Cooperative Control of Multi-Agent Systems

Cooperative Control of Multi-Agent Systems

Theory and Applications

Edited by
Yue Wang
Clemson University, USA

Eloy Garcia
InfoSciTex Corp, USA
and Air Force Research Laboratory, USA

David Casbeer
Air Force Research Laboratory, USA

Fumin Zhang
Georgia Institute of Technology, USA

WILEY

Registered Offices
John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

Editorial Office
The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Limit of Liability/Disclaimer of Warranty: While the publisher and authors have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor authors shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

Library of Congress Cataloging-in-Publication Data applied for

Hardback ISBN: 9781119266129

Cover Design: Wiley
Cover Image: © Stocktrek Images/Gettyimages

Set in 10/12pt Warnock by SPi Global, Chennai, India

10 9 8 7 6 5 4 3 2 1

To our advisors, students, and family

Contents

List of Contributors

Yongcan Cao
Department of Electrical and Computer Engineering
University of Texas at San Antonio
San Antonio, TX
USA

David Casbeer
The Control Science Center of Excellence
Air Force Research Laboratory
Wright-Patterson AFB, OH
USA

Doo-Hyun Cho
Department of Aerospace Engineering
Korea Advanced Institute of Science and Technology
Daejeon
South Korea

Han-Lim Choi
Department of Aerospace Engineering
Korea Advanced Institute of Science and Technology
Daejeon
South Korea

David A. Copp
Center for Control, Dynamical Systems, and Computation
University of California
Santa Barbara, CA
USA

Jorge Cortes
Department of Mechanical and Aerospace Engineering
University of California
San Diego, CA
USA

Ran Dai
The Aerospace Engineering
Department
Iowa State University
Ames, IA
USA

Eloy Garcia
InfoSciTex Corp, USA and
Air Force Research
Laboratory
Wright-Patterson AFB, OH
USA

João P. Hespanha
Center for Control,
Dynamical Systems, and
Computation
University of California
Santa Barbara, CA
USA

Derek Kingston
The Control Science Center of
Excellence
Air Force Research
Laboratory
Wright-Patterson AFB, OH
USA

Cameron Nowzari
Department of Electrical and
Systems Engineering
University of Pennsylvania
Pennsylvania, PA
USA

George J. Pappas
Department of Electrical and
Systems Engineering
University of Pennsylvania
Pennsylvania, PA
USA

Steven A. P. Quintero
Center for Control,
Dynamical Systems, and
Computation
University of California
Santa Barbara, CA
USA

Sivakumar Rathinam
Department of Mechanical
Engineering
Texas A&M University
College Station, TX
USA

Corey Schumacher
Air Force Research
Laboratory
Wright-Patterson AFB, OH
USA

Rajnikant Sharma
Department of Electrical and
Computer Engineering
Utah State University
Logan, UT
USA

Chuangchuang Sun
The Aerospace Engineering Department
Iowa State University
Ames, IA
USA

Li Wang
Department of Mechanical Engineering
Clemson University
Clemson, SC
USA

Xiaofeng Wang
Department of Electrical Engineering
University of South Carolina
Columbia, SC
USA

Yue Wang
Department of Mechanical Engineering
Clemson University
Clemson, SC
USA

Fumin Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA
USA

Zheqing Zhou
Department of Electrical Engineering
University of South Carolina
Columbia, SC
USA

Preface

This book presents new developments in both the fundamental research and applications in the field of multi-agent systems where a team of agents cooperatively achieve a common goal. Multi-agent systems play an important role in defense and civilian sectors and have the potential to impact on areas such as search and rescue, surveillance, and transportation. Cooperative control algorithms are essential to the coordination among multiple agents and hence realization of an effective multi-agent system. The contents of this book aim at linking basic research and cooperative control methodologies with more advanced applications and real-world problems.

The chapters in this book seek to provide recent developments in the cooperative control of multi-agent systems from a practical perspective. Chapter 1 provides an overview of the state of the art in multi-agent systems and summarizes existing works in consensus control, formation control, synchronization and output regulation, leader and/or target tracking, optimal control, coverage control, passivity-based control, and event-triggered control. Chapter 2 develops sensor placement algorithms for a team of autonomous unmanned vehicles (AUVs) for a path covering problem with monitoring applications in GPS-denied environments. Chapter 3 proposes vision-based output-feedback MPC algorithms with moving horizon estimation for target tracking using fixed-wing unmanned aerial vehicles (UAVs) in measurements gathering and real-time decision-making tasks. Chapter 4 presents the continuous-time projection-based consensus algorithms for multi-UAV simultaneous arrival problem under velocity constraints and finds the convergence rate of the proposed consensus algorithms. Chapter 5 discusses the asset-based weapon-target assignment (WTA) problem to find the optimal launching time of a weapon to maximize the sum of asset values with time-dependent rewards. Chapter 6 presents a coordinated decision algorithm where a group of UAVs is assigned to a set of targets to minimize some cost terms associated with the mission. Chapter 7 provides a formal analysis of event-triggered control and communication techniques for multi-agent average consensus problems. Chapter 8 solves network topology design and identification problems for dynamic networks. Chapter 9 discusses stochastic interaction for distributed multi-agent systems and presents results about the probabilities to achieve multi-agent coordination. Finally, Chapter 10 addresses a cooperative coverage control problem employing wheeled mobile robots (WMRs) and UAVs.

August 2016

Yue Wang
Clemson University

Eloy Garcia
Air Force Research Laboratory,
Wright-Patterson AFB

David Casbeer
Air Force Research Laboratory,
Wright-Patterson AFB

Fumin Zhang
Georgia Institute of Technology

Acknowledgment

The editors would like to thank the authors of all the chapters and reviewers who worked together on this book. A special acknowledgment goes to all the graduate students in the Interdisciplinary & Intelligent Research (I^{2}R) Laboratory in the Mechanical Engineering Department at Clemson University, who assisted the editors to review chapters and provided useful feedbacks to improve the quality of the book. The first editor would like to thank the support from the National Science Foundation under Grant No. CMMI-1454139. The editors would also like to thank Wiley and its staff for the professional support.

1

Introduction

Yue Wang[1], Eloy Garcia[2], David Casbeer[2] and Fumin Zhang[3]

[1] *Department of Mechanical Engineering, Clemson University, Clemson, SC, USA*
[2] *The Control Science Center of Excellence, Air Force Research Laboratory, Wright-Patterson AFB, OH, USA*
[3] *School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA*

1.1 Introduction

Many military and civilian applications require a team of agents to coordinate with each other to perform specific tasks without human intervention. In those systems, individual agents (e.g., unmanned underwater/ground/aerial vehicles) have limited capabilities due to short sensing and communication ranges, and small computational power. However, their collective behavior exhibits significant advantages compared to a single sophisticated agent, including large-scale spatial distribution, robustness, high scalability, and low cost [1]. The deployment of large-scale multi-agent systems with constrained costs and smaller sizes can thus achieve tasks that are otherwise unable to be finished by a single agent. Teams of engineered multi-agent systems can collect and process data and perform tasks cooperatively [2–8]. Multi-agent systems play an important role in a wide range of applications such as search and rescue [9], tracking/classification [10–14], surveillance [15, 16], space exploration [17], and radiation shielding and site clearing [18]. Multi-agent systems have also been considered and utilized in fields such as cooperative mobile robotics [19], distributed artificial intelligence and computing [20–22], wireless sensor

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

networks [23], biology [24], social study [25], smart grids [26], traffic management [27, 28], and supply-chain management [29]. Therefore, the use of multi-agent system technologies in both everyday modern society and national defense and homeland security is bound to tremendously increase. In this book, we aim to provide an overview of recent progresses made in the cooperative control of multi-agent systems on both fundamental theory development as well as applications.

In the control community, multi-agent system theory has focused on developing vehicle motion control laws for various tasks including consensus and formation control [2, 30–43], coverage control [44–48], target search and tracking [3–5, 49, 50], task allocation problems [25, 51–53], sensor management problems [14], output regulation [54, 55], optimization [56], and estimation. Three types of control schemes for multi-agent systems have been proposed in the open literature, that is, centralized [57], decentralized [58], and distributed multi-agent control [1]. The centralized control scheme assumes global knowledge of the multi-agent system and seeks to achieve some control objective considering all agents' states, which inevitably suffers from the scalability issue. The decentralized control scheme computes control actions based only on an agent's local information while the more popular distributed control scheme takes both the agent's own information and neighboring agents' information into account to calculate the control action. Both the decentralized and distributed control algorithms provide scalable solutions and can be implemented under minimal connectivity properties. On the other hand, connectivity preserving protocols are developed for multi-agent systems to keep connected and hence guarantee motion stability [59, 60]. The problem has been considered in scenarios such as flocking [61, 62], rendezvous [59, 63], and formation control [64, 65]. The control hierarchy for multi-agent systems can be categorized into two classes, that is, top-down and bottom-up methodologies [66]. The top-down scheme assigns an overarching objective for the multi-agent system and designs control action for each individual agent to achieve this objective. The top-down multi-agent task decomposition is often difficult. While the bottom-up scheme directly defines each individual agent's local control action and their cooperation protocol, which however cannot guarantee any global objective. The paper [67] provides an overview of progresses made in the distributed multi-agent coordination. The books [64, 68] provide an introduction to the distributed control of multi-agent systems. The book [1] discusses the distributed control of multi-agent systems from four main themes, or dimensions: distributed control and computation, adversarial interactions, uncertain evolution, and complexity management. A special category of multi-agent systems, multi-robot systems, has become one of the most important areas of research in robotics [19]. Significant advance has been made in distributed control and collaboration of multi-robot systems in control theory and artificial intelligence [68–70]. There are a considerable amount of works on multi-agent consensus and formation control, and synchronization. We briefly summarize the main results as follows.

The multi-agent consensus control problem ensures that a group of mobile agents stays connected and reaches agreement while achieving some performance objective [64]. The papers [71, 72] provide a good survey of consensus problems in multi-agent cooperative control. In [64], the consensus problem is considered over dynamic interaction graphs by adding appropriate weights to the edges in the graphs. Theoretical results regarding consensus seeking under both time-invariant and dynamically changing information exchange topologies are summarized. Applications of consensus protocols to multi-agent coordination are investigated. In [73, 74], consensus algorithms are extended for second-order nonlinear dynamics in a dynamic proximity network. Necessary and sufficient conditions are given to ensure second-order consensus. In [75], leader-following consensus algorithms are developed for a linear multi-agent system on a switching network, where the input of each agent is subject to saturation. In [76], multi-agent consensus based on the opinion dynamics introduced by Krause is studied. A new proof of convergence is given with all agents in the same cluster holding the same opinion (represented by a real number). Lower bounds on the inter-cluster distances at a stable equilibrium are derived. In [33], multi-agent consensus is considered for an active leader-tracking problem under variable interconnection topology. The effects of delays on multi-agent consensus have been considered in [77].

The paper [78] provides a survey of formation control of multi-agent systems. The existing results are categorized into position-, displacement-, and distance-based control. The finite-time formation control for nonlinear multi-agent systems is investigated in [43]. A small number of agents navigate the whole team based on the global information of the desired formation while the other agents regulate their positions by the local information in a distributed manner. A class of nonlinear consensus protocols is first ensured and then applied to the formation control. In [79], a model-independent coordination strategy is proposed for multi-agent formation control in combination with tracking control for a virtual leader. The authors show that the formation error can be stabilized if the agents can track their respective reference points perfectly or if the tracking errors are bounded. In [80], a decentralized cooperative controller for multi-agent formation control and collision avoidance is developed based on the navigation function formalism. The control law is designed as the gradient of a navigation function whose minimum corresponds to the desired formation. Multi-agent formation control with intermittent information exchange is considered in [81]. Energy-based analysis is utilized to derive stability conditions. The paper [82] investigates rotating consensus and formation control problems of second-order multi-agent systems based on Lyapunov theory. Both theoretical and experimental results are presented in [42] on multi-agent decentralized control that achieves leader–follower formation control and collision avoidance for multiple nonholonomic robots.

In [83], synchronization approach is developed for trajectory tracking of multiple mobile robots while maintaining time-varying formations. In [84], synchronization algorithms are designed in a leader–follower cooperative tracking control problem where the agents are modeled as identical general linear systems on a digraph containing a spanning tree. The control framework includes full-state feedback control, observer design, and dynamic output feedback control. In [54], a distributed control scheme is adopted for robust output regulation in a multi-agent system where both the reference inputs and disturbances are generated by an exosystem. In [55], the output regulation problem is extended to multi-agent systems where a group of subsystems cannot access the exogenous signal. In [85], output consensus algorithms are developed for heterogeneous agents with parametric uncertainties. The multi-agent output synchronization problem is also studied in [86] where the coupling among the agents is nonlinear and there are communication delays. In [87], a general result for the robust output regulation problem has been studied for linear uncertain multi-agent systems. In [88], finite-time synchronization is proposed for a class of second-order nonlinear homogenous multi-agent systems with a leader–follower architecture. A finite-time convergent observer and an observer-based finite-time output feedback controller are developed to achieve the goal.

In [89], distributed tracking control is developed for linear multi-agent systems and a leader whose control input is nonzero, bounded, and not available to any follower. The paper [90] considers multi-agent tracking of a high-dimensional active leader, whose state not only keeps changing but also may not be measured. A neighbor-based local state-estimator and controller is developed for each autonomous following agent. A collision-free target-tracking problem of multi-agent robot system is considered in [91], where a cost function using a semi-cooperative Stackelberg equilibrium point component with weights tuned by a proportional-derivative (PD)-like fuzzy controller is formulated. The distributed finite-time tracking control of second-order multi-agent systems is considered in [92]. Observer-based state feedback control algorithms are designed to achieve finite-time tracking in a multi-agent leader-follower system and extended to multiple active leaders. There are also a lot of works focusing on multi-agent target tracking. In [93], the optimal sensor placement and motion coordination strategies for mobile sensor networks are developed in a target-tracking application. Gradient-descent decentralized motion planning algorithms are developed in [94] for multiple cooperating mobile sensor agents for the tracking of dynamic targets. The problem of target tracking and obstacle avoidance for multi-agent systems is considered in [95]. A potential function-based motion control algorithm is proposed to solve the problem where multiple agents cannot effectively track the target while avoiding obstacles at the same time.

The book [96] gives an overview of optimal and adaptive control methods for multi-agent systems. In [56], a distributed subgradient method is developed to solve a multi-agent convex optimization problem where every agent minimizes its own objective function while exchanging information locally with other agents in the network over a time-varying topology. An inverse optimality-based distributed cooperative control law is designed in [97] to guarantee consensus and global optimality of multi-agent systems, where the communication graph topology interplays with the agent dynamics. The work [98] applies stochastic optimal control theory to multi-agent systems, where the agent dynamics evolve with Wiener noise. The goal is to minimize some cost function of different agent–target combinations so that decentralized agents are distributed optimally over a number of targets. An optimal control framework for persistent monitoring using multi-agent systems is developed in [99] to design cooperative motion control laws to minimize an uncertainty metric in a given mission space. The problem leads to hybrid systems analysis, and an infinitesimal perturbation analysis (IPA) is used to obtain an online solution.

Coverage control considers the problem of fully covering a task domain using multi-agent systems. The problem can be solved by either deploying multiple agents to optimal locations in the domain or designing dynamic motion control laws for the agents so as to gradually cover the entire domain. The former solutions entail locational optimization for networked multi-agent systems. Voronoi diagram–based approaches are introduced in [100] to develop decentralized control laws for multiple vehicles for optimal coverage and sensing policies. Gradient descent–based schemes are utilized to drive a vehicle toward the Voronoi centeriod for optimal localization. In [101], the discrete coverage control law is developed and unified with averaging control laws over acyclic digraphs with fixed and controlled-switching topology. In [102], unicycle dynamics are considered and the coverage control algorithms are analyzed with an invariance principle for hybrid systems. The latter solutions focus on the case when the union of the agents' sensor cannot cover the task domain and hence dynamic motion control needs to be designed so that the agents can travel and collaboratively cover the entire domain [103]. A distributed coverage control scheme is developed in [104, 105] for mobile sensor networks, where the sensor has a limited range and is defined by a probabilistic model. A gradient-based control algorithm is developed to maximize the joint detection probabilities of random events taking place. Effective coverage control is developed to dynamically cover a given 2D region using a set of mobile sensor agents [46, 106]. Awareness-based coverage control has been proposed to dynamically cover a task domain based on the level of awareness an agent has with respect to the domain [48]. The paper [107] extends the awareness coverage control by defining a density function that characterizes the importance of each point in the domain and the desired awareness coverage level as a nondecreasing differentiable function of the density distribution. In [108], awareness and persistence coverage control are addressed simultaneously so that the mission domain can be covered periodically while the desired awareness is satisfied.

Passivity-based control approaches have also been developed to guarantee the stability of multi-agent systems [109]. Passivity is an energy-based method and a stronger system property that implies stability [110, 111]. A system is passive if it does not create energy, that is, the stored energy is less than the supplied energy. The negative feedback interconnection and parallel interconnection of passive systems are still passive. The paper [112] discusses the stabilization and output synchronization for a network of interconnected nonlinear passive agents by characterizing the information exchange structure. In [113], a passivity-based cooperative control is developed for multi-agent systems and the group synchronization is proved with the proposed backstepping controller using the Krasovskii–LaSalle invariance principle. The paper [114] introduces a discrete-time asymptotic multi-unmanned aerial vehicle (UAV) formation control that uses a passivity-based method to ensure l_2^m stability in the presence of overlay network topology with delays and data loss. Passivity-based motion coordination has also been used in [115] for the attitude synchronization of rigid bodies in the leader–follower case with communication delay and temporary communication failures. The work [116] uses the multiple Lyapunov function method for the output synchronization of a class of networked passive agents with switching topology. The concept of stochastic passivity is studied for a team of agents modeled as discrete-time Markovian jump nonlinear systems [117]. Passivity-based approaches have also been widely used in the bilateral teleoperation of robots and multi-agent systems. A good amount of work has utilized the scattering wave transformation and two-port network theory to provide stability of the teleoperation under constant communication delays for velocity tracking. A passifying PD controller is developed in [118] for the bilateral teleoperation of multiple mobile slave agents coupled to a single master robot under constant, bounded communication delays. The paper [119] extends the passivity-based architecture to guarantee state (velocity as well as position) synchronization of master/slave robots without using the wave scattering transformation. Passivity-based control strategies are also utilized for the bilateral teleoperation of multiple UAVs [120].

Extensive results presenting algorithms and control methodologies for multi-agent systems cooperation rely on continuous communication between agents. Continuous actuation and continuous measurement of local states may be restricted by particular hardware limitations. A problem in many scenarios is given by the limited communication bandwidth where neighboring agents are not capable of communicating continuously but only at discrete time instants. Limitations and constraints on inter-agent communication may affect any multi-agent network. Consensus problems, in particular, have been analyzed in the context of noncontinuous actuation and noncontinuous inter-agent communication. Several techniques are devised in order to schedule sensor and actuation updates. The sampled-data (periodic) approach [121–123], and [124] represents a first attempt to address these issues. The implementation of periodic communication represents a simple and practical tool that addresses the continuous communication constraint. However, an important drawback of periodic transmission is that it requires synchronization between the agents in two similar aspects: sampling period and sampling time instants, both of which are difficult to meet in practice. First, most results available require every agent to implement the same sampling period. This may not be achievable in many networks of decentralized agents and it is also difficult to globally redefine new sampling periods. Second, not only the agents need to implement the same sampling periods, but also they need to transmit information all at the same time instants. Under this situation each agent is also required to determine the time instants at which it needs to transmit relevant information to its neighbors. Even when agents can adjust and implement the same sampling periods, they also need to synchronize and transmit information at the same time instants for the corresponding algorithms to guarantee the desired convergence properties. Besides being a difficult task to achieve in a decentralized way, the synchronization of time instants is undesirable because all agents are occupying network resources at the same time instants. In wireless networks, the simultaneous transmission of information by each agent may increase the likelihood of packet dropouts since agents that are supposed to receive information from different sources may not be able to successfully receive and process all information at the same time.

Therefore, event-triggered and self-triggered controls for multi-agent systems have been considered for agents with limited resources to gather information and actuate. The event-triggered schemes allow each agent to only send information across the network intermittently and independently determine the time instants when they need to communicate [57]. The use of event-triggered control techniques for decentralized control and coordination has spurred a new area of research that relaxes previous assumptions and constraints associated with the control of multiple agents. In event-triggered control [125–130], a subsystem monitors its own state and transmits a state measurement to the non-collocated controller only when it is necessary, that is, only when a measure of the local subsystem state error is above a specified threshold. In general, the state error measures the difference between the current state and the last transmitted state value. The controller transmits an update by examining the measurement errors with respect to some state-dependent threshold and hence requires continuous monitoring of state error. In many instances, it is possible to reduce communication instances using event-triggered communication with respect to periodic implementations. This is of great importance in applications where bandwidth or communication resources are scarce. Consensus problems where all agents are described by general linear models [131, 132], have been studied assuming continuous communication among agents. Event-triggered control and communication methods for agents with linear dynamics were recently studied in [133–138]. Event-triggered control methods have also been applied to analyze consensus problems with limited actuation rates. In [139], agents with single integrator dynamics are considered and an event-triggered control technique is implemented in order for each agent to determine the time instants to update their control inputs. Continuous exchange of information is assumed in [139] and the event-triggered controller is only used to avoid continuous actuation at each node. In general, the decentralized event-triggered consensus problem with limited communication is a more challenging problem than the event-triggered control for limiting actuation updates. The main reason is that agents need to take decisions (on when to transmit their state information) based on outdated neighbor state updates. In this scenario, each agent has continuous access to its own state; however, it only has access to the last update transmitted by its neighbors. Several approaches for the event-triggered consensus with limited communication are documented in [140–145]. In this sense, event-triggered control provides a more robust and efficient use of network bandwidth. Its implementation in multi-agent systems also provides a highly decentralized way to schedule transmission instants, which does not require synchronization compared to periodic sampled-data approaches. Different problems concerning the transmission of information in multi-agent networks such as communication delays and packet dropouts have been explicitly addressed using event-triggered control methods [146]. In the extended self-triggered control, each agent will compute its next update time based on the available information from the last sampled state, without the necessity to keep track of the state error in order to determine when a future sample of the state should be taken. In [140], an event-based scheduling is developed for multi-agent broadcasting and asymptotic convergence to average consensus is guaranteed. This paradigm has also been extended to distributed estimation and optimization [147].

1.2 Chapter Summary and Contributions

Chapter 2 develops sensor deployment algorithms for a team of autonomous unmanned vehicles (AUVs) for path coverage problem with monitoring applications in GPS-denied environments. The approach used in this chapter tracks the AUV position in GPS-denied environments by analyzing the radio signals received from a suitably positioned network of proxy landmarks. This problem is referred to as the landmark placement problem (LPP) and it is required to use minimum number of landmarks to cover the entire path of the AUV. Two α-approximate ($\alpha = 13$ and 5, respectively) algorithms are proposed to solve the LPP in polynomial time and provide solutions whose cost is at most α times from the optimum. It is assumed that a target in a vehicle's path is defined to be covered by a landmark and the distance between a target and a landmark is at most equal to R. A greedy algorithm is first proposed for a simpler LPP where all the targets lie within a vertical strip of width equal to $\sqrt{3}R$ and the landmarks are restricted to be on a single, vertical line. The algorithm is then extended to a general LPP by partitioning the plane into vertical strips of width $\sqrt{3}R$ with approximation ratio $\alpha = 13$. The second approximate algorithm with $\alpha = 5$ is developed based on a 4-approximation algorithm for a unit disc problem. Two phases are involved in this algorithm: (i) identification of a subset of targets using a simple greedy algorithm and (ii) addition of landmarks in the vicinity of each target in the subset. Both theoretical guarantees and numerical simulations are provided to show the performance of the proposed approximation algorithms.

Chapter 3 proposes vision-based cooperative target tracking control laws for two fixed-wing UAVs in measurements gathering and real-time decision-making tasks. To mitigate a single UAV's inability to maintain close proximity to a target and hence obtain accurate measurements for tracking purpose, multiple UAVs are deployed for cooperative target tracking. In this chapter, the standoff target tracking approach is used where two UAVs orbit the target at a nominal standoff distance while maintaining orthogonal viewing angles so as to minimize the joint/fused geolocation error covariance. The work promotes a practical solution that yields robust coordination under the following realistic conditions: unknown constant wind, non-negligible roll dynamics with roll-angle setpoint limits, unpredictable and evasive target motion, and the availability of only noisy, partial information of the overall system's states. The motion of the individual vehicles is optimized and robust so as to gather the best joint measurements of a given quantity, object, or area of interest and take into account real-world conditions, such as environmental disturbances and unmodeled dynamics. An output-feedback control approach is deployed to achieve the desired robustness, and a fourth-order Dubins vehicle model with roll dynamics is considered. The tracking solution incorporates adaptive estimates of the wind into the online model predictive control (MPC) and moving horizon estimation (MHE) optimization. The MPC and MHE are combined into a single min–max optimization, that is, a desired cost function is maximized with respect to disturbance and measurement noise variables and minimized with respect to control input variables. Simulations are performed using aircraft models having six degrees of freedom and target logs taken from live tracking experiments.

Chapter 4 discusses how to find the convergence rate of continuous-time consensus algorithms for multi-UAV simultaneous arrival problem. The requirement is that the UAVs must achieve consensus on the expected time-to-arrival (ETA) before any actual arrivals. Assume that a team of agents are required to simultaneously visit some prespecified targets and the path for each individual agent to follow has been precomputed. To arrive at their targets at the same time, agents have to adjust their velocities during the motion, based on the information communicated with their neighbors. Real-time planning schemes need to be developed to overcome the uncertainties due to UAVs flying in dynamical environments. This chapter considers the consensus-based simultaneous arrival problem with fixed velocity constraints under a connected and undirected communication graph. It is challenging to analyze the stability and the convergence rate of the consensus algorithms. Each UAV estimates its own ETA and communicates it with its neighbors in real-time so that they can reach consensus on the ETA. A continuous-time projection operator is introduced to ensure smoothness of the state and input trajectories, when saturation happens due to the velocity constraints. The projection-based operator is used to enforce the constraints on the velocity. The convergence of the resulting closed-loop system is proved. The aforementioned consensus algorithm shows asymptotic property, and hence simultaneous arrival will be reached in an infinite amount of time. In practice, the length of the paths is always finite, and the agents are required to achieve consensus in finite amount of time. Hence, the ϵ-consensus approach is further investigated for practical consideration. An upper bound on the convergence rate is derived when ϵ-consensus can be achieved. A sufficient condition in terms of the path length and UAVs' minimal and maximal velocity is presented to guarantee feasibility of the simultaneous arrival problem.

Chapter 5 addresses the weapon–target assignment (WTA) problem, that is, how to assign defensive weapons to intersect the aimed targets to minimize the damage of assets or maximize the probability of destroying the target and hence the damage of targets. This work particularly focuses on time-dependent WTA (TSWTA) problems that seek to find the optimal launching time of a weapon to maximize the sum of asset values after defensive weapons are assigned to corresponding targets. The TSWTA problem is formulated as a mixed-integer nonlinear program (MINLP), under the assumption that target–assets engagements are independent of weapon–target engagement. It is shown that the TSWTA exhibits the monotonically nondecreasing property similar to other WTA problems. Based on this property, the TSWTA can be formulated as the problem that maximizes the nondecreasing objective function under a partition matroid constraint. A provable suboptimality lower bound of the value achieved by a greedy heuristic maximization algorithm is obtained. Computational experiments are also conducted to demonstrate good performance achieved by the proposed heuristic algorithms for this combinatorial optimization problem.

Chapter 6 presents a cooperative decision problem in which a group of UAVs is tasked to eliminate a set of targets while minimizing different cost terms during the duration of the mission. The environment where the mission is performed contains a set of threats representing radar sites that are able to identify and potentially harm the UAVs. The radar sites are more effective in identifying a given UAV if the UAV travels near the threat position. In a first instance of the problem, each UAV needs to independently compute its own optimal path in order to reach the destination point where a main target is located. The optimal path is the one that minimizes a combined cost that captures path length and threat risks. In order to minimize threat risk, the approach followed in Chapter 6 is to design a Voronoi diagram using the threat positions. This means that the UAVs minimize exposure to threats when traveling along the edges of the Voronoi diagram. The optimal trajectory to reach a main target is transformed into a graph search where the weights of each edge are determined by two factors: the length of the edge and the threat risk that the UAV is exposed to by traveling along that edge. The problem is further extended by endowing the UAVs with extra munitions that can be used to eliminate a subset of threats. A problem of distributed assignment of threats is then formulated and solved by identifying individual optimal decisions and by implementing a distributed consensus-based auction algorithm. The assignment of threats to eliminate is performed sequentially in order for UAVs to take advantage of other UAVs decisions and assignments. In this way, cooperation among UAVs is induced since the cost of the new optimal path of each UAV can be significantly improved not only by its own decisions but also by traveling along paths where previous threats have already been eliminated by other UAVs. The timing constraints associated with the distributed decisions and assignments of threats is explicitly considered in Chapter 6. In addition, the existence of multiple main targets is considered and different approaches to assign UAVs to main targets are proposed.

Chapter 7 studies event-triggered control and communication techniques for multi-agent systems coordination. This work provides an overview of several event-triggered control techniques to achieve multi-agent coordination. The focus of the chapter is on the problem of average consensus, where a group of agents seek to agree on the average of their initial states. An introduction is provided for event-triggered control strategies applied to consensus problems. Centralized event-triggered control, decentralized event-triggered communication and control, periodic event-triggered coordination are introduced in detail. A detailed comparison among different techniques is presented. Several aspects associated with the use of these techniques such as decentralization, type of event threshold employed, and continuous sensing of local states are analyzed. The chapter provides formal analysis of several controllers and event–threshold implementations. The conditions necessary to achieve average consensus are also studied. Finally, open problems within this important area of research are addressed.

Chapter 8 solves network topology design (NTD) and identification problems. For the NTD problem, a limited number of edges are considered, and these edges and the associated edge weights are optimally allocated among multiple agents to improve certain network performance. While the network topology identification (NTI) problem is to satisfy the response between specified input and observed output. Solving both problems involves determining binary variables and the combination of them is exponentially increasing. The cardinality constraint on the edge set for the NTD problem is handled as a rank constraint on the to-be-determined matrix, and the NTD problem is formulated as a rank-constrained optimization problem. The approach for solving NTI problem handles unknown binary variables as continuous variables by adding a quadratic constraint on each binary variable and then reformulates the problem as a quadratically constrained quadratic programming (QCQP) problem, which can be equivalently transformed into a rank-one constrained optimization problem. Then for both NTD and NTI problems, an iterative rank minimization algorithm is proposed to solve the uniformly formulated rank-constrained optimization problems, where each iteration is formulated as a convex optimization problem.

Chapter 9 considers stochastic interaction among groups of agents and presents relevant results about the probabilities to achieve coordination on variables of interest. The results presented in the chapter are roughly divided into two parts. The first part is concerned with fixed interaction communication graphs. In this case, the agents select the static undirected communication links and, therefore, the fixed communication graph, from a set of available candidates. In terms of communication graphs, the set of candidate graphs considered is the set of all possible undirected graphs. For each interaction graph, only one adjacency (or Laplacian) matrix is associated with it in order to uniquely define the interaction among agents. A lower bound on the probability of coordination is determined under this scenario. In addition, it is shown that the probability of coordination is strictly increasing as the number of agents increase. In the second part of Chapter 9, the probability of coordination is analyzed for the case where the directed interaction graph is switching. In this case, the communication links are not static. Instead, directed links between any two agents appear and disappear as time evolves. Under this scenario, it is demonstrated that coordination with probability 1, coordination in probability, and coordination in the *r*th mean are equivalent.

Chapter 10 develops distributed motion control algorithms of heterogeneous multi-agent systems for the coverage control of unknown and large-scale (i.e., the union of sensor regions cannot cover the entire domain) environments. To achieve full coverage of an unknown domain, the coverage task is decomposed into two distinct, however closely related, subtasks, that is, domain boundary tracking and coverage control. This work considers UAVs with down-facing board view cameras for the boundary tracking task and wheeled mobile robots (WMRs) for the coverage control task. The UAVs can move quickly and maintain a minimum altitude; however, it cannot lift a heavy payload and has to delegate the analysis of its sensor data to an off-board computer. Meanwhile, the WMRs move relatively slow but can carry more sensors and perform onboard computation. Nonholonomic constraints of the robots and nonisotropic sensor models are considered in the control law development for practical applications. A complete communication strategy between the UAVs and WMRs is discussed for information exchange. The inner (autopilot) and outer (wall follower motion control) loop feedback control strategy is adopted for the UAVs. Awareness-based coverage control law is developed for the WMRs based on dynamic awareness dynamics, 2D Leibniz rule, and practical consideration of actuation saturation. The state of awareness represents how aware each coverage robot is of the event occurring at the domain. Intermittent state updates between neighboring robots are considered for distributed multi-agent systems for the mapped part of the task domain. The awareness coverage error metrics are defined and proved to converge to zero under the proposed motion control strategies using Lyapunov-like analysis. A perturbation control law is deployed if the robot is trapped in a local minimum.

References

1 Shamma, J.S. (2007) *Cooperative Control of Distributed Multi-Agent Systems*, Wiley Online Library.

2 Léchevin, N., Rabbath, C.A., and Lauzon, M. (2009) A decision policy for the routing and munitions management of multiformations of unmanned combat vehicles in adversarial urban environments. *IEEE Transactions on Control Systems Technology*, **17** (3), 505–519.

3 Jin, Y., Liao, Y., Minai, A.A., and Polycarpou, M.M. (2006) Balancing search and target response in cooperative unmanned aerial vehicle (UAV) teams. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, **36** (3), 571–587.

4 Flint, M., Polycarpou, M., and Fernández-Gaucherand, E. (2002) Cooperative control for multiple autonomous UAV's searching for targets, in Proceedings of the 41st IEEE Conference on Decision and Control, pp. 2823–2828.

5 Sinha, A., Kirubarajan, T., and Bar-Shalom, Y. (2005) Autonomous ground target tracking by multiple cooperative UAVs, in IEEE Aerospace Conference, pp. 1–9.

6 Fiorelli, E., Leonard, N.E., Bhatta, P., Paley, D.A., Bachmayer, R., and Fratantoni, D.M. (2004) Multi-AUV control and adaptive sampling in Monterey Bay, in IEEE Autonomous Underwater Vehicles 2004: Workshop on Multiple AUV Operations (AUV04), pp. 134–147.

7 Bowling, M. and Veloso, M. (2000) An analysis of stochastic game theory for multiagent reinforcement learning, Tech. Rep. CMU-CS-00-165, Computer Science Department, Carnegie Mellon University.

8 Tuyls, K. and Parsons, S. (2007) What evolutionary game theory tells us about multiagent learning. *Artificial Intelligence*, **171** (7), 406–416.

9 Jennings, J.S., Whelan, G., and Evans, W.F. (1997) Cooperative search and rescue with a team of mobile robots, in Advanced Robotics, 1997. ICAR'97. Proceedings., 8th International Conference on, IEEE, pp. 193–200.

10 Sujit, P.B. and Beard, R. (2007) Distributed sequential auctions for multiple UAV task allocation, in Proceedings of 2007 American Control Conference, pp. 3955–3960.

11 Wang, Y., Hussein, I.I., and Erwin, R.S. (2008) Awareness-based decision making for search and tracking, in American Control Conference (ACC). Invited Paper.

12 Wang, Y. and Hussein, I.I. (2009) Bayesian-based decision making for object search and characterization, in American Control Conference (ACC).

13 Wang, Y., Hussein, I.I., Brown, D.R. III, and Erwin, R.S. (2010) Cost-aware sequential Bayesian tasking and decision-making for search and classification, in American Control Conference (ACC).

14 Mahler, R. (2003) Objective functions for Bayeisan control-theoretic sensor management, I: Multitarget first-moment approximation, in Proceedings of IEEE Aerospace Conference.

15 Tang, Z. and Özgüner, Ü. (2005) Motion planning for multitarget surveillance with mobile sensor agents. *IEEE Transctions on Robotics*, **21** (5), 898–908.

16 Ny, J.L., Dahleh, M., and Feron, E. (2006) Multi-agent task assignment in the bandit framework, in Proceedings of the 45th IEEE Conference on Decision and Control.

17 Ren, W. and Beard, R. (2004) Decentralized scheme for spacecraft formation flying via the virtual structure approach. *Journal of Guidance, Control, and Dynamics*, **27** (1), 73–82.

18 Logenthiran, T. (2012) Multi-agent system for control and management of distributed power systems, Ph.D. thesis, National University of Singapore.

19 Arai, T., Pagello, E., and Parker, L.E. (2002) Editorial: advances in multi-robot systems. *IEEE Transactions on Robotics and Automation*, **18** (5), 655–661.

20 Ferber, J. (1999) *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley Professional.

21 Weiss, G. (ed.) (2000) *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, The MIT Press.

22 Ferber, J. (1999) *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, vol. **1**, Addison-Wesley Reading.

23 Sandhu, J.S., Agogino, A.M., Agogino, A.K. et al. (2004) Wireless sensor networks for commercial lighting control: decision making with multi-agent systems, in AAAI Workshop on Sensor Networks, vol. **10**, pp. 131–140.

24 Ren, L.H., Ding, Y.S., Shen, Y.Z., and Zhang, X.F. (2008) Multi-agent-based bio-network for systems biology: protein-protein interaction network as an example. *Amino Acids*, **35** (3), 565–572.

25 de Weerdt, M.M., Zhang, Y., and Klos, T.B. (2007) Distributed task allocation in social networks, in Proceedings of the 6th International Conference on Autonomous Agents and Multiagent Systems.

26 Pipattanasomporn, M., Feroze, H., and Rahman, S. (2009) Multi-agent systems in a distributed smart grid: design and implementation, in Power Systems Conference and Exposition, 2009. PSCE'09. IEEE/PES, IEEE, pp. 1–8.

27 Arel, I., Liu, C., Urbanik, T., and Kohls, A. (2010) Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transport Systems*, **4** (2), 128–135.

28 Adler, J.L. and Blue, V.J. (2002) A cooperative multi-agent transportation management and route guidance system. *Transportation Research Part C: Emerging Technologies*, **10** (5), 433–454.

29 Yadati, C., Witteveen, C., and Zhang, Y. (2010) Coordinating agents: an analysis of coordination in supply-chain management tasks, in Proceedings of the 2nd International Conference on Agents and Artificial Intelligence (ICAART).

30 Blondel, V.D., Hendrickx, J.M., Olshevsky, A., and Tsitsiklis, J.N. (2005) Convergence in multiagent coordination, consensus, and flocking. in 44th IEEE Conference

on Decision and Control and 2005 European Control Conference, pp. 2996–3000.

31 Ren, W. and Beard, R.W. (2005) Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, **50** (5), 655–661.

32 Olfati-Saber, R. (2006) Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, **51** (3), 401–420.

33 Hong, Y., Hu, J., and Gao, L. (2006) Tracking control for multi-agent consensus with an active leader and variable topology. *Automatica*, **42** (7), 1177–1182.

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

35 Egerstedt, M. and Hu, X. (2001) Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, **17** (6), 947–951.

36 Leonard, N.E. and Fiorelli, E. (2001) Virtual leaders, artificial potentials and coordinated control of groups, in Proceedings of the 40th IEEE Conference on Decision and Control, pp. 2968–2973.

37 Beard, R.W., Lawton, J., and Hadaegh, F.Y. (2001) A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, **9** (6), 777–790.

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

39 Lafferriere, G., Williams, A., Caughman, J., and Veerman, J.J.P. (2005) Decentralized control of vehicle formations. *System & Control Letters*, **54**, 899–910.

40 Ren, W. (2006) Consensus based formation control strategies for multi-vehicle systems, in Proceedings of the 2006 American Control Conference, pp. 4237–4242.

41 Porfiri, M., Roberson, D.G., and Stilwell, D.J. (2007) Tracking and formation control of multiple autonomous agents: a two-level consensus approach. *Automatica*, **43** (8), 1318–1328.

42 Mastellone, S., Stipanović, D.M., Graunke, C.R., Intlekofer, K.A., and Spong, M.W. (2008) Formation control and collision avoidance for multi-agent non-holonomic systems: theory and experiments. *International Journal of Robotics Research,* **27** (1), 107–126.

43 Xiao, F., Wang, L., Chen, J., and Gao, Y. (2009) Finite-time formation control for multi-agent systems. *Automatica,* **45** (11), 2605–2611.

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

45 Li, W. and Cassandras, C.G. (2005) Distributed cooperative coverage control of sensor networks, in 44th IEEE Conference on Decision and Control and 2005 European Control Conference, pp. 2542–2547.

46 Hussein, I.I. and Stipanović, D.M. (2007) Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *IEEE Transactions on Control Systems Technology,* **15** (4), 642–657.

47 Schwager, M., Rus, D., and Slotine, J.J. (2009) Decentralized, adaptive coverage control for networked robots. *International Journal of Robotics Research,* **28** (3), 357–375.

48 Wang, Y. and Hussein, I.I. (2010) Awareness coverage control over large-scale domains with intermittent communications. *IEEE Transactions on Automatic Control,* **55** (8), 1850–1859.

49 Martínez, S. and Bullo, F. (2006) Optimal sensor placement and motion coordination for target tracking. *Automatica,* **42** (4), 661–668.

50 Hu, J. and Feng, G. (2010) Distributed tracking control of leader-follower multi-agent systems under noisy measurement. *Automatica,* **46** (8), 1382–1387.

51 Ota, J. (2006) Multi-agent robot systems as distributed autonomous systems. *Advanced Engineering Informatics,* **20** (1), 59–70.

52 Parker, L.E. (2008) Distributed intelligence: overview of the field and its application in multi-robot systems. *Journal of Physical Agents,* **2** (1), 5–14.

53 Vincent, R., Fox, D., Ko, J., Konolige, K., Limketkai, B., Morisset, B., Ortiz, C., Schulz, D., and Stewart, B. (2008)

Distributed multirobot exploration, mapping, and task allocation. *Annals of Mathematics and Artificial Intelligence*, **52** (2-4), 229–255.

54 Wang, X., Hong, Y., Huang, J., and Jiang, Z.P. (2010) A distributed control approach to a robust output regulation problem for multi-agent linear systems. *IEEE Transactions on Automatic Control*, **55** (12), 2891–2895.

55 Su, Y. and Huang, J. (2012) Cooperative output regulation of linear multi-agent systems. *IEEE Transactions on Automatic Control*, **57** (4), 1062–1066.

56 Nedić, A. and Ozdaglar, A. (2009) Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, **54** (1), 48–61.

57 Dimarogonas, D.V. and Johansson, K.H. (2009) Event-triggered control for multi-agent systems, in Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on, IEEE, pp. 7131–7136.

58 De Gennaro, M.C. and Jadbabaie, A. (2006) Decentralized control of connectivity for multi-agent systems, in Proceedings of the 45th IEEE Conference on Decision and Control, IEEE, pp. 3628–3633.

59 Su, H., Wang, X., and Chen, G. (2010) Rendezvous of multiple mobile agents with preserved network connectivity. *System & Control Letters*, **59** (5), 313–322.

60 Dimarogonas, D.V. and Kyriakopoulos, K.J. (2008) Connectedness preserving distributed swarm aggregation for multiple kinematic robots. *IEEE Transactions on Robotics*, **24** (5), 1213–1223.

61 Zavlanos, M.M., Tanner, H.G., Jadbabaie, A., and Pappas, G.J. (2009) Hybrid control for connectivity preserving flocking. *IEEE Transactions on Automatic Control*, **54** (12), 2869–2875.

62 Su, H., Wang, X., and Chen, G. (2009) A connectivity-preserving flocking algorithm for multi-agent systems based only on position measurements. *International Journal of Control*, **82** (7), 1334–1343.

63 Xiao, F., Wang, L., and Chen, T. (2012) Connectivity preservation for multi-agent rendezvous with link failure. *Automatica*, **48** (1), 25–35.

64 Ji, M. and Egerstedt, M.B. (2007) Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, **23** (4), 693–703.

65 Wen, G., Duan, Z., Su, H., Chen, G., and Yu, W. (2012) A connectivity-preserving flocking algorithm for multi-agent dynamical systems with bounded potential function. *IET Control Theory and Applications*, **6** (6), 813–821.

66 Crespi, V., Galstyan, A., and Lerman, K. (2008) Top-down vs bottom-up methodologies in multi-agent system design. *Autonomous Robots*, **24** (3), 303–313.

67 Cao, Y., Yu, W., Ren, W., and Chen, G. (2013) An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, **9** (1), 427–438.

68 Bullo, F., Cortés, J., and Martinez, S. (2009) *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*, Princeton University Press.

69 Russell, S.J., Norvig, P., Canny, J.F., Malik, J.M., and Edwards, D.D. (1995) *Artificial Intelligence: A Modern Approach*, vol. **74**, Prentice Hall, Englewood Cliffs, NJ.

70 Jones, J.L. and Flynn, A.M. (1993) *Mobile Robots: Inspiration to Implementation*, AK Peters, Ltd.

71 Ren, W., Beard, R.W., and Atkins, E.M. (2005) A survey of consensus problems in multi-agent coordination, in American Control Conference, 2005. Proceedings of the 2005, IEEE, pp. 1859–1864.

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

73 Yu, W., Chen, G., and Cao, M. (2010) Some necessary and sufficient conditions for second-order consensus in multi-agent dynamical systems. *Automatica*, **46** (6), 1089–1095.

74 Su, H., Chen, G., Wang, X., and Lin, Z. (2011) Adaptive second-order consensus of networked mobile agents with nonlinear dynamics. *Automatica*, **47** (2), 368–375.

75 Su, H., Chen, M.Z., Lam, J., and Lin, Z. (2013) Semi-global leader-following consensus of linear multi-agent systems with input saturation via low gain feedback. *IEEE*

Transactions on Circuits and Systems I: Regular Papers, **60** (7), 1881–1889.

76 Blondel, V.D., Hendrickx, J.M., and Tsitsiklis, J.N. (2009) On Krause's multi-agent consensus model with state-dependent connectivity. *IEEE Transactions on Automatic Control*, **54** (11), 2586–2597.

77 Papachristodoulou, A., Jadbabaie, A., and Munz, U. (2010) Effects of delay in multi-agent consensus and oscillator synchronization. *IEEE Transactions on Automatic Control*, **55** (6), 1471–1477.

78 Oh, K.K., Park, M.C., and Ahn, H.S. (2015) A survey of multi-agent formation control. *Automatica*, **53**, 424–440.

79 Egerstedt, M.B. and Hu, X. (2001) Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, **17** (6), 947–951.

80 De Gennaro, M.C. and Jadbabaie, A. (2006) Formation control for a cooperative multi-agent system using decentralized navigation functions, in 2006 American Control Conference, IEEE, p. 6.

81 Hayakawa, T., Matsuzawa, T., and Hara, S. (2006) Formation control of multi-agent systems with sampled information, in *Proceedings of the IEEE Conference on Decision and Control*, CiteSeer, pp. 4333–4338.

82 Lin, P. and Jia, Y. (2010) Distributed rotating formation control of multi-agent systems. *System & Control Letters*, **59** (10), 587–595.

83 Sun, D., Wang, C., Shang, W., and Feng, G. (2009) A synchronization approach to trajectory tracking of multiple mobile robots while maintaining time-varying formations. *IEEE Transactions on Robotics*, **25** (5), 1074–1086.

84 Zhang, H., Lewis, F.L., and Das, A. (2011) Optimal design for synchronization of cooperative systems: state feedback, observer and output feedback. *IEEE Transactions on Automatic Control*, **56** (8), 1948–1952.

85 Kim, H., Shim, H., and Seo, J.H. (2011) Output consensus of heterogeneous uncertain linear multi-agent systems. *IEEE Transactions on Automatic Control*, **56** (1), 200–206.

86 Chopra, N. and Spong, M.W. (2006) Output synchronization of nonlinear systems with time delay in communication, in

Proceedings of the 45th IEEE Conference on Decision and Control, IEEE, pp. 4986–4992.

87 Su, Y., Hong, Y., and Huang, J. (2013) A general result on the robust cooperative output regulation for linear uncertain multi-agent systems. *IEEE Transactions on Automatic Control*, **58** (5), 1275–1279.

88 Du, H., He, Y., and Cheng, Y. (2014) Finite-time synchronization of a class of second-order nonlinear multi-agent systems using output feedback control. *IEEE Transactions on Circuits and Systems I: Regular Papers*, **61** (6), 1778–1788.

89 Li, Z., Liu, X., Ren, W., and Xie, L. (2013) Distributed tracking control for linear multiagent systems with a leader of bounded unknown input. *IEEE Transactions on Automatic Control*, **58** (2), 518–523.

90 Hong, Y. and Wang, X. (2009) Multi-agent tracking of a high-dimensional active leader with switching topology. *Journal of Systems Science and Complexity*, **22** (4), 722–731.

91 Harmati, I. and Skrzypczyk, K. (2009) Robot team coordination for target tracking using fuzzy logic controller in game theoretic framework. *Robotics and Autonomous Systems*, **57** (1), 75–86.

92 Zhao, Y., Duan, Z., Wen, G., and Zhang, Y. (2013) Distributed finite-time tracking control for multi-agent systems: an observer-based approach. *System & Control Letters*, **62** (1), 22–28.

93 MartíNez, S. and Bullo, F. (2006) Optimal sensor placement and motion coordination for target tracking. *Automatica*, **42** (4), 661–668.

94 Chung, T.H., Burdick, J.W., and Murray, R.M. (2006) A decentralized motion coordination strategy for dynamic target tracking, in Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006, IEEE, pp. 2416–2422.

95 Yan, J., Guan, X.P., and Tan, F.X. (2010) Target tracking and obstacle avoidance for multi-agent systems. *International Journal of Automation and Computing*, **7** (4), 550–556.

96 Lewis, F.L., Zhang, H., Hengster-Movric, K., and Das, A. (2013) *Cooperative Control of Multi-Agent Systems:*

Optimal and Adaptive Design Approaches, Springer Science & Business Media.

97 Movric, K.H. and Lewis, F.L. (2014) Cooperative optimal control for multi-agent systems on directed graph topologies. *IEEE Transactions on Automatic Control*, **59** (3), 769–774.

98 Wiegerinck, W., van den Broek, B., and Kappen, H. (2012) Stochastic optimal control in continuous space-time multi-agent systems, *arXiv preprint arXiv:1206.6866*.

99 Cassandras, C.G., Lin, X., and Ding, X. (2013) An optimal control approach to the multi-agent persistent monitoring problem. *IEEE Transactions on Automatic Control*, **58** (4), 947–961.

100 Cortes, J., Martinez, S., Karatas, T., and Bullo, F. (2002) Coverage control for mobile sensing networks, in Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on, vol. **2**, IEEE, pp. 1327–1332.

101 Gao, C., Cortés, J., and Bullo, F. (2008) Notes on averaging over acyclic digraphs and discrete coverage control. *Automatica*, **44** (8), 2120–2127.

102 Kwok, A. and Martìnez, S. (2010) Unicycle coverage control via hybrid modeling. *IEEE Transactions on Automatic Control*, **55** (2), 528–532.

103 Wang, Y. and Hussein, I.I. (2012) *Search and Classification Using Multiple Autonomous Vehicles: Decision-Making and Sensor Management*, vol. **427**, Springer Science & Business Media.

104 Li, W. and Cassandras, C.G. (2005) Distributed cooperative coverage control of sensor networks, in Proceedings of the 44th IEEE Conference on Decision and Control, IEEE, pp. 2542–2547.

105 Zhong, M. and Cassandras, C.G. (2011) Distributed coverage control and data collection with mobile sensor networks. *IEEE Transactions on Automatic Control*, **56** (10), 2445–2455.

106 Hussein, I.I. and Stipanovic, D.M. (2006) Effective coverage control using dynamic sensor networks, in Proceedings of the 45th IEEE Conference on Decision and Control, IEEE, pp. 2747–2752.

107 Song, C., Feng, G., Fan, Y., and Wang, Y. (2011) Decentralized adaptive awareness coverage control for multi-agent networks. *Automatica*, **47** (12), 2749–2756.

108 Song, C., Liu, L., Feng, G., Wang, Y., and Gao, Q. (2013) Persistent awareness coverage control for mobile sensor networks. *Automatica*, **49** (6), 1867–1873.

109 Chopra, N. and Spong, M.W. (2006) Passivity-based control of multi-agent systems, in S. Kawamura and M. Svinin (eds), *Advances in Robot Control*, Springer-Verlag, pp. 107–134.

110 Van der Schaft, A. (2012) *L2-Gain and Passivity Techniques in Nonlinear Control*, Springer Science & Business Media.

111 Ortega, R., Perez, J.A.L., Nicklasson, P.J., and Sira-Ramirez, H. (2013) *Passivity-Based Control of Euler-Lagrange Systems: Mechanical, Electrical and Electromechanical Applications*, Springer Science & Business Media.

112 Hirche, S. and Hara, S. (2008) Stabilizing interconnection characterization for multi-agent systems with dissipative properties. *IFAC Proceedings Volumes*, **41** (2), 1571–1577.

113 Listmann, K.D., Woolsey, C.A., and Adamy, J. (2009) Passivity-based coordination of multi-agent systems: a backstepping approach, in Control Conference (ECC), 2009 European, IEEE, pp. 2450–2455.

114 LeBlanc, H., Eyisi, E., Kottenstette, N., Koutsoukos, X.D., and Sztipanovits, J. (2010) A passivity-based approach to deployment in multi-agent networks, in ICINCO (1), pp. 53–62.

115 Igarashi, Y., Hatanaka, T., Fujita, M., and Spong, M.W. (2009) Passivity-based attitude synchronization in *SE*(3). *IEEE Transactions on Control Systems Technology*, **17** (5), 1119–1134.

116 Zhu, Y., Qi, H., and Cheng, D. (2009) Synchronisation of a class of networked passive systems with switching topology. *International Journal of Control*, **82** (7), 1326–1333.

117 Wang, Y., Gupta, V., and Antsaklis, P.J. (2013) Stochastic passivity of discrete-time markovian jump nonlinear systems, in 2013 American Control Conference, IEEE, pp. 4879–4884.

118 Rodríguez-Seda, E.J., Troy, J.J., Erignac, C.A., Murray, P., Stipanovic, D.M., and Spong, M.W. (2010) Bilateral

teleoperation of multiple mobile agents: coordinated motion and collision avoidance. *IEEE Transactions on Control Systems Technology*, **18** (4), 984–992.

119 Rodríguez-Seda, E.J., Stipanović, D.M., and Spong, M.W. (2012) Teleoperation of multi-agent systems with nonuniform control input delays. *Integrated Computer-Aided Engineering*, **19** (2), 125–136.

120 Giordano, P.R., Franchi, A., Secchi, C., and Bülthoff, H.H. (2011) Bilateral teleoperation of groups of uavs with decentralized connectivity maintenance, in *Robotics: Science and Systems*, CiteSeer.

121 Cao, Y. and Ren, W. (2010) Multi-vehicle coordination for double integrator dynamics under fixed undirected/directed interaction in a sampled-data setting. *International Journal of Robust and Nonlinear Control*, **20**, 987–1000.

122 Hayakawa, T., Matsuzawa, T., and Hara, S. (2006) Formation control of multi-agent systems with sampled information, in 45th IEEE Conference on Decision and Control, pp. 4333–4338.

123 Liu, H., Xie, G., and Wang, L. (2010) Necessary and sufficient conditions for solving consensus of double integrator dynamics via sampled control. *International Journal of Robust and Nonlinear Control*, **20** (15), 1706–1722.

124 Qin, J. and Gao, H. (2012) A sufficient condition for convergence of sampled-data consensus for double integrator dynamics with nonuniform and time-varying communication delays. *IEEE Transactions on Automatic Control*, **57** (9), 2417–2422.

125 Astrom, K.J. (2008) Event based control, in A. Astolfi and L. Marconi (eds), *Analysis and Design of Nonlinear Control Systems*, Springer-Verlag, Berlin, pp. 127–147.

126 Tabuada, P. (2007) Event-triggered real-time scheduling of stabilizing control tasks. *IEEE Transactions on Automatic Control*, **52** (9), 1680–1685.

127 Garcia, E. and Antsaklis, P.J. (2014) Optimal model-based control with limited communication, in 19th IFAC World Congress, pp. 10 908–10 913.

128 Wang, X. and Lemmon, M. (2011) Event-triggering in distributed networked control systems. *IEEE Transactions on Automatic Control*, **56** (3), 586–601.

129 Persis, C.D., Sailer, R., and Wirth, F. (2013) Parsimonious event-triggered distributed control: a zeno free approach. *Automatica*, **49** (7), 2116–2124.

130 Garcia, E. and Antsaklis, P.J. (2012) Output feedback model-based control of uncertain discrete-time systems with network induced delays, in 51st IEEE Conference on Decision and Control, pp. 6647–6652.

131 Li, Z., Duan, Z., Chen, G., and Huang, L. (2010) Consensus of multiagent systems and synchronization of complex networks: a unified viewpoint. *IEEE Transactions on Circuits and Systems I: Regular Papers*, **57** (1), 213–224.

132 Ma, C.Q. and Zhang, J.F. (2010) Necessary and sufficient conditions for consensusability of linear multi-agent systems. *IEEE Transactions on Automatic Control*, **55** (5), 1263–1268.

133 Liu, T., Hill, D.J., and Liu, B. (2012) Synchronization of dynamical networks with distributed event-based communication, in 51st IEEE Conference on Decision and Control, pp. 7199–7204.

134 Zhu, W., Jiang, Z.P., and Feng, G. (2014) Event-based consensus of multi-agent systems with general linear models. *Automatica*, **50** (2), 552–558.

135 Garcia, E., Cao, Y., and Casbeer, D.W. (2014) Cooperative control with general linear dynamics and limited communication: centralized and decentralized event-triggered control strategies, in American Control Conference, pp. 159–164.

136 Demir, O. and Lunze, J. (2012) Event-based synchronisation of multi-agent systems, in IFAC Conference on Analysis and Design of Hybrid Systems, pp. 1–6.

137 Garcia, E., Cao, Y., and Casbeer, D.W. (2014) Decentralized event-triggered consensus with general linear dynamics. *Automatica*, **50** (10), 2633–2640.

138 Garcia, E., Cao, Y., and Casbeer, D.W. (2014) Event-triggered cooperative control with general linear dynamics and communication delays, in IEEE Conference on Decision and Control, pp. 2914–2919.

139 Dimarogonas, D.V., Frazzoli, E., and Johansson, K.H. (2012) Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, **57** (5), 1291–1297.

140 Seyboth, G.S., Dimarogonas, D.V., and Johansson, K.H. (2013) Event-based broadcasting for multi-agent average consensus. *Automatica*, **49** (1), 245–252.

141 Garcia, E., Cao, Y., Yu, H., Antsaklis, P.J., and Casbeer, D.W. (2013) Decentralized event-triggered cooperative control with limited communication. *International Journal of Control*, **86** (9), 1479–1488.

142 Yu, H. and Antsaklis, P.J. (2012) Quantized output synchronization of networked passive systems with event-driven communication, in American Control Conference, pp. 5706–5711.

143 Yin, X. and Yue, D. (2013) Event-triggered tracking control for heterogeneous multi-agent systems with Markov comunication delays. *Journal of the Franklin Institute*, **350** (5), 1312–1334.

144 Chen, X. and Hao, F. (2012) Event-triggered average consensus control for discrete-time multi-agent systems. *IET Control Theory and Applications*, **6** (16), 2493–2498.

145 Guo, M. and Dimarogonas, D.V. (2013) Nonlinear consensus via continuous, sampled, and aperiodic updates. *International Journal of Control*, **86** (4), 567–578.

146 Garcia, E., Cao, Y., and Casbeer, D.W. (2016) Decentralized event-triggered consensus of double integrator multi-agent systems with packet losses and communication delays. *IET Control Theory and Applications*, **10** (15), 1835–1843.

147 Lemmon, M. (2010) Event-triggered feedback in control, estimation, and optimization, in A. Bemporad, M. Heemels, and M. Johansson (eds), *Networked Control Systems*, Lecture Notes in Control and Information Sciences, vol. **405**, Springer, pp. 293–358.

2

Sensor Placement Algorithms for a Path Covering Problem

Sivakumar Rathinam[1] *and Rajnikant Sharma*[2]

[1] *Department of Mechanical Engineering, Texas A& M University, College Station,, TX, USA*
[2] *Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA*

Localization is a critical technology required for navigation of autonomous unmanned vehicles (UVs) in applications of national importance such as disaster management, border surveillance, and monitoring of civilian infrastructure including oil pipelines, power grids, harbors, inland waterways, and intelligent transportation systems where GPS signals can be easily jammed either intentionally or unintentionally. The oil pipeline network in the United States roughly consists of 1.9 million miles of pipeline, of which 155,000 miles of pipeline transport hazardous liquids [1]. Monitoring and maintenance of these large networks plays an important role in protecting their structural integrity and functional reliability and is an issue of national importance. A system of aerial UVs and ground sensors are seen as ideal platforms for performing repeated inspections of pipelines and can significantly aid in the detection of oil leakages, discoloring of vegetation, earth movement, and erosion around or near the pipelines [1, 2]. Aerial UVs have already been used in maritime applications for monitoring coastal regions for environmental protection and intelligence gathering [3]. Recent events such as the unintentional jamming by technicians in San Diego harbor [4], which disrupted the GPS signals in the entire San Diego area, show that interferences, design errors, and flawed management practices [4–6] can

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

cause GPS service outages, which in turn can crucially affect all the systems that depend on GPS information. This chapter addresses a fundamental question that arises while deploying a team of autonomous UVs for monitoring applications in GPS-denied environments.

The position of a vehicle is estimated by combining its motion information (accelerations, linear and angular velocity) from intrinsic sensors (such as inertial measurement unit (IMU) and encoder) and information (range and bearing angle from vehicle external environment) from extrinsic sensors (camera, laser range finder, GPS receiver). Among existing localization techniques, simultaneous localization and mapping (SLAM) methods are considered to be the holy grail for navigation of autonomous robots in unknown environments [7, 8]. In GPS-denied environments, relative position measurements using range or bearing angle sensors to known landmarks can be used to localize the vehicle. An extended Kalman filter (EKF) or its information from extended information filter (EIF) can be used to estimate the vehicle's states. The estimation algorithm will provide meaningful localization estimates (consistent and bounded) if and only if the sensors provide enough information for localization, or, in other words, if the system is observable. It has been shown that the bound on uncertainty is related with the eigenvalues of the observability gramian [9]. In our previous work [10] where the vehicles carry bearing sensors, we have shown that each vehicle needs at least measurements from two different landmarks in order for the system to be observable at any time.

In applications where known landmarks are not available or standard estimation algorithms cannot be used, additional localization vehicles or proxy landmarks can be suitably placed to provide the required information (Figure 2.1). For example, given a map of the test area, one way to track the position of an aerial vehicle in the test area is to use vision sensors and apply suitable image processing/filtering techniques across a sequence of sensed images. These techniques are computationally intensive for a small UV with resource constraints and can significantly slow down the speed of the vehicle. An alternative would be to track the position of the aerial vehicle by analyzing the radio signals received from suitably positioned neighboring

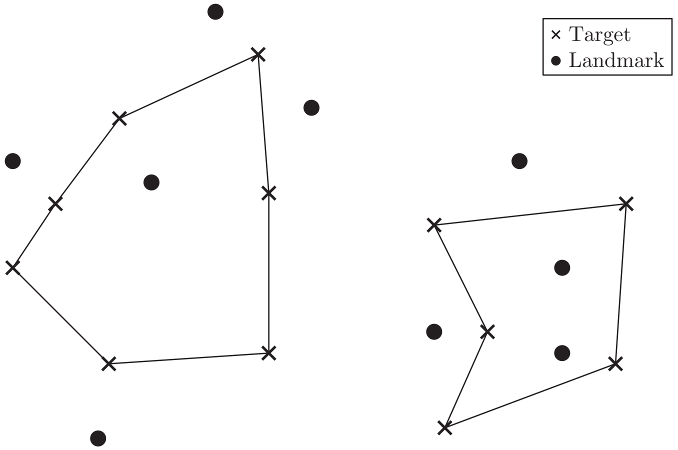


Figure 2.1 Scenario where the landmarks must be suitably placed to facilitate accurate tracking of the vehicle's paths visiting the targets.

vehicles, which act as proxy landmarks. A network of proxy landmarks (or simply landmarks as referred in this chapter) can be carefully placed to cover the entire path of the aerial vehicle thus making use of the localization techniques [11–13]. Suitably placing landmarks applies to indoor robots also; for example, landmarks with easily recognizable features can be placed in an indoor environment where the robot simply uses these landmarks to travel the given path.

This chapter addresses the fundamental problem of suitably placing a minimum number of landmarks such that the position of a vehicle traveling a given path can be estimated as precisely as possible relative to the landmarks. The paths for the vehicles are given in the form of way points. Based on our prior work [10], it is assumed that the location of a vehicle can be estimated accurately if there are at least two landmarks within a distance (say R) from the vehicle's location. The maximum bound R between the landmarks and a vehicle's path will ensure that the vehicle's position error is in an acceptable range and can be computed using the uncertainty analysis in [9]. The landmarks are also placed so that any two landmarks are at least separated by p units; this distance constraint is to ensure that the landmarks are separated sufficiently for collision avoidance and safety. This problem is referred to as the landmark placement problem (LPP).

The LPP belongs to a class of geometric, multiset cover problems [14–16], which has received significant attention in the optimization community over the last five decades. LPP is a generalization of the classic unit disk cover problem [14] and is NP-hard. Therefore, we are interested in developing approximation algorithms for this problem. An α- approximation algorithm is an algorithm that runs in polynomial time and provides a solution whose cost is at most α times from the optimum for any instance of the problem. The factor α is commonly referred to as the approximation ratio or approximation factor of the algorithm. We are currently not aware of any approximation algorithms for the LPP. However, for the case when each target must be covered by just one landmark, there are approximation algorithms in [14, 17]. In this chapter, we develop two approximation algorithms with an approximation ratio of 13 and 5, respectively, to solve the LPP. This chapter builds on our preliminary work on this problem in [18]. In addition to proving the theoretical guarantees, numerical simulations are performed on thousands of problem instances to corroborate the performance of the proposed algorithms. These simulations show that the proposed algorithms produce solutions with bounds that are significantly better than the guarantees indicated by the approximation factor.

2.1 Problem Statement

A target waypoint in a vehicle's path (also referred to as target) is defined to be covered by a landmark if the distance between the target and the landmark is at most equal to R. We assume there are m targets. Each target i is located on a ground plane with its position coordinates denoted as (x_i, y_i). The locations of the targets are distinct; no two targets correspond to the same position on the plane. The objective of the LPP is to place the landmarks such that each target is covered using at least two distinct landmarks, the distance between any two landmarks is at least equal to p units, and the number of landmarks is minimized. We assume that $p \leq \frac{R}{2}$ (this is a reasonable assumption

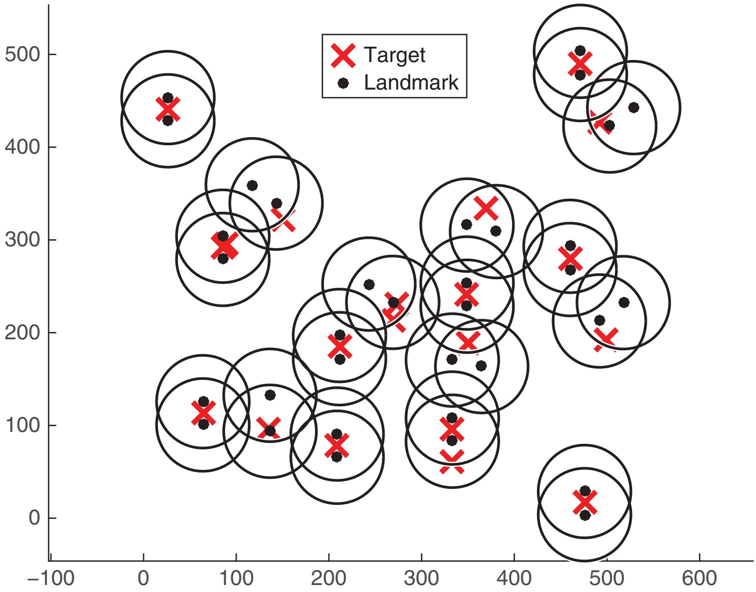


Figure 2.2 A feasible solution to the landmark placement problem. For this example, *R* was chosen to be 50 units and *p* was set to 5 units. A landmark is present at the center of each circle. The circle shows the boundary of the area covered by its respective landmark. One can verify that each target is covered by at least two well-spaced landmarks in this example.

for all practical scenarios we consider). Refer to Figure 2.2 for an illustration of a feasible solution to this problem.

2.2 Algorithm *Approx*$_1$

We first present a greedy algorithm for a simpler LPP where all the targets lie within a vertical strip of width equal to $\sqrt{3}R$ and the landmarks are restricted to be on a single, vertical line. We will use this result to later present an approximation algorithm for the general LPP. The feasible solution we generate both for the simpler and the general LPP is referred to as a line-restricted solution because we place all the landmarks only on vertical lines. This simplification enables us to do a geometric analysis and prove the approximation ratio.

2.2.1 Algorithm for Targets That Lie Within a Strip

Let all the targets lie within a vertical strip such that the x-coordinate of each target lies in the interval $[-\frac{\sqrt{3}}{2}R, +\frac{\sqrt{3}}{2}R]$. Each landmark must be placed on the y-axis such that each target is covered by at least two landmarks. Consider the following greedy algorithm for this problem:

- For any target i, let $(0, \alpha_i)$ and $(0, \beta_i)$ (with $\alpha_i > \beta_i$) be the two points where the circle of radius R centered at target i intersects the y-axis. Note that the line segment joining these two intersection points will be of length greater than R but smaller than $2R$. Without loss of generality, sort all the targets such that $\beta_i \geq \beta_j$ if $i < j$.
- For $i = 1, 2, \ldots, m$, do the following:
  1) If target i is not covered by any landmark, then add two landmarks at locations $(0, \beta_i + p)$ and $(0, \beta_i)$.
  2) If target i is covered by only one landmark, then add another landmark at location $(0, \beta_i)$.

Lemma 2.2.1 The greedy algorithm for the special case of the LPP finds an optimal solution in the order of $m_s \log m_s$ steps where m_s is the number of targets in the vertical strip.

Proof: Let $m_s \geq 2$ (the case of $m_s = 1$ is trivial). The sorting step in the greedy algorithm consumes the majority of the computation time and requires m_s log m_s steps to implement. Now, consider the steps inside the *for – loop* of the greedy algorithm. A landmark is added in the line segment joining $(0, \alpha_i)$ and $(0, \beta_i)$ for target i only if target i has not been covered by at least two landmarks at the end of the $i - 1th$ iteration. Let $U = \{i_1, i_2, \ldots, i_k\}$ be the targets for which at least one landmark was added during the greedy algorithm. Here, the notation is such that i_1 is the first uncovered target considered, i_2 is the second uncovered target considered, and so on. Without loss of generality, we can assume that the union of all the line segments corresponding to the targets forms a connected set; if this is not the case, then the proof here can be applied to all the line segments corresponding to each of the connected sets separately.

Note that the greedy algorithm produces a solution that covers all the targets in the vertical strip using $|U| + 1$ landmarks. The lemma will follow if we can show that the optimal number of landmarks needed to cover each and every target in U is equal to $|U| + 1$. We show this claim in the following discussion: it is easy to verify that the line segments corresponding to the selected targets in U (from the greedy algorithm) satisfy the following properties:

- For any two distinct targets $a, b \in U$, the line segment [1] corresponding to a is not a subset of the line segment corresponding to b and vice versa.
- For any target i_l, $\beta_{i_l} > \alpha_{i_q}$ for any $q \geq l + 2$. This essentially states the fact that the line segment corresponding to target i_l will not intersect with the line segment of any target i_q for $q \geq l + 2$.
- If there are at least two targets in U, then $\beta_{i_1} + p > \alpha_{i_2}$.

The first two properties imply that a landmark can only cover at most two targets exactly once. The third property implies that three landmarks are needed to cover targets i_1 and i_2. Therefore, one needs at least $|U| + 1$ landmarks to cover all the targets in U. Hence proved. □

2.2.2 Algorithm for a General Set of Points

The algorithm *Approx* for solving a general LPP is as follows:

1) Partition the plane into vertical strips of width $\sqrt(3)\,R$. Let S represent the set of all the vertical lines running down the center of each of these strips.
2) For each strip, apply the greedy algorithm to place landmarks to cover all the targets in the strip.

As there are a finite number of strips and the greedy algorithm runs in polynomial time for each of the strips, *Approx* runs in polynomial time. In the following section, we prove that the approximation factor of *Approx* is 13. We prove this result by first showing that there is always a feasible line-restricted solution that covers any disk [2] C in an optimal solution using at most

[1] A line segment can be viewed as a convex hull of its two end points.

[2] A disk here refers to the set of all the points that are at most R units away from a landmark in an optimal solution.

13 landmarks. We construct this feasible solution independently for each strip by covering each of the points in the strip using at least two distinct landmarks. Since the greedy algorithm in *Approx* finds an optimal solution for each strip independently, the number of landmarks used by *Approx* is at most equal to the number of landmarks used by the feasible solution. It then follows that the approximation ratio of *Approx* is 13. We first state a result in [14], which reduces covering a set of points in a strip into covering a pair of line segments on the boundary of the strip. We then use this result in the lemmas that follow. We construct a feasible solution for two distinct cases of the position of any landmark in an optimal solution.

Proposition 2.2.2 (Liu and Lu [14]): consider a vertical strip of width $\sqrt{3}R$ and the line L running down the center of the vertical strip. Let C denote the disk of radius R whose center is inside the vertical strip. Let L' and L'' be the two line segments obtained by intersecting C and the boundary of the vertical strip. Let S be a set of landmarks on line L. Each and every common point present in both the disk C and the vertical strip is covered by the landmarks in S if and only if each and every point in line segments L' and L'' is covered by the landmarks in S.

2.2.3 Proof of the Approximation Ratio

Lemma 2.2.3 Let C be the disk corresponding to a landmark chosen in an optimal solution. Let the x-coordinate of the landmark (x_c) be located anywhere in the interval $(\frac{2-\sqrt(3)}{2}R, \frac{3\sqrt(3)-2}{2}R)$. Then, a feasible line-restricted solution can cover any target in C twice using at most 10 landmarks.

Proof: Without loss of generality, let us assume that the landmark at the center of C is located on the x-axis, that is, at $(x_c, 0)$. If $x_c \in (\frac{2-\sqrt(3)}{2}R, \frac{3\sqrt(3)-2}{2}R)$, then C will intersect at most two vertical strips in a line-restricted solution. Let the corresponding center lines of the two strips be denoted by L_1 and L_2 as shown in Figure 2.3. Let S_1 and S_2 be the set of all the common points between the disk C and each of the two strips, respectively.

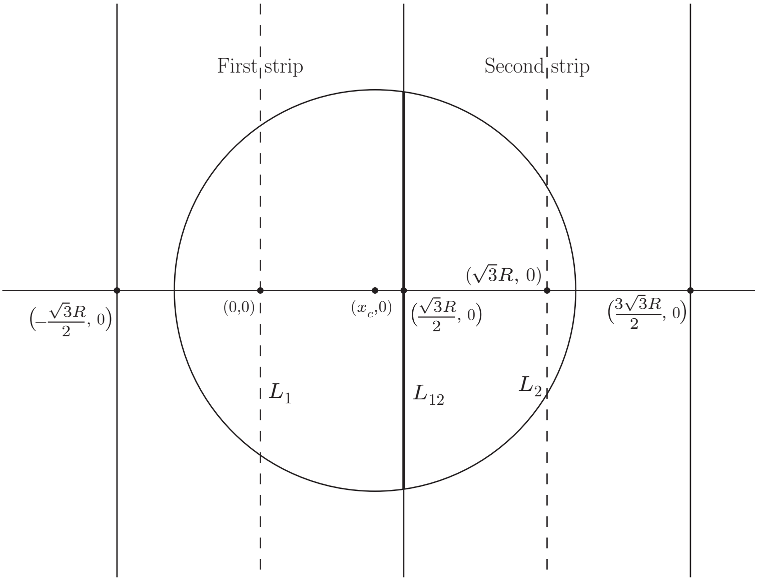

First strip
Second strip
$(\sqrt{3}R,\,0)$
$(-\frac{\sqrt{3}R}{2},\,0)$
(0,0)
$(x_c,0)$
$(\frac{\sqrt{3}R}{2},\,0)$
$(\frac{3\sqrt{3}R}{2},\,0)$
L_1
L_{12}
L_2

Figure 2.3 An illustration for the proof of Lemma 2.2.3.

Also, let L_{12} be the line segment obtained by intersecting C and the boundary between the two strips (refer to the Figure 2.3). The feasible solution must ensure that each of the points in S_1 (S_2) is covered twice by the landmarks placed on L_1 (L_2). From Proposition 2.2.2, the landmarks placed in L_1 can cover each and every point in S_1 at least twice if it can cover any point in L_{12} at least twice. Due to the limits placed on x_c, the length of the line segment L_{12} must be smaller than $2R$ but greater than R [3]. By simple geometry, as shown in Figure 2.4, since $0 < p \leq \frac{R}{2}$, one can verify that five landmarks located at $(0, 0)$, $(0, \frac{R}{2})$, $(0, \frac{R}{2} + p)$, $(0, -\frac{R}{2})$, $(0, -\frac{R}{2} - p)$ will cover every point in L_{12} at least twice. Similarly, five landmarks located on line L_2 at $(\sqrt(3)R, 0)$, $(\sqrt(3)R, \frac{R}{2})$, $(\sqrt(3)R, \frac{R}{2} + p)$, $(\sqrt(3)R, -\frac{R}{2})$, $(\sqrt(3)R, -\frac{R}{2} + p)$ will cover any point in L_{12}, and consequently any point in S_2 at least twice. Hence proved. □

[3] To be precise, the length of the line segment L_{12} will be smaller than $2R$ but greater than $2\sqrt{2\sqrt{3} - 3}R$.

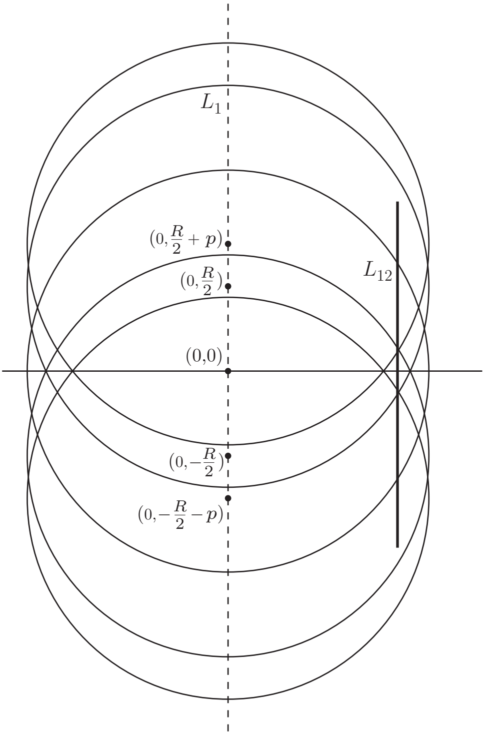


Figure 2.4 Placing five landmarks at the marked locations will cover every point in the segment L_{12} at least twice.

Lemma 2.2.4 Let C be the disk corresponding to a landmark chosen in an optimal solution. Let the x-coordinate of the landmark at the center of C be located anywhere in the interval $(-\frac{2-\sqrt(3)}{2}R, \frac{2-\sqrt(3)}{2}R)$. Then, a feasible, line-restricted solution can cover any target in C completely using at most 13 landmarks.

Proof: Without loss of generality, let us assume that the landmark at the center of C is located on the x-axis, that is, at $(x_c, 0)$ and $x_c \in (0, \frac{2-\sqrt(3)}{2}R)$. If $x_c \in (0, \frac{2-\sqrt(3)}{2}R)$, then C will intersect at most three vertical strips in a line-restricted solution. Let the corresponding center lines of the three strips be denoted by L_1, L_2, and L_3, as shown in Figure 2.5. Let S_1, S_2, and S_3 be the set of all the common points between the disk C and each of the three strips. Also, let L_{12} be the line segment obtained by intersecting C and the boundary between the first two strips, and let L_{23} be

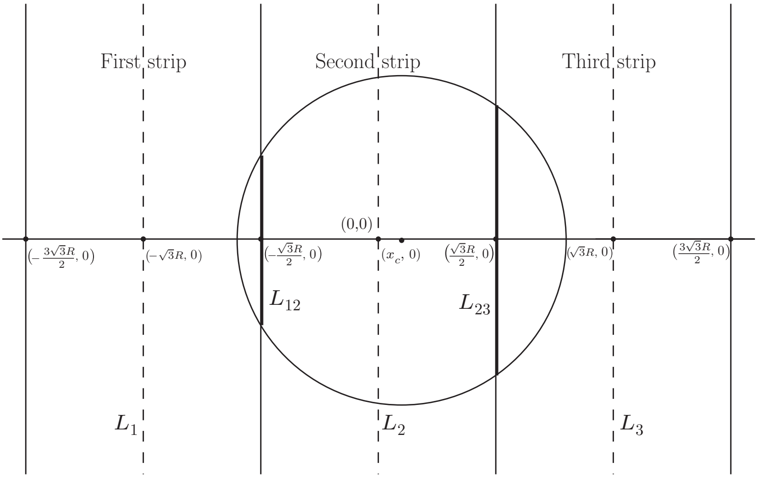


Figure 2.5 An illustration for the proof of Lemma 2.2.4.

the line segment obtained by intersecting C and the boundary between the second and third strip. Due to the limits placed on x_c, the length of the line segment L_{12} must be smaller than R while the length of L_{23} must be smaller than $2R$ but greater than R. By simple geometry, one can verify that three landmarks located at $(-3\frac{\sqrt{3}}{2}R, 0)$, $(-3\frac{\sqrt{3}}{2}R, p)$, $(-3\frac{\sqrt{3}}{2}R, -p)$ will cover every point in L_{12} at least twice. Similarly, five landmarks located in line L_2 at $(0, 0)$, $(0, \frac{R}{2})$, $(0, \frac{R}{2} + p)$, $(0, -\frac{R}{2})$, *and* $(0, -\frac{R}{2} + p)$ will cover any point in L_{12} and L_{23} at least twice. Also, five landmarks located in line L_3 at $(\sqrt{(3)}R, 0)$, $(\sqrt{(3)}R, \frac{R}{2})$, $(\sqrt{(3)}R, \frac{R}{2} + p)$, $(\sqrt{(3)}R, -\frac{R}{2})$, *and* $(\sqrt{(3)}R, -\frac{R}{2} + p)$ will cover any point in L_{23} at least twice. Hence proved. □

Theorem 2.2.5 The approximation ratio of *Approx* is 13.

Proof: The feasible solution is constructed by considering each strip independently and covering each of the points in each strip using at least two distinct landmarks. Since the greedy algorithm in *Approx* finds an optimal solution for each strip independently, the number of landmarks used by *Approx* is at most equal to the number of landmarks used by the feasible solution. In addition, the feasible line-restricted solution can cover the disk of radius

R centered at any landmark in an optimal solution using at most 13 landmarks. It then follows that the approximation ratio of *Approx* is 13. □

Remark 2.2.6 we conjecture that the approximation factor of $Approx_1$ can be further improved to 6.5. Currently, the approximation factor of 13 is proved by showing that it will take at most 13 landmarks in the feasible solution constructed by the algorithm to cover each and every target assigned to a landmark in the optimal solution. However, this proof does not exploit the fact that one needs at least two landmarks in an optimal solution to cover any target, and, therefore, it could only take $\lceil \frac{13}{2} \rceil$ landmarks in any feasible solution to cover each target twice. However, we currently do not have a proof of this conjecture.

The following sections present a second algorithm for the LPP with an approximation factor of 5.

2.3 Algorithm $Approx_2$

Our next algorithm $Approx_2$ is based on a recent 4-approximation algorithm developed for a unit disk problem in [17]. $Approx_2$ works in two phases. In the first phase, a subset of targets is identified such that the distance between any two targets in the subset is greater than $2R$. This subset can be found using a simple greedy algorithm which we will outline later. In the next phase, landmarks are added in the near vicinity of each of the targets in the subset such that all the targets are covered according to the constraints stated in the LPP. The following are the main steps of the proposed algorithm:

Let S denote the set of all the targets. Let L denote all the chosen landmarks. Also, let C represent the subset of S such that the distance between any two targets in C is greater than $2R$. Initially, C and L are empty.

1) **Phase 1:**
   (a) Choose a target $i \in S$ with the least x_i coordinate and the least y_i coordinate.

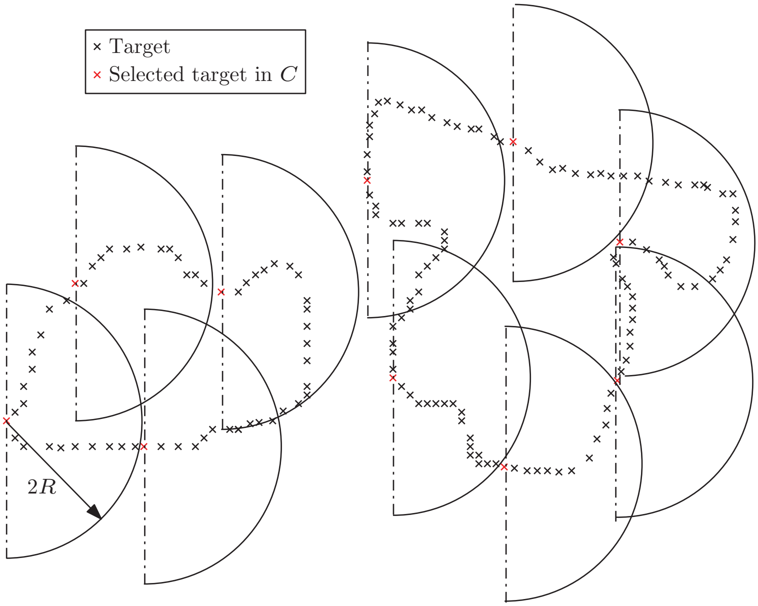


Figure 2.6 Output after the greedy algorithm in phase 1 for a sample instance. The targets selected in the set *C* are also shown. Each of the semicircles is centered at a target in *C* and has a radius equal to 2*R*. Note that every other target not in *C* is present within at least one of the semicircles.

   (b) Let $S_i \subseteq S$ be the set of all the targets such that the distance between any target $x \in S_i$ and i is at most equal to $2R$. Let $C := C \cup \{i\}$ and $S := S \backslash S_i$.
   (c) Go to step (a) until S is empty. Refer to Figure 2.6.

2) **Phase 2:** For each $i \in C$, do the following:
   (a) Add at most 10 landmarks in the region surrounding target i as shown in Figure 2.7. Let these landmarks be denoted as L_i. Then, $L := L \cup L_i$. Note that each target in S_i will be covered by at least two landmarks in L_i.

Lemma 2.3.1 *Approx*$_2$ produces a feasible solution for the LPP for any instance of the problem.

Proof: By construction, phase 1 of *Approx*$_2$ produces a subset of targets C such that the distance between any two

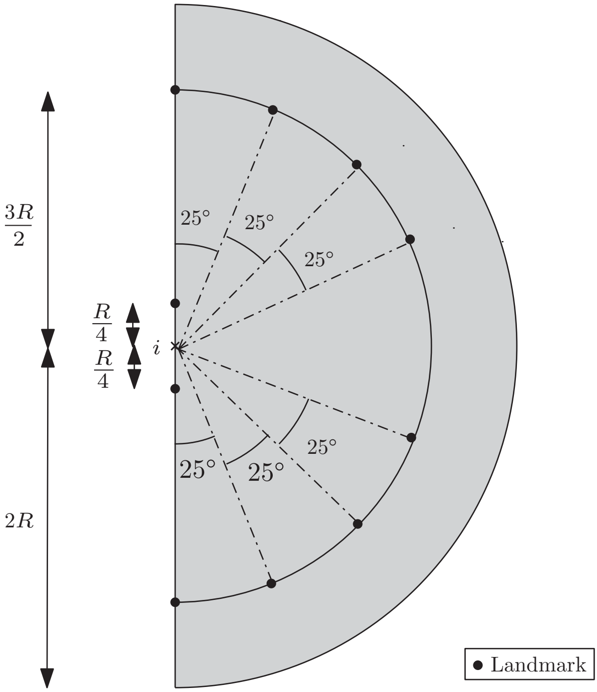


Figure 2.7 The shaded region in the figure around target *i* is denoted by $\mathcal{SC}_i$ and is defined as $\{(x, y) : \sqrt{(x - x_i)^2 + (y - y_i)^2} \leq 2R, x \geq x_i\}$. Ten landmarks are placed to cover each point in this region at least twice. Note that the distance between any two landmarks is at least equal to $p \quad (\leq \frac{R}{2})$ units.

targets in C is greater than $2R$. Suppose $\mathcal{SC}_i := \{(x, y) : \sqrt{(x - x_i)^2 + (y - y_i)^2} \leq 2R, x \geq x_i\}$ for any target $i \in C$. $\mathcal{SC}_i$ is the set of all the points that lie within a semicircle of radius $2R$ with i as its center as shown in Figure 2.7. The landmarks are placed in $\mathcal{SC}_i$ such that the distance between any two of these landmarks is at least equal to $\frac{R}{2} \geq p$. Also, by construction, note that for any distinct targets $i, j \in C$, the distance between any landmark placed in $\mathcal{SC}_i$ is at least $\frac{R}{2}$ units away from any landmark placed in $\mathcal{SC}_j$. Therefore, the placement of the landmarks satisfies the safety constraints outlined in LPP. The 10 landmarks are placed in $\mathcal{SC}_j$ cover each point in $\mathcal{SC}_j$ at least

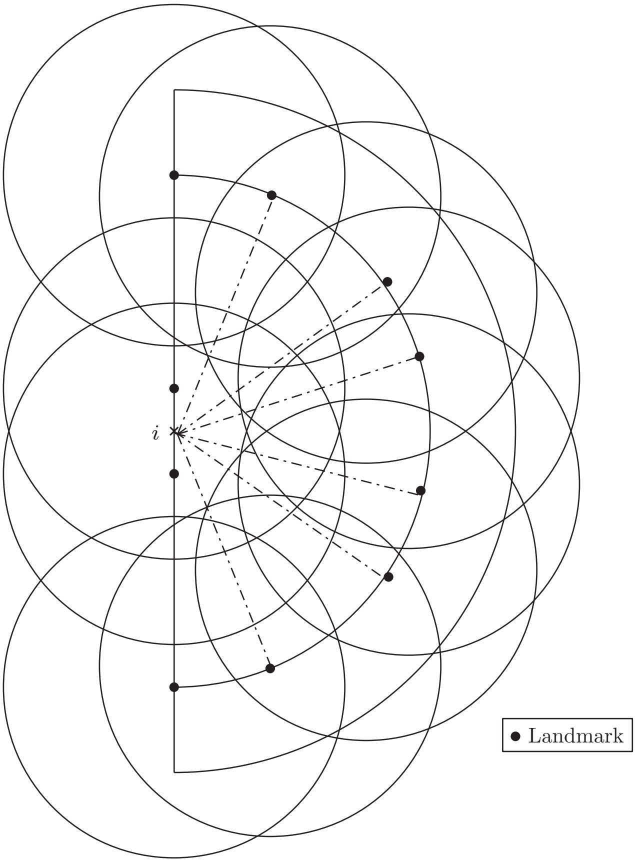


Figure 2.8 Each circle shown in the figure is centered at a landmark and has a radius equal to *R*. Any point in the area is covered by at least two landmarks.

twice as shown in Figure 2.8. Also, each target not in C must belong to $\mathcal{SC}_i$ for some $i \in C$. Therefore, any target must be covered by at least two landmarks. Hence proved. □

Theorem 2.3.2 The approximation ratio of $Approx_2$ is 5, that is, $Approx_2$ finds a feasible solution in polynomial time with the number of placed landmarks at most equal to five times the optimum number of landmarks for the LPP.

Proof: It is easy to verify that the number of steps required to implement both the phases of $Approx_2$ is linear in the given number of landmarks. No landmark can cover more than one target in C because the distance between any targets in C is greater than $2R$. Therefore, the number of landmarks required to cover all the targets in C must be equal to $2|C|$. Hence, the number of landmarks in any optimal solution, n^*, must be at least equal to $2|C|$, that is, $n^* \geq 2|C|$. Also, the number of landmarks placed in any feasible solution produced by $Approx_2$, n_f, is equal to $10|C|$. Now, $2|C| \leq n^* \leq n_f \leq 10|C|$. Therefore, $n_f \leq 5n^*$. Hence proved. □

2.4 Numerical Results

The approximation algorithms were implemented on problem instances with 30, 40, 50, 60, 70, 80, 90, and 100 targets. For a given number of targets, we generated 1000 instances. Each target was uniformly sampled from a square area of 500×500 units. A target is covered by a landmark if the distance between the target and the landmark is at most $R = 50$ units. Both the approximation algorithms were implemented on each of the instances, and we computed both the average and the worst-case performance of the algorithms over the tested instances.

For a given problem instance I, the bound on the *a posterior guarantee* provided by an approximation algorithm is defined as $\frac{n_f}{n_{lb}}$, where n_f is the number of landmarks found by the approximation algorithm and n_{lb} is the lower bound on the optimal number of landmarks for the LPP. All the algorithms were coded in MATLAB and the computations were run on a Dell Precision Workstation (Intel Xeon Processor @2.53 GHz, 12 GB RAM). The running time of both approximation algorithms was less than a second for each of the tested instances.

The max and average *a posterior guarantee* of the solutions found by the approximation algorithms are shown in Tables 2.1 and 2.2. Even though the theoretical guarantees of the algorithms $Approx_1$ and $Approx_2$ were 13 and 5, respectively, the numerical results show that the max and average *a posterior guarantee* was less than 2.09 and 2.65, respectively, for both approximation algorithms. In fact, $Approx_1$ with a larger

Table 2.1 Numerical guarantees obtained for *Approx*$_1$.

No. of points	Average a posteriori bound	Max a posteriori bound
30	1.44	1.95
40	1.50	2.00
50	1.53	2.04
60	1.56	2.04
70	1.58	2.04
80	1.60	1.97
90	1.60	2.18
100	1.61	2.07

Table 2.2 Numerical guarantees obtained for *Approx*$_2$.

No. of points	Average a posteriori bound	Max a posteriori bound
30	1.67	2.30
40	1.79	2.36
50	1.86	2.42
60	1.93	2.65
70	1.98	2.57
80	2.03	2.50
90	2.06	2.60
100	2.09	2.59

theoretical guarantee performed better in simulations compared to $Approx_2$. These results imply that the proposed algorithms produced solutions with bounds that are significantly better than the guarantees indicated by the approximation factor. These results also lead to some open questions with regard to the current theoretical results: the approximation ratio of 13 for $Approx_1$ can be significantly improved as outlined in Remark 2.2; the ideas presented in both algorithms may be combined to develop algorithms with better theoretical and numerical guarantees.

2.5 Conclusions

This chapter presented two new approximation algorithms for covering a set of target way points at least twice using a minimum number of landmarks. Simulation results were also presented to corroborate the performance of the proposed algorithms. There are several future directions for this work. First, this chapter used simplistic assumptions on the model of a range sensor; more general models where the performance of a range sensor degrades more smoothly as a function of distance from the landmark can be more useful. Second, as the landmark placement relies very much on the target waypoints it needs to cover; path planning must be coupled with landmark placement. Third, sensors may be heterogeneous; some sensors may be based on range while the other sensors may be based on bearing. Sensor placement algorithms that can address this heterogeneity are useful.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1527748 and Grant No. 1526551.

References

1 Roper, W. and Dutta, S. (2005) Remote sensing and GIS applications for pipeline security assessment, in *ESRI User Conference Proceedings*.

2 Hausamann, D., Zirnig, W., and Schreier, G. (2003) Monitoring of gas transmission pipelines - a customer driven civil UAV application, in *ODAS 2003-5th ONERA-DLR Symposium, June 2003*.

3 O'Young, S. and Hubbard, P. (2007) RAVEN: a maritime surveillance project using small UAV, in *Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference* on, pp. 904–907, doi: 10.1109/EFTA.2007.4416878.

4 Coffed, J. (2014) *The Threat of GPS Jamming: The Risk to An Information Utility*, Exelis Inc.

5 Hoey, D. and Benshoof, P. (2005) Civil GPS systems and potential vulnerabilities, in Proceedings of the 18th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2005), IEEE Computer Society, Long Beach, CA, pp. 1291–1295.

6 Carroll, J.V. (2003) Vulnerability assessment of the U.S. transportation infrastructure that relies on the global positioning system. *Journal of Navigation*, **56**, 185–193, doi: 10.1017/S0373463303002273.

7 Bailey, T. and Durrant-Whyte, H. (2006) Simultaneous localization and mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, **13** (3), 108–117.

8 Durrant-Whyte, H. and Bailey, T. (2006) Simultaneous localisation and mapping (SLAM): Part I the essential algorithms. *IEEE Robotics and Automation Magazine*, **13** (2), 99–110.

9 Song, Y. and Grizzle, J.W. (1995) The extended Kalman filter as a local asymptotic observer for descrete -time nonlinear systems. *Journal of Mathematical Systems, Estimation, and Control*, **5**, 59–78.

10 Sharma, R., Beard, R.W., Taylor, C.N., and Quebe, S. (2012) Graph-based observability analysis of bearing-only cooperative localization. *IEEE Transactions on Robotics*, **28** (2), 522–529.

11 Kurazume, R. and Hirose, S. (1998) Study on cooperative positioning system: optimum moving strategies for CPS-III, in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. **4**, pp. 2896–2903, doi: 10.1109/ROBOT.1998.680642.

12 Sanderson, A.C. (1998) A distributed algorithm for cooperative navigation among multiple mobile robots. *Advanced Robotics*, **12**, 335–349.

13 Roumeliotis, S.I. and Bekey, G.A. (2000) Collective localization: a distributed Kalman filter approach to localization of groups of mobile robots, in *Proceedings of IEEE International Conference on Robotics and Automation ICRA '00*, vol. **3**, pp. 2958–2965, doi: 10.1109/ROBOT.2000.846477.

14 Liu, P. and Lu, D. (2014) A fast 25/6-approximation for the minimum unit disk cover problem, in *CoRR*, http://arxiv.org/abs/1406.3838.

15 Dash, D., Bishnu, A., Gupta, A., and Nandy, S. (2013) Approximation algorithms for deployment of sensors for line segment coverage in wireless sensor networks. *Wireless Networks*, **19** (5), 857–870, doi: 10.1007/s11276-012-0506-4.

16 Vazirani, V.V. (2001) *Approximation Algorithms*, Springer-Verlag.

17 Biniaz, A., Maheshwari, A., Smid, M.H.M., and Liu, P. (2015) A faster 4-approximation algorithm for the unit disk cover problem, in *Proceedings of the 27th Canadian Conference on Computational Geometry, CCCG 2015, Kingston, Ontario, Canada, August 10-12*, 2015, http://research.cs.queensu.ca/cccg2015/CCCG15-papers/12.pdf (accessed 20 October 2016).

18 Rathinam, S. and Sharma, R. (2015) A multiple vehicle path covering problem with localization constraints: formulation and algorithms, in *American Control Conference, 2015. Proceedings of the 2015*, submitted.

3

Robust Coordination of Small UAVs for Vision-Based Target Tracking Using Output-Feedback MPC with MHE

Steven A. P. Quintero, David A. Copp and João P. Hespanha

Center for Control, Dynamical Systems, and Computation, University of California, Santa Barbara, CA, USA

Small unmanned aerial vehicles (UAVs) are relatively inexpensive mobile sensing platforms that are finding increasingly widespread use in both commercial and military sectors due to their ability to autonomously perform tasks that would be too demanding, dangerous, or mundane for a human operator. Such tasks include agricultural monitoring, exploration and mapping, search and rescue, and surveillance and tracking, to name a few. As the proliferation of small UAVs has allowed them to be manufactured at much lower costs than their larger counterparts, these robotic agents are now being deployed as multi-agent teams to perform missions in a cooperative fashion. The increased numbers have enabled the fleet to overcome the limitations of lower quality sensors and reduced computational resources that typically accompany small, inexpensive mobile robots, as they are able to distribute sensing and computation across the fleet. The plural nature of the fleet also enables redundancy and synoptic coverage of an object or area of interest.

One commonly underlying theme for UAV applications is that these robotic agents are faced with gathering measurements and making decisions without the aid of a human operator. Furthermore, the decision making must not only be made in real time as new information becomes available but it must also be done robustly to avoid mission failure. In addition, as a measure of quality can typically be associated with the information

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

gathered, the motion of the individual vehicles can be optimized so as to gather the best *joint* measurements of a given quantity, object, or area of interest. For example, a team of UAVs might be tasked with obtaining a diverse set of pressure measurements in a hurricane, and in the military reconnaissance application of target tracking, a UAV team might be tasked with tracking a moving ground vehicle whose motion is unpredictable. In both cases, the UAVs must coordinate their present actions in light of future measurement gathering performance, and their joint actions must be robust to real-world conditions, such as environmental disturbances and unmodeled dynamics. If fixed-wing aircraft are employed, as they often are for their speed and endurance, the motion constraints of the vehicles, namely airspeed limits and a minimum turning radius, pose a significant challenge for any feedback control or motion planning solution.

One final challenge for a team of UAVs is that typically only noisy, partial information is available for decision making. Hence, a state estimation algorithm is typically employed to estimate unmeasured system states and improve the accuracy of measured states. Thus, the corresponding decision-making algorithm, which is usually independent of the state estimator, must be inherently robust to noise in the estimated system states to avert mission failure, and the overall control solution is one of *output feedback*. In addition, parametric uncertainty may also be present in the system dynamics, and thus estimation of the unknown parameters may be done online in real time to overcome the uncertainty. While such uncertainty may take the form of time constants in the vehicle dynamics, it may also be present in the form of an environmental disturbance such as wind, which can be detrimental to motion planning algorithms if neglected. This is especially true for small or micro air vehicles where wind constitutes a significant portion of the vehicle's ground speed.

Thus, in this chapter, we first aim to highlight solutions for the robust, coordinated control of multi-agent systems that represent the state of the art, where a special emphasis is placed on output-feedback approaches. Second, we present a novel, output-feedback approach that both enables the robust, optimal control of smaller multi-agent systems with nonlinear dynamics and is suitable for real-world implementation, as corroborated by real-time, high-fidelity simulations. We do this in the context of the particularly challenging problem of *vision-based target tracking* with small, fixed-wing UAVs. In this application, one employs multiple, camera-equipped UAVs traveling at a fixed airspeed and constant altitude to jointly measure and track the position of a moving ground vehicle whose motion is unpredictable. The main benefit of focusing on this problem is that it has nearly all of the aforementioned real-world conditions to which agents in a coordinated, multi-agent system must be robust. Namely, the motion planning for the team must be robust to target vehicle motion that may be random, evasive, or otherwise, and hence the evolution of some of the overall system states is unpredictable and may be adverse to the given objective. In addition, planning must be done in spite of noisy, partial information for all vehicle states and unmeasured, nonzero wind conditions, and it must be done with significant foresight, as the motion constraints of the UAVs may cause greedy, myopic approaches to either have poor long-term tracking performance or lose the possibly evasive target altogether. And lastly, one must utilize a motion model for the aircraft dynamics with a sufficient degree of realism, else ignored dynamics will have a detrimental effect on tracking performance.

3.1 Vision-Based Target Tracking

In vision-based target tracking, image processing software is responsible for determining the centroid pixel coordinates of the ground target moving in the image frame. Using these pixel coordinates, along with the intrinsic and extrinsic camera parameters and terrain data, one can estimate the three-dimensional location of the target in inertial coordinates and compute the associated error covariance [1]. This vision-based measurement of the target's position is also referred to as the *geolocation* estimate. The error associated with the geolocation estimate is highly sensitive to the UAV's position relative to that of the target. As the UAV's planar distance from the target increases, the measurement error's covariance grows, and the associated confidence ellipse becomes significantly elongated in the viewing direction. When a UAV is directly above the target, the measurement error is smallest, as the corresponding confidence ellipse is circular. Thus, a UAV would ideally hover directly above the target, but the relative dynamics between a fixed-wing UAV and a moving ground target typically preclude this viewing position from being maintained over a period of time. To mitigate a single UAV's inability to maintain close proximity to the target, one can employ multiple UAVs to gather measurements, which are then fused to obtain an improved geolocation estimate. This is referred to as cooperative (or coordinated) target tracking.

Considerable work has been done in the general area of coordinated target tracking, with coordinated standoff tracking comprising the greatest body of work in this area. In standoff tracking, two UAVs orbit the target at a nominal standoff distance while maintaining orthogonal viewing angles. This practice minimizes the joint/fused geolocation error covariance at the fixed nominal standoff distance, as the confidence ellipses corresponding to the individual measurement errors are orthogonal [2]. Standoff tracking with perfect knowledge of the target state has been studied in [3] and [4], where the most prevalent control strategies involve the use of vector fields and nonlinear feedback. Approaches with only partial information of the target state are presented in [5, 6], and [7]. These works utilize observers, adaptive control, and extended Kalman filtering, respectively, to estimate the full target state. Note that [7] utilizes nonlinear model predictive control (MPC) to achieve the desired standoff configuration for a target that accelerates but is not necessarily evasive.

The preceding works have designed UAV coordination policies that attempt to improve the estimate of the target state without directly solving a dynamic optimization that minimizes some metric of the estimation error. However, a number of works have employed optimal control to achieve this objective. In [8], Sinha et al. demonstrate a decentralized tracking approach wherein each member of a UAV team maximizes its information regarding the target state while accounting for its teammates' track states and survival and detection probabilities as well. Miller et al. utilize the framework of partially observable Markov decision processes (POMDPs) in [9] to enable two UAVs to track a moving ground target and present a new approximate solution, as nontrivial POMDP problems are typically intractable to solve exactly [10]. Stachura et al. [11] employ online receding horizon control to enable two variable-airspeed UAVs to track a stochastic ground target using bearing-only sensors in the presence of packet losses when communicating with the base station where target state estimation takes place. In [12], Ding et al. study the problem of optimally controlling two Dubins vehicles and their pan-tilt-zoom cameras to maximize the geolocation information of a stochastic ground target and show that maintaining orthogonal viewing angles is essential in the case of terrestrial pursuit vehicles and less pronounced for airborne vehicles.

While the preceding optimization-based methods consider short planning horizons, for example, 2–3 s, Quintero et al. consider the optimal coordination of two UAVs to gather the best joint vision-based measurements of a moving target over considerably longer planning horizons of at least 30 s, where no restrictions are placed on the vehicles other than kinematics. First, in [13], Quintero et al. consider two Dubins vehicles tracking a constant-velocity target in multiple scenarios differentiated by target speeds. The results in all scenarios show that coordination of the distances to target is more effective for achieving the said goal than the traditional practice of solely coordinating viewing angles. Second, in [14], Quintero et al. advance the work in [13] by considering a more realistic problem formulation that instead makes use of stochastic fourth-order motion models for both the UAV team and ground target. Using a novel regression-based dynamic programming approach, the authors solve the formidable 9-D stochastic optimal control problem offline and show that distance coordination can still be achieved under more realistic settings that include UAV roll dynamics, environmental disturbances, and dynamical uncertainty.

There is another notable work in the area of target tracking that is outside both of the aforementioned categories of standoff tracking and optimization-based control. For example, in [15], Triplett et al. focus on the challenges of communication, control, and state estimation as multiple UAVs track a moving ground vehicle. In this work, a decentralized extended Kalman filter is used to estimate the target state while a turn-rate based steering controller is utilized to have the pursuit group's centroid track the target, achieve a desired intervehicle spacing, and avoid obstacles. The results show that increased communication frequency and sensor reliability improve the estimation performance of the system. In addition, a few works, namely [16] and [17], have proposed using sinusoidal turn-rate control inputs that approximate the optimal behavior of [13] at higher speeds.

While all of the preceding works do not constitute an exhaustive literature review on target tracking approaches, they are representative of the primary techniques taken to address this challenging problem. Furthermore, in all of the aforementioned works, at least one or more assumptions are made that create severe hindrances for real-world implementation. More specifically, the works mentioned thus far assume at least one of the following, either explicitly or implicitly:

1) Coordinated circular trajectories are optimal, namely those trajectories resulting from coordinated standoff tracking.
2) The UAVs are controlled by commanding turn rate, so roll dynamics are ignored.
3) The UAV airspeed can be changed quickly and reliably over a significant range.
4) Target motion is predictable.
5) Greedy planning horizons are adequate for optimal tracking.
6) All system states are known exactly.

In addition, the approaches not relying on optimization do not always place explicit constraints on the control effort. We also highlight the fact that the target's motion may not only be unpredictable but also evasive, which was not considered in any of the previous works and requires a robust planning solution with adequate foresight to mitigate the effects of the inherent dynamical limitations of the fixed-wing UAVs. The present work avoids the said practices and assumptions to promote a more practical solution that yields robust coordination under the following realistic conditions: unknown constant wind, non-negligible roll dynamics with roll-angle setpoint limits, evasive target motion, and the availability of only noisy, partial information of the overall system states.

An output-feedback control approach that can be used to achieve the desired robustness was recently introduced by Copp and Hespanha [18] and combines robust MPC with moving horizon estimation (MHE). As described in [19] and

[20], robust MPC involves an online dynamic optimization aimed at minimizing a cost function over a finite planning horizon in light of worst-case disturbances on a dynamical system. MHE also involves an online optimization problem but for the purpose of state estimation of nonlinear systems, and it has been shown to have advantages over state-of-the-art alternatives [21]. While the two optimizations have traditionally been computed separately, in the framework of [18], the two are combined into a single min–max optimization. More specifically, a desired cost function is maximized with respect to disturbance and measurement noise variables and minimized with respect to control input variables. The min–max optimization provides state estimates over a fixed, finite window into the past and an optimal control input sequence into the future that is simultaneously robust to worst-case estimates of the state as well as worst-case disturbances to the plant. Through real-time high-fidelity flight simulations, this combined robust MPC/MHE approach is shown to be a viable, practical solution for the present particularly challenging nonlinear problem of autonomous vehicle coordination.

The work in this chapter is based on work that appears in the conference paper [22] in a simpler form. In this chapter, significant extensions to [22] have been made. Specifically, whereas the traditional third-order Dubins vehicle model was utilized in [22], here we consider a more realistic fourth-order model that incorporates roll dynamics. Furthermore, the problem formulation in the earlier conference paper considered the ideal condition of no wind. As wind can have a considerable effect on small unmanned aircraft, the present tracking solution incorporates adaptive estimates of the wind into the online MPC/MHE optimization, thereby explicitly addressing the effects of wind and adding robustness to the motion planning of the UAVs. Lastly, the Dubins vehicle model was used to validate the approach taken in [22]. In contrast, Section 3.4 illustrates the effectiveness of the approach in realistic environments by using high-fidelity simulations that make use of an aircraft model with six degrees of freedom and target logs from actual tracking experiments at Camp Roberts, California, USA.

The remainder of the chapter is organized as follows. Section 3.2 describes the dynamics and measurement model that compose the problem of vision-based target tracking. Section 3.3 discusses the cost function and the robust output-feedback MPC/MHE solution. Section 3.4 presents and discusses simulation results for multiple scenarios. Finally, Section 3.5 provides conclusions and plans for future work.

3.2 Problem Formulation

Consider two camera-equipped UAVs tasked with estimating the state of a target vehicle moving evasively in the ground plane. The UAVs are fixed-wing aircraft that fly at a constant altitude and have an autopilot that regulates roll angle, airspeed, and altitude to the desired setpoints via internal feedback loops. Furthermore, these underactuated aircraft are assumed to fly at a constant airspeed since the range of permissible airspeeds for such small aircraft may be very limited, as noted in [23] and Section 5.1 of [4]. The roll angle setpoint is hence the sole control input that affects the horizontal plant dynamics. The target vehicle moves in the ground plane and is subject to a maximum acceleration and maximum speed that is considerably less than the slowest UAV's airspeed, which means that the groundspeed of each UAV will always exceed that of the target in light wind conditions. Each UAV makes measurements of the target's position using a gimbaled video camera, and we assume that the target is detected at all times and kept in the center of the camera's field of view by onboard software. We first discuss the dynamical models for each type of vehicle and then proceed to describe their measurement models.

3.2.1 UAV Dynamics

The basis for our UAV motion model is the kinematic guidance model given by Equation 9.20 of [24], where the altitude, altitude rate, and airspeed states have been omitted since the said quantities are treated as being constant in this work. Note that similar models have also been used in [14, 25], and [26]. To present the model, we assume that UAV j, where $j \in \{1, 2\}$, flies at a constant airspeed s_j and at a fixed altitude h_j. Its roll-angle setpoint is denoted by u_j and has a maximum absolute limit of

$\overline{u} \in \mathbb{R}_{>0}$, which we take to be the same for both UAVs without loss of generality. Accordingly, $u \in \mathcal{U} := [-\overline{u}, \overline{u}] \times [-\overline{u}, \overline{u}]$. The state of UAV j comprises its planar position $\mathfrak{p}_j \in \mathbb{R}^2$, its heading angle $\psi_j \in \mathbb{R}$, and its roll angle $\phi_j \in \mathbb{R}$. Furthermore, we denote the state of UAV j by $\xi^{(j)} = [\xi_1^{(j)}\ \xi_2^{(j)}\ \xi_3^{(j)}\ \xi_4^{(j)}]^\top \in \mathbb{R}^4$, where $(\xi_1^{(j)}, \xi_2^{(j)}) := \mathfrak{p}_j$, $\xi_3^{(j)} := \psi_j$, and $\xi_4^{(j)} := \phi_j$. The position is given in an east-north-up coordinate frame while the roll and heading angles are used in a 3-2-1 (yaw-pitch-roll) Euler angle sequence that describes the transformation from the vehicle-fixed inertial frame to the aircraft body frame, where the pitch angle is assumed to be zero in this work. We also assume that the UAV flies in a nonzero constant wind $w \in \mathbb{R}^2$, which is an unknown quantity that must be estimated. Furthermore, we assume that the wind speed is no more than half the minimum UAV airspeed, that is, $w \in \mathcal{W}$, where

$$\mathcal{W} := \{v \in \mathbb{R}^2 | \|v\| \leq 0.5 \min\{s_1, s_2\}\}. \tag{3.1}$$

Thus, the kinematic model for UAV j used in this work is given by

$$\frac{\mathrm{d}\xi^{(j)}(t)}{\mathrm{d}t} = F(\xi^{(j)}, u_j) := \begin{pmatrix} s_j \cos \xi_3^{(j)} + w_1 \\ s_j \sin \xi_3^{(j)} + w_2 \\ (\alpha_g / s_j)\phi_j \\ -\alpha_{\phi,j}(\phi_j - u_j), \end{pmatrix}. \tag{3.2}$$

where α_g is the vertical acceleration due to gravity and $1/\alpha_{\phi,j}$ is the time constant for UAV j's roll dynamics that are governed by the implementation of the autopilot's control loops and state estimator. Note that we have used the small-angle approximation for the roll angle in the heading rate dynamics, that is, $\dot{\psi}_j = (\alpha_g / s_j) \tan \phi_j \approx (\alpha_g / s_j)\phi_j$. The reasons for this are discussed next.

While the majority of work on target tracking treats the problem in continuous time, this work addresses the problem in discrete time since measurements of the target's position are available at discrete time instances $t = kT_s$ s, where $k \in \mathbb{Z}_{\geq 0}$ and $T_s > 0$ is the measurement sampling period. Accordingly, we assume synchronized zero-order holds of T_s s on the UAVs' control inputs. The corresponding discrete-time equations of motion are denoted by

$$\xi^{(j)+} = f_a(\xi^{(j)}, u_j) = \begin{bmatrix} f_{a,p}(\xi^{(j)}, u_j) \\ f_{a,o}(\xi^{(j)}, u_j) \end{bmatrix},$$

where the subscript "a" refers to the fact that the discrete-time dynamics are those of an air vehicle. Furthermore, we have partitioned the discrete-time dynamics into the position dynamics $f_{a,p}(\xi^{(j)}, u_j) : \mathbb{R}^4 \rightarrow \mathbb{R}^2$ corresponding to $\mathfrak{p}_j$ and the orientation dynamics $f_{a,o}(\xi^{(j)}, u_j) : \mathbb{R}^4 \rightarrow \mathbb{R}^2$ corresponding to (ψ_j, ϕ_j). As the system of differential equations given by (3.2) does not readily emit an analytic solution for the position update equations $f_{a,\mathfrak{p}}(\xi^{(j)}, u_j)$, we utilize a second-order Lie series approximation that is well suited for numerical optimization. Namely, we approximate the discrete-time position dynamics as follows:

$$\begin{aligned} &f_{a,p}(\xi^{(j)}, u_j) \\ &\quad\approx \begin{bmatrix} I_2 & \mathbf{0}_{2\times 2} \\ \mathbf{0}_{2\times 2} & \mathbf{0}_{2\times 2} \end{bmatrix} \left(\xi^{(j)} + T_s F(\xi^{(j)}, u_j) + \frac{T_s^2}{2} \frac{\partial F}{\partial \xi^{(j)}} F(\xi^{(j)}, u_j) \right), \end{aligned}$$

where I_n is an $n \times n$ identity matrix, $\mathbf{0}_{n\times n}$ is an $n \times n$ matrix of zeros, and $\partial F / \partial \xi^{(j)}$ is the Jacobian of (3.2) and yields the following:

$$\frac{\partial F}{\partial \xi^{(j)}} F(\xi^{(j)}, u_j) = \begin{pmatrix} \alpha_g \xi_4^{(j)} \sin \xi_3^{(j)} \\ -\alpha_g \xi_4^{(j)} \cos \xi_3^{(j)} \\ (\alpha_g \alpha_{\phi,j} / s_j)(\xi_4^{(j)} - u_j) \\ \alpha^2 (\xi_4^{(j)} - u_j) \end{pmatrix}. \tag{3.3}$$

To determine the orientation dynamics, we note that

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} \xi_3^{(j)}(t) \\ \xi_4^{(j)}(t) \end{bmatrix} = \begin{bmatrix} 0 & \alpha_g / s_j \\ 0 & -\alpha_{\phi,j} \end{bmatrix} \begin{bmatrix} \xi_3^{(j)} \\ \xi_4^{(j)} \end{bmatrix} + \begin{bmatrix} 0 \\ \alpha_{\phi,j} \end{bmatrix} u_j = A_o \begin{bmatrix} \xi_3^{(j)} \\ \xi_4^{(j)} \end{bmatrix} + b_o u_j.$$

By defining the matrix $\Phi \in \mathbb{R}^{2\times 2}$ and vector $\Gamma \in \mathbb{R}^2$ as $\Phi := \exp(A_o T_s)$ and $\Gamma := \int_0^{T_s} \exp(A_o \tau) b_o d\tau$, respectively, we have

$$f_{a,o}(\xi^{(j)}, u_j) = \Phi \begin{bmatrix} \xi_3^{(j)} \\ \xi_4^{(j)} \end{bmatrix} + \Gamma u_j.$$

Before presenting the target's dynamics, we note that the use of $\tan \phi$ for the heading rate dynamics in (3.2) leads to a more complicated expression for (3.3), which includes multiplication by $\tan \phi$ and division by $\cos^2 \phi$. Its use would also necessitate an approximation to the heading rate dynamics as well. Furthermore, if one wished to employ a higher order Lie series for the solution of (3.2), the use of $\tan\phi$ adds even more complexity to the results. As such complexity can pose a great deal of difficulty for numerical optimization, we have avoided the use of $\tan\phi$ in (3.2). Nonetheless, we shall see that the small-angle and Lie series approximations lend themselves to tractable nonlinear optimizations that can be solved in real time to produce good tracking performance.

3.2.2 Target Dynamics and Overall State Space

We place no nonholonomic constraints on the ground vehicle and simply model the target as a double integrator moving in the ground plane. The target's planar position is denoted by $\mathfrak{p}_g \in \mathbb{R}^2$ and is measured in the same local east-north-up coordinate frame as the UAVs. The corresponding velocity vector is $\mathfrak{v} \in \mathbb{R}^2$, and the acceleration input vector is $d \in \mathbb{R}^2$. The overall target state is $\eta = [\eta_1\ \eta_2\ \eta_3\ \eta_4]^\top \in \mathbb{R}^4$, where $(\eta_1, \eta_2) := \mathfrak{p}_g$ and $(\eta_3, \eta_4) := \mathfrak{v}$. We assume a T_s-s ZOH on the target's acceleration input synchronized with that of both UAVs, yielding the straightforward discrete-time linear dynamics

$$\eta^+ = f_g(\eta, d) = A\eta + Bd, \tag{3.4}$$

where

$$A = \begin{bmatrix} I_2 & T_s I_2 \\ \mathbf{0}_{2\times 2} & I_2 \end{bmatrix} \text{ and } B = \begin{bmatrix} (T_s^2/2) I_2 \\ T_s I_2 \end{bmatrix}.$$

To keep the problem realistic and well-posed, we take the target's acceleration input d to belong to

$$\mathcal{D} := \{d \in \mathbb{R}^2 |\ \|\mathfrak{v} + dT_s\|_2 \le \overline{v}, \|d\|_\infty \le \overline{d}\},$$

where $\overline{v}$ is the maximum allowable target speed, and $\overline{d}$ is the maximum absolute acceleration along either the east or north direction. Typically, we take $\overline{v}$ to be less than the smaller of the two UAV airspeeds so that the problem is well-posed.

Now that we have presented all vehicle models, we define the overall state as $x := (\xi^{(1)}, \xi^{(2)}, \eta) \in \mathbb{R}^{12}$. The overall dynamics are thus given by

$$x^+ = f(x, u, d) := \begin{pmatrix} f_a(\xi^{(1)}, u_1) \\ f_a(\xi^{(2)}, u_2) \\ f_g(\eta, d) \end{pmatrix}.$$

3.2.3 Measurement Error Models

We turn our attention to the overall measurement model in vision-based target tracking. The measurement vector associated with the state of UAV j is denoted by $y_a^{(j)} \in \mathbb{R}^4$ and is given by

$$y_a^{(j)} = \xi^{(j)} + n_a^{(j)}, \quad n_a^{(j)} = [n_{a,1}^{(j)}\ n_{a,2}^{(j)}\ n_{a,3}^{(j)}\ n_{a,4}^{(j)}]^\top, \tag{3.5}$$

where $(n_{a,1}^{(j)}, n_{a,2}^{(j)}) \sim \mathcal{N}(0, \sigma_{\mathfrak{p}}^2 I_2)$, $(n_{a,3}^{(j)}, n_{a,4}^{(j)}) \sim \mathcal{N}(0, \sigma_\theta^2 I_2)$, $\sigma_{\mathfrak{p}}^2$ is the variance of the uncorrelated noise on the UAV's north and east position coordinates, and σ_θ^2 is the variance on the UAV's Euler angles, namely roll and yaw (heading).

To describe the measurement error on the target's position, we primarily follow the work of [1] and summarize key ideas here. Since we use similar notation for vectors and matrices, we refer the reader to the aforementioned work for illustrations of these quantities. We begin by noting that each UAV's camera makes image-plane measurements of the target. The dominant source of geolocation error arises from the error in the sensor attitude matrix $T_S^T(\boldsymbol{\theta}_j)$ that relates the coordinates of the line-of-sight vector $\mathfrak{u}_{\mathrm{j}}^S$, which is measured from the UAV to the target in the north-east-down sensor frame (centered at UAV j's position), to the coordinates of the same vector in the local east-north-up topographic coordinate frame. This transformation is a nonlinear function of the 3-2-1 Euler-angle sequence of yaw, pitch, and roll denoted by $\boldsymbol{\theta}_j \in \mathbb{R}^3$. Image tracking software controls the camera's gimbal platform to keep the target in the center of the camera's field of view and reports the Euler angles of the camera sensor as well as the line-of-sight vector $\mathfrak{u}_{\mathrm{j}}^S$. Here, a superscript "$S$" denotes a quantity in the sensor coordinate frame while the absence thereof indicates a quantity in the topographic coordinate frame.

The three-dimensional target position measured by UAV j with 3D position $\mathfrak{s}_j = [\mathfrak{p}_j^\top, h_j]^\top$ is denoted by $\mathfrak{o}_j$. Its estimate is given by

$$\hat{\mathfrak{o}}_j = \hat{\mathfrak{s}}_j + \hat{r}_j T_S^T(\hat{\boldsymbol{\theta}}_j)\mathfrak{u}_{\mathrm{j}}^S = \hat{\mathfrak{s}}_j + \hat{r}_j \hat{\mathfrak{u}}_j,$$

where

$$\begin{aligned} \hat{\mathfrak{s}}_j &= \mathfrak{s}_j + \tilde{\mathfrak{s}}, \quad \tilde{\mathfrak{s}} \sim \mathcal{N}(0, \mathrm{diag}(\sigma_{\mathfrak{p}}^2 I_2, \sigma_a^2)), \\ \hat{\boldsymbol{\theta}}_j &= \boldsymbol{\theta}_j + \tilde{\boldsymbol{\theta}}, \quad \tilde{\boldsymbol{\theta}} \sim \mathcal{N}(0, \sigma_\theta^2 I_3), \end{aligned} \tag{3.6}$$

and σ_a^2 denotes the variance of the measurement noise on the UAVs' altitude h_j. Also, the 3D distance from UAV j to the target is denoted by $r_j = \| \mathfrak{o}_j - \hat{\mathfrak{s}}_j \|_2$, and its estimate $\hat{r}_j$ is provided by the flat-Earth approximation $\hat{r}_j = (h_0 - \hat{\mathfrak{s}}_{j,3})/\hat{\mathfrak{u}}_{j,3}$, where h_0 is the height of the ground plane in the topographic coordinate frame and is taken to be zero in this work without loss of generality. Since all camera angles are measured with respect to the UAV attitude, we take the noise on the estimate of the camera's attitude to be the same as that on the estimate of the UAV's attitude. Thus, for example, the noise on the estimate of UAV j's heading angle is the same as that on the estimate of its camera's yaw angle, that is, the first element of $\boldsymbol{\theta}_j$.

From the preceding measurement equation, one can show that the covariance $P_{\mathfrak{o},j} \in \mathbb{R}^{3\times3}$ associated with the error $\tilde{\mathfrak{o}}_j := \hat{\mathfrak{o}}_j - \mathfrak{o}_j$ in the three-dimensional position of the target is proportional to the product of r_j^2 and the covariance of the Euler-angle sequence estimate $\hat{\boldsymbol{\theta}}_j$ given in (3.6). The exact analytic expression for $P_{\mathfrak{o},j}$ is derived in both [1] and [13] and is omitted here for brevity. Since we are tracking in the ground plane, only the upper left 2×2 submatrix of $P_{\mathfrak{o},j}$ is relevant and is denoted by $P_j \in \mathbb{R}^{2\times2}$.

Since the UAVs collect independent measurements of the target, the fused measurement y_g of the target's true position $\mathfrak{p}_g$ can be computed using the best linear unbiased estimate, which is as follows:

$$\begin{aligned} y_g &= \mathcal{P}(P_1^{-1}\hat{\mathfrak{p}}_g^{(1)} + P_2^{-1}\hat{\mathfrak{p}}_g^{(2)}) \\ &= [I_2\ \mathbf{0}_{2\times2}]\eta + n_g = \mathfrak{p}_g + n_g, \end{aligned} \tag{3.7}$$

where $\mathcal{P} = (P_1^{-1} + P_2^{-1})^{-1}$, $\hat{\mathfrak{p}}_g^{(j)} = [I_2\ \mathbf{0}_{2\times1}]\hat{\mathfrak{o}}_j$, $\mathbf{0}_{m\times n}$ denotes the $m \times n$ zeros matrix, and n_g is noise that is adequately approximated by a zero-mean Gaussian distribution with covariance $\mathcal{P}$. The confidence ellipse corresponding to the fused geolocation error covariance (GEC) $\mathcal{P}$ has the property that it is small when at least one UAV is close to the target and only slightly less when both aircraft are directly above the target. Therefore, it is advantageous for at least one UAV to be near the target at any given time. Note that the expressions for the true covariances P_j are based on the true Euler angles and true vehicle altitudes, which are unknown quantities, and hence estimates of the covariances $\hat{P}_j$ based on the raw Euler angle estimates $\hat{\boldsymbol{\theta}}_j$ and raw altitude estimates $\hat{\mathfrak{s}}_{j,3}$ are used in (3.7) in practice.

Finally, the measurement model corresponding to the overall state x is given by combining (3.5) and (3.7) as follows:

$$y := (y_a^{(1)}, y_a^{(2)}, y_g) = Cx + n, \tag{3.8}$$

where $n := (n_a^{(1)}, n_a^{(2)}, n_g)$ and $C := [I_{10}\ \mathbf{0}_{10\times 2}]$. Since the target velocity is not measured directly, the control law used in this framework will be based on output feedback.

3.3 Robust Output-Feedback MPC/MHE

If only one UAV is considered in the target tracking problem, the problem can be posed as a two-player zero-sum game in which the UAV tries to minimize its squared 3D distance to the target while the target tries to maximize this same quantity. Moreover, by seeking to minimize a scalar quantity proportional to its individual covariance, the UAV's aim becomes that of gathering the best individual vision-based measurements of the target. In the two-UAV case, the UAVs ideally coordinate their movements in order to ensure that at least one UAV is close to the target to keep the fused GEC comparatively low. This practice translates into the UAVs gathering the best joint vision-based measurements of the target. In addition, the UAVs should keep their individual distances to the target sufficiently small to maintain adequate resolution of the target in the camera's image plane for effective visual detection. This motivates us to choose the following criterion that the UAVs (Player 1) would like to minimize and the target (Player 2) would like to maximize:

$$g(x) := \beta_1 \frac{r_1^2 r_2^2}{r_1^2 + r_2^2} + \beta_2 (r_1^2 + r_2^2). \tag{3.9}$$

In this criterion, β_1 and β_2 are positive weighting constants. The first term in (3.9) is motivated by noting that the size of the confidence ellipse associated with P_j is proportional to r_j^2 and that the fused GEC has the form $\mathcal{P} = (P_1^{-1} + P_2^{-1})^{-1}$. Moreover, the previous matrix expression is simplified to one that is scalar and more compatible with numerical optimization by replacing the individual covariances with the respective 3D distances. This term enforces distance coordination so that one UAV is always close to the target, which improves measurement quality just as in [13]. The second term in (3.9) penalizes the individual UAV distances to the target to ensure that the size of the target in each UAV's image plane is sufficiently large for reliable detection by image processing software. While other optimality criteria may be considered, we aim to utilize a simpler expression than those found in [12] and [13] that achieves similar behavior and lends itself to efficient numerical computation. We shall see that the distance coordination of [13] is indeed induced by choosing the criterion (3.9).

For our control approach, we use the output-feedback MPC with MHE approach described in [18]. This requires us to solve a finite-horizon online optimization problem at each step k. Solving this online optimization problem uses the last $L \in \mathbb{Z}_{\geq 1}$ output measurements in order to give us an estimate of the current state at step k (the MHE problem) while simultaneously computing policies for both Players 1 and 2 to use for the next $K \in \mathbb{Z}_{\geq 1}$ steps into the future (the MPC problem).

As previously discussed in Section 3.1, the problem we address in this chapter is similar to that of coordinating UAVs for target tracking described in [22], but here we incorporate the more realistic UAV model given by (3.2) that includes unknown wind disturbances. To solve this problem, while simultaneously computing the state estimates and future control sequence, we can estimate the unknown wind w by including the estimate $\hat{w} \in \mathcal{W}$ as an optimization variable, where $\mathcal{W}$ is given by (3.1). More details about using this combined MPC/MHE approach for adaptive control and learning can be found in [27].

Given a discrete-time signal $z : \mathbb{Z}_{\geq 0} \to \mathbb{R}^n$ and two steps $k_0, k \in \mathbb{Z}_{\geq 0}$ with $k_0 \leq k$, we denote by $z_{k_0:k}$ the sequence $\{z_{k_0}, z_{k_0+1}, \ldots, z_k\}$. Then, formally, the control objective is to select the control signal $u_k \in \mathcal{U}$, $\forall k \in \mathbb{Z}_{\geq 0}$ so as to minimize a criterion of the form

$$
\begin{aligned}
&J_k(x_{k-L}, u_{k-L:k+K-1}, d_{k-L:k+K-1}, y_{k-L:k}) := \\
&\quad \sum_{\ell=k}^{k+K} g(x_\ell) + \sum_{\ell=k}^{k+K-1} \lambda_u \parallel u_\ell - \varphi_\ell \parallel^2 \\
&\qquad - \sum_{\ell=k-L}^{k+K-1} \lambda_d \parallel d_\ell \parallel^2 - \sum_{\ell=k-L}^{k} \lambda_n \parallel y_\ell - Cx_\ell \parallel_2^2, \qquad (3.10)
\end{aligned}
$$

where $g(x)$ is given in (3.9), the second term in the right-hand side penalizes the difference between the roll command pair u_ℓ and the roll angle pair $\varphi_\ell := (\phi_{1,\ell}, \phi_{2,\ell}) \in \mathbb{R}^2$ at time ℓ, and λ_u, λ_d, and λ_n are positive scalars. Here, $\phi_{j,\ell}$ refers to UAV j's roll angle at time ℓ. The scalar λ_u is a tuning weight that determines the penalty for the UAVs applying roll commands that are different from their current roll angles. Therefore, λ_u may be chosen large enough to ensure that the rate of change of the roll angle does not exceed an actuator's limitations. The scalar λ_d is a tuning weight that determines how much of a penalty the target incurs when using large accelerations. Finally, the scalar λ_n is a tuning weight that can be thought of as determining how much of a penalty is incurred for choosing less likely values for the measurement noise (in this case, larger values).

Defining the criterion as in (3.10) emphasizes the dependence on the unknown initial state x_{k-L}, the unknown input sequence for the target $d_{k-L:k+K-1}$, the measured output sequence $y_{k-L:k}$, and the input sequence for the UAVs $u_{k-L:k+K-1}$. The input sequence $u_{k-L:k+K-1}$ is comprised of two distinct sequences: the (known) past inputs $u_{k-L:k-1}$ that have already been applied and the future inputs $u_{k:k+K-1}$ that still need to be selected.

At a given step $k \in \mathbb{Z}_{\geq 0}$, we do not know the values x_{k-L} and $d_{k-L:k+K-1}$ on which the criterion (3.10) depends, so we optimize this criterion under worst-case assumptions on these variables, leading to the following finite-dimensional min–max optimization

$$\min_{\hat{u}_{k:k+K-1|k} \in \mathcal{U}} \quad \max_{\hat{x}_{k-L|k} \in \mathcal{X}, \hat{d}_{k-L:k+K-1|k} \in \mathcal{D}, \hat{w} \in \mathcal{W}}$$

$$J_k(\hat{x}_{k-L|k}, u_{k-L:k-1}, \hat{u}_{k:k+K-1|k}, \hat{d}_{k-L:k+K-1|k}, y_{k-L:k}), \qquad (3.11)$$

where the arguments $u_{k-L:k-1}, \hat{u}_{k:k+K-1|k}$ correspond to the sequence $u_{k-L:k+K-1}$ in the definition of $J_k(\cdot)$ in (3.10). The subscript $_{\cdot|k}$ in the (dummy) optimization variables in (3.11) emphasizes that this optimization is repeated at each time step $k \in \mathbb{Z}_{\geq 0}$.

At different time steps, these optimizations typically lead to different solutions, which generally do not coincide with the real control input, target input, and noise. We can view the optimization variables $\hat{x}_{k-L|k}$, $\hat{d}_{k-L:k+K-1|k}$, and $\hat{w}$ as (worst-case) estimates of the initial state, target input, and wind speed, respectively, based on the past inputs $u_{k-L:k-1}$ and outputs $y_{k-L:k}$ available at time k. The past measurement noises $n_{k-L:k}$ are not included as explicit optimization variables because they are uniquely defined by the output equation (3.8) and the choices of the optimization variables $\hat{x}_{k-L|k}$ and $\hat{d}_{k-L:k-1|k}$.

Inspired by MPC, at each time k, we use as the control input the first element of the sequence

$$\hat{u}^*_{k:k+K-1|k} = \{\hat{u}^*_{k|k}, \hat{u}^*_{k+1|k}, \hat{u}^*_{k+2|k}, \ldots, \hat{u}^*_{k+K-1|k}\} \in \mathcal{U}$$

that minimizes (3.11), leading to the following control law:

$$u_k = \hat{u}^*_{k|k}, \quad \forall k \geq 0.$$

3.4 Simulation Results

Now we demonstrate the effectiveness of the MPC/MHE control approach to the problem of two fixed-wing UAVs performing vision-based target tracking of a moving ground vehicle. To underscore the robustness of the approach to real-world conditions such as constant winds and unmodeled aircraft dynamics, we present the results of several simulations that were performed in real time using the flight simulator Aviones [28]. This simulator utilizes an aircraft model with six degrees of freedom and allows the user to simulate constant winds. Furthermore, we simulate three different scenarios. The first scenario involves tracking a constant-velocity target, which represents the "easiest" target to be tracked. The second scenario involves tracking an evasive target, where the worst-case target acceleration d^*_k, computed by solving the optimization (3.11), is applied as the target's actual input. This results in the most "difficult" target to be tracked. Finally, the third scenario uses an experimental target log that was generated by a human physically driving a vehicle casually yet unpredictably. This scenario represents the most realistic target motion. In each of these scenarios, realistic levels of noise and unmeasured wind disturbances are included in the models presented in Section 3.2. For each of

Table 3.1 Simulation parameters.

Parameter	Description	Value	Units
$\overline{u}$	Max UAV roll angle	35	deg.
(s_1, s_2)	UAV airspeeds	$(15, 15)$	m/s
$\overline{d}$	Max target accel.	$3/\sqrt{2}$	m/s^2
$\overline{v}$	Max target speed	10	m/s
$\sigma_{\mathfrak{p}}^2$	N/E position variance	2.5^2	m^2
σ_a^2	Altitude variance	4^2	m^2
σ_θ^2	Euler angle variance	3^2	deg.^{2}
(h_1, h_2)	UAV altitudes	$(40, 45)$	m
β_1	Coord. coefficient	0.04	—
β_2	Dist. coefficient	0.002	—
λ_u	UAV roll-angle coefficient	5	—
λ_d	Target acceleration coefficient	10	—
λ_n	Noise coefficient	40	—
T_s	Sampling period	1	s
L	Backward horizon	8	—
K	Forward horizon	10	—

the three target scenarios, three different wind speeds are also considered, namely 0, 3, and 6 m/s.

The parameters pertaining to all three simulation scenarios are provided in Table 3.1. Given the maximum UAV turn rate, the total time it takes a UAV to make a full loop is $2\pi s_1/(\alpha_g \tan \overline{u}) \approx 13.72$ s, where we have used the relationship of bank angle to turn rate given by Equation 9.20 of [24]. Hence, the future planning horizon of $KT_s = 10$ s allows the UAVs to consider the impact of beginning to loop around the target. In addition, in the cost function (3.9), the coefficients β_1 and β_2 are chosen to place a greater emphasis on distance coordination than on keeping individual distances to the target small. The maximum target speed of 10 m/s is chosen smaller than the fixed airspeed of the UAVs (15 m/s) in order to make the problem well-posed for wind speeds less than 5 m/s. The UAVs fly at fixed altitudes of 40 and 45 m, respectively. Other parameters included in Table 3.1 include the scalar cost function weights

Table 3.2 Computation time and percent of convergence.

Target scenario	Min. (ms)	Mean (ms)	Std. dev. (ms)	Converge (%)
Constant-Velocity, 0 m/s wind	30	37	19	93
Constant-Velocity, 3 m/s wind	30	90	61	94
Constant-Velocity, 6 m/s wind	29	82	56	95
Evasive, 0 m/s wind	31	49	26	97
Evasive, 3 m/s wind	31	45	26	97
Evasive, 6 m/s wind	30	45	30	99
Log, 0 m/s wind	30	42	24	92
Log, 3 m/s wind	30	51	37	91
Log, 6 m/s wind	30	54	29	95

λ_u, λ_d, and λ_n, the forward and backward horizon lengths K and L, respectively, as well as the variances for the UAVs' positions, altitudes, and Euler angles.

To solve the nonlinear, nonconvex, min–max optimization problem (3.11) at each time step, we use a primal-dual-like interior point method that is designed specifically for this MPC/MHE approach and is described in [29]. Table 3.2 shows the computation times using the C programming language on a laptop with a 2.3 GHz Intel® Core™ i7 processor for each scenario that we investigated. If the solver does not converge to a solution within a reasonable number of iterations (a maximum of 140 in this case), the optimization is reattempted up to a maximum of 4 additional times with a new random start condition each time. If the solver fails to converge after 5 attempts, the optimal control from the previous time step is used, that is, $u_k = \hat{u}^*_{k|k-1}$. The last column in the table shows the percentage of time steps for which the solver successfully converges in one of those five allowed attempts and the total computation time is less than the sampling time of $T_s = 1$ s. Even for this difficult nonlinear, nonconvex problem, the solver converges over 90% of the time for every scenario. We note that the percent of convergence is highest for the evasive target scenario because the true trajectories in this scenario best match the solution of the min–max optimization, which computes trajectories based on worst-case evasive target motion.

The other columns in Table 3.2 show the minimum and mean total computation times, as well as their standard deviations for all of the time steps in which the solver converged. Regardless of the scenario, the minimum computation time is about 30 ms. The mean computation times vary, but all of them are between 37 and 90 ms. These computation times clearly show that this problem can be solved in real time using the primal-dual-like interior point method described in [29].

To determine the steady-state tracking performance of each scenario, 5 min of steady-state behavior is considered, where the effects of initial conditions have been removed by discarding 30 s of initial data. During the initial 30 s, the UAVs circle a stationary target, and the past measurement buffer is filled.

3.4.1 Constant-Velocity Target

We first consider a constant-velocity target, that is, $d_k \equiv 0$ in (3.4), where the target travels at 7.5 m/s (one half of the UAVs' fixed speed) in the positive x-direction. The UAVs' inputs are computed by solving the min–max optimization (3.11) and hence are prepared for worst-case target motion as well as worst-case noise and wind.

The results for this scenario with the UAVs subjected to 6 m/s wind to the east, that is, $w = [6 \quad 0]^\top$, where w is presented in (3.2), are provided in Figures 3.1–3.3. In Figure 3.1, one can see how the UAVs make loops so that their average (ground) speed matches that of the target. Figure 3.2 indicates that this is done in

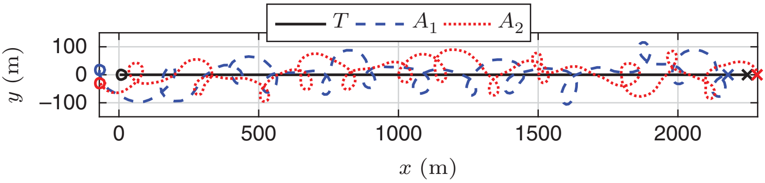


Figure 3.1 Trajectories of two UAVs, subject to 6 m/s wind, tracking a constant-velocity target over a 5-min window. The starting positions of all vehicles are denoted by an "o" while the ending positions are indicated by an "×". In the legend, $\mathcal{T}$ corresponds to the target while $\mathcal{A}_1$ and $\mathcal{A}_2$ refer to the UAVs.

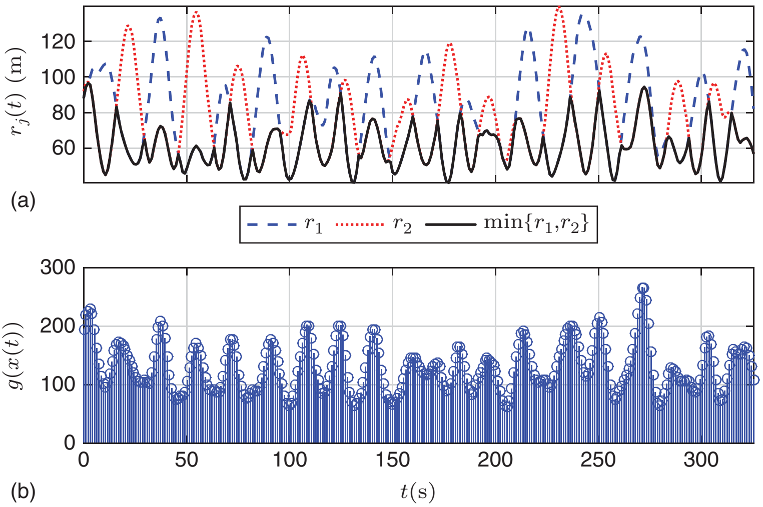


Figure 3.2 3D distances r_j (a) and stage cost $g(x)$ (b) for two UAVs, subject to 6 m/s wind, tracking a constant-velocity target. The solid black line shows $\min\{r_1, r_2\}$.

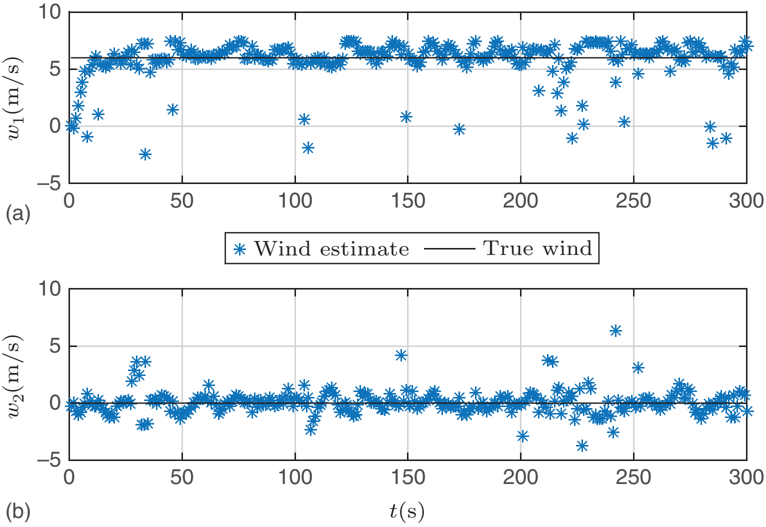


Figure 3.3 Estimate of the wind speed in the *x*-direction (a) and *y*-direction (b) for the constant-velocity target scenario.

a coordinated manner so that at least one UAV is never very far from the target, as indicated by the solid black curve depicting $\min\{r_1, r_2\}$. This can also be seen by the alternating dark gray (dashed) and light gray (dotted) curves that depict the individual UAV distances to the target. In this case, the wind is blowing at 6 m/s in the positive x-direction, so it is helping the UAVs catch up to the target. The estimate of the wind is initialized at $[0 \quad 0]^{\mathsf{T}}$ but quickly converges close to the true value of the wind and stays close to the true value for the rest of the simulation, as shown in Figure 3.3.

A summary of the performance for this scenario and all three different wind speeds considered is given in Table 3.3. Note that for the various cases the wind's magnitude changed while its direction remained the same, which holds true for all target motion scenarios considered. From the table, one can see statistics associated with the cost $g(x)$, the individual 3D distances r_1 and r_2, the estimation error $\|\mathfrak{p}_g - \hat{\mathfrak{p}}_g\|$, and the actual measurement noise n_g on the ground target's position. In addition, the degree of coordination between the UAVs is indicated by the sample Pearson correlation coefficient ϱ for the UAV distance pairs r_1 and r_2, where ϱ is in general a measure of the linear correlation between two random variables and belongs to the interval $[-1, 1]$. A more negative value for ϱ indicates stronger anticorrelation, which in the present setting implies that when one UAV is relatively far from the target, the other is likely to be rather close. Overall, while the average cost is quite close for the various wind speeds, one should notice

Table 3.3 Constant velocity target.

Statistic	Wind = 0 m/s	Wind = 3 m/s	Wind = 6 m/s	Units
avg $g(x)$	132.8	126.0	130.7	m^2
var $g(x)$	3226	2608	1768	m^4
avg($\min\{r_1, r_2\}$)	62.01	62.10	61.22	m
$\max\{r_1, r_2\}$	169.3	138.0	139.7	m
avg $\|\mathfrak{p}_g - \hat{\mathfrak{p}}_g\|$	5.65	5.29	5.31	m
avg $\|n_g\|$	5.44	5.17	5.48	m
ϱ	−0.182	−0.079	−0.430	N/A

that the cost variance var $g(x)$ is quite different for each case and decreases as the wind speed increases. In addition, for the wind speed of 6 m/s, the Pearson correlation coefficient was most negative, indicating that the greatest degree of coordination was achieved by the UAVs at this speed. Moreover, the wind speed of 6 m/s in the direction of travel of the target proved to be beneficial to the UAVs' coordination efforts, as it allowed them to generally have at least one agent relatively close to the target and thereby reduce the variability in the cost.

3.4.2 Evasive Target

We now consider an evasive target, where the optimal worst-case d_k^* computed from the min–max optimization (3.11) is applied as the target's actual input. Results for the UAVs flying in 3 m/s winds to the east, that is, $w = [3 \quad 0]^\top$ with w given in (3.2), are provided in Figures 3.4–3.6.

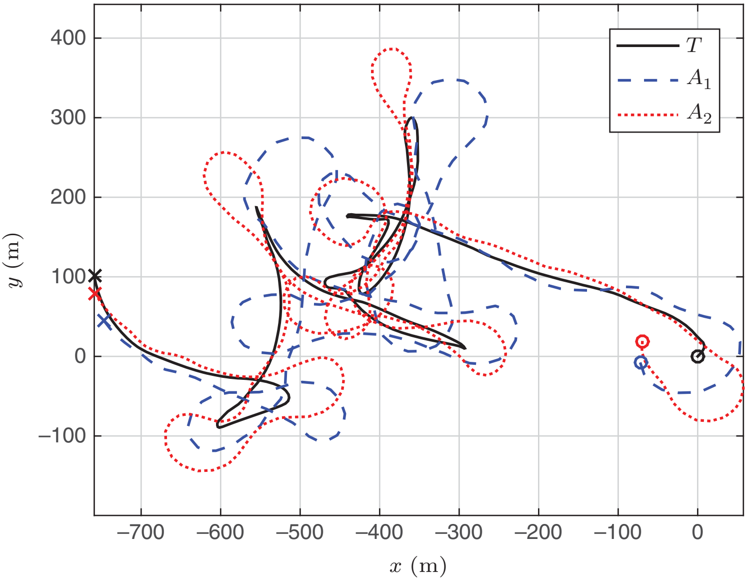


Figure 3.4 Trajectories of two UAVs, subject to 3 m/s wind, tracking an evasive target over a 5-min window. The starting positions of all vehicles are denoted by an "o" while the ending positions are indicated by an "×". In the legend, $\mathcal{T}$ corresponds to the target while $\mathcal{A}_1$ and $\mathcal{A}_2$ refer to the UAVs.

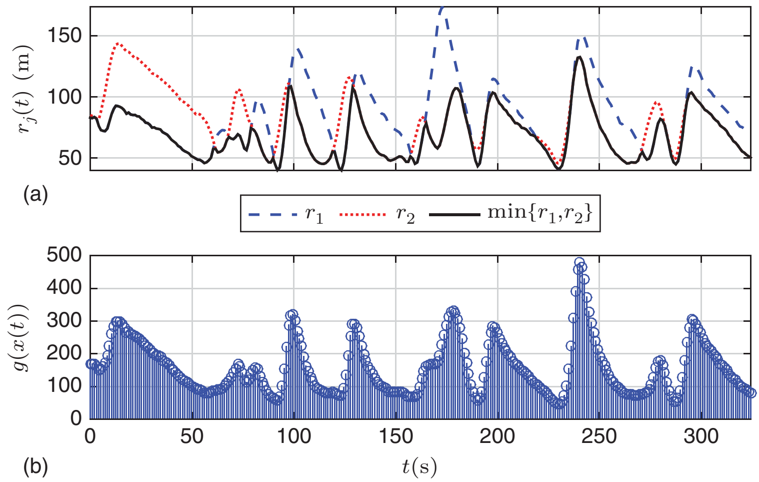


Figure 3.5 3D distances r_j (a) and stage cost $g(x)$ (b) for two UAVs, subject to 3 m/s wind, tracking an evasive target. The solid black line shows $\min\{r_1, r_2\}$.

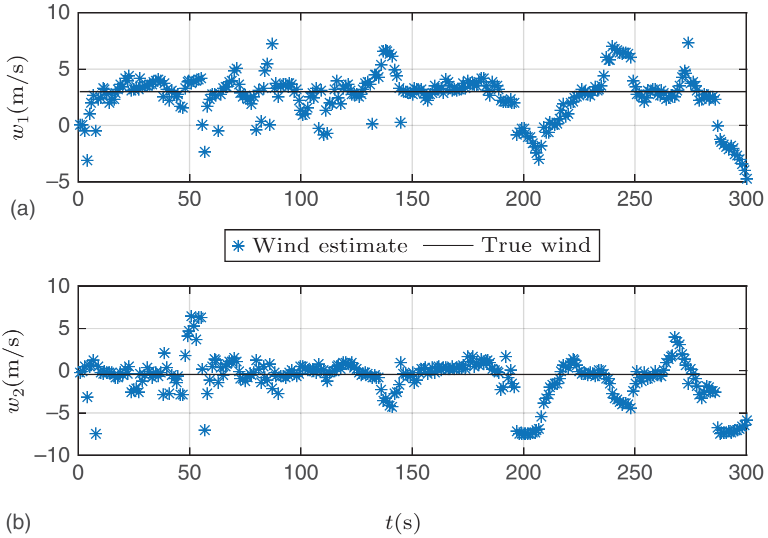


Figure 3.6 Estimate of the wind speed in the x-direction (a) and y-direction (b) for the evasive target scenario.

By observing the vehicle trajectories in Figure 3.4, one can see that the optimal trajectory for the target is quite erratic. Indeed, the target takes advantage of the UAVs' kinematic constraints by making sharp turns and forcing the UAVs to make loops at their maximum allowable turn rate. The target also makes use of the wind in this case as the target generally moves in the negative x-direction, thereby forcing the UAVs to fly against the wind.

From Figure 3.5, the minimum UAV distance from the target frequently peaks above 100 m, which means that both UAVs were often far from the target simultaneously. This is also reflected in the fact that the stage cost peaks at values much higher than in the case of a constant-velocity target. Furthermore, these peaks correspond to the periods of time just after the sharp movements in the target's trajectory shown in Figure 3.4.

A summary of the performance for this scenario and all three different wind speeds considered is given in Table 3.4. From the third column of the table, one can see that the evasive target can especially take advantage of the wind in the case of 6 m/s wind speed because the fixed airspeed of the UAVs is 15 m/s, and the maximum speed of the ground target is 10 m/s. Therefore, in this case, the evasive target can move at its maximum speed directly into the wind so that its speed of 10 m/s is faster than the effective ground speed of the UAVs flying directly into the wind (9 m/s), which renders the UAVs unable to catch the target. While the increase in all quantities considered is significant for this case, of particular interest is the fact that the measurement noise magnitude $\| n_g \|$ went up considerably, as it is related to $\min\{r_1, r_2\}$,

Table 3.4 Evasive target.

Statistic	Wind = 0 m/s	Wind = 3 m/s	Wind = 6 m/s	Units
avg $g(x)$	157.3	159.9	343.6	m^2
var $g(x)$	6466	6956	9673	m^4
avg($\min\{r_1, r_2\}$)	71.06	69.19	91.47	m
$\max\{r_1, r_2\}$	159.3	173.7	235.5	m
avg $\| \mathfrak{p}_g - \hat{\mathfrak{p}}_g \|$	5.12	4.88	6.83	m
avg $\| n_g \|$	5.86	6.14	10.4	m
ϱ	0.582	0.272	0.622	N/A

which also saw a drastic increase. Of course, the increased measurement noise in turn caused a rise in the estimation error magnitude $\| \mathfrak{p}_g - \hat{\mathfrak{p}}_g \|$, which shows the overall importance of coordinating UAV distances to the target when possible.

Overall, the target's evasive maneuvers hinder UAV coordination efforts and thereby increase measurement noise. This is especially evident by the fact that the Pearson correlation coefficient is positive for all evasive target cases, which means that the UAVs were often relatively far from the target at the same time. In addition, the cost variance was also much higher in all cases in comparison with the constant-velocity target scenario. Nonetheless, for an evasive target, the UAVs are still able to robustly track the target in the sense that their maximum 3D distance from the target is not only bounded, but is also only slightly larger than in the case of constant target velocity, provided that the UAVs' groundspeed is able to match or exceed the target's speed at all times.

3.4.3 Experimental Target Log

Finally, we now consider target motion taken from the log of a live tracking experiment at Camp Roberts, California, USA, in which a person drove a car casually yet unpredictably. Results with the UAVs subjected to 0 m/s wind are provided in Figures 3.7–3.9. One should observe from these results that the minimum 3D distance to the target is kept below 100 m, and the cost is always less than 300 m^2. Thus, even though the target does not maintain a constant velocity, the UAVs are able to coordinate and keep the cost quite low.

A summary of the performance for this scenario and all three different wind speeds considered is given in Table 3.5, where the wind direction is to the southeast in all instances of nonzero wind, that is, for $\| w \| \neq 0$, $w / \| w \| = (1/\sqrt{2})[1 \quad -1]^{\top}$. From the table, one can see that the average cost and average minimum distances are quite comparable to the constant-velocity scenario, albeit the cost variances are somewhat higher in this case. Nonetheless, the cost variances are still considerably lower than in the evasive case. One should also take notice that the correlation coefficient is always negative, meaning that the UAVs are able to coordinate in all cases considered, whereas the

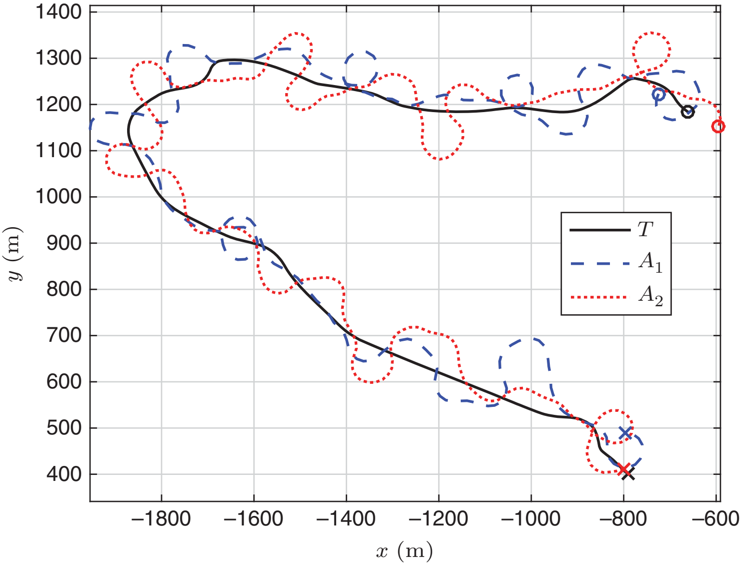


Figure 3.7 Trajectories of two UAVs, subject to 0 m/s wind, tracking a target moving based on an experimental target log over a 5-min window. The starting positions of all vehicles are denoted by a "○" while the ending positions are indicated by a "×". In the legend, $\mathcal{T}$ corresponds to the target while $\mathcal{A}_1$ and $\mathcal{A}_2$ refer to the UAVs.

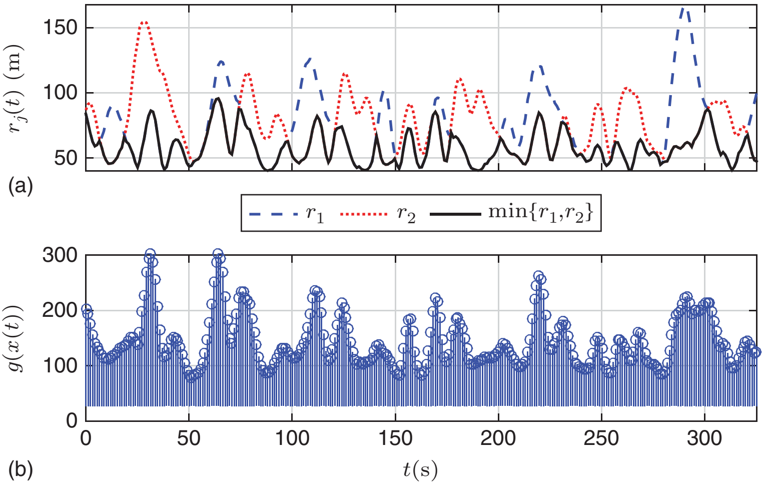


Figure 3.8 3D distances r_j (a) and stage cost $g(x)$ (b) for two UAVs tracking a target moving according to an experimental target log. The minimum distance r_j is shown as a solid black line.

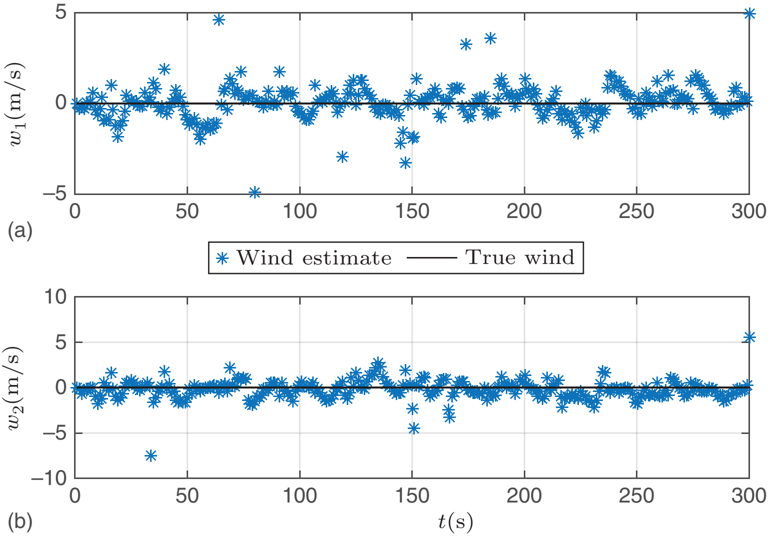


Figure 3.9 Estimate of the wind speed in the *x*-direction (a) and *y*-direction (b) for the experimental target log scenario.

Table 3.5 Experimental target log.

Statistic	**Wind = 0 m/s**	**Wind = 3 m/s**	**Wind = 6 m/s**	**Units**
avg $g(x)$	117.8	135.0	132.6	m^2
var $g(x)$	2269	4112	4406	m^4
avg(min$\{r_1, r_2\}$)	57.98	62.71	60.36	m
max$\{r_1, r_2\}$	167.4	204.8	169.6	m
avg $\|\mathfrak{p}_g - \hat{\mathfrak{p}}_g\|$	5.99	6.45	5.68	m
avg $\|n_g\|$	4.98	5.35	5.63	m
ϱ	−0.219	−0.030	−0.123	N/A

coefficients were always positive for an evasive target. Moreover, through distance coordination, the UAVs are able to keep the average measurement noise n_g on the target's position smaller than in the evasive scenario, wherein the target was able to thwart the UAVs' coordination efforts and drive the average magnitude of the state-dependent measurement noise higher.

3.5 Conclusion and Future Work

We have presented an effective, optimization-based control approach for two fixed-wing UAVs to robustly perform vision-based target tracking of a moving ground vehicle under realistic conditions. Namely, we utilized the novel approach based on min–max MPC combined with MHE that was first presented in [22] and adapted the approach to use fourth-order UAV models with roll dynamics along with the parameter estimation scheme of [27] to estimate constant winds. With simulations performed in real time using aircraft models having six degrees of freedom for the UAVs and target logs taken from live tracking experiments, we showed that the MPC/MHE approach produces a controller that is robust to unmodeled aircraft dynamics, the unmeasured environmental disturbance of wind, and information about the target state that is both noisy and incomplete.

Regarding the effectiveness of the approach for target tracking, the robust optimal controller enables the UAVs to coordinate their distances to the target when the wind speeds and target motion are not exceptionally adverse, even though the target and wind velocities are unmeasured quantities. This coordination keeps the state-dependent measurement noise at modest levels, thereby allowing the UAVs to effectively track the target. When the target is evasive, the UAVs are generally not able to coordinate their distances to the target. Nonetheless, for wind conditions in which each UAV's groundspeed always exceeds the target's speed, the UAVs are still able to track the evasive target, albeit at a somewhat reduced performance level. Finally, we note that casual, yet unpredictable, target motion from live tracking experiments with a nonholonomic ground vehicle results in performance that is considerably better than that arising from the worst-case evasive motion of a 2D double integrator, which has fewer motion constraints than a typical ground vehicle and is therefore more difficult for the underactuated UAVs to track.

Overall, we have shown that the output-feedback MPC/MHE approach is a viable approach for addressing high-dimensional, very nonlinear (nonconvex) problems involving the robust coordination of mobile robots under realistic settings. We foresee this being a powerful tool for solving nonlinear, multi-agent control problems with nonconvex objectives that require explicit robustness to both adversarial disturbances and parametric uncertainty in the dynamics, as well as noisy, partial information about the system state.

Future research interests involve addressing the problem of target tracking for a single UAV in three dimensions, as well as the possible use of airspeed control to provide an additional degree of freedom in the motion planning. In addition, we wish to investigate both the behavior and tracking performance of three UAVs with the present problem formulation in order to evaluate whether or not there is a diminishing return on investment.

References

1 Mallick, M. (2007) Geolocation using video sensor measurements, in IEEE International Conference on Information Fusion, Quebec, Canada.

2 Frew, E.W. (2007) Sensitivity of cooperative geolocation to orbit coordination, in AIAA Guidance, Navigation, and Control Conference, Hilton Head, SC, pp. 3869–3892.

3 Ma, L. and Hovakimyan, N. (2013) Cooperative target tracking in balanced circular formation: multiple UAVs tracking a ground vehicle, in American Control Conference, IEEE, pp. 5386–5391.

4 Kingston, D.B. (2007) Decentralized Control of Multiple UAVs for perimeter and target surveillance, Ph.D. thesis, Brigham Young University.

5 Peterson, C. and Paley, D.A. (2011) Multivehicle coordination in an estimated time-varying flowfield. *Journal of Guidance, Control, and Dynamics*, **34** (1), 177–191.

6 Summers, T.H., Akella, M.R., and Mears, M.J. (2009) Coordinated standoff tracking of moving targets: control laws and information architectures. *AIAA Journal of Guidance, Control, and Dynamics*, **32** (1), 56–69.

7 Kim, S., Oh, H., and Tsourdos, A. (2013) Nonlinear model predictive coordinated standoff tracking of a moving ground vehicle. *Journal of Guidance, Control, and Dynamics*, **36** (2), 557–566.

8 Sinha, A., Kirubarajan, T., and Bar-Shalom, Y. (2005) Autonomous ground target tracking by multiple cooperative UAVs, in Aerospace Conference, 2005 IEEE, 1–9, doi: 10.1109/AERO.2005.1559601.

9 Miller, S.A., Harris, Z.A., and Chong, E.K.P. (2009) A POMDP framework for coordinated guidance of autonomous UAVs for multitarget tracking. *EURASIP Journal on Advances Signal Processing*, **2009** (1), 1–17.

10 Thrun, S., Burgard, W., and Fox, D. (2005) *Probabilistic Robotics*, The MIT Press.

11 Stachura, M., Carfang, A., and Frew, E.W. (2009) Cooperative target tracking with a communication limited active sensor network, in International Workshop on Robotic Wireless Sensor Networks, Marina Del Rey, CA.

12 Ding, C., Morye, A.A., Farrell, J.A., and Roy-Chowdhury, A.K. (2012) Coordinated sensing and tracking for mobile camera platforms, in American Control Conference, IEEE, pp. 5114–5119.

13 Quintero, S.A.P., Papi, F., Klein, D.J., Chisci, L., and Hespanha, J.P. (2010) Optimal UAV coordination for target tracking using dynamic programming, in IEEE Conference on Decision and Control, Atlanta, GA.

14 Quintero, S.A.P., Ludkovski, M., and Hespanha, J.P. (2015) Stochastic optimal coordination of small UAVs for target tracking using regression-based dynamic programming. *Journal of Intelligent and Robotic Systems*, **82** (1), 135–162, doi: 10.1007/s10846-015-0270-7.

15 Triplett, B.I., Klein, D.J., and Morgansen, K.A. (2007) Distributed estimation for coordinated target tracking in a cluttered environment, in Proceedings of Robocomm, Athens, Greece.

16 Lalish, E., Morgansen, K., and Tsukamaki, T. (2007) Oscillatory control for constant-speed unicycle-type vehicles, in IEEE Conference on Decision and Control.

17 Regina, N. and Zanzi, M. (2011) UAV guidance law for ground-based target trajectory tracking and loitering, in Aerospace Conference, IEEE, doi: 10.1109/AERO.2011.5747522.

18 Copp, D.A. and Hespanha, J.P. (2014) Nonlinear output-feedback model predictive control with moving

horizon estimation, in IEEE Conference on Decision and Control, Los Angeles, CA, pp. 3511–3517.

19 Bemporad, A. and Morari, M. (1999) Robust model predictive control: a survey, in A. Garulli, A. Tesi, and A. Vicino (eds), *Robustness in Identification and Control*, Springer-Verlag, pp. 207–226.

20 Chen, H.S. and Allgöwer, F. (1997) A game theoretic approach to nonlinear robust receding horizon control of constrained systems, in American Control Conference.

21 Rawlings, J.B. and Bakshi, B.R. (2006) Particle filtering and moving horizon estimation. *Computers and Chemical Engineering*, **30** (10), 1529–1541.

22 Quintero, S.A.P., Copp, D.A., and Hespanha, J.P. (2015) Robust UAV coordination for target tracking using output-feedback model predictive control with moving horizon estimation, in American Control Conference, pp. 3758–3764.

23 Collins, G.E., Stankevitz, C.R., and Liese, J. (2011) Implementation of a sensor guided flight algorithm for target tracking by small UAS, in Ground/Air Multi-Sensor Interoperability, Integration, and Networking for Persistent ISR II, vol. **8047**, SPIE.

24 Beard, R.W. and McLain, T.W. (2012) *Small Unmanned Aircraft: Theory and Practice*, Princeton University Press.

25 Quintero, S.A.P. and Hespanha, J.P. (2014) Vision-based target tracking with a small UAV: optimization-based control strategies. *Control Engineering Practice*, **32**, 28–42. doi: 10.1016/j.conengprac.2014.07.007.

26 Saunders, J. and Beard, R.W. (2008) Tracking a target in wind using a micro air vehicle with a fixed angle camera, in American Control Conference, Seattle, WA, pp. 3863–3868.

27 Copp, D.A. and Hespanha, J.P. (2016) Control of complex systems: theory and applications, in K.G. Vamvoudakis and S. Jagannathan (eds), *Addressing Adaptation and Learning in the Context of MPC and MHE*, Elsevier, pp. 187–209.

28 Aviones (2007) Avaliable at http://aviones.sourceforge.net (accessed 20 October 2016).

29 Copp, D.A. and Hespanha, J.P. (2014) Nonlinear output-feedback model predictive control with moving horizon estimation, Tech. Rep., University California, Santa Barbara, CA, http://www.ece.ucsb.edu/~hespanha/published/mpcmhe-tech-report-20160908.pdf (accessed 03 November 2016).

4

Projection-Based Consensus for Time-Critical Coordination of Unmanned Aerial Vehicles under Velocity Constraints

Xiaofeng Wang[1]*, Eloy Garcia*[2]*, Zheqing Zhou*[1]*,*
Derek Kingston[2] *and David Casbeer*[2]

[1] *Department of Electrical Engineering, University of South Carolina, Columbia, SC, USA*
[2] *The Control Science Center of Excellence, Air Force Research Laboratory, Wright-Patterson AFB, OH, USA*

4.1 Introduction

Simultaneous arrival is one of the classic applications in cooperative control, where a group of cooperating unmanned aerial vehicles (UAVs) must reach their assigned destinations simultaneously [1, 2]. Traditional approaches estimate the expected time-to-arrival (ETA) ahead of time with the predefined profiles on the paths and the velocity. Then, each UAV simply follows its related profiles and attempts to reach its destination at the predefined arriving time [3, 4]. Since the arriving time for every UAV is the same in the profiles, they are supposed to arrive at their destinations at the same moment eventually.

In practice, however, such offline design approaches may not be suitable when flying in dynamical environments. For instant, unanticipated events may happen (e.g., a strong wind may affect the extreme speed of the UAV) and the predefined profiles may no longer be valid. Sticking to the original profiles will only steer the overall system away from its normal operating point and eventually lead to mission failure. Instead, a real-time planning scheme is more desirable. Each UAV estimates its own ETA based on their position and speed telemetry and communicates it with its neighbors in real time through the network so that

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

they can reach consensus on the ETA during the flight before any UAV arrives at its destinations [5]. By doing so, even if disturbances exist, the UAVs can still adjust their agreement on the ETA based on the current situation.

One thing worth pointing out is that most of such real-time approaches rely on consensus algorithms where a group of agents attempt to reach an agreement on a common value in a distributed way by negotiating with their neighbors [6, 7]. In consensus-based simultaneous arrival, convergence rate plays a very important role because consensus on the ETA must be reached before any UAV arrives at its destination. Without velocity constraints, it is well-known that convergence rate of the consensus algorithm is related to the minimal positive eigenvalue of the Laplacian of a connected graph [8–10]. When velocity constraints exist, however, it becomes very challenging to analyze stability and the convergence rate of consensus algorithms.

There has been some work on constrained consensus. For instance, system dynamics were considered in [11, 12] and communication constraints were discussed in [13]. A more detailed survey on constrained consensus can be found in [14]. Speaking of state and input constraints, the work in [15] and [16] addressed input saturation for continuous-time and discrete-time multi-agent systems, respectively. The convergence rate, however, was not provided. The work in [17], based on set projection, presented discrete-time consensus algorithms for systems with state constraints. Convergence rate was derived for some specific cases. The work in [18] extended the results in [17] to the case where communication delay exists. Continuous-time consensus with state constraints was studied in [19] where logarithmic barrier functions were introduced to define the repulsive and attractive components in the consensus protocol. This approach requires an agent to transmit the auxiliary variable, in addition to its state, to its neighbors. Moreover, it requires that the origin must be included in all the constraint sets. Projection-based consensus with state constraints was studied in our recent work [20]. A special type of input saturation for continuous-time consensus was presented in [21], where the control input can only take the boundary value or 0, following the bang–bang control law.

This chapter studies real-time coordination for simultaneous arrival with fixed-velocity constraints, which is formulated as a constrained continuous-time consensus problem. We introduce a continuous-time projection operator [22–24] that ensures smoothness of the state and input trajectories, when saturation happens due to the velocity constraints. Based on this projection operator, we present a continuous-time consensus protocol for UAVs to coordinate their ETA and velocity. We show the convergence of the resulting closed-loop system and derive bounds on the convergence rate. Based on these results, we provide a sufficient condition that guarantees the feasibility of the simultaneous arrival problem, in terms of the length of the paths as well as the UAVs' minimal and maximal velocities. Simulations verify the theoretical findings.

This chapter is organized as follows. Section 4.2 formulates the problem. The projection-based consensus algorithm is introduced in Section 4.3. The analysis of convergence, convergence time, and problem feasibility can be found in Sections 4.4–4.6, respectively. Simulation results are presented in Section 4.7. Section 4.8 summarizes the results.

4.2 Problem Statement

4.2.1 Notations

We denote by $\mathbb{R}^n$ the n-dimensional real vector space, by $\mathbb{R}^+$ the set of the positive real numbers, and by $\mathbb{R}_0^+$ the set of the non-negative real numbers. We use $\|\cdot\|$ to denote the Euclidean norm of a vector and the induced 2-norm of a matrix. The maximal and minimal singular values of a matrix P are denoted by $\sigma_{\max}(P)$ and $\sigma_{\min}(P)$, respectively. Given a collection of n scalars $\tau_1, \tau_2, \dots, \tau_n$, let $\tau = (\tau_1, \dots, \tau_n)^\top$. Let $\mathbf{1}_n = (1, \dots, 1)^\top \in \mathbb{R}^n$. For a function of time $x(t)$, sometimes we drop t and use just x for brevity if it is clear in context.

For a team of n agents, the communication among them is described by an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ in this chapter, where $\mathcal{V} = \{1, 2, \dots, n\}$ denotes the agent set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the edge set. An edge $(i,j) \in \mathcal{E}$ means that agent j can obtain information from agent i. Since the undirected graph is considered in this chapter, $(i,j) \in \mathcal{E}$ implies $(j,i) \in \mathcal{E}$, which means that agent i can also obtain information from agent j. If $(i,j) \in \mathcal{E}$, then agent i is called a neighbor of agent j. The set $\mathcal{V}_i \in \mathcal{V}$ is called the neighboring set of agent i and $|\mathcal{V}_i|$ is its cardinality. A path from agent i to agent j is a sequence $i, p_1, p_2, \ldots, p_k, j$, where $p_l \in \mathcal{V}$ for $l = 1, 2, \ldots, k$ and $(i, p_1), (p_k, j), (p_l, p_{l+1}) \in \mathcal{E}$ for $l = 1, 2, \ldots, k-1$. A graph is called connected if for any $i, j \in \mathcal{V}$, there exists at least a path from i to j.

The adjacency matrix $A \in \mathbb{R}^{n\times n}$ of a graph $\mathcal{G}$ is defined by $a_{ij} = 1$ if $(i,j) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. Given an undirected graph, $a_{ij} = a_{ji}$, which means A is symmetric. The Laplacian matrix L of $\mathcal{G}$ is defined as $L = D - A$, where D represents the degree matrix which is a diagonal matrix with entries $d_{ii} = |\mathcal{V}_i| = \sum_{j\in\mathcal{V}_i} a_{ij}$. Notice that $L\mathbf{1}_n = 0$. If the graph is connected, the corresponding Laplacian matrix has only one eigenvalue equal to 0, $\lambda_1(L) = 0$, and the other eigenvalues are positive and real, $\lambda_i(L) > 0$ for $i = 2, \ldots, n$.

4.2.2 Problem Formulation

This section provides a rigorous formulation of the simultaneous arrival problem for multi-agent systems. Suppose that a team of n agents are tasked to simultaneously visit some prespecified targets and the path for each individual agent to follow has been precomputed. To arrive at their targets at the same time, agents have to adjust their velocity during the motion, based on the information communicated with their neighbors.

Let $l_i(t)$ and $v_i(t)$ denote the length of the remaining path and the velocity of agent i at time t, respectively, where $l_i : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ and $v_i : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ for $i = 1, \ldots, n$. Obviously, the following relation holds:

$$l_i(t) = l_i(0) - \int_0^t v(s)\mathrm{d}s,$$

which implies

$$\dot{l}_i(t) = -v_i(t), \tag{4.1}$$

with the initial condition

$$l_i(0) = l_{i,0} \tag{4.2}$$

$$v_i(0) = v_{i,0}. \tag{4.3}$$

The velocity of agent i must satisfy the following constraint:

$$v_i(t) \in [v_i^{\min}, v_i^{\max}],$$

where $v_i^{\min}, v_i^{\max} \in \mathbb{R}_0^+$ are known and $v_i^{\min} < v_i^{\max}$. Notice that $[v_i^{\min}, v_i^{\max}]$ does not have to include 0 and $v_{i,0} \in [v_i^{\min}, v_i^{\max}]$.

At time t, the ETA of agent i is denoted by $\tau_i(t)$, which satisfies

$$\tau_i(t) = \frac{l_i(t)}{v_i(t)}. \tag{4.4}$$

Obviously, when agent i arrives its destination at some time, say t_i^*, then $\tau(t_i^*) = 0$ and $l_i(t_i^*) = 0$.

Taking the time derivative of $\tau_i(t)$, we have the following differential equation:

$$\dot{\tau}_i = \frac{v_i\dot{l}_i - l_i\dot{v}_i}{v_i^2} = -1 - \frac{\tau_i}{v_i}\dot{v}_i. \tag{4.5}$$

Let

$$\dot{v}_i(t) = u_i(t), \tag{4.6}$$

where $u_i(t)$ is the input that controls the velocity of agent i.

Within this framework, we are interested in consensus protocols under a connected and undirected communication graph that can guarantee simultaneous arrival subject to the velocity constraints.

4.3 Projection-Based Consensus Algorithm

To arrive simultaneously, agents have to achieve an agreement on the ETA that is feasible to all of them before any agents physically arrive at their destinations. Ideally, if there were no constraints on the velocity, the consensus protocol can be easily designed as

$$u_i = -\frac{c\, v_i}{\tau_i} \sum_{j \in \mathcal{V}_i} (\tau_j - \tau_i),$$

where $\mathcal{V}_i \subseteq \mathcal{V}$ is the neighboring set of agent i and c is a positive constant controlling the convergence rate. Assume that the communication graph is undirected, that is, if $j \in \mathcal{V}_i$, then $i \in \mathcal{V}_j$. One can easily verify that this consensus protocol will guarantee $\tau_i(t) - \tau_j(t) \to 0$ as $t \to \infty$ [5].

When constraints are posed on the velocity, saturation will happen when the velocity approaches the boundary value. To ensure smoothness of the velocity over the interval $[v_i^{\min}, v_i^{\max}]$ while still taking saturation into account, we introduce the following projection-based consensus protocol:

$$u_i(t) = \text{Proj}_{[v_i^{\min}, v_i^{\max}]}\left(v_i, -\frac{c\, v_i}{\tau_i} \sum_{j \in \mathcal{V}_i} (\tau_j - \tau_i) \right). \tag{4.7}$$

The projection-based operator Proj : $\mathbb{R} \times \mathbb{R} \times \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is defined as follows:

$$\text{Proj}_{[v_i^{\min}, v_i^{\max}]}(v_i, a) = \begin{cases} a & \text{if } f_i(v_i) \leq 0 \\ a & \text{if } f_i(v_i) > 0 \text{ and } \dfrac{\partial f_i(v_i)}{\partial v_i} a \leq 0 \\ a(1 - f_i(v_i)) & \text{if } f_i(v_i) > 0 \text{ and } \dfrac{\partial f_i(v_i)}{\partial v_i} a > 0, \end{cases} \tag{4.8}$$

where

$$f_i(v_i) = \frac{\left(v_i - \frac{v_i^{\min} + v_i^{\max}}{2} \right)^2 - (\rho_i^{\max})^2}{\epsilon (\rho_i^{\max})^2} \tag{4.9}$$

$$\rho_i^{\max} = \frac{v_i^{\max} - v_i^{\min}}{2\sqrt{1 + \epsilon}}, \tag{4.10}$$

and $\epsilon \in (0, 1)$. Letting

$$\underline{v}_i = -\rho_i^{\max} + \frac{v_i^{\max} + v_i^{\min}}{2}$$

$$\overline{v}_i = \rho_i^{\max} + \frac{v_i^{\max} + v_i^{\min}}{2},$$

we can easily verify that

- $[\underline{v}_i, \overline{v}_i] \subset [v_i^{\min}, v_i^{\max}]$;
- $f_i(v_i) \leq 0$ when $v_i \in [\underline{v}_i, \overline{v}_i]$;

- $0 < f_i(v_i) < 1$ when $v_i \in (v_i^{\min}, \underline{v}_i) \cup (\overline{v}_i, v_i^{\max})$;
- $f_i(v_i^{\min}) = f_i(v_i^{\max}) = 1$.

Notice that $[\underline{v}_i, \overline{v}_i]$ can be arbitrarily close to $[v_i^{\min}, v_i^{\max}]$ if ϵ is close enough to 0. Obviously, at time t, if $\tau_i(t) \in \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right]$, agent i can arrive at its destination in $\tau_i(t)$ unit-of-time by maintaining the velocity at time t (which is $\frac{l_i(t)}{\tau_i(t)} \in [\underline{v}_i, \overline{v}_i] \subset [v_i^{\min}, v_i^{\max}]$). So $\tau_i(t)$ is a feasible ETA for agent i at time t and we call the set $\left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right]$ the feasible set of the ETA of agent i at time t.

The projection-based operator is often used in adaptive control [25]. It can enforce the estimated uncertainty always stay inside a prespecified region with the guarantee of differentiability. In our case, we use it to enforce the constraints on the velocity. Notice that if $v_{i,0} \in [v_i^{\min}, v_i^{\max}]$, the inequality $v_i(t) \in [v_i^{\min}, v_i^{\max}]$ always holds for any $t \geq 0$ by the definition of the projection-based operator [25]. Also, it generates a smooth input $u_i(t)$, which is more suitable for UAV applications since the smooth input profile can be more easily implemented than an input with discontinuities. As a result, not only input saturation is addressed, but, to some extent, saturation on the input rate of change is dealt with as well.

4.4 Convergence Analysis

This section discusses the convergence of the proposed consensus protocol. Let T^* be the time when the first agent arrives at its destination. Notice that before T^*, $\tau_i(t) > 0$ and $l_i(t) > 0$ for any $i \in \mathcal{V}$. Before introducing the result on convergence, we need the following lemma.

Lemma 4.4.1 For any $t \in [0, T^*]$, if

$$\cap_{i=1}^{n} \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right] \neq \emptyset \tag{4.11}$$

and there exist $i, j \in \mathcal{V}$ such that $v_i(t) \in [v_i^{\min}, \underline{v}_i]$ and $v_j(t) \in [\overline{v}_j, v_j^{\max}]$, then the inequality $\tau_i(t) \geq \tau_j(t)$ must hold.

Proof: We prove this statement using contradiction method. Suppose that $\tau_i(t) < \tau_j(t)$ is true. Then

$$\frac{l_i(t)}{\underline{v}_i} \leq \frac{l_i(t)}{v_i(t)} = \tau_i(t) < \tau_j(t) = \frac{l_j(t)}{v_j(t)} \leq \frac{l_j(t)}{\overline{v}_j},$$

which means that $\left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right] \cap \left[\frac{l_j(t)}{\overline{v}_j}, \frac{l_j(t)}{v_{j}}\right] = \emptyset$ at time t. This is contradicted with the assumption that $\cap_{i=1}^{n} \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right] \neq \emptyset$. Therefore, $\tau_i(t) \geq \tau_j(t)$ must hold. ■

Lemma 4.4.1 indicates that if the intersection of the feasible sets of the ETAs is not empty, the agents whose velocity is close to the minimum ($v_i(t) \in [v_i^{\min}, \underline{v}_i]$) will have greater ETAs than those agents with almost maximal velocity ($v_i(t) \in [\overline{v}_i, v_i^{\max}]$). This lemma will be often used in the following proofs, with which we are able to present the first result in this chapter.

Theorem 4.4.2 If (4.11) holds for all $t \in [0, T^*]$ using the consensus protocol in equation (4.7), then $\tau_i(t) - \tau_j(t) \to 0$ for any $i, j \in \mathcal{V}$ as $T^* \to \infty$ and $t \to \infty$.

Proof: In the following discussion, we drop the index t if it is clear in context. Let $z_i(t) = \sum_{j \in \mathcal{V}_i} (\tau_j(t) - \tau_i(t))$. Then $z(t) = -L\tau(t)$, where L is the Laplacian of the graph. Consider the potential function $V(\tau) = \frac{1}{2}\tau^\top L \tau$.

$$\dot{V} = \tau^\top L \dot{\tau} = \tau^\top L \begin{pmatrix} -1 - \dfrac{\tau_1}{v_1}\dot{v}_1 \\ \vdots \\ -1 - \dfrac{\tau_n}{v_n}\dot{v}_n \end{pmatrix}.$$

Since $L\mathbf{1}_n = 0$, the preceding equation can be further derived as

$$\dot{V} = \tau^\top L \begin{pmatrix} -\dfrac{\tau_1}{v_1}\dot{v}_1 \\ \vdots \\ -\dfrac{\tau_n}{v_n}\dot{v}_n \end{pmatrix} = (-z_1, \ldots, -z_n) \begin{pmatrix} -\dfrac{\tau_1}{v_1}\dot{v}_1 \\ \vdots \\ -\dfrac{\tau_n}{v_n}\dot{v}_n \end{pmatrix}$$

$$= \sum_{i=1}^{n} z_i \cdot \frac{\tau_i}{v_i} \cdot \mathrm{Proj}_{[v_i^{\min}, v_i^{\max}]} \left(v_i, -\frac{c v_i}{\tau_i} z_i \right).$$

Let

$$\mathcal{S}(t) = \left\{ i \in \mathcal{V} \mid f_i(v_i) > 0 \text{ and } \frac{\partial f_i(v_i)}{\partial v_i} \left(-\frac{c v_i}{\tau_i} z_i \right) > 0 \right\},$$

where f_i is defined in (4.9). Then

$$\dot{V} = \sum_{i \in \mathcal{S}(t)} -z_i^2 c(1 - f_i(v_i)) - \sum_{i \in \mathcal{V}/\mathcal{S}(t)} c z_i^2.$$

This inequality implies that $\tau(t)$ will converge to the equilibrium $\tau^* = (\tau_1^*, \dots, \tau_n^*)$, where agent i satisfies either

$$\begin{aligned} & v_i^* \in \{v_i^{\min}, v_i^{\max}\} && (\Leftrightarrow f_i(v_i^*) = 1), \quad \text{or} \\ & \tau_i^* = \frac{1}{|\mathcal{V}_i|} \sum_{j \in \mathcal{V}_i} \tau_j^* && (\Leftrightarrow z_i^* = 0), \end{aligned} \tag{4.12}$$

and $v_i^* \in \mathbb{R}^+$ is the corresponding velocity of agent i.

We now show that $\tau_i^* = \frac{1}{|\mathcal{V}_i|} \sum_{j \in \mathcal{V}_i} \tau_j^*$ for all $i \in \mathcal{V}$ using contradiction method. Suppose that the set $\Lambda = \{i \in \mathcal{V} \mid \tau_i^* - \frac{1}{|\mathcal{V}_i|} \sum_{j \in \mathcal{V}_i} \tau_j^* \neq 0\}$ is not empty. According to the aforementioned analysis, we have $v_i^* \in \{v_i^{\min}, v_i^{\max}\}$ for any $i \in \Lambda$. Without the loss of generality, assume that the set $\Lambda_m = \{i \in \Lambda \mid v_i^* = v_i^{\min}\}$ has at least one element. Let $i_0 = \arg\max_{i \in \Lambda_m} \tau_i^*$. Then the inequality

$$\tau_{i_0}^* < \frac{1}{|\mathcal{V}_{i_0}|} \sum_{j \in \mathcal{V}_{i_0}} \tau_j^* \tag{4.13}$$

must hold; otherwise, if $\tau_{i_0}^* > \frac{1}{|\mathcal{V}_{i_0}|} \sum_{j \in \mathcal{V}_{i_0}} \tau_j^*$, then $-\frac{v_{i_0}^*}{\tau_{i_0}^*} \sum_{j \in \mathcal{V}_{i_0}} (\tau_j^* - \tau_{i_0}^*) > 0$, given $\tau_{i_0}(t) > 0$ over $[0, T^*)$. Since $v_{i_0}^* = v_{i_0}^{\min}$, we have

$$f_{i_0}(v_{i_0}^*) = 1 > 0$$

$$\left. \frac{\partial f_{i_0}(v_{i_0})}{\partial v_{i_0}} \right|_{v_{i_0} = v_{i_0}^{\min}} = \frac{v_{i_0}^{\min} - v_{i_0}^{\max}}{\epsilon (\rho_{i_0}^{\max})^2} < 0.$$

Therefore,

$$\left. \frac{\partial f_{i_0}(v_{i_0})}{\partial v_{i_0}} \right|_{v_{i_0} = v_{i_0}^{\min}} \left(-\frac{c v_{i_0}^*}{\tau_{i_0}^*} \sum_{j \in \mathcal{V}_{i_0}} (\tau_j^* - \tau_{i_0}^*) \right) < 0.$$

By the definition of the projection-based operator in (4.8), we know

$$\dot{v}_{i_0} = \mathrm{Proj}_{[v_{i_0}^{\min}, v_{i_0}^{\max}]}\left(v_{i_0}, -\frac{cv_{i_0}^*}{\tau_{i_0}^*} \sum_{j \in \mathcal{V}_{i_0}} (\tau_j^* - \tau_{i_0}^*) \right)$$
$$= -\frac{cv_{i_0}^*}{\tau_{i_0}^*} \sum_{j \in \mathcal{V}_{i_0}} (\tau_j^* - \tau_{i_0}^*) > 0.$$

It implies that $v_{i_0}(t)$ will move away from $v_{i_0}^{\min}$, toward the center of $[v_{i_0}^{\min}, v_{i_0}^{\max}]$, which indicates that $v_{i_0}^* = v_{i_0}^{\min}$ is no longer the equilibrium.

With inequality (4.13), let $i_1 \in \mathcal{V}_{i_0}$ be the agent satisfying

$$i_1 = \arg\max_{j \in \mathcal{V}_{i_0}} \tau_j^*. \tag{4.14}$$

Obviously, $\tau_{i_0}^* < \tau_{i_1}^*$ must hold. By the contrapositive statement of Lemma 4.4.1, we know that $v_{i_1}^* \notin (\overline{v}_{i_1}, v_{i_1}^{\max}]$. Meanwhile, by the definition of i_0, $v_{i_1}^* \neq v_{i_1}^{\min}$. Therefore, $v_{i_1}^* \notin \{v_{i_1}^{\min}, v_{i_1}^{\max}\}$ and we have

$$\tau_{i_1}^* = \frac{1}{|\mathcal{V}_{i_1}|} \sum_{j \in \mathcal{V}_{i_1}} \tau_j^* \tag{4.15}$$

according to (4.12). Since we consider an undirected graph, $i_0 \in \mathcal{V}_{i_1}$. With $\tau_{i_0}^* < \tau_{i_1}^*$, we know that there must exist $i_2 \in \mathcal{V}_{i_1}$ such that

$$\tau_{i_1}^* < \tau_{i_2}^*$$

according to (4.15). Keeping such reasoning, since the graph is connected, we can sort all τ_i^* greater than $\tau_{i_0}^*$, starting from $\tau_{i_0}^*$ and ending at $\max_{i \in \mathcal{V}} \tau_i^*$. Then there always exists $i_k \in \mathcal{V}_{i_{k-1}}$ that satisfies

$$\tau_{i_0}^* < \cdots < \tau_{i_{k-1}}^* < \tau_{i_k}^* \tag{4.16}$$
$$\tau_{i_k}^* \geq \tau_j^*, \forall j \in \mathcal{V}_{i_k}$$
$$\tau_{i_{k-1}}^* \in \mathcal{V}_{i_k}.$$

Obviously,

$$\tau_{i_k}^* > \frac{1}{|\mathcal{V}_{i_k}|} \sum_{j \in \mathcal{V}_{i_k}} \tau_j^*,$$

which means that $v^*_{i_k} \in \{v^{\min}_{i_k}, v^{\max}_{i_k}\}$ according to (4.12). By the definition of i_0, we know that $v^*_{i_k} = v^{\max}_{i_k}$. However, by Lemma 4.4.1, if $v_{i_0}(t) \in [v^{\min}_{i_0}, v_{i_0})$ and $v_{i_k}(t) \in (\overline{v}_{i_k}, v^{\max}_{i_k}]$, $\tau^*_{i_0} \geq \tau^*_{i_k}$ must hold, which is contradicted with inequality (4.16). Similar analysis applies to the case when the set $\{i \in \Lambda \mid v^*_i = v^{\max}_i\}$ has at least one element. Therefore, we conclude that $\tau^*_i = \frac{1}{|\mathcal{V}_i|}\sum_{j\in\mathcal{V}_i}\tau^*_j$ for all $i \in \mathcal{V}$, which means $L\tau^* = 0$. It implies $\tau^*_i = \tau^*_j$ for any $i, j \in \mathcal{V}$. ■

Remark 4.4.3 Equation (4.11) means that the ETA feasible sets of agents at least have one shared ETA that the agents can agree with. A more general condition is that there exists $\hat{t} \in [0, T^*)$ such that (4.11) holds over $[\hat{t}, T^*]$, which implies that (4.11) ultimately holds. However, we would like to point out that this more general condition will not significantly change the analysis in this chapter since we can always treat $\hat{t}$ as the initial time. Another important thing worth mentioning is that the assumption of (4.11) can be relaxed in the later discussion where verifiable conditions are presented to enforce the satisfaction of (4.11).

Theorem 4.4.2 shows asymptotic property of the consensus algorithm. Notice that it will take infinite amount of time for agents to converge to the exact equilibrium. We know that to have simultaneous arrival, consensus on the ETA must be achieved before any actual arrival. Since the algorithm needs infinite amount of time to reach the equilibrium, the actual arrival time theoretically should also be infinity so that it is always later than the moment when consensus is achieved. That is why we introduce the condition $T^* \to \infty$, which is to guarantee the well-poseness of the algorithm, that is, to make sure that the remaining path $l_i(t)$ and the ETA $\tau_i(t)$ are always greater than 0 during the converging process. Otherwise, if $l_i(t) < 0$ for example, it means that agent i has passed its destination, which does not make sense in simultaneous arrival. In fact, $T^* \to \infty$ implies that the path of each agent, $l_i(t)$, is arbitrarily long.

In practice, however, the length of the paths is always finite. It requires agents to achieve consensus in finite amount of time. Instead of reaching the exact equilibrium, a more practical way is to ensure that $\tau_i(t) - \tau_j(t)$ enters and stays in a small neighborhood of the origin before any actual arrival happens. Once $\tau_i(t) - \tau_j(t)$ enters such sets, we claim that consensus is achieved. Such an approach can be referred to ε-consensus [26]. In the following sections, we further explore this idea with discussions on the convergence time of the consensus algorithm and the feasibility of the simultaneous arrival problem.

4.5 Convergence Time

In this section, we develop bounds on the convergence time when ε-consensus can be achieved. Before presenting the main results, we first introduce the following definition of ε-consensus.

Definition 4.5.1 A protocol makes the agents reach ε-consensus in finite time if there exist a finite time $T > 0$ and a small positive constant ε such that the state $\tau(t) \in \Phi$ for all $t \geq T$, where

$$\Phi \triangleq \{\tau \in \mathbb{R}^n \mid V(\tau) \leq \varepsilon\} \text{ and} \tag{4.17}$$

$$V(\tau) = \tau^{\mathsf{T}} L \tau. \tag{4.18}$$

Notice that $V(\tau) = \sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V}_i}(\tau_j - \tau_i)^2$. If $\tau(t)$ stays in the set Φ, $|\tau_j(t) - \tau_i(t)|$ will be bounded by a very small number. In that case, we say that consensus on the ETA is reached. One important thing for the simultaneous arrival problem is that the time when all $\tau_i(t)$ enters the set Φ, denoted as T^*, should be earlier than any actual arrival happens. To fulfill this objective, we need to study when $\tau(t)$ will enter the set Φ on the one hand. On the other hand, we need to estimate when agents will arrive at their destinations.

To address the question of when $\tau(t)$ enters Φ, we study the convergence rate of the consensus protocol and derive an upper bound on the actual entering time T^*. We start from the following lemma, which provides the convergence rate of V over a short time interval.

Lemma 4.5.2 Given $t_0 \in [0, T^*]$ and $\delta_0 \in \mathbb{R}^+$, assume that (4.11) holds for any $t \in [t_0, T^*]$ and $\| V(\tau(t_0)) \| \leq \delta_0$. Given any positive constant $\delta_1 < \delta_0$, $\tau_i(t)$ will enter the set $\{\tau \mid V(\tau) \leq \delta_1\}$ and $t_1 - t_0 \leq \frac{\delta_0 - \delta_1}{c\,\xi^2}$, where $t_1 > t_0$ is the entering time instant,

$$\xi = \frac{\sqrt{\delta_1}}{d\sqrt{n\sigma_{\max}(L^{\dagger})}}, \tag{4.19}$$

$$d = \max_{i_0 i_1 \cdots i_k \in \mathcal{P}} |\mathcal{V}_{i_0}| \left(1 + \sum_{m=1}^{k-1} \prod_{j=1}^{m} (|\mathcal{V}_{i_j}| - 1) \right), \tag{4.20}$$

where $L^{\dagger}$ is the pseudoinverse matrix of L satisfying $L = LL^{\dagger}L$, and $\mathcal{P}$ denotes the set of all paths in the graph.

Proof: Let $z_i(t) = \sum_{j \in \mathcal{V}_i} (\tau_j(t) - \tau_i(t))$. Consider the time derivative of $V(\tau) = \tau^{\top} L \tau$:

$$\dot{V} \leq - \sum_{i \in \mathcal{V}} b_i(t) z_i(t)^2, \tag{4.21}$$

where $b_i(t) = c(1 - f_i(v_i(t)))$ if

$$i \in \mathcal{S}(t) = \left\{ i \in \mathcal{V} \mid f_i(v_i(t)) > 0 \text{ and } \frac{\partial f_i(v_i)}{\partial v_i} \left(-\frac{c v_i(t)}{\tau_i(t)} z_i(t) \right) > 0 \right\};$$

otherwise $b_i(t) = c$.

Let $i_M = \arg\max_i \tau_i$ and $i_m = \arg\min_i \tau_i$. There are two possible cases: (i) $b_i \neq c$ for both $i = i_m$ and $i = i_M$; (ii) $b_i = c$ for some $i \in \{i_m, i_M\}$.

Case I: We show that Case I does not hold, using contradiction method. Suppose that $b_{i_m} \neq c$ and $b_{i_M} \neq c$. Then $i_m, i_M \in \mathcal{S}(t)$. First, consider i_m. The relation $i_m \in \mathcal{S}(t)$ means that $v_{i_m} \in (\overline{v}_{i_m}, v_{i_m}^{\max}] \cup [v_{i_m}^{\min}, \underline{v}_{i_m})$ and

$$\frac{\partial f_{i_m}(v_{i_m})}{\partial v_{i_m}} \left(-\frac{c v_{i_m}}{\tau_{i_m}} z_{i_m} \right) > 0. \tag{4.22}$$

By the definition of i_m, we have

$$\frac{1}{|\mathcal{V}_{i_m}|} \sum_{j \in \mathcal{V}_{i_m}} \tau_j > \tau_{i_m},$$

which means $z_{i_m} > 0$. Applying this inequality into inequality (4.22) yields $\frac{\partial f_{i_m}(v_{i_m})}{\partial v_{i_m}} < 0$ and therefore $v_{i_m} \in [v_{i_m}^{\min}, v_{\,i_m})$ by the definition of f_{i_m} in (4.9). Similarly, we have $v_{i_M} \in (\overline{v}_{i_M}, v_{i_M}^{\max}]$. By Lemma 4.4.1, we know that $\tau_{i_m} \geq \tau_{i_M}$ must hold, which is contradicted with the fact $\tau_{i_m} < \tau_{i_M}$ according to the definitions of τ_{i_m} and τ_{i_M}. Therefore, either $b_{i_m} = c$ or $b_{i_M} = c$ holds, or both.

Case II: Obviously, when $\tau(t)$ enters the set $\{\tau \mid V(\tau) \leq \delta_1\}$, it will stay inside the set according to inequality (4.21). Let us consider $\dot{V}$ when $V(\tau(t)) > \delta_1$. Define $\Lambda_t = \{i \in \mathcal{V} \mid b_i(t) = c\}$. By the aforementioned analysis, Λ_t is always nonempty over $[0, T^*]$.

When $V(\tau(t)) \geq \delta_1$, we have

$$\sigma_{\max}(L^{\dagger}) z^{\top} z \geq z^{\top} L^{\dagger} z = \tau^{\top} L \tau = V(\tau) \geq \delta_1,$$

where $L^{\dagger}$ is the pseudoinverse matrix of L. Therefore, $\| z(t) \| \geq \sqrt{\frac{\delta_1}{\sigma_{\max}(L^{\dagger})}}$, which means that there exists at least an $i \in \mathcal{V}$ such that

$$|z_i(t)| \geq \eta \triangleq \sqrt{\frac{\delta_1}{n\sigma_{\max}(L^{\dagger})}}.$$

If $i \in \Lambda_t$, we know $\dot{V} \leq -c\eta^2$ by inequality (4.21).

Consider the case when $|z_i| < \eta$ for any $i \in \Lambda_t$. Then any i satisfying $|z_i| \geq \eta$ belongs to $\mathcal{V}/\Lambda_t$. Without the loss of generality, assume that at time t there exists at least an $i \in \mathcal{V}/\Lambda_t$ such that $v_i \in (\overline{v}_i, v_i^{\max}]$ and $|z_i| \geq \eta$. Let

$$\begin{aligned}
&\Theta_t = \{i \in \mathcal{V}/\Lambda_t \mid v_i(t) \in (\overline{v}_i, v_i^{\max}]\text{and } |z_i(t)| \geq \eta\}, \\
&i_0 = \arg\min_{i \in \Theta_t} \tau_i, \\
&\Omega = \{j \in \mathcal{V} \mid \tau_j < \tau_{i_0}\}.
\end{aligned}$$

Notice that any $i \in \mathcal{V}$ satisfying $v_i \in [v_i^{\min}, \underline{v}_i)$ and $b_i \neq c$ does not belong to Ω according to Lemma 4.4.1. Therefore, for any $j \in \Omega$ we know either $b_j = 1$ or $v_j \in (\overline{v}_j, v_j^{\max}] \wedge b_j \neq 1$, besides $|z_j| < \eta$.

Since $i_0 \notin \Lambda_t$, which means $b_{i_0} \neq c$, we have $\frac{\partial f(v_{i_0})}{\partial v_{i_0}} \left(-\frac{cv_{i_0}}{\tau_{i_0}} z_{i_0} \right) > 0$. Notice that $v_{i_0} \in (\overline{v}_{i_0}, v_{i_0}^{\max}]$ implies $\frac{\partial f_i(v_{i_0})}{\partial v_{i_0}} > 0$. Then $-\frac{cv_{i_0}}{\tau_{i_0}} z_{i_0} > 0$, which means $z_{i_0} = \sum_{j \in \mathcal{V}_{i_0}} (\tau_j - \tau_{i_0}) < -\eta < 0$. There exists at least $i_1 \in \Omega \cap \mathcal{V}_{i_0}$ such that $\tau_{i_1} - \tau_{i_0} < -\frac{\eta}{|\mathcal{V}_{i_0}|}$.

If $b_{i_1} \neq c$, which means $v_{i_1} \in (\overline{v}_{i_1}, v_{i_1}^{\max}]$, we have $\sum_{j \in \mathcal{V}_{i_1}} (\tau_j - \tau_{i_1}) < 0$ following the same analysis for i_0. Then

$$\sum_{j \in \mathcal{V}_{i_1}/\{i_0\}} (\tau_j - \tau_{i_1}) < \tau_{i_1} - \tau_{i_0} < -\frac{\eta}{|\mathcal{V}_{i_0}|}. \tag{4.23}$$

Therefore, there exists at least $i_2 \in \Omega \cap \mathcal{V}_{i_1}/\{i_0\}$ such that $\tau_{i_2} - \tau_{i_1} < -\frac{\eta}{|\mathcal{V}_{i_0}|(|\mathcal{V}_{i_1}|-1)}$. Otherwise, if $b_{i_1} = 1$ and $|z_{i_1}| > \xi$, then $\dot{V} \leq -c\,\xi^2$; if $b_{i_1} = 1$ and $|z_{i_1}| < \xi$, then $-\xi < \sum_{j \in \mathcal{V}_{i_1}} (\tau_j - \tau_{i_1}) < \xi$. So

$$\sum_{j \in \mathcal{V}_{i_1}/\{i_0\}} (\tau_j - \tau_{i_1}) < \tau_{i_1} - \tau_{i_0} + \xi < -\frac{\eta}{|\mathcal{V}_{i_0}|} + \xi.$$

Therefore, there exists at least $i_2 \in \Omega \cap \mathcal{V}_{i_1}/\{i_0\}$ such that

$$\tau_{i_2} - \tau_{i_1} < -\frac{\eta}{|\mathcal{V}_{i_0}|(|\mathcal{V}_{i_1}| - 1)} + \frac{\xi}{|\mathcal{V}_{i_1}| - 1}.$$

Combining this inequality and inequality (4.23), we know that either the aforementioned inequality holds or $\dot{V} \leq -c\,\xi^2$. Keeping such reasoning, we reach some i_k such that either

$$\begin{aligned} \tau_{i_k} - \tau_{i_{k-1}} &< -\frac{\eta}{|\mathcal{V}_{i_0}| \prod_{j=1}^{k-1}(|\mathcal{V}_{i_j}| - 1)} \\ &\quad + \xi \sum_{m=1}^{k-1} \frac{1}{\prod_{j=m}^{k-1}(|\mathcal{V}_{i_j}| - 1)} \end{aligned} \tag{4.24}$$

$$\tau_{i_k} \leq \tau_j, \quad \forall j \in \mathcal{V}_{i_k} \tag{4.25}$$

or $|z_{i_k}| \geq \xi$, which implies $\dot{V} \leq -c\,\xi^2$. (To guarantee inequality (4.25), we can always keep this process until the smallest τ_i^* is reached where inequality (4.25) is trivially satisfied.) The aforementioned inequalities imply

$$\begin{aligned} z_{i_k} &= \sum_{j \in \mathcal{V}_{i_k}} (\tau_j - \tau_{i_k}) \geq \tau_{i_{k-1}} - \tau_{i_k} \text{ (because of inequality (4.25))} \\ &> \frac{\eta}{|\mathcal{V}_{i_0}| \prod_{j=1}^{k-1}(|\mathcal{V}_{i_j}| - 1)} - \xi \sum_{m=1}^{k-1} \frac{1}{\prod_{j=m}^{k-1}(|\mathcal{V}_{i_j}| - 1)} \\ &\geq \xi > 0. \end{aligned}$$

So $|z_{i_k}| \geq \xi$ holds. Notice that even if $v_{i_k} \in (\overline{v}_{i_k}, v_{i_k}^{\max}]$, we have $\frac{\partial f(v_{i_k})}{\partial v_{i_k}} \left(-\frac{c v_{i_k}}{\tau_{i_k}} z_{i_k}\right) < 0$, which implies $b_{i_k} = 1$. So, overall we have

$\dot{V} \leq -c\,\xi^2$ for the worst-case when d is maximized. Given this decreasing rate, we know that $t_1 - t_0 \leq \frac{\delta_0 - \delta_1}{c\,\xi^2}$. ■

With Lemma 4.5.2, we have the following result on the convergence time for V to decrease from $V(\tau(0))$ to ε.

Proposition 4.5.3 Given a positive constant ε, let T^* be the time when $\tau(t)$ enters the set Φ. If (4.11) holds for all $t \in [0, T^*]$, then

$$T^* \leq \overline{T} \triangleq \frac{d^2 n \sigma_{\max}(L^\dagger)}{c} \ln\left(\frac{V(\tau(0))}{\varepsilon}\right). \tag{4.26}$$

Proof: Partition the interval $[\varepsilon, V(\tau(0))]$ into p pieces $\varepsilon = \theta_0, \theta_1, \ldots, \theta_p = V(\tau(0))$ with $\Delta\theta = \theta_k - \theta_{k-1} = \frac{V(\tau(0)) - \varepsilon}{p}$ being the length of each subinterval. By Lemma 4.5.2, it takes at most $\frac{(\theta_k - \theta_{k-1}) d^2 n \sigma_{\max}(L^\dagger)}{c\theta_{k-1}}$ for V to decrease from θ_k to θ_{k-1}. Thus,

$$T^* \leq \sum_{k=0}^{p} \frac{d^2 n \sigma_{\max}(L^\dagger)\Delta\theta}{c\theta_{k-1}}.$$

Letting $\Delta\theta \to 0$, we have

$$\begin{aligned} T^* &\leq \lim_{\Delta\theta \to 0} \sum_{k=0}^{p} \frac{d^2 n \sigma_{\max}(L^\dagger)\Delta\theta}{c\theta_{k-1}} \\ &= \frac{d^2 n \sigma_{\max}(L^\dagger)}{c} \int_{\varepsilon}^{V(\tau(0))} \frac{1}{\theta} d\theta \\ &= \frac{d^2 n \sigma_{\max}(L^\dagger)}{c} \ln\left(\frac{V(\tau(0))}{\varepsilon}\right). \end{aligned}$$

■

Remark 4.5.4 If the parameter d is large, the upper bound on T^* will be large, which means that it might take more time for agents to reach consensus on the ETA. The parameter d is completely determined by the topology of the graph. To be more precise, it is determined by the degrees of agents. For instant, if the connected graph of n agents forms a circle where the degree of each agent is 2, then $d = 2n$. Notice that for a path of length k, $i_0, i_1, \dots, i_k$, d is only affected by the degrees of the first k agents. Since d is developed for the worst case, the resulting upper bound $\overline{T}$ might be conservative. Fortunately, we can still reduce $\overline{T}$ by setting a large c.

Remark 4.5.5 Notice that the bound on T^*, which is $\overline{T}$, is also affected by $\sigma_{\max}(L^\dagger)$. Since $L^\dagger$ is the pseudoinverse of L, we have $\sigma_{\max}(L^\dagger) = \frac{1}{\lambda_2(L)}$, where $\lambda_2(L)$ is the smallest positive eigenvalue of L. It is well known that $\lambda_2(L)$ determines the connectivity of the graph. By the definition of $\overline{T}$ in (4.26), strong connectivity will result in large $\lambda_2(L)$ and therefore small $\sigma_{\max}(L^\dagger)$ that helps to reduce $\overline{T}$. Meanwhile, it will lead to large d that may cause large $\overline{T}$. This is different from the case of unconstrained consensus, where strong connectivity implies fast convergence. The relation between connectivity and $\overline{T}$ will be an interesting research topic to be investigated in the future.

4.6 Feasibility

This section discusses feasibility of the simultaneous arrival problem: given a set of initial conditions, is simultaneous arrival achievable under the proposed consensus protocol? Although Proposition 4.5.3 shows the convergence time, the result is based on the assumption of (4.11). The question that follows naturally is whether this assumption holds given the initial conditions. Obviously, we never expect the assumption being violated before ε-consensus is reached. Meanwhile, it is unexpected that any agent arrives before ε-consensus is achieved. Therefore, the entering time T^* must be smaller than (i) the time when (4.11) is violated; and (ii) the time when the first agent arrives at its destination.

We first study the time interval when (4.11) holds, starting from the bounds on $l_i(t)$. By (4.1), we know that $-v_i^{\max} \leq \dot{l}_i \leq -v_i^{\min}$. Therefore, we have

$$l_i(0) - v_i^{\max} t \leq l_i(t) \leq l_i(0) - v_i^{\min} t,$$

which indicates

$$\frac{l_i(0) - v_i^{\max} t}{\underline{v}_i} \le \frac{l_i(t)}{\underline{v}_i} \le \frac{l_i(0) - v_i^{\min} t}{\underline{v}_i}$$
$$\frac{l_i(0) - v_i^{\max} t}{\overline{v}_i} \le \frac{l_i(t)}{\overline{v}_i} \le \frac{l_i(0) - v_i^{\min} t}{\overline{v}_i}.$$

With these bounds, the feasible set of the time-to-arrival $\left[\frac{l_i(t)}{v_i^{\max}}, \frac{l_i(t)}{v_i^{\min}}\right]$ satisfies

$$\left[\frac{l_i(0) - v_i^{\min} t}{\overline{v}_i}, \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i}\right] \subseteq \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right]$$
$$\subseteq \left[\frac{l_i(0) - v_i^{\max} t}{\overline{v}_i}, \frac{l_i(0) - v_i^{\min} t}{\underline{v}_i}\right]. \qquad (4.27)$$

Obviously, if $\cap_{i=1}^{n} \left[\frac{l_i(0) - v_i^{\min} t}{\overline{v}_i}, \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i}\right] \neq \emptyset$, then $\cap_{i=1}^{n} \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i}\right] \neq \emptyset$.

Lemma 4.6.1 If

$$\max_{i\in\mathcal{V}} \frac{l_i(0) - v_i^{\min} t}{\overline{v}_i} \le \min_{i\in\mathcal{V}} \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i}, \qquad (4.28)$$

then $\cap_{i=1}^{n} \left[\frac{l_i(0) - v_i^{\min} t}{\overline{v}_i}, \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i}\right] \neq \emptyset$.

Proof: The proof is straightforward. If the maximum of the lower bounds of the sets is not greater than the minimum of the upper bounds, then the intersection of these sets is nonempty.

■

To enforce inequality (4.28) holds at least over a short period, we need the condition that

$$\max_{i\in\mathcal{V}} \frac{l_i(0)}{\overline{v}_i} < \min_{i\in\mathcal{V}} \frac{l_i(0)}{\underline{v}_i}. \qquad (4.29)$$

This condition guarantees that inequality (4.28) holds when $t = 0$. Let $T_\cap$ denote the time when inequality (4.28) is violated for the first time, that is,

$$T_\cap \triangleq \max\left\{ s \in \mathbb{R}^+ \;\middle|\; \max_{i\in\mathcal{V}} \frac{l_i(0) - v_i^{\min} t}{\overline{v}_i} \le \min_{i\in\mathcal{V}} \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i}, \quad \forall t \in [0, s] \right\}.$$

Notice that when $T^* < T_\cap$, consensus on the ETA is achieved before the intersection of the ETA feasible sets becomes empty.

Another important point is that consensus is expected before any agent arrives at its destination ($l_i(t) = 0$). Obviously, the objective can be achieved if $T^* < \min_{i\in\mathcal{V}} \frac{l_i(0)}{v_i^{\max}}$, where $\frac{l_i(0)}{v_i^{\max}}$ is the shortest time for agent i to arrive at the destination. According to these observations, we summarize the main result in the following theorem:

Theorem 4.6.2 Consider the system defined by (4.1)–(4.6) with a connected communication graph. Assume that inequality (4.29) holds and the positive constant c is chosen such that $\overline{T} < T_m$, where $\overline{T}$ is defined in (4.26) and $T_m = \min\left\{ \min_{i\in\mathcal{V}} \frac{l_i(0)}{v_i^{\max}}, T_\cap \right\}$. Then simultaneous arrival can be achieved, that is, the ETA $\tau_i(t)$ will converge into the set Φ defined in (4.17) before any agent's arrival.

Proof: By the definition of T_m, we know that inequality (4.28) holds for any $t \in [0, T_m]$ and therefore for any $t \in [0, \overline{T}]$ since $\overline{T} < T_m$. Therefore, by Lemma 4.6.1, $\cap_{i=1}^n \left[\frac{l_i(0) - v_i^{\min} t}{\overline{v}_i}, \frac{l_i(0) - v_i^{\max} t}{\underline{v}_i} \right] \neq \emptyset$ holds over $[0, \overline{T}]$, which implies $\cap_{i=1}^n \left[\frac{l_i(t)}{\overline{v}_i}, \frac{l_i(t)}{\underline{v}_i} \right] \neq \emptyset$ by (4.27). By Proposition 4.5.3, $\tau_i(t)$ goes into the set Φ at T^*, which is less than $\overline{T}$. Notice that the definition of T_m ensures that $l_i(0) \ge 0$ for all $t \in [0, T_m]$. Since $T^* \le \overline{T} < T_m$, we can conclude that consensus on the ETA can be achieved before any agent's arrival. ■

Remark 4.6.3 By the definition of $\overline{T}$ in (4.26), it can be arbitrarily small by enlarging c. Thus, the condition $\overline{T} < T_m$ can always be satisfied. The most critical condition for feasibility is then inequality (4.29). Notice that if $\epsilon \to 0$ in (4.10), we have $\overline{v}_i \to v_i^{\max}$ and $\underline{v}_i \to v_i^{\min}$. Therefore, inequality (4.29) actually places the requirement on the initial condition, in terms of the length of the paths, and the maximal/minimum velocity. Basically, Theorem 4.6.2 says that if the initial condition satisfies inequality (4.29), simultaneous arrival can always be achieved under our consensus protocol with c large enough.

4.7 Simulation

This section presents the simulation results. We first consider the system with four agents, where the lengths of the paths are

$$l(0) = \{6, 8, 5, 9\}.$$

The velocity of each agent satisfies

$$v_1 \in [1, 5], \quad v_2 \in [2, 8],$$
$$v_3 \in [1, 5], \quad v_4 \in [2, 7].$$

We choose $\epsilon = 0.01$. Then $\overline{v}_i$ and $\underline{v}_i$ are computed as follows:

$$\underline{v}_1 = 1.0099, \quad \underline{v}_2 = 2.0149, \quad \underline{v}_3 = 1.0099, \quad \underline{v}_4 = 2.0124,$$
$$\overline{v}_1 = 4.9901, \quad \overline{v}_2 = 7.9851, \quad \overline{v}_3 = 4.9901, \quad \overline{v}_4 = 6.9876.$$

Equation (4.29) can be verified as

$$\max_{i \in \mathcal{V}} \frac{l_i(0)}{\overline{v}_i} = 1.2880 < 3.9704 = \min_{i \in \mathcal{V}} \frac{l_i(0)}{\underline{v}_i}.$$

The communication graph is simply a chain, where agent i can communicate with agent $i - 1$ and $i + 1$ except that agent 1 and N only talk to agent 2 and $N - 1$, respectively. We set $\varepsilon = 0.1$, $c = 40$, and $v_i(0) = \overline{v}_i$, the following parameters can be computed:

$$d = 4, \quad \Lambda_{\max}(L^{\dagger}) = 1.7071, \quad \overline{T} = 0.5434,$$
$$T_{\cap} = 0.7290, \quad \min_{i \in \mathcal{V}} \frac{l_i(0)}{v_i^{\max}} = 1,$$

which verifies the condition

$$\overline{T} < T_m = \min \left\{ \min_{i \in \mathcal{V}} \frac{l_i(0)}{v_i^{\max}}, T_{\cap} \right\}.$$

The simulation results are presented in Figures 4.1–4.3. Figure 4.1 shows that history of the velocity of each individual

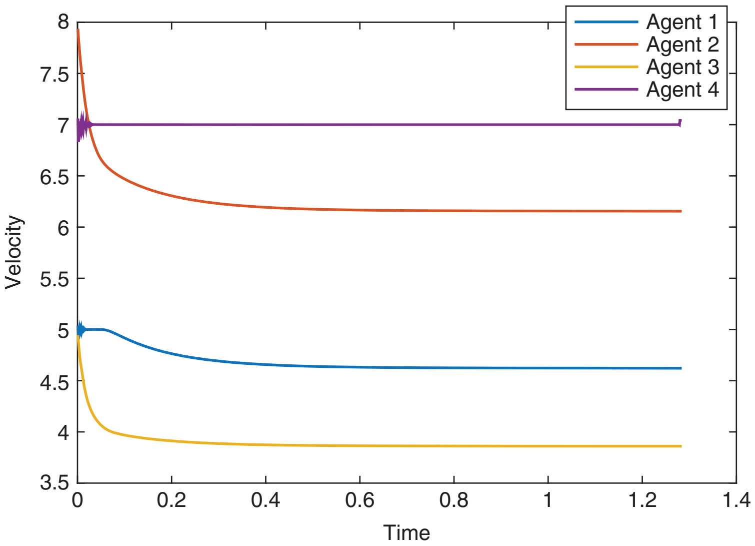


Figure 4.1 The history of the velocity with $c = 40$.

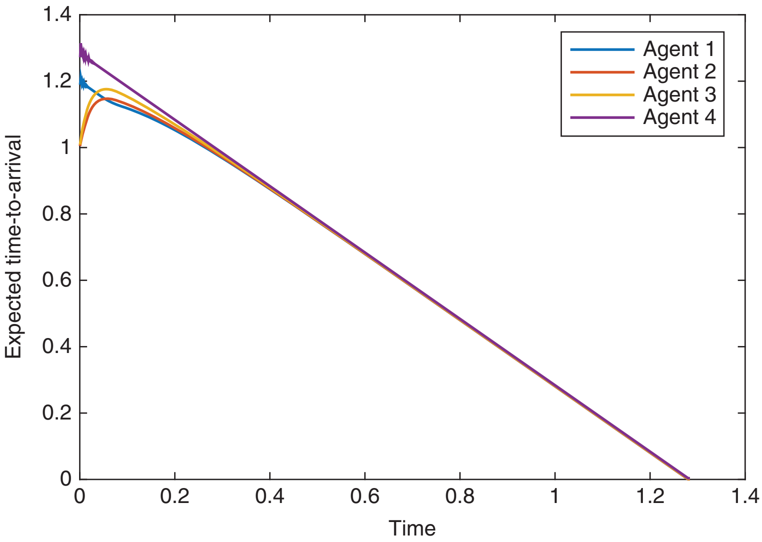


Figure 4.2 The history of the ETA with $c = 40$.

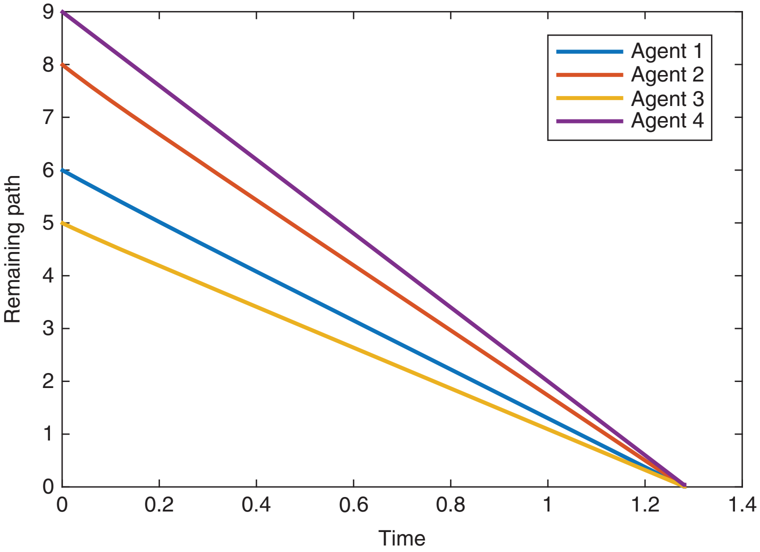


Figure 4.3 The history of the remaining paths with $c = 40$.

agent, which eventually converges to a constant. Notice that the velocity of agent 4 reaches its maximum from $\overline{v}_4$ since its path is the longest. Figure 4.2 plots the history of the ETA. It can be seen that the ETA converges after $t = 0.6$, which is close to the derived bound on T^* ($\overline{T} = 0.5434$)[1]. The same observation can also be found in Figure 4.1. Figure 4.3 shows the history of the remaining path. We can see that eventually agents complete their trips at the same time.

When the parameter c is changed to be 1, simultaneous arrival cannot be achieved any more from Figures 4.4–4.6. From Figure 4.6, we can find that when agent 2 first arrives, agents 1 and 3 are relatively close to their destinations, but agent 4 still has a long distance to travel. Therefore, from Figure 4.4, we can see that agent 2 reduces its velocity to wait for others. So does agent 3 with a smaller decreasing rate. Agent 4 already reaches the maximum of its velocity, trying to catch up with others.

[1] Of course, $\overline{T}$ heavily depends on the size of the ultimate set determined by ε. If we reduce ε, $\overline{T}$ will be larger.

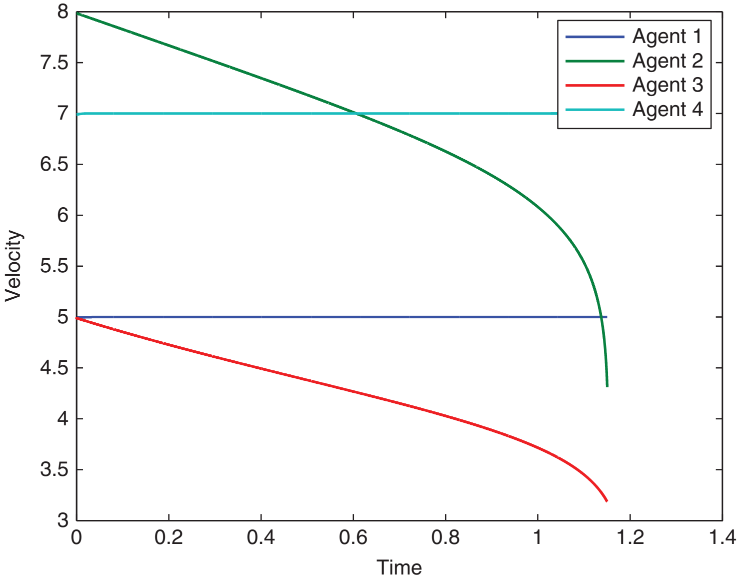


Figure 4.4 The history of the velocity with $c = 1$.

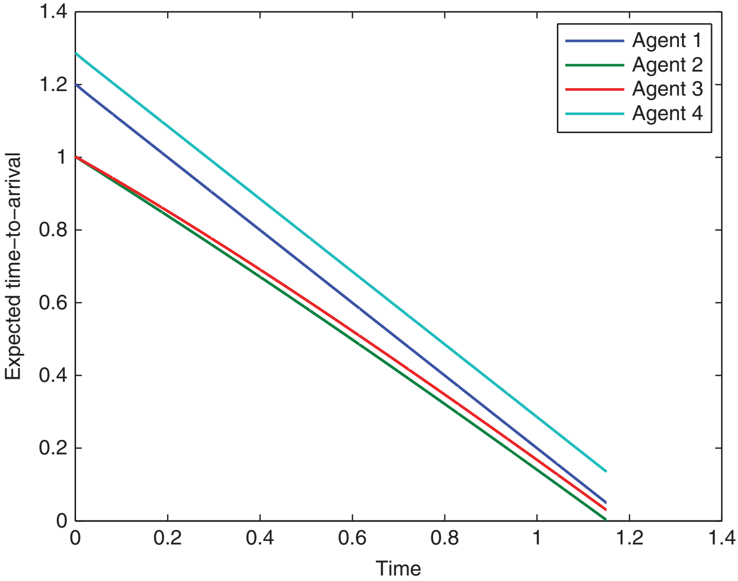


Figure 4.5 The history of the ETA with $c = 1$.

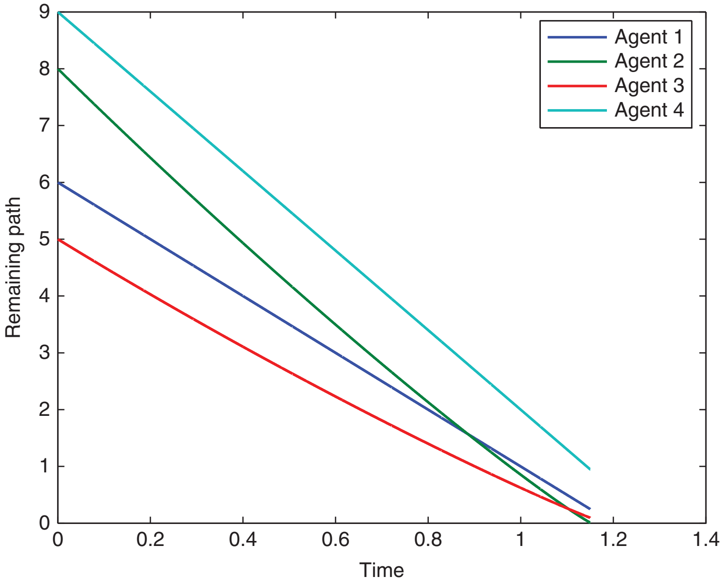


Figure 4.6 The history of the remaining paths with $c = 1$.

In this case, the convergence rate of the consensus is not fast enough and arrival happens earlier than consensus is reached.

The third simulation considers time-varying constrains on the velocity. The disturbances are added to affect the velocity constraints as follows:

$$v_1 \in [1 + w_1(t), 5 + \overline{w}_1(t)], \quad v_2 \in [2 + w_2(t), 8 + \overline{w}_2(t)],$$
$$v_3 \in [1 + w_3(t), 5 + \overline{w}_3(t)], \quad v_4 \in [2 + w_4(t), 7 + \overline{w}_4(t)],$$

where $w_i(t)$ and $\overline{w}_i(t)$ are randomly generated satisfying uniform distributed over $[-0.3, 0.3]$. the parameter c is set back to 40. Figures 4.7–4.9 show the simulation results. We can find in Figures 4.8 and 4.9 that simultaneous arrival can still be achieved. However, in Figure 4.7, there is a small oscillation in the velocity of agent 4 at the beginning, because at that time the velocity is close to its upper limit and tends to increase while the maximum velocity happens to be reduced by the disturbance. The velocities of other agents are away from the boundary. So they are not affected by disturbances. Another observation is

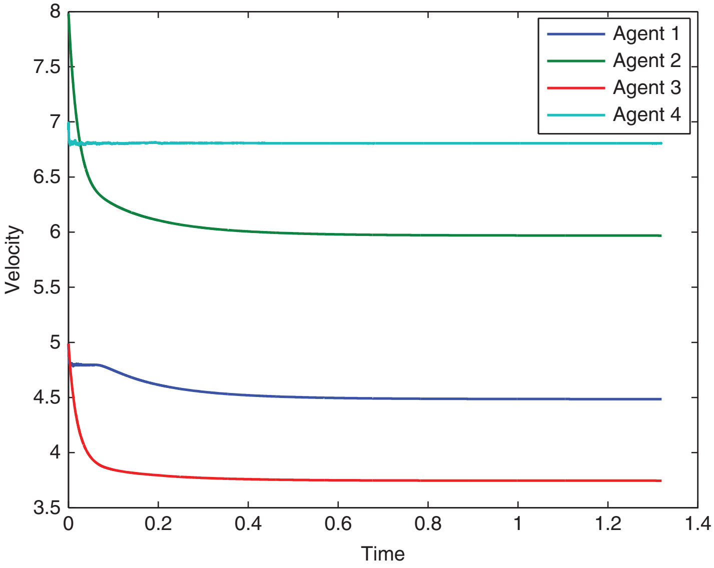


Figure 4.7 The history of the velocity with disturbances.

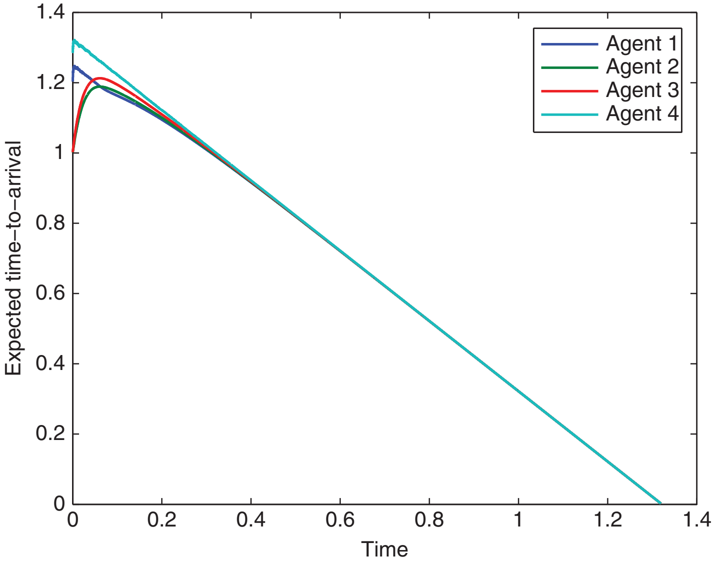


Figure 4.8 The history of the ETA with disturbances.

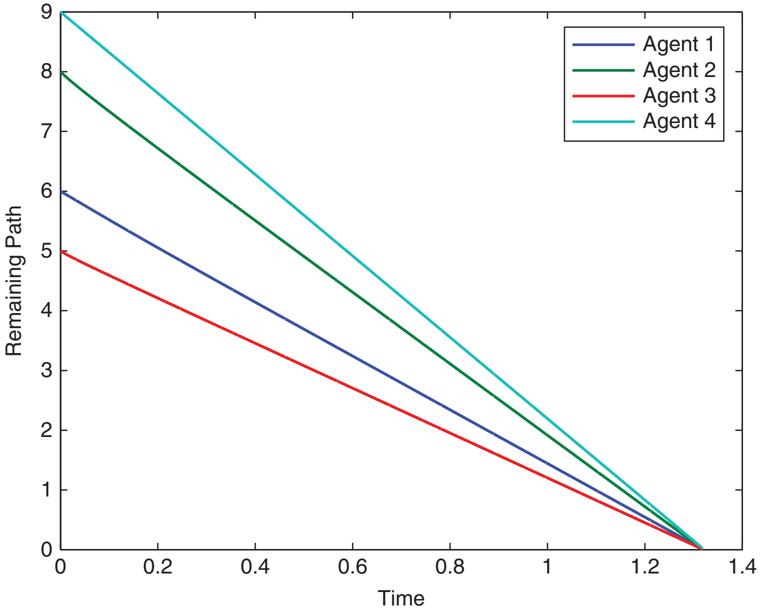


Figure 4.9 The history of the remaining paths with disturbances.

that although consensus is reached, the velocity converges to a different equilibrium, compared with the first simulation.

4.8 Summary

The simultaneous arrival problem with velocity constraints is formulated in this chapter as a continuous-time consensus problem. To deal with constraints, we introduce the projection-based operator to smooth the saturation effect. With this operator, we presented a continuous-time consensus algorithm, of which, we show the convergence and derive the convergence rate. By applying these results in the simultaneous arrival problem, we obtain a sufficient condition on the problem feasibility as shown in (4.29). There are still many open problems to be solved. For instance, when deriving the convergence time, we consider ε-consensus, which means that as long as $|\tau_i(t) - \tau_j(t)| \leq \varepsilon$, we claim that consensus on the ETA is reached. An alternative is to consider finite-time consensus, where the exact consensus will be achieved [27]. Also, the convergence rate might be conservative since it considers the worst-case over the entire runtime. A less conservative convergence rate is expected in the future work.

References

1 McLain, T.W. and Beard, R.W. (2000) Trajectory planning for coordinated rendezvous of unmanned air vehicles, in Proceedings of the AIAA Guidance, Navigation, and Control Conference, vol. 4369, AIAA Reston, VA, pp. 1–8.

2 McLain, T.W., Chandler, P.R., Rasmussen, S., and Pachter, M. (2001) Cooperative control of UAV rendezvous, in Proceedings of the American Control Conference, vol. **3**, IEEE, pp. 2309–2314.

3 Beard, R.W., McLain, T.W., Goodrich, M.A., and Anderson, E.P. (2002) Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, **18** (6), 911–922.

4 Xargay, E., Dobrokhodov, V., Kaminer, I., Pascoal, A.M., Hovakimyan, N., and Cao, C. (2012) Time-critical cooperative control of multiple autonomous vehicles. *IEEE Control Systems Magazine*, **32** (5), 49–73.

5 Kingston, D.B., Ren, W., and Beard, R.W. (2005) Consensus algorithms are input-to-state stable, in Proceedings of American Control Conference, IEEE, pp. 1686–1690.

6 Lynch, N.A. (1996) *Distributed Algorithms*, Morgan Kaufmann.

7 Ren, W., Beard, R.W., and Atkins, E.M. (2005) A survey of consensus problems in multi-agent coordination, in American Control Conference, 2005. Proceedings of the 2005, IEEE, pp. 1859–1864.

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

9 Blondel, V., Hendrickx, J., Olshevsky, A., and Tsitsiklis, J. (2005) Convergence in multi-agent coordination consensus and flocking, in Proceedings of the Joint 44th IEEE Conference on Decision and Control and European Control Conference, pp. 2996–3000.

10 Ren, W., Beard, R., and Atkins, E. (2005) A survey of consensus problems in multi-agent coordination, in Proceedings of the American Control Conference, pp. 1859–1864.

11 Seo, J.H., Shim, H., and Back, J. (2009) Consensus of high-order linear systems using dynamic output feedback compensator: low gain approach. *Automatica*, **45** (11), 2659–2664.

12 Li, Z., Duan, Z., Chen, G., and Huang, L. (2010) Consensus of multiagent systems and synchronization of complex networks: a unified viewpoint. *IEEE Transactions on Circuits and Systems I: Regular Papers*, **57** (1), 213–224.

13 Carli, R., Fagnani, F., Speranzon, A., and Zampieri, S. (2008) Communication constraints in the average consensus problem. *Automatica*, **44** (3), 671–684.

14 Wang, Q., Gao, H., Alsaadi, F., and Hayat, T. (2014) An overview of consensus problems in constrained multi-agent coordination. *Systems Science & Control Engineering: An Open Access Journal*, **2** (1), 275–284.

15 Abdessameud, A. and Tayebi, A. (2010) On consensus algorithms for double-integrator dynamics without velocity measurements and with input constraints. *System & Control Letters*, **59** (12), 812–821.

16 Yang, T., Meng, Z., Dimarogonas, D.V., and Johansson, K.H. (2014) Global consensus for discrete-time multi-agent systems with input saturation constraints. *Automatica*, **50** (2), 499–506.

17 Nedic, A., Ozdaglar, A., and Parrilo, P.A. (2010) Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, **55** (4), 922–938.

18 Lin, P. and Ren, W. (2014) Constrained consensus in unbalanced networks with communication delays. *IEEE Transactions on Automatic Control*, **59** (3), 775–781.

19 Lee, U. and Mesbahi, M. (2011) Constrained consensus via logarithmic barrier functions, in Decision and Control and

European Control Conference (CDC-ECC), 2011 50th IEEE Conference on, IEEE, pp. 3608–3613.

20 Wang, X. and Zhou, Z. (2015) Projection-based consensus for continuous-time multi-agent systems with state constraints, in Decision and Control (CDC), 2015 IEEE 54th Annual Conference on, pp. 1060–1065, doi: 10.1109/CDC.2015.7402012.

21 Shi, G. and Hong, Y. (2009) Distributed coordination of multi-agent systems with switching structure and input saturation, in Proceedings of the 48th IEEE Conference on Decision and Control, Held Jointly with the 28th Chinese Control Conference, IEEE, pp. 895–900.

22 Larchev, G.V., Campbell, S.F., and Kaneshige, J.T. (2010) Projection operator: a step towards certification of adaptive controllers. National Technical Information Service, Alexandria, VA, 22312, pp. 109–118.

23 Lavretsky, E., Gibson, T.E., and Annaswamy, A.M. (2011) Projection operator in adaptive systems.

24 Pomet, J.B. and Praly, L. (1992) Adaptive nonlinear regulation: estimation from the Lyapunov equation. *IEEE Transactions on Automatic Control*, **37** (6), 729–740.

25 Ioannou, P. and Sun, J. (1996) *Robust Adaptive Control*, Prentice Hall.

26 Bauso, D., Giarré, L., and Pesenti, R. (2009) Consensus for networks with unknown but bounded disturbances. *SIAM Journal on Control and Optimization*, **48** (3), 1756–1770.

27 Cortés, J. (2006) Finite-time convergent gradient flows with applications to network consensus. *Automatica*, **42** (11), 1993–2000.

5

Greedy Maximization for Asset-Based Weapon–Target Assignment with Time-Dependent Rewards

Doo-Hyun Cho and Han-Lim Choi

Department of Aerospace Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea

5.1 Introduction

The weapon–target assignment (WTA) is a problem that assigns defensive weapons appropriately to offensive targets to minimize the damage of own-force assets (called asset-based WTA) [1] or to maximize the damage of offensive targets (called target-based WTA) [2]. The WTA problem can be formulated as a mixed-integer nonlinear program (MINLP), which is known to be NP-complete [3]; thus, the computational cost exponentially blows up as the number of targets and the defensive weapons increases. Even for a small-sized WTA (e.g., with 20 weapons for 20 targets), an exact algorithm has not been successfully developed [4].

Therefore, most previous work has focused devising heuristics (or meta-heuristic) methods for the WTA problem such as ant colony optimization (ACO) [5], particle swarm optimization (PSO) [6], Tabu search (TS) [7, 8], simulated annealing (SA) [9], and genetic algorithm (GA) [10]. While the aforementioned work just implemented the generic optimization methods to WTAs, there have also been approaches that adapt/combine meta-heuristics methods tailored to WTAs. ACO is adapted for solving general WTA problem [11], and the search performance of GA algorithms are enhanced by embedding ACO to have locally optimal offspring [12]. In [13], the WTA problem is

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

solved with basic idea of PSO, called hybrid PSO algorithm. And in [14], the advantages of PSO and GA are combined to obtain rapid convergence and higher solution precision. In [15], TS is utilized to improve the decision tree to solve the naval warfare resource management problem, which is similar to WTA problem. Xin et al. [16] proposed TS-based algorithms to solve a generic asset-based DWTA problem. Lee et al. [17] proposed a GA with greedy eugenics to apply to general WTA problems. Hong et al. [18] proposed a heuristic algorithm to improve a GA that generates the initial population. In [19], GA, TS, SA, and variable neighborhood search combinatorial optimization techniques are applied to the WTA problem and their results are compared. A network flow-based construction heuristic and a very large-scale neighborhood (VLSN) search algorithm was suggested by Ahuja et al. [4].

WTA problems can often be categorized into two groups—static WTA (SWTA) [20] and dynamic WTA (DWTA) [21]. This categorization occurs with the difference of whether an information feedback between previous and next assignment stage exists. In other words, the DWTA problem is a generalization of its static counterpart; a method for the SWTA can be utilized as a subroutine of the dynamic case [4]. In the general SWTA problem, it is assumed that every parameter for agents, targets, and defensive missiles is fixed, and all of the defensive weapons are launched simultaneously. This simplified setting may not be appropriate to present a real-world engagement situation, in which some type of rules of engagements need to be satisfied and geometric aspects of the weapon deployment can make differences. One of the major factors this work tries to focus on is the launching time of a defensive missile. Usually, an aerial enemy target maneuvers at very high speed; therefore, the length of the possible time interval for assigning defensive missiles to the specific target is only from few to tens of seconds. Also, the kill probability of the target by any defensive weapons depends on time of engagement; there exists the optimal launching time, which maximizes the kill probability. If the defensive missile has not been assigned with the optimal launching time, then the kill probability is expected to decrease as the assigned time gets further from the optimal time. Hence, it is important to consider appropriate launching time when assigning the defensive missiles to the targets. Here, we define this kind of problem as **time-dependent SWTA** (TSWTA) problem. Although there exist lots of research about WTA problem, we are unaware of any studies that have considered the time-dependent reward in WTA problem. In these days, with a high-powered radar, the model and the impact point of each target can be predicted with a high degree of accuracy.

In [22], the basic static (target-based) WTA problem was studied in the framework of combinatorial optimization theory, and it was shown that the domain of the WTA problem constructs a partition matroid with an objective function being monotone nondecreasing. This chapter takes advantage of findings in [22] and extends it to show that the TSWTA also exhibits the monotonically nondecreasing property similar to other WTA problems. This property allows for adoption of the greedy maximization algorithm with provable suboptimality bound, which is $\frac{1}{1+\mu}$ of the optimum with some positive number μ.

The rest of the chapter is organized as follows: In Section 5.2, the problem formulation is proposed to solve the TSWTA with several constraints. The monotonically nondecreasing property of the objective function is discussed in Section 5.3. Based on this property, it is shown that the TSWTA can be formulated as the problem, which maximizes the monotonically nondecreasing function under a partition matroid constraint. In Section 5.4, the details of the proposed algorithm are presented, while computational results are reported in Section 5.5. Summary and conclusions are provided in Section 5.6.

5.2 Problem Formulation

For the purpose of this chapter, several terms are defined to mean the following: An *asset* is a valuable property to be protected. A *target* is an object that threats an asset with a property of destroying some portion of the aimed asset if finally impacts on it. To protect the given assets, defensive *weapons* (i.e., guided missiles) are launched to intersect the aimed targets from one of the *weapon farms* with a certain degree of probability that exists in a time window to destroy the target (Figures 5.2, 5.3b, c). From the decision-making perspective, each weapon farm plays a role of decision-making agent; there are multiple of these agents that cooperatively defense the assets by assigning their resources. Figure 5.1 illustrates the engagement setup.

The TSWTA problem can be formulated as an MINLP with the following set of variables and constraints, under the assumption that the target–assets engagements are independent of weapon–target engagements.

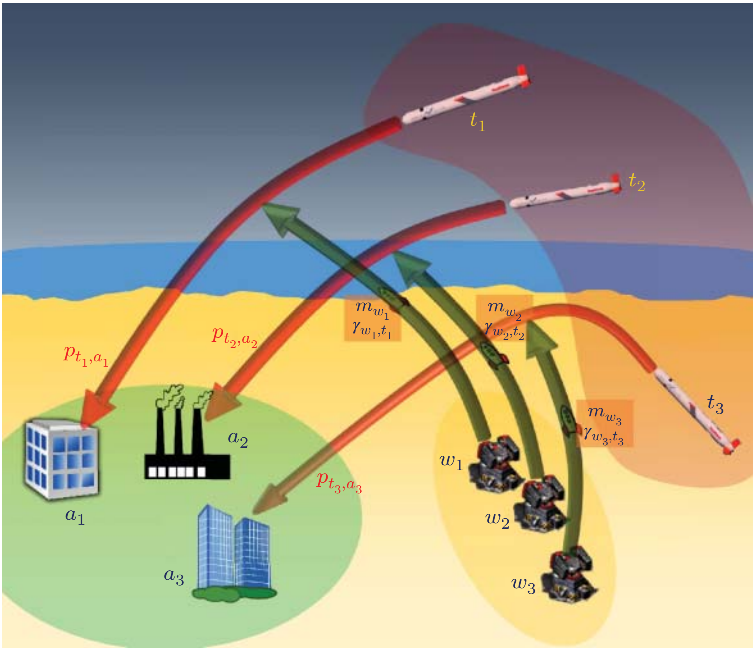


Figure 5.1 Battlefield Diagram. The targets (three incoming missiles on the upper right) are aimed to attack each asset (buildings at the bottom left) along the arrows, and the defensive weapons (or interceptor missiles) from each weapon farm (at the bottom) are launched to intercept targets along the arrows and defend the assets. Each weapon farm may launch multiple defensive weapons, although not depicted in the figure.

5.2.1 Problem Variables

1) Input Parameters

T Set of all targets, $t \in T$.

A Set of all assets, $a \in A$.

W Set of all weapon farms, $w \in W$.

M_w Set of all defensive weapons in weapon farm w, $m_w \in M_w$.

M Set of all weapons. $M = (M_1 \cup M_2 \cup \ldots \cup M_{|W|})$.

$v(a)$ Value of an asset a, $v(a) \in \mathbb{R}^+$.

$p_{t,a}$ Target t's destruction ratio for corresponding asset a, $p_{t,a} \in [0, 1]$. If a target t is not aimed to an asset a, then $p_{t,a}$ is set to 0.

τ_w^d Launching time delay of a weapon farm w.

$\gamma_{w,t}(\tau)$ Kill probability of target t by a missile launched from the weapon farm w. Here, τ is a missile launching time from a weapon farm w. $\gamma_{w,t}(\tau) \in [0, 1], \tau \in \mathbb{R}^+$

$\mathbf{x}_a, \mathbf{x}_t, \mathbf{x}_w.$ Position vector of asset a, target t, and weapon farm w $\mathbf{x} \in \mathbb{R}^2, \mathbb{R}^3$

2) Decision Variables

$\theta_{w,t}(m) \in \boldsymbol{\Theta}$ Assignment of a missile m in weapon farm w. The parameter "t" indicates that the weapon goes toward the corresponding target. $\theta_{w,t}(m) \in \{0, 1\}$. $\theta_{w,t}(m) = 1$ if a missile m in weapon farm w is assigned to target t, otherwise 0

$\tau_w(m) \in \mathbf{T}$ Scheduled launching time of a missile m in weapon farm w. $\tau_w(m) \in \mathbb{R}^+$

5.2.2 Constraints

The following constraints should be satisfied with the decision of assignment:

1) Weapon assignment constraint

$$\sum_{t \in T} \theta_{w,t}(m) \leq 1 \quad \forall w \in W, m \in M_w. \tag{5.1}$$

Equation (5.1) states that each defensive weapon m can be assigned to only single target.

2) Launching delay constraint

$$|\tau_w(m_1) - \tau_w(m_2)| \geq \tau_w^d \quad \forall w \in W, m_1, m_2 \in M_w, m_1 \neq m_2$$

(5.2)

Equation (5.2) states that the gap of launching time between missile m_1 and m_2 cannot be shorter than the weapon farm w's launching time delay.

5.2.3 Objective Function

$$\max J = \sum_{a=1}^{|A|} \left[v(a) \cdot \prod_{t=1}^{|T|} \left\{ 1 - p_{t,a} \right.\right.$$
$$\left.\left. \times \left(\prod_{w=1}^{|W|} \prod_{m=1}^{|M_w|} (1 - \gamma_{w,t}(\tau_w(m)) \cdot \theta_{w,t}(m)) \right) \right\} \right]. \tag{5.3}$$

The objective value J is the sum of asset values after defensive weapons in weapon farms are assigned to corresponding targets. Suppose the value of the asset a is $v(a)$, and the target t is known to attack the asset a with a destruction ratio $p_{t,a}$. As the number of the assigned defensive weapons to the target t increases and if the targets t aim to the asset a, the term multiplied with $p_{t,a}$, that is, the second line of the (5.3), becomes smaller, therefore the value of J increases. Here, the kill probability $\gamma_{w,t}$ depends on the launching time $\tau_w(m)$, so assigning the defensive weapons to the target with appropriate time should be considered. And $\theta_{w,t}(m)$ represents that assigned defensive weapons are only considered when calculating the kill probability. The main purpose is, by properly assigning weapons, to find the solution to maximize the overall survivability of the given assets threatened by the offensive targets.

5.3 Properties of the Objective Function

In this section, it is shown that the problem formulated in Section 5.2 is a maximization problem of a nondecreasing function on a partition matroid (the definition of a matroid is given later).[1] By this, the lower bound of the value achieved by a (sequential) greedy algorithm can be obtained.

[1] In this section, the form of proof is referenced from [22].

5.3.1 Preliminary—Greedy Algorithm

The greedy algorithm is a process to obtain an approximate solution to a set selection problem with some respective objective function. The algorithm works in stages, as opposed to simultaneously choosing a set, and picks the best (and feasible) choice among possible alternatives at each stage. Thus, the choice at each stage is locally optimal but may not result in the optimal choice of the original set selection problem.

It has been known that the greedy algorithm produces the optimal solution for the problems that satisfy the following two properties: greedy choice property and optimal substructure [23]. The first means that a prior selection does not affect the result of future choices, and the second implies that an optimal solution can be constructed efficiently from optimal solutions of its subproblems.

Although the greedy algorithm does not guarantee solution optimality unless the aforementioned two properties are satisfied, it often produces a practically acceptable solution with low computational complexity. In addition, for a certain class of problems, we can prove that the greedy algorithm produces a solution with guaranteeing the worst-case performance. The following sections detail the derivation of the lower bound for the WTA problem in this chapter.

5.3.2 Preliminary—Maximization of Set Function

Here, we focus on the following maximization problem. A ground set X of n elements and a monotonically nondecreasing set function $f : 2^X \rightarrow \mathbb{R}^+$ are given, and the object of the problem is to find

$$\max_{S \in \mathcal{I}} f(S), \tag{5.4}$$

where $\mathcal{I} \subseteq 2^X$ is an independence family.

Matroid. A matroid is a pair $\mathcal{M} = (X, \mathcal{I})$ where $\mathcal{I} \subseteq 2^X$ is a collection of independence family, satisfying (i) $B \in \mathcal{I}$ and $A \subseteq B$ implies that $A \in \mathcal{I}$, and (ii) $A, B \in \mathcal{I}$, $|A| < |B|$ implies that $\exists i \in B \backslash A; A \cup i \in \mathcal{I}$.

Partition Matroid. A matroid $\mathcal{M} = (X, \mathcal{I})$ is a partition matroid if X is partitioned into l sets $X_1, X_2, \ldots, X_l$ with associated integers $k_1, k_2, \ldots, k_l$, and a set $S \subseteq X$ is independent if and only if $|S \cap X_i| \leq k_i$.

Nondecreasing. A set function f is nondecreasing if $f(A) \leq f(B)$ for all $A \subseteq B$.

Submodular Function. A function f is submodular if $\forall A \subseteq B \subseteq X$ and $i \in X \backslash B, f_i(A) \geq f_i(B)$ where $f_i(A) = f(A \cup \{i\}) - f(A)$ and $f_i(B) = f(B \cup \{i\}) - f(B)$.

Elemental Curvature. The elemental curvature of a monotone nondecreasing function f is defined as

$$\alpha = \max_{S \subset X, i,j \in X \backslash S, i \neq j} \frac{f_i(S \cup \{j\})}{f_i(S)}. \tag{5.5}$$

Or α is the smallest value such that, for every $S \subset X$ and $i, j \in X \backslash S$,

$$f(S \cup \{j\} \cup \{i\}) - f(S \cup \{j\}) \leq \alpha[f(S \cup \{i\}) - f(S)].$$

Here $\alpha \in [0, +\infty)$.

Theorem 5.3.1 Let X be a finite set and $f : 2^X \to \mathbb{R}_+$ a monotone nondecreasing set function with $f(\emptyset) = 0$ and elemental curvature α. Then, the greedy algorithm achieves a $\frac{1}{1+\mu}$ approximation for any matroid $\mathcal{M} = (X, \mathcal{I})$, where $\mu = \alpha$ if $0 < \alpha \leq 1$, and $\mu = \frac{\alpha^K(1-\alpha^K)}{K(1-\alpha)}$ if $\alpha > 1$, and K is the cardinality of the largest maximal independent sets.

Proof: Refer to [22], pp. 35–37. ■

5.3.3 Weapon Target Assignment—Lower Bound with Greedy Algorithm

Let the asset aimed by the target t be designated as $a(t)$, and the set of targets aimed at each asset $a(t)$ as H_a (refer a in Section 5.2.1).

Defining a ground set $X = \{(t, w, m | t \in T; w \in W; m \in M\}$, $\mathcal{I}$, the independence family of X, is defined as follows:

$$\begin{aligned} \mathcal{I} = \{S \subseteq X :\ & (|S| \leq |M|) \wedge \\ & (\nexists m, t_1 \neq t_2 : (t_1, w, m) \in S \wedge (t_2, w, m) \in S) \wedge \\ & (\nexists \tau_w(m_1), m_1 \neq m_2 : (t, w, m_1) \in S \wedge (t, w, m_2) \in S \wedge \end{aligned}$$

$$|\tau_w(m_1) - \tau_w(m_2)| < \tau_w^d)\}. \tag{5.6}$$

Here, $\mathcal{I}$ follows the constraint equations described in Section 5.2.2. Also define a function f on a subset $S \in \mathcal{I}$,

$$f(S) = \sum_{a=1}^{|A|} v(a) \cdot \left[\prod_{t \in H_a} \left\{ 1 - p_{t,a(t)} \times \left(\prod_{(w,m):(t,w,m) \in S} (1 - \gamma_{w,t}(\tau_w(m))) \right) \right\} - \prod_{t \in H_a} (1 - p_{t,a(t)}) \right]. \tag{5.7}$$

Note that there is a one-to-one correspondence between the set S and the feasible solutions $[\boldsymbol{\Theta}; \mathbf{T}]$ (set of decision variables defined at Section 5.2.1) to the problem defined earlier, and furthermore, the corresponding solutions have identical values. Hence, the defined problem (Section 5.2, (5.3)) is equivalent to solving $\max_{S \in \mathcal{I}} f(S)$, Eq. (5.4).

And, assigning defensive weapons $m \in M$ in each weapon farm $w \in W$ to targets $t \in T$ is essentially a set partition problem, that is, $\mathcal{M} = (X, \mathcal{I})$ is a partition matroid. Let $X_{w,m} = \{(1, w, m), \dots, (|T|, w, m)\}, w = 1, \dots, |W|, m = 1, \dots, |M_w|$ for each w. It is apparent that $X = X_{1,1} \cup \dots \cup X_{1,M_1} \cup X_{2,1} \cup \dots \cup X_{|W|,|M_{|W|}|}$. Since each defensive weapon can only be assigned to at most one target, $\forall S \in \mathcal{I}$ and $X_{w,m}$, we have $|S \cap X_{w,m}| \leq 1$. It follows that $\mathcal{I}$ is an independent set family and $\mathcal{M}$ is a partition matroid.

(a) **Nondecreasing Property of f.**
From (5.7), $f(\emptyset) = 0$. Let $\delta_{t,w,m} = (1 - \gamma_{w,t}(\tau_w(m)))$ for simplicity, and $\forall S \in \mathcal{I}, s = (t_1, w_1, m_1) \in X \backslash S$.

$$\begin{aligned} f(S \cup \{s\}) - f(S) &= \sum_{a=1}^{|A|} v(a) \cdot \left[\prod_{t \in H_a} \left\{ 1 - p_{t,a(t)} \cdot \left(\prod_{(w,m):(t,w,m) \in S \cup \{s\}} \delta_{t,w,m} \right) \right\} \right. \\ &\quad \left. - \prod_{t \in H_a} \left\{ 1 - p_{t,a(t)} \cdot \left(\prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m} \right) \right\} \right]. \end{aligned} \tag{5.8}$$

This result shows that f is a monotone nondecreasing function with $f(\emptyset) = 0$.

(b) **Optimal Lower Bound—Elemental Curvature $\boldsymbol{\alpha}$.**

The elemental curvature α of$f$ is calculated here. For each $S \in \mathcal{I}, s_1 = (t_1, w_1, m_1) \in X \backslash S, s_2 = (t_2, w_2, m_2) \in X \backslash S,$ and $s_1 \neq s_2$. To compute α, there are three cases to be considered:

(1) If $t_1 = t_2$, then the assets aimed by $t_1, t_2 - a(t_1), a(t_2)$ are same.

$$f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})$$

$$= v(a(t_1)) \times \prod_{\substack{t \in H_{a(t_1)} \\ \backslash \{t_1\}}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m}\right)$$

$$\times \prod_{(w,m):(t_1,w,m) \in S \cup \{s_2\}} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)), \tag{5.9}$$

$$f(S \cup \{s_1\}) - f(S)$$

$$= v(a(t_1)) \times \prod_{t \in H_{a(t_1)} \backslash \{t_1\}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m}\right)$$

$$\times \prod_{(w,m):(t_1,w,m) \in S} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)). \tag{5.10}$$

Dividing (5.9) by (5.10), the following equation is obtained:

$$\frac{f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})}{f(S \cup \{s_1\}) - f(S)} = 1 - \gamma_{w_1,t_1}(\tau_{w_1}(m_1)) \leq 1. \tag{5.11}$$

(2) If $t_1 \neq t_2$, that is, targets are different and $a(t_1) = a(t_2)$ (targets are aiming at same asset),

$$f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})$$

$$= v(a(t_1)) \prod_{t \in H_{a(t_1)} \backslash \{t_1\}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S\{s_2\}} \delta_{t,w,m}\right),$$

$$\times \prod_{(w,m):(t_1,w,m) \in S} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)) \tag{5.12}$$

$$f(S \cup \{s_1\}) - f(S)$$

$$= v(a(t_1)) \times \prod_{t \in H_{a(t_1)} \setminus \{t_1\}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m}\right)$$
$$\times \prod_{(w,m):(t_1,w,m) \in S} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)). \quad (5.13)$$

Dividing (5.12) by (5.13), the following equation is obtained:

$$\frac{f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})}{f(S \cup \{s_1\}) - f(S)}$$
$$= \frac{1 - p_{t_2,a(t_1)} \cdot \prod_{(w,m):(t_2,w,m) \in S \cup \{s_2\}} \delta_{t_2,w,m}}{1 - p_{t_2,a(t_1)} \cdot \prod_{(w,m):(t_2,w,m) \in S} \delta_{t_2,w,m}}$$
$$= \frac{1 - p_{t_2,a(t_1)} \cdot \prod_{(w,m):(t_2,w,m) \in S} \delta_{t_2,w,m} \cdot \delta_{t_2,w_2,m_2}}{1 - p_{t_2,a(t_1)} \cdot \prod_{(w,m):(t_2,w,m) \in S} \delta_{t_2,w,m}} \geq 1. \quad (5.14)$$

From the definition of the submodular function, the function$f$ is not submodular in this case.

(3) If $t_1 \neq t_2$, that is, targets are different and $a(t_1) \neq a(t_2)$ (targets are aiming at different assets),

$$f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})$$
$$= v(a(t_1)) \times \prod_{t \in H_{a(t_1)} \setminus \{t_1\}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m}\right),$$
$$\times \prod_{(w,m):(t_1,w,m) \in S} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)), \quad (5.15)$$

$$f(S \cup \{s_1\}) - f(S)$$
$$= v(a(t_1)) \times \prod_{t \in H_{a(t_1)} \setminus \{t_1\}} \left(1 - p_{t,a(t_1)} \cdot \prod_{(w,m):(t,w,m) \in S} \delta_{t,w,m}\right)$$
$$\times \prod_{(w,m):(t_1,w,m) \in S} (\delta_{t_1,w,m}) \cdot p_{t_1,a(t_1)} \gamma_{w_1,t_1}(\tau_{w_1}(m_1)). \quad (5.16)$$

Therefore,

$$\frac{f(S \cup \{s_1\} \cup \{s_2\}) - f(S \cup \{s_2\})}{f(S \cup \{s_1\}) - f(S)} = 1 \quad (5.17)$$

From (5.11), (5.14), and (5.17),

$$\begin{aligned}
&\frac{f(S\cup\{s_1\}\cup\{s_2\})-f(S\cup\{s_2\})}{f(S\cup\{s_1\})-f(S)}\\
&\le\frac{1-p_{t_2,a(t_1)}\cdot\prod_{(w,m):(t_2,w,m)\in S}\delta_{t_2,w,m}\cdot\delta_{t_2,w_2,m_2}}{1-p_{t_2,a(t_1)}\cdot\prod_{(w,m):(t_2,w,m)\in S}\delta_{t_2,w,m}}\\
&\le\frac{1-p_{t_2,a(t_1)}\cdot\delta_{t_2,w_2,m_2}}{1-p_{t_2,a(t_1)}}.
\end{aligned}\tag{5.18}$$

From (5.5) and (5.18), the elemental curvature can be calculated as follows:

$$\alpha=\max_{t\in T,w\in W,m\in M_w}\frac{1-p_{t,a(t)}\cdot\delta_{t,w,m_w}}{1-p_{t,a(t)}}.\tag{5.19}$$

$\alpha\ge 1$, since $p_{t,a},\delta_{t,w,m}\in[0,1]$.

5.4 Algorithmic Details

As discussed in Section 5.3.3, the performance of the greedy algorithm applied to the static WTA in (5.3) can be lower bounded. This section details the algorithmic process to implement the greedy approach, in particular, taking into account the temporal dependence of the reward on the intercept time. The proposed greedy algorithm works in two phases: (i) *time slot generation* and (ii) *greedy maximization.*

5.4.1 Time Slot Generation

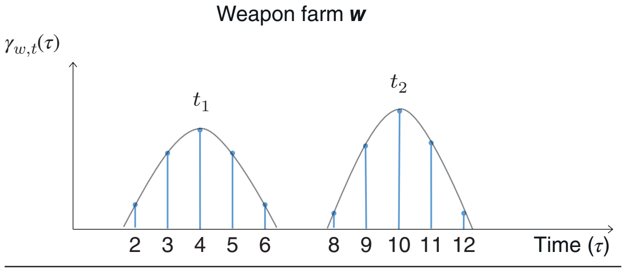
The first phase of the algorithm is to generate time slot for each weapon farm and target pair, $(w,t)\in W\times T$. Since it is very inefficient to search infinitely many candidates of launching time τ for each (w,t), the way of choosing possible launching times is suggested with consideration of the launching time delay of weapon farm τ^d. Here the vector of chosen launching times for each (w,t) is defined as *slot*. Algorithm 1 shows the way of the vector generation. For each (w,t), an empty vector is given and named as *slot*. And then, calculate whether the weapon farm w has a possibility of killing the target t. If it has, then put the time value which maximizes $\gamma_{w,t}$ at *slot*, else continue the procedure to the next (w, t) pair. Put the next time value at *slot* with obeying the launching time delay constraint (5.2) until no more candidate has left. A simple result using this phase is shown in Figure 5.2. The output vector of this phase, *slot*, is used in the second phase to get the solution of TSWTA.

Algorithm 5.1 Generate Time Slot

```
procedure GENTIMESLOT $(W, T)$
    for $w := 1$to $|W|$ do
        for $t := 1$to $|T|$ do
            $slot_{(w,t)} \leftarrow \varnothing$
            $temp\_\tau = \underset{\tau \in \mathbb{R}^+}{\text{argmax}} \left[\gamma_{w,t}(\tau)\right]$
            if $\gamma_{w,t}(\tau) = 0$ then
                continue
            end if
            while (1) do
                $temp\_\tau \leftarrow \underset{\tau \in \mathbb{R}^+}{\text{argmax}} \left[\gamma_{w,t}(\tau)\right]$with time interval at least $\tau_w^d$ among other elements in $slot_{(w,t)}$
                if $\gamma_{w,t}(\tau) = 0$ then
                    break
                else
                    $slot_{(w,t)}(end + 1) \leftarrow temp\_\tau$
                end if
            end while
        end for
    end for
    Return $slot$
end procedure
```

5.4.2 Greedy Maximization

The second phase of this algorithm is to find the solution of the TSWTA problem greedily. At first, the phase starts with initializing several variables. The variable J_{tempmax} starts with the value obtained using (5.20). Equation (5.20) is an objective function,



$$slot_{(w,t_1)} = \quad 4 \quad 3 \quad 5 \quad 2 \quad 6$$

$$slot_{(w,t_2)} = \quad 10 \quad 9 \quad 11 \quad 8 \quad 12$$

Figure 5.2 An example of generating time slot by Algorithm 1.

without assigning any defensive weapons to the targets.

$$J = \sum_{a=1}^{|A|} \left[v_i(a) \cdot \left(\prod_{t=1}^{|T|} \left[1 - p_{t,a}\right] \right) \right]. \tag{5.20}$$

Using the result of the first phase, calculate the objective function value with adding one defensive weapon for every (w, t) pair for each iteration, and save the assignment that showed a biggest marginal gain. During the assignment for each weapon farm w, if there exists a launching time duplication between new assignment and previously assigned solution, then remove the new one and use next candidate by deleting the first element of $slot_{w,t}$. The iteration terminates when (i) all weapons in w are assigned, (ii) no elements are left in *slot* vector, or (iii) no improvement happens on the objective value. Details of finding the heuristic solution are shown in Algorithm 2.

5.5 Numerical Case Studies

5.5.1 Simple TSWTA Example

In this section, the properties of the TSWTA problem are analyzed in detail. To check the property of objective function values, a number of problem instances are created to change the parameters for the simulation. Before investigating the characteristics of the TSWTA problem, the simple instance with

Algorithm 5.2 Greedy Maximization Algorithm

```
procedure GreedyMax ($A$, $W$, $T$, $slot$)
    Initialize $X, X_{temp}, X_{tempmax} \leftarrow \mathbf{0} \in \mathbf{R}^n$    ▷ n : # of state vector element
    Initialize $J, J_{temp}, J_{tempmax} \leftarrow 0$
    $J_{tempmax} \leftarrow$ objfunc_no_defense()
    while (true) do
        for $w$ := 1to $|W|$ do
            if $w$'s weapons are all allocated then
                delete $slot_{w,t}[1]$
            end if
            for $t$ := 1to $|T|$ do
                if $slot_{w,t}[1]$ is overlapping with $w$'s preallocated weapons then
                    delete $slot_{w,t}[1]$
                end if
                if $slot_{(w,t)} = \varnothing$ then
                    continue
                end if
                $X_{temp} \leftarrow X$
                modify $X_{temp}$ (assign additional weapon from $w$ to target $t$) with launching time $slot_{w,t}[1]$
                $J_{temp} = objfunc(X_{temp})$
                if $J_{tempmax} < J_{temp}$ then
                    $X_{tempmax} \leftarrow X_{temp}$
                    $J_{tempmax} \leftarrow J_{temp}$
                end if
                delete $slot_{w,t}[1]$
            end for
        end for
        $X \leftarrow X_{tempmax}$
        $J \leftarrow J_{tempmax}$
        if $slot = \varnothing$ or no improvement of $J$ then
            break
        end if
    end while
    Return $X, J$
end procedure
```

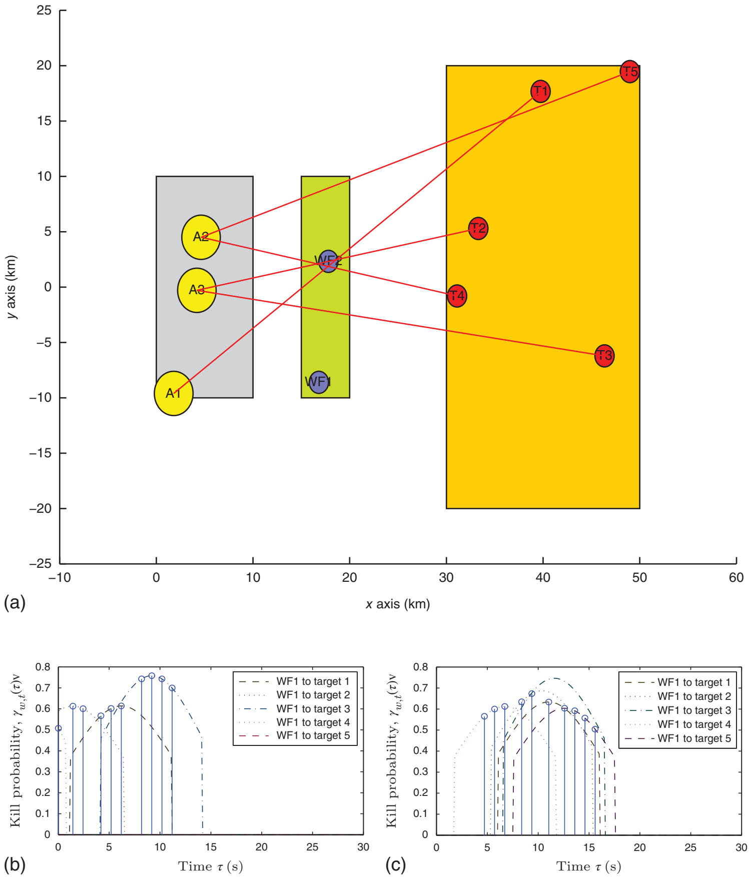


Figure 5.3 Simple example of TSWTA problem, $|A| = 3, |T| = 5, |W| = 2$. (a) Diagram of overall situation. 2D situation with time $\tau = 0$. Left rectangle: asset region, central rectangle: weapon farm region, right rectangle: initial target-searching region. (b) Kill probability of defensive weapons in weapon farm 1 to each target versus time. (c) Kill probability of defensive weapons in weapon farm 2 to each target versus time.

$|A| = 3, |T| = 5,$ and $|W| = 2$ is shown to help understand how the simulation is executed. Overall situation is shown in Figure 5.3. In Figure 5.3a, the position of the assets, weapon farms, and the targets are marked with circles. Each target has a destination asset, and it is indicated by a solid line. The figure describes when the time τ is 0, so the target would move along the line as time goes on. Figure 5.3b and c show the kill probability from weapon farms to targets with respect to time. It is assumed in the simulation that the optimal launching time from the weapon farm to the targets are known in advance, and all of the kill probability graphs follow the trend of a normal distribution function with the optimal launching time as the central axis. The stems in Figure 5.3 indicate the assignment of the defensive weapons in each weapon farm to the appropriate targets by the algorithm proposed earlier.

Values of each parameter used in the simulation are arranged in Table 5.1, and the simulation results are shown in Figures 5.4–5.6. Table 5.1 shows that the simulation is done with several stochastic parameters (initial position $\mathbf{x}$, destruction ratio $p_{t,a}$, kill probability $\gamma_{w,t}(\tau)$, etc.). Even with this random situation, some interesting properties can be found from the simulation results. First, if the same number of defensive weapons are assigned, the case with large number of weapon farms showed the tendency of getting higher $\sum v(a)$ value.

Table 5.1 Parameters for simple TSWTA example.

Parameters		Values
Asset	$\lvert A\rvert$	5, 10, 30
	$\mathbf{x}_a$	$[0, 10] \times [-10, 10](km)$
	$v(a)$	1
Weapon farm	$\lvert W\rvert$	1, 2, 5, 10
	$\lvert M\rvert$	100
	$\mathbf{x}_w$	$[15, 20] \times [-10, 10](km)$
	τ_w^d	1(s)
	Velocity	$[0.9, 1.1]$(km/s)
	max $\gamma_{w,t}(\tau)$	$[0.6, 0.8]$
Target	$\lvert T\rvert$	10, 20, 60
	Initial $\mathbf{x}_t$	$[30, 50] \times [-20, 20](km)$
	Velocity	$[1.5, 2.5](km/s)$
	$p_{t,a}$	$[0.3, 0.7]$

This happens since the defender is more flexible from the constraint of weapon launching time if the number of weapon farm is bigger. There exists a certain time window that the kill probability $\gamma_{w,t}$ is nonzero value for each weapon farm–target pair, and because of the time delay τ_w^d when launching the weapon in weapon farm w, there exists a limitation for the number of weapon allocation even in the case that the weapon farm w has infinitely many weapons. Second, all of the graphs in Figures 5.4–5.6 are monotonically nondecreasing function. This property is proven in Section 5.3. As the number of assigned missiles increases, the objective function value increases smoothly like a step function. This is because the objective function consists of the sum of product collections, which makes the expectation of the asset value decreasing exponentially. It seems that the heuristic algorithm assigns the missiles to the targets that aim to a specific asset, which means the algorithm is trying to protect certain assets with high probability assurance. If the number of missile is not enough to protect all of the assets, some are discarded from being protected.

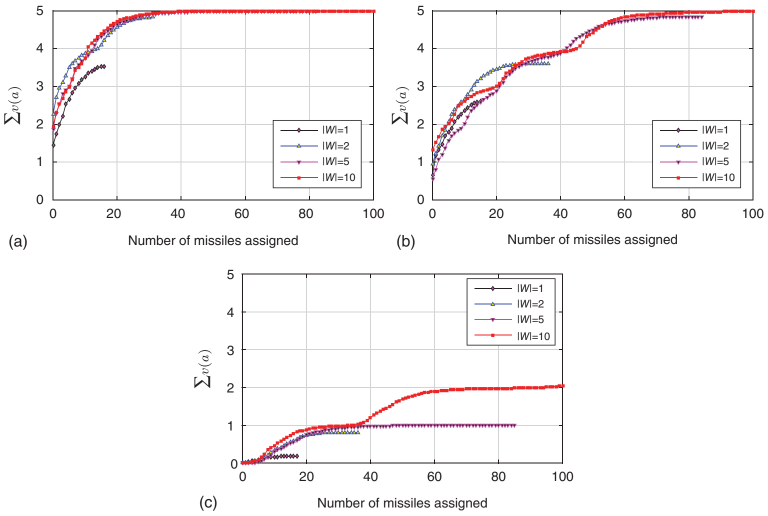


Figure 5.4 Simulation results for various cases. (a) $|A| = 5, |T| = 10$. (b) $|A| = 5, |T| = 20$. (c) $|A| = 5, |T| = 60$.

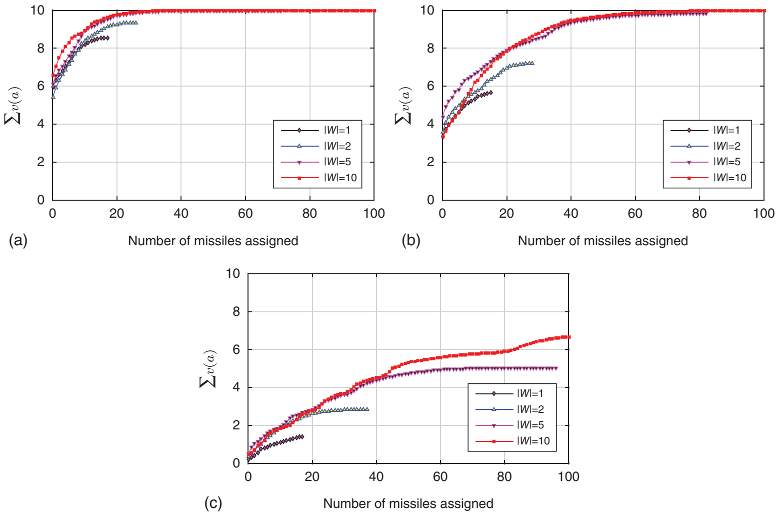


Figure 5.5 Simulation results for various cases.(a) $|A| = 10$, $|T| = 10$. (b) $|A| = 10$, $|T| = 20$. (c) $|A| = 10$, $|T| = 60$.

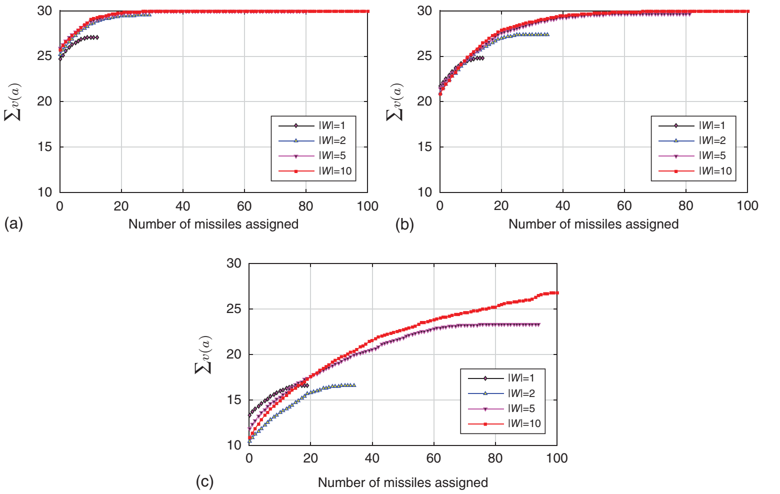


Figure 5.6 Simulation results for various cases. (a) $|A| = 30$, $|T| = 10$. (b) $|A| = 10$, $|T| = 20$. (c) $|A| = 10$, $|T| = 60$.

The computing time took less than few seconds for most of the cases except the number of the targets and the weapon farms became more than a hundred.

5.5.2 Realistic Interceptor-Ballistic Target Assignment

To identify the applicability of the suggested algorithm, a simulation for a real-world example is carried out. A situation is similar to Figure 5.1. Trajectories of targets and location of each target and asset used in the simulation are shown in Figure 5.7a. Detailed assumptions are as follows:

- Battlefield is assumed as on Earth surface: longitude $\in [100^\circ, 120^\circ]$, latitude $\in [30^\circ, 45^\circ]$.
- Targets are assumed as ballistic missiles, and they are launched from the northern region. Target launching time is uniformly distributed in [0,300] (s).
- Targets are launched from four different launching sites; 4, 6, 8, and 10 missiles are launched from each site (totally 28 targets attacking the assets).
- The targets' destination point is known *a priori*, and the destruction ratio, $p_{t,a}$ is in $[0.3, 0.7]$.
- Weapon farms and assets are in the southern region.
- Defense weapons are assumed as guidance missiles and launched from a weapon farm.
- The number of assets, $|A|$, is 6, and each has same value, $v(a) = 1$.
- Number of targets assigned to each asset:
  $a_1 = 14, a_2 = 5, a_3 = 4, a_4 = 3, a_5 = 2, a_6 = 0$
- Weapon farms are located in four different sites. Eight identical weapon launchers exist in each site (totally 32 weapon farms), and each launcher has eight weapons. In other words, each weapon farm site has 64 weapons, totally 256 weapons.
- A kill probability information of a defense weapon for each target is given for the simplicity of the problem, and $\max \gamma_{w,t}(\tau) \in [0.6, 0.8]$.

The simulation results are shown in Figure 5.7. Totally 28 targets are fired, and each weapon has kill probability for each target during the engagement. Eight weapons are given for each weapon farm, and in Figure 5.7b each stem shows the

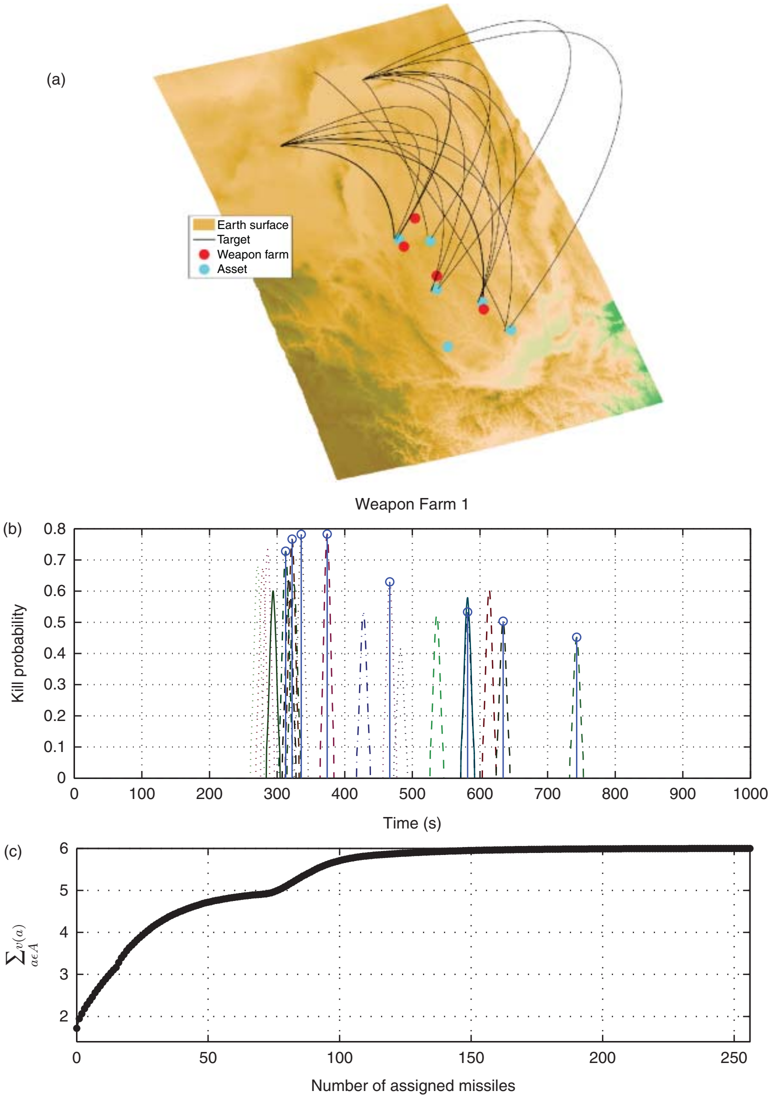


Figure 5.7 Simulation result for the real-world case. (a) Targets are launched from northern region and aiming at assets located in southern region. Weapon farms are near the assets to protect the set of assets. (b) One sample of a weapon farm's assignment result. (c) Number of assigned missiles versus sum of asset values.

assignment of the weapons to the targets. Similarly to the earlier section, Figure 5.7c shows a monotonically nondecreasing property. As the number of the assigned weapon increases, the sum of the expected asset survival value converges to six—which means that all of the assets will be protected from a given threat. The order of the assignment is not shown in the result figures, but it is discovered that in the early stages the weapons are assigned to the set of targets, which aim to asset a_5. a_5 is the asset with minimal threat from enemy targets except for a_6 from the assumption of this simulation. And this phenomenon comes from the characteristic of the suggested algorithm—which assigns the weapon with the maximal marginal gain. From Figure 5.7c, it seems that the user can decide the number of weapons to assign during the engagement.

5.6 Conclusion

In this chapter, the problem called TSWTA problem is defined and analyzed in the way of combinatorial optimization and computational experiments. It is shown that the TSWTA can be formulated as an optimization of a monotonically nondecreasing function over a partition matroid. From the property of nondecreasing function, it was shown that a greedy maximization heuristic algorithm can guarantee at least $\frac{1}{1+\mu}$ optimality approximation, where $\mu = \frac{\alpha^K(1-\alpha^K)}{K(1-\alpha)}$. Numerical experiments validated that the proposed heuristic algorithm gives good performance within several seconds, and the applicability of the suggested algorithm is identified with a simulation for a real-world case. One of the possible future directions is to decentralize the assignment decisions so that each weapon farm agent assigns its resources based on information sharing with neighboring agents (rather than based on the global information).

Acknowledgment

This work was conducted at High-Speed Vehicle Research Center of KAIST with the support of Defense Acquisition Program

Administration (DAPA) and Agency for Defense Development (ADD).

References

1 Eckler, A.R. and Burr, S.A. (1972) Mathematical Models of Target Coverage and Missile Allocation, *Military Operations Research Society*.

2 Matlin, S. (1970) A review of the literature on the missile-allocation problem. *Operations Research*, **18** (2), 334–373.

3 Lloyd, S.P. and Witsenhausen, H.S. (1986) Weapon allocation is NP-complete, in Summer Computer Simulation Conference.

4 Ahuja, R.K., Kumar, A., Jha, K.C., and Orlin, J.B. (2007) Exact and heuristic algorithms for the weapon-target assignment problem. *Operational Research*, **55** (6), 1136–1146.

5 Dorigo, M., Maniezzo, V., and Colorni, A. (1991) Ant system: an autocatalytic optimizing process, Tech. Rep., Dipartimento di Elettronica e Informazione Politecnico di Milano, Piazza Leonardo da Vinci 32 20133 Milano, Italy.

6 Kennedy, J. and Eberhart, R. (1995) *Particle Swarm Optimization, Neural Networks*, vol. **4**, IEEE, Perth, WA, pp. 1942–1948.

7 Glover, F. (1989) Tabu search - Part I. *ORSA Journal on Computing*, **1** (3), 190–206.

8 Glover, F. (1990) Tabu search - Part II. *ORSA Journal on Computing*, **2** (1), 4–32.

9 Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P. (1983) Optimization by simulated annealing. *Science*, **220** (4598), 671–680.

10 Goldberg, D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

11 Lee, Z.-J., Lee, C.Y., and Su, S.F. (2002) An immunity based ant colony optimization algorithm for solving weapon-target assignment problem. *Applied Soft Computing*, **2** (1), 39–47.

12 Lee, Z.J. and Lee, W.L. (2003) A hybrid search algorithm of ant colony optimization and genetic algorithm applied to weapon-target assignment problems. *Intelligent Data Engineering and Automated Learning*, **2690**, 278–285.

13 Bo, Z., Feng-Xing, Z., and Jia-hua, W. (2011) A novel approach to solving weapon-target assignment problem based on hybrid particle swarm optimization algorithm, in Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on, vol. **3**, IEEE, pp. 1385–1387.

14 Wang, S. and Chen, W. (2012) Solving weapon-target assignment problems by cultural particle swarm optimization, in *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2012 4th International Conference on*, vol. **1**, IEEE, pp. 141–144.

15 Blodgett, D.E., Gendreau, M., Guertin, F., Potvin, J.Y., and Séguin, R. (2003) A tabu search heuristic for resource management in naval warfare. *Journal of Heuristics*, **9** (2), 145–169.

16 Xin, B., Chen, J., Zhang, J., Dou, L., and Peng, Z. (2010) Efficient decision makings for dynamic weapon-target assignment by virtual permutation and tabu search heuristics. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, **40** (6), 649–662.

17 Lee, Z.-J., Su, S.F., and Lee, C.Y. (2003) Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics. *Intelligent Data Engineering and Automated Learning*, **33** (1), 113–121.

18 Hong, S.S., Han, M.M., Choi, H.J., and Mun, C.M. (2012) A study of population initialization method to improve a genetic algorithm on the weapon target allocation problem. *Journal of Korean Institute of Intelligent Systems*, **22** (5), 540–548.

19 Tokgöz, A. and Bulkan, S. (2013) Weapon target assignment with combinatorial optimization techniques. *International Journal of Advanced Research in Artificial Intelligence*, **2** (7). 39–50

20 Hosein, P. and Athans, M. (1990) Preferential defense strategies, Part I: The static case, Tech. Rep., Massachusetts Institute of Technology, Cambridge, MA 02139.

21 Hosein, P. and Athans, M. (1990) Preferential defense strategies, Part II: The dynamic case, Tech. Rep., Massachusetts Institute of Technology, Cambridge, MA 02139.

22 Wang, Z., Moran, B., Wang, X., and Pan, Q. (2016) Approximation for maximizing monotone non-decreasing set functions with a greedy method. *Journal of Combinatorial Optimization*, **31** (1), 29–43.

23 Cormen, T.H. (2009) *Introduction to Algorithms*, MIT Press.

6

Coordinated Threat Assignments and Mission Management of Unmanned Aerial Vehicles

Eloy Garcia and David Casbeer

*The Control Science Center of Excellence, Air Force Research Laboratory,
Wright-Patterson AFB, OH, USA*

6.1 Introduction

Autonomous decentralized decision making and path planning for a group of unmanned aerial vehicles (UAVs) in contested environments represents a challenging problem in cooperative control. This type of problem involves a coordinated choice of actions such as which targets to attack and by which UAV while minimizing the use of resources that include fuel, weapon inventory (for survivability in uncertain environments), and transmission of information. Additional constraints need to be satisfied such as avoidance of certain areas, selection of trajectories that minimize radar exposure, and plan paths that satisfy vehicle dynamics and constraints.

Several authors have addressed related problems in different ways and under different assumptions. The complexity of the problem related to coordination of vehicles and uncertain number of threats, their locations, and capabilities makes this a challenging problem. In [1] coordinated path planning for multiple vehicles in an environment containing multiple radar sites was studied. The main UAV objectives in that reference are the minimization of radar exposure and the simultaneous arrival to the main target by all UAVs. A similar problem was described

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

in [2]. In this case, there exist several targets, and UAVs need to select the target to visit according to individual and group criteria. Several algorithms for group path planning in uncertain environments were presented in [3] where the authors consider the probabilities of UAV loss and the effects on other UAV tasks. The paper [4] proposes a decision policy for a group of combat vehicles maneuvering in uncertain urban environments. The unmanned vehicles need to follow some of the predefined paths as given by the urban terrain and arrive simultaneously to the main target location. In their way to the main target, vehicles face adversarial ground units and need to decide the paths to follow, which ground units to attack, and how to subdivide the group agents into subgroups or formations. Other references discussing path planning for multiple vehicles operating in adversarial environments include [5–10].

The wide area search munitions (WASM) problem is a special type of task assignment problem in the sense that several tasks need to be performed on the same target and in strict sequential order [11–16]. The UAVs are required to perform different tasks on stationary targets with known locations. The tasks for each one of the targets include classify, attack, and verify the destruction of such targets. These tasks need to be performed in sequential manner for any given target and with relative timing constraints resulting in coupled assignments of vehicles performing tasks on the same target. The authors of [17] also addressed the WASM problem where two classes of vehicles were considered: target recognition vehicles and attack vehicles. Target recognition vehicles have better sensor capabilities than attack vehicles and they are able to perform classification and verification tasks. On the other hand, attack vehicles are able to engage and strike the targets. The additional task of searching is included and it is assumed that both classes of vehicles are capable of performing this task. The papers [18] and [19] proposed a load-balancing method for the WASM problem that accounts for uncertainties in the environment and on the communication between agents.

The work in this chapter extends the results on [1] and [2] in order to select the best threats to eliminate in order to obtain improved paths. Threat selection was first discussed in [20] where the existence of only one main target was addressed. In this chapter, we propose a distributed approach for selection of threats to eliminate when more than one target is present. This requires a joint optimal selection of main target to reach and associated threats to eliminate. A decentralized approach for multiple UAVs is proposed where agents bid and agree on assignments and recalculate new paths and costs based on assigned threats to other members of the team.

Decentralized assignment of tasks represents a challenging problem in cooperative control and an interesting application of UAVs. For instance, the authors of [21] presented a robust task assignment algorithm for uncertain environments that reduces communication between vehicles. In [22], an algorithm for the assignment of targets to visit each vehicle in a group of vehicles was proposed. Each vehicle turning constraints were explicitly considered in order to determine the flyable paths for each agent. In a related problem, the work in [23] address the cooperative searching and coverage of polygon areas by a team of heterogeneous UAVs.

In [24], the consensus-based bundle algorithm (CBBA) [25] is extended to address replanning in dynamic environments and considering agents with different capabilities. The authors of [26] used a similar approach, the asynchronous CBBA (ACBBA) for agents that communicate over an asynchronous channel. The CBBA has also been extended to consider coupled constraints and different types of temporal constraints [27].

The chapter is organized as follows. The problem details are presented in Section 6.2. Our approach for designing individual optimal paths and for determining optimal threats to attack is presented in Section 6.3. This section also considers decentralized assignment of tasks. Two types of constraints related to the decentralized selection of threats are described in Section 6.4. In Section 6.5, we consider the case in which there exist multiple main targets; however, each UAV is only able to visit one of those main targets. Then, it is necessary for the UAVs to not only select the optimal threats to attack but also to determine the optimal main target to visit. Relevant conclusions are presented in Section 6.6.

6.2 Problem Statement

6.2.1 Preliminaries

For a team of N agents, the communication among them can be described by a directed graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{1, \ldots, N\}$ denotes the agent set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the edge set. An edge (i, j) in the set $\mathcal{E}$ denotes that agent j can obtain information from agent i, but not necessarily vice versa. For an edge $(i, j) \in \mathcal{E}$, agent i is a neighbor of agent j. The set $\mathcal{N}_j$ is called the set of neighbors of agent j, and N_j is its cardinality. A directed path from agent i to agent j is a sequence of edges in a directed graph of the form $(i, p_1), (p_1, p_2), \ldots, (p_{\kappa-1}, p_\kappa)(p_\kappa, j)$, where $p_\ell \in \mathcal{V}, \forall \ell = 1, \ldots, \kappa$. A directed graph is *strongly connected* if there is a directed path from every agent to every other agent. A directed graph has *a directed spanning tree* if there exists at least one agent with directed paths to all other agents.

The adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$ of a directed graph $\mathcal{G}$ is defined by $a_{ij} = 1$ if $(j, i) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. The Laplacian matrix $\mathcal{L}$ of $\mathcal{G}$ is defined as $\mathcal{L} = \mathcal{D} - \mathcal{A}$, where $\mathcal{D}$ represents the degree matrix, which is a diagonal matrix with entries $d_{ii} = \sum_{j \in \mathcal{N}_i} a_{ij}$. If a directed graph has a directed spanning tree, then the corresponding Laplacian matrix has only one eigenvalue equal to zero, $\lambda_1 = 0$, and the remaining eigenvalues λ_i, $i = 2, \ldots, N$, have positive real parts.

6.2.2 Mission Description

Consider a group of vehicles

$$\mathcal{N} = \{1, 2, \ldots, N\} \tag{6.1}$$

and a set of targets

$$\mathcal{M} = \{1, 2, \ldots, M\}. \tag{6.2}$$

In addition, let us consider a set of threats (radar sites)

$$\mathcal{T} = \{1, 2, \ldots, T\}. \tag{6.3}$$

Typically, we have that $T \gg M$. The number of threats and their location are obtained from a priori information such as information resulting from a surveillance stage or measurements and threat detections by the N UAVs when first approaching the adversarial area.

The approach we follow in order to minimize radar exposure is similar to [1] and [2]. In this approach, a Voronoi diagram is constructed based on the region of interest and the threat locations. The construction of the Voronoi diagram involves the partition of the region containing the targets and the threats into $T + M$ convex polygons or cells. Each cell of the Voronoi diagram contains exactly one threat or one target, and every location inside a particular cell is closer to the corresponding threat or target than to any other of these elements. The choice of traveling along edges of the Voronoi diagram provides minimum threat exposure by maximizing the distance of the UAV with respect to every threat.

In this work, we include the targets when constructing the Voronoi diagram because the targets need to be visited at their exact location. The choice of including the main targets helps to easily find a small set of possible paths to reach a main target from the closest vertices of the diagram and it also helps to avoid traveling close to a threat while approaching a main target. Each UAV is assumed to initially have a limited supply of two types of munitions. Munitions of type I are used against threats. They can be engaged from a certain distance and cause damage to relatively small assets. Their use against threats is justified since the threats represent radar sites of small dimensions. Munitions of type II are used against main targets. They provide greater damage but their use is limited by positioning the UAV at the exact target location. Main targets represent valuable assets that in general can extend over larger areas. In addition, the elimination of main targets is of higher priority than that of threats, and then the use of munitions inflicting greater damage is needed in this case.

In the rest of the chapter, we consider the case in which each UAV has resources of type II to attack only one main target; therefore, each UAV is constrained to visit only one main target. The attack of one target by one UAV does not guarantee total elimination of the target. Thus, more UAVs can be assigned to visit the same target in order to increase the damage inflicted on the target. Elimination of main targets represents one of the main goals of the mission; the other being the minimization of threat risk or threat exposure by each UAV while performing the mission.

In the presence of multiple targets, each UAV has to choose the best target to visit according to local and global criteria. This case is discussed in more detail in Section 6.5. For now, we consider the particular case where only one main target exists. This case helps to ease the presentation of the distributed threat assignment method and to highlight its results in terms of reduction of threat risk for the group of UAVs.

Figure 6.1 shows an example of a Voronoi diagram. The scenario in this figure includes 1 UAV, 1 target, and 19 threats. In the following, we describe the steps to construct our specific Voronoi diagram.

1) We include artificial, regularly spaced, threats around the main target. Each artificial threat lies on a virtual circle that contains the domain of interest, that is, the circle surrounds all threats and the target as well. The center of the circle is the location of the main target and its radius, r_a, is greater

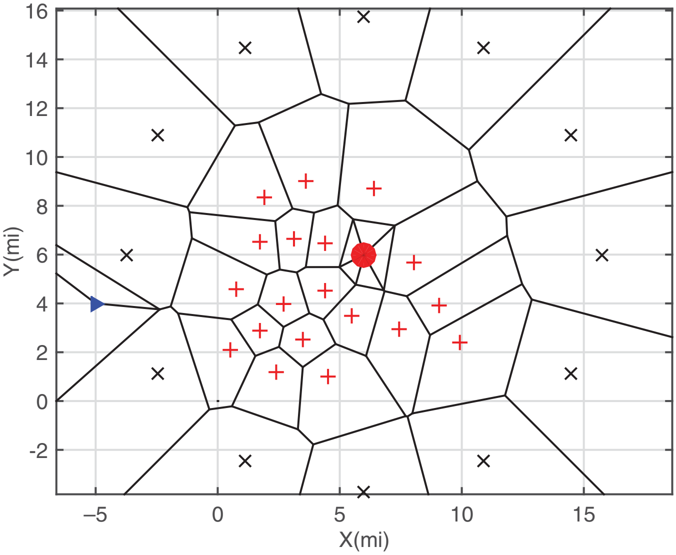


Figure 6.1 Voronoi diagram. Main target: •. threats: +. artificial threats: ×. UAV: ▷.

than the maximum distance from the target to any threat, that is,

$$r_a > \max_j d_{m,j}, \tag{6.4}$$

where $d_{m,j}$ represents the distance from the main target to threat $j \in \mathcal{T}$. The artificial threats are included in order to generate paths around threats, that is, to enclose real threats in convex polygons that consist only of finite vertices. Otherwise, some threats will be inside cells that have vertices at infinity, which severely limits the choice of possible paths. Because the artificial threats do not represent any risk, they are only used in motion planning by the UAVs but are not considered when each UAV computes the costs associated to threats. As we mentioned earlier, we view the main targets as threats when constructing the initial Voronoi diagram.

2) New edges are added from the target's closest vertices (vertices corresponding to the target polygon) to the exact target location. One of these edges provides the final edge of the UAV optimal path, joining the path along edges to the exact location of the final target. These edges are included because main targets need to be visited at their exact location.
3) Each UAV adds one more artificial threat at its current location. New edges are added from the current UAV location to the finite vertices of its surrounding cell. This step provides a simple approach for the UAV to join edges of the Voronoi diagram at safe locations and to continue traveling along these edges while avoiding to come close to a threat during the process.

All edges are considered to be undirected (they can be traveled in both directions) except for those connecting the initial UAV position to the vertices of its Voronoi cell and the edges connecting the target cell vertices to the target location.

The Voronoi diagram can be modified by eliminating certain threats. The costs associated with distance to be traveled and with threat risks can be significantly decreased by carefully selecting the threats to be eliminated. This selection process is thoroughly discussed in the following section.

Finally, once each UAV finds its optimal path to reach the main target, the last step is to obtain a smooth trajectory that can be used by fixed-wing aircraft with minimum turning radius constraint. The Dubins vehicle model is a simple representation, yet, it captures important constraints associated to aircraft. Each UAV dynamics is given by

$$\begin{aligned}\dot{x}_i &= v_i \cos\theta_i \\ y_i &= v_i \sin\theta_i \\ \dot{\theta}_i &= \Omega_i u_i,\end{aligned} \tag{6.5}$$

where v_i and Ω_i represent the speed and the maximum turning rate of vehicle i, and $u_i \in \{-1, 0, 1\}$ for $i \in \mathcal{N}$.

We assume that each vehicle travels at constant speed. Also, for simplicity, we assume that all UAVs have the same speed and the same maximum turning rate, that is, $v_i = v$ and $\Omega_i = \Omega$ for $i \in \mathcal{N}$.

Because the optimal paths are given by a subset of edges of the Voronoi diagram, it is necessary to determine associated flyable trajectories that satisfy the constraints (6.5). We implement the method presented in [2] to generate the flyable trajectories based on Voronoi diagram edges.

6.3 Decentralized Assignment of Threats

The focus of this section is, first, on determining the UAVs' individual optimal paths and preferred threats and, then, on achieving global agreements on the threats to be eliminated based on a decentralized consensus algorithm.

6.3.1 Optimal Individual Paths and Selections

Two types of costs are associated to each one of the edges of the Voronoi diagram constructed following steps 1–3 described earlier: length cost and threat cost. The length cost of edge g, denoted by $\Phi_{\text{length},g}$, is simply the length of the edge

$$\Phi_{\text{length},g} = L_g. \tag{6.6}$$

The threat cost associated to a given edge g, which we denote as $\Phi_{\text{threat},g}$, is calculated based on radar exposure of a vehicle traveling along that edge. To simplify the calculation of radar exposure, it is assumed that the vehicle's radar signature is uniform in all directions and it is proportional to $1/d^4$, where d represents the distance from the vehicle to the threat. In order to accurately calculate the threat cost along a given edge, the integration of the cost along the edge is required. Instead of integrating costs for every edge we follow the much more computationally affordable three-point approximation [2], which is given by

$$\Phi_{\text{threat},g} = \frac{\alpha L_g}{3} \sum_{j=1}^{T} \left(\frac{1}{d^4_{1/6,g,j}} + \frac{1}{d^4_{1/2,g,j}} + \frac{1}{d^4_{5/6,g,j}} \right). \tag{6.7}$$

In this approximation, three points along edge g are used. The points are $L_g/6$, $L_g/2$, and $5L_g/6$. The expressions $d_{1/6,g,j}$, $d_{1/2,g,j}$, and $d_{5/6,g,j}$ represent, respectively, the distance from each one of the selected three points on edge g to threat j; $\alpha > 0$ is a constant scale factor.

The total cost for traveling along edge g is given by a weighted sum of its length and threat costs

$$\Phi_g = \kappa \Phi_{\text{length},g} + (1 - \kappa)\Phi_{\text{threat},g}, \tag{6.8}$$

where $\kappa \in [0, 1]$ provides additional flexibility in weighting more the exposure to threats or the distance to travel. The total cost of a path $\mathcal{P} = \{g_1, g_2, \dots\}$ selected by agent i is represented by J_i and it is given by the sum of the total costs of the edges $\{g_1, g_2, \dots\}$ making up that path

$$J_i = \sum_{g \in \mathcal{P}} \Phi_g. \tag{6.9}$$

Let $J_i^*(0)$ represent the cost of the optimal path of agent i when the agent is not yet tasked to eliminate any threat. Also, let $J_i^*(j)$ represent the cost of the optimal path of agent i if the agent plans to eliminate threat j, for $j \in \mathcal{T}$. Similarly, $J_i^*(j_1, j_2)$ is the cost of the optimal path when agent i plans to eliminate both threats j_1 and j_2. The notation can be extended in a similar way to include plans with more than two threats that are planned to be eliminated.

Following the Voronoi diagram method, the search of optimal paths is transformed into a search over a discretized space of possible solutions, which can be implemented for online computations and also for replanning assignments during mission execution. The approach in [1] and [2] is extended in this chapter by allowing agents to attack and destroy a limited number of threats. A new Voronoi diagram is obtained by eliminating one or more threats. It is clear that a good selection of threats to destroy brings a significant benefit from both points of view: reducing length costs and reducing threat costs. Suppose a given agent can spend resources to eliminate up to T_i threats. Then, the individual optimization problem that each UAV tries to solve can be expressed as follows:

$$\mathcal{J}_i^*(T_i^*) = \min_{j_1, j_2, \ldots, j_{T_i}} J_i^*(j_1, j_2, \ldots, j_{T_i}) \tag{6.10}$$

subject to the edges created by the associated Voronoi diagrams, where the notation $J_i^*(j_1, j_2, \ldots, j_{T_i})$ represents the estimated minimum total cost that results by eliminating the current selection of threats $j_1, j_2, \ldots, j_{T_i}$. Note that this is a two-step optimization process in which each UAV searches for the best combination of T_i threats (that results in the optimal cost $\mathcal{J}_i^*(T_i^*)$) to destroy which, in turn, is based on the optimal costs (paths) of each individual combination $J_i^*(j_1, j_2, \ldots, j_{T_i})$.

The estimation of the new total cost when one or more threats are scheduled to be attacked is performed by using the edges of the new Voronoi diagram assuming those threats are eliminated and only including threat costs associated to the remaining threats. The solution of problem (6.10) provides the optimal selection of threats that the particular UAV should attack in order to minimize its own total cost subject to the conditions imposed by the Voronoi partitions and the limited number of resources it is able to use to attack threats.

6.3.2 Decentralized Assignment Algorithm

An additional extension with respect to the work in [1] and [2] is given by the implementation of a decentralized communication scheme in order for agents to agree on the set of assignments. In this work, we do not restrict the agents to form a complete communication graph in which every agent can communicate with every other agent. We relax this condition, and agents can form a directed and strongly connected communication graph. The task assignment algorithm used in this section is based on the consensus-based auction algorithm (CBAA) [25]. Because of the timing and coupling constraints, which are explained later, we choose to execute this algorithm in a sequential planning manner where agents select only one threat to bid at a time

($T_i = 1$) and only one assignment is made, that is, there is only one winner after a single execution of the algorithm. Then, the costs are updated and the algorithm is executed again until all vehicles have their corresponding number of assignments. This means that, although each UAV has resources to eliminate more than one threat, the agents do not try to choose their preferred threats all at once. Instead, they only search for one threat assignment; then, they bid and agree on some threat assignments. After every bidding process, all agents that still do not have a complete list of individual assignments will search again for a new threat to add to its list. The difference with respect to the previous bidding process is that one of the threats has been assigned to the group; hence, that threat is not considered by any UAV in the new search. New Voronoi diagrams are obtained after every assignment, which means that the preferences of any UAV are not necessarily the same after every bidding process regardless of which agent wins the assignments.

In this section, we present a modification of the CBAA [25], and similar to [28], in order to make this series of assignments and to obtain new costs for the remaining agents. Two variables are defined for each agent: $x_i(\lambda)$, which represents the current selected threat for agent i and $y_i(\lambda)$, which represents the associated bid to the current selected threat, where λ indexes the iterations within the consensus algorithm. Define the vector $\chi_i(\lambda) = [x_i(\lambda)\ y_i(\lambda)\ \tau_i(\lambda)]^T$, where $\tau_i(\lambda)$ is defined in Section 6.4.

Initialization. Each UAV uses Dijkstra's algorithm [29] to compute the best path based on the edge costs (6.8). All edges are considered to be undirected (they can be traveled in both directions) except for those connecting the initial UAV position to the vertices of its Voronoi cell and the edges connecting the target cell vertices to the target location. Each UAV obtains the best path for each threat option, $J_i^*(j)$ for $j \in \mathcal{T}(t)$, that is, Dijkstra's algorithm is used once for every available threat $j \in \mathcal{T}(t)$. Then, the UAV selects the best threat to attack based on the costs $J_i^*(j)$, which returns the optimal cost $\mathcal{J}_i^*(1^*)$. The current set of tasks $\mathcal{T}(t)$ varies as the threats are assigned. Then, every time that the UAVs find and transmit bids, the variable x_i is initialized as follows:

$$x_i(0) = \arg\min_j J_i^*(j; \mathcal{T}(t)). \tag{6.11}$$

For consistency, agents bid their best normalized gain in performance

$$y_i(0) = \frac{J_i^*(0;\mathcal{T}(t)) - \mathcal{J}_i^*(1^*;\mathcal{T}(t))}{J_i^*(0;\mathcal{T}(t))}. \tag{6.12}$$

The notation for the calculated costs has been expanded in order to clarify the dependence on the current set of available threats to bid $\mathcal{T}(t)$. $J_i^*(0;\mathcal{T}(t))$ represents the optimal cost when no threat is selected to be attacked at the current stage, that is, the optimal cost considering all remaining threats $j \in \mathcal{T}(t)$, and $\mathcal{J}_i^*(1^*;\mathcal{T}(t))$ is the optimal cost obtained by choosing the best single threat to attack from the available set $\mathcal{T}(t)$. $y_i(0)$ is the bid that agent i places on its preferred threat to destroy which has been stored in $x_i(0)$ and it represents a measure of how much the optimal cost can be reduced by using resources to eliminate that threat and according to the current conditions and previous assignments.

Consensus process. The agents use the variables $y_i(\lambda)$ in order to converge to the best possible value $x_i(\lambda)$. In this phase, the agents receive the vectors $\chi_i(\lambda)$ from their neighbors, that is, from agent i' such that $i' \in \mathcal{N}_i$. Note that self-loops are used, that is, $a_{ii} = 1$ for $i \in \mathcal{N}$, which means that $i \in \mathcal{N}_i$ for $i \in \mathcal{N}$. Then, each agent i selects the best cost available. The best cost among agents is selected according to the best normalized performance gain, that is,

$$y_i(\lambda + 1) = \max_{i' \in \mathcal{N}_i}\{y_{i'}(\lambda)\} \tag{6.13}$$

and the new threat choice is updated according to the winning bid,

$$x_i(\lambda + 1) = x_\nu(\lambda), \tag{6.14}$$

where

$$\nu \in \arg \max_{i' \in \mathcal{N}_i}\{y_{i'}(\lambda)\}. \tag{6.15}$$

It is unlikely that two or more agents will initially obtain the same exact bid for their preferred threats to destroy at the beginning of a given bidding process. This is so because the agents are located at different points and the individual optimal costs are heavily dependent on the UAV's current position. However, possible ties in selection of threats can be broken in different ways such as using the vehicles identity.

The main difference between the CBAA in [25] and our proposed algorithm is that our protocol makes all agents agree on a single task instead of forcing them to take on different tasks each one. This algorithm guarantees convergence to a single (same for all agents) value and the convergence time is bounded as follows:

Theorem 6.3.1 *[25]* Consider a group of agents (6.1) that communicate using a directed and strongly connected graph and define $d_{ii'}$ as the shortest path between agents i and i' for $i, i' \in \mathcal{N}$. Let $D = \max_{ii'}\{d_{ii'}\}$ be the diameter of the graph. Then all agents converge to the same value at time $\lambda = \lambda_c$, where λ_c satisfies

$$\lambda_c \leq 1 + D. \tag{6.16}$$

The execution of the consensus protocol described earlier guarantees only one assigned threat and knowledge by every UAV of which threat has been assigned. After convergence of the consensus process each agent compares its initial bid (6.12) to the winning bid $y_i(\lambda_c)$ to decide if it is the current winner of the assignment. If the local agent is not the winner, then it recomputes new costs with an updated set of threats: $\mathcal{T} = \mathcal{T} - \{x_i(\lambda_c)\}$. If the local agent is the winner, it adds the new assignment to its list of tasks. In addition, if the local agent still has resources available to add more tasks to its list, that is, it still does not have T_i threats assigned to itself, then, it updates the set of remaining threats and recomputes new costs according to the new threat set.

Note that when an agent has won its corresponding number of assignments then it does not bid on new assignments but it collaborates as "communication link" during the consensus process in order to keep a connected graph at every assignment process.

6.4 Assignment Constraints

One of the main effects of multiple UAVs eliminating different threats is that costs for all other UAVs change when one UAV eliminates or it wins an assignment, and it is scheduled to eliminate a given threat. This is a very important feature of cooperation of multiple agents such that decisions by a given agent can be improved by considering threats already assigned to other agents. An agent can recompute the costs associated to a Voronoi diagram that is constructed assuming assigned threats are eliminated and obtain improved decisions on its own. Two particular problems arise in the analysis of this decentralized assignment problem: timing constraints and coupling constraints.

6.4.1 Timing Constraints

Knowledge of assigned threats can be used to obtain better decisions. However, the assigned threat is not eliminated immediately as it is assigned. If another vehicle decides to travel along a path where that previously assigned threat is relatively near and has not been destroyed yet, then the real threat cost of that path is significantly higher than the estimated cost assuming the threat has been eliminated. In this section, the timing constraint just described is dealt with by estimating and transmitting the time of attack of the assigned UAV. This information is used by other UAVs to obtain more accurate threat costs when updating paths after an assignment has been agreed by all UAVs. The time of attack is estimated by assuming the selected threat can be engaged when the UAV reaches a certain position relative to the threat.

Each UAV attaches the estimated time of attack $\tau_i(0)$ to its selected threat to bid $x_i(0)$. $\tau_i(\lambda)$ corresponds to the third element of the broadcasting vector $\chi_i(\lambda)$. This estimated variable remains attached, so the winning bid has its associated time of attack and every agent that does not win an assignment in the current execution has access to this information when recomputing new costs.

During the consensus process, $\tau_i(\lambda)$ is updated as follows:

$$\tau_i(\lambda + 1) = \tau_v(\lambda), \tag{6.17}$$

where v was defined in (6.15).

All UAVs that did not win an assignment after the first or subsequent executions of the consensus problem recompute new paths and costs using a Voronoi diagram that now contains time-varying threats. Since the computation of threat costs along any edge is time independent (6.7), it is computationally expensive to compute different costs for every edge according to the time in which is traveled in every possible path. Instead, each agent implements a simple approach, which is based on first finding the k best (shortest) paths assuming the already assigned threats are nonexistent, then adding the additional threat costs associated to each assigned threat (just before its time of attack) to every one of the k best paths, and then selecting the best updated path. A generalization of Dijkstra's algorithm [29] to find the k shortest paths is implemented in this case.

Lemma 6.4.1 Let $H = \{1, 2, \ldots, k\}$ be the ordered set of the k best paths for agent i without considering the effect of assigned threats. Let $C = \{c_1, c_2, \ldots, c_k\}$ be the set of associated path costs with $c_1 < c_2 < \cdots < c_k$. Let $D = \{d_1, d_2, \ldots, d_k\}$ be the corresponding set of new path costs when the assigned threats are considered according to their time of attack. Let

$$\hat{d} = \min_h \{d_h\} \tag{6.18}$$

for $h \in H$. Then, there exist an integer $k > 1$ such that

$$\hat{d} < c_k \tag{6.19}$$

and

$$h^* = \arg\min_h \{d_h\} \tag{6.20}$$

for $h \in H$ is the optimal path for agent i subject to the current Voronoi partition and the already assigned threats with their corresponding times of attack. The corresponding optimal cost is $d^* = \hat{d}$.

Proof: First, we show that $d_h \geq c_h$ for any $h \in H$. In order to obtain d_h, a non-negative threat cost is added to c_h. The added threat cost is equal to zero only when the assigned threats are scheduled to be immediately attacked at the time of the assignment process. The length costs remain unchanged.

The initial costs c_h are obtained in a similar way to J_i in (6.9) but without considering the threat cost associated to already assigned threats since these are time-varying costs. From (6.8) the new path costs d_h are only different than the path costs c_h by adding the time-dependent threat costs as follows:

$$d_h = c_h + \sum_{g \in h} \phi_{h,g}, \tag{6.21}$$

where

$$\phi_{h,g} = (1 - \kappa)\frac{\alpha L_g}{3} \sum_{j=1}^{P_a} \left(\frac{\rho_{1/6,g}(j,t)}{d^4_{1/6,g,j}} + \frac{\rho_{1/2,g}(j,t)}{d^4_{1/2,g,j}} + \frac{\rho_{5/6,g}(j,t)}{d^4_{5/6,g,j}} \right), \tag{6.22}$$

P_a is the number of assigned threats and $\rho_{1/6,g}(j,t) = 1$ if the estimated time when agent i reaches the point 1/6 along edge g following path h is less than the estimated time of attack corresponding to threat j, and 0 otherwise. The cost $\phi_{h,g} \geq 0$ represents the partial threat cost of edge g associated to the assigned threats. Every edge g making up the path h is considered and the resulting summation is added to c_h as shown in (6.21).

Now, suppose (6.19) holds, then we have that $\hat{d} < c_{k+\mu} \leq d_{k+\mu}$ for $\mu = 0, 1, 2, \ldots$, which means that there is no other path with smaller cost than h^* and the optimal cost is $d^* = \hat{d}$. It is easy to see that for the case of large T (large number of remaining threats) there exists an infinite number of increasing c_h cost paths by construction of the Voronoi diagram following steps 1–3 in Section 6.2.2. The existence of a k such that (6.19) holds is guaranteed even for the case when all threats have been assigned ($T = 0$) and the main targets remain. The addition of artificial threats (step 1 in Section 6.2.2), which are never removed when computing new Voronoi diagrams, provides infinite number of paths around the target and assigned threats (possibly encircling the target many times waiting until assigned threats are eliminated) with increasing c_h cost. Therefore, it is always possible to find the integer k such that (6.19) holds. ■

Lemma 6.4.1 suggests a smart way to find the best path considering the effects of assigned threats. Algorithm 1 describes a method to quickly find the optimal path by limiting the number of new costs d_h that need to be computed. Given a set H of k best paths with associated costs C, the algorithm keeps track of the current minimum new cost $\hat{d}$ and checks condition in Lemma 6.4.1 to decide if further new costs are unnecessary in order to terminate the search since the optimal path has been obtained. Note that h^* is the optimal path with respect to the individual choice of threat to attack and the previous decisions, which are the already assigned threats. Future assignments may change the Voronoi diagram, the length, and the threat costs. This coupling effect between past, current, and future assignments is discussed in Section 6.4.2.

Algorithm 6.1 *Find_best_path (H,C)*

```
$\hat{d} = \infty$
**while** $h \leq k$ **do**
    $d_h = c_h + \sum_{g \in h} \phi_{h,g}$
    $\hat{d}=\min\left\{\hat{d}, d_h\right\}$
    $\hat{h}=\arg\min\left\{\hat{d}, d_h\right\}$
    **if** $\hat{d} < c_{h+1}$ **then**
        $d^* = \hat{d}$
        $h^* = \hat{h}$
        $h = k + 1$
    **else**
        $h = h + 1$
    **end if**
**end while**
```

Remark 6.4.2 A different method to find the optimal path and cost in which the time-varying threat costs are subtracted instead of added, as it was done in Lemma 6.4.1, can also be implemented. In this case, the initial costs correspond to paths obtained based on the original Voronoi diagram containing all threats (assigned and remaining) and considering the threat costs associated to assigned threats. Then, partial threat costs are subtracted according to the assigned threats' times of attack. However, this approach is more conservative since the paths created from eliminating threats are not considered, which potentially avoids good path choices. The approach used in this section is riskier by allowing paths that may pass near assigned threats when they still exist. This risky paths are dealt with using Algorithm 1, which considers the timing constraints for every candidate path and disregards those that are deemed too risky.

Remark 6.4.3 The velocity for each UAV is assumed to be constant, although it can be different for every vehicle. In this case, there are many alternatives for a UAV after one or more assignments have been made. For instance, if the best initial path with cost c_1 is found to be highly risky because d_1 increases by a high amount with respect to c_1, then a choice may be to fly to a different area to maneuver far from the previous assignments where the partial costs (6.22) are very small. Another choice may be to use a similar path to the one with cost c_1 but with added edges at the beginning in order to make time and use the similar path when the assigned threats have already been eliminated and do not add a significant cost to that modified path. The best selection from all possible choices is given by Algorithm 1 in terms of length, threat, and time-varying threat costs. If the assumption is relaxed such that velocity can be adjusted and/or loiter is permitted then many other choices can be allowed. However, the problem is further complicated not only by analyzing new variables but also the effects of reducing speed and loitering with respect to calculation of threat costs need to be analyzed.

6.4.2 Coupled Decision Making

The decentralized assignment of threats in this particular problem creates different paths and associated costs as each agent selects a threat to eliminate. An (centralized) optimal set of decisions can be found by an exhaustive search of all possible combinations assuming that a central unit is able to obtain all the required information (locations of threats and targets and current locations of UAVs), and it is also able to communicate the decisions to the entire fleet. Since every agent makes decisions on a limited set of information and it is not able to directly communicate to every other agent then an overall optimal solution is difficult to reach. A preliminary approach that offers good results in terms of cooperative behavior is based on sequential assignments where only agents without

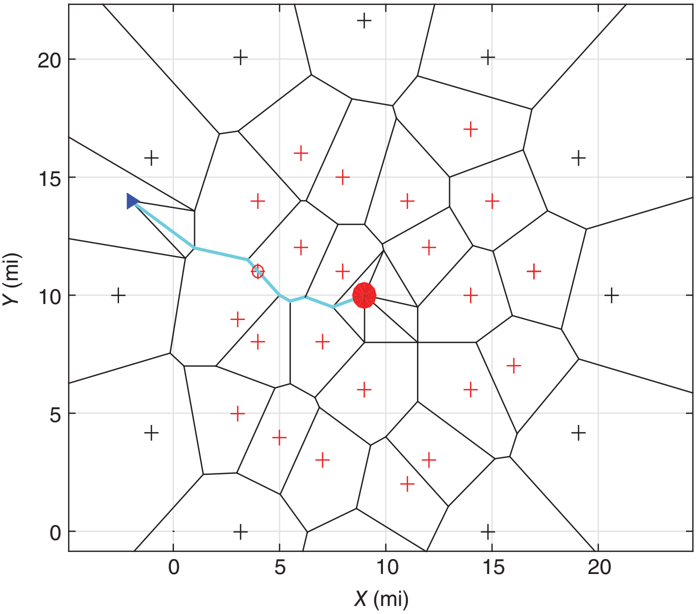


Figure 6.2 Assignment 1: UAV 1.

their complete number of assigned tasks recompute their paths and costs and bid again on remaining threats.

Figures 6.2–6.7 show an example of the sequential assignment of one threat to each one of three UAVs. Three total assignments are made. Some of these figures show unsuccessful bids. The scenario consists of 3 UAVs, 1 main target, and 23 threats. The vehicles' speed and maximum turning rate are, respectively, $v = 240$ ft/s and $\Omega = 0.046$ rad/s. The communication graph is given by $a_{12} = a_{21} = a_{23} = a_{32} = 1$, and the remaining elements of the adjacency graph are equal to zero. The individual computation of optimal paths involves the calculation of length and threat costs where the parameters $\kappa = 0.45$ and $\alpha = 1$ are used by every UAV. Each assignment is resolved by following the consensus algorithm described in Section 6.3.2. After one or more threats have been assigned, the threat costs and optimal paths are calculated based on Algorithm 1.

Figures 6.2–6.4 show the first computation of optimal threat to attack by each UAV and the corresponding optimal path.

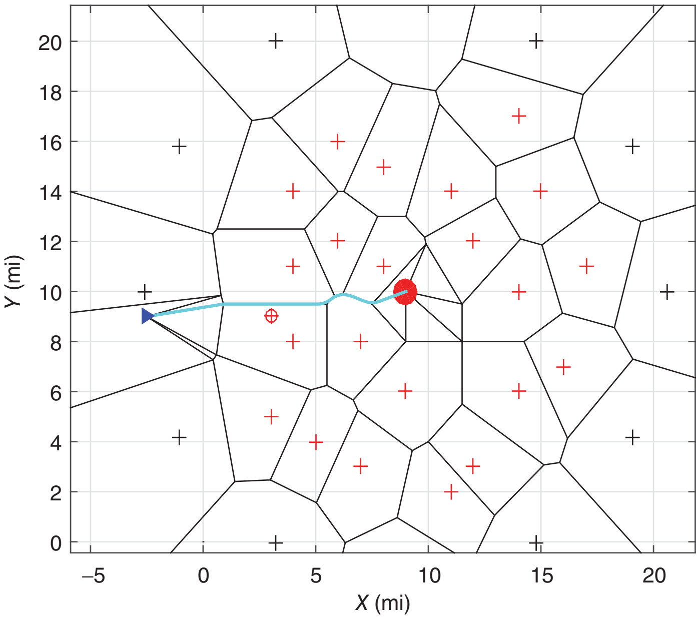


Figure 6.3 Assignment 1: UAV 2.

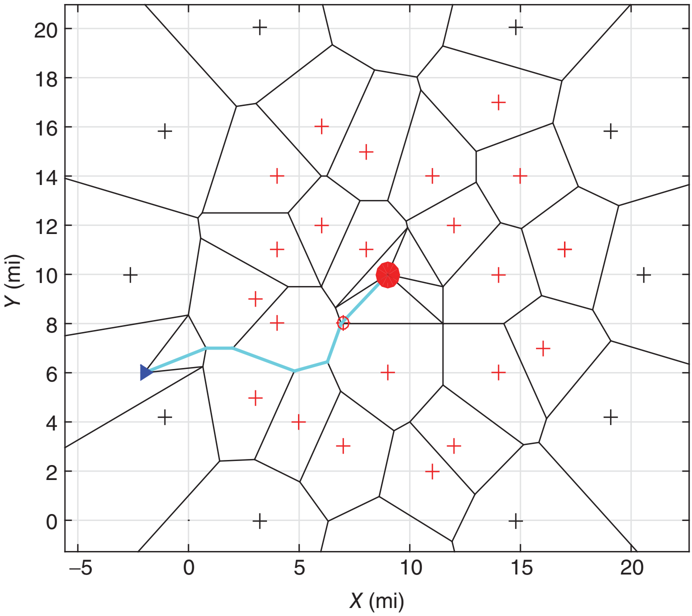


Figure 6.4 Assignment 1: UAV 3.

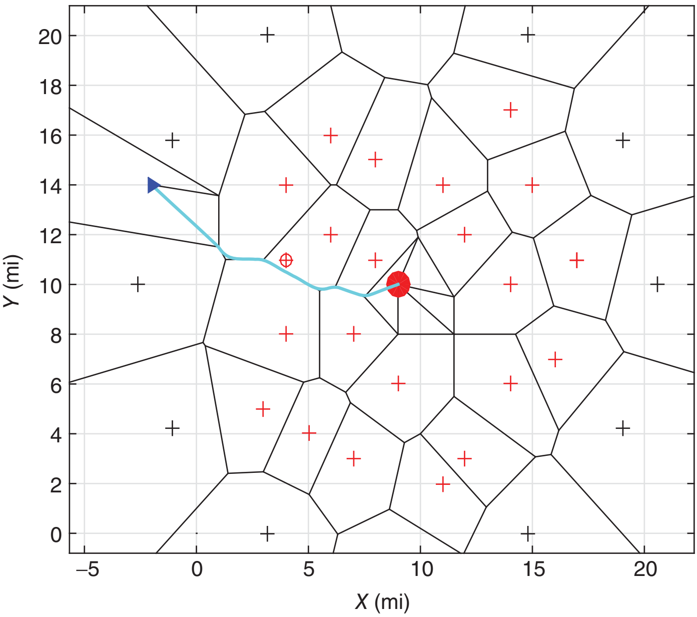


Figure 6.5 Assignment 2: UAV 1.

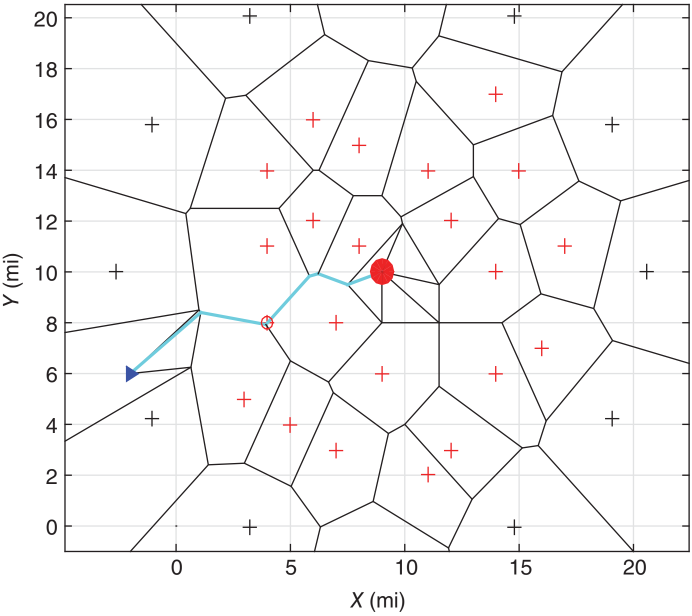


Figure 6.6 Assignment 2: UAV 3.

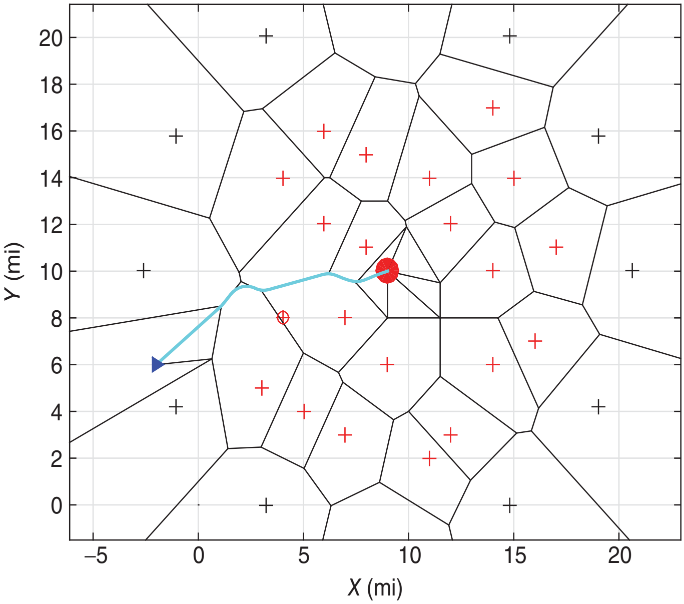


Figure 6.7 Assignment 3: UAV 3.

The optimal threat choice by each UAV is represented by the encircled threat, while the optimal path resulting from that choice is represented by the bold edges. The initial UAV positions are given by UAV 1:$(-2, 14)$, UAV 2:$(-2.5, 9)$, and UAV 3:$(-2, 6)$. Each UAV bids on its own preferred threat to destroy using its best gain in performance (6.12). In this example, at the first iteration of the consensus algorithm, each UAV bids on a different threat and UAV 2 wins the first assignment. The remaining UAVs recompute the costs based on a new Voronoi diagram and assuming the threat is eliminated at time τ_2. These new diagrams and costs result in the choices shown in Figures 6.5–6.6. Now, only agents 1 and 3 bid on their preferred threats, and agent 1 wins this assignment. In the final assignment, only agent 3 recomputes new costs, bids, and wins the assignment, its optimal path is shown in Figure 6.7. The final step is trajectory smoothing and computation of waypoints for navigation considering the minimum turning radius constraint of each vehicle. Here, we assume that the maximum turning rate is the same for all UAVs and they also travel at the same speed. The generation of smooth trajectories is only performed by the UAVs after they win their assignment. For instance, Figure 6.3 shows a smooth trajectory since this figure represents the assignment won by agent 2. Similarly, Figures 6.5 and 6.7 represent the assignments for agents 1 and 3, respectively; these figures show the corresponding smooth trajectories for each vehicle. On the other hand, Figures 6.2, 6.4, and 6.6 represent unsuccessful bids, and it is not necessary to compute waypoints for a constrained trajectory in these cases.

It is noteworthy to see how agents improve their decisions and total costs based on previous threats assigned to teammates. For instance, if UAV 3 would have won the first assignment it would have reduced its cost by 20% with respect to its optimal cost $J_3(0)$, the optimal cost obtained in the case no one of the threats is eliminated. An additional reduction of 3% with respect to $J_3(0)$ is obtained by considering the scheduled elimination (at the associated time of attack) of the first threat assigned plus its second choice. Considering two threats scheduled to be eliminated plus its third and assigned choice, UAV 3 is able to obtain an additional 2% reduction of its total cost with respect to $J_3(0)$ (for a total of 25%) that was obtained thanks to its teammates and using its teammates' decisions in an intelligent way, that is, taking the best advantage of the actions of other members of the group. By comparing Figures 6.4, 6.6, and 6.7, we can easily see this improved cost for UAV 3, where the length of its optimal path shrinks and its exposure to radar sites is reduced as well.

6.5 Multiple Main Targets

In this section, we consider the existence of more than one main target. An approach is proposed such that each UAV determines both the main target to visit and the subset of threats it will eliminate in its way to reach the selected main target.

When multiple targets exist, each UAV needs to choose only one of them. This is so because each UAV has limited resources to attack only one target. In addition, it is allowed that several UAVs can attack one particular target in order to increase the inflicted damage on the targets.

We first discuss a simple approach for each UAV to determine its preferred target to visit without considering other UAVs assigned to some of the targets. When UAVs only consider their individual costs to reach a target, then the results provided in Sections 6.3 and 6.4 can be repeated for each target. Every UAV compares between optimal costs to reach each target and finds the minimum cost among optimal paths for each target.

Extending the notation used in previous sections let $\mathcal{J}_i^*(\mathcal{T}, 1^*, m)$ represent the optimal cost obtained by choosing the best single threat to attack from the available set $\mathcal{T}$ if the agent chooses to visit target $m \in \mathcal{M}$. Agent i determines the target to visit, denoted by m_i^*, by computing

$$m_i^* = \arg \min_m \mathcal{J}_i^*(\mathcal{T}, 1^*, m), \tag{6.23}$$

Agent i still bids its normalized performance gain (6.12). The search (6.23) is only performed by agent i when it has not been assigned any threat yet. Once agent i chooses its target and the first threat to attack, then other threat assignments are restricted to only target q_i^*.

By employing the target selection criteria (6.23), it is likely that one or more targets are never visited by any UAV. It can occur that all UAVs choose the same target to attack.

Let us consider the following scenario: 4 UAVs are tasked to visit 2 main targets in a hostile environment where 24 threats are present. The UAV's coordinates are given by UAV 1:$(-6, 14)$, UAV 2:$(-5, 9)$, UAV 3:$(-4.5, 6.5)$, and UAV 4:$(-5.5, 2)$. The locations of the main targets are $T_1 : (10, 12)$ and $T_2 : (11, 7)$. The problem parameters are $\kappa = 0.5$, $\alpha = 1$, $\gamma = 0.7$, $v = 240$ ft/s, and $\Omega = 0.046$.

The communication graph is given by $a_{12} = a_{21} = a_{23} = a_{32} = a_{34} = a_{43} = 1$ and the remaining elements of the adjacency graph are equal to zero. We now consider the case where each vehicle has resources to eliminate up to two threats, that is, $T_i = 2$. The assignments with the corresponding estimated time of attack and the destination target are shown in Table 6.1.

Figures 6.8–6.11 show the optimal trajectories of each one of the four UAVs where all of them decided to visit target 1. Each figure shows the second assigned threat to the particular UAV and the optimal trajectory corresponding to that assignment.

Table 6.1 Multiple targets.

Agent	Threat coordinates (mi)	Time of attack (s)	Main target
3	(8,11)	143.71	1
1	(4,14)	61.61	1
3	(4,8)	30.16	1
1	(6,12)	109.48	1
2	(11,14)	281.24	1
2	(8,15)	262.71	1
4	(5,4)	181.62	1
4	(9,6)	283.22	1

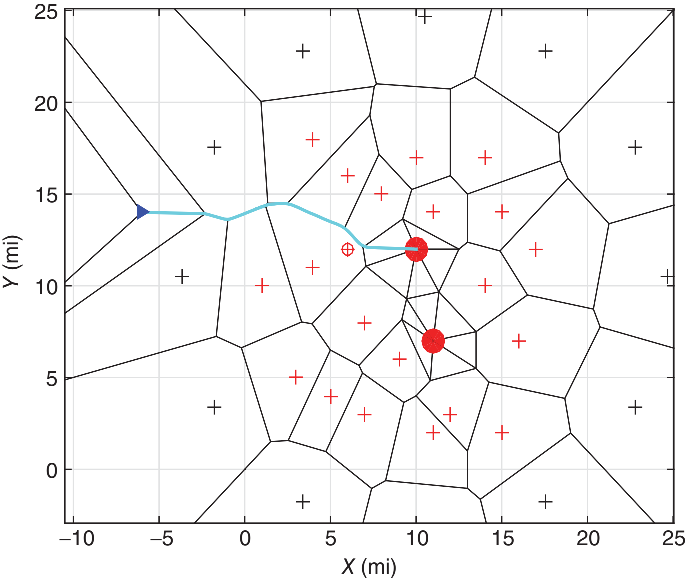


Figure 6.8 Multiple main targets: UAV 1.

For example, Figure 6.8 shows the second assignment of UAV 1, which is the threat located at (6, 12). The previous assigned threats are not visible. Those threats correspond to rows 1–3 in Table 6.1.

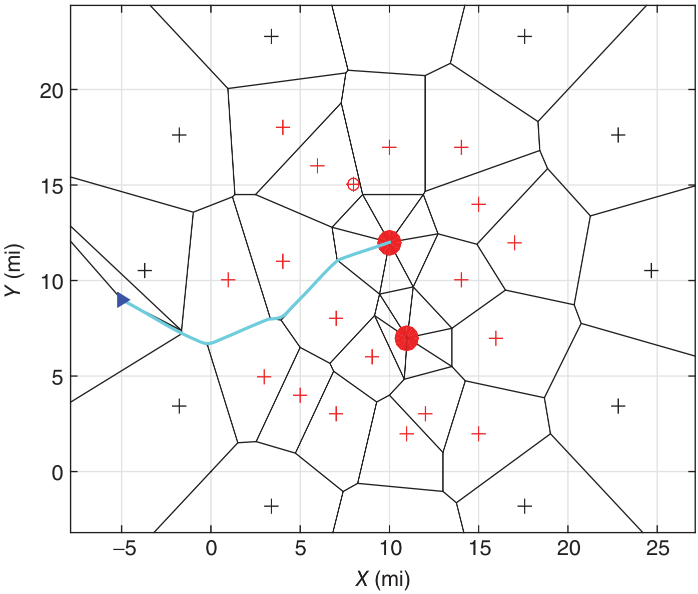


Figure 6.9 Multiple main targets: UAV 2.

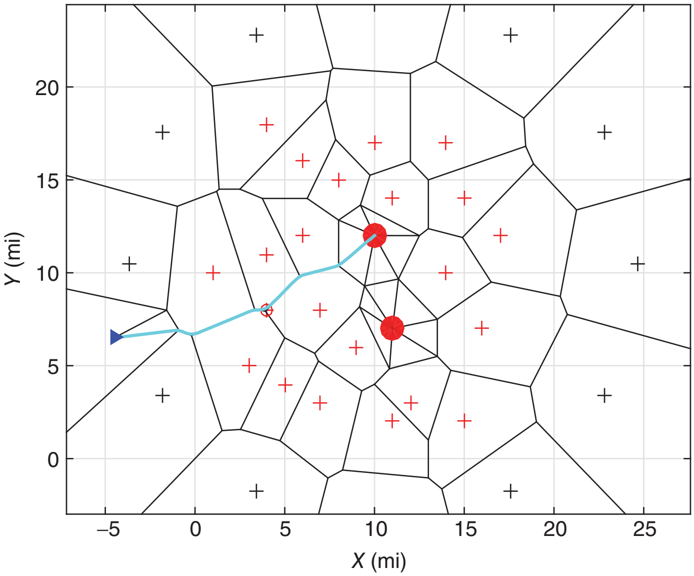


Figure 6.10 Multiple main targets: UAV 3.

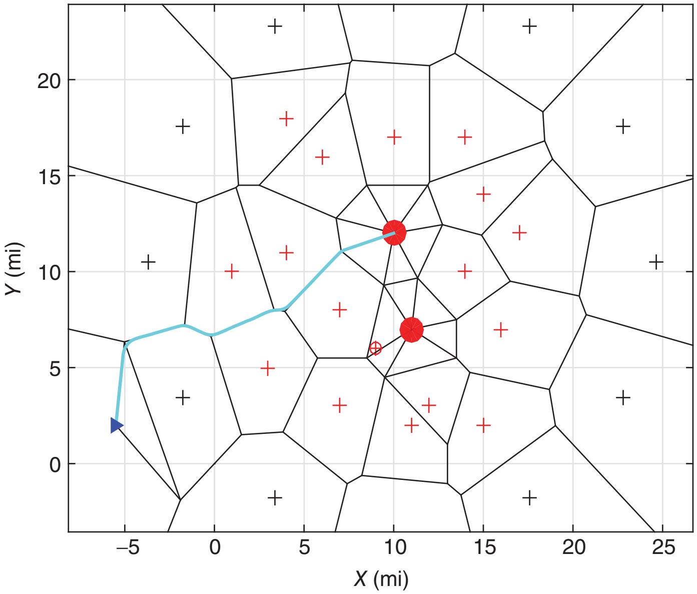


Figure 6.11 Multiple main targets: UAV 4.

Because one of the main goals of the mission is to attack as many targets as possible, the UAVs should find a trade-off between spreading their attack (visit many targets instead of all of them attacking only one target) and maintaining low length and threat costs. Overall, the objective of UAVs is to reach different targets as long as the underlying cost (distance and threat risk) does not increase too much. The method that is being proposed relies on reward functions for every main target. These functions depend on the number of UAVs assigned to each particular target; in detail, the target reward decreases as more vehicles plan to visit the target.

Let the reward associated to target $m \in \mathcal{M}$ be given by

$$R_m(\mu_m) = e^{-\gamma \mu_m}, \tag{6.24}$$

where $\gamma > 0$ is a design parameter and μ_m is a non-negative integer representing the number of UAVs that choose to visit target m. The variables μ_m, for $m \in \mathcal{M}$, vary during the assignment processes as UAVs make their decisions on which target to visit.

Finally, in order for each UAV to account for the rewards when deciding what target to visit, (6.23) is modified as follows:

$$m_i^* = \arg \min_m \frac{\mathcal{J}_i^*(\mathcal{T}, 1^*, m)}{R_m(\mu_m)}. \tag{6.25}$$

The parameter $\gamma > 0$ represents a tuning parameter that balances the global objective of spreading (visit many targets) and the local objective of keeping low individual cost.

Consider the same previous scenario. In this case, UAVs 1 and 3 have the same assignments at about the same attack times. Their optimal paths are similar to the ones shown in Figures 6.8 and 6.10, respectively. The new list of assignments is shown in Table 6.2. Now, agents 2 and 4 choose to visit target 2. Figures 6.12 and 6.13 show the optimal trajectories for UAVs 2 and 4, respectively.

Still, there will be scenarios where some targets may be relatively far and/or heavily guarded. The approach based on reward functions allows for an intelligent decision by the UAVs where they may decide to leave a target unvisited if it is deemed to be too risky. For example, consider the scenario where three UAVs, which are initially positioned at UAV 1:$(0, 10)$, UAV 2:$(1, 7.5)$, and UAV 3:$(0.5, 5)$, are tasked to visit any of the two targets located at Target_1:$(17, 8)$ and Target_2:$(22, 19)$. The hostile environment contains 30 threats. In this example, many of the threats are in the vicinity of target 2. The UAVs use the following parameters: $\kappa = 0.45$, $\alpha = 1$, and $\gamma = 0.56$. The list of

Table 6.2 Multiple targets with target rewards.

Agent	Threat coordinates (mi)	Time of attack (s)	Main target
3	(8,11)	143.71	1
2	(14,10)	330.91	2
1	(4,14)	61.61	1
3	(4,8)	30.16	1
2	(7,8)	114.78	2
4	(9,6)	283.27	2
1	(6,12)	109.48	1
4	(4,11)	167.09	2

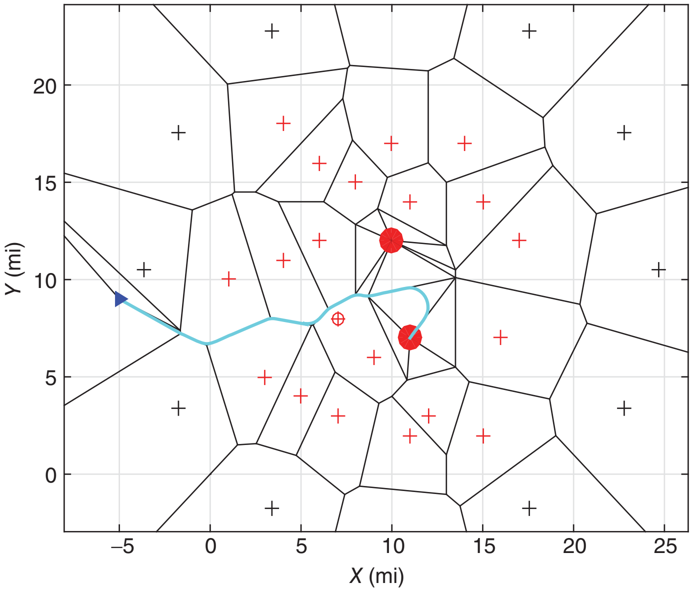


Figure 6.12 Rewards: UAV 2.

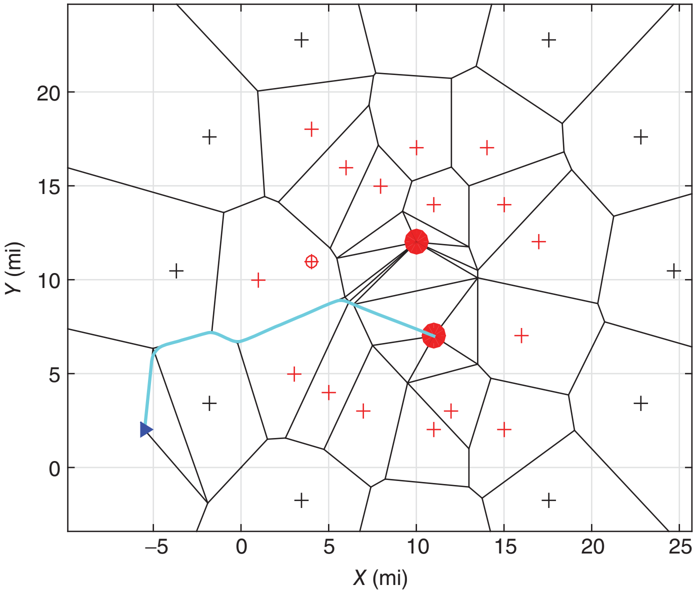


Figure 6.13 Rewards: UAV 4.

Table 6.3 Multiple Targets: three UAVs.

Agent	Threat coordinates (mi)	Time of attack (s)	Main target
1	(8,11)	24.78	1
1	(12,8)	123.96	1
2	(15,7)	180.03	1
2	(14,11)	150.74	1
3	(15,14)	261.80	1
3	(17,14)	317.49	1

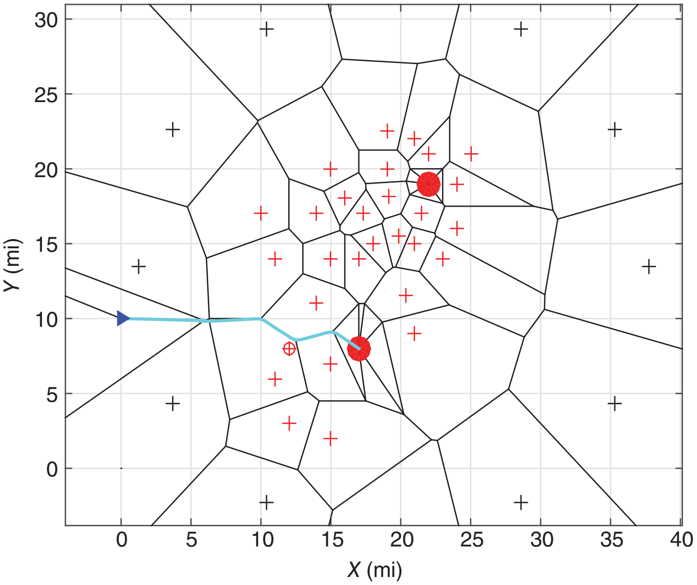


Figure 6.14 HR: UAV 1.

assignments is shown in Table 6.3. It can be seen that target 2 was considered to be too risky and all UAVs eventually decided to visit target 1 as it is shown in Figures 6.14–6.16.

The use of reward functions allows for different constraints to be imposed in the assignment of targets. For instance, if each target needs to be visited by at least one UAV, then the tuning

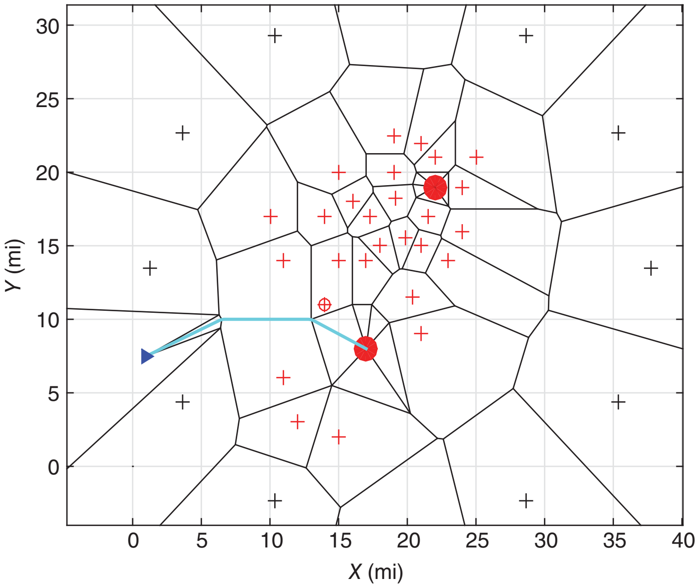


Figure 6.15 HR: UAV 2.

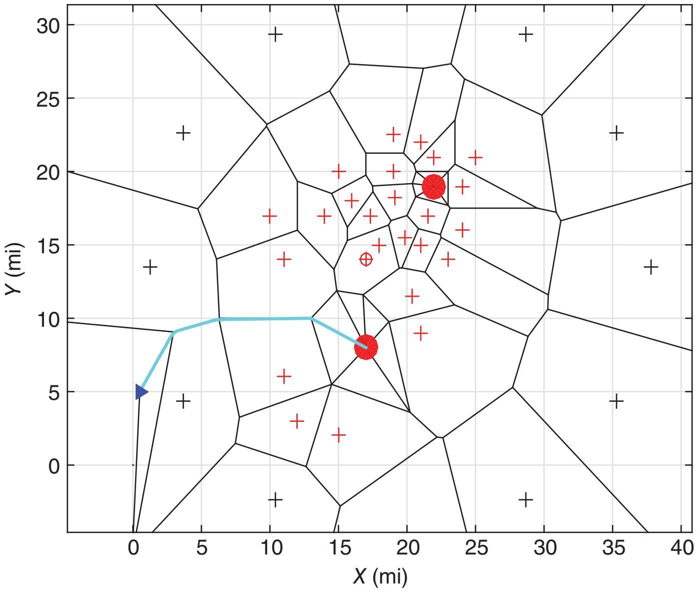


Figure 6.16 HR: UAV 3.

parameter γ can be appropriately increased in a way that the high risk associated to one target is compensated by the assignment of UAVs to less riskier targets.

6.6 Conclusions

A decentralized scheme for cooperation and coordination of actions of multiple UAVs has been presented. The problem scenario involves the selection by every UAV of which threats to eliminate and which target to visit in order to minimize a combined cost that includes radar exposure and length of travel to main target. It has been shown that the decentralized selection of the best set of threats to attack is a complex problem due to coupling and timing constraints. In order to decouple the resulting decisions, the approach taken by each UAV is to optimize its decisions subject to previous assigned threats but disregarding possible future assignments. Timing constraints have been integrated into the proposed solution where every agent considers the existence of assigned threats with their associated time of attack and plans accordingly.

References

1 Chandler, P.R., Rasmussen, S., and Pachter, M. (2000) UAV cooperative path planning, in Proceedings of the AIAA Guidance, Navigation, and Control Conference, pp. 1255–1265.

2 Beard, R.W., McLain, T.W., Goodrich, M., and Anderson, E.P. (2002) Coordinated target assignement and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, **18** (6), 911–922.

3 Bellingham, J.S., Tillerson, M., Alighanbari, M., and How, J.P. (2002) Cooperative path planning for multiple UAVs in dynamic and uncertain environments, in Proceedings of the 41st IEEE Conference on Decision and Control, pp. 2816–2822.

4 Lechevin, N., Rabbath, C.A., and Lauzon, M. (2009) A decision policy for the routing and munitions management of

multiformations of unmanned combat vehicles in adversarial urban environments. *IEEE Transactions on Control Systems Technology*, **17** (3), 505–519.

5 Jun, M. and D'Andrea, R. (2003) Path planning for unmanned aerial vehicles in uncertain and adversarial environments, in S. Butenko, R. Murphey, and P.M. Pardalos (eds), *Cooperative Control: Models, Applications and Algorithms*, Springer US, pp. 95–110.

6 Archibald, J.K., Hill, J.C., Jepsen, N.A., Stirling, W.C., and Frost, R.L. (2008) A satisficing approach to aircraft conflict resolution. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, **38** (4), 510–521.

7 Miller, B., Stepanyan, K., Miller, A., and Andreev, M. (2011) 3D path planning in a threat environment, in Proceedings of the IEEE Conference on Decision and Control-European Control Conference, pp. 6864–6869.

8 Ernest, N.D., Garcia, E., Casbeer, D.W., Cohen, K., and Schumacher, C. (2015) Multi-agent cooperative decision making using genetic fuzzy logic systems, in AIAA Infotech@Aerospace Conference, paper AIAA 2015-0888.

9 Besada-Portas, E., de la Torre, L., de la Cruz, B., and de Andres-Toro, J.M. (2010) Evolutionary trajectory planner for multiple UAVs in realistic scenarios. *IEEE Transactions on Robotics*, **26** (4), 619–634.

10 Suresh, M. and Ghose, D. (2012) UAV group and coordination tactics for ground attack missions. *IEEE Transactions on Aerospace and Electronic Systems*, **48** (1), 673–692.

11 Mitchell, J.W., Schumacher, C., Chandler, P.R., and Rasmussen, S.J. (2003) Communication delays in the cooperative control of wide area search munitions via iterative network flow, Tech. Rep., DTIC Document.

12 Schumacher, C., Chandler, P., and Rasmussen, S.J. (2002) Task allocation for wide area search munitions, in American Control Conference, pp. 1917–1922.

13 Schumacher, C., Chandler, P., Rasmussen, S.J., and Walker, D. (2003) Task allocation for wide area search munitions with variable path length, in American Control Conference, pp. 3472–3477.

14 Shima, T., Rasmussen, S.J., and Chandler, P. (2007) UAV team decision and control using efficient collaborative estimation. *ASME Journal of Dynamic Systems, Measurement, and Control*, **129**, 609–619.

15 Garcia, E. and Casbeer, D.W. (2013) UAV cooperative task allocation with communication delays and conflict resolution, in Proceedings of the AIAA Infotech@Aerospace Conference, paper AIAA 2013-4580.

16 Garcia, E. and Casbeer, D.W. (2016) Cooperative task allocation for unmanned vehicles with communication delays and conflict resolution, in AIAA Journal of Aerospace and Information Systems, **13** (2), 1–13.

17 Jin, Y., Minai, A.A., and Polycarpou, M.M. (2003) Cooperative real-time search and task allocation in UAV teams, in Decision and Control, 2003. Proceedings. 42nd IEEE Conference on, vol. **1**, IEEE, pp. 7–12.

18 Finke, J., Passino, K.M., and Sparks, A. (2003) Cooperative control via task load balancing for networked uninhabited autonomous vehicles, in Decision and Control, 2003. Proceedings. 42nd IEEE Conference on, vol. **1**, IEEE, pp. 31–36.

19 Finke, J., Passino, K.M., and Sparks, A.G. (2006) Stable task load balancing strategies for cooperative control of networked autonomous air vehicles. *IEEE Transactions on Control Systems Technology*, **14** (5), 789–803.

20 Garcia, E. and Casbeer, D.W. (2014) UAV coordinated decision making and mission management, in Proceedings of the 2014 ASME Dynamic Systems and Control Conference.

21 Dionne, D. and Rabbath, C.A. (2007) Multi-UAV decentralized task allocation with intermittent communications: the DTC algorithm, in Proceedings of the American Control Conference, pp. 5406–5411.

22 Rathinam, S., Sengupta, R., and Darbha, S. (2007) A resource allocation algorithm for multivehicle systems with nonholonomic constraints. *IEEE Transactions on Automation Science and Engineering*, **4** (1), 98–104.

23 Maza, I. and Ollero, A. (2007) Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms, in *Distributed Autonomous Robotic Systems*, vol. **6**, Springer, pp. 221–230.

24 Ponda, S.S., Redding, J., Choi, H.L., How, J.P., Vavrina, M., and Vian, J. (2010) Decentralized planning for complex missions with dynamic communication constraints, in Proceedings of the American Control Conference, pp. 3998–4003.

25 Choi, H.L., Brunet, L., and How, J.P. (2009) Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, **25** (4), 912–926.

26 Johnson, L.B., Ponda, S.S., Choi, H.L., and How, J.P. (2011) Asynchronous decentralized task allocation for dynamic environments, in Proceedings of the AIAA Infotech@Aerospace Conference.

27 Whitten, A.K., Choi, H.L., Johnson, L.B., and How, J.P. (2011) Decentralized task allocation with coupled constraints in complex missions, in Proceedings of the American Control Conference, pp. 1642–1649.

28 Garcia, E., Cao, Y., and Casbeer, D.W. (2014) Decentralized sub-optimal minimum-time consensus, in Proceedings of the 2014 ASME Dynamic Systems and Control Conference.

29 Dijkstra, E.W. (1959) A note on two problems in connection with graphs. *Numerische Mathematik*, **1** (1), 269–271.

7

Event-Triggered Communication and Control for Multi-Agent Average Consensus

Cameron Nowzari[1], *Jorge Cortes*[2] *and George J. Pappas*[1]

[1] *Department of Electrical and Systems Engineering, University of Pennsylvania, Pennsylvania, PA, USA*
[2] *Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA, USA*

7.1 Introduction

In this chapter, we look at one of the canonical driving examples for multi-agent systems: average consensus. In this scenario, a group of agents seek to agree on the average of their initial states. Depending on the particular application, such states might correspond to sensor measurements, estimates about the position of a target, or some other data that need to be fused. Due to its numerous applications in networked systems, many algorithmic solutions exist to the multi-agent average consensus problem; however, a majority of them rely on agents having continuous or periodic availability of information from other agents. Unfortunately, this assumption leads to inefficient implementations in terms of energy consumption, communication bandwidth, network congestion, and processor usage. Motivated by these observations, our main goal here is the design of provably correct distributed event-triggered strategies that autonomously decide when communication and control updates should occur so that the resulting asynchronous network executions still achieve average consensus.

The literature and motivation behind multi-agent average consensus is Extensive; see, for example, [1–4] and references therein. This chapter aims to provide a conceptual

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

introduction to event-triggered control strategies applied to consensus problems. Triggered controllers seek to understand the trade-offs between computation, communication, sensing, and actuator effort in achieving a desired task with a guaranteed level of performance. Early works [5] only consider tuning controller executions to the state evolution of a given system, but these ideas have since been extended to consider other tasks such as when to take the sample of a state or when to broadcast information over a wireless network; see [6] and references therein for a recent overview. Among the many references in the context of multi-agent systems, [7] specifies the responsibility of each agent in updating the control signals, [8] considers network scenarios with disturbances, communication delays, and packet drops, and [9] studies decentralized event-based control that incorporates estimators of the interconnection signals among agents. These works are all concerned with designing event triggers that ultimately determine when control signals should be updated in addition to how. Several works have explored the application of event-triggered ideas to the acquisition of information by the agents rather than only for actuation. To this end, [10–12] combine event-triggered controller updates with sampled data that allow for the periodic evaluation of the triggers. Instead, some works [13] drop the need for periodic access to information by considering event-based broadcasts, where agents decide with local information only when to obtain further information about neighbors. Self-triggered control [14, 15] relaxes the need for local information by deciding when a future sample of the state should be taken based on the available information from the last sampled state. Team-triggered coordination [16] combines the strengths of event- and self-triggered control into a unified approach for networked systems.

7.1.1 Organization

Table 7.1 shows the progression of event-triggered consensus problems that are covered in this chapter. It should be noted that this is a very narrow scope on the field of event-triggered consensus problems intended to introduce the high-level ideas behind event-triggered communication and control laws and provide insight into how they are designed. In particular, this

Table 7.1 Event-triggered multi-agent average consensus.

References	Triggered comm?	Trigger dependence	Memory structure	Graph structure	Trigger evaluation	Provably no Zeno?
[17]	No	State	Centralized	Undirected	Continuous	Yes
[17]	No	State	Decentralized	Undirected	Continuous	No
[18]	No	State	Centralized	Directed	Continuous	Yes
[18]	No	State	Decentralized	Directed	Continuous	No
[19]	Yes	Time	Decentralized	Undirected	Continuous	No
[19]	Yes	Time	Requires λ_2	Undirected	Continuous	Yes
[20]	Yes	State	Requires N	Undirected	Continuous	No
[21]	Yes	State	Requires K	Directed	Periodic	Yes
[12]	Yes	State	Decentralized	Undirected	Periodic	Yes
[22, 23]	Yes	State	Decentralized	Directed	Periodic	Yes

chapter only discusses works that consider single-integrator dynamics and no uncertainties (e.g., disturbances, noise, quantization, wireless communication issues). Given that this is currently an active area of research, it goes without saying that there are many important related works that are not highlighted here. Examples include scenarios with disturbances, sensor noise, delayed communication, quantized communication, packet drops, more general dynamics, dynamic topologies, and heterogeneous agents, to name a few. Lastly, it should also be noted that although the table references journal articles that first present these ideas going back to 2012, preliminary results from these works have been presented at various conferences as early as 2008. The contents of the chapter are summarized next.

The first application of event-triggered control to the multi-agent consensus problem was in [17], where the authors propose a Lyapunov-based event-triggered control strategy that dictates when agents should update their control signals. Unfortunately, its implementation relies on each agent having perfect information about their neighbors at all times. Identifying this limitation, the authors in [19] propose an event-triggered communication and control law that not only determines when agents should update their control signals but also when they should broadcast state information to their neighbors. However, the drawback of the proposed algorithm is that it is a time-dependent triggering rule with design parameters that are difficult to choose to yield good performance. Instead, a state-dependent triggering rule is proposed in [20], which better aligns the events with the desired task; this is explained in more detail later. Lastly, all the aforementioned algorithms assume that continuous evaluation of some function is possible to determine exactly when some event has occurred. Even in scenarios where Zeno behavior (an infinite number of events occurring in a finite period of time) can be provably avoided, the time between events may still be arbitrarily small, which can be problematic for digital implementations. Consequently, the works [12, 21–23] propose algorithms that only require triggering functions to be evaluated periodically rather than continuously. Finally, we close the chapter by identifying some shortcomings of the current state of the art and ideas for future work.

7.2 Preliminaries

This section introduces some notational conventions and notions on graph theory. Let $\mathbb{R}$, $\mathbb{R}_{>0}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{Z}_{>0}$ denote the set of real, positive real, nonnegative real, and positive integer numbers, respectively. We denote by $\mathbf{1}_N$ and $\mathbf{0}_N \in \mathbb{R}^N$ the column vectors with entries all equal to 1 and 0, respectively. We let $\|\cdot\|$ denote the Euclidean norm on $\mathbb{R}^N$. We let $diag(\mathbb{R}^N) = \{x \in \mathbb{R}^N \mid x_1 = \ldots = x_N\} \subset \mathbb{R}^N$ be the agreement subspace in $\mathbb{R}^N$. For a finite set S, we let $|S|$ denote its cardinality. Given $x, y \in \mathbb{R}$, Young's inequality states that, for any $\epsilon \in \mathbb{R}_{>0}$,

$$xy \leq \frac{x^2}{2\epsilon} + \frac{\epsilon y^2}{2}. \tag{7.1}$$

A weighted directed graph (or weighted digraph) $\mathcal{G} = (V, E, W)$ is comprised of a set of vertices $V = \{1, \ldots, N\}$, directed edges $E \subset V \times V$ and weighted adjacency matrix $W \in \mathbb{R}_{\geq 0}^{N \times N}$. Given an edge $(i, j) \in E$, we refer to j as an out-neighbor of i and i as an in-neighbor of j. The sets of out- and in-neighbors of a given node i are $\mathcal{N}_i^{\text{out}}$ and $\mathcal{N}_i^{\text{in}}$, respectively. The weighted adjacency matrix $W \in \mathbb{R}^{N \times N}$ satisfies $w_{ij} > 0$ if $(i, j) \in E$ and $w_{ij} = 0$ otherwise. The graph $\mathcal{G}$ is *undirected* if and only if $w_{ij} = w_{ji}$ for all $i, j \in V$. A path from vertex i to j is an ordered sequence of vertices such that each intermediate pair of vertices is an edge. A digraph $\mathcal{G}$ is strongly connected if there exists a path from all $i \in V$ to all $j \in V$. The out- and in-degree matrices D^{out} and D^{in} are diagonal matrices where

$$d_i^{\text{out}} = \sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}, \quad d_i^{\text{in}} = \sum_{j \in \mathcal{N}_i^{\text{in}}} w_{ji},$$

respectively. A digraph is weight-balanced if $D^{\text{out}} = D^{\text{in}}$. The (weighted) Laplacian matrix is $L = D^{\text{out}} - W$. Based on the structure of L, at least one of its eigenvalues is 0 and the rest of them have nonnegative real parts. If the digraph $\mathcal{G}$ is strongly connected, 0 is a simple eigenvalue with associated eigenvector $\mathbf{1}_N$. The digraph $\mathcal{G}$ is weight-balanced if and only if $\mathbf{1}_N^T L = \mathbf{0}_N$ if and only if $L_s = \frac{1}{2}(L + L^T)$ is positive semidefinite. For a strongly connected and weight-balanced digraph, zero is a simple eigenvalue of L_s. In this case, we order its eigenvalues as

$\lambda_1 = 0 < \lambda_2 \leq \ldots \leq \lambda_N$, and note the inequality

$$x^T L x \geq \lambda_2(L_s) \| x - \frac{1}{N}(\mathbf{1}_N^T x)\mathbf{1}_N \|^2, \tag{7.2}$$

for all $x \in \mathbb{R}^N$. The following property will also be of use later,

$$\lambda_2(L_s) x^T L x \leq x^T L_s^2 x \leq \lambda_N(L_s) x^T L x. \tag{7.3}$$

This can be seen by noting that L_s is diagonalizable and rewriting $L_s = S^{-1}DS$, where D is a diagonal matrix containing the eigenvalues of L_s.

7.2.1 Event-Triggered Control of Linear Systems

Here, we provide a very basic working introduction to the general idea of event-triggered control by working through a simple linear control problem. The exposition closely follows [24]. The remainder of this chapter then focuses on how these elementary ideas are extended to be applied to much more in the context of multi-agent consensus on networks. We refer the interested reader to [6] for further details on the subject of event-triggered control in general.

Consider a linear control system

$$\dot{x} = Ax + Bu, \tag{7.4}$$

with $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$. Our starting point is the availability of a linear feedback controller $u^* = Kx$ such that the closed-loop system

$$\dot{x} = (A + BK)x,$$

is asymptotically stable. Given a positive definite matrix $Q \in \mathbb{R}^{n \times n}$, let $P \in \mathbb{R}^{n \times n}$ be the unique solution to the Lyapunov equation $(A + BK)^T P + P(A + BK) = -Q$. Then, the evolution of the Lyapunov function $V_c(x) = x^T P x$ along the trajectories of the closed-loop system is

$$\dot{V}_c = x^T((A + BK)^T P + P(A + BK))x = -x^T Q x.$$

Consider now a sample-and-hold implementation of the controller, where the input is not updated continuously, but instead at a sequence of to-be-determined times $\{t_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}} \subset \mathbb{R}_{\geq 0}$,

$$u(t) = Kx(t_\ell), \quad t \in [t_\ell, t_{\ell+1}). \tag{7.5}$$

Such an implementation makes sense in practical scenarios given the inherent nature of digital systems. With this controller implementation, the closed-loop system can be written as

$$\dot{x} = (A + BK)x + BKe,$$

where $e(t) = x(t_\ell) - x(t)$, $t \in [t_\ell, t_{\ell+1})$, is the state error. Then, the objective is to determine the sequence of times $\{t_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ to guarantee some desired level of performance for the resulting system. To make this concrete, define the function

$$V(t, x_0) = x(t)^T P x(t),$$

for a given initial condition $x(0) = x_0$ (here, $t \mapsto x(t)$ denotes the evolution of the closed-loop system using (7.5)). We define the performance of the system via a function $S : \mathbb{R}_{\geq 0} \times \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ that upper bounds the evolution of V. Then, the sequence of times $\{t_\ell\}$ can be implicitly defined as the times at which

$$V(t, x_0) \leq S(t, x_0) \tag{7.6}$$

is not satisfied. More specifically, this is an *event-triggered* condition that updates the actuator signal whenever $V(t_\ell, x_0) = S(t_\ell, x_0)$. Assuming that solutions are well defined, it is not difficult to see that if the performance function satisfies $S(t, x_0) \leq \beta(t, |x_0|)$, for some $\beta \in \mathcal{KL}$, then the closed-loop system is globally uniformly asymptotically stable. Moreover, if β is an exponential function, the system is globally uniformly exponentially stable.

Therefore, one only needs to guarantee the lack of Zeno behavior. We do this by choosing the performance function S so that the inter-event times $t_{\ell+1} - t_\ell$ are lower bounded by some constant positive quantity. This can be done in a number of ways. For the linear system (7.4), it turns out that it is sufficient to select S satisfying $\dot{V}(t_\ell) < \dot{S}(t_\ell)$ at the event times t_ℓ (this fact is formally stated in Theorem 7.2.1). To do so, choose $R \in \mathbb{R}^{n \times n}$ positive definite such that $Q - R$ is also positive definite. Then, there exists a Hurwitz matrix $A_s \in \mathbb{R}^{n \times n}$ such that the Lyapunov equation

$$A_s^T P + P A_s = -R$$

holds. Consider the hybrid system,

$$\begin{aligned} \dot{x}_s &= A_s x_s, \quad t \in [t_\ell, t_{\ell+1}), \\ x_s(t_\ell) &= x(t_\ell), \end{aligned}$$

whose trajectories we denote by $t \mapsto x_s(t)$, and define the performance function S by

$$S(t) = x_s^T(t)Px_s(t).$$

Letting $y = [x^T, e^T]^T \in \mathbb{R}^n \times \mathbb{R}^n$, we write the continuous-time dynamics as

$$\dot{y} = Fy, \quad t \in [t_\ell, t_{\ell+1}),$$

where

$$F = \begin{bmatrix} A + BK & BK \\ -A - BK & -BK \end{bmatrix}.$$

With a slight abuse of notation, we let $y_\ell = [x^T(t_\ell), 0^T]^T$ be the state y at time t_ℓ. Note that $e(t_\ell) = 0$, for all $\ell \in \mathbb{Z}_{\geq 0}$, by definition of the update times. With this notation, we can rewrite

$$S(t) = (Ce^{F_s(t-t_\ell)}y_\ell)^T P(Ce^{F_s(t-t_\ell)}y_\ell),$$
$$V(t) = (Ce^{F(t-t_\ell)}y_\ell)^T P(Ce^{F(t-t_\ell)}y_\ell),$$

where

$$F_s = \begin{bmatrix} A_s & 0 \\ 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} I & 0 \end{bmatrix}.$$

The condition (7.6) can then be rewritten as

$$f(t, y_\ell) = y_\ell^T(e^{F^T(t-t_\ell)}C^T PCe^{F(t-t_\ell)} - e^{F_s^T(t-t_\ell)}C^T PCe^{F_s(t-t_\ell)})$$
$$y_\ell \leq 0.$$

Note that because we consider a deterministic system here, with the information available at time t_ℓ, it is possible to determine the next time $t_{\ell+1}$ at which (7.6) is violated by computing $t_{\ell+1} = h(x(t_\ell))$ as the time for which

$$f(h(x(t_\ell)), y_\ell) = 0. \tag{7.7}$$

The following result from [24] provides a uniform lower bound $t_{\min}$ on the inter-event times $\{t_{\ell+1} - t_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$.

Theorem 7.2.1 ***(Lower bound on inter-event times for event-triggered approach)*** Given the system (7.4) with controller (7.5) and controller updates given by the event-triggered policy (7.7), the inter-event times are lower bounded by

$$t_{\min} = \min\{t \in \mathbb{R}_{>0} \mid \det(M(t)) = 0\} > 0,$$

where

$$M(t) = \begin{bmatrix} I & 0 \end{bmatrix} \left(e^{Ft} C^T P C e^{Ft} - e^{F_s t} C^T P C e^{F_s t} \right) \begin{bmatrix} I \\ 0 \end{bmatrix}.$$

Note that the aforementioned result can also be interpreted in the context of a periodic controller implementation: any period less than or equal to $t_{\min}$ results in a closed-loop system with asymptotic stability guarantees.

7.3 Problem Statement

We let $\mathcal{G}$ denote the connected, undirected communication graph that describes the communication topology in a network of N agents. In other words, agent j can communicate with agent i if j is a neighbor of i in $\mathcal{G}$. We denote by $x_i \in \mathbb{R}$ the state of agent $i \in \{1, \dots, N\}$ and consider single-integrator dynamics

$$\dot{x}_i(t) = u_i(t). \tag{7.8}$$

Then, the distributed controller

$$u_i^*(x) = -\sum_{j \in \mathcal{N}_i} (x_i - x_j) \tag{7.9}$$

is known to drive the states of all agents to the average of the initial conditions [1, 25]. This is formalized in Theorem 7.3.1.

Theorem 7.3.1 ***(Continuous controller)*** Given the dynamics (7.8), if all agents implement the control law (7.9), then multi-agent average consensus is achieved; that is,

$$\lim_{t \to \infty} x_i(t) = \frac{1}{N} \sum_{j=1}^{N} x_j(0) \tag{7.10}$$

for all $i \in \{1, \dots, N\}$.

Unfortunately, implementing (7.9) in a digital setting is not possible since it requires all agents to have continuous access to the state of their neighbors, and the control inputs $u_i(t)$ must also be updated continuously. This is especially troublesome in the context of wireless network systems since this means agents must communicate with each other continuously as well. Instead, this chapter is interested in event-triggered communication and control strategies to relax these requirements.

7.4 Centralized Event-Triggered Control

Consider the dynamics (7.8) and the ideal control law (7.9). Letting $x = (x_1, \dots, x_N)^T$ and $u = (u_1, \dots, u_N)^T$, the closed-loop dynamics of the ideal system is given by

$$\dot{x}(t) = -Lx(t). \tag{7.11}$$

As stated before, implementing this requires all agents to continuously update their control signals, which is not realistic for digital controllers. Instead, following the basic idea for event-triggered control presented in Section 7.2.1, let us consider a digital implementation of this ideal controller

$$u(t) = -Lx(t_\ell), \quad t \in [t_\ell, t_{\ell+1}), \tag{7.12}$$

where the event times $\{t_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ are to be determined such that the system still converges to the desired state. Let $e(t) = x(t_\ell) - x(t)$ for $t \in [t_\ell, t_{\ell+1})$ be the state measurement error. For simplicity, we denote by $\hat{x}(t) = \hat{x}(t_\ell)$ for $t \in [t_\ell, t_{\ell+1})$ as the state that was used in the last computation of the control signal. The closed-loop dynamics of the controller (7.12) is then given by

$$\dot{x}(t) = -L\hat{x}(t) = -L(x(t) + e(t)). \tag{7.13}$$

The problem can now be formalized as follows.

Problem 7.4.1 ***(Centralized event-triggered control)*** Given the closed-loop dynamics (7.13), find an event trigger such that the sequence of times $\{t_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ ensures multi-agent average consensus (7.10) is achieved.

Following [17], to solve this problem we consider the Lyapunov function

$$V(x) = x^T Lx.$$

Given the closed-loop dynamics (7.13), we have

$$\dot{V} = x^T L\dot{x} = -x^T LL(x+e) = -\underbrace{\|Lx\|^2}_{\text{“good”}} - \underbrace{x^T LLe}_{\text{“bad”}}.$$

For simplicity, we are not interested in characterizing any specific performance as in Section 7.2.1. Instead, we are only interested in asymptotic stability. The main idea of event-triggered control is then to determine when the controller should be updated (i.e., when e should be set to 0) by balancing the “good” term against the “bad” term. More specifically, we are interested in finding conditions on the error e such that $\dot{V} < 0$ at all times. Using norms, we can bound

$$\dot{V} \leq -\|Lx\|^2 + \|Lx\|\|L\|\|e\|.$$

Then, if we enforce the error e to satisfy

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

with $\sigma \in (0,1)$ for all times, we have

$$\dot{V} \leq (\sigma - 1)\|Lx\|^2,$$

which is strictly negative for all $Lx \neq 0$. The following centralized event trigger ensures that this is satisfied at all times.

Theorem 7.4.2 *(Centralized event-triggered control)* Given the closed-loop dynamics (7.13), if the update times are determined as the times when

$$f(x,e) \triangleq \|e\| - \sigma \frac{\|Lx\|}{\|L\|} = 0, \tag{7.14}$$

then the system achieves multi-agent average consensus.

In other words, given a control update at time t_ℓ, the next time $t_{\ell+1}$ is given by

$$t_{\ell+1} = \min\{t' > t_\ell \mid \|e(t')\| = \sigma \frac{\|Lx(t')\|}{\|L\|}\}.$$

The algorithm is formalized in Table 7.2.

The proof of convergence to the desired state then follows directly from the proof of Theorem 7.3.1 and the fact that the

Table 7.2 Centralized event-triggered control.

```
At times $t \in [t_\ell, t_{\ell+1})$, system (continuously) performs:
set $\hat{x}(t) = x(t_\ell)$
set $e(t) = \hat{x}(t) - x(t)$
if $\|e(t)\| = \sigma \frac{\|Lx(t)\|}{\|L\|}$ then
    set $t_{\ell+1} = t$
    set $\hat{x}(t) = x_i(t_{\ell+1})$
    set $\ell = \ell + 1$
end if
set $u(t) = -L\hat{x}(t)$
```

sum of all states is still an invariant quantity. Furthermore, the authors in [17] are able to rule out the existence of Zeno behavior (formally defined later) by showing there exists a positive time

$$\tau = \frac{\sigma}{\|L\|(1+\sigma)}$$

bounding the inter-event times, that is,

$$t_{\ell+1} - t_\ell \geq \tau$$

for all $\ell \in \mathbb{Z}_{\geq 0}$.

Definition 7.4.3 ***(Zeno behavior)*** If there exists $T > 0$ such that $t_\ell \leq T$ for all $\ell \in \mathbb{Z}_{\geq 0}$, then the system is said to exhibit *Zeno behavior*.

The centralized event-triggered controller (7.12) with triggering law (7.14) relaxes the requirement that agents need to continuously update their control signals, but it still has many issues. One of them is that the event trigger $f(x, e)$ requires full state information to implement. Next, we provide a distributed solution instead of a centralized one.

7.5 Decentralized Event-Triggered Control

In the previous section, we presented a centralized event-triggered control law to solve the multi-agent average consensus problem. Unfortunately, implementing this requires a centralized decision maker and requires all agents in the network to update their control signals simultaneously. In this section, we relax this requirement by following [17].

Let us now consider a distributed digital implementation of the ideal controller (7.9). In this case, we assume that each agent i has its own sequence of event times $\{t^i_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$. At any given time t, let $\hat{x}_i(t) = x_i(t^i_\ell)$ for $t \in [t^i_\ell, t^i_{\ell+1})$ be the state of agent i at its last update time. The distributed event-triggered controller is then given by

$$u_i(t) = -\sum_{j \in \mathcal{N}_i} (\hat{x}_i(t) - \hat{x}_j(t)). \tag{7.15}$$

It is important to note here that the latest updated state $\hat{x}_j(t)$ of agent $j \in \mathcal{N}_i$ appears in the control signal for agent i. This means that when an event is triggered by a neighboring agent j, agent i also updates its control signal accordingly. As in the centralized case, let $e_i(t) = x_i(t^i_\ell) - x_i(t)$ be the state measurement error for agent i. Then, letting $\hat{x} = (\hat{x}_1, \ldots, \hat{x}_N)^T$ and $e = (e_1, \ldots, e_N)^T$, the closed-loop dynamics of the controller (7.15) is given by

$$\dot{x}(t) = -L\hat{x}(t) = -L(x(t) + e(t)). \tag{7.16}$$

The problem can now be formalized as follows.

Problem 7.5.1 ***(Decentralized event-triggered control)*** Given the closed-loop dynamics (7.16), find an event trigger for each agent i such that the sequence of times $\{t^i_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ ensures multi-agent average consensus (7.10) is achieved.

Following [17], to solve this problem we again consider the Lyapunov function

$$V(x) = x^T L x.$$

Given the closed-loop dynamics (7.16), we have

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

As before, we are interested in finding conditions on the error e such that $\dot{V} < 0$ at all times; however, we must now do this in a distributed way. For simplicity, let $Lx \triangleq z = (z_1, \dots, z_N)^T$. Then, expanding out $\dot{V}$ yields

$$\begin{aligned}\dot{V} &= -\sum_{i=1}^{N} z_i^2 - \sum_{j \in \mathcal{N}_i} z_i(e_i - e_j) \\ &= -\sum_{i=1}^{N} z_i^2 - |\mathcal{N}_i| z_i e_i + \sum_{j \in \mathcal{N}_i} z_i e_j.\end{aligned}$$

Using Young's inequality (7.1) and the fact that $\mathcal{G}$ is symmetric, we can bound this by

$$\dot{V} \leq -\sum_{i=1}^{N} (1 - a|\mathcal{N}_i|) z_i^2 + \frac{1}{a} |\mathcal{N}_i| e_i^2$$

for all $a > 0$. Letting $a \in (0, 1/|\mathcal{N}_i|)$ for all i, if we can enforce the error of all agents to satisfy

$$e_i^2 \leq \frac{\sigma_i a(1 - a|\mathcal{N}_i|)}{|\mathcal{N}_i|} z_i^2$$

with $\sigma_i \in (0, 1)$ for all times, we have

$$\dot{V} \leq \sum_{i=1}^{N} (\sigma_i - 1)(1 - a|\mathcal{N}_i|) z_i^2,$$

which is strictly negative for all $Lx \neq 0$. The following decentralized event trigger ensures that this is satisfied at all times.

Theorem 7.5.2 ***(Decentralized event-triggered control)*** Given the closed-loop dynamics (7.16), if the updates times of each agent i are determined as the times when

$$f_i(x_i, e_i, \{x_j\}_{j \in \mathcal{N}_i}) \triangleq e_i^2 - \frac{\sigma_i a(1 - a|\mathcal{N}_i|)}{|\mathcal{N}_i|} z_i^2 = 0, \tag{7.17}$$

with $0 < a < 1/|\mathcal{N}_i|$, then the system achieves multi-agent average consensus.

Note that the trigger (7.17) can be evaluated by agent i using only information about its own and neighbors' states. The algorithm is formalized in Table 7.3.

Table 7.3 Decentralized event-triggered control.

```
At times $t \in [t^i_\ell, t^i_{\ell+1})$, agent $i$ (continuously) performs:

set $z_i(t) = \sum_{j \in \mathcal{N}_i}(x_i(t) - x_j(t))$
set $e_i(t) = \hat{x}_i(t) - x_i(t)$
if $e_i(t)^2 = \frac{\sigma_i a(1-a|\mathcal{N}_i|)}{|\mathcal{N}_i|} z_i(t)^2$ then
    set $t^i_{\ell+1} = t$
    broadcast $\hat{x}_i(t) = x_i(t^i_{\ell+1})$ to neighbors $j \in \mathcal{N}_i$
    set $\ell = \ell + 1$
end if
set $u_i(t) = -\sum_{j \in \mathcal{N}_i}(\hat{x}_i(t) - \hat{x}_j(t))$
```

The proof of convergence to the desired state then directly follows from the proof of Theorem 7.3.1 and the fact that the sum of all states is still an invariant quantity. Furthermore, the authors in [17] are able to show that at all times there exists one agent i for which the inter-event times are strictly positive. Unfortunately, this is not enough to rule out Zeno behavior (an infinite number of triggers occurring in a finite time period), which is quite problematic, both from a pragmatic and theoretical viewpoint, as the trajectories of the system are no longer well defined beyond the accumulation point in time.

Remark 7.5.3 ***(Convergence and Zeno behavior)*** It should be noted here that when we refer to a "proof of convergence" for any closed-loop dynamics, it is only valid for trajectories that do not exhibit Zeno behavior. Consequently, being able to guarantee Zeno behaviors do not occur is extremely important in validating the correctness of a given algorithm. ■

Remark 7.5.4 ***(Directed graphs)*** All the work from [17] has also been extended to consider weight-balanced directed graphs in [18]. For brevity, we defer the discussion on directed graphs to Section 7.6.1. ■

The decentralized event-triggered controller (7.15) with triggering law (7.17) relaxes the requirement that agents need to continuously update their control signals; however, there are still some severe issues. Although each agent now has a local event-triggering condition, it requires continuous information about all of its neighbors to implement it. This is still troublesome in a wireless network setting where this implies continuous communication among agents is still required. We address this next.

7.6 Decentralized Event-Triggered Communication and Control

In the previous sections, we presented event-triggered *control* laws to determine when control signals should be updated; however, this relied on the continuous availability of some state information. In particular, each agent i requires exact state information about their neighbors $j \in \mathcal{N}_i$ to evaluate the trigger (7.17) and determine when its control signal u_i should be updated. Instead, we are now interested in developing event-triggered *communication and control* laws such that each agent i must not only determine when to update its control signal but also when to communicate with its neighbors. For simplicity, we refer to communication and control together as "coordination."

As in the previous section, we assume each agent i has its own sequence of event times $\{t_\ell^i\}_{\ell \in \mathbb{Z}_{\geq 0}}$. However, these update times now correspond to when messages are broadcast in addition to when control signals are updated. At any given time t, let $\hat{x}_i(t) = x_i(t_\ell^i)$ for $t \in [t_\ell^i, t_{\ell+1}^i)$ be the last broadcast state of agent i. Then, at any given time t, agent i only has access to the last broadcast state $\hat{x}_j(t)$ of its neighbors $j \in \mathcal{N}_i$ rather than exact states $x_j(t)$.

The distributed event-triggered controller is then still given by

$$u_i(t) = -\sum_{j \in \mathcal{N}_i} (\hat{x}_i(t) - \hat{x}_j(t)). \tag{7.18}$$

It is important to note here that the latest broadcast state $\hat{x}_j(t)$ of agent $j \in \mathcal{N}_i$ appears in the control signal for agent i. This means that when an event is triggered by a neighboring agent j, agent i also updates its control signal accordingly. As before, let $e_i(t) = x_i(t_\ell^i) - x_i(t)$ be the state measurement error for agent i. Then, letting $\hat{x} = (\hat{x}_1, \dots, \hat{x}_N)^T$ and $e = (e_1, \dots, e_N)^T$, the closed-loop dynamics of the controller (7.15) is given by

$$\dot{x}(t) = -L\hat{x}(t) = -L(x(t) + e(t)). \tag{7.19}$$

The problem can now be formalized as follows. However, it should be noted that we are now looking for a strictly *local* event trigger for each agent i that does not require exact information about its neighbors. More specifically, we recall the result of Theorem 7.5.2 and notice that the event trigger for agent i depends on the exact state $x_j(t)$ of all its neighbors $j \in \mathcal{N}_i$. In this section, we are interested in finding a trigger that only depends on the last broadcast information $\hat{x}_j(t)$ instead.

Problem 7.6.1 ***(Decentralized event-triggered coordination)*** Given the closed-loop dynamics (7.16), find a local event trigger for each agent i such that the sequence of times $\{t^i_\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ ensures multi-agent average consensus (7.10) is achieved.

Here, we present two classes of event-triggered coordination solutions to the aforementioned problem: time-dependent and state-dependent triggers. The time-dependent event trigger to solve this problem was first developed in [19] and is presented next. The algorithm is formalized in Table 7.4.

Theorem 7.6.2 ***Decentralized event-triggered coordination (time-dependent)*** Given the closed-loop dynamics (7.16), if the updates times of each agent i are determined as the times when

$$f_i(e_i(t), t) \triangleq \|e_i(t)\| - (c_0 + c_1 e^{-\alpha t}) = 0, \tag{7.20}$$

Table 7.4 Decentralized event-triggered coordination (time-dependent).

```
At times $t \in [t^i_\ell, t^i_{\ell+1})$, agent $i$ (continuously) performs:
set $\hat{x}_i(t) = x_i(t^i_\ell)$
set $e_i(t) = \hat{x}_i(t) - x_i(t)$
**if** $|e_i(t)| = c_0 + c_1 e^{-\alpha t}$ **then**
    set $t^i_{\ell+1} = t$
    broadcast $\hat{x}_i(t) = x_i(t^i_{\ell+1})$ to neighbors $j \in \mathcal{N}_i$
    set $\ell = \ell + 1$
**end if**
set $u_i(t) = -\sum_{j \in \mathcal{N}_i} (\hat{x}_i(t) - \hat{x}_j(t))$
```

with constants $c_0, c_1 \geq 0$ and $c_0 + c_1 > 0$, then the system reaches a neighborhood of multi-agent average consensus upper-bounded by

$$r = \|L\|\sqrt{N}c_0/\lambda_2(L).$$

Moreover, if $c_0 > 0$ or $0 < \alpha < \lambda_2(L)$, then the closed-loop system does not exhibit Zeno behavior.

The proof of convergence is shown in the appendix; however, we are now also interested in guaranteeing that Zeno behavior does not occur to verify the correctness of the algorithm as mentioned earlier.

The main drawback of the event-triggered communication and control law proposed in Theorem 7.6.2 is that although the parameters c_0, c_1, and α play very important roles in the performance of the algorithm (e.g., convergence speed and amount of triggers), there is no good way of choosing these parameters a priori, without any global knowledge. Furthermore, the initial condition also plays an important role in the performance of the algorithm.

In particular, we focus our discussion here on the parameters c_0 and α and their effects on convergence and possible Zeno behaviors. We begin with the more desirable $c_0 = 0$ case, as in this case the result of Theorem 7.6.2 states that the system will asymptotically achieve exact multi-agent average consensus as defined in (7.10). However, in this case we require $\alpha < \lambda_2(L)$ to guarantee Zeno behaviors can be avoided and, unfortunately, $\lambda_2(L)$ is a global quantity that requires knowledge about the entire communication topology to compute. There are indeed methods for estimating this quantity in a distributed way (see, e.g., [26, 27]), but we do not discuss this here. On the other hand, when $c_0 > 0$, we can guarantee that Zeno behaviors are avoided regardless of our choice of α; however, we lose the asymptotic convergence guarantee. That is, for $c_0 > 0$ we can only guarantee convergence to a neighborhood of the desired average consensus state.

As a result of the aforementioned discussion, we see that there is no way the agents can choose the parameters c_0, c_1, and α to ensure asymptotic convergence to the average consensus state while also guaranteeing Zeno executions are avoided.

Consequently, more recent works have proposed a local Lyapunov-based event-triggering condition that only relies on currently available information and no exogenous signals (e.g., time). This also naturally aligns when events are triggered with the progression of the task as encoded in the Lyapunov function. The state-dependent event trigger to solve this problem was first developed in [20] and improved upon in [23] (removed global parameter a requirement); we present this next.

Following [23], to solve this problem we consider a different Lyapunov function,

$$V(x) = \frac{1}{2}(x - \bar{x}\mathbf{1})^T(x - \bar{x}\mathbf{1}),$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ is the average of all initial conditions. Then, given the closed-loop dynamics (7.16), we have

$$\dot{V} = -x^T\dot{x} - \bar{x}\mathbf{1}^T\dot{x} = -x^T L\hat{x} - \bar{x}\mathbf{1}^T L\hat{x} = -x^T L\hat{x},$$

where we have used the fact that the graph is weight-balanced in the last equality. As before, we are interested in finding conditions on the error e such that $\dot{V} < 0$ at all times; however, we must now do it without access to neighboring state information. Recalling $e_i(t) = \hat{x}_i(t) - x_i(t)$, we can expand this out to

$$\begin{aligned}\dot{V} &= -\hat{x}^T L\hat{x} + e^T L\hat{x} \\ &= -\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(\frac{1}{2}(\hat{x}_i - \hat{x}_j)^2 - e_i(\hat{x}_i - \hat{x}_j)\right).\end{aligned}$$

Using Young's inequality for each product (see [23] for the reason why this choice)

$$e_i(\hat{x}_i - \hat{x}_j) \leq e_i^2 + \frac{1}{4}(\hat{x}_i - \hat{x}_j)^2$$

yields

$$\begin{aligned}\dot{V} &\leq -\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(\frac{1}{2}(\hat{x}_i - \hat{x}_j)^2 - e_i^2 - \frac{1}{4}(\hat{x}_i - \hat{x}_j)^2\right) \\ &= -\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(\frac{1}{4}(\hat{x}_i - \hat{x}_j)^2 - e_i^2\right) \\ &= \sum_{i=1}^{N} e_i|\mathcal{N}_i| - \sum_{j\in\mathcal{N}_i}\left(\frac{1}{4}(\hat{x}_i - \hat{x}_j)^2\right).\end{aligned}$$

If we can enforce the error of all agents to satisfy

$$e_i^2 \le \sigma_i \frac{1}{4|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (\hat{x}_i - \hat{x}_j)^2$$

with $\sigma_i \in (0, 1)$ for all times, we have

$$\dot{V} \le \sum_{i=1}^{N} \frac{\sigma_i - 1}{4} \sum_{j \in \mathcal{N}_i} (\hat{x}_i - \hat{x}_j)^2,$$

which is strictly negative for all $L\hat{x} \neq 0$. The following decentralized event trigger ensures this is satisfied at all times.

Theorem 7.6.3 ***(Decentralized event-triggered coordination (state-dependent))*** Given the closed-loop dynamics (7.16), if the update times of each agent i are determined as the times when

$$f_i(e_i) \triangleq e_i^2 - \sigma_i \frac{1}{4|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (\hat{x}_i - \hat{x}_j)^2 \ge 0, \tag{7.21}$$

then the system achieves multi-agent average consensus.

It should be noted here that unlike all the other triggers presented so far, this trigger is given by an inequality rather than an equality. This is a result of the state-dependent triggering function that agents use to determine when to communicate. Since agents are asynchronously sending each other messages, the information they have about one another is also changing discontinuously (Table 7.5).

7.6.1 Directed Graphs

Up until now we have assumed that the communication graph was always undirected. Here, we extend the previous results to cases where the communication graph $\mathcal{G}$ is directed but strongly connected and weight-balanced.

More specifically, we say that agent i can only send messages to its out-neighbors $j \in \mathcal{N}_i^{\text{out}}$. Similarly, it can only receive messages broadcast by its in-neighbors $j \in \mathcal{N}_i^{\text{in}}$. Conveniently, the closed-loop system dynamics is still given by (7.16) where the only difference now is L is not symmetric. However, because it

Table 7.5 Decentralized event-triggered coordination (state-dependent).

At times $t \in [t^i_\ell, t^i_{\ell+1})$, agent i (continuously) performs:

```
set $\hat{x}_i(t) = x_i(t^i_\ell)$
set $e_i(t) = \hat{x}_i(t) - x_i(t)$
**if** $e_i(t)^2 \geq \sigma_i \frac{1}{4|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} (\hat{x}_i(t) - \hat{x}_j(t))^2$ **then**
    set $t^i_{\ell+1} = t$
    broadcast $\hat{x}_i(t) = x_i(t^i_{\ell+1})$ to neighbors $j \in \mathcal{N}_i$
    set $\ell = \ell + 1$
**end if**
set $u_i(t) = -\sum_{j \in \mathcal{N}_i} (\hat{x}_i(t) - \hat{x}_j(t))$
```

is weight-balanced we still have that the sum of all states is an invariant quantity,

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{1}_N^T x(t)) = \mathbf{1}_N^T \dot{x}(t) = -\mathbf{1}_N^T L \hat{x}(t) = 0.$$

Remark 7.6.4 ***(Weight-balanced assumption)*** It should be noted that the weights of the directed graph for any digital implementations are design parameters that can be chosen to make a given directed communication topology weight-balanced. The works [28, 29] present provably correct distributed strategies that, given a directed communication topology, allow a network of agents to find such weight edge assignments.

Remarkably, the same analysis from the previous section almost directly follows and admits a similar triggering law. More specifically, it can be shown that if we can enforce the error of all agents to satisfy

$$e_i^2 \leq \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j \in \mathcal{N}_i^{\text{out}}} (\hat{x}_i - \hat{x}_j)^2,$$

with $\sigma_i \in (0, 1)$ for all times, we have

$$\dot{V} \leq \sum_{i=1}^{N} \frac{\sigma_i - 1}{4} \sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij} (\hat{x}_i - \hat{x}_j)^2, \tag{7.22}$$

Table 7.6 Decentralized event-triggered coordination on directed graphs.

```
At times $t \in [t_\ell^i, t_{\ell+1}^i)$, agent $i$ (continuously) performs:

set $\hat{x}_i(t) = x_i(t_\ell^i)$
set $e_i(t) = \hat{x}_i(t) - x_i(t)$
if $e_i(t)^2 \geq \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i(t) - \hat{x}_j(t))^2$ then
    set $t_{\ell+1}^i = t$
    broadcast $\hat{x}_i(t) = x_i(t_{\ell+1}^i)$ to in-neighbors $j \in \mathcal{N}_i^{\text{in}}$
    set $\ell = \ell + 1$
end if
set $u_i(t) = -\sum_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i(t) - \hat{x}_j(t))$
```

which is strictly negative for all $L\hat{x} \neq 0$. The following decentralized event trigger ensures this is satisfied at all times (Table 7.6).

Theorem 7.6.5 ***(Decentralized event-triggered coordination on directed graphs)*** Given the closed-loop dynamics (7.16), if the communication graph $\mathcal{G}$ is weight-balanced and the updates times of each agent i are determined as the times when

$$f_i(e_i) \triangleq e_i^2 - \sigma_i \frac{1}{4d_i^{out}} \sum_{j\in\mathcal{N}_i} w_{ij}(\hat{x}_i - \hat{x}_j)^2 \geq 0, \tag{7.23}$$

then the system achieves multi-agent average consensus.

Unfortunately, most of the algorithms presented here are not guaranteed to avoid Zeno behaviors making them risky to implement on real systems. Moreover, the one algorithm that can in some cases guarantee no Zeno behavior requires some global information. In some cases, modifications can be made to theoretically ensure that no Zeno behavior occurs; however, there may still be an arbitrarily small amount of time between any two events (see, e.g., [23]) making it undesirable from an implementation viewpoint. This is addressed in Remark 7.6.6 and the following section.

Remark 7.6.6 ***(Implementation)*** We note here an important issue regarding the connection between Zeno executions and implementation. In general, dedicated hardware can only operate at some maximum frequency (e.g., a physical device can only broadcast a message or evaluate a function a finite number of times in any finite period of time). This means that ensuring that a system does not exhibit Zeno behavior may not be enough to guarantee that the algorithm can be implemented on a physical system if the physical hardware cannot match the speed of actions required by the algorithm. More specifically, it is guaranteed that Zeno behavior does not exist if the sequence of times $t_\ell^i \to \infty$ as $\ell \to \infty$; however, this is not as strong as ensuring that there exists a minimum time in between triggers $t_{\ell+1}^i - t_\ell^i \geq \tau^{\min} > 0$, which is a more pragmatic constraint when considering physical hardware. ■

In light of Remark 7.6.6, we consider enforcing a minimum time between events in the following section.

7.7 Periodic Event-Triggered Coordination

Throughout this chapter, we have assumed that all event triggers can be evaluated continuously. That is, the exact moment at which a triggering condition is met, an action (e.g., state broadcast and control signal update) is carried out. However, this may still be an unrealistic assumption when considering digital implementations. More specifically, a physical device cannot continuously evaluate whether a triggering condition has occurred or not. This observation motivates the need for studying sampled-data (or periodically checked) event-triggered coordination strategies.

Specifically, given a sampling period $h \in \mathbb{R}_{>0}$, we let $\{t_{\ell'}\}_{\ell' \in \mathbb{Z}_{\geq 0}}$, where $t_{\ell'+1} = t_\ell' + h$, denote the sequence of times at which agents evaluate the decision of whether to broadcast their state to their neighbors. This type of design is more in line with the constraints imposed by real-time implementations, where individual components work at some fixed frequency, rather than continuously. An inherent and convenient feature of this strategy is the lack of Zeno behavior (since inter-event times are naturally lower bounded by h).

Consequently, we begin by revisiting the result of Theorem 7.6.5. Intuitively, as long as the sampling period h is small enough, the closed-loop system with a periodically checked event-triggering condition will behave similarly to the system with triggers being evaluated continuously. The proof of convergence for the triggering law in Theorem 7.6.5 hinges on the fact that

$$e_i^2(t) \leq \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j \in \mathcal{N}_i} w_{ij}(\hat{x}_i(t) - \hat{x}_j(t))^2$$

for all times t. Instead, since we now assume that the triggering function 7.17 can only be evaluated periodically, we have that

$$e_i^2(t_{\ell'}) \leq \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j \in \mathcal{N}_i} w_{ij}(\hat{x}_i(t_{\ell'}) - \hat{x}_j(t_{\ell'}))^2 \tag{7.24}$$

is only guaranteed at the specific times $\{t_{\ell'}\}_{\ell' \in \mathbb{Z}_{\geq 0}}$ at which the triggering function can be evaluated. The algorithm is formalized in Table 7.7.

It should be noted that this algorithm is identical to the one in Table 7.6 except it is only executed periodically now rather than continuously. The following result then provides a sufficient condition on how small the period h has to be to guarantee convergence. The result is obtained by analyzing what happens to the Lyapunov function V in between these times.

Theorem 7.7.1 ***(Periodic event-triggered coordination)*** Given the closed-loop dynamics (7.16), if the communication graph $\mathcal{G}$ is weight balanced and the update times of each agent i are determined as the times $t' \in \{0, h, 2h, \dots\}$ when

$$f_i(e_i) \triangleq e_i^2 - \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j \in \mathcal{N}_i} w_{ij}(\hat{x}_i - \hat{x}_j)^2 \geq 0,$$

Table 7.7 Periodic event-triggered coordination on directed graphs.

```
At times $t \in \{0, h, 2h, \dots\}$, agent $i$ performs:
set $\hat{x}_i(t) = x_i(t_\ell^i)$
set $e_i(t) = \hat{x}_i(t) - x_i(t)$
**if** $e_i(t)^2 \geq \sigma_i \frac{1}{4d_i^{\text{out}}} \sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i(t) - \hat{x}_j(t))^2$ **then**
    set $t_{\ell+1}^i = t$
    broadcast $\hat{x}_i(t) = x_i(t_{\ell+1}^i)$ to neighbors $j \in \mathcal{N}_i^{\text{in}}$
    set $\ell = \ell + 1$
**end if**
set $u_i(t) = -\sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i(t) - \hat{x}_j(t))$
```

and $h \in \mathbb{R}_{>0}$ satisfies

$$\sigma_{\max} + 4hw_{\max}|\mathcal{N}_{\max}^{\text{out}}| < 1, \tag{7.25}$$

where $w_{\max} = \max_{i\in\{1,\ldots,N\},j\in\mathcal{N}_i^{\text{out}}} w_{ij}$ and $|\mathcal{N}_{\max}^{\text{out}}| = \max_{i\in\{1,\ldots,N\}} |\mathcal{N}_i^{\text{out}}|$, then the system achieves multi-agent average consensus.

Note that checking the sufficient condition (7.25) requires knowledge of the global quantities $\sigma_{\max}$, $w_{\max}$, and $\mathcal{N}_{\max}^{\text{out}}$. Ensuring that this condition is met can either be enforced a priori by the designer or, alternatively, the network can execute a distributed initialization procedure, for example, [3, 30], to compute these quantities in finite time. Once known, agents can compute h by instantiating a specific formula to select it that is guaranteed to satisfy (7.25).

A drawback of this algorithm is that the period h must be the same for all agents, requiring synchronous action. It is not difficult to envision asynchronous versions of this algorithm for which correctness guarantees have not currently been established.

7.8 Conclusions and Future Outlook

This chapter has presented a high-level overview of the ideas behind event-triggered communication and control applied to multi-agent average consensus problems. Although Table 7.1 makes it look like there is a complete story concerning event-triggered consensus problems, this is certainly not true as there still remain many issues to be addressed regarding asynchronism, guarantees on non-Zeno behavior, and practical considerations. There are indeed still exciting new directions being explored at the time of writing that would only serve to expand this table in the future. For instance, all event-triggered protocols discussed in this chapter assume that all agents are able to “listen” for incoming messages at all times. In other words, when a message is broadcast by an agent i, this message is immediately received by a neighboring agent $j \in \mathcal{N}_i$ who immediately (or within some reasonable time due to delays, etc.) reacts to this event by updating its control signal. However, this may not be possible in all scenarios, which presents a whole new set of technical challenges. For example, some recent preliminary results have been developed in this setup motivated by the need for coordinating submarines [31, 32], where agents can only communicate when they are at the surface of the water. While a submarine is submerged, any message broadcast by another submarine cannot be received until it resurfaces.

References

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

2 Bullo, F., Cortés, J., and Martínez, S. (2009) *Distributed Control of Robotic Networks*, Applied Mathematics Series, Princeton University Press. Electronically available at http://coordinationbook.info (accessed 21 October 2016).

3 Ren, W. and Beard, R.W. (2008) *Distributed Consensus in Multi-Vehicle Cooperative Control*, Communications and Control Engineering, Springer-Verlag.

4 Mesbahi, M. and Egerstedt, M. (2010) *Graph Theoretic Methods in Multiagent Networks*, Applied Mathematics Series, Princeton University Press.

5 øAström, K.J. and Bernhardsson, B.M. (2002) Comparison of Riemann and Lebesgue sampling for first order stochastic systems, in IEEE Conference on Decision and Control, Las Vegas, NV, pp. 2011–2016.

6 Heemels, W.P.M.H., Johansson, K.H., and Tabuada, P. (2012) An introduction to event-triggered and self-triggered control, in IEEE Conference on Decision and Control, Maui, HI, pp. 3270–3285.

7 Mazo, M. Jr., and Tabuada, P. (2011) Decentralized event-triggered control over wireless sensor/actuator networks. *IEEE Transactions on Automatic Control*, **56** (10), 2456–2461.

8 Wang, X. and Lemmon, M.D. (2011) Event-triggering in distributed networked control systems. *IEEE Transactions on Automatic Control*, **56** (3), 586–601.

9 Stöker, C., Vey, D., and Lunze, J. (2013) Decentralized event-based control: stability analysis and experimental evaluation. *Nonlinear Analysis: Hybrid Systems*, **10**, 141–155.

10 Xie, G., Liu, H., Wang, L., and Jia, Y. (2009) Consensus in networked multi-agent systems via sampled control: fixed topology case, in American Control Conference, St. Louis, MO, pp. 3902–3907.

11 Heemels, W.P.M.H. and Donkers, M.C.F. (2013) Model-based periodic event-triggered control for linear systems. *Automatica*, **49** (3), 698–711.

12 Meng, X. and Chen, T. (2013) Event based agreement protocols for multi-agent networks. *Automatica*, **49** (7), 2125–2132.

13 Zhong, M. and Cassandras, C.G. (2010) Asynchronous distributed optimization with event-driven communication. *IEEE Transactions on Automatic Control*, **55** (12), 2735–2750.

14 Anta, A. and Tabuada, P. (2010) To sample or not to sample: self-triggered control for nonlinear systems. *IEEE Transactions on Automatic Control*, **55** (9), 2030–2042.

15 Wang, X. and Lemmon, M.D. (2009) Self-triggered feedback control systems with finite-gain L_2 stability. *IEEE Transactions on Automatic Control*, **54** (3), 452–467.

16 Nowzari, C. and Cortés, J. (2016) Team-triggered coordination for real-time control of networked cyber-physical systems. *IEEE Transactions on Automatic Control*, **61** (1), 34–47.

17 Dimarogonas, D.V., Frazzoli, E., and Johansson, K.H. (2012) Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, **57** (5), 1291–1297.

18 Liu, Z., Chen, Z., and Yuan, Z. (2012) Event-triggered average-consensus of multi-agent systems with weighted and direct topology. *Journal of Systems Science and Complexity*, **25** (5), 845–855.

19 Seyboth, G.S., Dimarogonas, D.V., and Johansson, K.H. (2013) Event-based broadcasting for multi-agent average consensus. *Automatica*, **49** (1), 245–252.

20 Garcia, E., Cao, Y., Yu, H., Antsaklis, P., and Casbeer, D. (2013) Decentralised event-triggered cooperative control

with limited communication. *International Journal of Control*, **86** (9), 1479–1488.

21 Guo, G., Ding, L., and Han, Q. (2014) A distributed event-triggered transmission strategy for sampled-data consensus of multi-agent systems. *Automatica*, **50**, 1489–1496.

22 Meng, X., Xie, L., Soh, Y.C., Nowzari, C., and Pappas, G.J. (2015) Periodic event-triggered average consensus over directed graphs, in IEEE Conference on Decision and Control, Osaka, Japan, pp. 4151–4156.

23 Nowzari, C. and Cortés, J. (2016) Distributed event-triggered coordination for average consensus on weight-balanced digraphs. *Automatica*, **68**, 237–244.

24 Mazo, M. Jr.,, Anta, A., and Tabuada, P. (2009) On self-triggered control for linear systems: guarantees and complexity, in European Control Conference, Budapest, Hungary, pp. 3767–3772.

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

26 Aragues, R., Shi, G., Dimaragonas, D.V., Sagues, C., and Johansson, K.H. (2012) Distributed algebraic connectivity estimation for adaptive event-triggered consensus, in American Control Conference, Montreal, Canada, pp. 32–37.

27 Yang, P., Freeman, R.A., Gordon, G.J., Lynch, K.M., Srinivasa, S.S., and Sukthankar, R. (2010) Decentralized estimation and control of graph connectivity for mobile sensor networks. *Automatica*, **46** (2), 390–396.

28 Gharesifard, B. and Cortés, J. (2012) Distributed strategies for generating weight-balanced and doubly stochastic digraphs. *European Journal of Control*, **18** (6), 539–557.

29 Rikos, A.I., Charalambous, T., and Hadjicostis, C.N. (2014) Distributed weight balancing over digraphs. *IEEE Transactions on Control of Network Systems*, **1** (2), 190–201.

30 Lynch, N.A. (1997) *Distributed Algorithms*, Morgan Kaufmann.

31 Adaldo, A., Liuzza, D., Dimarogonas, D.V., and Johansson, K.H. (2015) Control of multi-agent systems with event-triggered cloud access, in European Control Conference, Linz, Austria, pp. 954–961.

32 Nowzari, C. and Pappas, G.J. (2016) Multi-agent coordination with asynchronous cloud access, in American Control Conference, Boston, MA, to appear.

33 Khalil, H.K. (2002) *Nonlinear Systems*, Prentice Hall, 3rd edn.

Appendix

Proof of Theorem 7.3.1: Consider the Lyapunov function

$$V(x) = \frac{1}{2}x^T Lx.$$

Then, given the dynamics (7.8) and the continuous control law (7.9),

$$\dot{V}(x) = x^T L\dot{x} = -x^T L^T Lx = -\|Lx\|^2,$$

where we have used the fact that L is symmetric. It is now clear that using the continuous control law (7.9) we have $\dot{V}(x) < 0$ for all $Lx \neq 0$. Using LaSalle's Invariance Principle [33], it can then be shown that

$$x(t) \to \{Lx = 0\} = \{x_i = x_j \forall i, j \in \{1, \dots, N\}\}$$

as $t \to \infty$. Combining this with the fact that the sum of all states is an invariant quantity concludes the proof,

$$\frac{d}{dt}(\mathbf{1}_N^T x(t)) = \mathbf{1}_N^T \dot{x}(t) = -\mathbf{1}_N^T Lx(t) = 0.$$

■

Proof of Theorem 7.6.2: Let $\delta(t) = x(t) - \overline{x}\mathbf{1}$, where $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$ is the average of all initial conditions. Then, $\dot{\delta}(t) = -L\delta(t) - Le(t)$, yielding

$$\delta(t) = e^{-Lt}\delta(0) - \int_0^t e^{-L(t-s)}Le(s)ds.$$

Taking norms,

$$\begin{aligned}\|\delta(t)\| &\leq \|\delta(0)e^{-Lt}\| + \int_0^t \|e^{-L(t-s)}Le(s)\|ds \\ &\leq e^{-\lambda_2(L)t}\|\delta(0)\| + \int_0^t e^{-\lambda_2(L)(t-s)}\|Le(s)\|ds,\end{aligned}$$

where the second inequality follows from [19, Lemma 2.1].

Using the condition

$$|e_i(t)| \leq c_0 + c_1 e^{-\alpha t},$$

it follows that

$$\begin{aligned}
\|\delta(t)\| &\leq e^{-\lambda_2 t}\|\delta(0)\| + \|L\|\sqrt{N} \\
&\quad \int_0^t e^{-\lambda_2(t-s)}(c_0 + c_1 e^{-\alpha s})ds \\
&= e^{-\lambda_2 t}\left(\|\delta(0)\| - \|L\|\sqrt{N}\left(\frac{c_0}{\lambda_2} + \frac{c_1}{\lambda_2 - \alpha}\right)\right) \\
&\quad + e^{-\alpha t}\frac{\|L\|\sqrt{N}c_1}{\lambda_2 - \alpha} + \frac{\|L\|\sqrt{N}c_0}{\lambda_2}.
\end{aligned}$$

The convergence result then follows by taking $t \to \infty$. ■

Proof of Theorem 7.7.1: Since (7.24) is only guaranteed at the sampling times under the periodic event-triggered coordination algorithm presented in Table 7.7, we analyze what happens to the Lyapunov function V in between them. For $t \in [t_{\ell'}, t_{\ell'+1})$, note that

$$e(t) = e(t_{\ell'}) + (t - t_{\ell'})L\hat{x}(t_{\ell'}).$$

Substituting this expression into $\dot{V}(t) = -\hat{x}^T(t)L\hat{x}(t) + e^T(t)L\hat{x}(t)$, we obtain

$$\begin{aligned}
\dot{V}(t) &= -\hat{x}^T(t_{\ell'})L\hat{x}(t_{\ell'}) + e^T(t_{\ell'})L\hat{x}(t_{\ell'}) \\
&\quad + (t - t_{\ell'})\hat{x}^T(t_{\ell'})L^T L\hat{x}(t_{\ell'}),
\end{aligned}$$

for all $t \in [t_{\ell'}, t_{\ell'+1})$. For a simpler exposition, we drop all arguments referring to time $t_{\ell'}$ in the sequel. Following a similar discussion to Section 7.6, it can be shown that

$$\dot{V}(t) \leq \sum_{i=1}^{N} \frac{\sigma_i - 1}{4} \sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i - \hat{x}_j)^2 + (t - t_{\ell'})\hat{x}^T L^T L\hat{x}.$$

Note that the first term is exactly what we have when we are able to monitor the triggers continuously (7.22).

Using the fact that $\left(\sum_{k=1}^{p} y_k\right)^2 \leq p\sum_{k=1}^{p} y_k^2$ (which follows directly from the Cauchy–Schwarz inequality), we bound

$$\hat{x}^T L^T L\hat{x} = \sum_{i=1}^{N}\left(\sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i - \hat{x}_j)\right)^2$$

$$\leq \sum_{i=1}^{N} |\mathcal{N}_i^{\text{out}}| w_i^{\max} \sum_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i - \hat{x}_j)^2$$

$$= |\mathcal{N}_{\max}^{\text{out}}| w_{\max} \sum_{i=1}^{N} \sum_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i - \hat{x}_j)^2, \tag{7.26}$$

where $w_i^{\max} = \max_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}$. Hence, for $t \in [t_{\ell'}, t_{\ell'+1})$,

$$\dot{V}(t) \leq \sum_{i=1}^{N} \left(\frac{\sigma_i - 1}{4} + h w_{\max} |\mathcal{N}_{\max}^{\text{out}}| \right) \sum_{j\in\mathcal{N}_i^{\text{out}}} w_{ij}(\hat{x}_i - \hat{x}_j)^2$$

$$\leq \left(\frac{\sigma_{\max} - 1}{2} + 2h w_{\max} |\mathcal{N}_{\max}^{\text{out}}| \right) \hat{x}^T L \hat{x}.$$

Then, by using (7.25), it can be shown that there exists $\mathcal{B} > 0$ such that

$$\dot{V}(t) \leq \frac{1}{2\mathcal{B}} \left(\sigma_{\max} + 4h w_{\max} |\mathcal{N}_{\max}^{\text{out}}| - 1 \right) V(x(t)),$$

which implies the result. See [23] for details. ■

8

Topology Design and Identification for Dynamic Networks

Chuangchuang Sun and Ran Dai

The Aerospace Engineering Department, Iowa State University, Ames, IA, USA

8.1 Introduction

Network structure/topology is essential in many types of multi-agent involved missions, where interaction among agents is determined by the underlying network structure. For example, successful control of leader–follower network toward desired states requires identifying node indices of leaders, connectivity between leaders and followers, and interaction relationship among followers [1]. In fact, designing or obtaining network topology is a prerequisite for operations such as spacecraft formation, mobile robot rendezvous, unmanned aerial vehicle flocking [2]. In this chapter, we address two types of problem, network topology design (NTD) and network topology identification (NTI), both of which are classified as NP-hard problems.

Related works in the area of network design have been pursued predominately for allocating edge weights [3–5]. For geometry-dependent networks, efforts have been focused on determining the location of mobile nodes to establish or maintain a connected network [6, 7]. The work described here focuses on the combinatorial problem of determining both the topology and edge weights of a nongeometric undirected network that will optimize a desired performance index with constraint on the cardinality of edge set. In practical applications, that is, establishing a communication network, the cost of constructing

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

a connected network is evaluated by the number of edges. Therefore, considering cardinality constraint on the edge set has economic impacts and benefits. For a nongeometric-based network, if the existence and absence of an edge between a pair of nodes is represented by binary values, then the off-diagonal entries of the adjacency matrix are all expressed as binary values as well. The objective is to determine these binary values and their associated weights in order to optimize the performance index when the number of edges has an upper bound.

Due to the cardinality constraint on the edge set, the NTD problem can be cast as an optimal resource allocation problem, where the edge set is handled as limited resources. We refer to such problems as cardinality-constrained optimization problems. Cardinality-constrained optimization problems have extensive applications in feedback control and networked sensing systems. Due to the nonconvex cardinality function, a surrogate model, that is, L_1 norm function, is generally used to approximately represent the cardinality function [8–11]. In general, the L_1 norm method is able to find at least a feasible solution when aiming to design a sparse linear system [12–14]. However, when the cardinality function appears in the constraint, these surrogate models are no longer applicable to cardinality-constrained optimization problems, as an approximate model cannot guarantee satisfying the exact cardinality inequality constraint. Heuristic search, that is, genetic algorithm [15, 16], and branch-and-bound method [17, 18] have been applied to small-scale NTD as both of them are time consuming. Based on the fact that cardinality of a vector equals to the rank of a diagonal matrix with such vector as the diagonal elements, the cardinality-constrained optimization is a special case of rank-constrained optimization problem (RCOP).

Meanwhile, extensive work has been developed in the area of identification for a linear time-invariant (LTI) system, which is controllable and observable. The well-known Kung et al. [19] and subspace methods [20] find "similar" state-space matrices that match the response between input and output. Such a "black box" model constructed by input–output data is applied in this chapter for solving NTI problems. We assume that the dynamics of each node in a connected network is governed by the consensus protocol. Thus, the linear transfer matrix of LTI systems is defined by the negative graph Laplacian that directly reflects the network topology. However, finding “similar” state-space matrices is far away from the objective of identifying the exact network topology. Previous work in [21] used the “black-box” setup to establish the generating function of graph Laplacian by observing input–output data from selected network nodes. Another work in [22] used an integrated spectral characterization of graphs and similarity transformation approach to find an approximate graph Laplacian. Reconstruction of tree-like networks and sparse networks can be found in the recent work of [23–26]. However, mapping the exact graph Laplacian from the “similar” state-space matrices constructed by input–output data is very challenging. The major reason comes from the involvement of unknown binary variables representing the edge set of the network. For a network with given number of nodes, the number of network topology configurations is an exponential function about its number of nodes. It will be extremely time consuming to find the exact graph Laplacian that has the closest spectra to the “similar” state transfer matrix while keeping the similarity properties.

The similarity transformation between graph Laplacian and “similar” state-space matrices as bilinear constraints. The NTI problem is then transformed as an optimization problem with bilinear constraints, as well as constraints on the Laplacian structure. We further generalize the NTI problem as a nonconvex quadratically constrained quadratic programming (QCQP) problem. A general QCQP can be equivalently transformed into a linear matrix programming problem by introducing a to-be-determined rank-one matrix [27]. Through these conversions, the NTI problem is also equivalently formulated as an RCOP, where the constrained rank is equal to 1.

RCOPs are to minimize a convex function subject to a convex set of constraints and rank constraints on unknown matrices. They have received extensive attention due to their wide applications in signal processing, model reduction, and system identification, just to name a few [28–30]. Due to the discontinuous and nonconvex nature of the rank function, most of the existing methods solve relaxed or simplified RCOPs by introducing an approximate function, that is, log-det or nuclear norm heuristic methods [31, 32]. However, a relaxed function cannot represent the exact rank function and performance of the heuristic method is not guaranteed in general. In the worst case, when the unknown matrix is positive semidefinite and has equality trace constraint, the nuclear norm heuristic method is not applicable to such type of problems. Other methods for RCOPs mainly focus on alternating projection-based methods [33–36] and combined linearization and factorization algorithms [37, 38]. However, these iterative approaches depend on the initial guess, and fast convergence cannot be guaranteed.

After reviewing the literature, we come to a conclusion that a more efficient approach for general RCOPs is required. This chapter introduces an iterative rank minimization (IRM) method to solve RCOPs, where each iteration is formulated as a semidefinite programming (SDP) problem. To validate the effectiveness and efficiency of the proposed algorithm, we apply IRM to two representative NTD problems and one NTI problem where dynamics of each node is driven by consensus protocol. One type of NTD is to pursue a fast convergence rate of consensus-based multi-agent system and the other is to minimize the total effective resistance of an electrical resistor network. For small-scale NTD and NTI problems where global optimal solution can be determined by exhaustive examination of all feasible solutions, results from IRM are compared to the best result from exhaustive search.

This chapter is organized as follows. Section 8.2 introduces the formulations of two types of NTD problems. In Section 8.3, we formulate the NTI problem and transform the original formulation into QCQPs and then rank-one constrained optimization problems. We then introduce an iterative method in Section 8.4 to solve RCOPs. Simulation results for NTD and NTI with different nodes are presented in Section 8.5. We conclude the chapter in Section 8.6.

8.2 Network Topology Design Problems

We consider an undirected, weighted network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V} = \{1, 2, \dots, n\}$ and edge set E consisting of two element subsets of $\mathcal{V}$. We use nodes or agents interchangeably with vertices. The Laplacian matrix of $\mathcal{G}$ is defined as

$\mathcal{L} = \mathcal{A}\mathbf{diag}(\mathbf{g})\mathcal{A}^T$, where $\mathcal{A} \in \mathbb{R}^{n\times m}$ is the incidence matrix, $\mathbf{g} \in \mathbb{R}^m$ are weights of the edge set, and $m \leq n(n-1)/2$. Note that if $\{v_i, v_j\} \notin \mathcal{E}$, the corresponding edge weight is 0.

8.2.1 Network Design for Fast Convergence of Consensus Protocol

For a multi-agent system, the consensus protocol of the overall system is represented as

$$\dot{\mathbf{x}} = -\mathcal{L}(\mathcal{G})\mathbf{x}, \tag{8.1}$$

which will drive states of each agent to the consensus set $\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{x}_i = \mathbf{x}_j, \forall\ v_i, v_j \in \mathcal{V}\}$ by exchanging state information with connected agents in the specified network $\mathcal{G}$. It is well known that $\mathcal{L}(\mathcal{G})$ is positive semidefinite, and we sort the eigenvalues of $\mathcal{L}(\mathcal{G})$ in the nondecreasing order as $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. We quote the following well-known Lemma [39], which can be used to determine the rate of convergence to the consensus set. It is known that for a connected network $\mathcal{G}$, the (undirected) consensus protocol converges to the consensus set $\mathcal{C}$ with a rate of convergence that is dictated by $\lambda_2(\mathcal{G})$.

With $\lambda_2(\mathcal{L})$ assigned as the objective to be maximized, our goal is to design the weighted topology of a connected network such that the sum of the edges weights and cardinality of the edge set are bounded. Mathematically, this type of NTD problem can be formulated as

$$\begin{aligned} J &= \max\nolimits_g \lambda_2(\mathcal{L}) \\ s.t.\ \ g_j &\geq 0, j = 1, \ldots, m \\ \mathbf{1^T g} &= \mathbf{g_t} \\ \lambda_2(\mathcal{L}) &> 0 \\ \mathbf{Card}(\mathbf{g}) &\leq r, \end{aligned} \tag{8.2}$$

where $\mathbf{g_t}$ is the upper bound on sum of the weights, r is the upper bound on cardinality of $\mathbf{g}$, and $\mathbf{Card}(\bullet)$ denotes the cardinality of "$\bullet$." However, λ_2 in (8.2) is not an explicit function of the design variables, $\mathbf{g}$. In addition, computing the second smallest eigenvalue of a to-be-determined matrix is computationally complicated and nonlinear. A similarity transformation method is discussed later to transform the network connectivity constraint into a linear matrix inequality (LMI) constraint.

An orthogonal matrix $P = [p_1, p_2, \dots, p_{n-1}, \mathbf{1}/\sqrt{n}]$ is constructed here with unit vectors p_i's chosen as $p_i^T\mathbf{1} = 0$ $(i = 1, 2, \dots, n-1)$ and $p_i^T p_j = 0$ $(i \neq j)$. By the similarity transformation, we get

$$\mathcal{L} \sim P^T \mathcal{L}(\mathcal{G}) P, \tag{8.3}$$

where symbol "$\sim$" indicates the similarity between two matrices.

Lemma 8.2.1 For a graph Laplacian $\mathcal{L}(\mathcal{G})$, $\alpha > \lambda_2$ if and only if $\mathcal{L}(\mathcal{G}) + \alpha\mathbf{11^T/n} \geq \lambda_2\mathbf{I}$ [17].

Proof: It is easy to confirm that the matrix $\alpha\mathbf{11}^T/n$ has one eigenvalue equal to α with corresponding eigenvector of $\mathbf{1}$ and the remaining eigenvalues are all equal to zero. Let us assume that the eigenvectors of matrix $\alpha\mathbf{11}^T/n$ are denoted by $P = [p_1, p_2, \dots, p_{n-1}, \mathbf{1}/\sqrt{n}]$, where all elements of P satisfies conditions stated earlier. From (8.3), one has

$$P^T \mathcal{L}(\mathcal{G}) P \sim P^T \begin{pmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{pmatrix} P,$$

where $\lambda_1 = 0$ with eigenvector $\mathbf{1}$. We can now proceed to determine the eigenvalues of matrix $\mathcal{L}(\mathcal{G}) + \alpha\mathbf{11}^T/n$,

$$P^T(\mathcal{L}(\mathcal{G}) + \alpha\mathbf{11}^T/n)P \sim P^T \begin{pmatrix} \lambda_1 + \alpha & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_n \end{pmatrix} P.$$

As we assigned $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$, $\mathcal{L}(\mathcal{G}) + \alpha\mathbf{11}^T/n \geq \lambda_2 I$ is satisfied if and only if $\alpha \geq \lambda_2$. ■

Based on the LMI representation of connectivity constraint, the original NTD problem is reformulated as

$$\begin{aligned} & J = \max_{\mathbf{g},\lambda_2} \lambda_2 \\ s.t. \quad & \mathbf{g} \geq 0 \\ & \mathbf{1^T g = g_t} \\ & \lambda_2(\mathcal{L}) > 0 \\ & \mathbf{Card(g)} \leq r \\ & \mathcal{L}(\mathcal{G}) + \alpha\mathbf{11^T/n} \geq \lambda_2\mathbf{I}, \end{aligned} \tag{8.4}$$

where λ_2 is handled as an unknown variable to be determined together with **g** and α is selected as a large number. As we want to maximize λ_2 and $\mathcal{L}(\mathcal{G}) + \alpha\mathbf{1}\mathbf{1}^{\mathrm{T}}/\mathbf{n}$ acts as an upper bound on λ_2, formulation in (8.4) is an equivalent conversion of the original problem formulated in (8.2). The objective function and constraints, except for the cardinality constraint, are convex, thus problem in (8.4) is classified a cardinality-constrained optimization problem.

8.2.2 Network Design for Minimum Total Effective Resistance

In a connected resistor network, the effective resistance between a pair of vertices is evaluated by the conductance of each edge in the network. In practical applications, that is, infusing control commands into the network, it is necessary to minimize the total effective resistance such that maximum currents between two vertices can be obtained. The authors of [40] determined that the design of edge weights of a network with given topology while ignoring cardinality constraints is a convex problem. In this chapter, we consider a similar problem to design the weighted network topology with an upper bound on **Card**(**g**). Following the formulation of network design problem for minimum total effective resistance in [40], our problem considering cardinality constraint on edge set can be expressed as

$$\begin{aligned} J &= \min\nolimits_{\mathbf{g},Y}\ n\,\mathbf{trace}(Y) \\ s.t. \quad & \mathbf{1}^{\mathrm{T}}\mathbf{g} = \mathbf{g}_{\mathrm{t}} \\ & g_j \geq 0, j = 1, \ldots, m \\ & \begin{bmatrix} \mathcal{L} + \mathbf{1}\mathbf{1}^{T}/\mathbf{n} & I \\ I & Y \end{bmatrix} \succeq \mathbf{0} \\ & \mathbf{Card}(\mathbf{g}) \leq r, \end{aligned} \tag{8.5}$$

where $Y \in \mathbb{S}^n$ is a slack matrix, $\mathbb{S}$ denotes a symmetric matrix, and the rest variables are defined similar to those in problem (8.2). Moreover, work in [40] indicated that the optimum of (8.5) is a complete graph without the cardinality constraint on edge set. It is obvious that the weighted NTD problem formulated in (8.5) is again a cardinality-constrained optimization problem.

8.2.3 Equivalent Conversion from Cardinality-Constrained Optimization Problems to RCOPs

The two types of NTD problems described in Sections 8.2.1 and 8.2.2 can be classified as a general cardinality-constrained optimization problem formulated as

$$\begin{aligned} &J = \min_{\mathbf{x}} f(\mathbf{x}) \\ s.t. \quad &\mathbf{x} \in C \\ &\mathbf{Card}(\mathbf{x}) \leq r, \end{aligned} \tag{8.6}$$

where $\mathbf{x} \in \mathbb{R}^p$, $f(\mathbf{x})$ is a convex function, and C is a convex set. Based on the fact that cardinality of a vector $\mathbf{x}$ equals to the rank of a diagonal matrix with diagonal elements $\mathbf{x}$, the cardinality function can be equivalently reformulated as $\mathbf{Card}(\mathbf{x}) = \mathbf{rank}(\mathbf{diag}(\mathbf{x}))$. Then a cardinality-constrained optimization problem can be equivalently transformed into a RCOP in the form of

$$\begin{aligned} &J = \min_{X} f(X) \\ s.t. \quad &X \in C \\ &\mathbf{rank}(X) \leq r, \end{aligned} \tag{8.7}$$

where $X \in \mathbb{S}_+^p$ is an unknown positive semidefinite matrix.

8.3 Network Topology Identification Problems

8.3.1 LTI System Identification

We start with introducing the black-box approach of identifying an LTI system by observing input–output data. Consider a linear continuous system in state-space format

$$\begin{aligned} \dot{\mathbf{x}}(t) &= A\mathbf{x}(t) + B\mathbf{u}(t) \\ \mathbf{y}(t) &= C\mathbf{x}(t) \end{aligned}, \tag{8.8}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^{r_\mathcal{I}}$ is the input vector, $\mathbf{y} \in \mathbb{R}^{r_\mathcal{O}}$ is the output vector, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r_\mathcal{I}}$, and $C \in \mathbb{R}^{r_\mathcal{O}\times n}$ are the state-space matrices. Through discrete integration, the continuous linear differential equation in (8.8) can be integrated at discrete sampling time $t_k, k = 1, \ldots, K$, and reformulated as discrete LTI system in the format

$$\begin{aligned}\mathbf{x}(t_{k+1}) &= A_d\mathbf{x}(t_k) + B_d\mathbf{u}(t_k)\\ \mathbf{y}(t_k) &= C_d\mathbf{x}(t_k)\end{aligned}, \tag{8.9}$$

where $A_d = e^{t_k A}$, $B_d = (\int_0^{t_k} e^{At}dt)B$, and $C_d = C$. For a given A_d, $A = 1/t_k \log_M A_d$, where $\log_M$ denotes the matrix logarithm. By propagation, the input–output expression for discrete LTI at sampling sequence p is expressed as

$$\mathbf{y}(p) = H(p)\mathbf{u}(p), \tag{8.10}$$

where $H(p) = \Gamma(p)\Omega(p)$ denotes the Hankel matrix and matrices $\Gamma(p)$ and $\Omega(p)$ are defined by the A_d, B_d, C_d in (8.9) as $\Gamma(p) = \begin{bmatrix} C_d \\ C_dA_d \\ \vdots \\ C_dA_d^{p-1}\end{bmatrix}$ and $\Omega(p) = \begin{bmatrix} B_d & A_dB_d & \dots & A_d^{p-1}B_d\end{bmatrix}$, respectively. By tracking a series of input and output data at discrete sampling time, the Hankel matrix can be constructed via impulse response parameters [19] or least-square estimation method [20]. There are many methods to reconstruct the extended controllability and observability matrices, $\Gamma(p)$ and $\Omega(p)$, from the Hankel matrix. For example, a commonly used one is the singular value decomposition method [19]. Results of this identification procedure lead to the realization of "similar" state-space matrices set (A_T, B_T, C_T) that represents the identified LTI system

$$\begin{aligned}\dot{\mathbf{x}}(t) &= A_T\mathbf{x}(t) + B_T\mathbf{u}(t)\\ \mathbf{y}(t) &= C_T\mathbf{x}(t)\end{aligned} \tag{8.11}$$

while satisfying the input–output response. Furthermore, from Kalman's theorem [41], the set (A, B, C) and (A_T, B_T, C_T) are linearly related by a nonsingular matrix $T \in \mathbb{R}^{n\times n}$ such that

$$A_T = TAT^{-1}, \quad B_T = TB, \quad C = CT^{-1}. \tag{8.12}$$

This system identification procedure can be applied to the NTI problem where the state-space transfer matrix is solely determined by the network topology.

We assume that the nodes belonging to the set $\mathcal{I} \in \mathcal{V}$, which has cardinality $r_{\mathcal{I}}$, are selected to receive infused control signals $\mathbf{u}$. Meanwhile, nodes set $\mathcal{O} \in \mathcal{V}$ with cardinality $r_{\mathcal{O}}$ is used to observe output response $\mathbf{y}$. Recall that when $\mathbf{x}_i$ denotes the state of dynamic agent i in the connected network $\mathcal{G}$, the consensus protocol of the overall system with input $\mathbf{u}$ is represented by

$$\dot{\mathbf{x}}(t) = -\mathcal{L}(\mathcal{G})\mathbf{x}(t) + B\mathbf{u}(t), \tag{8.13}$$

which will drive each agent to the consensus set $\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{x}_i = \mathbf{x}_j, \forall\, v_i, v_j \in \mathcal{V}\}$ by exchanging state information with connected agents in the specified network $\mathcal{G}$. $B \in \mathbb{R}^{n \times r_{\mathcal{I}}}$ is the control matrix with element $B_{i,j}$ set as one if node i, $i = 1, \dots, n$, is selected as the jth input, where $j = 1, \dots, r_{\mathcal{I}}$. For the output, we have

$$\mathbf{y}(t) = C\mathbf{x}(t), \tag{8.14}$$

where $C \in \mathbb{R}^{r_{\mathcal{O}} \times n}$ is the observation matrix with element $C_{j,i}$ set as one if node i, $i = 1, \dots, n$, is selected as the jth output, where $j = 1, \dots, r_{\mathcal{O}}$. An illustrative example is demonstrated in Figure 8.1.

Here, we assume that the LTI system governed by (8.13) and (8.14) is controllable and observable. Furthermore, the network to be identified is connected. Under these assumptions, the system identification procedure introduced earlier is applied to construct the "similar" state-space matrices set (A_T, B_T, C_T). The detailed steps of constructing (A_T, B_T, C_T) can be referred to [22]. However, the system identification procedure does not lead to the exact A, B, C matrices of the original system, nor does the matrix A_T share the graph Laplacian properties. Our

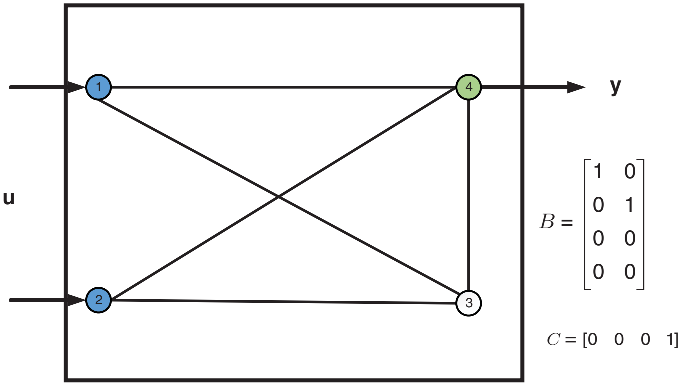

u
y
1
2
3
4
$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$
$C = [0 \quad 0 \quad 0 \quad 1]$

Figure 8.1 An illustrative example of NTI concept.

focus is to find the exact graph Laplacian from the obtained (A_T, B_T, C_T) set such that the final state-space matrices $(-\mathcal{L}(\mathcal{G}), B, C)$ satisfy the linear similarity relationship stated in (8.12) while constraining the graph Laplacian $\mathcal{L}(\mathcal{G})$ with the defined structure.

8.3.2 Formulation of NTIs as QCQPs

In summary, the NTI problem is to find $(-\mathcal{L}(\mathcal{G}), B, C)$ from constructed matrices set (A_T, B_T, C_T) obtained through system identification procedure. The desired matrix $-\mathcal{L}(\mathcal{G})$, sharing the same spectrum of A_T, has the minimum distance from A_T in the Frobenius norm [42]. In addition, the matrices set $(-\mathcal{L}(\mathcal{G}), B, C)$ is constrained by the similarity relationship and $\mathcal{L}(\mathcal{G})$ is constrained by the Laplacian structure. The aforementioned statement of NTI can be summarized as a constrained optimization problem, written as

$$\begin{aligned} & J = \min_{A,T} \ \|A - A_T\|_F^2 \\ s.t. \quad & A_T = TAT^{-1} \\ & B_T = TB \\ & C_T = CT^{-1} \\ & A_{i,j} = A_{j,i} \in \{0,1\}, \forall\ v_i, v_j \in V, i \neq j \\ & A\mathbf{1} = \mathbf{0}, \end{aligned} \tag{8.15}$$

where $A = -\mathcal{L}(\mathcal{G})$. The aforementioned formulation including both binary variables in A and continuous variables in T (which is nonsingular as mentioned before) under nonlinear constraints is a mixed-integer nonlinear optimization problem. In addition, as A_T is supposed to be similar to a negative Laplacian matrix of a graph, it must have the same eigenstructure, namely, all eigenvalues should be negative except the one which is zero. This can be used to check the feasibility of (8.15) or the system identification procedures. For a given matrix $\Phi \in \mathbb{R}^{n\times n}$, its Frobenius norm is determined by $\|\Phi\|_F := \sqrt{\mathbf{trace}(\Phi^T\Phi)} = \left(\sum_{i=1}^{n}\sum_{j=1}^{n}\Phi_{i,j}^2\right)^{\frac{1}{2}}$. Furthermore, the binary variables $A_{i,j}$ can be constrained via quadratic functions $A_{i,j}(A_{i,j} - 1) = 0, \forall\ v_i, v_j \in V, i \neq j$. With these transformations, problem in (8.15) can be transformed into a quadratic optimization problem with quadratic and linear constraints in the form of

$$\begin{aligned} J = \min\nolimits_{A,T} \sum_{i=1}^{n}\sum_{j=1}^{n} (A_{i,j} - A_{T\,i,j})^2 \\ s.t. \qquad TA - A_T T = \mathbf{0} \\ B_T = TB \\ C_T T = C \\ A_{i,j}(A_{i,j} - 1) = 0, \forall\ v_i, v_j \in V, i \neq j \\ A_{i,j} = A_{j,i} \\ A\mathbf{1} = \mathbf{0}. \end{aligned} \tag{8.16}$$

The first three constraints are from (8.12) and the following three are the properties of a graph Laplacian matrix. It is obvious that the quadratic constraints such as the binary variable constraints and bilinear constraints in (8.16) are nonconvex. The NTI problem is then transformed as a nonconvex inhomogeneous QCQP problem.

8.3.3 Equivalent Conversion from QCQPs to RCOPs

In general, the inhomogeneous QCQP problem can be formulated as

$$\begin{aligned} J = \min \mathbf{x}^T Q_0 \mathbf{x} + \mathbf{q}_0{}^T \mathbf{x} \\ s.t. \quad \mathbf{x}^T Q_j \mathbf{x} + \mathbf{q}_j{}^T \mathbf{x} + \mathbf{c}_j \leq 0, \ \forall\, j = 1, \ldots, m, \end{aligned} \tag{8.17}$$

where $\mathbf{x} \in \mathbb{R}^p, Q_j \in \mathbb{S}^p, \mathbf{q}_j \in \mathbb{R}^p, j = 0, \ldots, m$. In particular, linear constraints can be included as well if the Hessian matrix, Q_j, is set as a zero matrix. For the cases where Q_j is negative semidefinite, indefinite, or the quadratic constraints have equality ones, the problem can be considered to be concave, which is NP-hard.

The aforementioned QCQP problem with inhomogeneous quadratic function can be transformed into homogeneous ones by introducing a new variable $t \in \mathbb{R}$ and a new quadratic constraint $t^2 = 1$ in the following formulation:

$$\begin{aligned} J = \min \begin{bmatrix} \mathbf{x}^T & t \end{bmatrix} \begin{bmatrix} Q_0 & q_0/2 \\ q_0^T/2 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} \\ s.t. \quad \begin{bmatrix} \mathbf{x}^T & t \end{bmatrix} \begin{bmatrix} Q_j & q_j/2 \\ q_j^T/2 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ t \end{bmatrix} + c_j \leq 0, \forall j = 1,\ldots,m, \\ t^2 = 1. \end{aligned}$$

Then, $\mathbf{x}^*/t^*$ will be the solution of the original problem stated in (8.17) while $(\mathbf{x}^*, t^*)$ is the solution pair of (8.18). In addition, linear constraints in (8.17) can be rewritten in the aforementioned quadratic form as well by setting $Q_j = \mathbf{0}$. The homogeneous QCQP problem is formulated as

$$\begin{aligned} J &= \min \mathbf{x}^T Q_0 \mathbf{x} \\ s.t. \quad & \mathbf{x}^T Q_j \mathbf{x} \le c_j, \forall\, j = 1, \dots, m. \end{aligned} \tag{8.18}$$

Based on this fact, any inhomogeneous QCQP can be transformed into a homogeneous one. Without loss of generality, the following approach to solve nonconvex QCQP problems focuses on homogeneous QCQPs.

In order to solve the nonconvex QCQP in (8.18), the semidefinite relaxation method is first introduced to find a tight lower bound on the optimal objective value. By applying interior point method, the relaxed formulation can be solved via an SDP solver [43]. After introducing a rank-one positive definite matrix $X = \mathbf{x}\mathbf{x}^T$, the nonconvex QCQP problem in (8.18) is equivalent to

$$\begin{aligned} J &= \min_X \langle X, Q_0 \rangle \\ s.t. \quad & \langle X, Q_j \rangle \le c_j, \forall\, j = 1, \dots, m \\ & X = \mathbf{x}\mathbf{x}^T, \end{aligned} \tag{8.19}$$

where "$\langle \cdot \rangle$" denotes the inner product of two matrices, that is, $\langle A, B \rangle = \mathbf{trace}(A^T B)$. The nonconvex QCQP problem formulated in (8.19) can be classified as one type of RCOPs expressed in (8.7) where rank of the unknown matrix is constrained by one.

8.4 Iterative Rank Minimization Approach

Satisfying the rank constraint for an unknown matrix is computationally complicated. Existing methodologies for rank-constrained problems are mainly focused on matrix factorization and/or linearization [33, 44], which have slow convergence rate and are sensitive to the initial guess.

In fact, the number of nonzero eigenvalues of a matrix is identical to its rank. For an unknown square matrix $U \in \mathbb{S}_+^p$, it is not feasible to examine its eigenvalues before it is determined.

We focus on the fact that when a matrix rank is r, it has r nonzero eigenvalues. Therefore, instead of making constraint on the rank, we focus on constraining the eigenvalues of U such that the $p - r$ eigenvalues of U are all zeros. The eigenvalue constraints on matrices have been used for graph design [45] and are applied here for rank-constrained problems. Before addressing the detailed approach for rank-constrained problems, we first provide necessary observations that will be used subsequently in the approach.

Proposition 8.4.1 The $(r + 1)th$ largest eigenvalue Λ_{p-r} of matrix $U \in \mathbb{S}_{+}^{p}$ is less equal than e if and only if $eI_{p-r} - V^T\,UV \succeq 0$, where I_{p-r} is the identity matrix with a dimension of $p - r$, $V \in \mathbb{R}^{p\times(p-r)}$ are the eigenvectors corresponding to the $p - r$ smallest eigenvalues of U.

Proof: Assume that the eigenvalues of U are sorted in descending orders in the form of $[\Lambda_p, \Lambda_{p-1}, \ldots, \Lambda_1]$. Since the Rayleigh quotient of an eigenvector is its associated eigenvalue, then

$$V^T\,UV = \begin{bmatrix} \Lambda_{p-r} & 0 & \cdots & 0 \\ 0 & \Lambda_{p-r-1} & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Lambda_1 \end{bmatrix}.$$

Hence,

$$eI_{p-r} - V^T\,UV = \begin{bmatrix} e - \Lambda_{p-r} & 0 & \cdots & 0 \\ 0 & e - \Lambda_{p-r-1} & \cdots & 0 \\ \vdots & \cdots & \ddots & \vdots \\ 0 & 0 & \cdots & e - \Lambda_1 \end{bmatrix}.$$

Therefore, $e \geq \lambda_{p-r}$ if and only if $eI_{p-r} - V^T\,UV \succeq 0$. ■

Corollary 8.4.2 When $e = 0$ and U is a positive semidefinite matrix, $\mathbf{rank}(U) \leq r$ holds if and only if $eI_{p-r} - V^T\,UV \succeq 0$, where $V \in \mathbb{R}^{p\times(p-r)}$ are eigenvectors corresponding to the $p - r$ smallest eigenvalues of U.

However, for problem (8.7), before we solve X, we cannot obtain the exact V matrix, that is, the eigenvectors of X.

Therefore, an iterative method is proposed to solve (8.7) by gradually approaching the constrained rank. At each step k, we solve the following SDP problem formulated as

$$\begin{array}{rc} & J = \min_{X_k, e_k} f(X_k) + w_k e_k \\ s.t. & X_k \in C \\ & X_k \succeq \mathbf{0} \\ & e_k I_{p-r} - V_{k-1}^T X_k V_{k-1} \succeq 0 \end{array}, \tag{8.20}$$

where $w_k = w_0 t^k$ is the weighting factor at iteration k. w_k is increasing with the increment of k when $t > 1$ and $w_0 > 0$ are given parameters. $V_{k-1} \in \mathbb{R}^{p \times (p-r)}$ are the eigenvectors corresponding to the $p - r$ smallest eigenvalues of X solved at previous iteration $k - 1$. At each step, we are trying to optimize the original objective function and at the same time minimize parameter e such that when $e = 0$, the rank constraint on X is satisfied. The weighting factor, w_k, acts as a regularization factor, and increasing its values at each step will enforce the $r + 1$th largest eigenvalue to gradually reduce to zero. The aforementioned approach is repeated until $e \leq \epsilon$, where ϵ is a small threshold for the stopping criteria. It is straightforward to extend (8.20) to problems with multiple rank constraints. For brevity, the simplest version with one rank constraint is described here.

In addition, an initial starting point, V_0, is required at the first iteration $k = 1$. It is intuitive to use the relaxed solution by dropping the last constraint and the penalty term in the objective function in (8.20) for starting point. Under this assumption, X_0 is obtained via

$$\begin{array}{rc} & J = \min_{X_0} f(X_0) \\ s.t. & X_0 \in C \\ & X_0 \succeq \mathbf{0} \end{array}, \tag{8.21}$$

and V_0 are the eigenvectors corresponding to $p - r$ smallest eigenvalues of X_0.

The IRM approach is summarized here for general RCOPs, and proof of its linear convergence to local optimality can be referred to [46, 47].

```
**Algorithm: Iterative Rank Minimization for Solving (8.7)**
**Input:** Problem information $\mathcal{C}$, $w_0 > 0$, $t > 1$, $\epsilon$
**Output:** $X^*$ with minimum $f(X)$
**begin**

1) **Initialize** Set $k = 0$, solve the relaxed problem in (8.21)
   to obtain $V_0$ from $X_k$ via eigenvalue decomposition.
   set $k = k + 1$
2) **while** $e_k \geq \epsilon$
3)    Solve subproblem (8.20) and obtain $X_k, Y_k, Z_k, e_k$
4)    Update $V_k$ from $X_k$ via eigenvalue decomposition.
5)    $k = k + 1$
6)    Update $w_k$ via $w_k = w_{k-1} * t$
7) **end while**
8) Find $X^*$
**end**
```

8.5 Simulation Examples

In this section, simulation examples for the two types of NTD problems described in Section 8.2 and the NTI problem described in Section 8.3 are demonstrated to verify feasibility and efficiency of the proposed method. For small-scale NTD or NTI problems with small number of vertices, that is, $n \leq 7$, exhaustive search is able to find a global optimal solution within reasonable computational time after examining all feasible solutions. In order to do so, we enumerate all feasible solutions satisfying the specified constraints and find the one with best performance to compare with the result obtained from IRM. In addition, large-scale problems where exhaustive search is infeasible to find an optimal solution within reasonable computational time are demonstrated to verify the improved scalability and computational performance of proposed method.

All simulation is run on a desktop computer with a 3.50 GHz processor and a 16.0 RAM.

8.5.1 Example for Designing Fast Converging Consensus-based Network

The first example is to design the weighted network topology to support fast convergence of consensus-based network formulated in (8.4). We first consider a case in relatively small scale with $n = 6$, $r = 7$, and $g_t = 1$. In general, there are $\binom{n(n-1)/2}{r}$ possible configurations satisfying the cardinality constraint for a graph with n vertices and cardinality constraint of r on the edge set. The histogram in Figure 8.2 indicates the distribution of all feasible solutions satisfying the cardinality constraint. When λ_2 ranging from 0.0923 to 0.2228, Figure 8.2 demonstrates the number of feasible configurations for corresponding λ_2 value. Result from IRM is among one of the feasible configurations yielding the largest λ_2. Figure 8.3 shows the optimal topology with designed weight noted for each edge. For the exhaustive search, the average computational time to design edge weights for each configuration is 0.13 secs and there are $\binom{15}{7} = 6435$ configurations. Therefore, the overall computational time after examining all of the configurations is 846.86 s. However, it takes the IRM 5.51 s to converge to an optimal solution.

The second example sets $n = 20$, $r = 100$, and $g_t = 10$ where the global optimal solution cannot be obtained within reasonable computational time on a standard desktop. Figure 8.4 demonstrates the designed topology obtained from IRM and the value of e_k at each iteration is shown in Figure 8.5. Since $e_k \leq 10^{-5}$ at the 18th iteration of IRM, it verifies that the cardinality constraint is satisfied according to the statement of Proposition (8.4.1). In addition, Figure 8.6 shows the designed edge weights at each iteration. For a complete network without cardinality constraint on the edge set, the optimal solution leads to 45 edges for this case. With the cardinality constraint, at least 25 edges should have an edge weight equals to zero. In fact, certain edge weights in Figure 8.6 fluctuate between zero and nonzero, which demonstrates the design results on the network topology at each iteration and how it eventually converges to a

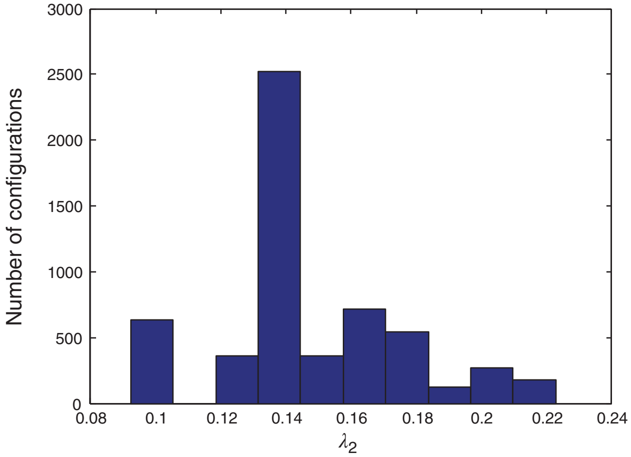


Figure 8.2 Distribution of all feasible solutions satisfying the cardinality constraint on edge set to maximize λ_2 for a network with $n = 6, r = 7$.

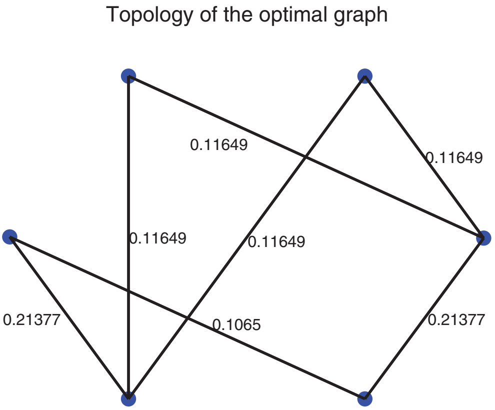


Figure 8.3 Optimal topology from IRM for maximum λ_2 with $n = 6, r = 7$. The total computation time for 44 iterations is 5.5100 s.

solution satisfying the cardinality constraint. It takes the IRM 7.00 s to find a convergent solution with 17 iterations.

8.5.2 Example for Designing Minimum Total Effective Resistance Network

In the second example, IRM is applied to design the weighted network topology to obtain the minimum total effective resistance network formulated in (8.5). In this case, the number of

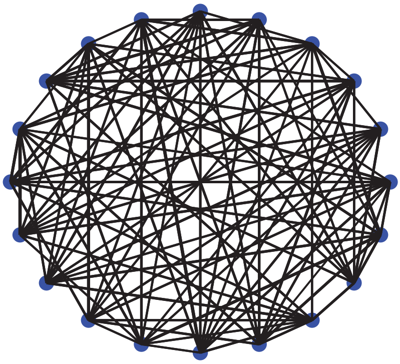


Figure 8.4 Optimal topology for maximum λ_2 with $n = 20, r = 100$.

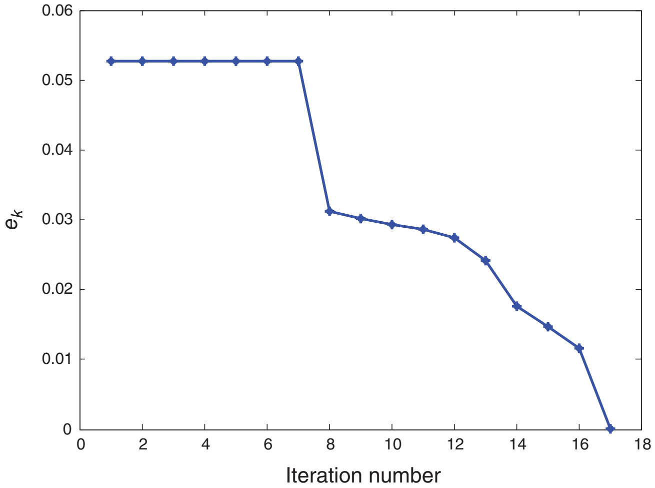


Figure 8.5 Convergence history of e_k for maximum λ_2 with $n = 20, r = 100$.

vertices and cardinality constraint on the edge set are set as $n = 20$, $r = 100$. Figures 8.7 and 8.8 demonstrate the designed topology and the history of edge weights, respectively. It takes the IRM 67.90 s to find a convergent solution with 65 iterations.

8.5.3 Example of NTI with Agent Dynamics Driven by Consensus Protocol

The third example provides a simulation results for identifying the topology of a six-node graph, where dynamics of the nodes

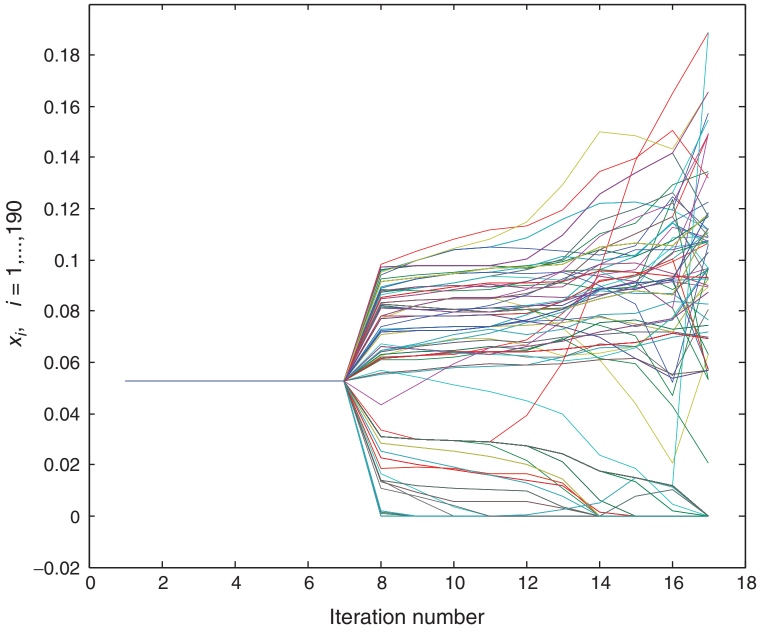


Figure 8.6 Designed edge weights for maximum λ_2 with $n = 20, r = 100$ at each iteration.

are governed by the Consensus Protocol. From the system identification procedures introduced in Section 8.3.1, the "similar" state-space matrices set (A_T, B_T, C_T) is listed here,

$$A_T = \begin{bmatrix} -2.873 & -0.720 & 0.183 & -0.480 & 1.594 & 0.393 \\ -0.720 & -4.227 & -1.197 & -0.076 & -0.927 & -0.723 \\ 0.183 & -1.197 & -5.17 & 0.139 & 0.368 & 0.351 \\ -0.479 & -0.077 & 0.139 & -4.625 & 0.726 & 0.678 \\ 1.594 & -0.927 & 0.368 & 0.726 & -2.813 & 0.908 \\ 0.398 & -0.723 & 0.351 & 0.678 & 0.908 & -2.325 \end{bmatrix},$$

$$B_T = \begin{bmatrix} 0.5152 & -0.0319 & -0.3207 \\ -0.0319 & 0.9979 & -0.0211 \\ -0.3207 & -0.0211 & 0.7878 \\ -0.2928 & -0.0193 & -0.1937 \\ -0.5515 & -0.0363 & -0.3649 \\ -0.4906 & -0.0323 & -0.3246 \end{bmatrix},$$

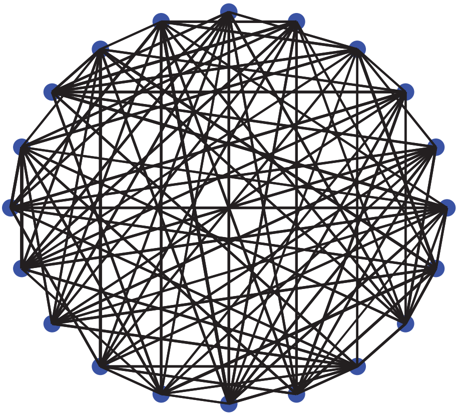


Figure 8.7 Optimal topology for minimum total effective resistance with $n = 20, r = 100$.

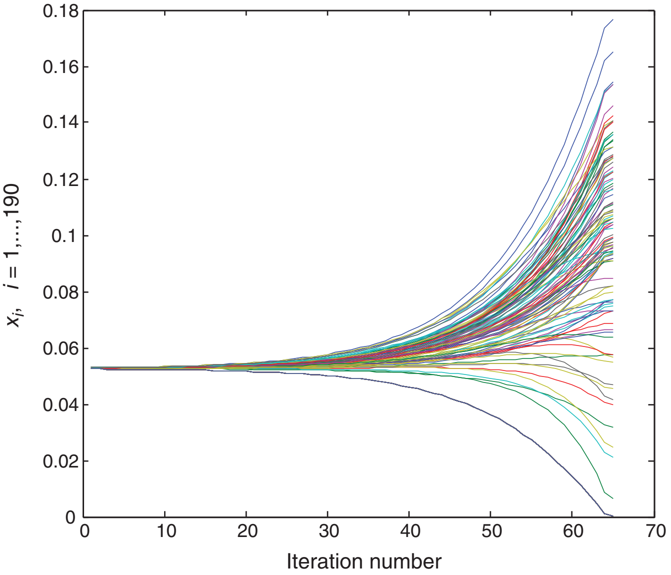


Figure 8.8 Designed edge weights for the minimum total effective resistance with $n = 20, r = 100$ at each iteration.

and $C_T = B'_T$. We define the nodes with indices $i = 1, 2, 3$ as the selected input and output nodes. Under this assumption, matrices B and C are set as $B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$ and $C = B^T$, respectively.

By formulating the NTI problem in the format described in (8.16) and applying the IRM method, the optimal transformation matrix T and the graph Laplacian are found as

$$T_{opt} = \begin{bmatrix} 0.515 & -0.031 & -0.320 & -0.292 & -0.551 & -0.490 \\ -0.031 & 0.997 & -0.021 & -0.019 & -0.036 & -0.032 \\ -0.320 & -0.021 & 0.787 & -0.193 & -0.364 & -0.324 \\ -0.292 & -0.019 & -0.193 & 0.823 & -0.333 & -0.296 \\ -0.551 & -0.036 & -0.364 & -0.333 & 0.372 & -0.558 \\ -0.490 & -0.032 & -0.324 & -0.296 & -0.558 & 0.503 \end{bmatrix},$$

$$L_{opt} = \begin{bmatrix} 3 & -1 & -1 & 0 & -1 & 0 \\ -1 & 4 & 0 & -1 & -1 & -1 \\ -1 & 0 & 4 & -1 & -1 & -1 \\ 0 & -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & -1 & 4 & 0 \\ 0 & -1 & -1 & -1 & 0 & 3 \end{bmatrix}. \tag{8.22}$$

Comparing with the original network demonstrated in Figure 8.9, it is verified that the network topology obtained from the proposed identification procedure is identical to the original one. In addition, the history of the second smallest eigenvalue at each iteration of the IRM algorithm is shown in Figure 8.10. It indicates that e_k in (8.20) quickly reduces to zero within 18 steps, where the stopping threshold is set as $\epsilon = 10^{-4}$.

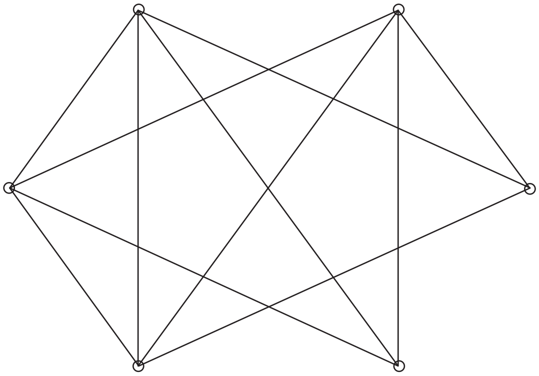


Figure 8.9 Original graph topology with six nodes for the simulation example of network identification problem.

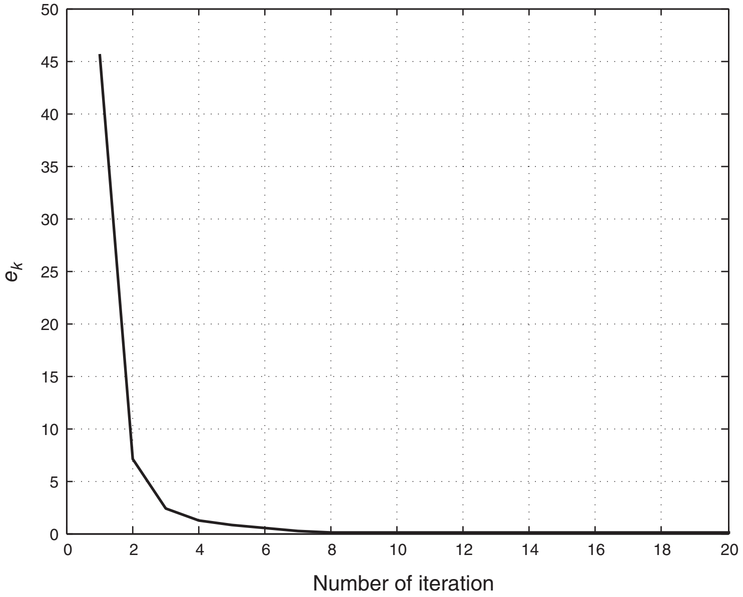


Figure 8.10 Simulation example for network identification of a six-node graph: convergence history of e_k at each iteration.

This fact verifies that we obtain a rank one matrix of X within a few iterative steps.

8.6 Conclusions

This chapter investigates the NTD and identification problems. The NTD problems include designing weighted edges for fast converging consensus-based network and minimum total effective resistance network, with cardinality constraint on the edge set. For the NTI problem, we assume dynamics of agents in the network is governed by consensus protocol. We formulate both NTD and NTI problems as RCOPs, and an iterative method with each iteration formulated as a convex optimization problem is proposed to solve general RCOPs. The effectiveness and efficiency of the proposed approach is verified by different types of simulation examples.

References

1 Chapman, A., Dai, R., and Mesbahi, M. (2011) Network topology design for UAV swarming with wind gusts, in *AIAA Guidance, Navigation, and Control Conference.*

2 Beard, A.W., Lawton, J., and Hadaegh, F.Y. (2001) A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, **9**, 777–790.

3 Xiao, L. and Boyd, S. (2003) Fast linear iterations for distributed averaging. *System & Control Letters*, **53**, 65–78.

4 Shafi, Y., Arcak, M., and El Ghaoui, L. (2010) Designing node and edge weights of a graph to meet Laplacian eigenvalue constraints, in *Prof. Allerton Conference.*

5 Ghosh, A., Boyd, S., and Saberi, A. (2008) Minimizing effective resistance of a graph. *SIAM Review*, **50** (1), 37–66, doi: 10.1137/050645452.

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

7 Dai, R., Maximoff, J., and Mesbahi, M. (2013) Optimal trajectory generation for establishing connectivity in proximity networks. *IEEE Transactions on Aerospace and Electronic Systems*, **49** (3), 1968–1981.

8 Candès, E., Wakin, M., and Boyd, S. (2008) Enhancing sparsity by reweighted 1 minimization. *Journal for Fourier Analysis and Applications*, **14** (5–6), 877–905.

9 Goldstein, T. and Osher, S. (2009) The split Bregman method for l1-regularized problems. *SIAM Journal on Imaging Sciences*, **2** (2), 323–343.

10 Hale, E.T., Yin, W., and Zhang, Y. (2007) A fixed-point continuation method for 11-regularized minimization with applications to compressed sensing. CAAM TR07-07, vol. 43, Rice University, p. 44.

11 Yin, W., Osher, S., Goldfarb, D., and Darbon, J. (2008) Bregman iterative algorithms for l_1-minimization with applications to compressed sensing. *SIAM Journal on Imaging Sciences*, **1** (1), 143–168.

12 Candes, E.J. and Tao, T. (2005) Decoding by linear programming. *IEEE Transactions on Information Theory*, **51** (12), 4203–4215.

13 Lin, F., Fardad, M., and Jovanovic, M.R. (2013) Design of optimal sparse feedback gains via the alternating direction method of multipliers. *IEEE Transactions on Automatic Control*, **58** (9), 2426–2431.

14 Fardad, M. and Jovanovic, M.R. (2014) On the design of optimal structured and sparse feedback gains via sequential convex programming, in *American Control Conference*, pp. 2426–2431.

15 Woodside-Oriakhi, M., Lucas, C., and Beasley, J.E. (2011) Heuristic algorithms for the cardinality constrained efficient frontier. *European Journal of Operational Research*, **213** (3), 538–550.

16 Wu, Z. and Wang, R. (2009) The consensus in multi-agent system with speed-optimized network. *International Journal of Modern Physics B*, **23** (10), 2339–2348, doi: 10.1142/S0217979209052327.

17 Dai, R. and Mesbahi, M. (2011) Optimal topology design for dynamic networks, in *Decision and Control and European Control Conference, 50th IEEE Conference on*, pp. 1280–1285.

18 Bertsimas, D. and Shioda, R. (2009) Algorithm for cardinality-constrained quadratic optimization. *Computational Optimization and Applications*, **43** (1), 1–22.

19 Kung, S.Y., Arun, K., and Rao, D. (1983) State-space and singular-value decomposition-based approximation methods for the harmonic retrieval problem. *Journal of the Optical Society of America*, **73** (12), 1799–1811.

20 Viberg, M. (1995) Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, **31** (12), 1835–1851.

21 Nabi-Abdolyyousefi, M. and Mesbahi, M. (2010) Network identification via node knock-out, in *49th IEEE Conference on Decision and Control*, pp. 2239–2244.

22 Nabi-Abdolyousefi, M. and Mesbahi, M. (2012) Sieve method for consensus-type network tomography. *IET Control Theory and Applications*, **6** (12), 1926–1932.

23 Napoletani, D. and Sauer, T.D. (2008) Reconstructing the topology of sparsely connected dynamical networks. *Physical Review E*, **77** (2), 026 103.

24 Materassi, D. and Innocenti, G. (2010) Topological identification in networks of dynamical systems. *IEEE Transactions on Automatic Control*, **55** (8), 1860–1871.

25 Ayazoglu, M., Sznaier, M., and Ozay, N. (2011) Blind identification of sparse dynamic networks and applications, in *Decision and Control and European Control Conference, 50th IEEE Conference on*, pp. 2944–2950.

26 Sanandaji, B.M., Vincent, T.L., and Wakin, M.B. (2011) Exact topology identification of large-scale interconnected dynamical systems from compressive observations, in *American Control Conference*, pp. 649–656.

27 d'Aspremont, A. and Boyd, S. (2003) Relaxations and randomized methods for nonconvex QCQPs. EE392o Class Notes, Stanford University.

28 Kim, S. and Moon, Y. (2006) Structurally constrained H2 and H1 control: a rank-constrained LMI approach. *Automatica*, **42** (9), 1583–1588.

29 Markovsky, I. (2008) Structured low-rank approximation and its applications. *Automatica*, **44** (4), 891–909.

30 Ten Berge, J.M. and Kiers, H.A. (1991) A numerical approach to the approximate and the exact minimum rank of a covariance matrix. *Psychometrika*, **56** (2), 309–315.

31 Fazel, M., Hindi, H., and Boyd, S.P. (2003) Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices, in *American Control Conference, Proceedings of the 2003*, vol. **3**, pp. 2156–2162.

32 Fazel, M. (2002) Matrix rank minimization with applications, Ph.D. thesis, Stanford University.

33 Grigoriadis, K.M. and Skelton, R.E. (1996) Low-order control design for LMI problems using alternating projection methods. *Automatica*, **32** (8), 1117–1125.

34 Delgado, R.A., Agüero, J.C., and Goodwin, G.C. (2014) A rank-constrained optimization approach: application to factor analysis, in *19th IFAC World Congress.*

35 Delgado, R.A., Agüero, J.C., and Goodwin, G.C. (2016) A novel representation of rank constraints for real matrices. *Linear Algebra and its Applications*, **496**, 452–462.

36 Dattorro, J. (2015) *Convex Optimization & Euclidean Distance Geometry*, Meboo Publishing USA.

37 Hassibi, A., How, J., and Boyd, S. (1999) A path-following method for solving BMI problems in control, in American Control Conference. Proceedings of the 1999, vol. **2**, pp. 1385–1389.

38 Meyer, G. (2011) Geometric optimization algorithms for linear regression on fixed-rank matrices, Ph.D. thesis, University of Liege, Belgium.

39 Mesbahi, M. and Egerstedt, M. (2010) *Graph Theoretic Methods in Multiagent Networks*, Princeton University Press.

40 Ghosh, A., Boyd, S., and Saberi, A. (2008) Minimizing effective resistance of a graph. *SIAM Review*, **50** (1), 37–66.

41 Kalman, R.E. (1963) Mathematical description of linear dynamical systems. *Journal of the Society for Industrial & Applied Mathematics, Series A: Control*, **1** (2), 152–192.

42 Recht, B., Fazel, M., and Parrilo, P.A. (2010) Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, **52** (3), 471–501.

43 Vandenberghe, L. and Boyd, S. (1996) Semidefinite programming. *SIAM Review*, **38**, 49–95.

44 El Ghaoui, L., Oustry, F., and AitRami, M. (1997) A cone complementarity linearization algorithm for static output-feedback and related problems. *IEEE Transactions on Automatic Control*, **42** (8), 1171–1176.

45 Shafi, S.Y., Arcak, M., and El Ghaoui, L. (2011) Graph weight design for Laplacian eigenvalue constraints with multi-agent systems applications, in *Decision and Control and European Control Conference, 2011 50th IEEE Conference on*, IEEE, pp. 5541–5546.

46 Sun, C. and Dai, R. (2015) Identification of network topology via quadratic optimization, in *American Control Conference*, pp. 5752–5757.

47 Sun, C. and Dai, R. (2015) An iterative approach to rank minimization problems, in *54th IEEE Conference on Decision and Control*, pp. 3317–3323.

9

Distributed Multi-Agent Coordination with Uncertain Interactions: A Probabilistic Perspective

Yongcan Cao[1], *David Casbeer*[2], *Eloy Garcia*[2] *and Corey Schumacher*[3]

[1] *Department of Electrical and Computer Engineering, University of Texas at San Antonio, San Antonio, TX, USA*
[2] *The Control Science Center of Excellence, Air Force Research Laboratory, Wright-Patterson AFB, OH, USA*
[3] *Air Force Research Laboratory, Wright-Patterson AFB, OH, USA*

9.1 Introduction

Distributed multi-agent coordination refers to the collective behavior that a team of networked agents reaches some desired configuration via distributed control protocols, also referred to as distributed control algorithms. One powerful tool in the design of distributed control algorithms is consensus, whose goal is to develop proper control algorithms such that networked agents reach some common state. If all agents can reach agreement on their final states, one can modify the algorithms such that the agents reach desired configuration. Recently, consensus has been used to develop numerous algorithms to solve various distributed coordination problems such as formation stabilization [1], flocking [2, 3], and coordinated tracking [4, 5].

One fundamental problem in the study of distributed coordination is to find the condition on the interaction graph such that coordination can be reached, where the interaction graph is a model of the information exchange pattern among a team of networked agents. It is natural that coordination can not be reached for an arbitrary initial state if the agents do not trade information. In other words, information exchange among these

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

agents is necessary to guarantee coordination. Until now, significant research effort has been devoted to find the minimum condition on the interaction graph such that coordination can be reached. For agents with single-integrator kinematics, also referred to as first-order systems, the minimum condition is that for any time t, there exists some T such that the union of the (directed) interaction graphs over the time period $[t, t+T]$ has a *directed spanning tree* [6, 7] (see Section 9.2.1 for the definition of "a directed spanning tree").

Recently, distributed coordination research has started to consider stochastic interaction topologies. More specifically, the interaction graph is chosen randomly from some given set, and due to the randomness, the minimum condition mentioned in the previous paragraph, in general, does not hold and distributed coordination cannot be guaranteed. For example, research has been conducted to find the condition on the interaction graph such that consensus can be reached *in probability* [8, 9], *with probability 1* [10, 11], and *in the mean-square sense* [12], where it is assumed that the interaction graphs are independent over time and the system dynamics are linear. By converting the consensus problem to a stability problem, the three, namely consensus in probability, consensus with probability 1, and consensus in the mean-square sense, are shown to be equivalent under an independent and identically distributed (i.i.d.) process for a stochastic linear system [13].

The contribution of the chapter is threefold. First, in the static case, a lower bound for the probability of coordination is calculated when each interaction graph is equally likely to be selected among the set containing all possible undirected graphs. We then show that the (exact) probability of coordination for n agents is strictly increasing with respect to n, whenever $n \geq 3$. Second, when the undirected fixed interaction graph is chosen randomly, with (possibly) unequal possibility, from the set containing all possible undirected graphs, it is shown that the probability of coordination for n agents approaches one as $n \to \infty$. Third, under a directed switching interaction graph, it is shown that (i) coordination with probability 1, (ii) coordination in probability, and (iii) coordination in the rth mean are equivalent by studying coordination in the sense of probability, *without* requiring an i.i.d. process and linear system dynamics. This differs from the existing work in [8–13], where consensus in the sense of probability is considered for linear systems and assuming temporal independence of the interaction graphs. Moreover, the study of distributed coordination in the sense of probability reveals some explicit relationship between deterministic and stochastic coordination.

The remainder of the chapter is organized as follows. In Section 9.2, graph theory notions and problem statement are presented as preliminaries. The main body of the chapter is given in Sections 9.3 and 9.4, where coordination is studied in the sense of probability under, respectively, a fixed interaction graph and a switching interaction graph. A short conclusion is given in Section 9.5 to summarize the contribution of the chapter.

9.2 Preliminaries

9.2.1 Graph Theory Notions

For a system with n networked agents, the interaction among them can be modeled by a *directed graph* (also called *digraph*) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \dots, n\}$ and $\mathcal{E} \subseteq \mathcal{V}^2$ represent, respectively, the agent set and the edge set. Each edge denoted as (i, j) means that agent j can receive the (state) information from agent i but not necessarily vice versa. Accordingly, agent i is a *neighbor* of agent j and the *neighbor set* of agent j is defined as $\mathcal{N}_j = \{i | (i, j) \in \mathcal{E}\}$. For any set $\mathcal{V}_k \subseteq \mathcal{V}$, the neighbor set of the set $\mathcal{V}_k$ is defined as $\mathcal{N}_{\mathcal{V}_k} = \{i | (i, j) \in \mathcal{E}, \forall i \notin \mathcal{V}_k, j \in \mathcal{V}_k\}$. The notation $\dim(\mathcal{E})$ denotes the *number of edges* in $\mathcal{G}$. A *path* from j_1 to j_m in a directed graph is a sequence of edges in a directed graph of the form $(j_1, j_2), (j_2, j_3), \dots, (j_{m-1}, j_m)$ where $j_k \in \mathcal{V}, k = 1, \dots, m$. A directed graph has *a directed spanning tree* if there exists at least one agent that has directed paths to all other agents. An undirected graph is a special type of the directed graphs, where $(i, j) \in \mathcal{E}$ implies that $(j, i) \in \mathcal{E}$. That is, the information exchange between agents is bidirectional. A path between j_1 and j_m in an undirected graph is defined analogously to that in a directed graph with a further requirement that the information exchange be bidirectional. An undirected graph is *connected* if there exists a path between every pair of agents. For an undirected graph $\mathcal{G}$, the number of undirected edges is $\frac{\dim(\mathcal{E})}{2}$ since the edges are always bidirectional. For graphs $\mathcal{G}_1 = (\mathcal{V}, \mathcal{E}_1)$, $\mathcal{G}_2 = (\mathcal{V}, \mathcal{E}_2)$, …, and $\mathcal{G}_\ell = (\mathcal{V}, \mathcal{E}_\ell)$ sharing the same agent set $\mathcal{V}$, the union graph of them is defined as $\bigcup_{i=1}^{\ell} \mathcal{G}_\ell \triangleq (\mathcal{V}, \bigcup_{i=1}^{\ell} \mathcal{E}_i)$.

The *adjacency matrix*, $\mathcal{A} = [a_{ij}] \in \mathbb{R}^{n\times n}$, for a team of n networked agents is constructed such that $a_{ii} = 0$ and $a_{ij} > 0$ if $(j, i) \in \mathcal{E}$, otherwise $a_{ij} = 0$. The (nonsymmetric) *Laplacian matrix* $\mathcal{L} = [\ell_{ij}] \in \mathbb{R}^{n\times n}$ is defined as $\ell_{ii} = \sum_{j=1,j\neq i}^{n} a_{ij}$ and $\ell_{ij} = -a_{ij}, i \neq j$.

9.2.2 Problem Statement

Consider a group of n networked agents described by

$$\dot{r}_i(t) = u_i(t), \quad i = 1, \dots, n, \tag{9.1}$$

where $r_i(t) \in \mathbb{R}$ and $u_i(t) \in \mathbb{R}$ are, respectively, the state and the control input associated with agent i at time t. The objective of distributed coordination is to ensure that $(r_i - d_i) - (r_j - d_j) \to 0$, where $d_i - d_j$ is the desired relative configuration. Based on the well-studied consensus algorithm [14–16], the following distributed coordination algorithms can be designed

$$u_i(t) = -\sum_{j=1}^{n} a_{ij}(t)[r_i(t) - d_i - r_j(t) + d_j], \quad i = 1, \dots, n, \tag{9.2}$$

where $a_{ij}(t)$ is the (i, j)th entry of the adjacency matrix $\mathcal{A}$ associated with the directed graph $\mathcal{G}$ at time t. The objective of (9.2) is to guarantee coordination, that is, $|r_i(t) - r_j(t)| \to d_i - d_j, \forall i, j \in \{1, \dots, n\}$, as $t \to \infty$ for an arbitrary initial condition $r_i(0)$, $i = 1, \dots, n$. Under a sampled-data setting, that is, $u_i(t) = u_i(kT), \forall t \in [kT, (k + 1)T)$, the closed-loop system of (9.1) using (9.2) can be written as

$$r_i[k + 1] = r_i[k] - T \sum_{j=1}^{n} a_{ij}[k](r_i[k] - d_i - r_j[k] + d_j), \tag{9.3}$$

for $i = 1, \dots, n$, where $r_i[k] = r_i(kT)$, $a_{ij}[k] = a_{ij}(kT)$, k is the discrete-time index, and T is the sampling period.

The existing research answers the question, "what is the necessary and/or sufficient condition on the interaction graph such that coordination is reached deterministically [7, 14–17] or stochastically [8–12]?" This chapter comes from a different perspective and investigates coordination when the condition on the interaction graph is not necessarily guaranteed. Instead, it determines the probability of coordination for stochastic interaction topologies.

In the following, we focus on the sampled-data model in (9.3). Nevertheless, the same conclusion still holds for the continuous-time model given by the closed-loop system of (9.1) using (9.2). Define $x_i \triangleq r_i - d_i$ and x as the column vector of all x_i. The closed-loop system of (9.3) can be written in a vector form as

$$x[k+1] = W_k x[k], \tag{9.4}$$

where $W_k \triangleq I_n - T\mathcal{L}[k]$ with $\mathcal{L}[k] = \mathcal{L}(kT)$ and $I_n \in \mathbb{R}^{n \times n}$ being the identity matrix. Distributed coordination is achieved for (9.1) using (9.2) if and only if $x_i - x_j \to 0$ for all $i, j \in \{1, \ldots, n\}$, which is equivalent to the statement that consensus is reached for agents with states x_i. Therefore, in the following text, we focus on analyzing the probability that consensus can be reached under the closed-loop system (9.4). Since the focus of this chapter is on the impact of the interaction graph on consensus and not on the discrete sampling, we let $T < \min_{i=1,\ldots,n,k=1,2,\ldots} \frac{1}{\sum_{j=1}^{n} a_{ij}[k]}$. It then follows that W_k are stochastic matrices.

9.3 Fixed Interaction Graph

In this section, we study distributed coordination in the sense of probability for a team of n networked agents when the interaction graph is *fixed and undirected* and randomly chosen from some given set $\mathcal{S}_f$. That is, $\mathcal{L}[k]$ in (9.4) is constant. If the interaction graph associated with $\mathcal{G}$ is connected for any $\mathcal{G} \in \mathcal{S}_f$, the probability of coordination is always *one* because any graph chosen from the set $\mathcal{S}_f$ must be connected. Similarly, if the interaction graph associated with $\overline{\mathcal{G}}$ is not connected for any $\overline{\mathcal{G}} \in \mathcal{S}_f$, the probability of coordination is always zero. In the rest of the section, we consider two cases: (i) Equal Possibility: each element in $\mathcal{S}_f$ is chosen with equal probability, and (ii) Unequal Possibility: each element in $\mathcal{S}_f$ is chosen with (possibly) unequal possibility.

Note that the adjacency matrix (correspondingly, Laplacian matrix) associated with an interaction graph is not necessarily unique because different a_{ij} can be chosen for any $a_{ij} > 0$. This case is excluded in the context of the section. Accordingly, we make the following assumption.

Assumption 9.3.1 For any interaction graph, there exists only one adjacency matrix (correspondingly, Laplacian matrix) associated with it.

Before proceeding, let $\begin{pmatrix} n \\ m \end{pmatrix} \triangleq \frac{n!}{m!(n-m)!}$ and $N_n \triangleq \frac{1}{2}n(n-1)$, where n and m are two positive integers with $n > m$.

9.3.1 Equal Possibility

In this section, we consider the case when the set $\mathcal{S}_f$ is given by

$$\mathcal{S}_f \triangleq \left\{ \mathcal{G} \mid \mathcal{V} = \{1, \dots, n\}, \frac{\dim(\mathcal{E})}{2} \in \{0, 1, \dots, N_n\} \right\}. \quad (9.5)$$

Recall that $\frac{\dim(\mathcal{E})}{2}$ denotes the number of undirected edges in $\mathcal{G}$. Under Assumption 9.3.1, $\mathcal{S}_f$ in (9.5) refers to all possible undirected graphs with n agents and an arbitrary number of edges.[1] Let $\dim(\mathcal{S}_f)$ denote the number of elements in $\mathcal{S}_f$. Under Assumption 9.3.1, $\dim(\mathcal{S}_f) = 2^{N_n}$. Note that $\dim(\mathcal{S}_f) \neq \dim(\mathcal{E})$ since $\dim(\mathcal{S}_f)$ refers to the number of undirected graphs while $\dim(\mathcal{E})$ refers to the number of undirected edges. It is expected that $\dim(\mathcal{S}_f)$ is larger than $\dim(\mathcal{E})$ because the number of undirected graphs, in general, is larger than one for a given number of edges since different graphs can be created by using the same number of edges. By equal possibility, we make the following assumption.

Assumption 9.3.2 Any element in $\mathcal{S}_f$ is chosen independently with probability $\frac{1}{\dim(\mathcal{S}_f)}$.

Let $p_{i,j}$ denote the probability of the *existence* of the undirected edge between agents i and j. Then, for any $\mathcal{G} \triangleq (\mathcal{V}, \mathcal{E}) \in \mathcal{S}_f$, the

[1] For a team of n agents, $\frac{\dim(\mathcal{E})}{2}$ is always within the set $\{0, 1, \dots, N_n\}$.

probability that $\mathcal{G}$ is chosen under Assumptions 9.3.1 and 9.3.2 is given by $\sqrt{\prod_{(i,j)\in\mathcal{E}} p_{i,j} \prod_{(i,j)\notin\mathcal{E}} (1-p_{i,j})}$, where the existence of the square root is due to the double count of (i,j) and (j,i) for an undirected graph. Therefore, Assumption 9.3.2 is true if and only if the probability of the *existence* of the undirected edge between any pair of agents is equal to the probability of the *nonexistence* of the undirected edge between them. Equivalently, the probability of the existence of the undirected edge between any pair of agents is equal to $\frac{1}{2}$.

For a fixed and undirected interaction graph, consensus is reached if and only if the undirected interaction graph is connected [15]. Due to Assumptions 9.3.1 and 9.3.2, when the interaction graph is chosen from the set $\mathcal{S}_f$, the probability of coordination can be written as

$$P_n^{\mathcal{S}_f} \triangleq \frac{\text{The number of connected graphs in } \mathcal{S}_f}{\dim(\mathcal{S}_f)}, \tag{9.6}$$

where $P_n^{\mathcal{S}_f}$ denotes the probability of coordination for a team of n agents when the interaction graph is chosen from the set $\mathcal{S}_f$. Before stating the main result in this section, the following lemma is needed.

Lemma 9.3.3 Let $\phi(m) \triangleq m!(n-m)!2^{m(n-m)}$, where m and n are positive integers with $n > m$. Then, $\min_{m=1,\ldots,n-1} \phi(m) = \phi(n-1)$ for any $n \geq 2$.

Proof: When $n = 2$, the proof of the lemma is trivial. We next consider the case when $n \geq 3$.

The natural logarithm of $\phi(m)$ is given by

$$\ln\,\phi(m) = \sum_{i=1}^{m} \ln\,i + \sum_{j=1}^{n-m} \ln\,j + m(n-m)\,\ln\,2.$$

Define $\psi(m) \triangleq \ln\,\phi(m+1) - \ln\,\phi(m)$. By computation, $\psi(m)$ is given by

$$\psi(m) = \ln\,(m+1) - \ln\,(n-m) + (n-2m-1)\,\ln\,2.$$

Note that $\phi(m) = \phi(n-m), \forall m = 1, \ldots, n-1$. When n is odd, $\min_{m=1,\ldots,n-1} \phi(m) = \min_{m=\frac{n-1}{2},\ldots,n-1} \phi(m)$. When $m = \frac{n-1}{2}$,

$\psi(m) = \ln(\frac{n+1}{2}) - \ln(\frac{n+1}{2}) + (n-2m-1)\ln 2 = 0$. In addition, $\psi(m) - \psi(m-1)$ is given by

$$\psi(m) - \psi(m-1) = \ln \frac{m+1}{2m} + \ln \frac{n-m+1}{2(n-m)} \leq 0.$$

Therefore, $\psi(m) \leq 0$ for $m = \left\{\frac{n-1}{2}, \ldots, n-1\right\}$. By recalling the definition of $\psi(m)$, it follows immediately that $\min_{m=1,\ldots,n-1} \phi(m) = \phi(n-1)$.

When n is even, the proof is similar to that for the case when n is odd. ■

Under Assumptions 9.3.1 and 9.3.2, we then have the following theorem regarding $P_n^{\mathcal{S}_f}$.

Theorem 9.3.4 Under Assumptions 9.3.1 and 9.3.2, the probability of coordination $P_n^{\mathcal{S}_f}$ for (9.3) is lower bounded by $1 - \frac{n(n-1)}{2^n}$.Moreover, as $n \to \infty$, $P_n^{\mathcal{S}_f} \to 1$.

Proof: Because distributed coordination is achieved for (9.1) using (9.2) if and only if $x_i - x_j \to 0$ for all $i, j \in \{1, \ldots, n\}$, which is equivalent to the statement that consensus is reached for agents with states x_i. Therefore, in the following, we focus on analyzing the probability that consensus can be reached under the closed-loop system (9.4).

We first prove that $P_n^{\mathcal{S}_f}$ is lower bounded by $1 - \frac{n(n-1)}{2^n}$. Then, the key is to compute the *number of connected graphs* in $\mathcal{S}_f$ according to (9.6). Because it is difficult to directly compute the number of connected graphs in $\mathcal{S}_f$, we turn our focus to the computation of the *number of disconnected graphs* in $\mathcal{S}_f$.

By definition, an undirected graph is *disconnected* if there does not exist a path between some pair of agents. Equivalently, an undirected graph is *disconnected* if there exist at least two *nonempty* subgroups $\mathcal{V}_1 \subset \mathcal{V}$ and $\mathcal{V}_2 \subset \mathcal{V}$ such that the following three conditions are satisfied:

$$\mathcal{V}_1 \bigcup \mathcal{V}_2 = \mathcal{V},$$

$$\mathcal{V}_1 \bigcap \mathcal{V}_2 = \emptyset, \tag{9.7}$$

$$\mathcal{N}_{\mathcal{V}_1} \bigcap \mathcal{N}_{\mathcal{V}_2} = \emptyset, \tag{9.8}$$

where $\emptyset$ denotes the empty set, $\mathcal{N}_{\mathcal{V}_1}$ and $\mathcal{N}_{\mathcal{V}_2}$ refer to the neighbor sets of, respectively, $\mathcal{V}_1$ and $\mathcal{V}_2$ (see Section 9.2.1 for the definition). An undirected graph for a team of n networked agents is composed of $m, 2 \leq m \leq n$, disconnected subgroups if and only if there exists a series of nonempty sets $\mathcal{V}_1, \dots, \mathcal{V}_m$, such that

- For any $i, j \in \{1, 2, \dots, m\}$ and $i \neq j$, $\mathcal{V}_i$ and $\mathcal{V}_j$ satisfy (9.7) and (9.8)
- $\bigcup_{\ell=1}^{m} \mathcal{V}_\ell = \mathcal{V}$.

Then, the number of the disconnected graphs in $\mathcal{S}_f$ is given by $\sum_{k=2}^{n} \mathcal{U}_k$, where $\mathcal{U}_k$, $k = 2, \dots, n$, denotes the number of undirected graphs that are composed of k disconnected subgroups. For the notational simplicity, we use $\mathcal{S}_k^{\text{ds}}$, $k = 2, \dots, n$, to denote the set of the undirected graphs that are composed of k disconnected subgroups. Note that it is challenging to compute $\sum_{k=2}^{n} \mathcal{U}_k$ directly. We next show that

$$\sum_{k=2}^{n} \mathcal{U}_k \leq \frac{1}{2} \sum_{m=1}^{n-1} \binom{n}{m} 2^{N_m} 2^{N_{n-m}}. \tag{9.9}$$

This is the key step to prove the theorem.

Note that $\binom{n}{m}$ refers to the number of choices such that any m agents are chosen from a set of n agents and $2^{N_m} 2^{N_{n-m}}$ refers to the total number of undirected graphs whenever the m agents and the other $n - m$ agents are disconnected, where 2^{N_m} (respectively, $2^{N_{n-m}}$) denotes the number of undirected graphs for m (respectively, $n - m$) agents. Essentially, the value of $\binom{n}{m} 2^{N_m} 2^{N_{n-m}}$ refers to the number of undirected graphs such that any m agents, chosen from a set of n agents, and the remaining $n - m$ agents are disconnected. Naturally, $\frac{1}{2} \sum_{m=1}^{n-1} \binom{n}{m} 2^{N_m} 2^{N_{n-m}}$ refers to the number of undirected graphs such that any $m, m = 1, \dots, n - 1$, agents, chosen from a set of n agents, and the remaining $n - m$ agents are disconnected. Here the factor $\frac{1}{2}$ is used because

- $\binom{n}{m} 2^{N_m} 2^{N_{n-m}}$ is unchanged if m is replaced by $n - m$;
- Both $\binom{n}{m} 2^{N_m} 2^{N_{n-m}}$ and $\binom{n}{m} 2^{N_m} 2^{N_{n-m}}$ with m replaced by $n - m$ are counted in $\sum_{m=1}^{n-1} \binom{n}{m} 2^{N_m} 2^{N_{n-m}}$.

Note that each $\mathcal{S}_k^{\mathrm{ds}}, k = 2, \ldots, n$, is always a subset of the set of the undirected graphs such that any $m, m = 1, \ldots, n-1$, agents, chosen from a set of n agents, and the remaining $n - m$ agents are disconnected by letting the number of an arbitrary subgroup, out of the k subgroups, be m. Note also that $\mathcal{S}_k^{\mathrm{ds}} \bigcap \mathcal{S}_l^{\mathrm{ds}} = \emptyset, \forall k \neq l, k, l \in \{2, \ldots, n\}$. Therefore, (9.9) holds apparently.

Based on (9.9), it follows that $P_n^{\mathcal{S}_f}$ is lower bounded by

$$\begin{aligned} & 1 - \frac{\frac{1}{2}\sum_{m=1}^{n-1}\binom{n}{m}2^{N_m}2^{N_{n-m}}}{\dim(\mathcal{S}_f)} \\ & = 1 - \frac{1}{2}\sum_{m=1}^{n-1}\frac{n!}{m!(n-m)!}\frac{2^{m(m-1)/2}2^{(n-m)(n-m-1)/2}}{2^{n(n-1)/2}} \\ & = 1 - \frac{1}{2}\sum_{m=1}^{n-1}\frac{n!}{m!(n-m)!2^{m(n-m)}} \\ & \geq 1 - \frac{n-1}{2}\max_{m=1,\ldots,n-1}\frac{n!}{m!(n-m)!2^{m(n-m)}}. \end{aligned}$$

It then follows from Lemma 9.3.3 that

$$\begin{aligned} P_n^{\mathcal{S}_f} & \geq 1 - \frac{n-1}{2}\frac{n!}{\min_{m=1,\ldots,n-1} m!(n-m)!2^{m(n-m)}} \\ & = 1 - \frac{n-1}{2}\frac{n!}{(n-1)!2^{n-1}} \\ & = 1 - \frac{n(n-1)}{2^n}. \end{aligned}$$

This completes the proof of the first statement.

The second statement is a direct consequence of the first statement. ■

Although the exact number of connected graphs in $\mathcal{S}_f$ can be written explicitly [18–20], it can be noticed that the exact form is given in terms of recurrence and the computation is rather complicated. Thus, it is challenging to compute the exact $P_n^{\mathcal{S}_f}$. Instead, we show in Theorem 9.3.4 that $P_n^{\mathcal{S}_f}$ is always lower bounded by $1 - \frac{n(n-1)}{2^n}$, which is easily obtained given an arbitrary number of agents. In addition, it becomes possible to analyze some property of $P_n^{\mathcal{S}_f}$ based on the lower bound.

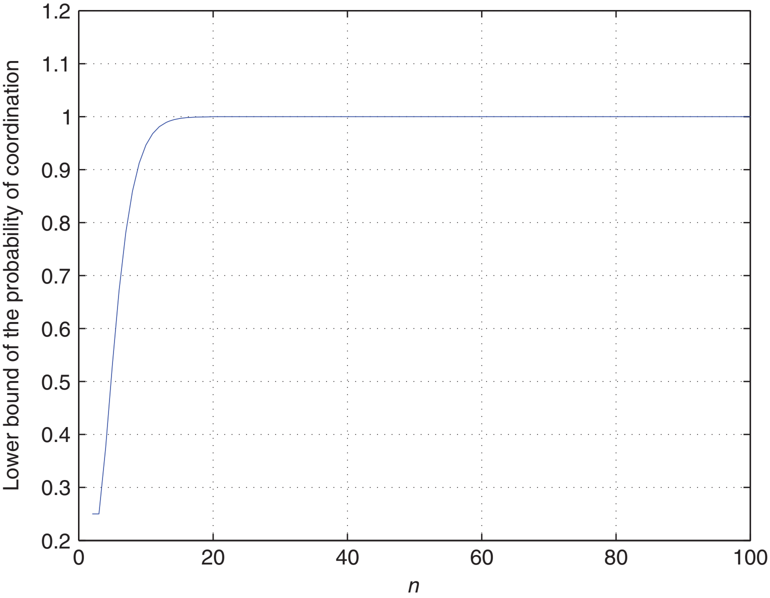


Figure 9.1 The lower bound of the probability of coordination for *n* agents.

Figure 9.1 illustrates the lower bound of the probability of coordination for n agents. It can be observed that the lower bound of the probability of coordination increases as the number of agents increases. In particular, when the number of agents grows to 20, the probability of coordination almost reaches 1.

Theorem 9.3.4 implies that the number of connected graphs is significantly larger than that of disconnected graphs when the number of agents is significantly large. As the number of agents, n, tends to infinity, the probability of coordination approaches one. In other words, coordination is always reached statistically as $n \to \infty$. Since the lower bound of the probability of coordination has been given in Theorem 9.3.4, it is also possible to find the minimum number of agents, although conservative, such that coordination is reached given any confident level $p^c \in (0, 1)$ under Assumptions 9.3.1 and 9.3.2 when S_f is given in (9.5).

One thing that is not mentioned in the proof of Theorem 9.3.4 is how conservative the lower bound could be. As analyzed in the proof of Theorem 9.3.4, $\frac{1}{2}\sum_{m=1}^{n-1}\begin{pmatrix} n \\ m \end{pmatrix} 2^{N_m}2^{N_{n-m}}$ refers to the number of undirected graphs such that any $m, m = 1, \ldots, n-1$, agents, chosen from a set of n agents, and the remaining $n - m$ agents are disconnected. Then, the number of undirected graphs with three disconnected subgroups will be counted multiple times in $\frac{1}{2}\sum_{m=1}^{n-1}\binom{n}{m}2^{N_m}2^{N_{n-m}}$ because the number of each subgroup can be considered m. Similarly, the number of undirected graphs with more than three disconnected subgroups will also be counted multiple times in $\frac{1}{2}\sum_{m=1}^{n-1}\binom{n}{m}2^{N_m}2^{N_{n-m}}$. One interesting property regarding the exact $P_n^{S_f}$ is given in the following theorem without knowing the exact formulation of $P_n^{S_f}$.

Theorem 9.3.5 Under Assumptions 9.3.1 and 9.3.2, $P_n^{S_f} < P_{n+1}^{S_f}$ for $n \geq 3$.

Proof: To prove the theorem, it is crucial to analyze the change $P_{n+1}^{S_f} - P_n^{S_f}$. In other words, it is essential to study how the numbers of disconnected graphs and connected graphs evolve as n increases by one. Denote $N_c(n)$ and $N_t(n)$ as, respectively, the number of connected graphs for n agents and the number of undirected graphs for n agents. Then, $N_t(n+1)$ satisfies $N_t(n+1) = N_t(n)\sum_{i=0}^{n}\binom{n}{i}$, where $\sum_{i=0}^{n}\binom{n}{i}$ denotes the number of choices when one extra agent is added to an n-agent network to form a $(n+1)$-agent network. By definition, $P_{n+1}^{S_f} = \frac{N_c(n+1)}{N_t(n)\sum_{i=0}^{n}\binom{n}{i}}$. Then, the theorem holds if $N_c(n+1) > N_c(n)\sum_{i=0}^{n}\binom{n}{i} = N_c(n)\sum_{i=1}^{n}\binom{n}{i} + N_c(n)$ for $n \geq 3$.

Let $N_c(n+1) = N_c^1(n+1) + N_c^2(n+1)$, where $N_c^1(n+1)$ denotes the total number of connected graphs that are composed of n agents forming a *connected* graph plus one extra agent, labeled as n_e, and $N_c^2(n+1)$ denotes the total number of connected graphs that are composed of n agents forming a *disconnected* graph plus n_e. Note that if the undirected graph for a team of n networked agents is connected, the undirected graph for the n agents plus n_e is connected if and only if n_e has at least one neighbor. Therefore, $N_c^1(n+1) = N_c(n)\sum_{i=1}^{n}\binom{n}{i}$. To prove the theorem, it then suffices to prove that $N_c^2(n+1) > N_c(n)$ for $n \geq 3$. Let us further divide the n agents into two subsets SS_1 and

SS_2, where SS_1 has only one agent while SS_2 has the remaining $n - 1$ agents. To guarantee that the n agents are disconnected, it suffices to assume that there is no interaction between SS_1 and SS_2. Let $\overline{SS_2} = SS_2 \bigcup n_e$. Then, $\overline{SS_2}$ has n agents and their interaction can be randomly chosen. Therefore, the number of connected graphs for $\overline{SS_2}$ is $N_c(n)$. Since we can let that n_e and the agent in SS_1 be connected, the number of connected graphs for $SS_1 \bigcup \overline{SS_2}$ is not smaller than $N_c(n)$. Because the choice of SS_2 and SS_1 is not unique for $n \geq 3$, the number of connected graphs for $SS_1 \bigcup \overline{SS_2}$ is strictly larger than $N_c(n)$. This completes the proof of the theorem. ■

Remark 9.3.6 The study of coordination in the sense of probability is presented in this section, where the interaction graph is *fixed and undirected*. Until now, it is unclear if a similar conclusion in this section is applicable to the case of a *fixed and directed* graph when S_f, by abuse of notation, refers to all possible directed graphs for a team of n agents. Due to the asymmetric nature associated with a directed graph, it is difficult to compute the probability that a directed graph either does or does not have a directed spanning tree. Meanwhile, the computation of $P_{n+1}^{S_f}$ based on $P_n^{S_f}$ is unclear since adding one or more edges between a group of n agents and an extra agent could possibly result in an unknown number of directed graphs that do not have a directed spanning tree even if the directed graph for the n agents has a directed spanning tree. As a consequence, the study of coordination in the sense of probability is challenging when the interaction graph is fixed and directed. It is our conjecture that $P_n^{S_f} \to 1$ as $n \to \infty$ and $P_n^{S_f} < P_{n+1}^{S_f}$ for $n \geq 3$ are still valid for a fixed and directed interaction graph under Assumptions 9.3.1 and 9.3.2.

9.3.2 Unequal Possibility

In this section, we consider the case of unequal possibility, where each element in S_f is not necessarily chosen with equal probability. That is, the probability of the existence of the undirected edge between any pair of agents is not necessarily equal to $\frac{1}{2}$. Furthermore, it is assumed in this subsection that any two a_{ij} are chosen independently.

Let $T(n,q)$ denote the number of undirected graphs with n agents and q undirected edges. Let $t(n,q)$ denote the number of undirected graphs with n agents and q undirected edges that are connected. Let $\beta(n,q) = \frac{t(n,q)}{T(n,q)}$ denote the ratio that an undirected graph with n agents and q undirected edges is connected. Before stating the main result, the following two lemmas are needed.

Lemma 9.3.7 [21] As $n \to \infty$, the following results hold: (1) $\beta(n,q) \to 0$ if $\psi \leq 0$; and (2) $\beta(n,q) \to 1$ as $\psi \to +\infty$, where $\psi \triangleq \frac{2q - n\log n}{n}$.

Lemma 9.3.8 For an undirected interaction graph where $a_{ij} = a_{ji} = 1$ with probability p and $a_{ij} = a_{ji} = 0$ with probability $1-p$, where $p \in (0,1)$, the probability that the number of undirected edges is less than $\lfloor n\log n \rfloor$ is equal to zero as $n \to \infty$, where $\lfloor n\log n \rfloor$ refers to the minimum integer that is not less than $n\log n$.

Proof: When any two a_{ij} are chosen independently, the probability that the number of undirected edges is less than $\lfloor n\log n \rfloor$ can be written as $\sum_{j=0}^{\lfloor n \log n \rfloor -1} \binom{N_n}{j} p^j (1-p)^{N_n - j}$, where $\binom{N_n}{j} p^j (1-p)^{N_n - j}$ is the probability when there exist $j, j = 0, 1, \ldots, \lfloor n\log n \rfloor - 1$, undirected edges. We then obtain that

$$\begin{aligned}
&\lim_{n\to\infty} \sum_{j=0}^{\lfloor n\log n\rfloor -1} \binom{N_n}{j} p^j (1-p)^{N_n - j} \\
&\leq \lim_{n\to\infty} (\lfloor n\log n \rfloor) \max_{j=0,\ldots,\lfloor n\log n\rfloor -1} \frac{(N_n)^j}{j!} p^j (1-p)^{N_n - j} \\
&\leq \lim_{n\to\infty} N_n \max_{j=0,\ldots,\lfloor n\log n\rfloor -1} \frac{(N_n)^j}{j!} p^j (1-p)^{N_n - j} \\
&= \max_{j=0,\ldots,\lfloor n\log n\rfloor -1} \lim_{n\to\infty} \frac{N_n^{j+1}}{j!} \left(\frac{p}{1-p}\right)^j \left[(1-p)^{\frac{N_n}{j+1}}\right]^{j+1},
\end{aligned}$$

where we have used the fact that $\lim_{n\to\infty} \lfloor n\log n \rfloor < \lim_{n\to\infty} N_n$ to derive the last inequality. When $\frac{p}{1-p} \leq 1$, it follows that

$$\begin{aligned}
&\lim_{n\to\infty} \sum_{j=0}^{\lfloor n\log n\rfloor -1} \binom{N_n}{j} p^j (1-p)^{N_n - j} \\
&\leq \max_{j=0,\ldots,\lfloor n\log n\rfloor-1} \lim_{n\to\infty} \frac{(N_n)^{j+1}}{j!}\left[(1-p)^{\frac{N_n}{j+1}}\right]^{j+1} \\
&= \max_{j=0,\ldots,\lfloor n\log n\rfloor-1} \lim_{n\to\infty} \frac{\left[N_n(1-p)^{\frac{N_n}{j+1}}\right]^{j+1}}{j!}.
\end{aligned}$$

Based on the definition of N_n, it then follows that $\lim_{n\to\infty} N_n (1-p)^{\frac{N_n}{j+1}} = \lim_{n\to\infty}\frac{n(n-1)}{2}(1-p)^{\frac{n(n-1)}{2(j+1)}}$. Taking $j \leq \lfloor n\log n\rfloor - 1$ into account yields that

$$\begin{aligned}
\lim_{n\to\infty} N_n(1-p)^{\frac{N_n}{j+1}} &\leq \lim_{n\to\infty}\frac{n(n-1)}{2}(1-p)^{\frac{n(n-1)}{2(\lfloor n\log n\rfloor)}} \\
&\leq \lim_{n\to\infty}\frac{n(n-1)}{2}(1-p)^{\frac{n(n-1)}{2(n\log n+1)}} \\
&= \lim_{n\to\infty}\frac{n(n-1)}{2}(1-p)^{\frac{n}{2\log\ n}},
\end{aligned}$$

where we have used the fact that $\lfloor n\log n\rfloor \leq n\log n + 1$ to derive the last inequality. When $p \in (0,1)$, it is obvious that $\frac{n(n-1)}{2}(1-p)^{\frac{n}{2\log\ n}} \to 0$ as $n \to \infty$. Thus, $\lim_{n\to\infty} N_n(1-p)^{\frac{N_n}{j+1}} = 0$ when $p \in (0,1)$. When $\frac{p}{1-p} \geq 1$, it follows that

$$\begin{aligned}
&\lim_{n\to\infty} \sum_{j=0}^{\lfloor n\log n\rfloor -1} \binom{N_n}{j} p^j (1-p)^{N_n - j} \leq \max_{j=0,\ldots,\lfloor n\log n\rfloor-1} \\
&\lim_{n\to\infty} \frac{\left(N_n\frac{p}{1-p}\right)^{j+1}}{j!}\left[(1-p)^{\frac{N_n}{j+1}}\right]^{j+1}.
\end{aligned}$$

By following a similar analysis to that for the case when $\frac{p}{1-p} \leq 1$, it can also be obtained that $\lim_{n\to\infty} \sum_{j=0}^{\lfloor n\log n\rfloor -1} \binom{N_n}{j} p^j (1-p)^{N_n-j} = 0$. ■

Lemma 9.3.7 characterizes the relationship between the probability of connectedness and the number of undirected edges as $n \to \infty$. Lemma 9.3.8 shows that the number of undirected edges is less than some function of the number of agents with a probability zero as $n \to \infty$. Both of them are used in the proof of the following theorem.

Theorem 9.3.9 For an undirected interaction graph where $a_{ij} = a_{ji} = 1$ with probability p and $a_{ij} = a_{ji} = 0$ with probability $1 - p$, where $p \in (0, 1)$, the probability of coordination for (9.3) satisfies that $P_n^{\mathcal{S}_f} \to 1$ as $n \to \infty$.

Proof: As discussed in Section 9.3.1, the probability of coordination $P_n^{\mathcal{S}_f}$ is identical to the probability that the undirected graph is connected. Note that $\mathcal{S}_f = \mathcal{S}_{uc} \bigcup (\mathcal{S}_f \backslash \mathcal{S}_{uc})$, where $\mathcal{S}_{uc}$ denotes the set of undirected graphs with the number of undirected edges being less than $\lfloor n \log n \rfloor$ and $\mathcal{S}_f \backslash \mathcal{S}_{uc}$, by notation, denotes the set of undirected graphs with the number of undirected edges being not less than $\lfloor n \log n \rfloor$. It is clear that $\mathcal{S}_{uc} \subset \mathcal{S}_f$. In order to prove the theorem, it is sufficient to show that

1. $\frac{\dim(\mathcal{S}_{uc})}{\dim(\mathcal{S}_f)} \to 0$ as $n \to \infty$, where $\dim(\mathcal{S}_{uc})$ and $\dim(\mathcal{S}_f)$ denote the number of elements in, respectively, $\mathcal{S}_{uc}$ and $\mathcal{S}_f$;
2. The probability that an arbitrary graph chosen from $\mathcal{S}_f \backslash \mathcal{S}_{uc}$ is connected is equal to one as $n \to \infty$.

Note that statement 1 is shown in Lemma 9.3.8. In order to prove the theorem, it suffices to prove statement 2. When $q \geq \lfloor n \log n \rfloor$, it follows that $\psi \to +\infty$ as $n \to \infty$, where ψ is defined in Lemma 9.3.7. It then follows from Lemma 9.3.7 that $\beta(n, q) \to 1$ as $n \to \infty$. By definition, the probability that an arbitrary graph chosen from $\mathcal{S}_f \backslash \mathcal{S}_{uc}$ is connected is given by $\frac{\sum_{q=\lfloor n \log n \rfloor}^{N_n} t(n,q)}{\sum_{q=\lfloor n \log n \rfloor}^{N_n} T(n,q)}$. Because $\lim_{n\to\infty} \beta(n, q) = \lim_{n\to\infty} \frac{t(n,q)}{T(n,q)} = 1$ for any $q \geq \lfloor n \log n \rfloor$, it follows from the fact that $\min_{q \geq \lfloor n \log n \rfloor} \beta(n, q) \leq \frac{\sum_{q=\lfloor n \log n \rfloor}^{N_n} t(n,q)}{\sum_{q=\lfloor n \log n \rfloor}^{N_n} T(n,q)} \leq \max_{q \geq \lfloor n \log n \rfloor} \beta(n, q)$ that $\lim_{n\to\infty} \frac{\sum_{q=\lfloor n \log n \rfloor}^{N_n} t(n,q)}{\sum_{q=\lfloor n \log n \rfloor}^{N_n} T(n,q)} = 1$. This completes the proof. ∎

In Theorem 9.3.9, it is assumed that for any pair of i and j, where $i, j \in \{1, \ldots, n\}$ and $i \neq j$, the possibility of $a_{ij} = 1$ is always equal to some common $p \in (0, 1)$. A generalization of Theorem 9.3.9 is given in the following corollary.

Corollary 9.3.10 For an undirected interaction graph where $a_{ij} = a_{ji} = 1$ with probability $p_{i,j}$ and $a_{ij} = a_{ji} = 0$ with probability $1 - p_{i,j}$, where $i \in \{1, \ldots, n\}, j < i$, and $p_{i,j} \in (0, 1)$, the probability of coordination for (9.3) satisfies that $P_n^{S_f} \to 1$ as $n \to \infty$.

Proof: The proof is similar to that of Theorem 9.3.9 by proving statements 1 and 2 in the proof of Theorem 9.3.9. ■

It is expected that the probability of coordination for a finite number of agents whose interaction graph is chosen to satisfy Theorem 9.3.9 depends on p. However, as $n \to \infty$, the probability of coordination $P_n^{S_f} \to 1$ regardless of the value of p.

9.4 Switching Interaction Graph

In this section, we study coordination in the sense of probability under a switching interaction graph. In the context of the section, we assume that the interaction graph is piecewise constant. More specifically, the interaction graph $\mathcal{G}$ is assumed to be constant for $t \in [(k-1)T, kT), k = 1, 2, \ldots$, and switches at kT, where T is a positive constant. Instead of assuming an undirected interaction graph as in Section 9.3, the interaction graph is assumed to be directed. In addition, Assumptions 9.3.1 and 9.3.2 are not needed here.

Before presenting the main result, we first define coordination in probability, coordination with probability 1, and coordination in the rth mean, which will be used later in the section. Note that the following definitions are rather standard.

Definition 9.4.1 For the closed-loop system (9.4), coordination is reached *in probability* if for any $\varepsilon > 0$,

$$\lim_{k\to\infty} \Pr(|r_i[k] - d_i - r_j[k] + d_j| \geq \varepsilon) = 0, \forall i, j \in \{1, \ldots, n\}, \tag{9.10}$$

where $\Pr(\cdot)$ denotes the probability that an event happens.

Definition 9.4.2 For the closed-loop system (9.4), coordination is reached *with probability 1* or *almost surely* if

$$\lim_{k\to\infty} \Pr(|r_i[k] - d_i - r_j[k] + d_j| = 0) = 1. \forall i, j \in \{1, \ldots, n\}. \tag{9.11}$$

Definition 9.4.3 For the closed-loop system (9.4), coordination is reached *in the rth mean* if for some $r \geq 1$,

$$\lim_{k\to\infty} E(|r_i[k] - d_i - r_j[k] + d_j|^r) = 0, \quad \forall i, j \in \{1, \ldots, n\},$$

where $E(\cdot)$ denotes the expected value.

Recall that the study of coordination in the sense of probability depends on the result on coordination under a deterministic setting (c.f. Section 9.3). The next lemma concerning coordination under a deterministic setting is necessary before proceeding.

Lemma 9.4.4 [6, 7] For the closed-loop system (9.4) with an arbitrary initial state, consensus is reached if and only if there exists a positive integer κ such that for any $k > 0$, the union of the interaction graphs across $[kT, (k + \kappa)T)$ has a directed spanning tree.

Clearly, a crucial role in determining whether or not coordination can be reached is the condition that there exists a positive integer κ such that the union of the interaction graphs across $[kT, (k + \kappa)T]$ has a directed spanning tree for any $k > 0$. For notational simplicity, we define Ξ as the event that there exists a positive integer κ such that the union of the interaction graphs across $[kT, (k + \kappa)T)$ has a directed spanning tree for any $k > 0$. Then, we have the following theorem regarding the probability of coordination under a switching interaction graph.

Theorem 9.4.5 For the closed-loop system of (9.3) with an arbitrary initial state, the probability of coordination is equal to the probability of event Ξ.

Proof: Due to the equivalence of (9.3) and (9.4) , the theorem is a direct result of Lemma 9.4.4 in the sense of probability. ■

Remark 9.4.6 When W_k are i.i.d., it is shown in [11] that consensus is reached either almost surely or almost never. Another interpretation is that the probability of consensus is either 1 or 0 (due to the Kolmogorov's 0–1 law). Nevertheless, when W_k are not necessarily i.i.d., the probability of consensus could be in the interval (0, 1). For example, consider a team of four agents with the interaction graph for $t \in [(k-1)T, kT), k = 1, 2, \ldots,$ being chosen randomly from

$$\begin{cases} \{\mathcal{G}_1, \mathcal{G}_2\} \text{ with some probability,} & k \bmod 2 = 0, \\ \{\mathcal{G}_2, \mathcal{G}_3\} \text{ with some probability,} & k \bmod 2 = 1, \end{cases}$$

where $\mathcal{G}_i, i = 1, \ldots, 3$, are given in Figures 9.2–9.4. If a graph is first chosen randomly from $\{\mathcal{G}_2, \mathcal{G}_3\}$ and the selected graph is applied every time $k \bmod 2 = 1$, then the probability of consensus is equivalent to the probability of choosing $\mathcal{G}_3$ in the set $\{\mathcal{G}_2, \mathcal{G}_3\}$.

The probability of coordination under a switching interaction graph is given in Theorem 9.4.5, where the probability of coordination is determined by the probability of event Ξ. In particular, the dependence of the interaction graphs over different time intervals might result in a probability of coordination that is neither zero nor one. We next study the necessary and sufficient condition on the interaction graph such that coordination can be reached in probability, with probability 1, and in the rth mean

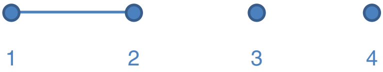


Figure 9.2 $\mathcal{G}_1$. The nodes represent the agents and the line connecting a pair of agents represents the undirected interaction between them.

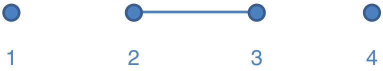


Figure 9.3 $\mathcal{G}_2$. The nodes represent the agents and the line connecting a pair of agents represents the undirected interaction between them.

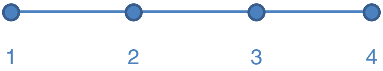


Figure 9.4 $\mathcal{G}_3$. The nodes represent the agents and the line connecting a pair of agents represents the undirected interaction between them.

without assuming the independence of the interaction graphs over different time intervals.

Lemma 9.4.7 For the closed-loop system (9.3) with an arbitrary initial state, coordination is reached with probability 1 or almost surely if and only if the probability of event Ξ is equal to one.

Proof: From Theorem 9.4.5, the probability of coordination is equal to the probability of event Ξ. Then, the probability of coordination is equal to one if and only if the probability of event Ξ is equal to one. ■

Remark 9.4.8 To understand Lemma 9.4.7, it is important to interpret the meaning behind the statement that the probability of event Ξ is equal to one. Recall that event Ξ refers to a property of some graph sequence whose length is infinite. Based on the definition of probability, we divide the graph sequences into two sets Ω_1 and Ω_2, where Ω_1 denotes the set of graph sequences such that each element in Ω_1 satisfies the property while Ω_2 denotes the set of graph sequences such that each element in Ω_2 does not satisfy the property. Let $\dim(\Omega_1)$ and $\dim(\Omega_2)$ denote, respectively, the number of elements in Ω_1 and the number of elements in Ω_2. If each element in $\Omega_1 \bigcup \Omega_2$ is chosen independently with equal probability, the probability of event Ξ is given by $\frac{\dim(\Omega_1)}{\dim(\Omega_1)+\dim(\Omega_2)}$. If $\frac{\dim(\Omega_1)}{\dim(\Omega_1)+\dim(\Omega_2)} = 1$, the probability of coordination is equal to one. However, event Ξ does not hold for any element in Ω_2.

In [11], a similar conclusion to that in Lemma 9.4.7 was shown by using ergodicity and probabilistic arguments when W_k *in (9.4) are i.i.d. stochastic matrices with positive diagonal entries.* By considering the problem in the sense of probability, Lemma 9.4.7 provides a different view of the problem with easy interpretation and proof. Moreover, the relationship between coordination under a deterministic setting and that under a stochastic setting is revealed thanks to the concept of probability. It is not required in Lemma 9.4.7 that W_k in (9.4) be i.i.d. stochastic matrices.

We next study the necessary and sufficient condition on the interaction graph to guarantee coordination in probability.

Lemma 9.4.9 For the closed-loop system (9.3) with an arbitrary initial state, coordination is reached in probability if and only if the probability of event Ξ is equal to one.

Proof: (*Necessity*) We prove this part by contradiction. Assume that coordination is reached in probability does not necessarily imply that the probability of event Ξ is equal to one. Note that (9.10) implies that for any $\varepsilon > 0$ and $\epsilon > 0$, there exists a positive integer κ such that for all $k \geq \kappa$,

$$\Pr(|x_i[k] - x_j[k]| \geq \varepsilon) < \epsilon, \quad \forall i, j \in \{1, \ldots, n\}. \tag{9.12}$$

Combining with the assumption means that (9.12) does not necessarily imply that the probability of event Ξ is equal to one. In other words, the probability of event Ξ is less than 1 when (9.12) holds. Note from Lemma 9.4.4, event Ξ happens if and only if

$$\lim_{k\to\infty} |x_i[k] - x_j[k]| = 0, \quad \forall i, j \in \{1, \ldots, n\}.$$

The statement that the probability of event Ξ is less than 1 implies that there exists at least a pair of agents l and m such that $\lim_{k\to\infty} |x_l[k] - x_m[k]| \neq 0$ with a *nonzero* probability because otherwise the probability of event Ξ is equal to 1. This in turn implies that there does not exist a positive integer κ such that for any $\varepsilon > 0$ and $\epsilon > 0$,

$$\Pr(|x_l[k] - x_m[k]| \geq \varepsilon) < \epsilon, \quad \forall k \geq \kappa,$$

which is a contradiction of the statement that for any $\varepsilon > 0$ and $\epsilon > 0$, there always exists a positive integer κ such that (9.12) holds. The proof of the necessity part is completed.

(*Sufficiency*) When the probability of event Ξ is equal to one, it follows from Lemma 9.4.7 that coordination is reached with probability 1, or almost surely. Then, the sufficiency part is proved by recalling the relationship between convergence in probability and convergence with probability 1. ■

Again, the next lemma presents the necessary and sufficient condition on the interaction graph to guarantee coordination in the rth mean.

Lemma 9.4.10 For the closed-loop system (9.3) with an arbitrary initial state, coordination is reached in the rth mean if and only if the probability of event Ξ is equal to one.

Proof: (*Necessity*) When the probability of event Ξ is equal to one, it follows from Lemma 9.4.9 that coordination is reached in probability. Then, the necessity part is proved by recalling the relationship between convergence in probability and convergence in the rth mean.

(*Sufficiency*) If event Ξ happens, it follows from Lemma 9.4.4 that coordination can be reached, which in turn implies that $\lim_{k\to\infty}|x_i[k]-x_j[k]|^r=0, \forall i,j\in\{1,\dots,n\}$. Moreover, it is always valid that $|x_i[k]-x_j[k]|\le \max_\ell x_\ell[0]-\min_\ell x_\ell[0], \forall i,j\in\{1,\dots,n\}, k=1,2,\dots,$ for the closed-loop system (9.4). Given that the probability of event Ξ is equal to p_Ξ, it follows from the definition of expectation that

$$\begin{aligned}&\lim_{k\to\infty} E(|x_i[k]-x_j[k]|^r)\\ &\quad\le 0\times p_\Xi+\lim_{k\to\infty}\max_{i,j\in\{1,\dots,n\}}|x_i[k]-x_j[k]|^r\times(1-p_\Xi)\\ &\quad\le(\max_\ell x_\ell[0]-\min_\ell x_\ell[0])^r\times(1-p_\Xi), i,j\in\{1,\dots,n\}.\end{aligned}$$

When $p_\Xi=1$, it follows that $\lim_{k\to\infty}E(|x_i[k]-x_j[k]|^r)\le 0$. Because $\lim_{k\to\infty}E(|x_i[k]-x_j[k]|^r)\ge 0$ due to the fact that $|x_i[k]-x_j[k]|^r\ge 0$, it then follows that $\lim_{k\to\infty}E(|x_i[k]-x_j[k]|^r)$ $=0$. This completes the proof of the sufficiency part. ■

Theorem 9.4.11 For the closed-loop system (9.3) with a random switching interaction graph and an arbitrary initial state, the following four statements are equivalent:

1. Coordination is reached in probability.
2. Coordination is reached with probability 1 or almost surely.
3. Coordination is reached in the rth mean.
4. The probability of event Ξ is equal to one.

Proof: The theorem is a direct result of Lemmas 9.4.7, 9.4.9, and 9.4.10. ■

Note that linear system dynamics are assumed in the previous part of the chapter. Instead of assuming linear dynamics as in (9.4), the next corollary presents a general case when nonlinear dynamics are assumed. We refer to the nonlinear discrete-time model used in [6] as

$$x_\ell(i+1)=f_\ell(i,x_1(t),\dots,x_n(t)),\quad \ell=1,\dots,n, \tag{9.13}$$

where $f_j, j = 1, \dots, n$, are continuous functions satisfying Assumption 1 in [6].

Corollary 9.4.12 For the nonlinear system (9.13) with a random switching interaction graph and an arbitrary initial state, the four statements in Theorem 9.4.11 are equivalent.

Proof: To prove the corollary, it is sufficient to show that Lemmas 9.4.7, 9.4.9, and 9.4.10 are still valid for the nonlinear system (9.13). Based on the proofs of Lemmas 9.4.7, 9.4.9, and 9.4.10, the three lemmas are still valid if $|x_i[k] - x_j[k]| \leq \max_\ell x_\ell[0] - \min_\ell x_\ell[0], \forall i, j \in \{1, \dots, n\}, k = 1, 2, \dots$, for the closed-loop system (9.13) and the conclusion presented in Lemma 9.4.4 still holds for the nonlinear system (9.13). Note that both of them are shown in [6]. This completes the proof of the corollary. ■

The existing research on the topic of coordination under a stochastic setting focuses on deriving the condition on the interaction graph such that coordination can be reached in probability [8, 9], almost surely [10, 11], in the mean-square sense [12]. A similar statement concerning the equivalence of them, as shown in Theorem 9.4.11, is given in [13], where a projection technique is used to convert the coordination problem to a stability problem. Note that W_k in (9.4) are assumed to be i.i.d. stochastic matrices in [13]. In addition, a linear system is needed in order to guarantee that Proposition III.2 in [13] holds. As a contrary, the equivalence of coordination in probability, coordination with probability 1, and coordination in the rth mean, is shown in Theorem 9.4.11 and Corollary 9.4.12, where W_k in (9.4) are stochastic matrices that are not necessarily i.i.d. and the agent dynamics are not necessarily linear. Moreover, different from the techniques in [8–11, 13], coordination under a stochastic setting is studied here in the sense of probability. The study of coordination in the sense of probability reveals a direct connection between coordination under a deterministic setting and that under a stochastic setting.

Simulation examples. To demonstrate the validity of the main results in this section, we conduct simulations for a team of four agents. In particular, we consider four different communication graphs whose Laplacian matrices are chosen as

$$L_1 = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$L_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$L_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix},$$

and

$$L_4 = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}.$$

The sampling period $T = 0.1$.

We consider two different cases:

1. The Laplacian matrix associated with the interaction graph for $t \in [(k-1)T, kT), k = 1, 2, \ldots,$ is chosen randomly as

$$\begin{cases} \{L_1, L_2\} \text{ with probability } 0.5, & k \bmod 2 = 1, \\ \{L_3, L_4\} \text{ with probability } 0.5, & k \bmod 2 = 0. \end{cases}$$

2. The Laplacian matrix associated with the interaction graph is chosen randomly with equal probability from $\{L_1, L_3\}$.

Figure 9.5 shows the simulation examples for Case 1 by running 100 repeated tests. It can be observed that all agents converge toward the same final state. In other words, coordination is expected. This is consistent with the main results in this section because the probability of Ξ is 1 in this case.

Figure 9.6 shows the simulation examples for Case 2 by running 100 repeated tests. Because the probability of Ξ is less

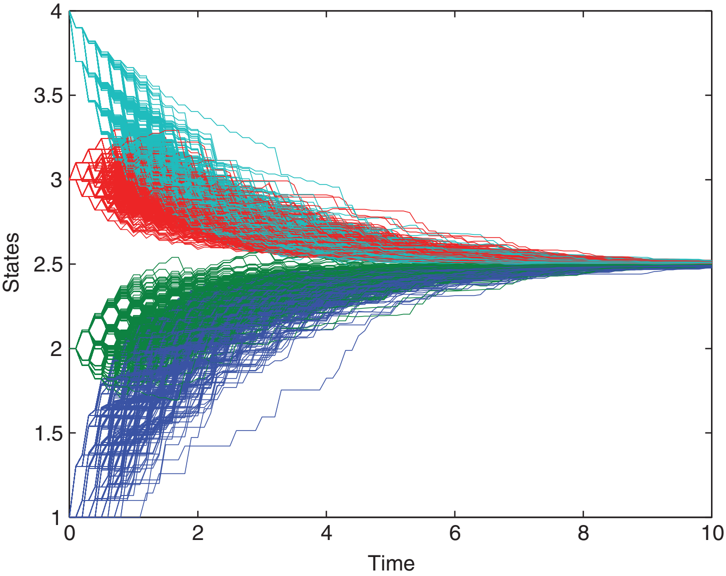


Figure 9.5 Simulation results for Case 1 with 100 repeated tests.

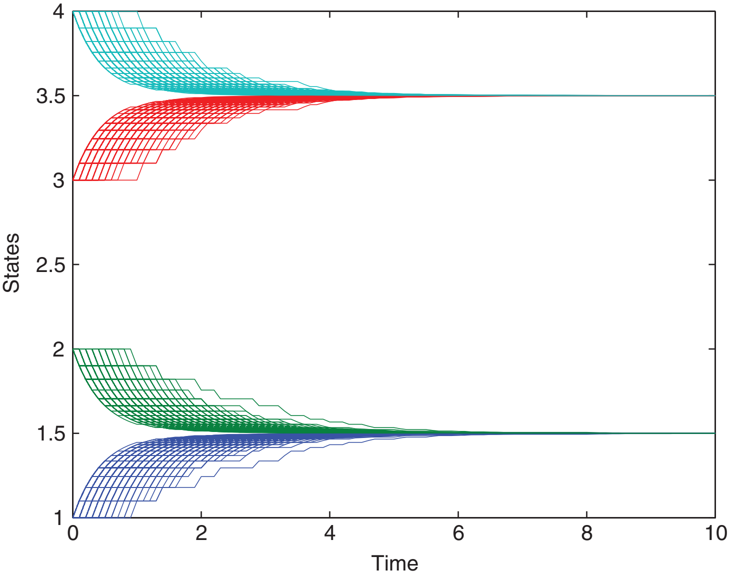


Figure 9.6 Simulation results for Case 2 with 100 repeated tests.

than 1,[2] it can be observed that the four agents cannot reach team coordination. However, two pairs can reach coordination on their final states. In particular, the first two agents will reach coordination because Ξ is one for the two agents. Similarly, the last two agents will reach coordination as well because Ξ is also one for them.

9.5 Conclusion

In this chapter, we studied distributed coordination in the sense of probability for both fixed interaction and switching interaction graphs. When the undirected fixed interaction graph is chosen randomly, with equal possibility, from the set containing all possible undirected graphs, we derived a lower bound for the probability of coordination and then showed that the (exact) probability of coordination for n agents is increasing with respect to n whenever $n \geq 3$. When the undirected fixed interaction graph is chosen randomly, with (possibly) unequal possibility, from the set containing all possible undirected graphs, the probability was shown to approach one as the number of agents approaches infinity. For the case of a directed switching interaction graph, the equivalence of coordination in probability, coordination with probability 1, and coordination in the rth mean was shown without requiring an i.i.d. random process or linear system dynamics.

References

1 Dimarogonas, D.V. and Kyriakopoulos, K.J. (2009) Inverse agreement protocols with application to distributed multi-agent dispersion. *IEEE Transactions on Automatic Control*, **54** (3), 657–663.

2 Olfati-Saber, R. (2006) Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, **51** (3), 401–420.

[2] In fact, the probability of Ξ is zero in this case.

3 Su, H., Wang, X., and Lin, Z. (2009) Flocking of multi-agents with a virtual leader. *IEEE Transactions on Automatic Control,* **54** (2), 293–307.

4 Cao, Y. and Ren, W. (2012) Distributed coordinated tracking with reduced interaction via a variable structure approach. *IEEE Transactions on Automatic Control,* **57** (1), 33–48.

5 Dong, W. (2012) Tracking control of multiple-wheeled mobile robots with limited information of a desired trajectory. *IEEE Transactions on Robotics,* **28** (1), 262–268.

6 Moreau, L. (2005) Stability of multi-agent systems with time-dependent communication links. *IEEE Transactions on Automatic Control,* **50** (2), 169–182.

7 Ren, W. and Beard, R.W. (2005) Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control,* **50** (5), 655–661.

8 Hatano, Y. and Mesbahi, M. (2005) Agreement over random networks. *IEEE Transactions on Automatic Control,* **50** (11), 1867–1872.

9 Wu, C.W. (2006) Synchronization and convergence of linear dynamics in random directed networks. *IEEE Transactions on Automatic Control,* **51** (7), 1207–1210.

10 Porfiri, M. and Stilwell, D.J. (2007) Consensus seeking over random weighted directed graphs. *IEEE Transactions on Automatic Control,* **52** (9), 1767–1773.

11 Tahbaz-Salehi, A. and Jadbabaie, A. (2008) A necessary and sufficient condition for consensus over random networks. *IEEE Transactions on Automatic Control,* **53** (3), 791–795.

12 Abaid, N. and Porfiri, M. (2011) Consensus over numerosity-constrained random networks. *IEEE Transactions on Automatic Control,* **56** (3), 649–654.

13 Song, Q., Chen, G., and Ho, D.W.C. (2011) On the equivalence and condition of different consensus over a random network generated by i.i.d. stochastic matrices. *IEEE Transactions on Automatic Control,* **56** (5), 1203–1207.

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

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

16 Moreau, L. (2004) Stability of continuous-time distributed consensus algorithms, in *Proceedings of the IEEE Conference on Decision and Control*, Paradise Island, Bahamas, pp. 3998–4003.

17 Lin, Z., Broucke, M., and Francis, B. (2004) Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*, **49** (4), 622–629.

18 Nijenhuis, A. and Wilf, H.S. (1979) The enumeration of connected graphs and linked diagrams. *Journal of Combinatorial Theory, Series A*, **27** (3), 356–359.

19 Gross, J.L. and Yellen, J. (eds) (2004) *Handbook of Graph Theory*, CRC Press, Boca Raton, FL.

20 The On-Line Encyclopedia of Integer Sequences®(OEIS®), http://oeis.org/A001187 (accessed 21 October 2016).

21 Wright, E.M. (1973) The probability of connectedness of a large unlabelled graph. *Bulletin of the American Mathematical Society*, **79** (4), 767–769.

10

Awareness Coverage Control in Unknown Environments Using Heterogeneous Multi-Robot Systems

Yue Wang and Li Wang

Department of Mechanical Engineering, Clemson University, Clemson, SC, USA

10.1 Introduction

Recent advancements in wireless communication, sensing, computing, and control technologies give rise to cyber-physical systems (CPS) [1]. Through the tight integration of the computing and communication in the cyber world with the physical world, CPS will transform how humans interact with and control the physical world [2]. Distributed multi-robot systems constitute a typical application of CPS [1, 3]. Various communication and network effects have already been considered in multi-robot CPS, for example, reconfiguration [4], time-delay [5], node failure [6], and packet loss [7]. However, dynamically formed heterogeneous multi-robot CPS induce more challenges to the control of the systems, especially when there is little knowledge about the environments [8]. There is a need for novel distributed real-time communication and control strategies with heterogeneous multi-robot CPS [9].

Multi-robot coverage control has attracted considerable attention because of its versatile applications in demining, search and rescue, and monitoring and surveillance [10–13]. Take oil leakage [14] or wildfire [15] scenario as an example; coverage control tasks may be interpreted as oil cleaning or fire extinguishing. Most of the existing results assume that the task domain is static and known beforehand and consider first-order kinematics for homogeneous agents. However, in many practical

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.

applications, the task domains are dynamically changing and unknown beforehand. For example, in a large-scale oil leakage monitoring application [14], the polluted area is unknown before the monitoring task and the leaked oil is constantly diffusing to nearby areas. Furthermore, the coverage control laws developed for autonomous agents may not be directly applicable to robots. These make effective coverage control in large-scale unknown environments an open challenge for researchers.

In this chapter, we discuss the utilization of heterogeneous multi-robot systems to perform coverage control task in large-scale unknown environments based on the work proposed in [13] for awareness coverage control in foreknown task domains using homogeneous multi-robot systems. The state of awareness represents how aware each agent/robot is of the event occurring at the domain. In order to achieve full awareness of the task domain and improve task efficiency, the coverage task in the unknown environment is decomposed into two distinct, however, closely related subtasks, that is, domain boundary tracking and coverage control in the currently known domain. Motion control strategies for the heterogeneous multi-robot team are developed to achieve satisfactory awareness of the unknown task domain.

We first review some related literature on coverage control for unknown environments. In [16], the deployment of mobile sensor network in unknown environments is considered, which however requires a large number of nodes to cover a large-scale area. In [17], Morse decomposition is developed to provide a coverage path planner for the robot to achieve complete coverage in unknown environments. In [18], an online terrain coverage algorithm is developed for an autonomous underwater vehicle in an unknown underwater environment. Multi-robot exploration of an unknown environment with reduced odometry error is investigated in [19]. Dynamic coverage of mobile objects is discussed in [11] based on maximum entropy principle with first-order kinematics for agents. However, coverage control strategies formulated in [17–19] only considered one or two robots, which are not scalable for multi-robot systems.

Heterogeneous multi-robot systems composed of complementary types of robots offers great flexibility and capabilities over homogeneous systems in performing complex tasks [20]. However, heterogeneity in multi-robot teams introduces a particular challenge to efficiently coordinate the different capabilities of the robots [21]. Reliable coverage control using both coverage and coordination vehicles are investigated in [22]. In [23], distributed coverage control laws for a network of heterogeneous robots are derived by incorporating location optimization framework to cover nonconvex task domain. However, only simplified single-integrator dynamics and omnidirectional sensor models are adopted in both [22] and [23]. A decentralized cooperative control strategy for heterogeneous vehicle groups composed of both ground vehicles and aerial vehicles is proposed in [24] for formation control problem.

In contrast to most works on coverage control which consider single-integrator agents with isotropic sensors, we model wheeled mobile robots (WMRs) with nonholonomic constraints and nonisotropic sensors in this chapter. In [10], coverage control using WMRs is investigated. In [25], gradient-descent coverage algorithms are presented for a group of nonholonomic vehicles via hybrid modeling. However, both works imply that the task domain is small scale, that is, the union of sensor regions can cover the entire domain. In [26], limited-range anisotropic sensor model is adopted for coverage control using multi-robot network, but first-order kinematics for both angular and linear velocities are adopted. In [27], the effect of nonisotropic sensor model is investigated in Voronoi-based coverage control problem, but the assumption of fixed and equal sensor orientation limits its application.

In summary, the main contribution of this chapter is threefold: (i) performing coverage control in large-scale unknown environments; (ii) the consideration of nonholonomic WMR model and nonisotropic limited-range sensor model in the derivation of coverage control laws, which enables a practical coverage control strategy ready to be implemented on physical robots; and (iii) coordinating heterogeneous robots with complementary capabilities to collaboratively explore the geometry of unknown task domain and achieve full coverage of the domain. The use of heterogeneous multi-robot systems contributes to higher efficiency and improved task performance.

The organization of the rest of the chapter is as follows. In Section 10.2, we introduce the robot models, the sensor models, communication strategies, and the state of awareness dynamics. In Section 10.3, distributed motion control strategies for both the boundary-tracking unmanned aerial vehicles (UAVs) and the ground coverage WMRs are developed to collaboratively achieve full awareness of the unknown task domain. Detailed simulation results are demonstrated in Section 10.4 to confirm the effectiveness of the overall control strategy. We conclude the chapter in Section 10.5.

10.2 Problem Formulation

In this chapter, a heterogeneous multi-robot team will be used to perform awareness coverage control in unknown environments. Two types of robots, boundary-tracking UAVs and ground coverage WMRs, are considered to perform two distinct, however, closely related subtasks simultaneously and communicate with each other in real time. The boundary-tracking UAVs equipped with down-facing board view cameras will be used to track the boundary of the task domain and provide map information to the coverage robots. The UAVs can move quickly and maintain a minimum altitude; however, it cannot lift a heavy payload and has to delegate the analysis of its sensor data to an off-board computer. Meanwhile, each ground coverage WMR is equipped with camera-like sensors with limited sensing capabilities to collaboratively perform awareness coverage control task in the domain detected by boundary-tracking UAVs. The WMRs move relatively slow but can carry more sensors and perform onboard computation.

10.2.1 Robot Models

Denote the set of boundary-tracking UAVs by $\mathcal{N} = \{1, 2, \dots, n\}$ and the set of ground coverage WMRs by $\mathcal{M} = \{1, 2, \dots, m\}$. Let the unknown domain to be covered by the robots be $D_f \subset \mathbb{R}^2$. We assume that the task domain is

1) convex and bounded without any inner island;
2) unknown to the robots before the task;

3) large scale, that is, the union of the sensing and communication regions of the coverage robots cannot cover the entire unknown task domain.

The boundary-tracking UAVs (e.g., AR. Drone, AscTec Pelican quadrotors) are assumed to be hovering at a fixed altitude z_i with constant roll and pitch angle. Hence, the pose of an UAV can be simplified as a planar representation $\mathbf{p}_i^b = [x_i^b, y_i^b, \theta_i^b]^T \in \mathbb{R}^3, i \in \mathcal{N}$, where x_i^b, y_i^b are the Cartesian coordinates of the UAV's center of mass projected onto the horizontal plane, θ_i^b is the orientation of the UAV in the plane. Following [28, 29], under the assumption of fixed altitude and ignoring wind effect in the environments, the simplified 2D dynamics of an UAV can be given as follows:

$$\dot{\mathbf{p}}_i^b = \begin{bmatrix} \dot{x}_i^b \\ \dot{y}_i^b \\ \dot{\theta}_i^b \end{bmatrix} = \begin{bmatrix} v_i^b \cos \theta_i^b \\ v_i^b \sin \theta_i^b \\ \omega_i^b \end{bmatrix}, \quad i \in \mathcal{N}, \tag{10.1}$$

where v_i^b, ω_i^b are the forward linear velocity and angular velocity of the boundary-tracking UAV. Note that here v_i^b are ω_i^b command velocities instead of inertial velocities under the assumptions given in [29]. Denote $\mathbf{p}_i^{br} = [x_i^b, y_i^b]^T \in \mathbb{R}^2$ as the reduced 2D form of $\mathbf{p}_i^b \in \mathbb{R}^3$ with no angular component.

For the ground coverage WMRs, nonholonomic differential drive robots (e.g., Khepera III, P3DX) are considered (Figure 10.1). Denote the pose of the coverage robot as

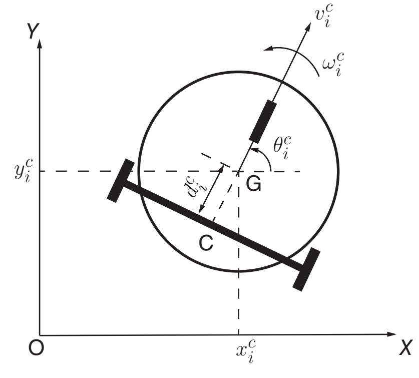


Figure 10.1 Nonholonomic differential drive wheeled mobile robot model.

$\mathbf{p}_i^c = [x_i^c, y_i^c, \theta_i^c]^T \in \mathbb{R}^3, i \in \mathcal{M}$, where x_i^c, y_i^c are the Cartesian coordinates of the center of mass, and θ_i^c is the robot orientation in the inertial Cartesian frame. Denote $\mathbf{p}_i^{cr} = [x_i^c, y_i^c]^T \in \mathbb{R}^2$ as the reduced 2D form of $\mathbf{p}_i^c \in \mathbb{R}^3$ with no angular component.

The kinematic equation of motion of a differential drive WMR satisfying the nonholonomic constraints is

$$\dot{\mathbf{p}}_i^c = \begin{bmatrix} \dot{x}_i^c \\ \dot{y}_i^c \\ \dot{\theta}_i^c \end{bmatrix} = \begin{bmatrix} v_i^c \cos\theta_i^c - d_i^c \omega_i^c \sin\theta_i^c \\ v_i^c \sin\theta_i^c + d_i^c \omega_i^c \cos\theta_i^c \\ \omega_i^c \end{bmatrix}, \quad i \in \mathcal{M}, \tag{10.2}$$

where v_i^c, ω_i^c are the forward linear velocity and angular velocity of the robot, and d_i^c is the offset between the center of mass G and the middle point C of the rear axle as shown in Figure 10.1. The nonholonomic constraints [30]

$$\dot{x}_i^c \sin\theta_i^c - \dot{y}_i^c \cos\theta_i^c + d_i^c \dot{\theta}_i^c = 0 \tag{10.3}$$

restrict the robot to move only in the direction normal to the axis of the driving wheels as long as the WMR satisfies the condition of pure rolling and nonslipping.

10.2.2 Sensor Models

A downward-facing vision sensor model with rectangular field-of-view is adopted for the UAVs as illustrated in Figure 10.2. The UAVs are assumed to be flying at a fixed altitude z_i with constant roll and pitch angles. Hence, the area of the UAV's field-of-view is also fixed. This fixed altitude of

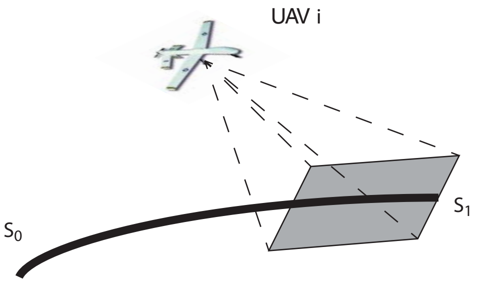
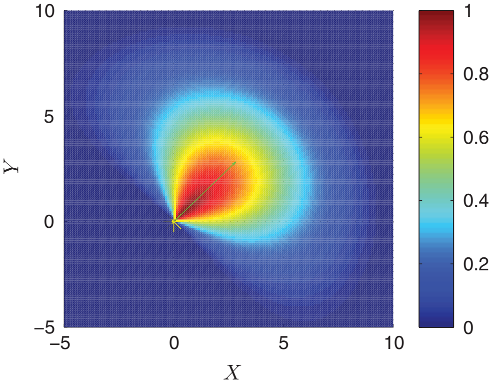


Figure 10.2 An UAV tracks the boundary of the task domain. The boundary line is represented by the solid line starts from S_0 and ends at S_1. The shaded area represents the camera sensing range.

Figure 10.3 Camera sensor model with pose $\mathbf{p}_i^c = [0, 0, \frac{\pi}{4}]$, where $M_i = 1$, $R_i = 10, \Theta_i = \frac{5}{9}\pi$.



the UAVs can be determined by maximizing the area that can be detected by the vision sensor while preserving enough resolution for accurate boundary tracking.

A vision-based nonisotropic camera sensor model, which is inspired by Wang and Hussein [31] and illustrated in Figure 10.3, is adopted for the ground coverage WMRs. We assume that the camera is mounted in the front of the robot and the orientation vector of the camera is the same as that of the robot and can be denoted as $\vec{i}(\mathbf{p}_i^c)$. Let $A_i(d_i, \alpha_i)$ be the coverage function of the sensor in the body-fixed polar frame of the WMR

$$A_i(d_i, \alpha_i) = \begin{cases} \frac{M_i(d_i^2 - R_i^2)^2(\alpha_i^2 - \Theta_i^2)^2}{R_i^4 \Theta_i^4} & \text{if } d_i \leq R_i, \alpha_i \leq \Theta_i \\ 0 & \text{otherwise,} \end{cases} \tag{10.4}$$

where R_i and Θ_i is the maximum radial and angular sensing range, respectively, $d_i = \|\mathbf{p}_i^{cr} - \tilde{\mathbf{q}}\|, \forall \tilde{\mathbf{q}} \in D_f$ is the radial distance from the sensor position to an arbitrary point $\tilde{\mathbf{q}}$ in the unknown domain D_f, and $\alpha_i = \angle(\vec{i}(\mathbf{p}_i^c), \tilde{\mathbf{q}} - \mathbf{p}_i^{cr})$ represents the angle from the maximum sensing direction. When the value of d_i and α_i increases, the sensing ability of the camera decreases until it becomes zero at $d_i = R_i$ and $\alpha_i = \Theta_i$. The maximum sensing capability is denoted as M_i at $d_i = 0$ and $\alpha_i = 0$. We denote the sensing domain of robot i at time t as $\mathcal{W}_i^c(t)$. Depending on specific applications, any other (non)isotropic sensor models can also be considered and the control laws to be developed in Section 10.3 will still hold.

10.2.3 Communication Strategies

Let the communication range of boundary-detecting UAVs and coverage robots be ρ^b and ρ^c, respectively. The boundary-tracking UAVs incrementally gather map information of the unknown task domain and broadcast their map information to the neighboring ground coverage WMRs. It is assumed that ρ^b is much larger than ρ^c, and a boundary-detecting UAV can always broadcast its map information to at least one neighboring coverage robot during the task execution process. For UAV $i \in \mathcal{N}$, let

$$\mathcal{G}^b_{i,\mathcal{N}} = \{j \in \mathcal{N} \,|\, \|\mathbf{p}^{br}_j - \mathbf{p}^{br}_i\| \le \rho^b, i \neq j\}$$

be the set of neighboring UAVs of i, and

$$\mathcal{G}^b_{i,\mathcal{M}} = \{j \in \mathcal{M} \,|\, \|\mathbf{p}^{br}_j - \mathbf{p}^{cr}_i\| \le \rho^c, i \neq j\}$$

be the set of neighboring coverage robots. Similarly, for coverage robot $i \in \mathcal{M}$, let

$$\mathcal{G}^c_{i,\mathcal{N}} = \{j \in \mathcal{N} \,|\, \|\mathbf{p}^{cr}_j - \mathbf{p}^{br}_i\| \le \rho^b, i \neq j\}$$

be the set of neighboring UAVs, and

$$\mathcal{G}^c_{i,\mathcal{M}} = \{j \in \mathcal{M} \,|\, \|\mathbf{p}^{cr}_j - \mathbf{p}^{cr}_i\| \le \rho^c, i \neq j\}$$

be the set of neighboring coverage robots.

Four different types of communication strategies are considered here. Denote $i, j \in \mathcal{N}$ as two boundary-detecting UAVs, and $k, l \in \mathcal{M}$ as two coverage robots.

- Type 1: When $\|\mathbf{p}^{cr}_k - \mathbf{p}^{cr}_l\| \le \rho_c$, two coverage robots k and l update both map and awareness coverage information with each other.
- Type 2: When $\rho_c < \|\mathbf{p}^{br}_k - \mathbf{q}^{cr}_i\| \le \rho_b$, boundary-detecting UAV i broadcasts map information to coverage robot k. Note that this type of communication is unidirectional.
- Type 3: When $\|\mathbf{p}^{br}_k - \mathbf{q}^{cr}_i\| \le \rho_c$, the coverage robot k not only gains new map information from the UAV i, but also relays other UAVs' map information to the neighboring UAV i.
- Type 4: When $\|\mathbf{p}^{br}_k - \mathbf{q}^{br}_l\| \le \rho_b$, two boundary-tracking UAVs i and j update map information with each other.

10.2.4 State of Awareness Dynamics

We now introduce the notion of state of awareness proposed in [13]. In the distributed framework, each robot has its own state of awareness distribution $\mathbf{x}_i(\tilde{\mathbf{q}}, t)$ as a measure of how "aware" the robot is of the event occurring at every point $\tilde{\mathbf{q}} \in D_f$ at time t. Without loss of generality, assume that $\mathbf{x}_i(\tilde{\mathbf{q}}, t) \in [-1, 0]$. The initial awareness is assumed to be $\mathbf{x}_i(\tilde{\mathbf{q}}, t_0) = -1, \forall \tilde{\mathbf{q}} \in D_f$ to represent no awareness at the beginning of the task. More negative values can be used to indicate the lack of awareness. The desired full awareness is set to be 0. The awareness dynamics are given as

$$\dot{\mathbf{x}}_i(\tilde{\mathbf{q}}, t) = -A_i(\tilde{\mathbf{q}}, \mathbf{p}_i^c)\mathbf{x}_i(\tilde{\mathbf{q}}, t), i \in \mathcal{M}. \tag{10.5}$$

Since A_i is a function of both $\tilde{\mathbf{q}}$ and $\mathbf{p}_i^c$ according to (10.4), we use the notation $A_i(\tilde{\mathbf{q}}, \mathbf{p}_i^c)$ here for the sake of simplicity.

Similarly, the dynamics of the overall awareness achieved by all the coverage robots in $\mathcal{M}$ can be defined as

$$\dot{\mathbf{x}}(\tilde{\mathbf{q}}, t) = -\sum_{i\in\mathcal{M}} A_i(\tilde{\mathbf{q}}, \mathbf{p}_i^c)\mathbf{x}(\tilde{\mathbf{q}}, t). \tag{10.6}$$

Whenever there are coverage robots within the communication range of robot i, they will share awareness information with each other at these moments. During these intermittent information sharing of robot i, the awareness update dynamics are given by

$$\begin{aligned} &\mathbf{x}_i(\tilde{\mathbf{q}}, t^+) = -1, \tilde{\mathbf{q}} \in D_i(t^+)\backslash D_i(t), \\ &\dot{\mathbf{x}}_i(\tilde{\mathbf{q}}, t^+) = \underbrace{-\mathbf{x}_i(\tilde{\mathbf{q}}, t)\sum_{j\in\mathcal{G}_i} \max\,(0, \mathbf{x}_j(\tilde{\mathbf{q}}, t) - \mathbf{x}_i(\tilde{\mathbf{q}}, t))}_{\text{neighbors' awareness}} \\ &\qquad \underbrace{-A_i(\tilde{\mathbf{q}}, \mathbf{p}_i^c)\mathbf{x}_i(\tilde{\mathbf{q}}, t)}_{\text{self-awareness}}, \tilde{\mathbf{q}} \in D_i(t^+), \end{aligned} \tag{10.7}$$

where t^+ represents the instantaneous information sharing moment. We assume that the UAVs will share mapping information with the UGVs and $D_i(t)$ is the dynamically evolving map of the unknown domain stored within robot $i \in \mathcal{M} \cup \mathcal{N}$ at time t before the update, and $D_i(t^+)$ is the map updated at t^+

$$D_i(t^+) = \begin{cases} \{\tilde{\mathbf{q}} \in \Omega(t) | \delta\Omega(t) = \bigcup_{k=1}^{n} C_{i,k}(t) \bigcup_{k=1}^{n} E_{i,k}(t)\}, & t \leq t_f \\ D_f, & t > t_f \end{cases}$$

where $C_{i,k}(t)$ denotes robot i's knowledge of the partial boundary information provided by boundary-tracking UAV $k \in \mathcal{N}$ at time t, $E_{i,k}(t)$ denotes the missing part of the boundary between neighboring $C_{i,k}(t)$ constructed by linear interpolation, $\delta\Omega(t)$ denotes the boundary of $\Omega(t)$, t_f is the time when the entire task domain is successfully constructed. Figure 10.4 shows an example of constructing the map using three UAVs.

Because the map of the unknown domain provided by the boundary-tracking UAVs is gradually increasing before the whole map is complete, all the points $\tilde{\mathbf{q}} \in D_i(t^+)\backslash D_i(t)$ in the newly gained part of the map at time step t^+ are assigned with initial awareness of -1. The dynamics for awareness update under intermittent communications are composed of two parts: the awareness gained from robot i's neighbors and the awareness gained from sensor measurements of the coverage robot i itself as given by (10.5). The dynamics of awareness gained from the neighbors are developed based on two intuitive beliefs: (i) a robot will gain more awareness information from its neighbors if its neighbors have better awareness. This is indicated by the term max $(0, \mathbf{x}_j(\tilde{\mathbf{q}}, t) - \mathbf{x}_i(\tilde{\mathbf{q}}, t))$; and (ii) a robot with less awareness information will gain more awareness through the interactions with its neighbors as shown by the term $-\mathbf{x}_i(\tilde{\mathbf{q}}, t)$. The awareness gained from neighbors will become zero when every coverage robot reaches the same awareness level or when the neighboring robots lose communication. At these moments, the state of awareness update dynamics is switched back to (10.5).

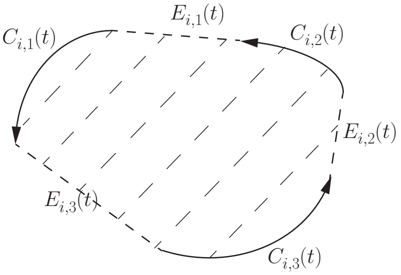


Figure 10.4 An illustrative example of robot i's map construction with information from three boundary-tracking UAVs.

Remark 10.2.1 Note that the newly added term in (10.7) will not cause any unbounded decrease or instability in awareness update. On the contrary, it will contribute to increased speed of convergence to the desired awareness level (see the detailed analysis in Section 10.3.2 in the proof of Theorem 10.3.4).

10.3 Cooperative Control of Heterogeneous Multi-Robot Systems

10.3.1 Motion Control for Boundary-Tracking UAVs

As illustrated in Figure 10.2, a boundary-tracking UAV equipped with down-facing camera extracts the boundary geometry of the task domain from captured images and then tracks the boundary trajectory as flight guidance. Similar to most flight applications, we adopt the inner and outer loop feedback control strategy [28]. For the inner loop feedback control, an autopilot is adopted to maintain a fixed altitude of the UAV. For the outer loop feedback control, the wall follower motion control algorithm in [32] is adopted to track the boundary of the task domain

$$\mathbf{u}_i^b(t) = \begin{bmatrix} v_i^b(t) \\ \omega_i^b(t) \end{bmatrix} = \begin{bmatrix} v_{i0}^b \\ \dfrac{-k_v \dot{d}_i^r + k_p(d_{i0}^r - d_i^r)}{v_{i0}^b \cos\theta_i^r} \end{bmatrix}, \tag{10.8}$$

where $\mathbf{u}_i^b(t)$ is the motion control law for the boundary-tracking UAV i, $v_i^b(t)$ and $\omega_i^b(t)$ are the corresponding linear and angular velocity control components as given in (10.1), d_i^r and θ_i^r are the relative distance and relative angle between the pose of the UAV projected on the horizontal plane and the closest segment of the domain boundary, v_{i0}^b is some constant linear velocity, d_{i0}^r is the desired distance from the boundary, and k_v, k_p are some positive designed controller gains.

10.3.2 Awareness Coverage Control for Coverage Robots

10.3.2.1 Awareness Metric

Consider the following global awareness metric

$$e_{gi}(t) = \int_{\mathcal{D}_i(t)} \frac{1}{2}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}, i \in \mathcal{M}, \tag{10.9}$$

which represents the global awareness error over the entire task domain up to time t as achieved by coverage robot i. The coverage goal of each coverage robot is to guarantee that the global awareness metric decreases and ultimately converges to a neighborhood of zero.

In a similar manner, we can define the local awareness error function for coverage robot i. This metric is used to develop a nominal motion control law for the robot, which will drive the awareness error within the sensory domain to a neighborhood of zero

$$e_i(t) = \int_{\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}. \tag{10.10}$$

10.3.2.2 Domain Coverage Algorithm

A distributed motion control law $\mathbf{u}_i^c(t)$ is developed based on the global awareness metric (10.9) and local awareness error function (10.10) to achieve full awareness of the unknown task domain. The overall control law $\mathbf{u}_i^c(t)$ consists of a nominal control law $\overline{\mathbf{u}}_i^c(t)$ and a perturbation control law $\overline{\overline{\mathbf{u}}}_i^c(t)$. At the beginning of the coverage task, coverage robot $i \in \mathcal{M}$ adopts the nominal control law, which drives the robot to minimize its local awareness error in $\mathcal{W}_i^c(t)$. Compared to static coverage schemes, the nominal control law drives the robot toward the point with least awareness within the sensory domain such that the local awareness level can be achieved dynamically. The local awareness error will be eventually driven to a neighborhood of zero. When $\|\mathbf{x}_i(\tilde{\mathbf{q}}, t)\| \leq \epsilon, \forall \tilde{\mathbf{q}} \in \mathcal{W}_i^c(t)$, where ϵ is defined as the awareness threshold value, the robot is said to have achieved local minimum. Then, the coverage robot will switch to the perturbation control law, which drives the robot out of local minimum to the area with insufficient awareness. By switching back and forth between those two control laws, full awareness will eventually be achieved in the entire task domain.

10.3.2.2.1 Nominal Control Law

Here, we develop a novel nominal control law $\overline{\mathbf{u}}_i^c(t)$ considering the nonholonomic constraints (10.3) of WMRs as well as the nonisotropic sensor model (10.4). We prove that this control law will eventually drive $\mathbf{x}_i(\tilde{\mathbf{q}}, t)$ to zero for every point in the sensory domain $\mathcal{W}_i^c$, that is, the full local awareness has been achieved. Let

$$\bar{\mathbf{u}}_i^c(t) = \begin{bmatrix} v_i^c(t) \\ \omega_i^c(t) \end{bmatrix} = \begin{bmatrix} -\mathrm{sat}_{v^c_{\max}}(h(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t))) \\ -\mathrm{sat}_{\omega^c_{\max}}(f(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t))) \end{bmatrix} \tag{10.11}$$

$$h(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t)) = \oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot \vec{i}(\mathbf{p}_i^c)\cdot \vec{n}(s,t)\mathrm{d}s$$

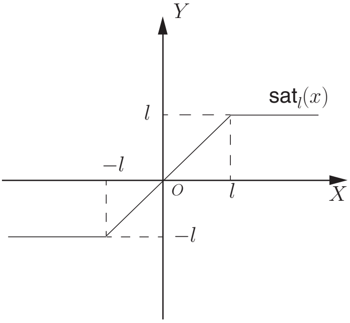
$$f(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t)) = \oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot r(s)\cdot \vec{j}(s,t)\cdot \vec{n}(s,t)\mathrm{d}s,$$

where $v^c_{\max}$ and $\omega^c_{\max}$ are the maximum linear velocity and angular velocity allowed by the robot. A saturation function [33] illustrated in Figure 10.5 is used to set a limit on $v_i^c(t)$ and $\omega_i^c(t)$ so that the linear and angular velocity control will not exceed the robot's capability. We use $\mathrm{sat}_l(x) : \mathbb{R} \to \mathbb{R}$ to denote the saturation function

$$\mathrm{sat}_l(x) = \mathrm{sgn}(x)\ \min\{l, |x|\}. \tag{10.12}$$

For a vector $\mathbf{u} \in \mathbb{R}^b$, we use $\mathrm{sat}_{\mathbf{l}}(\mathbf{u})$ to denote the vector saturation function, where $\mathrm{sat}_{\mathbf{l}}(\mathbf{u}) = [\mathrm{sat}_{l_1}(u_1), \mathrm{sat}_{l_2}(u_2), \dots, \mathrm{sat}_{l_b}(u_b)]$. $\delta\mathcal{W}_i^c(t)$ is the boundary of the local sensory domain $\mathcal{W}_i^c(t)$. The dummy variable s is the curvilinear coordinate of a point $\tilde{\mathbf{q}}$ on $\delta\mathcal{W}_i^c(t)$ as illustrated in Figure 10.6, ds is the element of curve length. The curvilinear coordinate of a point is given by the curve length from $\mathbf{p}_i^c$ to that point along the boundary curve of the sensory domain in the counterclockwise direction. $\vec{i}(\mathbf{p}_i^c)$ is the orientation of the robot as well as the sensor, $\vec{n}(s,t)$ is the normal unit vector at point $\tilde{\mathbf{q}}$ with respect to $\delta\mathcal{W}_i^c(t)$, $\vec{j}(s,t)$ is a

Figure 10.5 Saturation function.



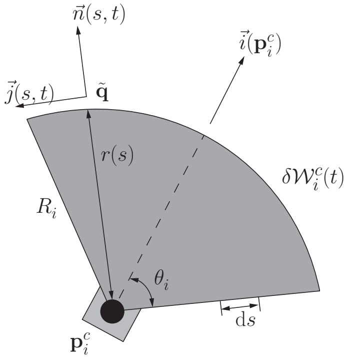


Figure 10.6 Vectors within the sensor range.

unit vector in the angular velocity's direction at point $\tilde{\mathbf{q}}$, and the distance is $r(s) = \|\mathbf{p}_i^{cr} - \tilde{\mathbf{q}}\|$.

We first consider the following 2D Leibniz rule [34], which will be applied to prove the convergence of local awareness error function (10.10) under the nominal control law.

Lemma 10.3.1 For any function $F : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}$, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\mathcal{W}_i^c(t)} F(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}$$
$$= \underbrace{\oint_{\delta\mathcal{W}_i^c(t)} F(s,t) \cdot \vec{v}(s,t) \cdot \vec{n}(s,t)\mathrm{d}s}_{\text{Motion of domain}} + \underbrace{\int_{\mathcal{W}_i^c(t)} \frac{\partial}{\partial t} F(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}}_{\text{Change of integrand}},$$

where $\vec{v}(s,t)$ denotes the velocity of a point $\tilde{\mathbf{q}} \in \delta\mathcal{W}_i^c(t)$ with curvilinear coordinate s on the boundary of sensory domain.

Proof: The derivation of the 2D Leibniz rule, also known as the differentiation under integral sign for time-varying planar domain, is given in [34]. By applying the chain rule, the differentiation is separated into two terms: contribution from the motion of domain and the change of integrand function. In our case, the motion of the domain is governed by the velocity $\mathbf{u}_i^c = [v_i^c, \omega_i^c]^T$ of the coverage robot, and the change of integrand function is governed by the dynamics of awareness update. ■

Next, consider the following condition.

Condition 10.3.2 $\mathbf{x}_i(\tilde{\mathbf{q}}, t) = 0, \forall \tilde{\mathbf{q}} \in \mathcal{W}_i^c(t)$.

This condition corresponds to the case when all the points in the coverage robot i's sensory domain $\mathcal{W}_i^c(t)$ has full awareness coverage and the local awareness error $e_i(t)$ is zero.

Lemma 10.3.3 For any $t \geq 0$, Condition 10.3.2 holds for coverage robot i, if and only if $e_i(t) = 0$ for some time $t \geq 0$ for coverage robot i.

Proof: See [13] for detailed proof. ■

Theorem 10.3.4 The nominal control law $\overline{\mathbf{u}}_i^c(t)$ given by (10.11) can guarantee that the local awareness error function $e_i(t)$ converges asymptotically to zero, under the velocity saturation constraints as well as the intermittent awareness updates with neighboring coverage robots.

Proof: Take the local awareness error function (10.10) as the Lyapunov function $V(t) = e_i(t) \geq 0$. From Lemma 10.3.3, $V(t) = 0$ if and only if Condition 10.3.2 holds. Thus, function $V(t)$ is positive definite.

Next, take the derivative of the Lyapunov function, according to Lemma 10.3.1, we get

$$\begin{aligned}\dot{V}(t) &= \frac{\mathrm{d}}{\mathrm{d}t} \int_{\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}} \\ &= \underbrace{\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s, t) \cdot \vec{v}(s, t) \cdot \vec{n}(s, t)\mathrm{d}s}_{\text{First term}} \\ &\quad + \underbrace{\int_{\mathcal{W}_i^c(t)} \frac{1}{2}\frac{\partial}{\partial t}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}}_{\text{Second term}}. \end{aligned} \tag{10.13}$$

By decomposing the velocity vector into linear and angular velocities, we can get $\vec{v}(s, t) = v_i^c(t) \cdot \vec{i}(\mathbf{p}_i^c) + \omega_i^c(t) \cdot r(s) \cdot \vec{j}(s, t)$.

Thus, the first term in (10.13) can be rewritten as

$$\begin{aligned}&\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot[v_i^c(t)\cdot\vec{i}(\mathbf{p}_i^c)+\omega_i^c(t)\cdot r(s)\cdot\vec{j}(s,t)]\cdot\vec{n}(s,t)\mathrm{d}s\\ &\quad=\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot v_i^c(t)\cdot\vec{i}(\mathbf{p}_i^c)\cdot\vec{n}(s,t)\mathrm{d}s\\ &\quad\quad+\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot\omega_i^c(t)\cdot r(s)\cdot\vec{j}(s,t)\cdot\vec{n}(s,t)\mathrm{d}s.\end{aligned}$$

Because $v_i^c(t)$, $\omega_i^c(t)$ are only functions of time but not s, they can be pulled outside of the integration. Based on (10.11), we have

$$\begin{aligned}&v_i^c(t)\cdot\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot\vec{i}(\mathbf{p}_i^c)\cdot\vec{n}(s,t)\mathrm{d}s\\ &\quad+\omega_i^c(t)\cdot\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot\vec{j}(s,t)\cdot\vec{n}(s,t)\mathrm{d}s\\ &\quad=-\operatorname{sat}_{v^c_{\max}}(h(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t)))\cdot h(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t))\\ &\quad\quad-\operatorname{sat}_{\omega^c_{\max}}(f(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t)))\cdot f(\mathbf{x}_i(\tilde{\mathbf{q}},t),\mathbf{p}_i^c(t)).\end{aligned}$$

Because the saturation applied on the linear and angular velocities of the robot does not change the direction of the velocity, we prove that the first term of (10.13)

$$\oint_{\delta\mathcal{W}_i^c(t)} \frac{1}{2}\mathbf{x}_i^2(s,t)\cdot\vec{v}(s,t)\cdot\vec{n}(s,t)\mathrm{d}s\le 0. \tag{10.14}$$

It also shows that the saturation on $\overline{\mathbf{u}}_i^c(t)$ will not affect the stability proof.

For the second term, when there are no intermittent communications between neighboring coverage robots, consider the dynamics of awareness in (10.5), we obtain

$$\int_{\mathcal{W}_i^c(t)} \frac{1}{2}\frac{\partial}{\partial t}\mathbf{x}_i^2(\tilde{\mathbf{q}},t)\mathrm{d}\tilde{\mathbf{q}}=-\int_{\mathcal{W}_i(t)} \mathbf{x}_i^2(\tilde{\mathbf{q}},t)A_i(\tilde{\mathbf{q}},\mathbf{p}_i^c)\mathrm{d}\tilde{\mathbf{q}}\le 0. \tag{10.15}$$

When the effect of neighboring robots is taken into consideration during intermittent communications, both the awareness gained from sensor measurements and neighbors are incorporated. According to (10.7), we can get the following:

$$\begin{aligned}\int_{\mathcal{W}_i^c(t)} \frac{1}{2}\frac{\partial}{\partial t}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}} \\ = -\int_{\mathcal{W}_i^c(t)} \mathbf{x}_i^2(\tilde{\mathbf{q}}, t)[A_i(\tilde{\mathbf{q}}, \mathbf{p}_i^c) \\ + \sum_{j\in\mathcal{G}_i} \max\,(0, \mathbf{x}_j(\tilde{\mathbf{q}}, t) - \mathbf{x}_i(\tilde{\mathbf{q}}, t))]\mathrm{d}\tilde{\mathbf{q}} \le 0. \qquad (10.16)\end{aligned}$$

Therefore, the awareness updates will not cause instability of the system, instead it contributes to increased speed of convergence. According to (10.13)–(10.16), the derivative of the Lyapunov function $\dot{V}(t) \le 0$.

Note that the equality holds if and only if Condition 10.3.2 holds. This can be seen as follows: First, if Condition 10.3.2 holds, both the first and second term in (10.13) equal to zero, hence $\dot{V}(t) = 0$. Second, if $\dot{V}(t) = 0$, but $\exists\tilde{\mathbf{q}} \in \mathcal{W}_i(t)$, such that $\mathbf{x}_i(\tilde{\mathbf{q}}, t) \ne 0$, the second term in (10.13) will always be nonzero. Thus, function $\dot{V}(t)$ is negative definite. According to asymptotic stability theorem, the nominal control law (10.11) will drive the local awareness error function (10.10) to zero. This completes the proof. ■

10.3.2.2.2 Perturbation Control Law

It has been proved that the nominal control law (10.11) will eventually drive the local awareness metric (10.10) to zero. However, this does not necessarily guarantee that the global awareness metric (10.9) will also be driven to zero. Let us consider the following condition:

Condition 10.3.5 $\|\mathbf{x}_i(\tilde{\mathbf{q}}, t)\| \le \epsilon, \forall\tilde{\mathbf{q}} \in \mathcal{W}_i^c(t)$, where $\epsilon > 0$ is the awareness tolerance.

This condition corresponds to the case when the local awareness error is within a neighborhood of zero, that is, the robot gets “stuck” in the local region, and makes little progress in coverage task. Using the nominal control law (10.11), the coverage robot is guaranteed to reach Condition 10.3.5 according to Theorem 10.3.4. Under this circumstance, a perturbation control law is adopted to drive the robot to the area with insufficient awareness level. Let $\mathcal{E}_i(t)$ be the set of all the points in the task domain $D_i(t)$ that have lower awareness than ϵ, that is,

$$\mathcal{E}_i(t) = \{\tilde{\mathbf{q}} \in D_i(t) | \|\mathbf{x}_i(\tilde{\mathbf{q}}, t)\| \geq \epsilon\}.$$

Denote the set $\overline{\mathcal{E}}_i(t)$ as

$$\overline{\mathcal{E}}_i(t) = \{\tilde{\mathbf{q}} \in \mathcal{E}_i(t) | \tilde{\mathbf{q}} = \operatorname{argmin}_{\tilde{\mathbf{q}} \in \mathcal{E}_i(t)} \|\mathbf{p}_i^{cr} - \tilde{\mathbf{q}}\|\},$$

which contains the points in set $\mathcal{E}_i(t)$ that are also closest to the robot position $\mathbf{p}_i^{cr}$.

The perturbation control law is given as

$$\bar{\bar{\mathbf{u}}}_i^c(t) = -k_i \mathrm{sat}_{\mathbf{u}_{\max}^c}(\mathbf{p}_i^{cr} - \mathbf{q}^*(t_s)), \tag{10.17}$$

where t_s is the switching time, $\mathbf{q}^*(t_s) \in \overline{\mathcal{E}}_i(t_s)$, k_i is a feedback controller gain. A detailed discussion about the choice of $\mathbf{q}^*(t_s)$ can be found in [13].

Note that for the perturbation control law, we simply consider the X, Y coordinates of the robot and develop $\bar{\bar{\mathbf{u}}}_i^c(t) = [u_x^c, u_y^c]$ as the control law. This is because we do not have orientation requirement for the robot to move from $\mathbf{p}_i^c$ to $\mathbf{q}^*(t_s)$. Hence, the perturbation control can be easily implemented in two steps: first rotate the robot to orient to $\mathbf{q}^*(t_s)$, then drive to destination point $\mathbf{q}^*(t_s)$ with linear velocity $\bar{\bar{\mathbf{u}}}_i^c(t)$. If there is orientation requirement for $\mathbf{q}^*(t_s)$, the problem can be simplified into driving a robot from current pose $\mathbf{p}_i^c$ to arbitrary desired pose $\mathbf{p}_d$. There are abundant literature on position and orientation control of nonholonomic mobile robots. For example, the simple PI controller developed in [30] can be used to perform this task.

Theorem 10.3.6 If a coverage robot i reaches Condition 10.3.5 and the set $\mathcal{E}_i(t)$ is nonempty, then the designed perturbation control law given by (10.17) will drive the system away from Condition 10.3.5.

Proof: If Condition 10.3.5 holds and the set $\mathcal{E}_i(t)$ is nonempty, the coverage robot switches to the perturbation control law. The coverage robot i will converge asymptotically to a neighborhood of $\mathbf{q}^*(t_s)$. At this time, Condition 10.3.5 no longer holds. ■

10.3.2.2.3 Overall Control Strategy

Combine the nominal control law (10.11) and the perturbation control law (10.17) with the switching condition, we obtain the following result:

Theorem 10.3.7 The overall control law:

$$\mathbf{u}_i^c(t) = \begin{cases} \overline{\mathbf{u}}_i^c(t), & \text{if Condition 10.3.5 does not hold} \\ \overline{\overline{\mathbf{u}}}_i^c(t), & \text{if Condition 10.3.5 holds} \end{cases} \tag{10.18}$$

can drive the global awareness error metric $e_{\mathrm{gi}}(t), i \in \mathcal{M}$ to a neighborhood of zero.

Proof: According to Theorem 10.3.4, the nominal control law (10.11) will eventually drive the local awareness error (10.10) to a neighborhood of zero, that is, Condition 10.3.5. Then if the set $\mathcal{E}_i(t)$ is nonempty, the robot will switch to perturbation control law (10.17) and drive the system away from Condition 10.3.5 according to Theorem 10.3.6. Hence, the robot i will switch back to nominal control law (10.11). This procedure is repeated until such a point $\mathbf{q}^*(t_s)$ does not exist. That is to say, only when Condition 10.3.5 holds and $\|\mathbf{x}_i(\tilde{\mathbf{q}}, t)\| \leq \epsilon, \forall \tilde{\mathbf{q}} \in D_f$, the coverage task is said to be accomplished and no further switching will be performed.

To complete the proof, we need to show that there will be finite switchings between the nominal control law (10.11) and perturbation control law (10.17). During the boundary-tracking process, the set $\mathcal{E}_i(t)$ is almost always growing when $t \leq t_f$. This is because the initial awareness for the newly constructed part of the map $\tilde{\mathbf{q}} \in D_i(t^+)\backslash D_i(t)$ is always set to -1, and added into set $\mathcal{E}_i(t)$. However, after the map of the task domain D_f is fully constructed at time t_f, there will be no new elements added into set $\mathcal{E}_i(t)$. When $t > t_f$, because of the existence of the perturbation control law, the number of elements in set $\mathcal{E}_i(t)$ begins to decrease until no element is left. Because the task domain is unknown but bounded, $D_i(t)$ will reach the final map D_f in finite time and the decreases of elements in $\mathcal{E}_i(t)$ will also occur finitely. When $\mathcal{E}_i(t) = \emptyset$, it means there is no point $\tilde{\mathbf{q}} \in D_f$ with awareness level lower than the threshold ϵ. This fact guarantees that there will be finite switchings between the nominal control law and perturbation control law to reach $e_{\text{gi}}(t) \leq \bar{\epsilon}$, where $\bar{\epsilon}$ is an upper bound given by

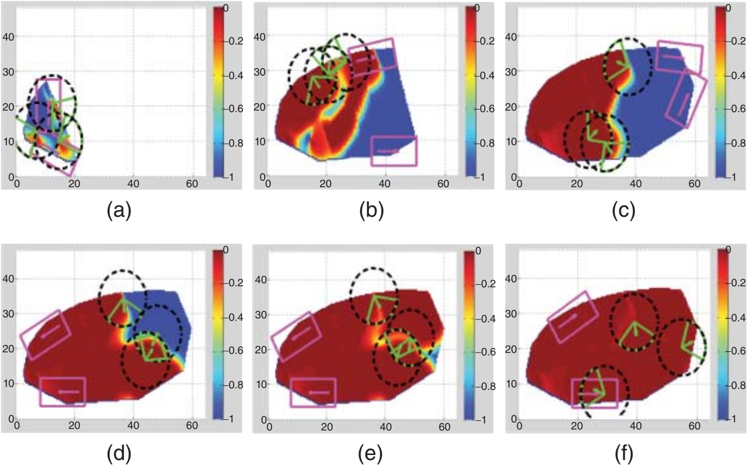
$$\begin{aligned} e_{\text{gi}}(t) &= \int\limits_{\mathcal{D}_f} \frac{1}{2}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}} = \|\int\limits_{\mathcal{D}_f} \frac{1}{2}\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\mathrm{d}\tilde{\mathbf{q}}\| \\ &= \int\limits_{\mathcal{D}_f} \frac{1}{2}\|\mathbf{x}_i^2(\tilde{\mathbf{q}}, t)\|\mathrm{d}\tilde{\mathbf{q}} \leq \frac{\epsilon^2 A_{D_f}}{2} = \bar{\epsilon}, \end{aligned}$$

where $t > t_f$, and A_{D_f} is the area of the final task domain D_f. This means that after the map of the task domain is fully constructed, the global awareness error $e_{\text{gi}}(t)$ will be driven to a neighborhood of zero, and full awareness of the entire task domain is guaranteed to be achieved. ■

10.4 Simulation Results

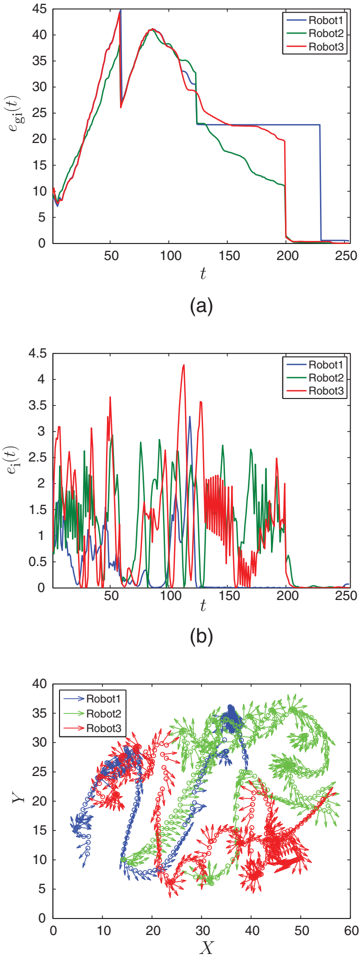
In the Matlab simulation, we consider a scenario of using two boundary-tracking UAVs and three coverage robots cooperatively to cover an unknown environment. Two boundary-tracking UAVs are sent out simultaneously from the same starting point in opposite directions of the task domain to detect the boundary of the domain. Three coverage robots are deployed near the boundary-tracking UAVs. Figure 10.7 shows the simultaneous boundary tracking and domain coverage process. The magenta arrows and rectangles represent orientations and sensing ranges (length 15 × width 8) of boundary-tracking UAVs. The arrows and sectors represent orientations and sensing ranges ($R_i = 10, \theta_i = \frac{5}{9}\pi$) of ground mobile coverage robots. It is assumed that the boundary-tracking UAVs can share their map information with the ground mobile coverage robots in real time. The communication ranges of the coverage robots are $\rho^c = 10$ and are visualized with dashed circles. The overall awareness level of the task domain is visualized with color map. For the sake of simplicity, d_i^c is set to be zero in the simulation. Note that the boundary-tracking UAVs continue flying around the boundary to broadcast map information after $t \geq t_f = 86$.

The evolution of the global awareness error metric (10.9), the local awareness metrics (10.10), and pose of the coverage robots are shown in Figure 10.8. As illustrated in Figure 10.8a, the global awareness metrics keep increasing when $t \leq t_f = 86$. This is because the map of the task domain is growing when the boundary-tracking UAVs are constructing the map and the growth rate of the map outpaces the covering rate of the coverage robots. Three abrupt drops occur at time steps $t = 58$ (as also shown in Figure 10.7b), $t = 200$ (as also shown in Figure 10.7d), and $t = 229$ (as also shown in Figure 10.7e) because the coverage robots enter into each other's neighborhood and the information sharing improves their awareness level. Time step $t = t_f = 86$ (as also shown in Figure 10.7c) corresponds to the moment when the boundary-tracking task is finished and a complete map of the task domain is obtained. Global full awareness of the unknown task domain is achieved at time step $t = 255$ (as also shown in Figure 10.7f), that is, the entire domain is covered with desired awareness level by the coverage robots. In Figure 10.8b, the local awareness metrics



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

Figure 10.7 Simulated simultaneous boundary tracking and domain coverage. Time steps (a) $t = 4$, (b) $t = 58$, (c) $t = 86$, (d) $t = 200$, (e) $t = 229$, and (f) $t = 255$.



(a)

(b)

(c)

Figure 10.8 Coverage metrics and robot poses. (a) Global awareness metrics. (b) Local awareness metrics. (c) Pose of coverage robots.

oscillate between high and low values due to the frequent switchings between the nominal control law (10.10) and perturbation control law (10.17). Both the global awareness metrics and local awareness metrics are shown to converge to zero. Figure 10.8c shows the poses of three coverage robots during the entire process. Due to the distributed nature of the control strategy, there are some redundancy in the coverage process.

10.5 Conclusion

In this chapter, we proposed a novel distributed control strategy for heterogeneous multi-robot systems to accomplish full awareness coverage of unknown environments. The task was decomposed into a boundary-tracking subtask performed by UAVs and a domain coverage subtask performed by WMRs. The nonholonomic constraints of WMR, nonisotropic sensor model, and awareness-based error metrics are considered when deriving motion control strategies. The performance of the cooperative boundary tracking and domain coverage tasks performed by the heterogeneous multi-robot system was demonstrated by Matlab simulations.

References

1 Fink, J., Ribeiro, A., and Kumar, V. (2012) Robust control for mobility and wireless communication in cyber-physical systems with application to robot teams. *Proceedings of the IEEE*, **100** (1), 164–178.

2 Rajkumar, R.R., Lee, I., Sha, L., and Stankovic, J. (2010) Cyber-physical systems: the next computing revolution, in *Proceedings of the 47th Design Automation Conference*, pp. 731–736.

3 Kim, M., Stehr, M.O., Kim, J., and Ha, S. (2010) An application framework for loosely coupled networked cyber-physical systems, IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC), December 2010, pp. 144–153.

4 Olfati-Saber, R. and Murray, R.M. (2002) Graph rigidity and distributed formation stabilization of multivehicle systems, in *Proceedings of the 41st IEEE Conference Decision and Control*, vol. **3**, pp. 2965–2971.

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

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

7 Sinopoli, B., Schenato, L., Franceschetti, M., Poola, K., Jordan, M.I., and Sastry, S.S. (2004) Kalman filtering with intermittent observations. *IEEE Transactions on Automatic Control*, **49** (9), 1453–1464.

8 Jones, E.G., Browning, B., Dias, M.B., Argall, B., Veloso, M., and Stentz, A. (2006) Dynamically formed heterogeneous robot teams performing tightly-coupled tasks, in *Proceedings of IEEE International Conference Robotics and Automation (ICRA)*, pp. 570–575.

9 Sha, L., Gopalakrishnan, S., Liu, X., and Wang, Q. (2008) Cyber-physical systems: a new frontier? IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichung, Taiwan, June 11-13, 2008.

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

11 Sharma, P., Salapaka, S.M., and Beck, C.L. (2012) Entropy-based framework for dynamic coverage and clustering problems. *IEEE Transactions on Automatic Control*, **57** (1), 135–150.

12 Hussein, I.I. and Stipanovic, D.M. (2007) Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *IEEE Transactions on Control Systems Technology*, **15** (4), 642–657.

13 Wang, Y. and Hussein, I.I. (2010) Awareness coverage control over large-scale domains with intermittent communications. *IEEE Transactions on Automatic Control*, **55** (8), 1850–1859.

14 Crone, T.J. and Tolstoy, M. (2010) Magnitude of the 2010 Gulf of Mexico oil leak. *Science*, **330** (6004), 634.

15 Yu, L., Wang, N., and Meng, X. (2005) Real-time forest fire detection with wireless sensor networks, in *International Conference on Wireless Communications, Networking and Mobile Computing*, vol. **2**, pp. 1214–1217.

16 Howard, A., Mataric, M.J., and Sukhatme, G.S. (2002) Mobile sensor network deployment using potential fields:

a distributed, scalable solution to the area coverage problem, in *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS02),* pp. 299–308.

17 Acar, E.U. and Choset, H. (2002) Sensor-based coverage of unknown environments: incremental construction of Morse decompositions. *International Journal of Robotics Research,* **21** (4), 345–366.

18 Hert, S., Tiwari, S., and Lumelsky, V. (1996) A terrain-covering algorithm for an AUV. *Autonomous Robots,* **3** (2), 91–119.

19 Rekleitis, I.M., Dudek, G., and Milios, E.E. (1997) Multi-robot exploration of an unknown environment, efficiently reducing the odometry error. *International Joint Conference on Artificial Intelligence,* **15**, 1340–1345.

20 Parker, L.E., Kannan, B., Tang, F., and Bailey, M. (2004) Tightly-coupled navigation assistance in heterogeneous multi-robot teams, in *Proceedings of IEEE International Conference on Intelligent Robots and Systems (IROS),* vol. **1**, Sendai, Japan, pp. 1016–1022.

21 Parker, L.E. (2000) Lifelong adaptation in heterogeneous multi-robot teams: response to continual variation in individual robot performance. *Autonomous Robots,* **8** (3), 239–267.

22 Hussein, I.I., Stipanovic, D.M., and Wang, Y. (2007) Reliable coverage control using heterogeneous vehicles, in *46th IEEE Conference on Decision and Control,* December 2007, pp. 6142–6147.

23 Pimenta, L., Kumar, V., Mesquita, R.C., and Pereira, G. (2008) Sensing and coverage for a network of heterogeneous robots, in *47th IEEE Conference on Decision and Control, Cancun, Mexico, December 2008,* pp. 3947–3952.

24 Tanner, H.G. and Christodoulakis, D.K. (2007) Decentralized cooperative control of heterogeneous vehicle groups. *Robotics and Autonomous Systems,* **55** (11), 811–823.

25 Kwok, A. and Martìnez, S. (2010) Unicycle coverage control via hybrid modeling. *IEEE Transactions on Automatic Control,* **55** (2), 528–532.

26 Laventall, K. and Cortés, J. (2009) Coverage control by multi-robot networks with limited-range anisotropic sensory. *International Journal of Control*, **82** (6), 1113–1121.

27 Gusrialdi, A., Hirche, S., Hatanaka, T., and Fujita, M. (2008) Voronoi based coverage control with anisotropic sensors, in *American Control Conference*, June 2008, pp. 736–741.

28 Regina, N. and Zanzi, M. (2009) 2D tracking and over-flight of a target by means of a non-linear guidance law for UAV, in *IEEE Aerospace Conference*, pp. 1–11.

29 Ren, W. and Beard, R.W. (2003) CLF-based tracking control for UAV kinematic models with saturation constraints, in *Conference on Decision and Control*, December 2003, pp. 3924–3929.

30 Bara, A. and Dale, S. (2009) Dynamic modelling and stabilization of wheeled mobile robot, in *Proceedings of the 5th WSEAS International Conference on Dynamical Systems and Control*, pp. 87–92.

31 Wang, Y. and Hussein, I.I. (2007) Cooperative vision-based multi-vehicle dynamic coverage control for underwater applications, in *IEEE International Conference on Control Applications*, pp. 82–87.

32 Das, A.K., Fierro, R., Kumar, V., Southall, B., Spletzer, J., and Taylor, C.J. (2001) Real-time vision-based control of a nonholonomic mobile robot. *IEEE International Conference on Robotics and Automation*, **2**, 1714–1719.

33 Hu, T. and Lin, Z. (2001) *Control Systems with Actuator Saturation: Analysis and Design*, Birkhauser, pp. 1–9.

34 Flanders, H. (1973) Differentiation under the integral sign. *American Mathematical Monthly*, **80** (6), 615–627.

Index

Cooperative Control of Multi-Agent Systems: Theory and Applications, First Edition.
Edited by Yue Wang, Eloy Garcia, David Casbeer, and Fumin Zhang.