 8.2.

For each neighbor $j \in \mathcal{N}_i(t_k)$, if j is in **Emerg** mode (and thus will be viewed as a static obstacle) or $\tilde{\mathcal{N}}_j(t_k) = \emptyset$, then agent j has higher priority (lines 3-5). Otherwise, agent j has higher priority in motion planning if agent j has advantage over agent i (lines 6-8). However, for the special case, *i.e.,* $|\mathcal{N}_i(t_k)| = |\mathcal{N}_j(t_k)|$ and $|\tilde{\mathcal{N}}_i(t_k)| = |\tilde{\mathcal{N}}_j(t_k)|$, neither agent i nor j has advantage over the other. In this case, the priority is determined by the initially uniquely assigned priority score for each agent i (*i.e.,* $P_i^0 \neq P_j^0, \forall i, j$). Denote by P_i^0 the priority score of agent i. We say that agent i has *priority* over j if $P_i^0 > P_j^0$ (lines 9-11).

Algorithm 8.2 *planningOrderAssignment*

Input: $\mathcal{N}_j(t_k), \tilde{\mathcal{N}}_j(t_k), P_j^0, j \in \mathcal{N}_i^+(t_k)$.
Output: Set of higher priority neighbors $\mathcal{Y}_i(t_k)$.

1: Initialize $\mathcal{Y}_i(t_k) = \emptyset$.
2: **for** $j \in \mathcal{N}_i(t_k)$ **do**,
3: **if** j is in **Emerg** mode or $\tilde{\mathcal{N}}_j(t_k) = \emptyset$ **then**,
4: $\mathcal{Y}_i(t_k) = \mathcal{Y}_i(t_k) \cup j$,
5: **else**
6: **if** j has advantage over i **then**,
7: $\mathcal{Y}_i(t_k) = \mathcal{Y}_i(t_k) \cup j$,
8: **else**
9: **if** neither agent i nor j has advantage over the other and $P_j^0 > P_i^0$ **then**,
10: $\mathcal{Y}_i(t_k) = \mathcal{Y}_i(t_k) \cup j$,
11: **end if**
12: **end if**
13: **end if**
14: **end for**

Proposition 8.3 (Deadlock-free in planning order assignment)**.** *The planning order assignment rule given in Algorithm 8.2 will result in no cycles, i.e.,* $\nexists \{q_m\}_1^{\hat{K}}, \hat{K} \geq 2$ *such that* $q_{\hat{K}} \in \mathcal{Y}_{q_1}(t_k)$ *and* $q_{m-1} \in \mathcal{Y}_{q_m}(t_k), \forall m = 2, \ldots, \hat{K}$.

Proof. Suppose that there exists a list of nodes $\{q_m\}_1^{\hat{K}}$ such that

$$q_{\hat{K}} \in \mathcal{Y}_{q_1}(t), q_{m-1} \in \mathcal{Y}_{q_m}(t), \forall m = 2, \ldots, \hat{K}, \tag{8.24}$$

at some time t_k. Firstly, we argue that there will be no **Emerg** mode agent in the circle. Without loss of generality, assume that agent $q_{m^*}, m^* \in \{2, \ldots, \hat{K}-1\}$ is in **Emerg** mode. According to Algorithm 8.2, agent q_{m^*} should have highest priority among all its neighbors, which means $q_{m^*} \in \mathcal{Y}_{q_{m^*-1}}(t)$ and $q_{m^*} \in \mathcal{Y}_{q_{m^*+1}}(t)$. This contradicts (8.24) and thus there will be **Emerg** mode agent in the circle.

Since there is no **Emerg** mode agent in the circle, one can get from Algorithm 8.2 that $q_{m_1} \in \mathcal{Y}_{q_{m_2}}(t_k) \Longrightarrow |\mathcal{N}_{q_{m_1}}(t_k)| > |\mathcal{N}_{q_{m_2}}(t_k)|$. Therefore, (8.24) holds if and only if

$$|\mathcal{N}_{q_1}(t_k)| = |\mathcal{N}_{q_2}(t_k)| = \cdots |\mathcal{N}_{q_{\hat{K}}}(t_k)|. \tag{8.25}$$

However, if (8.25) holds, the priority order will further be determined by $\tilde{\mathcal{N}}_{q_m}(t_k)$. Following the similar procedure, we will get

$$|\tilde{\mathcal{N}}_{q_1}(t_k)| = |\tilde{\mathcal{N}}_{q_2}(t_k)| = \cdots |\tilde{\mathcal{N}}_{q_{\hat{K}}}(t_k)|. \tag{8.26}$$

If both (8.25) and (8.26) hold, the priority order will be determined by the initially assigned priority score, which is unique for each agent. Therefore, there will be no circles. □

Remark 8.4. *The rationale behind the rule can be explained as follows. Since the motion planning is conducted sequentially within each subgraph based on the priority order obtained in Algorithm 8.2, then the total time required to complete the motion planning is given by $K\mathcal{O}(dt)$, where $\mathcal{O}(dt)$ represents the time complexity of one round of motion planning and K represents the number of rounds (if multiple agents conduct motion planning in parallel, it is counted as one round), which is determined by the priority assignment rule being used (e.g., if fixed priority is used, the number of rounds is $K = N$). In our rule, we assign the agent with more neighbors or more conflict neighbors the higher priority, and in this way, we try to minimize the number of rounds needed.*

Example 8.1. *Consider a group of 7 agents, whose communication relation and conflict relation (at time t_k) are depicted in Figure 8.3. According to the proposed planning order assignment rule, the motion planning can be completed in 3 rounds, where agents 1 and 3 are in the first round, agents 2, 4 and 7 are in the second round, and agent 5 is in the third round. Note that no motion planning is required for agent 6 since agent 6 has no conflict neighbor.*

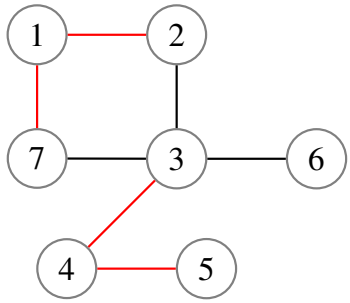

Figure 8.3: Communication and conflict graph, where both the black and red lines represent communication relation and the red lines represent conflict relation.

Motion planning

Before starting to plan, agent i needs to wait for the updated plan from the set of neighbors that has higher priority than agent i (*i.e.,* $j \in \mathcal{Y}_i(t_k)$) and consider them as moving obstacles. For those neighbors $j \in \mathcal{Y}_i(t_k)$, denote by $\xi_j^+([t_k, \infty))$ (correspondingly $x_j^+([t_k, \infty))$) the updated trajectory (position trajectory) of agent j at time t_k and let $t_j^{fl+}(t_k)$ be the first time that agent j leaves its sensing area $\mathcal{B}(x_j(t_k), R)$ according to the updated position trajectory $x_j^+([t_k, \infty))$. Then, similar to (8.19), one can define

$$S_j^+(t_k) := Q(x_j^+([t_k, t_j^{fl+}]))$$

as the updated set of cells traversed by agent j within $Q(\mathcal{B}(x_j(t_k), R))$. For each $X_l \in S_j^+(t_k)$, the time interval that agent j occupies the cell X_l is given by $\Gamma(x_j^+([t_k, t_j^{fl+}]), X_l)$. Suppose that $\Gamma(x_j^+([t_k, t_j^{fl+}]), X_l) = \cup_{l=1}^{\hat{m}}[\hat{a}_l, \hat{b}_l)$, where $\hat{m}$ is the number of disjoint intervals in $\Gamma(x_j^+([t_k, t_j^{fl+}]), X_l)$. Denote by $\mathcal{T}_j^+(X_l)$ the time interval that agent j reserves the area $Res_j(X_l)$ according to the updated position trajectory $x_j^+([t_k, \infty))$. Similarly to (8.22), it can be over-approximated by $\mathcal{T}_j^+(X_l) := \cup_{l=1}^{\hat{m}}[\hat{a}_l, \hat{b}_l + t^*(v_{j,\max})]$.

Then, the trajectory planning problem (TPP) can be formulated as follows:

$$\text{find} \quad \boldsymbol{\xi_i}([t_k, \infty)), \tag{8.27a}$$

subject to

$$(8.1) \text{ and } (8.2), \tag{8.27b}$$

$$\xi_i([t_0, t_k] \cup [t_k, \infty)) \models \varphi_i, \tag{8.27c}$$

$$\psi_i(\text{proj}_2(\xi_i([t_k, \infty)))) \subset \mathbb{F}, \tag{8.27d}$$

$$\begin{aligned} &\psi_i(\text{proj}_2(\xi_i(t))) \cap Res_j(X_l) = \emptyset,\ t \in \mathcal{T}_j^+(X_l), \\ &\qquad \forall j \in \mathcal{Y}_i(t_k), \forall X_l \in S_j^+(t_k). \end{aligned} \tag{8.27e}$$

Constraints (8.27d) and (8.27e) guarantee respectively that there will be no agent-obstacle collisions and inter-agent collisions for agent i.

When relatively simple specifications, *e.g.,* reach-avoid type of tasks, are considered, various existing optimization toolboxes, *e.g.,* IPOPT [160], ICLOCS2 [161], and algorithms, *e.g.,* the configuration space-time search [162], the Hamilton-Jacobian reachability-based trajectory planning [163], and RRT^X [164] can be utilized to solve (8.27). However, if the specifications are complex LTL formulas, the constraint (8.27c) is not easy to be verified online using the methods mentioned above (the PBA $\mathcal{P}_i$ can be used to verify (8.27c), however, it can not deal with the spatial-temporal collision avoidance constraint (8.27e) at the same time). Recently, an on-line RRT-based algorithm is proposed in [153] to generate local paths that guarantee the satisfaction of the global specification. Motivated by this work, a local trajectory generation algorithm is proposed in this chapter, which is outlined in Algorithm 8.3.

Before proceeding, the following notations are required. Denote by

$$NI_i(t_k) := \{Res_j(X_l), \mathcal{T}_j^+(X_l), \forall X_l \in S_j^+(t_k), \forall j \in \mathcal{Y}_i(t_k)\}$$

the set with respect to agent i which contains all local trajectory information of higher priority neighbors at time t_k. Given a state $s_i \in \hat{X}_i$, define the function $B_i : \hat{X}_i \to 2^{S_i}$ as a map from a state $s_i \in \hat{X}_i$ to a subset of valid Büchi states which correspond to s_i (*i.e.,* $B_i(s_i)) \subseteq \beta_{\mathcal{P}_i}(s_i)$, where $\beta_{\mathcal{P}_i}(s_i)$ is defined in (8.15)). Function B_i is used to capture the fact that given a partial trajectory, not all Büchi states in $\beta_{\mathcal{P}_i}$ are valid. At the task activation time t_0, one has $B_i(\xi_i(t_0)) = \beta_{\mathcal{P}_i}(\xi_i(t_0))$. During the online implementation, $B_i(\xi_i(t_k)), t_k > t_0$ is updated by

$$B_i(\xi_i(t_k)) = \beta_{\mathcal{P}_i}(\xi_i(t_k)) \cap \{B_i(\xi_i(t_{k-1})) \cup \texttt{Post}(B_i(\xi_i(t_{k-1})))\} \tag{8.28}$$

where

$$\texttt{Post}(B_i(\xi_i(t_{k-1}))) = \cup_{s_i \in B_i(\xi_i(t_{k-1}))} \texttt{Post}(s_i). \tag{8.29}$$

Algorithm 8.3 *localTrajectoryGeneration*

Input: $\xi_i(t_k), B_i(\xi_i(t_k)), \texttt{Post}(B_i(\xi_i(t_k))), \mathcal{P}_i, V_{\mathcal{P}_i}, \mathbb{O}$, and $NI_i(t_k)$.
Output: A local transition system $\mathcal{T}_{c,i}^L$ and a leaf node ξ_i^f.

1: Initialize $\mathcal{T}_{c,i}^L = (S_i^L, S_{i,0}^L, AP_{\varphi_i}, \rightarrow_{c,i}^L, L_{c,i})$ and $\xi_i^f = \emptyset$, where $S_i^L = S_{i,0}^L = \xi_i(t_k)$ and $\rightarrow_{c,i}^L = \emptyset$, $\texttt{St}(\xi_i(t_k)) = 0$.
2: **for** $k = 1, \ldots, N_i^{\max}$ **do**,
3: $\quad \xi_s \leftarrow$ *generateSample*$(SA_i(t_k))$,
4: $\quad \xi_n \leftarrow$ *nearest*(S_i^L, ξ_s),
5: $\quad u_i^* \leftarrow \arg\min_{u_i \in U_i}\{\|\xi_i(\xi_n, u_i, \tau_s) - \xi_s\|\}$,
6: $\quad \xi_r \leftarrow \xi(\xi_n, u_i^*, \tau_s)$,
7: $\quad B_i(\xi_r) \leftarrow \beta_{\mathcal{P}_i}(\xi_r) \cap \{B_i(\xi_n) \cup \texttt{Post}(B_i(\xi_n))\}$,
8: $\quad$ **if** $B_i(\xi_r) \neq \emptyset \wedge V_{\mathcal{T}_{c,i}}(\xi_r, B_i(\xi_r)) < \infty$ **then**,
9: $\quad\quad \mathbb{O}_i(t_k) \leftarrow$ *updateObstacle*$(\mathbb{O}, NI_i(t_k), [\texttt{St}(\xi_n)\tau_s, \ (\texttt{St}(\xi_n) + 1)\tau_s])$,
10: $\quad\quad$ **if** $\texttt{dist}(\text{proj}_2([\xi_n, \xi_r]), \mathbb{O}_i(t_k)) \geq D_i^{\text{sf}}$ **then**,
11: $\quad\quad\quad S_i^L \leftarrow S_i^L \cup \{\xi_r\}; \rightarrow_{c,i}^L = \rightarrow_{c,i}^L \cup \{\xi_n \xrightarrow{u_i^*} \xi_r\}$,
12: $\quad\quad\quad \texttt{St}(\xi_s) \leftarrow \texttt{St}(\xi_n) + 1$,
13: $\quad\quad$ **end if**
14: $\quad$ **end if**
15: $\quad$ **if** $\text{proj}_2(\xi_r) \notin \mathcal{B}(\text{proj}_2(\xi_i(t_k)), R)$, **then**
16: $\quad\quad k = N_i^{\max} + 1$,
17: $\quad\quad \xi_i^f \leftarrow \xi_r$,
18: $\quad$ **end if**
19: **end for**

At each time instant t_k, Algorithm 8.3 takes the state of agent i, i.e, $\xi_i(t_k)$, $B_i(\xi_i(t_k)), \texttt{Post}(B_i(\xi_i(t_k)))$, the offline computed PBA $\mathcal{P}_i$, potential function $V_{\mathcal{P}_i}$, the set of static obstacles $\mathbb{O}$, and the local trajectory information of higher priority neighbors, *i.e.,* $NI_i(t_k)$ as input. The output is a local CTS $\mathcal{T}_{c,i}^L := (S_i^L, S_{i,0}^L, AP_{\varphi_i}, \rightarrow_{c,i}^L, L_{c,i})$ that is constructed incrementally and a leaf node ξ_i^f. The root state of $\mathcal{T}_{c,i}^L$ is agent i's state $\xi_i(t_k)$. The function $\text{st} : S_i^L \rightarrow \mathbb{N}$ maps a state $x \in S_i^L$ to the number of time steps needed to reach the root state $\xi_i(t_k)$. Initially, $\mathcal{T}_{c,i}^L$ contains one state $\xi_i(t_k)$ and 0 transitions, *i.e.,* $\texttt{st}(\xi_i(t_k)) = 0$, and the leaf node $\xi_i^f = \emptyset$ (line 1). In each iteration (lines 2-19), a new state ξ_s is generated randomly from the set $SA_i(t_k)$ us-

ing the *generateSample* procedure (line 3), where $SA_i(t_k)$ is the sampling area around agent i, given by

$$SA_i(t_k) := \{(x, \theta, v) : x \in \mathcal{B}(\mathrm{proj}_2(\xi_i(t_k)), R + \eta), \\ \theta \in [-\pi, \pi], |v| \leq v_{i,\max}\}, \tag{8.30}$$

where $\eta > 0$ is an offline chosen constant, which guarantees that there exists $s \in SA_i(t_k)$ such that $\mathrm{proj}_2(s) \notin \mathcal{B}(\mathrm{proj}_2(\xi_i(t_k)), R)$. This condition is essential for checking the terminal condition (line 15). The *nearest* function (line 4) is a standard RRT primitive [156] which returns the nearest state in S_i^L to the new sample ξ_s. Then, one further finds, within the set of states that are reachable from ξ_n at time τ_s, the closest one to the new sample ξ_s, *i.e.,* ξ_r (line 6), and the corresponding input u_i^* is obtained (line 5). Here, τ_s is the sampling interval (the same one used for constructing the transition system $\mathcal{T}_i$ in Section 8.3.2). Once ξ_r is obtained, we further compute the subset of valid Büchi states which correspond to ξ_r, *i.e.,* $B_i(\xi_r)$, according to (8.28) (line 7). After that, if both conditions $B_i(\xi_r) \neq \emptyset$ and $V_{\mathcal{T}_{c,i}}(\xi_r, B_i(\xi_r)) < \infty$ are satisfied (which guarantees that there exists a path, starting from ξ_r, that reaches a self-reachable accepting state of $\mathcal{P}_i$, recall Remark 8.1), obstacles that appear during the time interval $[\mathrm{st}(\xi_n)\tau_s, (\mathrm{st}(\xi_n) + 1)\tau_s]$ are added into the workspace using the function *updateObstacles* (Algorithm 8.4). Finally, the state ξ_r is added into S_i^L and the transition relation $\xi_n \xrightarrow{u_i^*} \xi_r$ is added into $\rightarrow_{c,i}^L$ if the distance between the line segment $\mathrm{proj}_2([\xi_n, \xi_r])$ and the obstacles is no less than the safe distance of agent i (lines 10-11), and then the time step needed for ξ_r to reach the root state $\xi_i(t_k)$ is recorded (line 12). The algorithm is terminated when the local sampling tree reaches the outside of the sensing area of agent i, and the leaf node ξ_i^f is then given by the corresponding state ξ_r (line 15-18).

Algorithm 8.4 *updateObstacle*

Input: $\mathbb{O}$, $NI_i(t_k)$ and a time interval $[t_1, t_2]$.
Output: $\mathbb{O}_i(t_k)$.

1: $\mathbb{O}_i(t_k) \leftarrow \mathbb{O}$,
2: **for** $j \in \mathcal{Y}_i(t_k)$, **do**
3: **for** $X_l \in S_j^+(t_k)$, **do**
4: **if** $\mathcal{T}_j^+(X_l) \cap [t_k + t_1, t_k + t_2] \neq \emptyset$, **then**
5: $\mathbb{O}_i(t_k) \leftarrow \mathbb{O}_i(t_k) \cup (Res_j(X_l) \cap SA_i(t_k))$,
6: **end if**
7: **end for**
8: **end for**

Remark 8.5. *The complexity of one iteration of Algorithm 8.3 is the same as for the standard RRT. The functions generateSample and nearest are standard RRT primitives (one can refer to [156] for more details). The complexity of the updateObstacle process (Algorithm 8.4) at time t_k is $\mathcal{O}(1)$ since $|\mathcal{Y}_i(t_k)| \leq N - 1$ and $|S_j^+(t_k)| \leq \sharp\mathcal{B}(x_j(t_k), R), \forall j$, where $\sharp\mathcal{B}(x_j(t_k), R)$ represents the number of cells contained in the sensing area $Q(\mathcal{B}(x_j(t_k), R))$. The computation of $\mathrm{dist}(\mathrm{proj}_2([\xi_n, \xi_r]), \mathbb{O}_i(t_k))$ can be formulated as a convex optimization problem and solved in $\mathcal{O}(1)$ since there is a limited number of obstacles in $SA_i(t_k)$ and each obstacle is of the form of a convex cell. Moreover, the calculations of $B_i(\xi_r)$ and $V_{\mathcal{T}_{c,i}}(\xi_r, B_i(\xi_r))$ are of the complexity of $\mathcal{O}(1)$ since $\mathcal{P}_i, V_{\mathcal{P}_i}$ are computed offline.*

After the local CTS $\mathcal{T}_{c,i}^L$ is obtained, we further need to find a path, starting from the leaf node ξ_i^f, that reaches one of the maximal self-reachable accepting states $F_{p,i}^*$ of $\mathcal{P}_i$. Define

$$P_i(\xi_i^f) := \cup_{s_i \in B_i(\mathcal{T}_{c,i}^L)}(\xi_i^f, s_i)$$

as the set of states in the PBA $\mathcal{P}_i$ that correspond to ξ_i^f. Then, the process is outlined in Algorithm 8.5. Algorithm 8.5 takes the set $P_i(\xi_i^f)$ and the the PBA $\mathcal{P}_i$ as input. It first finds the state p_i^* in $P_i(\xi_i^f)$ that has the minimum potential (line 1). Then, the function *DijksTargets*($\mathcal{P}_i$, *source*, *targets*) (defined in [150]) computes a shortest path in $\mathcal{P}_i$ from "*source*" state to one of the state belonging to the set "*targets*" (line 2). The required path is then the projection of $\mathbf{p}_i$ on the state space of $\mathcal{T}_{c,i}$ (line 3).

Algorithm 8.5 *globalTrajectoryGenration*

Input: $P_i(\xi_i^f)$ and $\mathcal{P}_i$.
Output: a path $\boldsymbol{\rho}_i$.

$p_i^* = \min_{p_i \in P_i(\xi_i^f)} \{V_{\mathcal{P}_i}(p_i)\}$,
$\mathbf{p}_i \leftarrow$ *DijksTargets*$(\mathcal{P}_i, p_i^*, F_{p,i}^*)$,
$\boldsymbol{\rho}_i = \mathrm{pj}_{\hat{X}_i}(\mathbf{p}_i)$,

Algorithm 8.6 *motionCoordination*

Input: (offline) $M_i(X_l), \forall X_l \in \Phi$, $\mathcal{T}_{c,i}$, $\mathcal{B}_i$, $\mathcal{P}_i$, and $V_{\mathcal{P}_i}$.
Output: Real-time plan $\xi_i^+([t_k, \infty)), t_k \geq t_0$.

Initialize: $\xi_i^-([t_0, \infty)) \leftarrow \xi_i^0$, $B_i(\xi_i(t_0)) \leftarrow \beta_{\mathcal{P}_i}(\xi_i(t_0))$, $\mathtt{Post}(B_i(\xi_i(t_0))) \leftarrow \cup_{s_i \in B_i(\xi_i(t_0))} \mathtt{Post}(s_i)$, and agent i is in **Free** mode.
while $t_k > t_0$ and φ_i is not completed **do**,
    Compute $B_i(\xi_i(t_k))$ and $\mathtt{Post}(B_i(\xi_i(t_k)))$ according to (8.28) and (8.29),
    $\tilde{\mathcal{N}}_i(t_k) \leftarrow$ *conflictDetection*(),
    if $\tilde{\mathcal{N}}_i(t_k) \neq \emptyset$ **then**
        agent i switches to **Busy** mode,
        $\mathcal{Y}_i(t) \leftarrow$ *planningOrderAssignment*(),
        $(\mathcal{T}_{c,i}^L, \xi_i^f) \leftarrow$ *localTrajectoryGeneration*(),
        if $\xi_i^f \neq \emptyset$, **then**
            $\boldsymbol{\rho}_i \leftarrow$ *globalTrajectoryGeneration*(),
            $\xi_i^+([t_k, \infty)) \leftarrow$ *DijksTargets*$(\mathcal{T}_{c,i}^L, \xi_i(t_k), \xi_i^f) \uplus \boldsymbol{\rho}_i$,
            agent i switches to **Free** mode,
        else
            agent i switches to **Emerg** mode,
            $u_i \leftarrow u_i^{\mathrm{br}}$,
            $\xi_i^+([t_k, \infty)) \leftarrow \xi_i(\xi_i(t_k), u_i, [0, \infty))$,
        end if
    else
        $\xi_i^+([t_k, \infty)) \leftarrow \xi_i^-([t_k, \infty))$,
    end if
end while

It is possible that after the maximum number of iterations (*i.e.,* $N_i^{\max}$), there exists no local path that reaches the outside of the sensing area of agent i. In this case, the TPP (8.27) is considered infeasible, agent i switches to **Emerg** mode and the braking controller (8.7) is applied. When agent i is in **Emerg** mode, it will continue monitoring the environment (by updating $\mathcal{T}_{c,i}^L$). Once a feasible local path is found, it will switch back to **Free** mode. The whole motion coordination process is summarized in Algorithm 8.6, where the notation *DijksTargets*$(\mathcal{T}_{c,i}^L, \xi_i(t_k), \xi_i^f) \uplus \boldsymbol{\rho}_i$ (line 11) represents a path whose first path segment is given by *DijksTargets*$(\mathcal{T}_{c,i}^L, \xi_i(t_k), \xi_i^f)$ and second path segment is given by $\boldsymbol{\rho}_i$.

Now, we have the following result.

Theorem 8.1 (Safety)**.** *If the sensing radius of the agents satisfies* $R > 2\max_{i\in\mathcal{V}}\{D_i^{sf} + \Delta v_{i,\max}\}$*, where* Δ *is the conflict detection interval, then the resulting real-time plan of Algorithm 8.6 guarantees that there will be no obstacle-agent and inter-agent collisions for agent* i*.*

Proof. If the TPP (8.27) is feasible (agent i will be in **Free** mode after the trajectory planning process), the local trajectory generation algorithm (Algorithm 8.3) and the global trajectory generation algorithm (Algorithm 8.5) guarantees that the constraints (8.27d) and (8.27e) of (8.27) are satisfied. That is to say, there will be no obstacle-agent and inter-agent collisions for agent i. If the TPP (8.27) is infeasible (agent i will switch to **Emerg** mode), agent i would apply the braking controller u_i^{br} until it stops. Since $R > 2\max_{i\in\mathcal{V}}\{D_i^{\text{sf}} + \Delta v_{i,\max}\}$, it guarantees that conflict between any pair of agents (i, j) will be detected at the time that both agent i and j are outside of the braking area of the other. This means that there will be no inter-agent collision during the emergency stop process. On the other hand, when constructing $\mathcal{T}_i$, one has $\psi_i(\text{proj}_2(x_i)) \subset \mathbb{F}, \forall x_i \in \hat{X}_i$), *i.e.,* $\text{dist}(x_i, \mathbb{O}) \geq D_i^{\text{sf}}, \forall x_i \in \hat{X}_i$. Moreover, in the replanning process, it also requires that the distance between $\text{proj}_2([\xi_n, \xi_r])$ and static obstacles $\mathbb{O}$ is no less than D_i^{sf} (line 10, Algorithm 8.3). Therefore, there will be no obstacle-agent collision during the emergency stop process. Thus, no collision will occur for the whole process. □

Remark 8.6. *The framework proposed in this chapter can be extended to MAS where agents are of different dynamics. For example, if there exists a agent* i *whose dynamics is given by the single-integrator* $\dot{x}_i = u_i$ *with the input constraint* $\|u_i\| \leq u_{i,\max}$*. The braking controller can be chosen as* $u_i^{br}(t) = \mathbf{0}$ *and*

the safe distance $D_i^{sf} = 0$. If there exists a agent j whose dynamics is given by the double-integrator $\dot{x}_j = v_j, \dot{v}_j = u_j$ with the velocity constraint $\|v_j\| \leq v_{j,\max}$ and the input constraint $\|u_j\| \leq u_{j,\max}$. The braking controller can be chosen as

$$u_j^{br}(t) = \begin{cases} -u_{j,\max} \frac{v_j(t)}{\|v_j(t)\|}, & \text{if } \|v_j(t)\| \neq 0 \\ \mathbf{0}, & \text{otherwise} \end{cases}$$

and the safe distance $D_j^{sf} = v_{j,\max}^2/(2u_{j,\max})$.

Remark 8.7. *Due to the distributed fashion of the solution and the locally available information, the proposed motion coordination strategy is totally scalable in the sense that the computational complexity of the solution is not increasing with the number of agents. In addition, it is straightforward to extend the work to MAS scenarios where unknown static obstacles and moving obstacles are presented.*

Remark 8.8. *The workspace discretization, the planning order assignment and the over-approximation of the braking area influence the completeness (the ability to find a solution when one exists) of the proposed solution. To improve completeness, online refinement of the cell decomposition of the workspace, search the space of the prioritization scheme (e.g., the randomized search approach with hill-climbing [165]), and less-conservative approximation of the braking area (using the real-time orientation and velocity information) can be utilized. The disadvantage is that the resulting approach will be computationally more complex.*

8.4 Example

We illustrate the results on a MAS consisting of $N = 7$ agents. The velocity and input constraints for agents $i \in \{1, 2, 3, 6, 7\}$ are given by $|v_i| \leq 2\text{m/s}$ and $|\omega_i| \leq 115\text{rad/s}, |a_i| \leq 2\text{m/s}^2$, respectively. The velocity and input constraints for agents $i \in \{4, 5\}$ are given by $|v_i| \leq 1.5\text{m/s}$ and $|\omega_i| \leq 86\text{rad/s}, |a_i| \leq 1.5\text{m/s}^2$, respectively. Then, one can get from (8.11) that $D_i^{\text{sf}} = 0.707\text{m}, \forall i$. The sensing radius of each agent is $R = 5\text{m}$ and the conflict detection period is $\Delta = 0.05\text{s}$, then one has $R > 2\max_{i \in \mathcal{V}}\{D_i^{\text{sf}} + \Delta v_{i,\max}\}$. The common workspace for the group of agents is depicted in Figure 8.4, where the gray areas represent the static obstacles, the colored triangles represent the initial position and orentation of the agents, and the colored rectangles, marked as $X_l^f, l = 1, 2, \ldots, 9$, represent a set of target regions in the workspace. Initially, $v_i(t_0) = 0, \forall i$. A grid representation with grid

size $0.5\mathrm{m}$ is implemented as workspace discretization, one can verify that $L_c(x) = L_c(x'), \forall Q(x) = Q(x')$.

The specification for each agent is given respectively by $\varphi_1 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_1^f$, $\varphi_2 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_2^f$, $\varphi_3 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_3^f$, $\varphi_4 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_4^f \vee \mathsf{FG}X_5^f$, $\varphi_5 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_6^f$, $\varphi_6 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{FG}X_7^f$, and $\varphi_7 = \mathsf{G}(\mathbb{W} \wedge \neg\mathbb{O}) \wedge \mathsf{F}X_8^f\mathsf{U}\mathsf{F}X_9^f$.

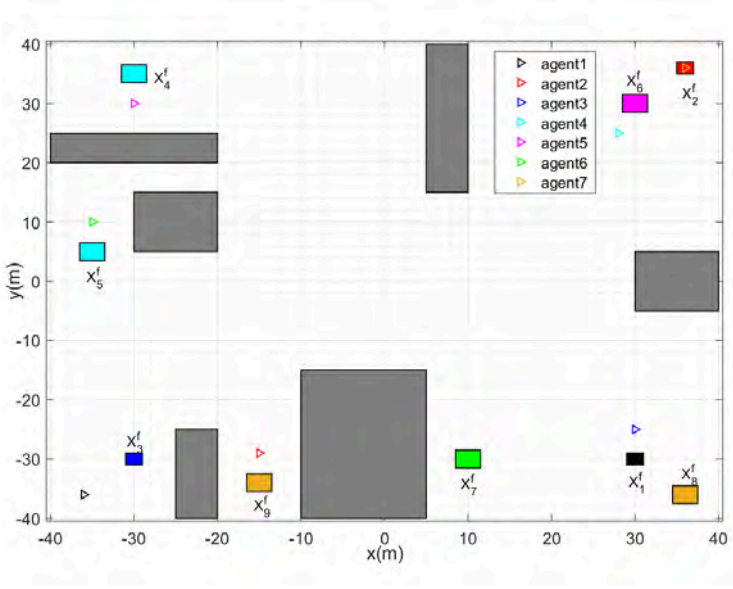

Figure 8.4: The workspace for the group of agents.

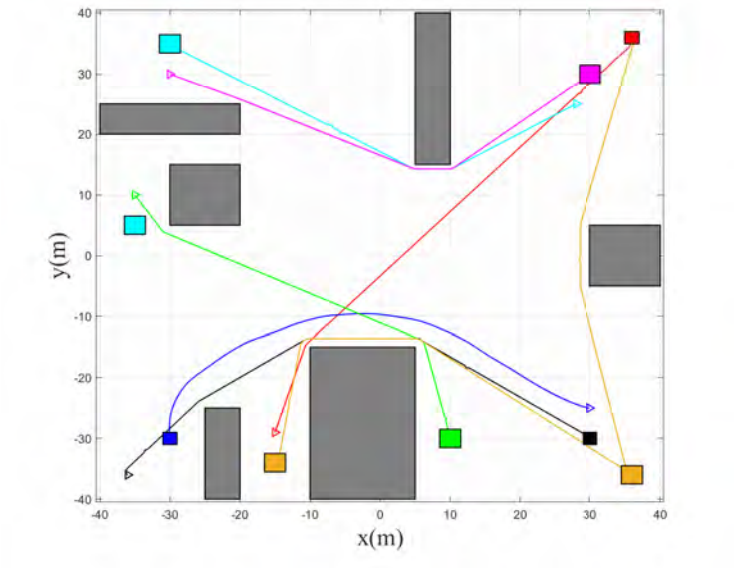

Figure 8.5: The position trajectories for each agent.

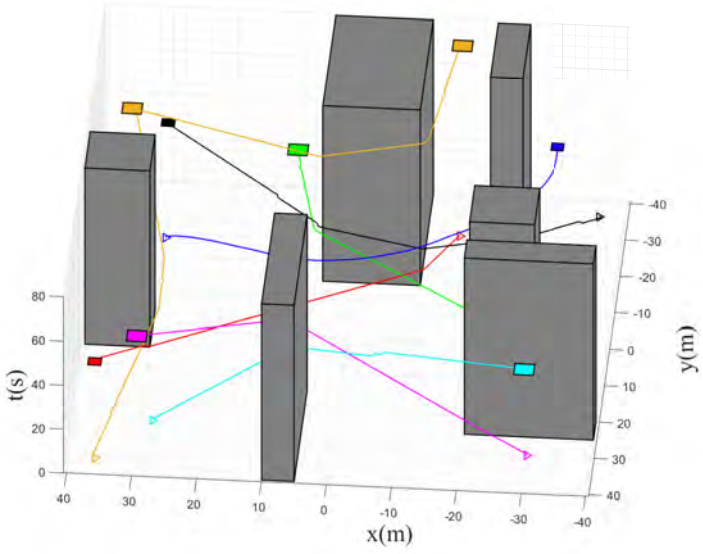

Figure 8.6: The evolution of the position trajectories with respect to time.

The initially planned trajectories are depicted first for each agent, where Figure 8.5 shows the position trajectories and Figure 8.6 shows the evolution of the position trajectories with respect to time. One can see that the initially planned trajectories satisfy the specification φ_i for each agent i. During online implementation, conflicts are detected among agents $(3, 6)$, $(4, 5)$, $(1, 3, 6)$ and $(1, 7)$ at time instants $19.4\text{s}, 22\text{s}, 22.45\text{s}$ and 43.6s, respectively. Whenever conflicts are detected, the planning order assignment module and the trajectory planning module are activated to solve the conflicts. The real-time moving trajectories for each agent are shown in Figs. 8.7-8.8, where Figure 8.7 shows the real-time position trajectories and Figure 8.8 shows the evolution of the position trajectories with respect to time. One can see that conflicts are resolved and each agent completes its specification eventually. A video recording of the real-time implementation can be found here: https://www.youtube.com/watch?v=68mchYhrbhY.

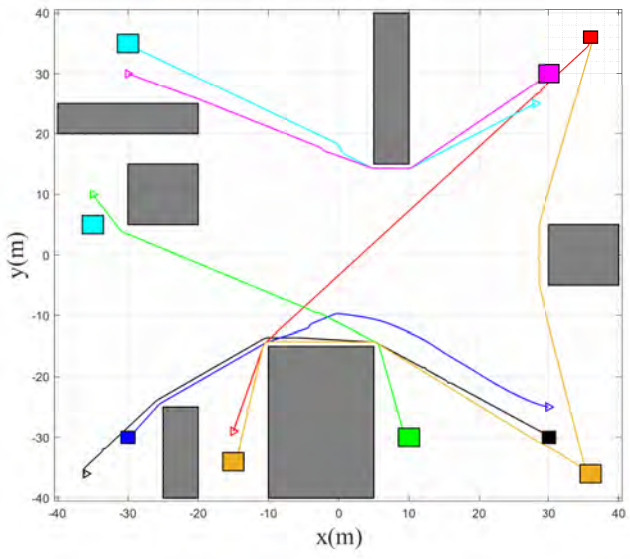

Figure 8.7: The real-time position trajectories for each agent.

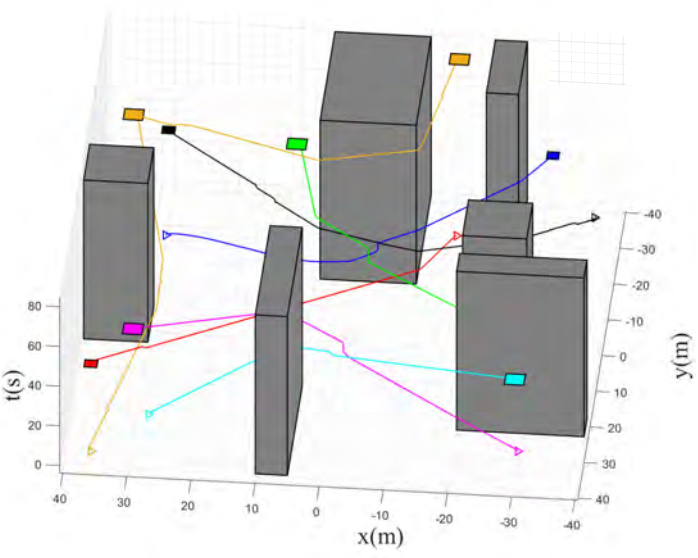

Figure 8.8: The evolution of the real-time position trajectories with respect to time.

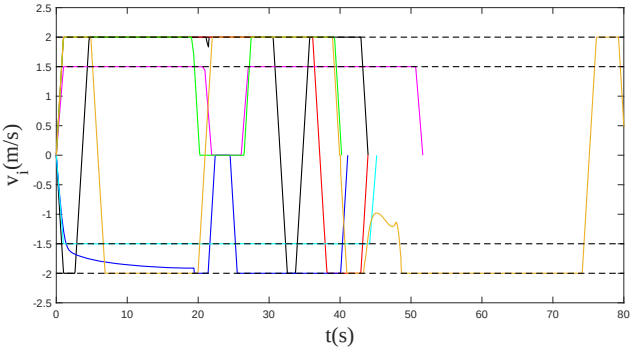

Figure 8.9: The real-time evolution of velocity v_i for each agent, where $v_{i,\max} = 2\text{m/s}, i \in \{1, 2, 3, 6, 7\}$ and $v_{i,\max} = 1.5\text{m/s}, i \in \{4, 5\}$.

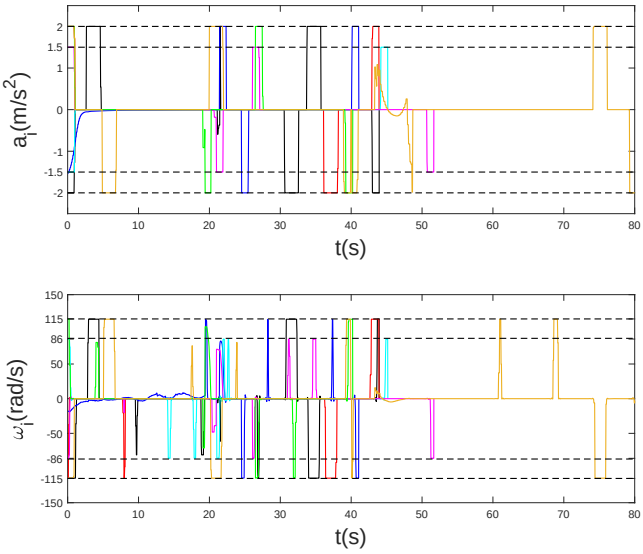

Figure 8.10: The real-time evolution of inputs (a_i, ω_i) for each agent, where $a_{i,\max} = 2\text{m/s}^2, \omega_{i,\max} = 115\text{rad/s}, i \in \{1, 2, 3, 6, 7\}$ and $a_{i,\max} = 1.5\text{m/s}^2, \omega_{i,\max} = 86\text{rad/s}, i \in \{4, 5\}$.

The real-time evolution of velocity v_i and inputs (a_i, ω_i) of each agent i are plotted in Figure 8.9 and Figure 8.10, respectively. One can see that the velocity constraints and the input constraints are satisfied by all agents at any time. All the simulations were run in Matlab 2018b on a DELL laptop of 2.6GHz using Intel Core i7.

8.5 Summary

In this chapter, the online MAMC problem for a group of agents moving in a shared workspace was considered. Under the assumptions that each agent has only local view and local information, and subject to both velocity and input constraints, a fully distributed motion coordination strategy was proposed for steering individual agents in a common workspace, where each agent is assigned an LTL specification. It was shown that the proposed strategy can guarantee collision-free motion of each agent.

Part IV

Conclusions and Future Research

Chapter 9

Conclusions and Future Research

In this chapter, we conclude the results of this thesis and discuss possible directions for future research.

9.1 Conclusions

This thesis presented results on task-oriented control and coordination of MAS under varying constraints. The first three chapters (Chapters 3-5) were concerned with cooperative control of MAS, in which communication constraint and time constraint were considered. Chapters 6-7 studied the control of uncertain systems under temporal logic specifications, where input constraint was considered. Finally, the motion coordination was investigated for MAS with local LTL specifications, where both the communication and the state and input constraints were taken into account. This part of result was presented in Chapter 8.

Chapter 3 studied the distributed ETC of multi-agent consensus. To relax the requirements of continuous communication and controller update, an asynchronous ETC strategy was proposed for the MAS. It was shown that consensus is achieved asymptotically as well as Zeno behavior is excluded.

Chapter 4 proposed a resource-efficient PETC strategy for multi-agent consensus. It was resource efficient in two aspects: 1) both the rate of communication and controller updates were reduced (compared to TTC) and 2) the requirements of continuous sensing and computing were relaxed (compared to ETC). The design of the PETC strategy can be divided into three major stages. In the first stage, an approach on finding the explicit formula

for MASP was proposed. In the second stage, an asynchronous PETC strategy was formulated. Finally, in the third stage, the constraint of limited data rate was considered. It was shown that exponential consensus can be achieved in all the cases.

Chapter 5 considered the target tracking for a leader-follower MAS. We assumed that the leader-follower MAS is subject to a sequence of dynamically activated tasks, each of which is associated with a relative deadline and can be completed at several QoSa levels. By taking into account the reward and cost of satisfying the tasks, a dynamic scheduling strategy was proposed. Based on the dynamic plan, distributed control laws were designed accordingly for leader and follower agents. It was shown that the proposed control laws guarantee the satisfaction of each task at its desired QoS level.

Chapter 6 studied the construction of discrete state-space symbolic models for continuous-time uncertain nonlinear systems. In this chapter, a novel stability notion called η-C-Ω-GPS and its Lyapunov function characterization, and a new relation called robust approximately (bi)simulation relation were proposed. It was shown that an uncertain concrete system, under the condition that there exists an admissible control interface such that the augmented system can be made η-C-Ω-GPS, robust approximately simulates its discrete state-space abstraction.

Chapter 7 was concerned with the robust satisfiability check and online control synthesis of uncertain discrete-time systems under STL specifications. In this chapter, the notion of tTLT was proposed first. Using the tTLT, a sufficient condition was obtained for the robust satisfiability check of the uncertain systems. When the system is deterministic, a necessary and sufficient condition for satisfiability was further obtained. After that, an online control synthesis algorithm was designed. The soundness of the algorithm was proven when the system is uncertain while the completeness was further proven when the underlying system is deterministic.

Chapter 8 investigated the motion coordination problem for MAS, where each agent is assigned an LTL specification. The group of agents were moving in a shared workspace, therefore coordination is essential for safety. Under the assumptions that each agent has only local view and local information, and subject to both state and input constraints, a fully distributed motion coordination strategy was proposed for each agent. It was shown that safety is guaranteed for all agents at any time.

9.2 Future Research

There are several interesting research directions based on the work of this thesis, which are discussed in this section.

Fundamental Tradeoff between Sensing, Computing, Communication, and Control

In Chapters 3-4, ETC and PETC strategies were proposed for MAS. It was shown that there is a tradeoff between the rate of communication and performance. In general, we see that the usage of sensing, computing, communication, and control resources are correlated with each other. However, how to systematically tradeoff between them is still an open problem. To this respect, proper metrics should be proposed to quantify the resource utilization of sensing, computing, communication, and control.

Safe Learning and Control for Uncertain Systems

Chapters 6-7 focus on the robust control approaches for uncertain systems. It is powerful as it provides theoretical guarantee on the property satisfaction, *e.g.,* safety, which is important for robotic systems. It will be interesting to extend the current work to scenarios where the environment is unstructured or the information about the system is partial known or even unknown. In this respect, learning-based approaches, such as Gaussian process regression [166] and policy improvement with integrals [167], are useful and important. A central problem is how to integrate learning and control for ensuring safety and mission completion (*e.g.,* temporal logic specifications).

Efficient Coordination under STL specifications

In Chapter 8, we developed a distributed motion coordination strategy for MAS under local LTL specifications. When considering STL specifications for the MAS, the spatial and temporal constraints embedded in STL formula pose new challenges for motion coordination. The tTLTs proposed in Chapter 7 is a powerful tool for the STL control synthesis. As a natural extension, it is interesting to explore how to apply this tool to efficient MAMC under STL specifications.

Despite of what were discussed above, we are also interested in problems such as efficient algorithms for online computation of reachable sets or tubes. In addition, the experimental validation of the proposed strategies by real-world systems will be pursued.

Bibliography

[1] P. Balaji and D Srinivasan, "An introduction to multi-agent systems," in *Innovations in multi-agent systems and applications-1*, Springer, 2010, pp. 1–27.

[2] M. Taiebat, A. L. Brown, H. R. Safford, S. Qu, and M. Xu, "A review on energy, environmental, and sustainability implications of connected and automated vehicles," *Environmental science & technology*, vol. 52, no. 20, pp. 11 449–11 465, 2018.

[3] W. Ren, "Formation keeping and attitude alignment for multiple spacecraft through local interactions," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 2, pp. 633–638, 2007.

[4] ——, "Distributed attitude alignment in spacecraft formation flying," *International journal of adaptive control and signal processing*, vol. 21, no. 2-3, pp. 95–113, 2007.

[5] Y. Q. Chen and Z. Wang, "Formation control: A review and a new consideration," in *2005 IEEE/RSJ International conference on intelligent robots and systems*, 2005, pp. 3181–3186.

[6] S.-J. Chung and J.-J. E. Slotine, "Cooperative robot control and concurrent synchronization of Lagrangian systems," *IEEE transactions on Robotics*, vol. 25, no. 3, pp. 686–700, 2009.

[7] A. Speranzon, C. Fischione, and K. H. Johansson, "Distributed and collaborative estimation over wireless sensor networks," in *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006, pp. 1025–1030.

[8] N. A. Lynch, *Distributed Algorithms*. Elsevier, 1996.

[9] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on automatic control*, vol. 50, no. 5, pp. 655–661, 2005.

[10] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 693–703, 2007.

[11] S. Mou, M. Cao, and A. S. Morse, "Target-point formation control," *Automatica*, vol. 61, pp. 113–118, 2015.

[12] S. Chakravorty, "Design and optimal control of multi-spacecraft interferometric imaging systems," Ph.D. dissertation, University of Michigan, 2004.

[13] P. Wang, F. Hadaegh, and K Lau, "Synchronized formation rotation and attitude control of multiple free-flying spacecraft," *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 1, pp. 28–35, 1999.

[14] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, 2007.

[15] S. A. Fayazi, A. Vahidi, and A. Luckow, "Optimal scheduling of autonomous vehicle arrivals at intelligent intersections via milp," in *2017 American control conference (ACC)*, 2017, pp. 4920–4925.

[16] H. Surmann and A. Morales, "Scheduling tasks to a team of autonomous mobile service robots in indoor enviroments," *Journal of Universal Computer Science*, vol. 8, no. 8, pp. 809–833, 2002.

[17] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT press, 2008.

[18] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*, Springer, 2004, pp. 152–166.

[19] P. Gastin and D. Oddoux, "Fast LTL to Büchi automata translation," in *International Conference on Computer Aided Verification*, Springer, 2001, pp. 53–65.

[20] C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, and G. J. Pappas, "Symbolic planning and control of robot motion [grand challenges of robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 61–70, 2007.

[21] C. Belta, B. Yordanov, and E. A. Gol, *Formal Methods for Discrete-Time Dynamical Systems*. Springer, 2017, vol. 89.

[22] Y. Zhou, D. Maity, and J. S. Baras, "Timed automata approach for motion planning using metric interval temporal logic," in *2016 European Control Conference (ECC)*, 2016, pp. 690–695.

[23] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theoretical Computer Science*, vol. 410, no. 42, pp. 4262–4291, 2009.

[24] L. Lindemann and D. V. Dimarogonas, "Control barrier functions for signal temporal logic tasks," *IEEE control systems letters*, vol. 3, no. 1, pp. 96–101, 2018.

[25] ——, "Control barrier functions for multi-agent systems under conflicting local signal temporal logic tasks," *IEEE Control Systems Letters*, vol. 3, no. 3, pp. 757–762, 2019.

[26] V. Raman, A. Donzé, D. Sadigh, R. M. Murray, and S. A. Seshia, "Reactive synthesis from signal temporal logic specifications," in *Proceedings of the 18th international conference on hybrid systems: Computation and control*, 2015, pp. 239–248.

[27] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 81–87.

[28] S. Sadraddini and C. Belta, "Robust temporal logic model predictive control," in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2015, pp. 772–779.

[29] C.-I. Vasile, V. Raman, and S. Karaman, "Sampling-based synthesis of maximally-satisfying controllers for temporal logic specifications," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 3840–3847.

[30] D. Aksaray, A. Jones, Z. Kong, M. Schwager, and C. Belta, "Q-learning for robust satisfaction of signal temporal logic specifications," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 6565–6570.

[31] V. Kurtz and H. Lin, "Bayesian optimization for polynomial time probabilistically complete STL trajectory synthesis," *arXiv preprint arXiv:1905.03051*, 2019.

[32] Y. Zhang and Y.-P. Tian, "Consentability and protocol design of multi-agent systems with stochastic switching topology," *Automatica*, vol. 45, no. 5, pp. 1195–1201, 2009.

[33] D. Yang, W. Ren, X. Liu, and W. Chen, "Decentralized event-triggered consensus for linear multi-agent systems under general directed graphs," *Automatica*, vol. 69, pp. 242–249, 2016.

[34] D. Angeli and E. D. Sontag, "Forward completeness, unboundedness observability, and their Lyapunov characterizations," *Systems & Control Letters*, vol. 38, no. 4-5, pp. 209–217, 1999.

[35] D. Angeli, "A Lyapunov approach to incremental stability properties," *IEEE Transactions on Automatic Control*, vol. 47, no. 3, pp. 410–421, 2002.

[36] M. Zamani, G. Pola, M. Mazo, and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1804–1809, 2011.

[37] H. K. Khalil and J. W. Grizzle, *Nonlinear Systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.

[38] J. R. Büchi, "On a decision method in restricted second order arithmetic," in *The Collected Works of J. Richard Büchi*, Springer, 1990, pp. 425–435.

[39] A. Dokhanchi, B. Hoxha, and G. Fainekos, "On-line monitoring for temporal logic robustness," in *International Conference on Runtime Verification*, Springer, 2014, pp. 231–246.

[40] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2011.

[41] M. Donkers and W. Heemels, "Output-based event-triggered control with guaranteed $\mathscr{L}_\infty$-gain and improved event-triggering," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 3246–3251.

[42] Y. Fan, G. Feng, Y. Wang, and C. Song, "Distributed event-triggered control of multi-agent systems with combinational measurements," *Automatica*, vol. 49, no. 2, pp. 671–675, 2013.

[43] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[44] H. Zhang, G. Feng, H. Yan, and Q. Chen, "Observer-based output feedback event-triggered control for consensus of multi-agent systems," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 9, pp. 4885–4894, 2013.

[45] Z. Zhang, F. Hao, L. Zhang, and L. Wang, "Consensus of linear multi-agent systems via event-triggered control," *International Journal of Control*, vol. 87, no. 6, pp. 1243–1251, 2014.

[46] W. Zhu, Z.-P. Jiang, and G. Feng, "Event-based consensus of multi-agent systems with general linear models," *Automatica*, vol. 50, no. 2, pp. 552–558, 2014.

[47] E. Garcia, Y. Cao, and D. W. Casbeer, "Decentralized event-triggered consensus with general linear dynamics," *Automatica*, vol. 50, no. 10, pp. 2633–2640, 2014.

[48] W. Chen and W. Ren, "Event-triggered zero-gradient-sum distributed consensus optimization over directed networks," *Automatica*, vol. 65, pp. 90–97, 2016.

[49] V. Dolk, M Abdelrahim, and W. Heemels, "Event-triggered consensus seeking under non-uniform time-varying delays," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10 096–10 101, 2017.

[50] M. Abdelrahim, R. Postoyan, J. Daafouz, and D. Nešić, "Robust event-triggered output feedback controllers for nonlinear systems," *Automatica*, vol. 75, pp. 96–108, 2017.

[51] P. Lancaster and L. Rodman, *Algebraic Riccati Equations*. Clarendon press, 1995.

[52] H. Zhang, F. L. Lewis, and A. Das, "Optimal design for synchronization of cooperative systems: State feedback, observer and output feedback," *IEEE Transactions on Automatic Control*, vol. 56, no. 8, pp. 1948–1952, 2011.

[53] M. Mazo and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2456–2461, 2011.

[54] W. H. Heemels, M. Donkers, and A. R. Teel, "Periodic event-triggered control for linear systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 4, pp. 847–861, 2012.

[55] W. Heemels and M. Donkers, "Model-based periodic event-triggered control for linear systems," *Automatica*, vol. 49, no. 3, pp. 698–711, 2013.

[56] R. Postoyan, A. Anta, W. Heemels, P. Tabuada, and D Nešić, "Periodic event-triggered control for nonlinear systems," in *52nd IEEE conference on decision and control*, 2013, pp. 7397–7402.

[57] R. Postoyan, P. Tabuada, D. Nešić, and A. Anta, "A framework for the event-triggered stabilization of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 982–996, 2014.

[58] W. Wang, R. Postoyan, D Nešić, and W. M. H. Heemels, "Stabilization of nonlinear systems using state-feedback periodic event-triggered controllers," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 6808–6813.

[59] A. Fu and M. Mazo, "Periodic asynchronous event-triggered control," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 1370–1375.

[60] M. Cao, F. Xiao, and L. Wang, "Second-order consensus in time-delayed networks based on periodic edge-event driven control," *Systems & Control Letters*, vol. 96, pp. 37–44, 2016.

[61] D. P. Borgers, R. Postoyan, A. Anta, P. Tabuada, D. Nešić, and W. Heemels, "Periodic event-triggered control of nonlinear systems using overapproximation techniques," *Automatica*, vol. 94, pp. 81–87, 2018.

[62] X. Meng and T. Chen, "Event based agreement protocols for multi-agent networks," *Automatica*, vol. 49, no. 7, pp. 2125–2132, 2013.

[63] Y. Liu, C. Nowzari, Z. Tian, and Q. Ling, "Asynchronous periodic event-triggered coordination of multi-agent systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 6696–6701.

[64] A. Wang, B. Mu, and Y. Shi, "Consensus control for a multi-agent system with integral-type event-triggering condition and asynchronous periodic detection," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 7, pp. 5629–5639, 2017.

[65] G. Guo, L. Ding, and Q.-L. Han, "A distributed event-triggered transmission strategy for sampled-data consensus of multi-agent systems," *Automatica*, vol. 50, no. 5, pp. 1489–1496, 2014.

[66] E. Garcia, Y. Cao, and D. W. Casbeer, "Periodic event-triggered synchronization of linear multi-agent systems with communication delays," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 366–371, 2016.

[67] X. Yin, D. Yue, and S. Hu, "Adaptive periodic event-triggered consensus for multi-agent systems subject to input saturation," *International Journal of Control*, vol. 89, no. 4, pp. 653–667, 2016.

[68] L. Ding, Q.-L. Han, X. Ge, and X.-M. Zhang, "An overview of recent advances in event-triggered consensus of multiagent systems," *IEEE transactions on cybernetics*, vol. 48, no. 4, pp. 1110–1123, 2017.

[69] D. Nesic, A. R. Teel, and D. Carnevale, "Explicit computation of the sampling period in emulation of controllers for nonlinear sampled-data systems," *IEEE transactions on Automatic Control*, vol. 54, no. 3, pp. 619–624, 2009.

[70] E. Garcia, Y. Cao, H. Yu, P. Antsaklis, and D. Casbeer, "Decentralised event-triggered cooperative control with limited communication," *International Journal of Control*, vol. 86, no. 9, pp. 1479–1488, 2013.

[71] X. Yi, J. Wei, and K. H. Johansson, "Self-triggered control for multi-agent systems with quantized communication or sensing," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 2227–2232.

[72] H. Li, S. Liu, Y. C. Soh, and L. Xie, "Event-triggered communication and data rate constraint for distributed optimization of multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 11, pp. 1908–1919, 2017.

[73] J. Ma, L. Liu, H. Ji, and G. Feng, "Quantized consensus of multiagent systems by event-triggered control," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2018.

[74] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on automatic control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[75] K. Ramamritham and J. A. Stankovic, "Dynamic task scheduling in hard real-time distributed systems," *IEEE software*, vol. 1, no. 3, p. 65, 1984.

[76] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM (JACM)*, vol. 20, no. 1, pp. 46–61, 1973.

[77] G. C. Buttazzo and J. A. Stankovic, *RED: Robust Earliest Deadline Scheduling*. University of Massachusetts, Department of Computer Science, 1993.

[78] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. C. Buttazzo, *Deadline Scheduling for Real-Time Systems: EDF and Related Algorithms*. Springer Science & Business Media, 2012, vol. 460.

[79] H. Aydin, R. Melhem, D. Mosse, and P. Mejía-Alvarez, "Optimal reward-based scheduling for periodic real-time tasks," *IEEE Transactions on Computers*, vol. 50, no. 2, pp. 111–130, 2001.

[80] J. K. Dey, J. Kurose, and D. Towsley, "On-line scheduling policies for a class of IRIS (increasing reward with increasing service) real-time tasks," *IEEE Transactions on Computers*, vol. 45, no. 7, pp. 802–813, 1996.

[81] M. Guinaldo and D. V. Dimarogonas, "A hybrid systems framework for multi agent task planning and control," in *2017 American Control Conference (ACC)*, 2017, pp. 1181–1186.

[82] G. Yang, C. Belta, and R. Tron, "Continuous-time signal temporal logic planning with control barrier functions," in *2020 American Control Conference (ACC)*, 2020, pp. 4612–4618.

[83] L. Lindemann and D. V. Dimarogonas, "Decentralized robust control of coupled multi-agent systems under local signal temporal logic tasks," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 1567–1573.

[84] A. Ilchmann and H. Schuster, "PI-funnel control for two mass systems," *IEEE Transactions on Automatic Control*, vol. 54, no. 4, pp. 918–923, 2009.

[85] C. P. Bechlioulis and G. A. Rovithakis, "Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance," *IEEE Transactions on Automatic Control*, vol. 53, no. 9, pp. 2090–2099, 2008.

[86] L. Macellari, Y. Karayiannidis, and D. V. Dimarogonas, "Multi-agent second order average consensus with prescribed transient behavior," *IEEE Transactions on Automatic Control*, vol. 62, no. 10, pp. 5282–5288, 2016.

[87] C. P. Bechlioulis, M. A. Demetriou, and K. J. Kyriakopoulos, "A distributed control and parameter estimation protocol with prescribed performance for homogeneous lagrangian multi-agent systems," *Autonomous Robots*, vol. 42, no. 8, pp. 1525–1541, 2018.

[88] J. G. Lee, S. Trenn, and H. Shim, *Synchronization with prescribed transient behavior: Heterogeneous multi-agent systems under funnel coupling*, 2019.

[89] Z. Allen-Zhu, Z. Liao, L. Orecchia, and M. Math, "Using optimization to find maximum inscribed balls and minimum enclosing balls," *arXiv preprint arXiv:1412.1001*, 2014.

[90] M. Guo, J. Tumova, and D. V. Dimarogonas, "Communication-free multi-agent control under local temporal tasks and relative-distance constraints," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 3948–3962, 2016.

[91] R. Goebel, R. G. Sanfelice, and A. R. Teel, "Hybrid dynamical systems," *IEEE control systems magazine*, vol. 29, no. 2, pp. 28–93, 2009.

[92] R. Goedel, R. G. Sanfelice, and A. R. Teel, *Hybrid Dynamical Systems: Modeling Stability, and Robustness*, 2012.

[93] C. Lu, J. A. Stankovic, S. H. Son, and G. Tao, "Feedback control real-time scheduling: Framework, modeling, and algorithms," *Real-Time Systems*, vol. 23, no. 1-2, pp. 85–126, 2002.

[94] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Transactions on Automatic Control*, vol. 52, no. 5, pp. 782–798, 2007.

[95] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Springer Science & Business Media, 2009.

[96] P. t. Ramadge and W. Wonham, "Modular feedback logic for discrete event systems," *SIAM Journal on Control and Optimization*, vol. 25, no. 5, pp. 1202–1218, 1987.

[97] R. Kumar and V. K. Garg, *Modeling and Control of Logical Discrete Event Systems*. Springer Science & Business Media, 2012, vol. 300.

[98] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*. Springer Science & Business Media, 2009.

[99] A. Arnold, A. Vincent, and I. Walukiewicz, "Games for synthesis of controllers with partial observation," *Theoretical computer science*, vol. 303, no. 1, pp. 7–34, 2003.

[100] R. Alur, P Madhusudan, and W. Nam, "Symbolic computational techniques for solving games," *International Journal on Software Tools for Technology Transfer*, vol. 7, no. 2, pp. 118–128, 2005.

[101] A. Girard, "Controller synthesis for safety and reachability via approximate bisimulation," *Automatica*, vol. 48, no. 5, pp. 947–953, 2012.

[102] R. Milner, *Communication and Concurrency*. Prentice hall Englewood Cliffs, 1989, vol. 84.

[103] D. Park, "Concurrency and automata on infinite sequences," in *Theoretical computer science*, Springer, 1981, pp. 167–183.

[104] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas, "Discrete abstractions of hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971–984, 2000.

[105] A. Girard, G. Pola, and P. Tabuada, "Approximately bisimilar symbolic models for incrementally stable switched systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 116–126, 2009.

[106] M. Zamani, P. M. Esfahani, R. Majumdar, A. Abate, and J. Lygeros, "Symbolic control of stochastic systems via approximately bisimilar finite abstractions," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3135–3150, 2014.

[107] G. Reissig, A. Weber, and M. Rungger, "Feedback refinement relations for the synthesis of symbolic controllers," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1781–1796, 2016.

[108] K. Mallik, A.-K. Schmuck, S. Soudjani, and R. Majumdar, "Compositional synthesis of finite-state abstractions," *IEEE Transactions on Automatic Control*, vol. 64, no. 6, pp. 2629–2636, 2018.

[109] J. Liu and N. Ozay, "Finite abstractions with robustness margins for temporal logic-based control synthesis," *Nonlinear Analysis: Hybrid Systems*, vol. 22, pp. 1–15, 2016.

[110] A. Saoud, A. Girard, and L. Fribourg, "Contract-based design of symbolic controllers for safety in distributed multiperiodic sampled-data systems," *IEEE Transactions on Automatic Control*, 2020.

[111] A. Girard and G. J. Pappas, "Hierarchical control system design using approximate simulation," *Automatica*, vol. 45, no. 2, pp. 566–571, 2009.

[112] J. Fu, S. Shah, and H. G. Tanner, "Hierarchical control via approximate simulation and feedback linearization," in *2013 American Control Conference*, 2013, pp. 1816–1821.

[113] K. Yang and H. Ji, "Hierarchical analysis of large-scale control systems via vector simulation function," *Systems & Control Letters*, vol. 102, pp. 74–80, 2017.

[114] S. W. Smith, M. Arcak, and M. Zamani, "Approximate abstractions of control systems with an application to aggregation," *Automatica*, vol. 119, p. 109 065, 2020.

[115] M. Zamani, N. van de Wouw, and R. Majumdar, "Backstepping controller synthesis and characterizations of incremental stability," *Systems & Control Letters*, vol. 62, no. 10, pp. 949–962, 2013.

[116] M. Mazo, A. Davitian, and P. Tabuada, "Pessoa: A tool for embedded controller synthesis," in *International Conference on Computer Aided Verification*, Springer, 2010, pp. 566–569.

[117] M. Rungger and M. Zamani, "SCOTS: A tool for the synthesis of symbolic controllers," in *Proceedings of the 19th international conference on hybrid systems: Computation and control*, 2016, pp. 99–104.

[118] M. Kloetzer and C. Belta, "A fully automated framework for control of linear systems from LTL specifications," in *International Workshop on Hybrid Systems: Computation and Control*, Springer, 2006, pp. 333–347.

[119] L. D'Alto and M. Corless, "Incremental quadratic stability," *Numerical Algebra, Control & Optimization*, vol. 3, no. 1, p. 175, 2013.

[120] B. Açıkmeşe and M. Corless, "Observers for systems with nonlinearities satisfying incremental quadratic constraints," *Automatica*, vol. 47, no. 7, pp. 1339–1348, 2011.

[121] E. Bartocci, J. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković, and S. Sankaranarayanan, "Specification-based monitoring of cyber-physical systems: A survey on theory, tools and applications," in *Lectures on Runtime Verification*, Springer, 2018, pp. 135–175.

[122] E. M. Wolff and R. M. Murray, "Optimal control of nonlinear systems with temporal logic specifications," in *Robotics Research*, Springer, 2016, pp. 21–37.

[123] J. Fu and U. Topcu, "Computational methods for stochastic control with metric interval temporal logic specifications," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 7440–7447.

[124] C. I. Vasile and C. Belta, "Sampling-based temporal logic path planning," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4817–4822.

[125] Y. Kantaros and M. M. Zavlanos, "Sampling-based optimal control synthesis for multirobot systems under global temporal tasks," *IEEE Transactions on Automatic Control*, vol. 64, no. 5, pp. 1916–1931, 2018.

[126] M. Chen, Q. Tam, S. C. Livingston, and M. Pavone, "Signal temporal logic meets Hamilton-Jacobi reachability: Connections and applications," in *Workshop on Algorithmic Foundations of Robotics*, 2018.

[127] Y. Gao, A. Abate, F. J. Jiang, M. Giacobbe, L. Xie, and K. H. Johansson, "Temporal logic trees for model checking and control synthesis of uncertain discrete-time systems," *arXiv preprint arXiv:2007.02271*, 2020.

[128] J. Jones, "Abstract syntax tree implementation idioms," in *Proceedings of the 10th conference on pattern languages of programs (plop2003)*, 2003, p. 26.

[129] A. B. Kurzhanski and V. Pravin, *Dynamics and Control of Trajectory Tubes: Theory and Computation*. Springer, 2014.

[130] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.

[131] I. M. Mitchell and J. A. Templeton, "A toolbox of Hamilton-Jacobi solvers for analysis of nondeterministic continuous and hybrid systems," in *International Workshop on Hybrid Systems: Computation and Control*, Springer, 2005, pp. 480–494.

[132] N. Murgovski and J. Sjöberg, "Predictive cruise control with autonomous overtaking," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 644–649.

[133] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari, "Multi-parametric toolbox 3.0," in *2013 European control conference (ECC)*, 2013, pp. 502–510.

[134] L. E. Parker, "Path planning and motion coordination in multiple mobile robot teams," *Encyclopedia of complexity and system science*, pp. 5783–5800, 2009.

[135] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles*, Springer, 1986, pp. 396–404.

[136] D. Panagou, "Motion planning and collision avoidance using navigation vector fields," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2513–2518.

[137] L. Gracia, F. Garelli, and A. Sala, "Reactive sliding-mode algorithm for collision avoidance in robotic systems," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2391–2399, 2013.

[138] H. Farivarnejad, S. Wilson, and S. Berman, "Decentralized sliding mode control for autonomous collective transport by multi-robot systems," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, 2016, pp. 1826–1833.

[139] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 661–674, 2017.

[140] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE control systems letters*, vol. 1, no. 2, pp. 310–315, 2017.

[141] L. Pallottino, V. G. Scordio, A. Bicchi, and E. Frazzoli, "Decentralized cooperative policy for conflict resolution in multivehicle systems," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1170–1183, 2007.

[142] G. A. Bekey, *Autonomous Robots: from Biological Inspiration to Implementation and Control*. MIT press, 2005.

[143] K. Azarm and G. Schmidt, "Conflict-free motion of multiple mobile robots based on decentralized motion planning and negotiation," in *Proceedings of International Conference on Robotics and Automation*, vol. 4, 1997, pp. 3526–3533.

[144] Y. Guo and L. E. Parker, "A distributed and optimal motion planning approach for multiple mobile robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 3, 2002, pp. 2612–2619.

[145] W. Sheng, Q. Yang, J. Tan, and N. Xi, "Distributed multi-robot coordination in area exploration," *Robotics and Autonomous Systems*, vol. 54, no. 12, pp. 945–955, 2006.

[146] C. Liu, C.-W. Lin, S. Shiraishi, and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 1, pp. 18–29, 2017.

[147] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[148] A. Ulusoy, S. L. Smith, X. C. Ding, and C. Belta, "Robust multi-robot optimal path planning with temporal logic constraints," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 4693–4698.

[149] Y. E. Sahin, P. Nilsson, and N. Ozay, "Multirobot coordination with counting temporal logics," *IEEE Transactions on Robotics*, 2019.

[150] M. Guo and D. V. Dimarogonas, "Multi-agent plan reconfiguration under local LTL specifications," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 218–235, 2015.

[151] E. A. Emerson, "Temporal and modal logic," in *Formal Models and Semantics*, Elsevier, 1990, pp. 995–1072.

[152] J.-C. Latombe, *Robot Motion Planning*. Springer Science & Business Media, 2012, vol. 124.

[153] C. I. Vasile and C. Belta, "Reactive sampling-based temporal logic path planning," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 4310–4315.

[154] ——, "Sampling-based temporal logic path planning," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4817–4822.

[155] S. Bhattacharya, V. Kumar, and M. Likhachev, "Search-based path planning with homotopy class constraints.," in *AAAI*, 2010, pp. 1230–1237.

[156] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and computational robotics: new directions*, no. 5, pp. 293–308, 2001.

[157] M. M. Quottrup, T. Bak, and R. Zamanabadi, "Multi-robot planning: A timed automata approach," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 5, 2004, pp. 4417–4422.

[158] T. M. Howard, C. J. Green, and A. Kelly, "Receding horizon model-predictive control for mobile robot navigation of intricate paths," in *Field and Service Robotics*, Springer, 2010, pp. 69–78.

[159] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization.," in *Robotics: science and systems*, Citeseer, vol. 9, 2013, pp. 1–10.

[160] L. T. Biegler and V. M. Zavala, "Large-scale nonlinear programming using IPOPT: An integrating framework for enterprise-wide dynamic optimization," *Computers & Chemical Engineering*, vol. 33, no. 3, pp. 575–582, 2009.

[161] Y. Nie, O. Faqir, and E. C. Kerrigan, "ICLOCS2: Solve your optimal control problems with less pain," 2018.

[162] D. Parsons and J. Canny, "A motion planner for multiple mobile robots," in *Proceedings., IEEE International Conference on Robotics and Automation*, 1990, pp. 8–13.

[163] M. Chen, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Robust sequential trajectory planning under disturbances and adversarial intruder," *IEEE Transactions on Control Systems Technology*, vol. 27, no. 4, pp. 1566–1582, 2018.

[164] M. Otte and E. Frazzoli, "RRT^X: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[165] M. Bennewitz, W. Burgard, and S. Thrun, "Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots," *Robotics and autonomous systems*, vol. 41, no. 2-3, pp. 89–99, 2002.

[166] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2018.

[167] P. Varnai and D. V. Dimarogonas, "Guided policy improvement for satisfying STL tasks using funnel adaptation," *arXiv preprint arXiv:2004.05653*, 2020.