MODEL PREDICTIVE CONTROL OF NONHOLONOMIC MOBILE ROBOTS

By

FENG XIE

Bachelor of Science
Zhejiang University
Hangzhou, CHINA
1997

Master of Science
Zhejiang University
Hangzhou, CHINA
2000
Oklahoma State University
Stillwater, USA
2004

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2007

MODEL PREDICTIVE CONTROL OF NONHOLONOMIC MOBILE ROBOTS

Dissertation Approved:

Dr. Martin Hagan

Committee Chair

Dr. Rafael Fierro

Thesis Adviser

Dr. James R. Whiteley

Dr. Weihua Sheng

Dr. A. Gordon Emslie

Dean of the Graduate College

PREFACE

Cooperative decision and control for mobile robot teams have been of great interest in the control community. A lot of effort has been put into this area. Groups of mobile robots working cooperatively lead to many advantages in a variety of critical applications. With the expectation that unmanned mobile robots can perform key roles in future civilian or military missions, the research on mobile robot control is likely to increase rapidly in the near future.

The objective of this dissertation is to use model predictive control (MPC) to coordinate the motion of nonholonomic mobile robots. Specifically, we consider the formation control of a group of mobile robots and trajectory tracking and point stabilization of nonholonomic vehicles.

The formation control problem is addressed by a Lyapunov-based nonlinear controller design for unmanned aerial vehicles (UAV) and in the context of MPC for nonholonomic mobile robots. For the UAV formation problem, a two-layered hierarchical control scheme is presented. Assuming that an autopilot operating in *holding mode* controls the UAV dynamics at the low-layer, a simplified nonholonomic model is constructed for the high-layer formation controller design. With the dynamic extension technique, three different nonlinear formation controllers are proposed. While the first two controllers, a feedback linearization controller and a sliding mode controller, assume full states information of the leader, the third robust controller only requires the knowledge of leader's position. By eliminating the requirement of leader's velocity and acceleration information, the robust controller reduces the inter-vehicle communication overhead and increases the reliability of the overall system. Stability properties of the controllers are proven using Lyapunov

theory. As for the nonholonomic mobile robot formation problem, we proposed a dual-mode MPC formation control algorithm. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed. For the choice of stabilizing terminal controller, a comparison between an input-output feedback linearization controller and a robust formation controller is given.

We proposed a novel *first-state contractive* MPC (FSC-MPC) approach for the problem of trajectory tracking and point stabilization of nonholonomic mobile robots. In the literature, most of the existing controllers address the trajectory tracking problem by assuming that the trajectory needed to be track is continuously excited by a reference robot. When the reference robot stops or even moves backward, most of the controller will fail. As for the point stabilization problem, discontinuous feedback controllers and time-varying algorithms are mostly found since by Brockett's theorem, smooth time-invariant feedback control laws do not exist. In addition, only a few controllers can handle the tracking and stabilization problems in the same control structure. In the literature, most of the stabilizing MPC methods address the stability by adding terminal state penalties in the performance index and/or imposing constraints on the terminal state at the end of the prediction horizon. The stability of the FSC-MPC algorithm is guaranteed by adding a contractive constraint on the first state at the beginning of the prediction horizon. With this first-state contractive constraint, the proposed FSC-MPC algorithm for nonholonomic mobile robot motion control achieves: (i) exponential stability, (ii) the ability to track a trajectory moving backward, and (iii) the capability of simultaneous tracking and point stabilization.

Simulation results are presented to verify the validity of the proposed control algorithms and demonstrate the performance of the proposed controllers.

ACKNOWLEDGMENTS

The journey to a Ph.D. degree is not easy. During this long journey, I have had the pleasure of working with several professors and colleagues, receiving help from them and learning a lot from them. I would like to express my gratitude to all of them.

I would like to thank my advisor, Dr. Rafael Fierro for his guidance, support, and enthusiasm for research. I am grateful for the freedom and flexibility he gave me for doing research in model predictive control in which I am especially interested. Thanks to Dr. James R. Whiteley for leading me into the research of model predictive control and being on my committee. Also, thanks to Dr. Martin Hagan for being the chair of my committee, and Dr. Weihua Sheng for being on my committee. Your help has been instrumental in completing the work in this dissertation.

I would like to thank Dr. Ximing (Tong) Zhang for his friendship and help on the controller presented in Chapter 4. Also many thanks to current and past members of the MARHES lab: James, Brent, Carlo, Yuan (Eric), Chris, Lorne, Justin, Kenny, and José. You made my life in Stillwater more exciting.

Especially, I would like to thank my beautiful wife, Yan and my parents. Without your supporting and encouraging, I would not be able to finish this work.

Last but not least, I gratefully acknowledge the sources of financial support. This work is supported in part by the National Science Foundation (NSF) grant CAREER #0348637 and by the U.S. Army Research Office under grant DAAD19-03-1-0142 (through the University of Oklahoma).

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

CHAPTER 1

Introduction

From the literal meaning, mobile robots are robots which can move around in their environment and are not fixed to one physical location. In recent years, the research and development of mobile robots are very active, mostly because of their better potential than other kinds of robots in replacing human beings in civilian and military applications.

By the environment in which they move, mobile robots can be classified into: ground robots (Figure 1.1), aerospace robots (Figure 1.2[1]), and underwater robots (Figure 1.3[2]).

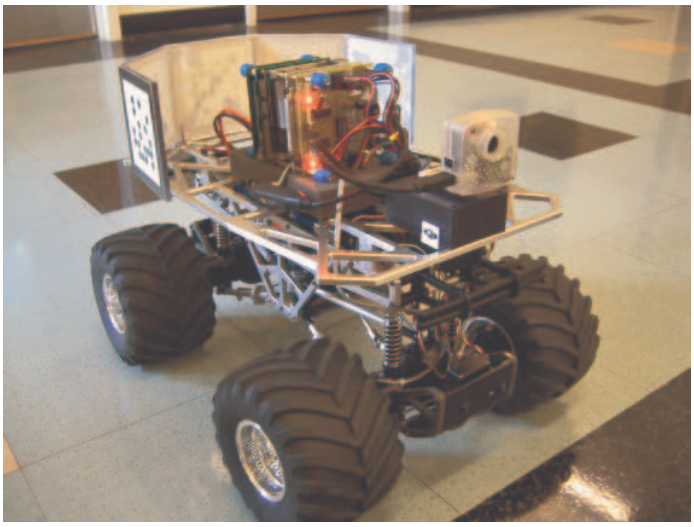


Figure 1.1: MARHES TXT-1 robot truck.

[1] http://www4.army.mil

[2] http://www.infotechfrance.com/london/upload/photo_9273.jpg

Figure 1.2: Unmanned aerial robot.

Figure 1.3: Underwater vehicle by ECA HYTEC of France.

Due to recent substantial developments in electronics and computing, it is now possible to find onboard embedded computers which have more computing power than the super computers available a few years ago. Exchanging information between mobile robots, such as unmanned aerial/ground vehicles (UAV/UGV) distributed over an area, is now possible by means of off-the-shelf ad-hoc wireless network devices. In addition, there are various small size, light weight sensing devices on the market ranging from laser range sensors to color CCD cameras. As a result, by exploiting current technology, one can

build a group of relatively small UAVs/UGVs each having satisfactory capabilities within a reasonable budget. The challenge here lies in designing control algorithms for mobile robots to perform complex tasks within dynamic environments.

1.1 Motivation

Cooperative decision and control of mobile robot teams have been of great interest in civilian and military applications, as these teams are expected to be capable of performing many key roles in future civilian or battlefield missions. A lot of effort has been put into this area and the use of mobile robots is likely to increase rapidly in the near future. Applications for cooperative control of multi-robot systems include [1]: Formation control, cooperative classification and surveillance, cooperative attack and rendezvous, environmental sampling, distributed aperture observing, intelligent highways, and air traffic control.

In application, groups of mobile robots working cooperatively lead to many advantages. Multi-robot systems are expected to outperform single-robot systems in function, fault tolerance, flexibility, size and cost [1] [2]. A short summary describing some important multi-robot capabilities is given below.

- *Distribution*: Multiple mobile robots can work simultaneously at different positions in the space. For example, during a surveillance task, the target can be monitored from different angles by a group of UAVs/UGVs distributed over the area. This will provide more detailed information of the target.

- *Multitasking*: Usually, a task can be decomposed into several sub-tasks which are capable of being handled at the same time. Parallel UAV/UGV operation can finish the task much faster than a single UAV/UGV can do. For example, during forest-fire monitoring, the time for scanning can be reduced in half with two UAVs working side by side.

- *Fault tolerance*: In a multi-robot group, robots' functions can overlap. Therefore, if one robot malfunctions, its functionality can be easily substituted by other robots. The whole group's functionality will not be affected by individual's failure. This increases the robustness of the system, which is critical in dangerous missions.

- *Flexibility*: The functionality of a group robots can be easily changed by combining different robots with different capabilities or enhanced by adding new robots.

- *cost-effectiveness*: Design a versatile robot which is capable of handling different tasks sometime might not feasible due to the limitation of robot size and payloads. However, with several robots each has simple functions, a cost-effective robot group can be built without losing the capability of different tasks handling.

Figure 1.4[3] shows three mobile robots moving an object cooperatively.

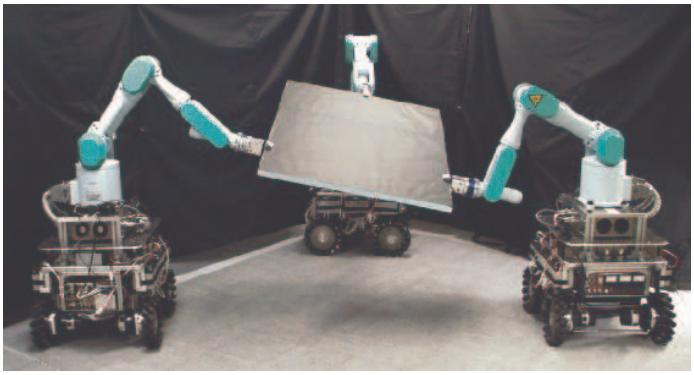


Figure 1.4: Three robots move an object cooperatively.

Formation control was inspired by the emergent self-organization observed in nature, like birds flocking and fish schooling, see Figure 1.5[4]. Each animal in a herd benefits by maximizing the chance of detecting predators or food and minimizing its encounters with predators. Teams of UAVs moving in formations with precisely defined geometries lead to many advantages, such as energy saving when the vortex forces are taken into account, see

[3] http://db.tohoku.ac.jp/whois/photo_resact/000000000841.tmp.d.jpg

[4] http://alanwhite.info/images/Birds_in_formation.JPG

Figure 1.6 [5]. Several experimental studies have verified the energy saved when flying in close formations [3]. In addition, formation control allows for intelligent *leaders* and single agent planning while *followers* can focus on other tasks. Leader-following is a common approach to build formations of multi-vehicle systems. The challenge here lies in designing a formation controller that is computationally simple, yet robust.

Figure 1.5: Birds flock.

Figure 1.6: Formation flight - Thunderbirds.

A nonholonomic model (*e.g.*, unicycle) is commonly adopted to describe vehicle's kine-

[5] `http://www.sky-flash.com`

matics in mobile robot motion coordination. Therefore, fundamental control problems, trajectory tracking and point stabilization of nonholonomic mobile robots, are inevitably encountered. During the past decades, those problems have received a lot of attention. For interested readers, the book [4] is a good starting point for understanding nonholonomic systems. Though numerous control algorithms can be found in the existing literature, the controller design is still challenging. For trajectory tracking, most of the controllers will fail when face a trajectory moving backward. However, a backward trajectory is common in leader-following formation when the leader tries to avoid some obstacles.

For point stabilization, according to Brockett's theorem [5], a smooth time-invariant feedback control law does not exist. Therefore, the controller design is more challenging. Most of the control design for point stabilization problem can be classified into two categories: (i) Discontinuous feedback laws, and (ii) time-varying algorithms. However, time-varying controllers are reported with slow convergence rates and the discontinuous controller design is complex. In addition, most of the existing approaches do not consider input constraints.

Since general cooperative objectives can be formulated into optimal control problems, optimization-based techniques are suited to multi-robot cooperative control. Model predictive control (MPC) or receding horizon control (RHC) in particular is an optimization-based approach. It has gained more and more attention in the control community. In addition, the inherent ability of MPC to handle constrained systems makes it a promising technique for cooperative control, especially for multi-vehicle formation control. With recent substantial developments in computing and solver techniques, real-time model predictive control of fast updating system can be found in literature.

1.2 Objective

The objective of this dissertation is to use model predictive control (MPC) to coordinate the motion of nonholonomic mobile robots. Specifically, we consider the formation control of a group of vehicles (UAVs/UGVs) and the trajectory tracking and point stabilization of nonholonomic mobile robots.

1.3 Statement of Contributions

The contributions of this dissertation can be summarized as follows.

- A robust formation controller is developed for the leader-following formation of UAVs. With the assumption that an autopilot operating in *holding mode* at the low-layer, we present a two-layered hierarchical control scheme which allows a team of UAVs to perform complex navigation tasks under limited inter-vehicle communication. Specifically, the robust control law eliminates the requirement of leader's velocity and acceleration information, which reduces the communication overhead (Published [6]).

- A dual-mode MPC algorithm that allows a team of mobile robots to navigate in formations is developed. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed (Published [7]).

- A *first-state contractive* model predictive control (FSC-MPC) algorithm is developed for the trajectory tracking and point stabilization problems of nonholonomic mobile robots. The stability of the proposed MPC scheme is guaranteed by adding a first-state contractive constraint and the controller is exponentially stable. The convergence is faster and no terminal region calculation is required. Tracking a trajectory

moving backward is no longer a problem under this MPC controller. Moreover, the proposed MPC controller has simultaneous tracking and point stabilization capability (Submitted [8]).

1.4 Dissertation Outline

The dissertation begins in Chapter 2 with a brief introduction of nonholonomic mobile robots. The kinematic model is developed and the Brockett theorem is introduced in Section 2.1.1 and 2.1.2. Then a short literature on nonholonomic mobile robots control of trajectory tracking and point stabilization is given in Section 2.1.3. Since formation control is one of the dissertation objectives, the shape and position of a formation are introduced in Section 2.2.1. Then in Section 2.2.2, different formation control approaches, such as leader-following, virtual structure, behavior based and graph theory based approaches are reviewed.

Chapter 3 introduces the model predictive control. The strategy of MPC is qualitatively depicted in Section 3.2. Though in industry, finite impulse response (FIR) or finite step response (FSR) models are used, MPC is always formulated based on state-space models in academia and literature. Generic MPC formulations that admits a state-space model, constraints and a quadratic performance index function are shown in Section 3.3. In detail, we give a discrete-time formulation for linear time-invariant systems and a continuous-time formulation for nonlinear systems. A short literature review is given in Section 3.4. Since in MPC schemes, the control sequence is obtained by solving a finite optimal control problem, the stability is not automatically guaranteed. Some discussion of the stability of MPC are given in Section 3.4. The key idea is to prove the monotonicity of the performance index function and use it as a Lyapunov function. For linear systems, if the open-loop system is stable, the monotonicity of the performance index function is easily guaranteed if one of the the weighting matrices is choosing by solving a Lyapunov equation. Generally, by

adding a terminal state equality constraint to the optimal control problem, the stability of the closed-loop system can be proven. As for nonlinear systems, the terminal state equality approach is still working theoretically. However, since the optimization control problem becomes nonlinear programming, finding the global optimum is computational expensive. Three approaches, the dual-mode, the quasi-infinite and the contractive algorithms, are introduced in the second half of Section 3.4.

In Chapter 4, we consider the formation control problem of UAVs. Lyapunov-based nonlinear controller design techniques are applied in this chapter. We start with a brief introduction of aircraft dynamics in Section 4.2. Based on the assumption that an autopilot running in the holding mode is used as a low-level controller, the lateral and longitudinal movements of a UAV can be decoupled and a simplified nonholonomic model of the aircraft is constructed. With this simplified model, the leader-following formation if defined later in this section. We proposed three nonlinear formation controllers in Section 4.3. The first one is a feedback linearization controller 4.3.1. The dynamic extension method is applied to design the controller. With this feedback linearization controller, we reduce the nonlinear error system into a linear system. Though theocratically sound, this technique is practical only under the assumption of a perfect plant model. Therefore, we propose a sliding mode controller in Section 4.3.2. The stability proof is also given in this section. Since a generic sliding vector function is used in the control design, a variety of available sliding vector functions can be substituted in to reduce the chattering and to achieve satisfactory performance. In Section 4.3.3, we propose a robust formation controller. Compare to the first two controllers, which assume full knowledge of leader aircraft's states, this robust controller eliminates the requirement of leader's velocity and acceleration information. Therefore, communication overhead can be reduced. A detailed stability proof can be found in this section. Finally, the performance of the proposed controllers is examined by simulations in Section 4.4.

The formation control problem is addressed again in the context of MPC in Chapter 5.

In Section 5.1, formation control by MPC literature is reviewed. We propose that it is more convenient to put the vehicles's nonholonomic constraints inside the MPC framework. In Section 5.2, we redefine the formation with graph theory. Then a dual-mode MPC formation controller is propose in Section 5.3. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. A detailed proof is given in Section 5.4. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed. For the choice of stabilizing terminal controller, a comparison between an input-output feedback linearization controller and a robust formation controller is given in Section 5.4.2 and 5.4.3. Finally, simulation results are presented in Section 5.5 and concluding remarks are given in Section 5.6.

In Chapter 6, we consider using MPC to control nonholonomic mobil robots. Since a nonholonomic model is commonly adopted to describe vehicle's kinematics in mobile robot motion coordination, fundamental control problems, trajectory tracking and point stabilization of nonholonomic mobile robots need further investigation. In the literature, most of the existing controllers address the trajectory tracking problem by assuming that the trajectory needed to be track is continuously excited by a reference robot. When the reference robot stops or even moves backward, most of the controller will fail. As for the point stabilization problem, discontinuous feedback controllers and time-varying algorithms are mostly found since by Brockett's theorem, smooth time-invariant feedback control laws do not exist. In addition, only a few controllers can handle the tracking and stabilization problems in the same control structure. In this chapter, we proposed a novel MPC approach for the control of nonholonomic mobile robots. Most stabilizing MPC methods address stability by adding terminal state penalties in the performance index and/or imposing constraints on the terminal state at the end of the prediction horizon. In the MPC algorithm we proposed, its stability is guarantees by adding a contractive constraint on the first state at the beginning of the prediction horizon. With this first-state contractive constraint, the pro-

posed MPC algorithm achieves: (i) Exponential stability, (ii) the ability to track a trajectory moving backward, and (iii) the capability of simultaneous tracking and point stabilization.

Chapter 6 is organized as follows. In Section 6.2, we introduce the robot kinematic model and the trajectory tracking and point stabilization problems of a nonholonomic mobile robot. A *first-state contractive* MPC algorithm is proposed in Section 6.3. Stability results of the proposed algorithm can be found in Section 6.4. In Section 6.5, simulation results are presented to show the effectiveness of our method. Finally, a summary is given in Section 6.6.

In Chapter 7, the FSC-MPC algorithm is extended to multi-robot formations. With the capability of simultaneous tracking and point stabilization, the FSC-MPC controller can achieve some practical formations without any special treatments. Simulation results are provided.

Finally, in Chapter 8, we summarize the main results of this dissertation and outline the future work.

CHAPTER 2

Nonholonomic Mobile Robots and Formations

2.1 Nonholonomic Mobile Robots

A nonholonomic mobile robot is a robot can only move in the direction normal to the axis of the driving wheels. Most of ground mobile robots (wheeled), part of aerial and underwater vehicles which move under some specific conditions can be considered as nonholonomic mobile robots.

Nonholonomic mobile robots have received a lot of attention in the past decades because of their ability to work in large application domains, such as: (i) Transportation, (ii) planetary exploration, (iii) surveillance, (iv) security inspection, (v) military targets tracking and attack, and (vi) human-machine-interfaces for people with mobility deficiency, to mention a few.

Due to the wide range of applications, the research of nonholonomic mobile robots is multidisciplinary and has many directions. As the dissertation objective stated in Section 1.2, we only consider the formation control of a group of vehicles and the trajectory tracking and point stabilization of nonholonomic mobile robots.

2.1.1 Kinematic Model

The kinematic model for a wheeled mobile robot (assumed to be of the unicycle type) under the nonholonomic constraint of *pure rolling and non-slipping* is considered throughout this dissertation

$$\dot{q} = S(q)u, \tag{2.1}$$

where $q(t), \dot{q}(t) \in \mathbb{R}^3$ are defined as

$$\begin{aligned} q &= [x\ y\ \theta], \\ \dot{q} &= [\dot{x}\ \dot{y}\ \dot{\theta}]. \end{aligned}$$

$x(t)$ and $y(t)$ are the position of the center of mass of the wheeled mobile robot in a Cartesian coordinate frame. $\theta(t) \in \mathbb{R}^1$ denote the orientation of the robot (see Figure 2.1). $\dot{x}(t)$ and $\dot{y}(t)$ are the Cartesian components of the linear velocity $v \in \mathbb{R}^1$ and $\dot{\theta}(t) \in \mathbb{R}^1$ denotes the angular velocity.

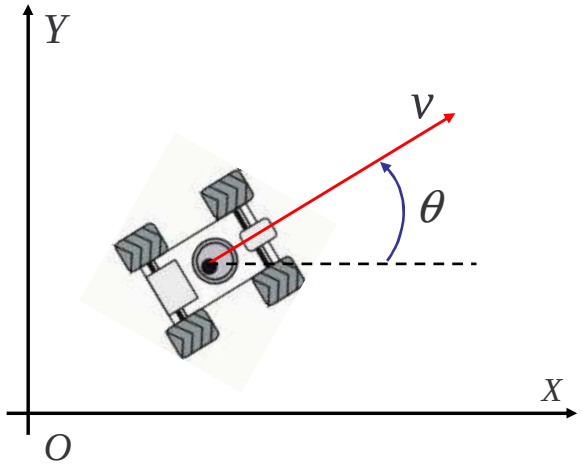


Figure 2.1: Nonholonomic wheeled mobile robot.

The matrix $S(q) \in \mathbb{R}^{3\times 2}$ is defined as follows

$$S(q) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}, \tag{2.2}$$

and the velocity vector $u(t) \in \mathbb{R}^2$ is defined as

$$u = \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{2.3}$$

where v and ω are the linear linear and angular velocity, respectively.

The *pure rolling and non-slipping* constraint means the robot can only move in the direction normal to the axis of the driving wheels. Mathematically, this constraint can be expressed as

$$\dot{x} \sin\theta - \dot{y} \cos\theta = 0. \tag{2.4}$$

A detailed analytical study of the kinematics of wheeled mobile robots (includes other types such as tricycle and car-like) can be found in [9].

For nonlinear systems, linear approximations can be the first step for analysis and control design. As we know, if the tangent linearized system is controllable, then the original nonlinear system is at least locally controllable and feedback stabilizable. By linearizing the system (2.1) about the equilibrium point ($q = 0, u = 0$), we have

$$\dot{q} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{2.5}$$

Clearly, by examining the rank of the controllability matrix

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \tag{2.6}$$

the linear system is not controllable.

For driftless nonlinear systems in the form

$$\dot{z} = \sum_{i=1}^{m} g_i(z) u_i, \tag{2.7}$$

where $z \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$, a sufficient condition (accessibility rank condition) for controllability is that, for any z, the dimension of the involutive closure of the distribution generated by the vector fields g_i is equal to n [10], that is

$$\dim \{\text{inv}\, \Delta\} = n, \ \Delta \equiv \text{span}\, \{g_i\}. \tag{2.8}$$

For the system (2.1), we have $n = 3$, vector fields

$$g_1 = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix}, \ g_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

and

$$\text{rank}\{g_1, g_2, [g_1, g_2]\} = \text{rank} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ \sin\theta & 0 & -\cos\theta \\ 0 & 1 & 0 \end{bmatrix} = 3, \tag{2.9}$$

where $[g_1, \ g_2]$ is the Lie bracket of g_1 and g_2

$$[g_1, g_2] = \frac{\partial g_2}{\partial p} g_1 - \frac{\partial g_1}{\partial p} g_2. \tag{2.10}$$

Therefore, $\dim \{\text{inv}\, \Delta\} = n$ and the system is controllable. However, for nonlinear systems, the existence of smooth time-invariant state feedback control laws cannot be implied from the controllability. This will be discussed in Section 2.1.2.

For control design, with a change of state coordinates, the model equations of the robot can be transformed into a simpler form. The following change of coordinates [11]

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ \cos\theta & \sin\theta & 0 \\ \sin\theta & -\cos\theta & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}, \tag{2.11}$$

and the change of inputs

$$\begin{aligned} u_1 &= \omega, \\ u_2 &= v - \omega x_3, \end{aligned} \tag{2.12}$$

transform the system (2.1) into

$$\begin{aligned} \dot{x}_1 &= u_1, \\ \dot{x}_2 &= u_2, \\ \dot{x}_3 &= x_2 u_1. \end{aligned} \tag{2.13}$$

This system belongs to a general class of systems called chained system [12] which has the form

$$\begin{aligned} \dot{x}_1 &= u_1, \\ \dot{x}_2 &= u_2, \\ \dot{x}_3 &= x_2 u_1, \\ &\vdots \\ \dot{x}_n &= x_{n-1} u_1. \end{aligned} \tag{2.14}$$

2.1.2 Brockett's Theorem

The problem of smooth time-invariant state feedback stabilization can be defined as follows.

Definition 2.1 *Find a state feedback $u = k(q)$, where $k(q)$ is a smooth function of q, such that the closed-loop system*

$$\dot{q} = S(q)k(q) = f(q), \tag{2.15}$$

is asymptotically stable.

However, as mentioned earlier, a controllable nonlinear system does not mean that it can be stabilized by a smooth time-invariant feedback control law. A general theorem on necessary conditions for smooth feedback stabilization of nonlinear systems is given by Brockett in [13].

Theorem 2.1 *Consider the nonlinear system* $\dot{x} = f(x,u)$ *with* $f(x_0,0) = 0$ *and* $f(\cdot,\cdot)$ *continuously differentiable in a neighborhood of* $(x_0,0)$*, necessary conditions for the existence of a continuously differentiable control law for asymptotically stabilizing* $(x_0,0)$ *are:*

1. *The linearized system has no uncontrollable modes associated with eigenvalues with positive real part,*

2. *There exists a neighborhood* N *of* $(x_0,0)$ *such that for each* $\xi \in N$ *there exists a control* $u_\xi(t)$ *defined for all* $t > 0$ *that drives the solution of* $\dot{x} = f(x,u_\xi)$ *from the point* $x = \xi$ *at* $t = 0$ *to* $x = x_0$ *at* $t = \infty$*,*

3. *The mapping* $\gamma : N \times \mathbb{R}^m \rightarrow \mathbb{R}^n$*,* N *a neighborhood of the origin, defined by* $\gamma : (x,u) \rightarrow f(x,u)$ *should be onto an open set of the origin.*

Details and the proof of this theorem can be found in [14]. In the particular case of driftless systems, a corollary to Brockett's theorem is the following [10].

Corollary 2.1 *Consider a driftless system of the form*

$$\dot{q} = \sum_{i=1}^{m} g_i(q)u_i,\ z \in \mathbb{R}^n\ u \in \mathbb{R}^m,\ m \leq n, \tag{2.16}$$

where the g_i *are smooth vector fields. If the vectors* $g_i(0)$ *are linear independent, i.e.*

$$\text{rank}[g_1(0), g_2(0), \cdots, g_m(0)] = m, \tag{2.17}$$

then a solution to the stabilization problem defined in ***Definition*** *2.1 exists if and only if* $m = n$*.*

Since **Corollary** 2.1 is not satisfied in the system (2.1) ($n = 3$, $m = 2$), stabilizing smooth time-invariant feedback laws $u = k(q)$ do not exist for the nonholonomic mobile robot.

2.1.3 Nonholonomic Mobile Robot Control

In this section, a brief literature review focusing on the trajectory tracking and point stabilization problems of nonholonomic mobile robots is given. A detailed summary of developments in control of nonholonomic systems can be found in [15].

The trajectory tracking problem focuses on stabilizing robots to a time-varying trajectory. Nonlinear feedback controllers are mostly found in the literature. Early results on local controllers can be found in [16, 10] using Lyapunov direct method. The problem is globally solved in [17] by nonlinear feedback. Dynamic feedback linearization is applied in [18, 19, 20]. Other techniques include approximate linearization [21], sliding mode control [22, 23] and backstepping [24, 25, 26]. A recursive technique for trajectory tracking of nonholonomic systems in chained form is derived from the backstepping paradigm [27]. However, a major restriction is that, the tracked linear velocity and angular velocity must not converge to zero, which means the tracked trajectory must be continuously excited. This restriction makes it impossible for a single controller to handle the tracking problem and point stabilization problem simultaneously. Consequently, the range of applications of the above mentioned controllers is limited. In addition, according to the authors in [28], the nonlinear internal dynamics of the closed-loop system under output feedback linearization controllers exhibit unstable properties when robots track a trajectory moving backward. To overcome this issue, full-state tracking techniques are explored in [29]. Model predictive controllers are reported in [30] for trajectory tracking. The MPC approach is based on the quasi-infinite algorithm proposed by authors in [31].

The point stabilization problem, which considers stabilizing robots to a final goal posture, is more challenging. As pointed out in Section 2.1.2, a smooth time-invariant feedback control law does not exist according to Brockett's theorem. Most of the control algorithms for point stabilization can be classified into three categories: (i) Smooth time-varying algorithms, (ii) discontinuous or piecewise smooth feedback laws, and (iii) middle strategies (discontinuous and time-varying). Smooth time-varying stabilization is pioneered by Sam-

son [11, 32, 33] and explored by other authors in [34, 35, 4, 36]. For discontinuous feedback controllers, see [37, 38, 10, 39, 40, 41, 42]. Middle strategies include [43]. As pointed out by authors in [40], time-varying control laws are extremely complex and only for a special class of nonholonomic systems a general strategy is available. Moreover, time-varying control laws produce very slow convergence and are intrinsically oscillating. A comparative experimental study of time-varying and discontinuous controllers for nonholonomic mobile robots is reported in [44]. Other techniques, such as dynamic feedback linearization [20], model predictive control [45], adaptive [46], neural networks [47], and backstepping [24] are also found in the literature.

Interesting results are given in [48] and [49]. With a special choice of the system state variables in polar-like coordinates, smooth feedback control laws can be globally stable. Since in those polar-like coordinates, the state is not defined at the origin, Brockett's Theorem does not hold anymore and a smooth time-invariant state feedback law for global asymptotic stability is not prevented by Brocketts negative result.

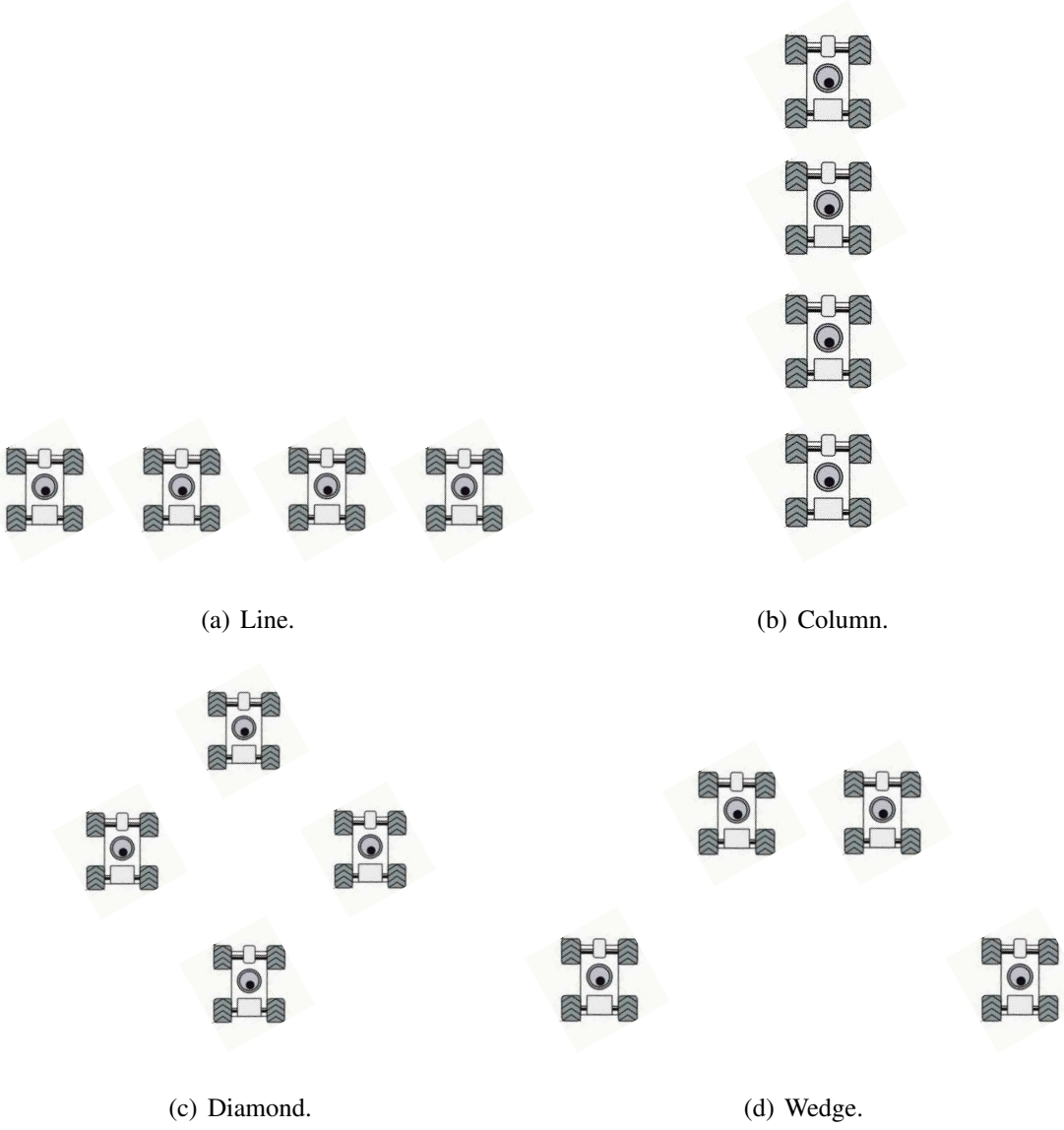
The problem of design controllers which can be used for both tracking and stabilization tasks has been explicitly addressed in [50] and [26]. Other controllers with simultaneous tracking and stabilization capabilities can be found in [51, 52, 20, 30].

Most of the controllers mentioned above only consider the kinematics of the vehicle. For the sake of handling dynamics, backstepping techniques are commonly used. The steps can be: (i) Design velocity controllers for the kinematic system (all the control algorithms mentioned above can be used), (ii) design a feedback velocity-following control law such that the robot's velocities asymptotically converge to the velocity inputs generated by the first controller, and (iii) calculate the actual toques by a computed-torque feedback controller with the second control signal as inputs. See [25], [26] for trajectory tracking, [53] for point stabilization and [24] for both.

2.2 Formations

2.2.1 Shapes and Positions

A formation describes the specific relationship among robots in a group. Shape and position are the two important characteristics of a formation. Figure 2.2 shows some of the common formation shapes in consideration, such as line, column, diamond, and wedge [54].



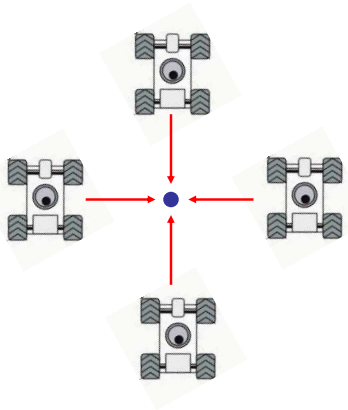
(a) Line. (b) Column.

(c) Diamond. (d) Wedge.

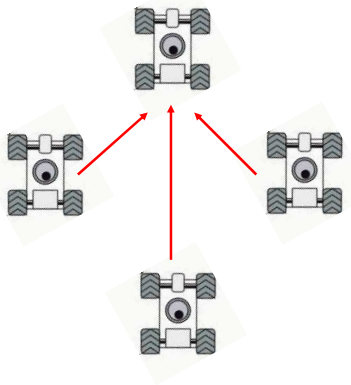
Figure 2.2: Formation shapes.

Besides the shape, each robot must have a specific position in the formation. Three

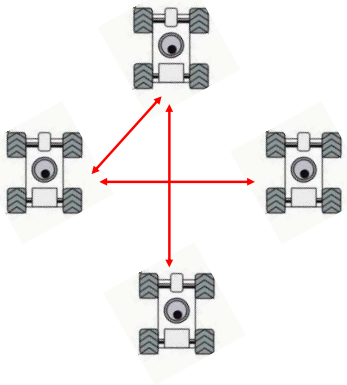
techniques for formation position determination have been identified in [54]. They are *unit-center-referenced*, *leader-referenced* and *neighbor-referenced* (see Figure 2.3). In a *unit-center-referenced* position, each robot maintains its desired relative position to a unit-center point which is the average of the x and y positions of all the robots involved in the formation. In a *leader-referenced* position, each robot (except the leader) maintains its desired relative position to a leader robot. In a *neighbor-referenced* position, each robot maintains its desired relative position with respect to one other predetermined robot.



(a) Unit-center-referenced.



(b) Leader-referenced.



(c) Neighbor-referenced.

Figure 2.3: Formation position determination.

2.2.2 Formation Control Strategies

Due to the attention received by formation control research, a lot of control strategies have been proposed. Approaches can be classified into leader-following, virtual structure, behavior based, and graph theory based.

The leader-following formation control is a important method [55, 56, 57, 58, 59, 60, 61]. In this approach, one or some mobile robots are designated as leaders which take care of the assignment of followers' relative positions and the global mission objective, such as trajectory tracking and obstacle avoidance. Simplicity is the major advantage of this approach since only the leader takes care of the global objective while the formations are guaranteed by individual robot's control law. However, when a single-leader architecture is used, the whole system will fail if the leader fails (single point of failure). In addition, the full state of the leader must be communicated to each member of the formation. Leader failure includes robot malfunction and communication error. Some attempts have been made to overcome the single point of failure. In [55], the architecture has some leader failure detection mechanisms and redefines the formation control graph according to some predefined strategies. As the number of robots increase, string stability and mesh stability need to be addressed [62, 63].

In the virtual structure approach, the entire formation is treated as a single entity. Desired states for each vehicle in the formation can be specified by place-holders in the virtual structure [64]. Similar ideas include the virtual leader [65], and the formation reference point [66]. With these approaches, it is easy to prescribe the coordinated behavior for the group. In addition, the virtual structure approach can maintain the formation during manoeuvres and a rigid geometric relationship among multiple robots. However, the requirement of the the formation to act as a virtual structure limits the class of potential application. Using virtual structure approach for mobile robots formation control is proposed by Lewis [67] and studied in [68, 69, 70, 71, 72].

The behavior based formation control approach [73, 74, 75, 76, 77, 78] is inspired by

formation behaviors in nature, like flocking and schooling. In [79], the author simulates flocks of birds and schools of fish with a simple egocentric behavioral model. In [54], the authors define two steps for formation maintenance: (i) Detect-formation-position which determines the robot's position in formation, and (ii) maintain-formation which generates motor commands to direct the robot toward the correct location. In behavioral approaches, the control action for each robot is defined by a weighted average of the control corresponding to each desired behavior. Possible behaviors can be, collision avoidance, obstacle avoidance, goal seeking and formation keeping. The advantage of the behavioral approach is that, when robots have multiple competing objectives, it is easy to derive control laws in a natural way. However, even this approach performs well in simulations and experiments, it is hard to do mathematical analysis of stability and robustness.

Recently, graph theory is applied to multiple robot formation control [80, 81, 82, 83, 84, 85]. Formations can be described by graphs treating each robot as a vertex and the relationships (control, sensing, communication-flow) among robots as edges. The communication links among systems are described by Laplacian matrices. The stability of the multi-robot system is guaranteed by the stability of each robot. However, the method is limited to linear robot models. In [86], the authors propose two controllers for nonholonomic mobile robots formation using graph theoretical methods. In the first controller, the robot model is transformed to a linear model by dynamic feedback linearization. The second controller, with the aid of the time-scaling technique and the properties of Laplacian matrix, overcomes the singularity of the first controller.

2.3 Summary

In this chapter, a brief review of nonholonomic mobile robots is given. Then the kinematic model is developed and the Brockett's theorem is introduced. In addition, a short literature review on nonholonomic mobile robots control of trajectory tracking and point stabiliza-

tion is given. Since formation control is one of the dissertation objectives, the shape and position of a formation are introduced. Different formation control algorithms, such as leader-following, virtual structure, behavior based and graph theory based approaches are reviewed.

CHAPTER 3

Model Predictive Control

3.1 Introduction

In the past decades, model predictive control (MPC), also known as receding horizon control (RHC), has received great interest in the control community. As an effective method to solve multi-variable constrained control problems, MPC has appeared in industry for more than 20 years and successful applications of MPC can be easily found.

The glorious past and present of MPC is due to its abilities of constraint handling, real-time prediction, optimizing and feedback correcting. Different from conventional control which uses pre-computed control laws, MPC is a control of the form in which the current control action is obtained by solving a finite horizon optimal control problem online at each sampling instant. The optimization yields an open loop optimal control sequence and the first control of this sequence is applied to the plant. The whole process will repeat at the following sampling instants.

Actually, the term Model Predictive Control does not designate a specific control strategy but a large range of control methods which explicitly use a plant model to calculate the control action by minimizing a finite horizon optimal control problem. The ideas appearing in all the predictive control family are: (i) A model of the plant which is used to predict the future response of the system at future time instants (horizon), (ii) the calculation of a

sequence of control action minimizing an optimal control problem using system's current states as the initial condition, and (iii) the receding strategy which involves the application of the first control action at each step, so that the horizon moves towards the future at each instant. Practically, this combination of feedforward and feedback makes MPC to outperform "passive" feedback control.

As a summary, the advantages of MPC can be [87]: "**Generality** - the ability to include generic models (linear or nonlinear) and constraints in the optimal control problem; **Reconfigurability** - the ability to redefine cost functions and constraints as needed to reflect changes in the system and/or environment".

Though MPC has gained great success in process industries, its applications for fast updating systems are dragged by the computational burden. Since an optimal control problem must be solved online, the sampling period needs to be long enough for the calculation. In process industries, usually plants under control are sufficiently 'slow' and can be satisfactorily linearized, the computational burden is not that critical. However, in highly nonlinear systems, an optimal solution may not be able to determined within a short sampling period. Sometimes, even a feasible solution may not be possible. Another issue is that, the stability of MPC algorithms is not automatically guaranteed since the control sequence is obtained from solving a finite optimal control problem. The implicit nature of the closed-loop system makes the proofs of stability of MPC a complicated job.

3.2 MPC Strategy

The strategy of MPC is depicted in Figure 3.1. This figure shows a generic MPC algorithm for a single-input-single-output (SISO) system.

At current time, say k, the system's future response (predicted output) $y_p(k)$ on a finite horizon N_p, say $[k \quad k+N_p]$, is predicted by the system model and the predicted control input $u_p(k)$, $[k \quad k+N_m]$. N_p is named as the prediction horizon and N_m is named as

the control horizon ($N_m \leq N_p$). Usually, the system's future response is expected to return to a desired set point $s(k)$ by following a reference trajectory $r(k)$ from the current states. The difference between the predicted output $y_p(k)$ and the reference trajectory $r(k)$ is called predicted error. An finite horizon optimal control problem with a performance index that usually penalizes the predicted control input and the predicted error is solved online and an optimal control input $u^*(k)$, $[k \quad k + N_m]$, which minimizes the predicted error is obtained. Only the first element of $u^*(k)$ is implemented to the plant. All the other elements are discarded. Then, at the next time interval $k + 1$, the whole procedure is repeated. The predicted control input $u_p(k + 1)$ at time $k + 1$ can be built by $u^*(k)$ with linear extrapolation. Since the prediction horizon and control horizon move one step further into future at each time interval, MPC is also named as *receding horizon control* (RHC).

In order to implement this strategy, the basic structure shown in Figure 3.2 is used.

3.3 MPC Formulation

Though in industry, finite impulse response (FIR) or finite step response (FSR) models are used, MPC is always formulated based on state-space models in academia and literature.

Consider the following discrete-time linear time-invariant system:

$$x(k+1) = Ax(k) + Bu(k), \tag{3.1}$$

where $x(k) \in \mathbb{R}^n$ and $u(k) \in \mathbb{R}^m$ are the state and control input, respectively. $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are the state and input matrices, respectively. The MPC implementation can be formulated by introducing the following open-loop optimization problem at every time interval k:

$$\min_{u(\cdot)} J_{(N_p,N_m)}(x_k), \tag{3.2}$$

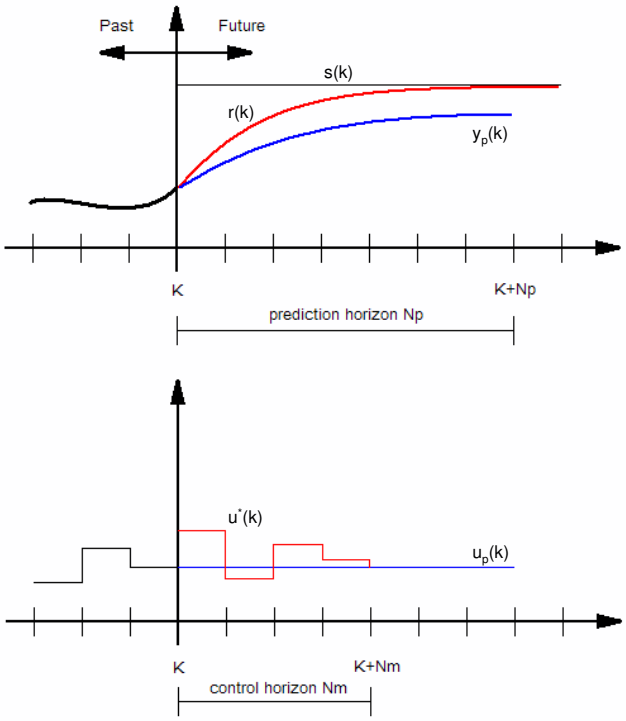


Figure 3.1: MPC concept.

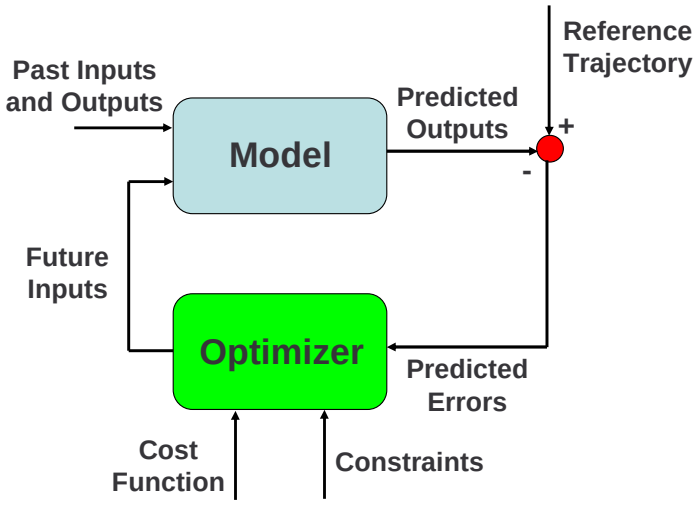


Figure 3.2: Basic structure of MPC.

subject to

$$
\begin{aligned}
& x(k+1) = Ax(k) + Bu(k), \\
& x_{min} \leq x(k+i) \leq x_{max}, i = 1, 2, \cdots, N_p, \\
& u_{min} \leq u(k+i) \leq u_{max}, i = 0, 1, 2, \cdots, N_m.
\end{aligned} \tag{3.3}
$$

The performance index is defined as

$$
\begin{aligned}
J_{(N_p,N_m)}(x_k) &= x^T(N_p)Px(N_p) + \sum_{i=1}^{N_p-1} x^T(k+i)Qx(k+i) \\
&\quad + \sum_{i=0}^{N_m} u^T(k+i)Ru(k+i),
\end{aligned} \tag{3.4}
$$

where $P \in \mathbb{R}^{n\times n}$, $Q \in \mathbb{R}^{n\times n}$, and $R \in \mathbb{R}^{m\times m}$ are the weighting matrices, and $P = P^T > 0$, $Q = Q^T > 0$, $R = R^T > 0$. N_p and N_m denote the length of the prediction horizon and the length of the control horizon, respectively. Usually, $N_p \geq N_m$. The first term on the right hand side of equation (3.4) is called terminal state penalty, the second term is called state penalty and the third term is called control penalty.

Equations (3.1)-(3.3) define a quadratic program problem and many algorithms and software packets are available to solve it. When the optimal control sequence $u^*_{(N_p,N_m)}(i \mid x(k))$, $i = 0, \cdots, N_m 1$ is obtained, only the first control $u^*_{(N_p,N_m)}(0 \mid x(k))$ is applied to the system. The rest of the control sequence is discarded. Then at the next time interval $k+1$, $x(k+1)$ is used as the new initial condition of the optimal problem (Equation (3.2) and the process is repeated.

As for nonlinear systems, the concept of MPC remains the same. Consider the following continuous-time nonlinear system,

$$
\dot{x}(t) = f(x(t), u(t)), \tag{3.5}
$$

where $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ are the state and control input, respectively.

Similar to the linear case, the optimization problem can be defined as:

$$
\min_{u(\cdot)} J_T(x(t), u(\cdot)), \tag{3.6}
$$

subject to

$$\begin{aligned}
&\dot{x}(t) = f(x(t), u(t)),\\
&x_{min} \leq x(s; x(t), t) \leq x_{max}, t \leq s \leq t+T,\\
&u_{min} \leq u(s) \leq u_{max}, t \leq s \leq t+T.
\end{aligned} \tag{3.7}$$

The performance index is defined as:

$$\begin{aligned}
J_T(x(t), u(\cdot)) \;\; = \;\; & \int_t^{t+T} (\|\bar{x}(s; x(t), t)\|_Q^2 + \|u(s)\|_R^2) ds \\
& + \|\bar{x}(t+T; x(t), t)\|_P^2.
\end{aligned} \tag{3.8}$$

In this case, the prediction horizon and the control horizon are the same and equal to T.

The generic MPC algorithm can be described as follows,

1. At current time t or k, measure the current state $x(t)$ or $x(k)$ as the initial condition,
2. Solve the finite optimization control problem (3.2) or (3.6) with the initial condition obtained in 1, yielding the optimal control sequence u^* over the control horizon
3. Apply the first element in control sequence u^* to the system, the remaining elements of the control sequence is discarded,
4. At time $t + \delta_t$ or $k + 1$, repeat from 1.

Note that, δ_t is the sampling time and $k = (t - t_0)/\delta_t$.

3.4 Literature Review

Since the objective of this dissertation is to use model predictive control to coordinate the motion of nonholonomic mobile robots, a review of theoretical results in MPC is now given. Two thorough survey papers [88], [89], which give good reviews on MPC's past, present and future, stability and optimality, are an excellent starting point for any interested reader in this area.

The history of model predictive control is quite different than other control design tools. This technique has its origin from industry before any theoretical foundation. The development of MPC can be traced to the work of Kalman in the early 1960s, which is known as the *Linear Quadratic Gaussian* (LQG). However, at that time, the industrial process control community either had no exposure to LQG technique or regarded it as impractical. The LQG failed to have a strong impact. This environment led to a more general model based control methodology developing in industry, in which the dynamic optimization problem is solved online at each control execution.

The first description of MPC appeared in 1976 [90] and later summarized in [91]. The authors called their approach model predictive heuristic control (MPHC), but the solution software is usually mentioned as IDCOM, an acronym for IDentification and COMmand. The main features of the IDCOM approach are: impulse response model, quadratic performance objective, reference trajectory for future plant output behavior, including input, and output constraints and heuristic iterative optimization algorithm.

Engineers at Shell Oil developed their own MPC technology and an unconstrained multi-variable control algorithm which they named dynamic matrix control (DMC) was presented in [92] and [93]. The main features of the DMC approach can be summarized as follows: linear step response model, quadratic performance objective, setpoint for future plant output behavior, and least-squares optimization algorithm. Later, the DMC algorithm was posing into a quadratic program (QP) in which input and output constraints appear explicitly. This modified DMC algorithm is called quadratic DMC (QDMC) [94, 95].

After the publication of papers addressing IDCOM and DMC/QDMC, interest in this filed starts to surge and new algorithms have been developed. Today, MPC applications have made this machinery a multi-million dollar industry. A survey of commercially available MPC technology can be found in [96]. Figure 3.3 [96] shows the approximate genealogy of industrial MPC algorithms.

Although the model predictive control formulation seems quite intuitive, the stability is

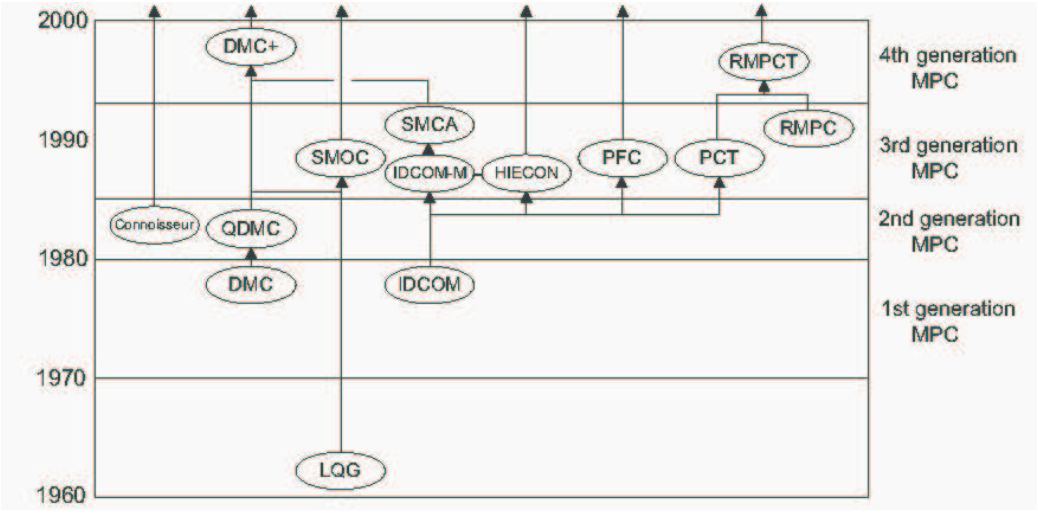


Figure 3.3: Approximate genealogy of industrial MPC algorithms.

not automatically guaranteed since the control sequence is obtained from a finite optimal control problem. Without the fine tuning of weighting matrices, the MPC algorithm formulated in (3.2), (3.6) may lead to divergent responses. Therefore, it is not surprising that much effort has been devoted to determine sufficient conditions for stability.

As a powerful analysis tool, Lyapunov methods are frequently encountered in MPC literature. Pointed out by the authors in [89], the performance index function is monotonic and it can be used as a Lyapunov function. The stability analysis of MPC has reached a relatively mature stage. A short summary is given in this section. Interested readers are referred to [89], [88] for excellent reviews of the stability properties of MPC.

As for the linear system, proofs of stability based on the monotonicity property of the performance index function have been proposed in [97], [98].

To simplify the notation, the prediction horizon and the control horizon are assumed to be equal $N_p = N_m = N$. Then we use J_N to replace $J_{(N_p,N_m)} = J_N$ which is defined in Equation (3.4).

The key idea of the monotonicity approach is using the performance index function J_N as a Lyapunov function. This means the following inequality of the index function needs

to be shown

$$J_N(x(k)) - J_N(x(k+1)) \geq 0 \quad \text{for} \quad x \neq 0.$$

Rewriting $J_N(x(k)) - J_N(x(k+1))$ gives:

$$\begin{aligned} J_N(x(k)) - J_N(x(k+1)) &= [x^T(k)Qx(k) + u_N^{*T}(x(k))Ru_N^*(x(k))] \\ &\quad +[J_{N-1}(x(k+1)) - J_N(x(k+1)]. \end{aligned} \tag{3.9}$$

With the assumption that $Q > 0$ and $R > 0$, the first term on the right hand side of Equation (3.9) is positive. However, in general, it is hard to assert whether the second term is nonnegative.

Several approaches have been proposed to assure the constantly decrease of the performance index J_N.

In most cases, if the open loop system is stable, by choosing the weighting matrix P as the solution of the Lyapunov equation [99]

$$A^T P A + Q = P, \tag{3.10}$$

J_N is non-increasing and the stability can be guaranteed.

In [100], the authors prove that when a terminal state equality constraint $x(k+N) = 0$ is imposed, the performance index J_N is non-increasing as a function of N. Then stability follows.

Another approach is to add a terminal constraint that forces the terminal state to be inside a positively invariant region. The decreasing property of J_N can be achieved by introducing a stabilizing local controller $u(k+i) = Lx(k+i)$ for $i > N$. In this case, the terminal penalty and the positively invariant region need to be defined with respect to the system $x(i+1) = (A+BL)x(i)$ rather than $x(i+1) = Ax(i)$. Furthermore, the positive invariance should be defined with the respect to the input and state constraints. The local feedback controller can be chosen by the infinite horizon unconstrained LQR method [101].

As for the nonlinear system, approaches introduced above are the idea underlying nonlinear MPC. With a nonlinear system model used, the optimization control problem to be solved on-line becomes nonlinear programming.

In [97] and [102], the idea of zero terminal constraints for nonlinear MPC is analyzed. The performance index function is employed as a Lyapunov function. To guarantee stability, a global optimum must be found at each time step. Though theoretically, the optimization problem with terminal equality constraint can be solved, the computation for finding the global optimum is very expensive. Even when a feasible solution exists, the convergence to that solution is not guaranteed.

A dual-mode MPC algorithm is proposed by authors in [103] to deal with both the global optimality and the feasibility problems. A terminal region is introduced to relax the terminal equality constraint. At the end of the prediction horizon, the terminal region must be reached. Inside this region, an asymptotically stabilizing controller is employed. With these modifications, a global optimum is no longer required. A feasible solution at the initial time will guarantee the feasibility at all future time steps. However, the terminal region is hard to calculate except for low order systems.

An algorithm called quasi-infinite MPC proposed by authors in [31] can overcome both the global optimization and the feasibility problem without using controller switching. In this method, the performance index function is formulated as

$$J_T(x(t), u(\cdot)) = \int_t^{t+T} (\|\bar{x}(s; x(t), t)\|_Q^2 + \|u(s)\|_R^2) ds + \|\bar{x}(t+T; x(t), t)\|_P^2.$$

The open-loop optimal control problem besomes

$$\min_{u(\cdot)} J_T(x(t), u(\cdot)),$$

subject to

$$\bar{x}(t+T; x(t), t) \in \Omega. \tag{3.11}$$

A weight matrix P needs to be determined such that the penalty of the terminal state $\bar{x}(t+T)$, which is the second term on the right hand side of the performance index, is bounded

by the infinite horizon cost after $t+T$

$$\|\bar{x}(t+T;x(t),t)\|_P^2 \leq \int_{t+T}^{\infty}(\|\bar{x}(s;x(t),t)\|_Q^2 + \|u(s)\|_R^2)ds$$
$$\forall \bar{x}(t+T;x(t),t) \in \Omega \tag{3.12}$$

The bound is established by assuming that the nonlinear system is controlled by a linear optimal state feedback controller within the region Ω after $t+T$. Again, a feasible control sequence solution at time t means feasible solutions in the future and stability of the closed-loop system is guaranteed. However, the difficulty of terminal region calculation is not improved by this quasi-infinite MPC method.

A contractive MPC idea is proposed in [104] and completed and proven in [105]. A constraint is added to the MPC formulation which forces both the actual and predicted state to contract. With this requirement, the stability can be proven.

All the methods introduced above need to solve nonlinear programming problems at each time step. Compare to the linear case, the computational requirement is huge. Intuitively, we could linearize the system. Then with the linearized system, all the methods developed for linear systems can be employed. This kind of approach can be found in [106], [107], and [108].

3.5 Summary

In this chapter, a brief introduction of MPC is given. MPC formulations for linear systems and nonlinear systems are briefly discussed in discrete-time and continues-time. Since a finite horizon optimal control problem is solved inside the MPC algorithm, the control law is not guaranteed to be stable. Different stabilizing MPC methods are introduced for linear and nonlinear systems.

CHAPTER 4

Nonlinear Formation Control of Unmanned Aerial Vehicles

In this chapter, we consider the problem of designing nonlinear formation controllers on a team of unmanned aerial vehicles (UAV) using off-the-shelf autopilots. Three nonlinear formation controllers are presented. The first two controllers require knowledge of the leader's velocity and acceleration. The third controller, on the other hand, does not have such requirements. Under these controllers, the formation of UAVs is proven to be stable. Simulations validate the performance of the proposed controllers.

4.1 Introduction

Due to recent developments in electronics and computing, it is now possible to find small size, light weight, powerful embedded computers, wireless network equipments and sensing devices on the market. As a result, by exploiting current technology, one can build a group of relatively small UAVs each having satisfactory capabilities within a reasonable budget. For tasks such as obtaining sensory measurements over a wide area (*e.g.*, forest fire monitoring [109]), multiple UAVs are desirable because they can accomplish the task more efficiently than a single UAV. Interested readers are referred to [110] where a survey

of UAV cooperative control is provided.

Teams of UAVs moving in formations with precisely defined geometries lead to many advantages, such as energy saving when the vortex forces are taken into account. Several experimental studies have verified the energy saved when flying in close formations [3]. In addition, formation control allows for intelligent *leaders* and single agent planning while *followers* can focus on other tasks. Leader-following is a common approach to build formations of multi-vehicle systems. The challenge here lies in designing a formation controller that is computationally simple, yet robust. In [111], a leader-following approach for formation flight is designed using input/output feedback linearization techniques. Furthermore, in [112] a framework for controlling a group of UAVs is developed. The controller design utilizes input/output *dynamic* linearization techniques based on a model that included the induced rolling moment generated by the lead aircraft over the wing aircraft. In [113], formation controllers are designed to maintain an optimal longitudinal separation needed to achieve the maximal reduction in the induced drag. Authors in [114] develop an interesting experimental testbed to investigate close formation flight.

In this chapter, based on a cost effective autopilot[1], we propose a two-layer control architecture for practical and robust formation control. In this control scenario, the autopilot provides stable velocities and height tracking on the lower-level during the mission flight. On the higher-level, nonlinear formation controllers ensure that leader and follower UAVs are in formations with desired relative distances and bearing angles.

The remaining of the chapter is organized as follows. Section 4.2 gives a brief introduction of aircraft dynamics and the leader-following formation. In Section 4.3, we present three different nonlinear formation controllers. Stability results are also provided in this section. Simulation results are presented in Section 4.4. Finally, concluding remarks are given in Section 4.5.

[1] A product of Cloud Cap Technology, Inc.

4.2 Preliminaries

Clearly, unmanned aerial vehicles are aircrafts without pilots. They are either remotely controlled or capable of conducting autonomous operations. Like manned aircrafts, UAVs can be classified by their sizes, types, methods of propulsion and their missions. They may be fixed-wing aircrafts or helicopters.

To control UAVs, we need to know the position and velocity of the aircraft in the air. This leads to the study of aircraft dynamics. This section gives a brief introduction of aircraft dynamics and the setup of leader-following formation.

4.2.1 Aircraft Dynamics

Aircraft motion may only "make sense" when it is represented in some coordinate systems. Therefore, it is necessary to define appropriate coordinate systems. In this report, all the coordinate systems are right-handed and orthogonal.

The earth-fixed reference frame, F_E: This coordinate system is defined like this: the origin is fixed to an arbitrary point on the surface of the Earth. The x_E axis points to North, the y_E axis points to East. Consequently the z_E axis points to the center of the Earth. Figure 4.1 shows this coordinate system.

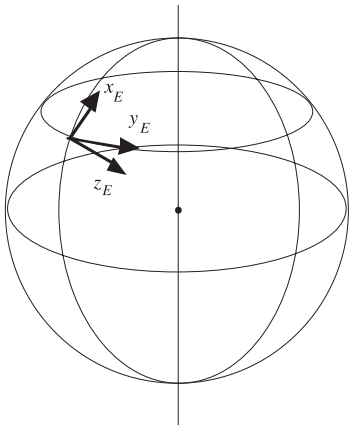


Figure 4.1: Earth-fixed reference frame.

The body-fixed reference frame, F_B: In this report, we use the following definition as the body-fixed reference frame. First, we assume that the aircraft has a plane of symmetry. Then the x_B and z_B axes lie in that plane of symmetry. The origin is fixed to aircraft's center of gravity. The x_B axis points to the head of the aircraft, the z_B axis lies in the plane of symmetry and points downward. The y_B axis is determined by the right-handed rule.

The stability-axis reference frame, F_S: We consider the aircraft in steady flight condition so that the relative wind is from a constant direction as seen from the aircraft. The velocity vector V_c of the aircraft is relative to the air mass. Then the projection of this velocity vector into the aircraft plane of symmetry is defined as x_S axis. The origin is fixed to aircraft's center of gravity. The z_S axis lies in the aircraft plane of symmetry and points downward. The y_B axis is then determined by the right-handed rule.

The wind-axis reference frame, F_W: The wind-axis system is defined as follows. The origin is fixed to aircraft's center of gravity. The x_W axis is in the direction of the velocity vector of the aircraft relative to the air. The z_W axis lies in the aircraft plane of symmetry and points downward. The y_W axis is determined by the right-handed rule. Note that x_W axis needs not lie in the plane of symmetry.

The body-fixed frame, the stability-axis frame and the wind-axis frame are related by two aerodynamic angles. The angle between the x_W axis and the x_S axis is called sidelip and is denoted by symbol β. The angle between the x_S axis and the x_B axis is called angle-of-attack and is denoted by symbol α. Figure 4.2 shows the three different frames and the two angles.

A point in the space can be interpreted differently with the respect to different coordinate systems. Pointed out by the Swiss mathematician *Leonhard Euler*, one reference frame can be placed in alignment with any other reference frame by three successive axis-rotations. The angles to rotate the axes are called *Euler angles*. With *Euler angles*, the transformation between different coordinate systems is represented by the transformation matrices and the coordinates of a point in different reference frames can be related. In

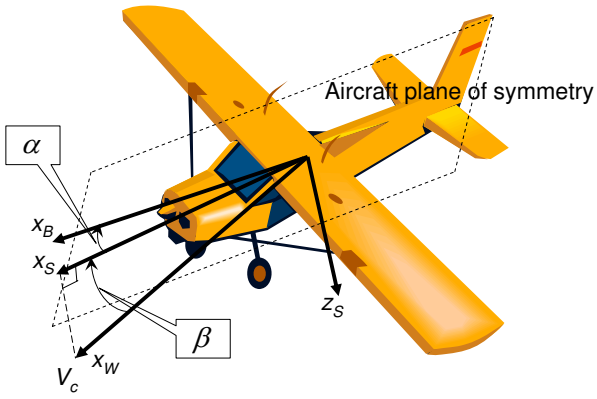


Figure 4.2: Reference frames and aerodynamic angles.

aircraft dynamics, the *Euler angles* between different reference frames are given special symbols. They are summarized as follows:

T_{F_2,F_1}	θ_x	θ_y	θ_z
$T_{B,E}$	ϕ	θ	ψ
$T_{W,E}$	μ	γ	χ
$T_{B,W}$	0	α	$-\beta$

Table 4.1: Euler angles.

The T_{F_2,F_1} notation means transformation from reference frame 1 to reference frame 2. For example, $T_{W,E}$ means transformation from the earth-fixed reference frame to the wind-axis reference frame.

With the coordinate system established, one can deduce aircrafts' equations of motion by applying Newton's law (see *e.g.*, [115, 116]). We simplify our analysis to an ideal scenario that the UAVs are flying in a wings-level steady-state and that the angle of attack and sideslip are considerably small, such that they can be ignored. With such simplification,

a 6-DOF nonlinear model can be set up as follows:

$$\begin{aligned}
\dot{x} &= V\cos\gamma\cos\chi, \\
\dot{y} &= V\cos\gamma\sin\chi, \\
\dot{z} &= -V\sin\gamma, \\
\dot{V} &= -\tfrac{D}{m} - g\sin\gamma, \\
\dot{\mu} &= p + (q\sin\mu + r\cos\mu)\tan\gamma, \\
\dot{\gamma} &= q\cos\mu - r\sin\mu, \\
\dot{\chi} &= (q\sin\mu + r\cos\mu)\sec\gamma, \\
\dot{\omega} &= J^{-1}\hat{\omega}J\omega + J^{-1}\tau + \bar{L},
\end{aligned} \tag{4.1}$$

where x, y, z are position states in the flat earth-fixed initial frame; yaw angle χ, pitch angle γ and roll angle μ are attitude states in the wind-axis frame; roll rate p, pitch rate q and yaw rate r are angular velocity states in the body-fixed frame; V is the linear velocity along the flying path; $\omega = [p \quad q \quad r]^T$; J is the inertia matrix; $\hat{\omega}$ is a skew-symmetric operator; τ is the external moment vector; g is the acceleration of gravity; m is the mass of the UAV; D is the drag and $\bar{L} = [\bar{L}_p \quad 0 \quad 0]^T$ is the rolling moment induced by the wake of the leader aircraft.

Usually, an off-the-shelf autopilot can provide two basic operational modes: (i) Waypoint tracking mode, and (ii) holding mode. In the first mode, a set of ordered points of interest (POI) can be uploaded into the autopilot before the mission or during the flight. The autopilot generates a path from the current position along these points of interest and provides control commands to the aircraft. In this mode, however, the user cannot precisely control the aircraft's position except waypoints. In addition, the distance between two successive waypoints must be long enough such that the autopilot system can generate the flight path. As for the holding mode, usually three channels are provided as follows: (1) the Mach hold, (2) the heading turn rate hold, and (3) the altitude hold. The autopilot continuously executes control commands sent to these hold channels. Although, some successful waypoint-based formation flight experiments have been reported in the literature

(see [117]), it would be more convenient to use the autopilot in holding mode when close formation flight is required, for instance, navigation in urban environments. In this chapter, we investigate the feasibility of using an off-the-shelf autopilot [118] in holding mode for the follower UAVs.

With an autopilot running in the holding mode, the lateral and longitudinal movements are decoupled [113]. We can write a simplified model of the aircraft as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\psi & 0 \\ \sin\psi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix}, \tag{4.2}$$

$$\begin{aligned} \dot{V} &= -\frac{1}{\tau_v} V + \frac{1}{\tau_v} V_c, \\ \dot{\omega} &= -\frac{1}{\tau_\omega} \omega + \frac{1}{\tau_\omega} \omega_c, \\ \ddot{h} &= -\frac{1}{\tau_{h_a}} \dot{h} - \frac{1}{\tau_{h_b}} h_i + \frac{1}{\tau_{h_b}} h_c, \end{aligned} \tag{4.3}$$

where x and y represent the positions in Cartesian coordinates, V is the velocity, ψ is the heading angle, and h is the altitude. V_c, ω_c, and h_c are the commands to the *Mach hold*, *heading-turn-rate hold*, and *altitude hold* channels of the autopilot, respectively. τ_v, τ_ω, τ_{h_a}, and τ_{h_b} are known positive constants that depend on the autopilot.

We will only use this simplified aircraft model in this chapter.

4.2.2 Leader-Following Formation

In this section, we set up a kinematic model for an UAV formation flight system. Assuming that the UAVs are flying at a constant altitude, we can consider an operator specified the i^{th} UAV motion $a_i \in \mathbb{R}^2$, $i \in \{1, 2, ..., N\}$ given as follows

$$a_i = x_i \hat{\imath} + y_i \hat{\jmath}, \tag{4.4}$$

where x_i and $y_i \in \mathbb{R}$ represent the respective Cartesian coordinates in an earth-fixed reference inertial frame F_E. Let $F_{jk}^d \in \mathbb{R}^2$ be a desired formation that allows a wing airplane k

to follow a leader airplane j given by

$$F_{jk}^{d} = [l_{jk}^{d} \quad \eta_{jk}^{d}]^{T}, \tag{4.5}$$

where $l_{jk}^{d} \in \mathbb{R}_{+}$ is the desired relative distance and $\eta_{jk}^{d} \in \left[\frac{\pi}{2}, \frac{3\pi}{2}\right]$ is the desired relative bearing angle as shown in Figure 4.3. Consequently by changing F_{jk}^{d} we are able to define different formation shapes. Then the actual formation for a wing airplane k to follow a leader airplane j is described by

$$F_{jk} = [l_{jk} \quad \eta_{jk}]^{T} \in \mathbb{R} \times \left[\frac{\pi}{2}, \frac{3\pi}{2}\right], \tag{4.6}$$

in which the relative distance is defined as

$$l_{jk}(t) = \|a_j - a_k\|_2 , \tag{4.7}$$

where $\|\cdot\|_2$ denotes the standard Euclidean norm. The relative bearing is given by

$$\eta_{jk} = \pi + \zeta - \psi_j, \tag{4.8}$$

where $\zeta = \arctan 2(y_j - y_k, x_j - x_k)$.

Taking the time derivative of equation (4.6) with system model (4.2), we have

$$\dot{F}_{jk} = \begin{bmatrix} \dot{l}_{jk} \\ \dot{\eta}_{jk} \end{bmatrix} = \begin{bmatrix} V_k \cos\gamma_{jk} - V_j \cos\eta_{jk} \\ \frac{1}{l_{jk}}(V_j \sin\eta_{jk} - V_k \sin\gamma_{jk} - l_{jk}\omega_j) \end{bmatrix}, \tag{4.9}$$

with $\gamma_{jk} \triangleq \psi_j + \eta_{jk} - \psi_k$. V_k is the linear velocity of the follower. V_j and ω_j are the linear and angular velocities of the leader.

Our control objective is to design a formation controller which drives the wing UAV to track the desired formation F_{jk}^{d}. To this end, we define the formation error as

$$e = F_{jk}^{d} - F_{jk}, \tag{4.10}$$

where $e \in \mathbb{R}^2$.

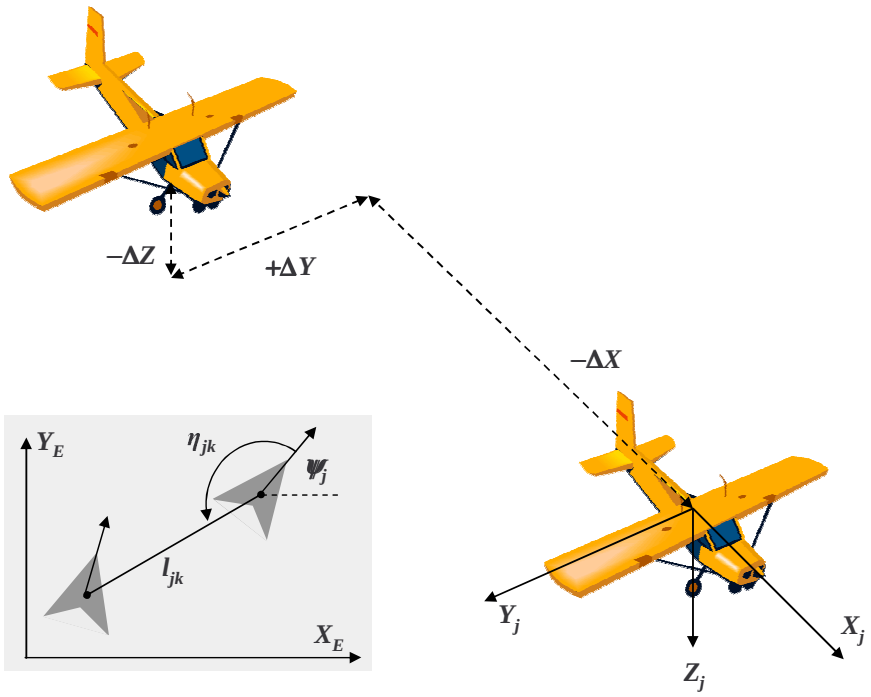


Figure 4.3: Leader-following formation.

4.3 Nonlinear Formation Control

4.3.1 Feedback Linearization

Taking the derivative of (4.10) with respect to time and combining with (4.9), the following is obtained

$$\begin{aligned} \dot{e} &= \dot{F}^d_{jk} - \dot{F}_{jk} \\ &= \dot{F}^d_{jk} - \begin{bmatrix} \cos\gamma_{jk} & 0 \\ -\frac{1}{l_{jk}}\sin\gamma_{jk} & 0 \end{bmatrix} \begin{bmatrix} V_k \\ \omega_k \end{bmatrix} - \begin{bmatrix} -V_j\cos\eta_{jk} \\ \frac{V_j}{l_{jk}}\sin\eta_{jk} - \omega_j \end{bmatrix}. \end{aligned} \tag{4.11}$$

Since ω_k does not appear, it is obvious that the input matrix of the error system (4.11) is not invertible. Thus we cannot design the control based on (4.11).

To facilitate the control design on this system, we first use the dynamic extension method from [119] to render (4.11) into a relative degree system.

Differentiating both sides of (4.11) with respect to time and after some algebraic and

trigonometric manipulation, we obtain

$$\begin{aligned}\ddot{e} &= \ddot{F}^d_{jk} - \ddot{F}_{jk} \\ &= \begin{bmatrix} -\cos\gamma_{jk} & -V_k \sin\gamma_{jk} \\ \frac{1}{l_{jk}}\sin\gamma_{jk} & -\frac{V_k}{l_{jk}}\cos\gamma_{jk} \end{bmatrix} \begin{bmatrix} \dot{V}_k \\ \omega_k \end{bmatrix} + \ddot{F}^d_{jk} - \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}, \end{aligned} \tag{4.12}$$

where

$$\begin{aligned} s_1 &= \frac{1}{l_{jk}} V_k^2 \sin^2\gamma + \frac{1}{l_{jk}} V_j^2 \sin^2\eta_{jk} - \dot{V}_j \cos\eta_{jk} - V_j \sin\eta_{jk}\omega_j \\ &\quad -\frac{2}{l_{jk}} V_j V_k \sin\gamma_{jk} sin\eta_{jk}, \\ s_2 &= -\frac{2}{l_{jk}^2} V_j V_k \sin(\gamma_{jk}+\eta_{jk}) + \frac{V_j^2}{l_{jk}^2}\sin 2\eta_{jk} + \frac{V_k^2}{l_{jk}^2}\sin 2\gamma_{jk} + \frac{\dot{V}_j}{l_{jk}}\sin\eta_{jk} \\ &\quad -\frac{1}{l_{jk}} V_j \omega_j \cos\eta_{jk} - \dot{\omega}_j. \end{aligned} \tag{4.13}$$

Define a new formation control input $u_k \triangleq [\dot{V}_k \;\; \omega_k]^T \in \mathbb{R}^2$ and notate the input matrix as

$$g(\cdot) = \begin{bmatrix} -\cos\gamma_{jk} & -V_k \sin\gamma_{jk} \\ \frac{1}{l_{jk}}\sin\gamma_{jk} & -\frac{V_k}{l_{jk}}\cos\gamma_{jk} \end{bmatrix}$$

and $S = [s_1 s_2]^T$, we can write the error system in (4.12) as

$$\ddot{e} = g(\cdot)u_k + \ddot{F}^d_{jk} - S. \tag{4.14}$$

It is not difficult to check that the new input matrix $g(\cdot)$ is nonsingular under the condition that the flight speed $V_k \geq V_{min} > 0$ and $l_{jk} \geq l_{min} > 0$, where V_{min} is the UAV stalling speed, and l_{min} is the minimum distance to avoid collision between the two UAVs.

Let us define the control input u_k as

$$u_k = g^{-1}(\cdot)(-\ddot{F}^d_{jk} + S - Ke - \dot{e}), \tag{4.15}$$

where $K = \mathrm{Diag}[k_1 \;\; k_2]$, and $k_{1,2} \in \mathbb{R}^+$.

Substitute (4.15) into (4.14), we obtain a new error system

$$\ddot{e} = -Ke - \dot{e}. \tag{4.16}$$

Clearly, we reduce the nonlinear error system (4.14) into a linear error system. With a carefully chosen K, system (4.16) can be stable. Therefore, under the action of control law (4.15), the formation can be kept.

4.3.2 Sliding Mode Control

In section 4.3.1, by using the feedback linearization technique, we obtain a stabilizing control law to reduce a nonlinear error system into a linear stable error system. Though theocratically sound, this technique is practical only under the assumption of a perfect plant model. However, a perfect model is not always available. The feedback linearization technique may not achieve acceptable performance in a real world application. Let us consider a sliding mode controller in this section.

Rewrite (4.11) into the following form,

$$\dot{e} = \dot{F}^d_{jk} - g_1(\cdot)V_k - g_2(\cdot)V_j - g_3(\cdot)\omega_j, \tag{4.17}$$

where the vector fields $g_1(\cdot)$, $g_2(\cdot)$, and $g_3(\cdot)$ are defined as

$$g_1(\cdot) = \begin{bmatrix} \cos\gamma_{jk} \\ -\frac{1}{l_{jk}}\sin\gamma_{jk} \end{bmatrix}, \tag{4.18}$$

$$g_2(\cdot) = \begin{bmatrix} -\cos\eta_{jk} \\ \frac{1}{l_{jk}}\sin\eta_{jk} \end{bmatrix}, \tag{4.19}$$

$$g_3(\cdot) = \begin{bmatrix} 0 \\ -1 \end{bmatrix}. \tag{4.20}$$

Differentiating both sides of (4.17) with respect to time and after some algebraic and

trigonometric manipulation, we obtain

$$\begin{aligned}
\ddot{e} &= \ddot{F}^d_{jk} - \dot{g}_1(\cdot)V_k - g_1(\cdot)\dot{V}_k - \dot{g}_2(\cdot)V_j - g_2(\cdot)\dot{V}_j - \dot{g}_3(\cdot)\omega_j - g_3(\cdot)\dot{\omega}_j \\
&= \ddot{F}^d_{jk} - \left[\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}\dot{\gamma}_{jk} + \frac{\partial g_1(\cdot)}{\partial l_{jk}}\dot{l}_{jk}\right] V_k - g_1(\cdot)\dot{V}_k - \dot{g}_2(\cdot)V_j - g_2(\cdot)\dot{V}_j - g_3(\cdot)\dot{\omega}_j \\
&= \ddot{F}^d_{jk} - \left[\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}(\omega_j + \dot{\eta}_{jk} - \omega_k) + \frac{\partial g_1(\cdot)}{\partial l_{jk}}\dot{l}_{jk}\right] V_k \\
&\quad -g_1(\cdot)\dot{V}_k - \dot{g}_2(\cdot)V_j - g_2(\cdot)\dot{V}_j - g_3(\cdot)\dot{\omega}_j \\
&= \ddot{F}^d_{jk} - \left[\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}\dot{\eta}_{jk} + \frac{\partial g_1(\cdot)}{\partial l_{jk}}\dot{l}_{jk}\right] V_k - g_1(\cdot)\dot{V}_k + \frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}V_k\omega_k \\
&\quad -\dot{g}_2(\cdot)V_j - \frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}V_k\omega_j - g_2(\cdot)\dot{V}_j - g_3(\cdot)\dot{\omega}_j.
\end{aligned}$$

Thus,

$$\ddot{e} = \ddot{F}^d_{jk} - \left[\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}\dot{\eta}_{jk} + \frac{\partial g_1(\cdot)}{\partial l_{jk}}\dot{l}_{jk}\right] V_k + g(\cdot)u_k + f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j. \tag{4.21}$$

where

$$\begin{aligned}
f_1(\cdot) &= \begin{bmatrix} -\dot{g}_2(\cdot) & -\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}V_k \end{bmatrix} \in \mathbb{R}^{2\times 2}, \\
f_2(\cdot) &= \begin{bmatrix} -g_2(\cdot) & -g_3(\cdot) \end{bmatrix} \in \mathbb{R}^{2\times 2};
\end{aligned}$$

also, we define the leader linear and angular velocity vector as $\varpi_j = [V_j \;\; \omega_j]^T \in \mathbb{R}^2$.

As a new approach in this sub-section, we assume that the leader's linear and angular velocity and acceleration vectors ϖ_j and $\dot{\varpi}_j$ are known to the follower aircraft (by leader's onboard instruments and the communication link between the leader and followers).

Let us design the following filtered signal $r = [r_1 \;\; r_2]^T \in \mathbb{R}^2$

$$r = \dot{e} + Ke, \tag{4.22}$$

where $K = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$, and $k_{1,2} \in \mathbb{R}_+$ are design gain constants. Differentiating (4.22) with respect to time and substituting in (4.21) produces

$$\begin{aligned}
\dot{r} &= \overset{..d}{F}_{jk} - \left(\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}}\dot{\eta}_{jk} + \frac{\partial g_1(\cdot)}{\partial l_{jk}}\dot{l}_{jk}\right) V_k + g(\cdot)u_k \\
&\quad +f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + K\dot{e}.
\end{aligned} \tag{4.23}$$

Based on the subsequent Lyapunov analysis, the formation control law becomes

$$\begin{aligned} u_k &= g^{-1}(\cdot)\Bigg[-\ddot{F}_{jk}^{d} + \left(\frac{\partial g_1(\cdot)}{\partial \gamma_{jk}} \dot{\eta}_{jk} + \frac{\partial g_1(\cdot)}{\partial l_{jk}} \dot{l}_{jk} \right) V_k \\ &\quad - f_1(\cdot)\varpi_j - f_2(\cdot)\dot{\varpi}_j - K\dot{e} - \phi(r) \Bigg], \end{aligned} \tag{4.24}$$

where $\phi(r) \in \mathbb{R}^2$ is a sliding vector function so that the sliding condition is guaranteed.

We now state the main stability result of this section for the proposed formation controller.

Theorem 4.1 *The control law of (4.24) ensures stable sliding surface dynamics of the system in (4.22) and that all system signals are bounded under closed-loop operation and the tracking error is asymptotically stable in the sense that*

$$\lim_{t\to\infty} e,\ \dot{e} = 0. \tag{4.25}$$

Proof: To prove the theorem, let us construct the following non-negative function

$$V = \frac{1}{2} r^T r. \tag{4.26}$$

By using the control law (4.24) in equation (4.23), we have

$$\dot{r} = -\phi(r). \tag{4.27}$$

Now, taking the time derivative of (4.26) and substituting (4.27) yields

$$\dot{V} = r^T \dot{r} = -r^T \phi(r). \tag{4.28}$$

The sliding vector function can be, for instance, $\phi(r) = r + a[\mathrm{sgn}(r_1)\ \mathrm{sgn}(r_2)]^T$, where $a \in \mathbb{R}$ is a positive constant. Then (4.28) becomes

$$\dot{V} = -\left\|r\right\|^2 - a\left|r\right| \leq -a\left|r\right|.$$

Therefore, V (4.26) is a Lyapunov function and system (4.22) is asymptotically stable, which means $\lim_{t\to\infty} r = 0$. From (4.22) and with reference to Lemma A.8 of [120], we conclude that $\lim_{t\to\infty} e = 0$ and $\lim_{t\to\infty} \dot{e} = 0$. ■

Remark 4.1 *A generic sliding vector function $\phi(r)$ is given in the above control design, so a variety of available sliding vector functions can be substituted in to reduce the chattering and to achieve satisfactory performance.*

4.3.3 Robust Control

The effectiveness of the control law introduced in Section 4.3.2 is based on the assumption, that the leader's position, attitude, velocity and acceleration information are all known to the follower. Those data can be obtained by onboard instruments (on leader and/or follower vehicles) and inter-vehicle communications. This is a common assumption in the formation control literature. However, in reality, due to the payload limitation or communication failure (for example, under electronic countermeasures), this assumption might not always hold. Consequently, control laws that assume full knowledge of leader aircraft's states may not guarantee a desired formation in the presence of communication failures. In [121], a failure detection and identification system based on an interacting multiple-model Kalman filter approach is proposed. When communication is unaccessible, it can provide accurate state estimation, which are required in formation flights. In [122, 123], graph theory is used to improve the robustness and fault tolerance of formation control when communication fails. A novel solution to a class of problems in feedback stabilization of coupled systems with limited communication is presented in [124].

In this section, we propose a robust nonlinear formation controller which requires no information of the leader's velocity and acceleration. Similar idea of the controller's design and analysis can be found in [125].

Specifically, by expanding the second term on the right side of equation (4.23) along with equation (4.9), we can rearrange (4.23) into

$$\begin{aligned} \dot{r} &= \ddot{F}_{jk}^{d} - \begin{bmatrix} \frac{1}{l_{jk}} \sin^2 \gamma_{jk} \\ \frac{2}{l_{jk}^2} \cos \gamma_{jk} \sin \gamma_{jk} \end{bmatrix} V_k^2 + \\ & g(\cdot)u_k + f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + f_3(\cdot)\varpi_j V_j + K\dot{e}, \end{aligned} \tag{4.29}$$

where $f_1(\cdot)$, $f_2(\cdot)$, and $f_3(\cdot)$ are functions of F_{jk}, V_k, and γ_{jk}. Note that $f_1(\cdot)$, $f_2(\cdot)$, and $f_3(\cdot)$ are bounded since $l_{jk} \geq l_{\min} > 0$ and $V_k \in \mathcal{L}_\infty$.

We make the following assumptions:

Assumption 4.1 *During the formation flight, the leader UAV is stably tracking some desired trajectories* $\varpi_j^d = \left[V_j^d \; \omega_j^d\right]^T$, $\dot{\varpi}_j^d = \left[\dot{V}_j^d \; \dot{\omega}_j^d\right]^T \in \mathbb{R}^2$ *with* ϖ_j^d, $\dot{\varpi}_j^d$, $\ddot{\varpi}_j^d \in \mathcal{L}_\infty$ *so that we can assume* ϖ_j, $\dot{\varpi}_j$, $\ddot{\varpi}_j \in \mathcal{L}_\infty$.

Assumption 4.2 *All other terms in (4.29) except* ϖ_j *and* $\dot{\varpi}_j$ *are known.*

Remark 4.2 *Using equation (4.3), it is possible to generate a leader UAV trajectory so that Assumption 4.1 holds.*

The follower control vector u_k in (4.23) becomes

$$\begin{aligned} u_k &= g^{-1}(\cdot)\Bigg(-\ddot{F}_{jk}^d + \begin{bmatrix} \frac{1}{l_{jk}} \sin^2 \gamma_{jk} \\ \frac{2}{l_{jk}^2} \cos \gamma_{jk} \sin \gamma_{jk} \end{bmatrix} V_k^2 \\ &\qquad -2K\dot{e} - K^2 e - \beta_1 \operatorname{sgn}(e(t))\Bigg), \end{aligned} \tag{4.30}$$

where β_1 is a constant positive control gain.

After substituting u_k into (4.29), the closed-loop system is given by

$$\begin{aligned} \dot{r} &= f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + f_3(\cdot)\varpi_j V_j \\ &\qquad -Kr - \beta_1 \operatorname{sgn}(e(t)). \end{aligned} \tag{4.31}$$

Before stating the main result of this section, we give the following lemma which will be invoked later.

Lemma 4.1 *Let the auxiliary function* $\Gamma(t) \in \mathbb{R}$ *be defined as follows*

$$\Gamma \triangleq r^T \left[f_1(\cdot)\varpi_j + f_2(\cdot)\dot{u}_j + f_3(\cdot)\varpi_j V_j - \beta_1 \operatorname{sgn}(e) \right]. \tag{4.32}$$

If the control gain β_1 is selected to satisfy the sufficient condition

$$\begin{aligned} \beta_1 \;\; > \;\; & \|f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + f_3(\cdot)\varpi_j V_j\|_2 + \\ & k_{min}^{-1} \left\| \frac{d(f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + f_3(\cdot)\varpi_j V_j)}{dt} \right\|_2, \end{aligned} \tag{4.33}$$

where $k_{min} = min\{k_1, k_2\}$, then

$$\int_{t_0}^{t} \Gamma(\tau) d\tau \leqslant \zeta_b, \tag{4.34}$$

where the positive constant $\zeta_b \in \mathbb{R}$ is defined as

$$\zeta_b \triangleq \beta_1 \|e(t_0)\|_1 - e^T(t_0) \left[f_1(t_0)\varpi_j(t_0) + f_2(t_0)\dot{\varpi}_j(t_0) + f_3(t_0)\varpi_j(t_0)V_j(t_0) \right] \tag{4.35}$$

where the notation $\| \cdot \|_1$ denotes the $\mathcal{L}_1$ norm.

Proof: Before giving a formal proof of Lemma 4.1, we first show that

$$\int_0^t \text{sgn}(y)\dot{y} d\tau = |y(t)| - |y(0)|. \tag{4.36}$$

Since

$$|y| = \sqrt{y^2}, \tag{4.37}$$

taking the derivative of $\sqrt{y^2}$ yields

$$\frac{d\sqrt{y^2}}{dt} = \frac{1}{2} \left(y^2\right)^{-\frac{1}{2}} 2y\dot{y} = \frac{y}{\sqrt{y^2}}\dot{y} = \frac{y}{|y|}\dot{y} = \text{sgn}(y)\dot{y}. \tag{4.38}$$

Then integrating both sides, we obtain

$$\int_0^t \text{sgn}(y)\dot{y} d\tau = \int_0^t \frac{d\sqrt{y^2}}{dt} d\tau = \sqrt{y^2}|_0^t = |y||_0^t = |y(t)| - |y(0)|. \tag{4.39}$$

Thus, equation (4.36) holds.

To simplify the notations, let us define

$$\Omega(t) = f_1(t)\varpi_j(t) + f_2(t)\dot{\varpi}_j(t) + f_3(t)\varpi_j(t)V_j(t).$$

After substituting (4.22) into (4.32) and then integrating $\Gamma(t)$ in time, we obtain

$$\begin{aligned}\int_{t_0}^{t}\Gamma(\tau)d\tau &= \int_{t_0}^{t}e^{T}K^{T}(\tau)\left(\Omega(\tau)-\beta_1\,\mathrm{sgn}\,e(\tau)\right)d\tau \\ &+ \int_{t_0}^{t}\frac{de^{T}(\tau)}{d\tau}\Omega(\tau)d\tau-\beta_1\int_{t_0}^{t}\frac{de^{T}(\tau)}{d\tau}\,\mathrm{sgn}(e(\tau))d\tau\,. \end{aligned} \tag{4.40}$$

After integrating the second term on the right-hand side of (4.40) by parts, and utilizing equation (4.36) for the third term, we have the following simplified expression

$$\begin{aligned}\int_{t_0}^{t}\Gamma(\tau)d\tau &= \int_{t_0}^{t}e^{T}K^{T}(\tau)\left(\Omega(\tau)-K^{T^{-1}}\frac{d\Omega(\tau)}{d\tau}-\beta_1\,\mathrm{sgn}(e(\tau))\right)d\tau \\ &+ e^{T}(t)\Omega(t)-e^{T}(t_0)\Omega(t_0)-\beta_1\|e\,(t)\,\|_1+\beta_1\|e\,(t_0)\,\|_1. \end{aligned} \tag{4.41}$$

We can now upper bound the right-hand side of (4.41), that is

$$\begin{aligned}\int_{t_0}^{t}\Gamma(\tau)d\tau &\leqslant \int_{t_0}^{t}k_{min}\|e\,(\tau)\,\|_1\left(\|\Omega(\tau)\|_2+k_{min}^{-1}\left\|\frac{d\Omega(\tau)}{d\tau}\right\|_2-\beta_1\right)d\tau \\ &+ \|e\,(t)\,\|_1\left(\|\Omega(t)\|_2-\beta_1\right)+\beta_1\|e\,(t_0)\,\|_1-e^{T}(t_0)\Omega(t_0) \end{aligned} \tag{4.42}$$

From (4.42), it is easy to see that if β_1 is chosen according to (4.33), then the first and second term on the right hand side are less than zero, then we have

$$\int_{t_0}^{t}\Gamma(\tau)d\tau \leqslant \beta_1\|e\,(t_0)\,\|_1-e^{T}(t_0)\Omega(t_0). \tag{4.43}$$

Clearly, (4.43) is (4.34). ■

We now state the main stability result for the second controller in the following theorem.

Theorem 4.2 *The control law of (4.30) ensures that all system signals are bounded under closed-loop operation and the tracking error is asymptotically stable in the sense that*

$$\lim_{t\to\infty}e(t),\ \dot{e}(t)=0. \tag{4.44}$$

Proof: Let $P(t)\in\mathbb{R}$ be an auxiliary function as follows

$$P(t)\triangleq\zeta_b-\int_{t_0}^{t}\Gamma(\tau)d\tau\geqslant 0, \tag{4.45}$$

where ζ_b and $\Gamma(t)$ have been defined in Lemma 4.1. Based on the non-negativity of $P(t)$, we define a Lyapunov function candidate V by

$$V \triangleq \frac{1}{2} r^T r + P. \tag{4.46}$$

After taking the time derivative of (4.46), we have $\dot{V} = r^T \dot{r} + \dot{P}$. Then utilize the closed-loop dynamics of (4.31) and (4.45), we can obtain the following expression

$$\begin{aligned} \dot{V} &= r^T [f_1(\cdot)\varpi_j + f_2(\cdot)\dot{\varpi}_j + f_3(\cdot)\varpi_j V_j \\ &\quad -Kr - \beta_1 \operatorname{sgn}(e(t))] - \Gamma \end{aligned} \tag{4.47}$$

Rearranging the first term on the right hand side of (4.47) and use the definition of Γ, we get

$$\dot{V} = -r^T K r \leqslant -k_{min} \|r\|^2. \tag{4.48}$$

Therefore, $r(t) \in \mathcal{L}_\infty \cap \mathcal{L}_2$ and $\lim_{t\to\infty} r(t) = 0$. With (4.22) again, it is easy to see $e(t), \dot{e}(t) \in \mathcal{L}_\infty \cap \mathcal{L}_2$ and $\lim_{t\to\infty} e(t), \dot{e}(t) = 0$. ■

4.4 Simulation Results

In this section, the performance of proposed controllers is tested by simulations. In order to minimize the chattering of sliding mode controller, instead of $\operatorname{sgn}(\cdot)$, $\arctan(\cdot)$ is used in (4.24) and (4.30). To simplify notation, the control algorithms described in Sections 4.3.1, 4.3.2 and 4.3.3 are referred as C_1, C_2, and C_3 , respectively.

4.4.1 Formation under Feedback Linearization Controller

The desired formation in this simulation is $l_{jk}^d = 100$ m and $\eta_{jk}^d = \frac{5}{4}\pi$ rad. Total simulation time is 60 seconds. Controller parameter is $K = \operatorname{Diag}[2,\ 2]$.

The leader is given a constant angular velocity command $\omega_{jc} = 0.05$ rad/s and a constant velocity command$V_{jc} = 17.5$ m/s. This means that the leader moves in a circle.

The initial conditions of the leader are $x_j(0) = y_j(0) = 0$ m, $z_j(0) = 1000$ m, $\psi_j(0) = \pi$ rad, $V_j(0) = 17.5$ m/s, and $\omega_j(0) = 0$ rad/s. For the follower, the initial conditions are $x_k(0) = 100$ m, $y_k(0) = 200$ m, $z_k(0) = 1000$ m, $\psi_j(0) = \pi$ rad, $V_k = 20$ m/s, and $\omega_k(0) = 0$ rad/s.

Figure 4.4 to Figure 4.7 show the response of the feedback linearization formation controller C_1. The trajectory of the 2-UAV team under the action of C_1 is presented in Figure 4.4. Figure 4.5 and 4.6 depict the control inputs and formation errors when C_1 is used. The actual relative position is shown in Figure 4.7. As it can be seen, formation errors converge to zero and the follower UAV is able to maintain the desired relative distance and the bearing angle with respect to the leader UAV.

4.4.2 Formation under Sliding Mode Controller

In this simulation, the desired formation is $l_{jk}^d = 100$ m and $\eta_{jk}^d = \frac{5}{4}\pi$ rad. Total simulation time is 200 seconds. Controller parameter is $K = \mathrm{Diag}[2,\ 2]$.

The leader is given a constant velocity command $V_{jc} = 17.5$ m/s. The angular velocity command ω_{jc} is 0 rad/s most of the time and $\omega_{jc} = 0.1$ rad/s for time periods $[30, 45]$, $[60, 75]$ and $[110, 125]$.

The initial conditions of the leader are $x_j(0) = y_j(0) = 0$ m, $z_j(0) = 1000$ m, $\psi_j(0) = \pi$ rad, $V_j(0) = 17.5$ m/s, and $\omega_j(0) = 0$ rad/s. For the follower, the initial conditions are $x_k(0) = 150$ m, $y_k(0) = 20$ m, $z_k(0) = 900$ m, $\psi_j(0) = \pi$ rad, $V_k = 20$ m/s, and $\omega_k(0) = 0$ rad/s.

Figure 4.8 to Figure 4.10 show the response of the sliding mode formation controller C_2. The trajectory of the 2-UAV team under the action of C_2 is presented in Figure 4.8. Figure 4.9 and 4.10 depict the control inputs and formation errors when C_2 is used. As it can be seen, formation errors converge to zero the win-airplane is able to maintain the desired relative distance and bearing angle with respect to the leader aircraft.

4.4.3 Formation under Robust Controller

To compare with the sliding mode formation controller, similar simulation setup is used. In this simulation, the desired formation is $l_{jk}^d = 100$ m and $\eta_{jk}^d = \frac{5}{4}\pi$ rad. Total simulation time is 200 seconds. Controller parameters are $K = \text{Diag}[2,\ 2]$ and $\beta_1 = \text{Diag}[5,\ 5]$.

The leader is given a constant velocity command $V_{jc} = 17.5$ m/s. The angular velocity command ω_{jc} is 0 rad/s most of the time and $\omega_{jc} = 0.1$ rad/s for time periods $[30, 45]$, $[60, 75]$ and $[110, 125]$.

The initial conditions of the leader are $x_j(0) = y_j(0) = 0$ m, $z_j(0) = 1000$ m, $\psi_j(0) = \pi$ rad, $V_j(0) = 17.5$ m/s, and $\omega_j(0) = 0$ rad/s. For the follower, the initial conditions are $x_k(0) = 150$ m, $y_k(0) = 20$ m, $z_k(0) = 900$ m, $\psi_j(0) = \pi$ rad, $V_k = 20$ m/s, and $\omega_k(0) = 0$ rad/s.

Figure 4.11 to Figure 4.13 show the response of the sliding mode formation controller C_2. The trajectory of the 2-UAV team under the action of C_3 is presented in Figure 4.11. It is close to the result of Section 4.4.2. Figure 4.12 and 4.13 depict the control inputs and formation error when C_2 is used. Again, the controller C_3 can maintain the win-airplane at the desired relative distance and bearing angle with respect to the lead aircraft.

Remark 4.3 *Note that the initial amplitudes of the control inputs under C_3 are higher than the ones given by C_2. We anticipate this behavior since C_3 assumes no information about the velocity and acceleration of the leader aircraft. Nevertheless, the overall performance of the closed-loop system under C_3 is comparable with the performance under C_2. C_3 is the preferred controller in case of inter-vehicle communication failure or when the communication bandwidth is limited. A switching-logic scheme can be designed so that C_2 is used when communication between UAVs is possible and the follower UAV has access to its leader's velocity and acceleration; otherwise, C_3 should be switched in. Stability of the switching formation controller becomes an issue that would require further investigation. This analysis is out of the scope of this dissertation.*

4.5 Summary

In this chapter, we present a two-layered control system that allows a team of UAVs to navigate in leader-following formations. At the low-layer, an off-the-shelf autopilot operating in *holding mode* stabilizes the UAV. At the higher layer, three stable nonlinear formation controllers are developed. This hierarchical control scheme allows a team of UAVs to perform complex navigation tasks under limited inter-vehicle communication. Specifically, the third robust control law eliminates the requirement of leader's velocity and acceleration information, which reduces the communication overhead.

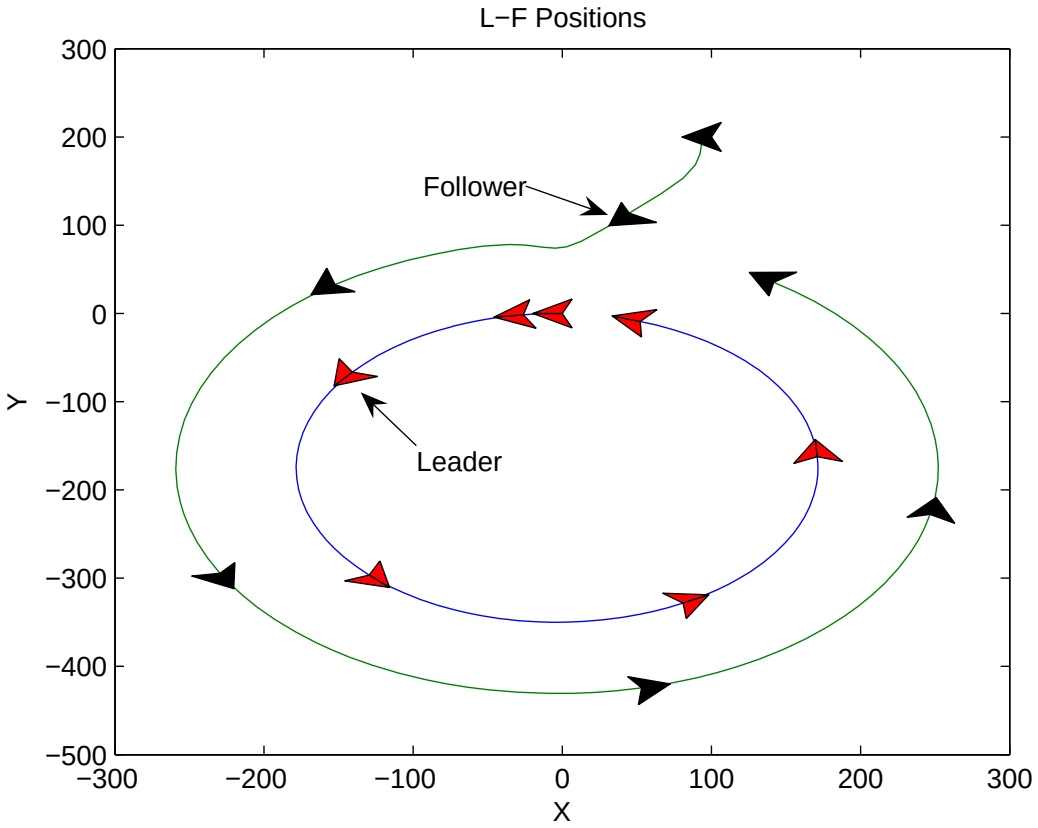


Figure 4.4: Trajectories of the UAVs in close formation under the action of C_1.

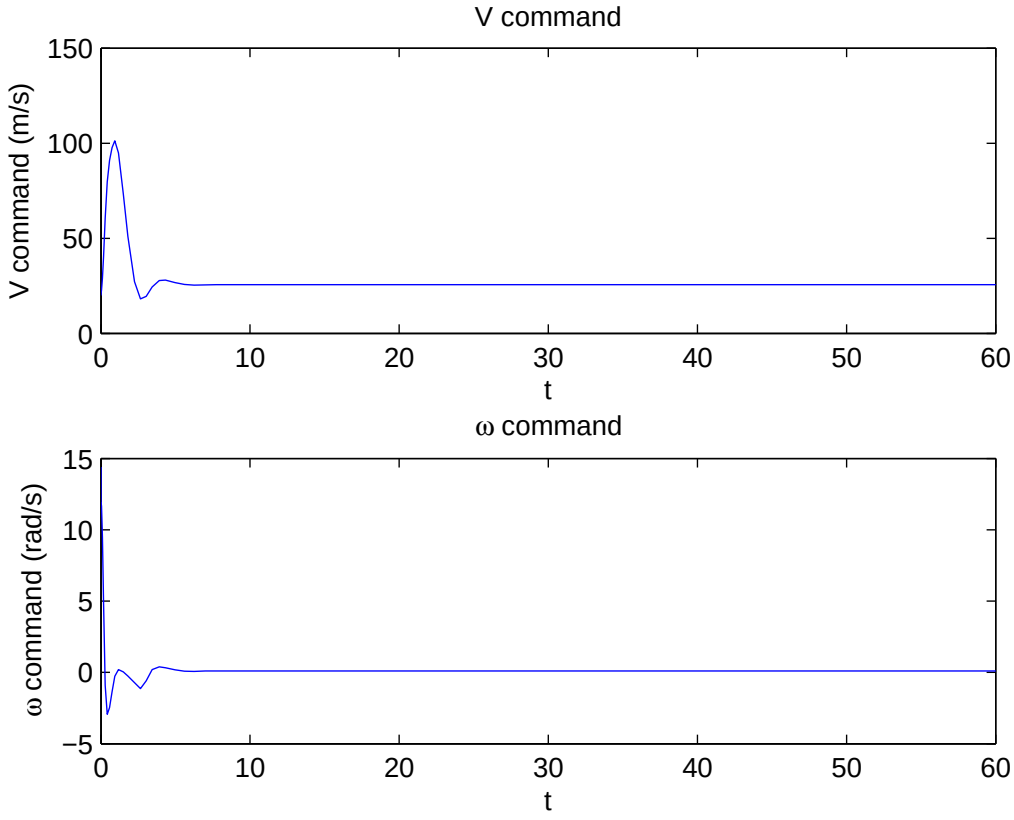


Figure 4.5: Control inputs generated by C_1.

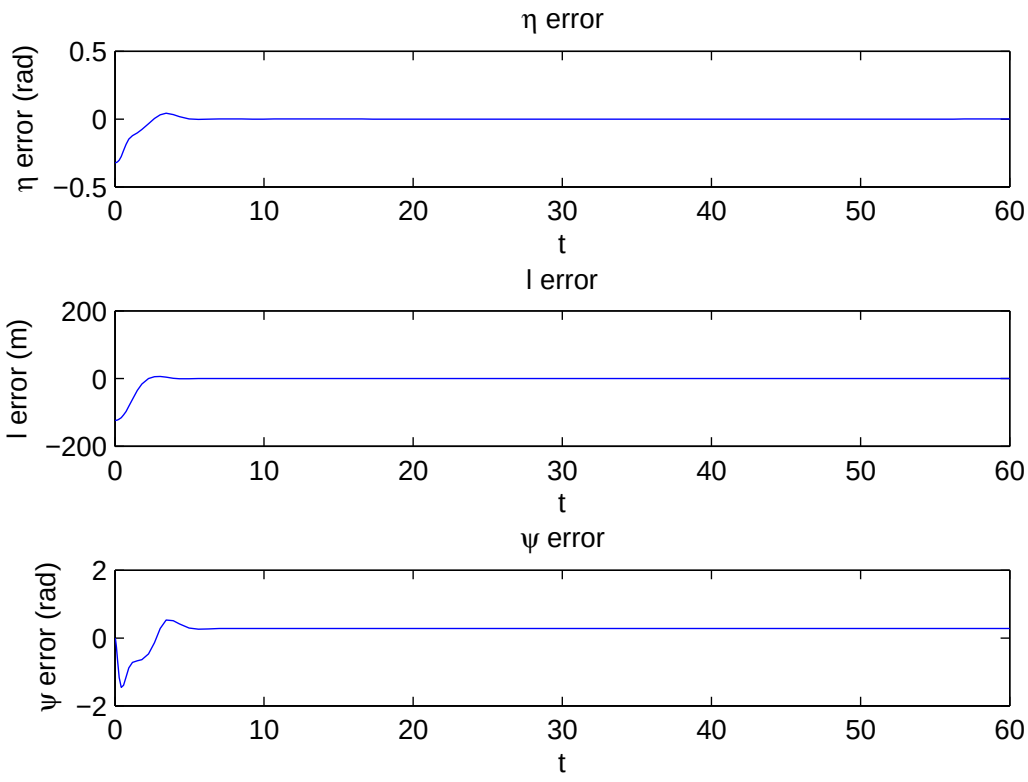


Figure 4.6: Formation errors under the action of C_1.

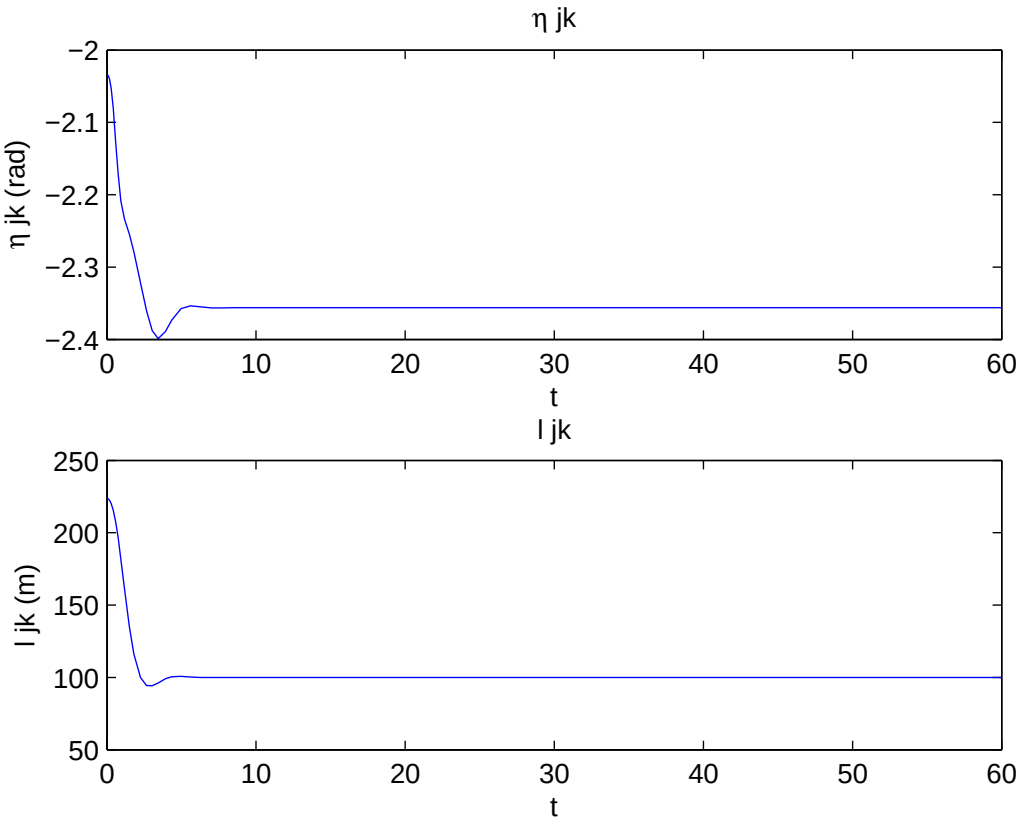


Figure 4.7: Relative position under the action of C_1.

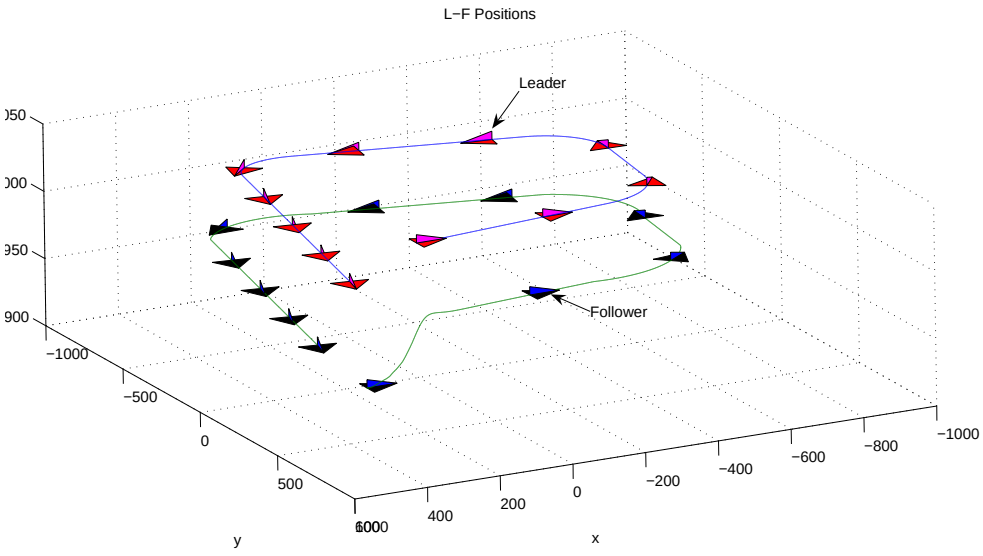


Figure 4.8: Trajectories of the UAVs in close formation under the action of C_2.

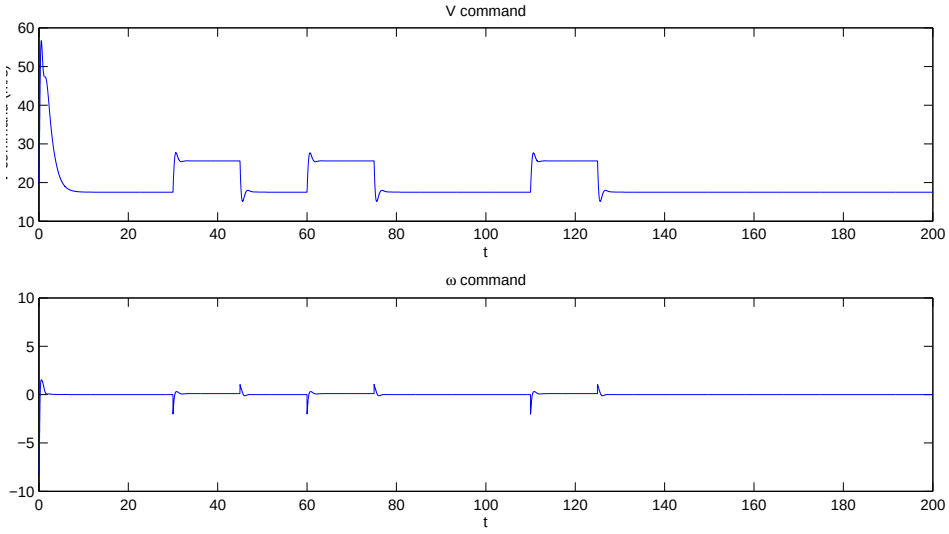


Figure 4.9: Control inputs generated by C_2.

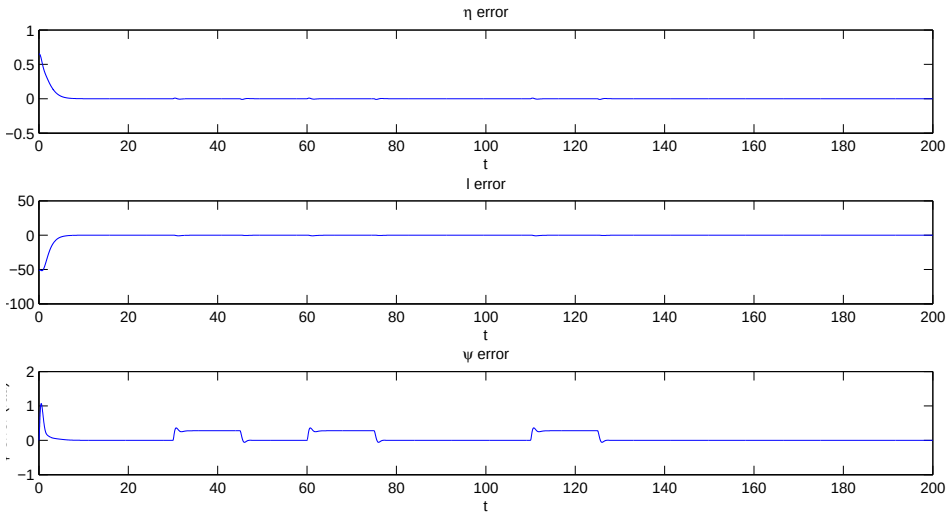


Figure 4.10: Formation errors under the action of C_2.

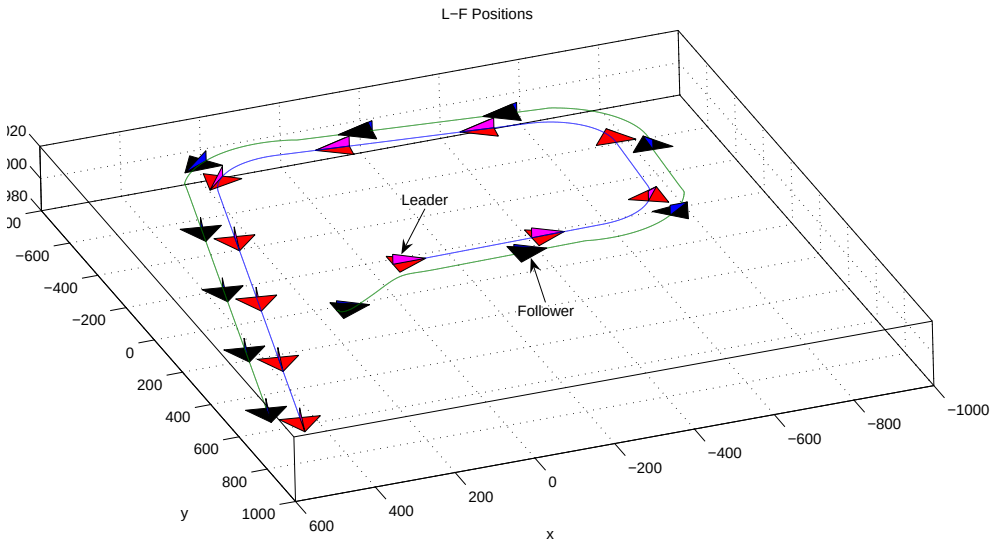


Figure 4.11: Trajectories of the UAVs in close formation under the action of C_3.

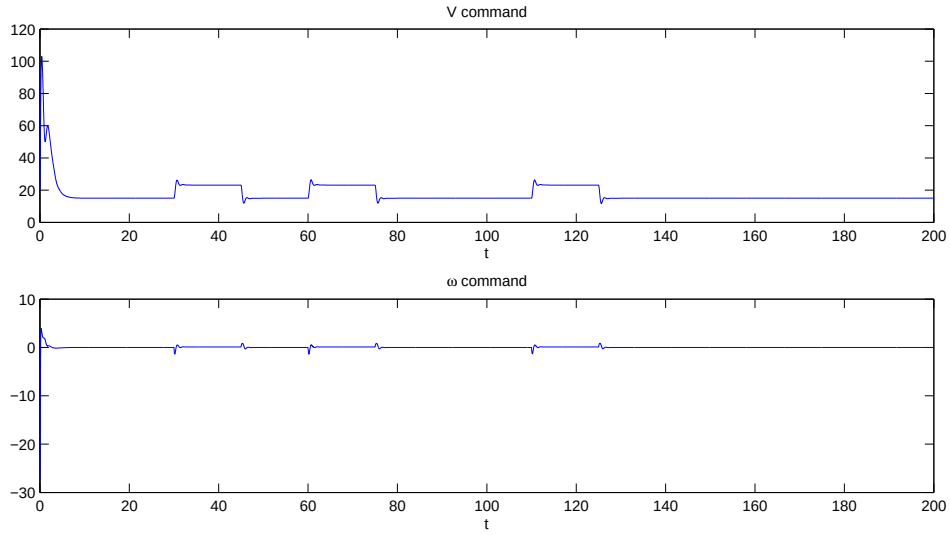


Figure 4.12: Control inputs generated by C_3.

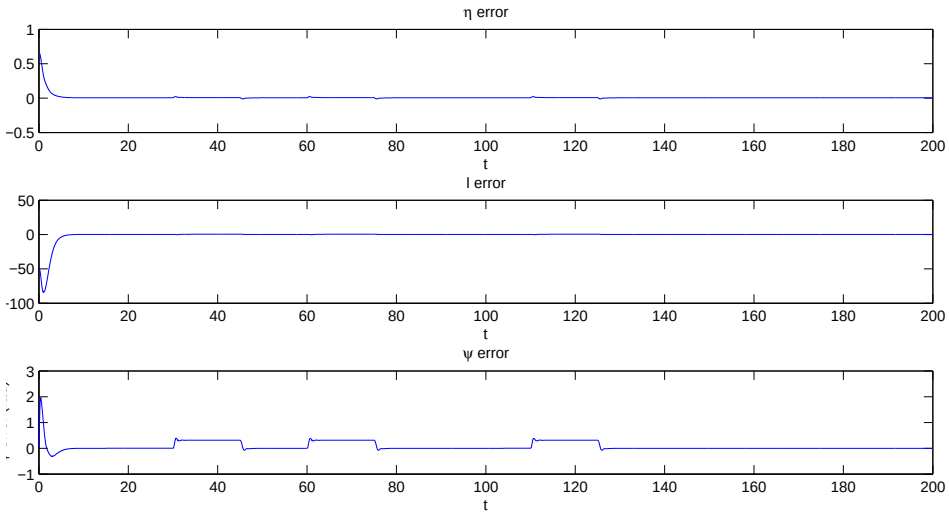


Figure 4.13: Formation errors under the action of C_3.

CHAPTER 5

Dual-mode Model Predictive Formation Control

In this chapter, we consider using model predictive control (MPC) to solve the problem of controlling a team of mobile robots with nonholonomic constraints to leader-following formations. We propose that it is more convenient to put the nonholonomic constraints inside the model predictive control framework. As the first step of exploration, a dual-mode MPC algorithm is developed. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. The effectiveness of the method is investigated by numerical simulations.

5.1 Introduction

A dynamic network consists of spatially distributed dynamic nodes (*e.g.*, autonomous vehicles, mobile sensors) which are coordinated by common set of goals and possible dynamic interaction between the nodes. There are many applications where a dynamic network may be more suitable than a single vehicle, especially where a distributed system of sensors is advantageous. For example, in search-and-rescue operations, deployment of many vehicles

over an area can allow for more thorough and faster coverage. Other applications, to mention a few, are environmental monitoring, surveillance and reconnaissance, acquisition and tracking. Yet without coordinating the movement of agents, any advantage of multi-vehicle deployment may be lost and damaging collisions or interference may occur.

One interesting problem in multi-robot coordination is how to drive a group of robots to a desired formation. Unmanned ground vehicle (UGV) formations can provide a promising and efficient alternative to existing techniques in a wide range of applications. Many researchers have been working on formation problems, and numerous control algorithms can be found in the literature (see *e.g.*, [126],[127], [128]).

Recently, model predictive control (MPC) or receding horizon control (RHC) has gained more and more attention in the control community. The inherent ability of MPC to handle constrained systems makes it a promising technique for cooperative control, especially for multi-vehicle formation control. Recent work includes [127], [128]. Other applications of MPC, such as controlling nonholonomic mobile robots are described in [30], [45]; for multi-vehicle coordination are given in [129]. The stability and feasibility of the MPC algorithms become a new challenge (see discussion in [130]).

In this chapter, based on previous work [131], [6], we show that it is more convenient to put the vehicles's nonholonomic constraints inside the MPC framework. Moreover, we present a novel MPC algorithm for mobile robot formations. Since a stabilizing terminal controller is switched in within a specified terminal constraint set, the proposed MPC algorithm is dual-mode [103]. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed. For the choice of stabilizing terminal controller, a comparison between an input-output feedback linearization controller used in [131] and a robust formation controller used in [6] is given.

The rest of the chapter is organized as follows. In Section 5.2, some preliminaries are briefly introduced. A dual-mode MPC algorithm are proposed in Section 5.3. Stability results are provided in Section 5.4. Section 5.5 contains simulation results. Finally, con-

cluding remarks are given in Section 5.6.

5.2 Preliminaries

The problem considered in this paper is to drive a team of nonholonomic vehicles to a desired formation. This section describes the model used for the mobile agents and the definition of formation.

5.2.1 Vehicle Model

Consider the planar motion of nonholonomic unicycle-model robots whose kinematics are determined by

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \tag{5.1}$$

where the subscript $i \in [1, \ldots, N]$ indicates the i^{th} UGV. (x_i, y_i) are the Cartesian coordinates of the robot, $\theta_i \in (-\pi, \pi]$ represents the orientation of the robot with respect to the positive x axis, and v_i and ω_i are linear and angular velocities, respectively.

5.2.2 Formation and Formation Control Graph

Definition 5.1 *A formation is a network of agents interconnected via their controller* specifications *that dictate the relationships each agent must maintain with respect to its neighbors. The interconnections between agents are modeled as edges in a directed acyclic graph, labeled by a given relationship [59], [132].*

Definition 5.2 *A formation control graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ is a directed acyclic graph consisting of the following:*

- *A finite set $\mathcal{V} = (V_1, \ldots, V_N)$ of N vertices and a map assigning each vertex V_i to a control system (5.1).*

- *An edge set* $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ *of pair-wise neighbors encoding the formation between agents. If the ordered pair* $(V_i, V_j) \in \mathcal{E}$*, then* $(V_j, V_i) \notin \mathcal{E}$*, and* $(V_k, V_j) \notin \mathcal{E}$ *for all* $k \in \{1, \ldots, N\} \backslash i$*.*

- *A set of constants* $\mathcal{F} = \left\{ F_{ij}^d \in \mathbb{R}_- \times \mathbb{R} \right\}$ *defining control objectives, or* set points*, for each node* j*, such that* $(V_i, V_j) \in \mathcal{E}$ *for some* $V_i, V_j \in \mathcal{V}$*.*

Consequently, by changing F_{ij}^d, we are able to define different formation shapes for the mobile robot team.

5.3 Controllers for Multi-Robot Coordination

5.3.1 Formation Error

Let a triplet $p_i = [x_i \;\; y_i \;\; \theta_i]^T$ describe the position and the orientation of the i^{th} mobile robot. Let $F_{ij}^d = \begin{bmatrix} \Delta x_{ij}^d & \Delta y_{ij}^d \end{bmatrix}^T$ be the desired formation between robots i and j. $\Delta x_{ij}^d \in \mathbb{R}_-$ and $\Delta y_{ij}^d \in \mathbb{R}$ are the desired position for robot j in a local Cartesian reference frame $\mathcal{C}$ attached to robot i. Then the actual formation for robot-pair i and j is described by $F_{ij} = [\Delta x_{ij} \quad \Delta y_{ij}]^T$.

Figure 5.1 shows the formation configuration for two UGVs.

Let us define the formation error for the j^{th} robot in a robot-pair $(R_i,\ R_j)$

$$x_j^e = \begin{bmatrix} \Delta x_{ij}^d - \Delta x_{ij} \\ \Delta y_{ij}^d - \Delta y_{ij} \\ \theta_i - \theta_j \end{bmatrix}. \tag{5.2}$$

5.3.2 Dual-mode MPC

Model predictive control (MPC), as an effective method to solve multi-variable constrained control problems, has been used in industry for more than 20 years. Different from conventional control which uses pre-computed control laws, MPC is a technique in which the

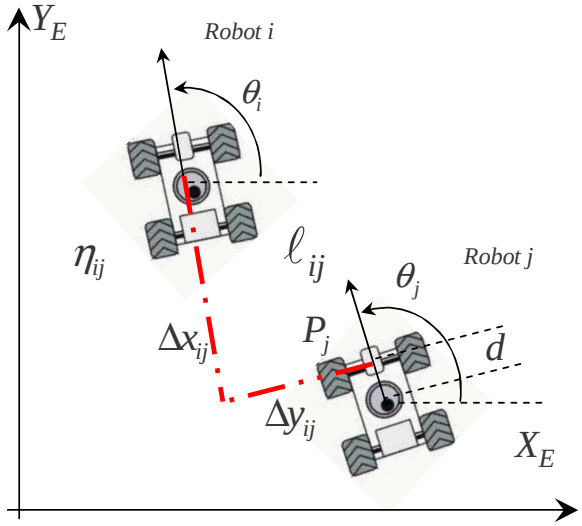


Figure 5.1: Formation configuration for two UGVs.

current control action is obtained by solving, at each sampling instance, a finite-horizon optimal control problem. Each optimization yields an open-loop optimal control sequence and the first control of this sequence is applied to the plant until the next sampling instance.

For a robot-pair $(R_i,\, R_j)$, which has an ordered pair $(V_i, V_j) \in \mathcal{E}$ in the *formation control graph* $\mathcal{G}$ and a *set point* $F_{ij}^d \in \mathcal{F}$, a control input u_j needs to be determined for robot j. With the assumption that robot i's current and future control action $\hat{u}_i$ are known to robot j, the formation-error system for robot $j \in \{1, \ldots, N\}$ at time t_k can be defined as follows

$$\begin{aligned} x_j^e(k+1) &= f(x_j^e(k), u_j(k)), \\ x_j^e(k) &\in \mathcal{X},\ u_j(k) \in \mathcal{U}, \end{aligned} \tag{5.3}$$

where $f(\cdot)$ is continuous at the origin, with $f(0,0) = 0$; $\mathcal{X} \subset \mathbb{R}^3$ contains the origin in its interior; $\mathcal{U}$ is a compact subset of $\mathbb{R}^2$ containing the origin in its interior.

To obtain the current control $u_j(k)$ at time t_k, where k is a nonnegative integer ($k \in \mathbb{Z}^*$), a finite-horizon optimal control problem

$$P(x_j^e, k) := \min_{u_j} \{J_H(x_j^e, k, u_j)\},$$

must be solved online. $J_H(x_j^e, k, u_j)$ is the performance index and $H \in \mathbb{N}$ is the horizon length (for simplicity, the prediction horizon equals the control horizon in this paper). Q and R are positive definite symmetric matrices.

To ensure stability of the MPC algorithm, a terminal equality constraint $x_j^e(k+H) = 0$ is commonly used. Therefore, $\Delta x_{ij} \rightarrow \Delta x_{ij}^d$, $\Delta y_{ij} \rightarrow \Delta y_{ij}^d$ and $\theta_j \rightarrow \theta_i$. However, an equality constraint usually makes the optimal control problem hard to solve. To balance the stability and feasibility, the terminal equality constraint can be relaxed to a terminal region. The MPC algorithm is only required to drive the error system to the edge of the terminal region. Inside the terminal region, a stabilizing terminal controller is switched in and it drives the error system to the equilibrium point. Such MPC algorithm is dual-mode [103].

Now let us define the terminal region $\mathcal{X}_f$, which is a convex compact subset of $\mathcal{X}$ containing the origin in its interior. Therefore, we can define a set $\mathcal{X}_f^c$, where $\mathcal{X}_f^c \cup \mathcal{X}_f = \mathcal{X}$ and $\mathcal{X}_f^c \cap \mathcal{X}_f = \varnothing$. Inside $\mathcal{X}_f$, a stabilizing terminal controller u_j^T is employed to drive the system (5.3) back to the origin. Note, the terminal region $\mathcal{X}_f$ should be positively invariant for the system $x_j^e(k+1) = f(x_j^e(k), u_j^T(k))$. Methods for constructing $\mathcal{X}_f$, which indeed is the terminal controller's region of attraction, can be found in [103], [133] (local linear controller case). For a nonlinear controller which has a region of attraction $\mathcal{X}$, such as (5.11), the terminal region can be a ball B_r, which contains the origin.

The incremental cost in the optimal control problem for robot $j \in \{1, \ldots, N\}$ is defined in the manner of [133]

$$L(x_j^e, u_j) = \lambda(x_j^e)\mathcal{L}(x_j^e, u_j), \tag{5.4}$$

where

$$\lambda(x_j^e) = \begin{cases} 0, & \text{if } x_j^e \in \mathcal{X}_f \\ 1, & \text{otherwise} \end{cases},$$

and

$$\mathcal{L}(x_j^e, u_j) = \|x_j^e\|_Q^2 + \|u_j\|_R^2. \tag{5.5}$$

Clearly, the incremental cost $L(\cdot)$ is continuous at the origin, with $L(0,0) = 0$.

Now, for robot $j \in \{1, \ldots, N\}$, given $(V_i, V_j) \in \mathcal{E}$ and $F_{ij}^d \in \mathcal{F}$, $q_i(k)$ and $q_j(k)$, $u_i(k+H-1, \cdots, k; k)$ at any update time instance t_k, the optimal control problem of dual-mode MPC algorithm is defined

$$P(x_j^e, k) := \min_{u_j} \{J_H(x_j^e, k, u_j)\}, \tag{5.6}$$

where

$$J_H(x_j^e, k, u_j) = \sum_{m=1}^{H} L(x_j^e(k+m; k), u_j(k+m-1; k)), \tag{5.7}$$

subject to

$$\begin{aligned} & x_j^e(k+1) = f(x_j^e(k), u_j(k)), \\ & x_j^e(k+H) \in \mathcal{X}_f, u_j(k) \in \mathcal{U}. \end{aligned} \tag{5.8}$$

The constraint $x_j^e(k+H) \in \mathcal{X}_f$ requires that the final state of the prediction horizon must reach the edge of the terminal region (see Fig 5.2).

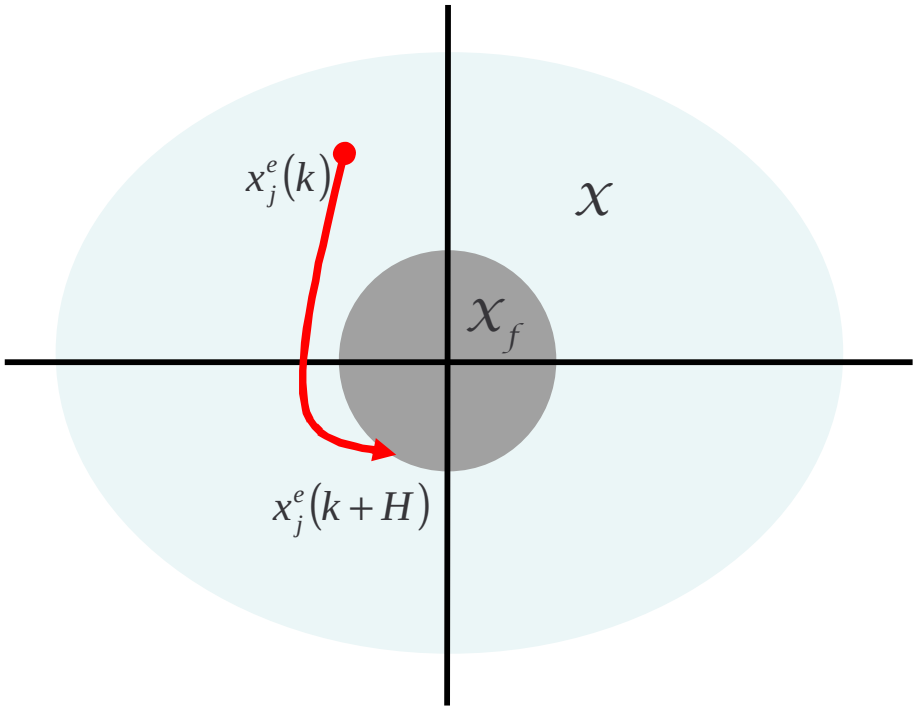


Figure 5.2: Terminal constraint of dual-mode MPC.

From the definition of incremental cost, the objective function $J_H(x_j^e, k, u_j)$ is greater than or equal to 0 and $J_H(x_j^e, k, u_j) = 0$ only when $x_j^e = 0$ and $u_j = 0$.

The solution of optimal control problem (5.6) is denoted as $u^*(k) = u_j^*(k+m-1;k)$. The optimal state trajectory under this control action is $x_j^{e*}(k) = x_j^{e*}(k+m;k)$. The corresponding optimal performance index is $J_H^*(k) = J_H(x_j^{e*}(k+m;k), k, u_j^*(k+m-1;k))$ where $m \in [1, \ldots, H]$.

Now the dual-mode model predictive controller for robot $j \in \{1, \ldots, N\}$ is stated in the following algorithm.

Algorithm:

Data: initial states of robots $p_i(0)$, $p_j(0)$, $H \in \mathbb{N}$.

Initial: At time instance $t_k = 0$, if $x_j^e(0) \in \mathcal{X}_f$, switch to the terminal controller u_j^T for all k such that $x_j^e(k) \in \mathcal{X}_f$. Else set $\hat{u}_i(l;k) = 0$ and $\hat{u}_j(l;k) = 0$ for all $l \in [k, \ldots, k+H-1]$. Then solve optimal control problem (5.6) for robot j and obtain $u_j^*(l;k)$, where $l \in [k, \ldots, k+H-1]$. Set $u_j^\circ(k) = u_j^*(k;k)$ and apply $u_j^\circ(k)$ to the system.

Controller:

1. At any time instance t_k:
   (a) Measure current state $p_j(k)$ and measure or receive current state $p_i(k)$.
   (b) If $x_j^e(k) \in \mathcal{X}_f$, switch to the terminal controller u_j^T for all k such that $x_j^e(k) \in \mathcal{X}_f$. Set $u_k^\circ(k) = u_j^T(k)$.
   (c) Else, with $\hat{u}_i(l;k)$ and $\hat{u}_j(l;k)$ as initial guess, solve optimal control problem (5.6) for robot j and obtain $u_j^*(l;k)$, where $l \in [k, \ldots, k+H-1]$. Set $u_j^\circ(k) = u_j^*(k;k)$.
2. Over time interval $[t_k, t_{k+1})$:
   (a) Apply $u_j^\circ(k)$ to the system.
   (b) If $x_j^e(k) \in \mathcal{X}_f$, set $\hat{u}_j(l;k+1) = u_j^T(l)$, $l \in [k+1, \ldots, k+H]$.

(c) Else, compute $\hat{u}_j(l;k+1)$ as

$$\begin{cases} u_j^*(l;k) & l \in [k+1,\ldots,\, k+H-1] \\ u_j^T(k+H) & l = k+H \end{cases}$$

(d) Transmit $\hat{u}_j(\cdot;k+1)$ to all robot n that $(v_j,\, v_n) \in \mathcal{E}$ and receive $\hat{u}_i(\cdot;k+1)$.

5.3.3 Terminal Controller

Many formation controllers can be used as the terminal controller. An input-output feedback linearization controller (denoted as Separation Bearing Controller, SBC) developed in [56] is used in our previous paper [131]. *Set points*, are desired distance l_{ij}^d and desired orientation η_{ij}^d relative to the leader. The control law determining $\boldsymbol{u}_j = [v_j\ \omega_j]^T$ based on the position of R_i, which stabilizes the position of R_j relative to R_i, is [56]

$$\begin{aligned} v_j &= s_{ij}\cos\gamma_{ij} - l_{ij}(b_{ij}+\omega_i)\sin\gamma_{ij} + v_i\cos(\theta_i-\theta_j), \\ \omega_j &= \frac{1}{d}\big(s_{ij}\sin\gamma_{ij} + l_{ij}(b_{ij}+\omega_i)\cos\gamma_{ij} + v_i\sin(\theta_i-\theta_j)\big), \end{aligned} \tag{5.9}$$

where

$$\begin{aligned} \gamma_{ij} &= \eta_{ij} + \theta_i - \theta_j, \\ s_{ij} &= k_1(l_{ij}^d - l_{ij}), \\ b_{ij} &= k_2(\eta_{ij}^d - \eta_{ij}), \end{aligned} \tag{5.10}$$

and k_1 and k_2 are positive constants.

Notice that, the SBC controller requires leader's velocity information, which may not be available. To overcome this limitation, a robust formation control law, u_j^R, which stabilizes the formation between robot-pair i and j is proposed in [6]

$$\begin{aligned} u_j^R = g^{-1}(\cdot)\Bigg(& \begin{bmatrix} \frac{1}{l_{ij}}\sin^2\gamma_{ij} \\ \frac{2}{l_{ij}^2}\cos\gamma_{ij}\sin\gamma_{ij} \end{bmatrix} v_j^2 \\ & -2K\dot{e_{ij}} - K^2 e_{ij} - \beta_1\,\mathrm{sgn}(e_{ij})\Bigg), \end{aligned} \tag{5.11}$$

where

$$g(\cdot) = \begin{bmatrix} -\cos\gamma_{ij} & -v_j \sin\gamma_{ij} \\ \frac{1}{l_{ij}}\sin\gamma_{ij} & -\frac{v_j}{l_{ij}}\cos\gamma_{ij} \end{bmatrix} \in \mathbb{R}^{2\times 2},$$

v_j is the linear velocity of robot j, $K = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix}$, $k_{1,2} \in \mathbb{R}_+$, and β_1 are positive constant control gains. Details of this robust formation controller have been explained in Section 4.3.3.

5.4 Stability Results

5.4.1 Dual-mode MPC

Since inside $\mathcal{X}_f$, a stabilizing terminal controller u_j^T is used, when the state enters $\mathcal{X}_f$, the error system will converge to the origin according to the stability properties of controller u_j^T. The stability of the system (5.3) is guaranteed if the state, $x_j^e(k)$, starting from any $x_j^e(0) \in \mathcal{X}\backslash\mathcal{X}_f$, reaches $\mathcal{X}_f$ within finite time under the dual-mode MPC *Algorithm* (see Fig. 5.3).

Assumption 5.1 *For the incremental cost $L(x_j^e, u_j)$, there exists a $\mathcal{K}$-function $\kappa(\cdot)$ such that $L(x_j^e, u_j) \geq \kappa(\|x_j^e\|)$ for all $x_j^e \in \mathcal{X}\backslash\mathcal{X}_f$ and all $u_j \in \mathcal{U}$.*

Assumption 5.2 *For all $x_j^e(k) \in \mathcal{X}\backslash\mathcal{X}_f$, $u^*(k)$ exists.*

Before presenting the main result of this sub-section, we state the following lemma (motivated by [103], [133]) which will be invoked later.

Lemma 5.1 *Suppose assumptions 5.1, 5.2 are satisfied. Then for all $k \in \mathbb{Z}^*$ such that both $x_j^e(k)$ and $x_j^e(k+1)$ are in $\mathcal{X}\backslash\mathcal{X}_f$, under the dual-mode MPC algorithm, the following inequality*

$$J_H^*(k+1) - J_H^*(k) \leq -\kappa(\|x_j^e(k+1)\|) \tag{5.12}$$

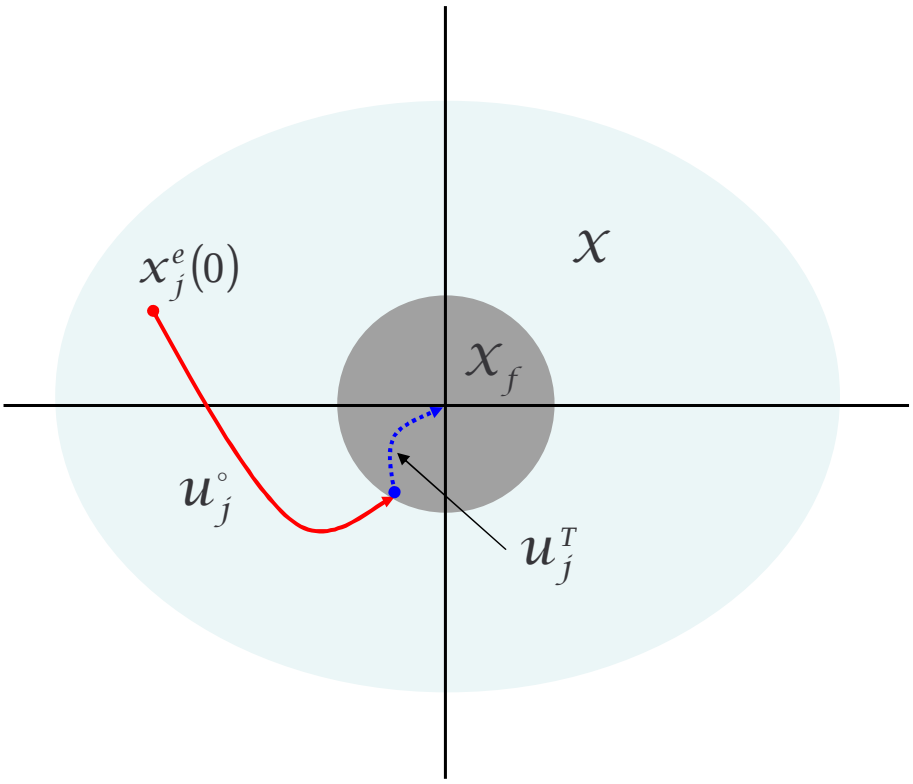


Figure 5.3: State trajectory.

holds.

Proof: By *Assumption* 5.2, $u_j^*(k)$ and $u_j^*(k+1)$ exist, so do the optimal performance index $J_H^*(k)$ and $J_H^*(k+1)$.

To find $J_H^*(k+1)$, the dual-mode MPC algorithm solves the optimal control problem (5.6) from an initial control guess $\hat{u}_j(l;k+1)$, $l \in [k+1,\ldots,k+H]$, which is constructed from the result of previous optimization on time t_k. Obviously, $J_H^*(k+1) \leq \hat{J}_H(x_j^e(\cdot), k+1, \hat{u}_j(l;k+1))$. Therefore,

$$J_H^*(k+1) - J_H^*(k) \leq \hat{J}_H(x_j^e(\cdot), k+1, \hat{u}_j(l;k+1)) - J_H^*(k). \tag{5.13}$$

Follow the construction of $\hat{u}_j(l;k+1)$, which is described in the algorithm given in Section 5.3.2, we have

$$\begin{aligned}
&\hat{J}_H(x_j^e(\cdot), k+1, \hat{u}_j(l;k+1)) - J_H^*(k) = \\
&-L(x_j^{e*}(k+1;k), u_j^*(k;k)) \\
&+L(x_j^e(k+H+1;k), u_j^L(k+H)).
\end{aligned} \tag{5.14}$$

Since $x_j^e(k+H+1;k) \in \mathcal{X}_f$, we have

$$L(x_j^e(k+H+1;k), u_j^L(k+H)) = 0.$$

In addition, assuming the system model is perfect, we have

$$L(x_j^{e*}(k+1;k), u_j^*(k;k)) = L(x_j^e(k+1), u_j^*(k;k)).$$

Therefore,

$$J_H^*(k+1) - J_H^*(k) \leq -L(x_j^e(k+1), u_j^*(k;k)). \tag{5.15}$$

Clearly, with *Assumption* 5.1, inequality (5.12) holds. ■

We now state the main stability result for the proposed dual-mode MPC in the following theorem.

Theorem 5.1 *Let assumptions 5.1 and 5.2 be satisfied. Using the terminal controller (5.11) and the kinematic model (5.1), the dual-mode MPC is asymptotically stabilizing with a region of attraction $\mathcal{X}$.*

Proof: According to the analysis at the beginning of this section, since inside $\mathcal{X}_f$, a stabilizing terminal controller u_j^T is used, when the state enters $\mathcal{X}_f$, the error system will converge to the origin asymptotically. We only need to prove that from any $x_j^e(0) \in \mathcal{X} \backslash \mathcal{X}_f$, under the dual-mode model predictive controller, the state will be driven into $\mathcal{X}_f$ within finite time.

As the definition of $\mathcal{X}_f$, it contains the origin in its interior. There must exist a constant $r > 0$ such that for all $x_j^e(\cdot) \in \mathcal{X} \backslash \mathcal{X}_f$, we have $\|x_j^e(\cdot)\| \geq r$. Then, with the definition of $\mathcal{K}$-function, inequality

$$\kappa(\|x_j^e(\cdot)\|) \geq \kappa(r) \tag{5.16}$$

holds.

Suppose that a finite time t_k does not exist such that $x_j^e(k) \in \mathcal{X}_f$. Because of Lemma 5.1, by adding inequality (5.12) from 0 to k, we have

$$\begin{aligned} J_H^*(k) - J_H^*(0) &\leq -\sum_{n=0}^{k} \kappa(\|x_j^e(n)\|) \\ &\leq -k \min\{\kappa(\|x_j^e(0)\|), \ldots, \kappa(\|x_j^e(k)\|)\}, \end{aligned} \tag{5.17}$$

for all $k \in \mathbb{Z}^*$. Then according to inequality (5.16), we have

$$-k \min\{\kappa(\|x_j^e(0)\|), \ldots, \kappa(\|x_j^e(k)\|)\} \leq -k\kappa(r). \tag{5.18}$$

This means

$$J_H^*(k) - J_H^*(0) \leq -k\kappa(r), \quad \text{for all } k \in \mathbb{Z}^*. \tag{5.19}$$

However, (5.19) implies that $J_H^*(k) \to -\infty$ as $k \to \infty$. This contradicts that $J_H(k) \geq 0$ for all $k \in \mathbb{Z}^*$. Therefore, there exists a time t_k such that $x_j^e(k) \in \mathcal{X}_f$. The stabilizing property of the controller follows. ■

5.4.2 Input-Output Feedback Linearization Controller

Theorem 5.2 *Assume that the lead vehicle's linear velocity along the path $g(t) \in SE(2)$ is lower bounded i.e., $v_i \geq V_{\min} > 0$, its angular velocity is also bounded i.e., $\|\omega_i\| < W_{\max}$, the relative velocity $\delta_v \equiv v_i - v_j$ and relative orientation $\delta_\theta \equiv \theta_i - \theta_j$ are bounded by small positive numbers ε_1, ε_2, and the initial relative orientations $\|\theta_i(t_0) - \theta_j(t_0)\| < c_1\pi$, with $0 < c_1 < 1$. If the control law (5.9) is applied to robot R_j, then the formation is stable, and the system outputs l_{ij}, η_{ij} converge exponentially to the desired values [134].*

Remark 5.1 *Note that, to guarantee stable behavior of R_j, we would require $v_i > 0$. Otherwise, the internal dynamics θ_j of R_j may be unstable. Let the orientation error be expressed as $\dot{e}_\theta = \omega_i - \omega_j$. After incorporating the angular velocity for the follower (5.9), we obtain*

$$\dot{e}_\theta = -\frac{v_i}{d_j} \sin e_\theta + \xi(\omega_i, e_\theta), \tag{5.20}$$

where $\xi(\cdot)$ is a nonvanishing perturbation for the nominal system (equation (5.20) with $\xi(\cdot) = 0$), which is itself (locally) exponentially stable. By using the stability of perturbed systems, it can be shown that system (5.20) is stable when $v_i > 0$. A detailed proof of Theorem 5.2 and explanation of internal dynamics can be found in [134].

5.4.3 Robust Formation Controller

A detailed stability proof can be found in Section 4.3.3.

Remark 5.2 *Notice that, the robust controller proposed here does not require the leader robot's velocity information and there are no internal dynamics. This is a big improvement to the SBC controller described in Section 5.3.3. The only limitation here is that the reference robot's velocity cannot be zero. Otherwise, v_j needs to be zero and the inverse of $g(\cdot)$ cannot be computed. In summary, for the SBC controller, it fails when $v_i \leq 0$. For the robust controller, it only fails when $v_i = 0$. In addition, this robust controller is globally stable, which means that it has a region of attraction $\mathcal{X}$. This alleviates the difficulty of finding a terminal region.*

5.5 Simulation Results

The effectiveness of the control algorithms presented in Section 5.3 is investigated by numerical simulations. In the figures, each robot is depicted by an arrow within a circle. The orientation of the robot is shown by the orientation of the arrow.

5.5.1 Tracking-Stabilizing-Tracking

A realistic scenario is illustrated in Figure 5.4. The reference robot 1 first moves forward from position $(0, 0)$. Then it stops for some time and finally starts to move backward. This scenario happens when some algorithms are implemented for the reference robot 1 to avoid obstacles. Usually this tracking-stabilizing-tracking case is not considered under a single controller approach. To keep the formation, a controller switching is required. However, a simulation shows that this case can be handled within the MPC framework. The desired formation is $F_{12}^d = [-20, -10]$ and it is achieved by the MPC controller. Note, a conventional MPC controller is used here.

5.5.2 Follow a Leader Moving Backward

Figure 5.5 shows the response of robot j following a reference robot i which is moving backward under the robust formation controller. This scenario cannot be handled by the SBC controller since it fails when the leader's velocity is negative. The desired formation is $\bar{F}_{ij}^d = \left[100, \frac{5\pi}{4}\right]$. Robot i starts from position $(0, 0)$ and moves backward with constant speed $v_i = -17.5$. Robot j starts from $(150, 200)$ and moves backward. The formation is achieved by the robust formation controller.

5.5.3 Control of Chain Formation

Simulations of five robots in chain formation under the dual-mode MPC algorithm are presented in this section. The robust formation controller (5.11) is used as the terminal controller.

Total simulation time is 50 seconds. The sample time is set to 0.5 second. Therefore, the total time instance is 100. The prediction horizon is set to $H = 6$. As shown in Figure 5.6, robot 1 moves independently and robots i, $i \in \{2, 3, 4, 5\}$, each follows robot $i - 1$ to form a chain of robots. The control action for robot 1 at different time instance is defined

by

$$u_1(k) = \begin{cases} [20 \quad 0]^T, & k \in [0, \ldots, 10] \\ [20 \quad 0.2]^T, & k \in [11, \ldots, 70] \\ [20 \quad 0]^T, & k \in [71, \ldots, 100] \end{cases}$$

The formations for each robot-pair are $F_{12}^d = F_{23}^d = F_{34}^d = F_{45}^d = [-20 \quad 0]^T$. The initial conditions for each robot are given as

$$\begin{aligned} p_1(0) &= [0 \quad 0 \quad \pi]^T, \\ p_2(0) &= [50 \quad 10 \quad \pi]^T, \\ p_3(0) &= [100 \quad -10 \quad \pi]^T, \\ p_4(0) &= [150 \quad 10 \quad \pi]^T, \\ p_5(0) &= [200 \quad -10 \quad \pi]^T. \end{aligned}$$

The control constrain set is defined as

$$\mathcal{U} = \left\{ [V \quad \omega]^T \in \mathbb{R}^2 : 15 \leq V \leq 50, -0.3 \leq \omega \leq 0.3 \right\}.$$

Figure 5.6 shows the formation response. The linear velocity control inputs for robots $2, 3, 4, 5$ are shown in Figure 5.7 and the angular velocity control inputs are shown in Figure 5.8. Clearly, all the control inputs satisfy the control constraints. As the desired formation is achieved, the linear velocity control inputs converge to 20 m/s and the angular velocity control inputs converge to 0 rad/s, which are the final control inputs for robot 1.

5.5.4 Control of Triangle Formation

A simulation of six robots in triangle formation is presented in this section. The edge set is $\mathcal{E} = \{(V_1, V_2), (V_1, V_3), (V_2, V_4), (V_2, V_5), (V_3, V_6)\}$. Robot 1 moves independently. The total simulation time is 15 seconds. Again, the sample time is set to 0.5 second. Therefore, the total time instance is 30. The prediction horizon is set to $H = 6$. The control action for

robot 1 is $u_1(k) = [20 \quad 0]^T$ for all the k. The *set points* for each robot are

$$F_{12}^d = [-20 \quad 20]^T,$$
$$F_{13}^d = [-20 \quad -10]^T,$$
$$F_{24}^d = [-20 \quad 10]^T,$$
$$F_{35}^d = [-20 \quad -10]^T,$$
$$F_{26}^d = [-20 \quad -10]^T.$$

The control constrain set is defined as $\mathcal{U} = \left\{[V \quad \omega]^T \in \mathbb{R}^2 : 15 \leq V \leq 40, -0.2 \leq \omega \leq 0.2\right\}$. The initial conditions for each robot are given by

$$p_1(0) = [0 \quad 0 \quad \pi]^T,$$
$$p_2(0) = [50 \quad 0 \quad \pi]^T,$$
$$p_3(0) = [70 \quad 5 \quad \pi]^T,$$
$$p_4(0) = [100 \quad -30 \quad \pi]^T,$$
$$p_5(0) = [100 \quad 0 \quad \pi]^T,$$
$$p_6(0) = [100 \quad 20 \quad \pi]^T.$$

The formation response is shown in Figure 5.9. The linear velocity control inputs for robots $2, 3, 4, 5, 6$ are shown in Figure 5.10 and the angular velocity control inputs are shown in Figure 5.11. Clearly, all the control inputs satisfy the control constraints. As the desired formation is achieved, the linear velocity control inputs converge to 20 m/s and the angular velocity control inputs converge to 0 rad/s, which are the control inputs for robot 1.

5.6 Summary

In this chapter, a dual-mode MPC algorithm that allows a team of mobile robots to navigate in formation is developed and proven to be stable. Simulations show the effectiveness of the proposed dual-mode MPC algorithm. Additionally, we show that it is more convenient to put the tracking and point stabilizing problems of nonholonomic robots inside the MPC

framework. For the choice of stabilizing terminal controller, analysis show that the robust formation controller is better than the SBC controller.

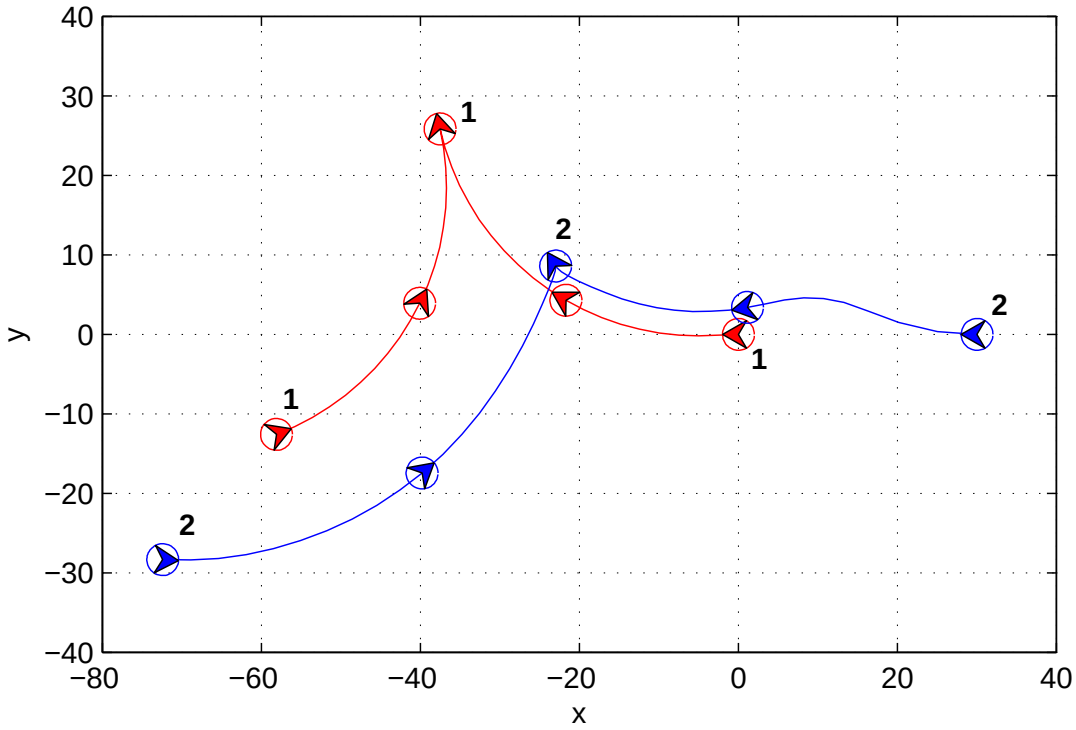


Figure 5.4: Trajectory of a robot following a reference vehicle which moves forward, stops, and then moves backward according to an MPC controller.

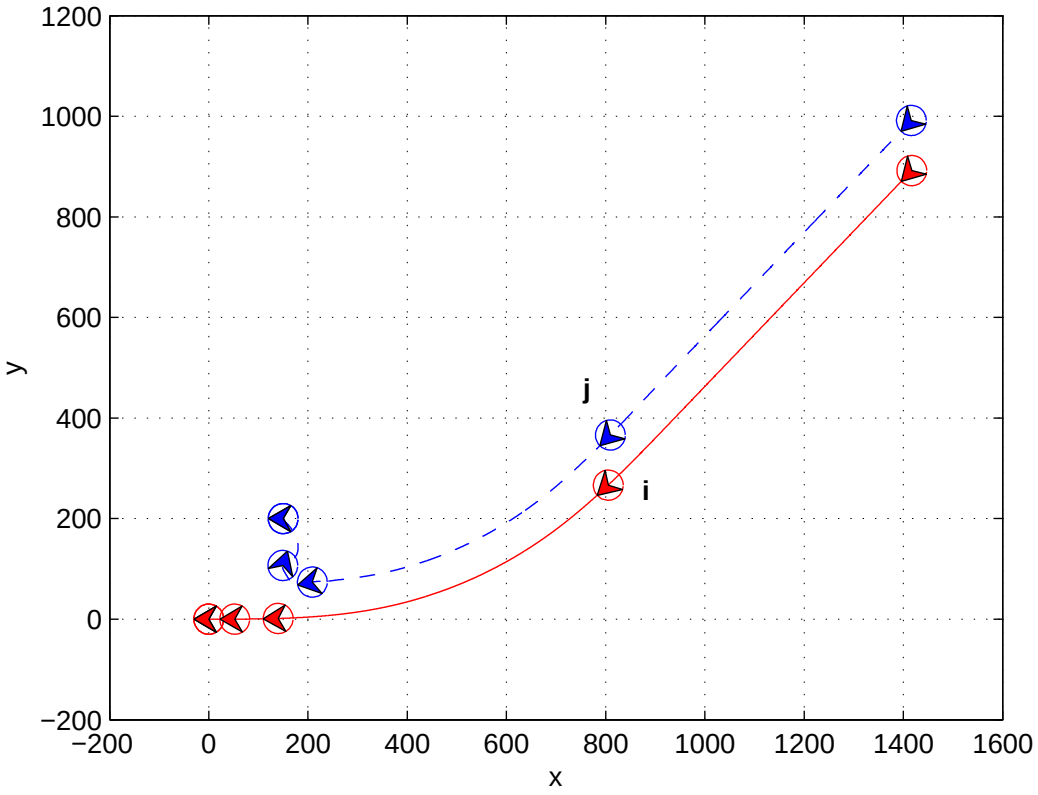


Figure 5.5: Trajectory of a robot following a reference robot which is moving backward according to the robust formation controller.

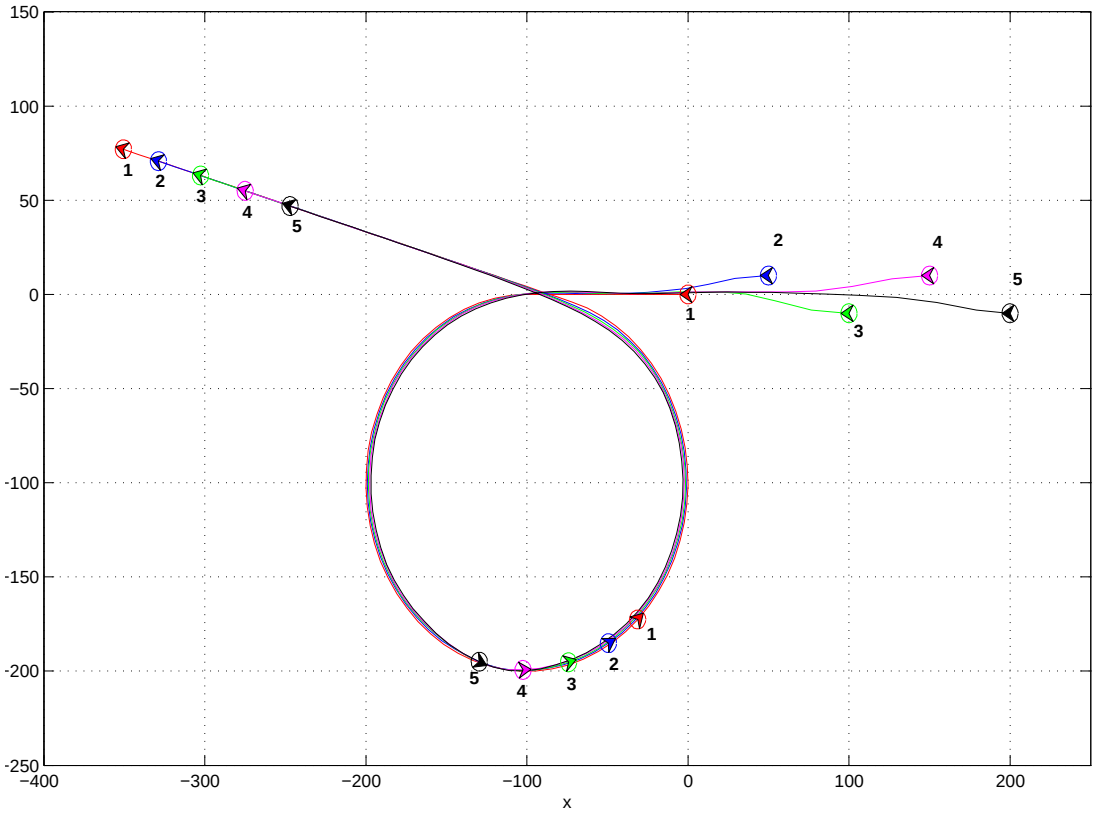


Figure 5.6: Five robots in chain formation according to a dual-mode MPC with robust formation controller as the terminal controller.

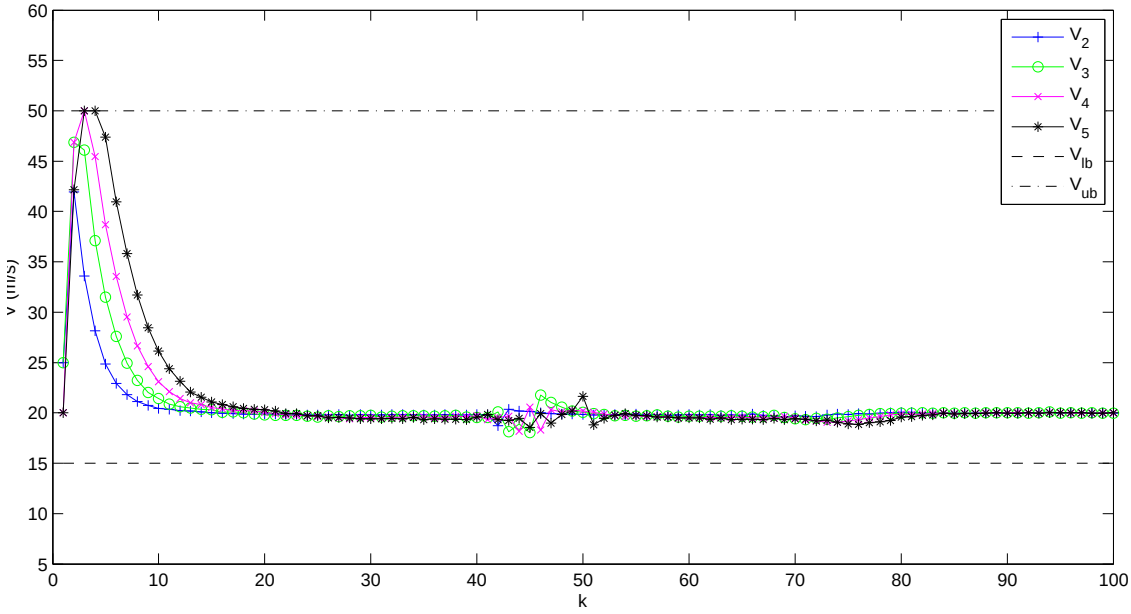


Figure 5.7: Linear velocity control inputs of chain formation.

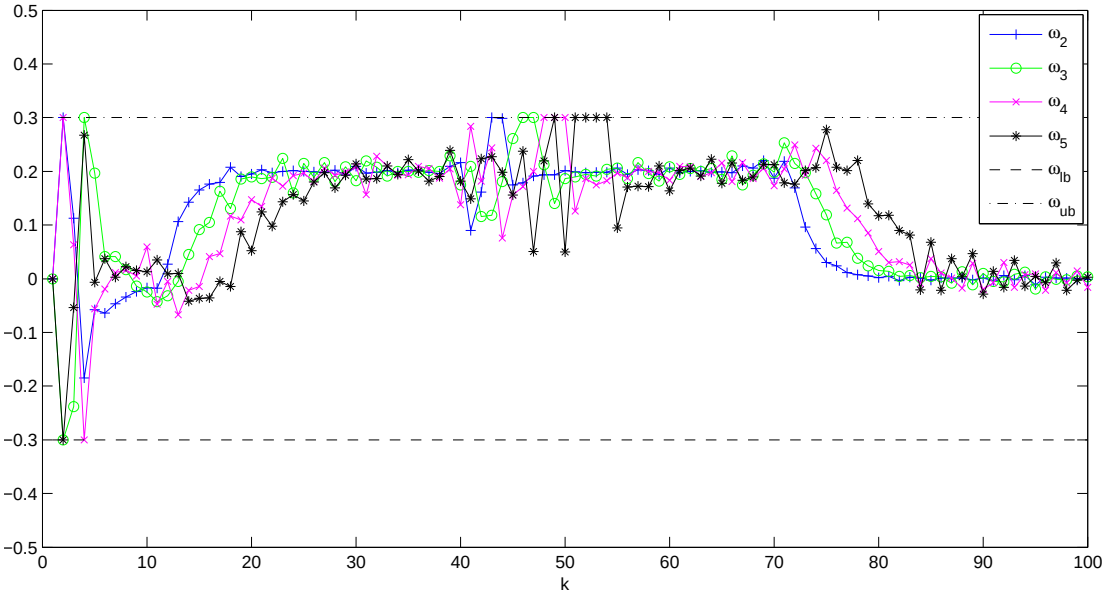


Figure 5.8: Angular velocity control inputs of chain formation.

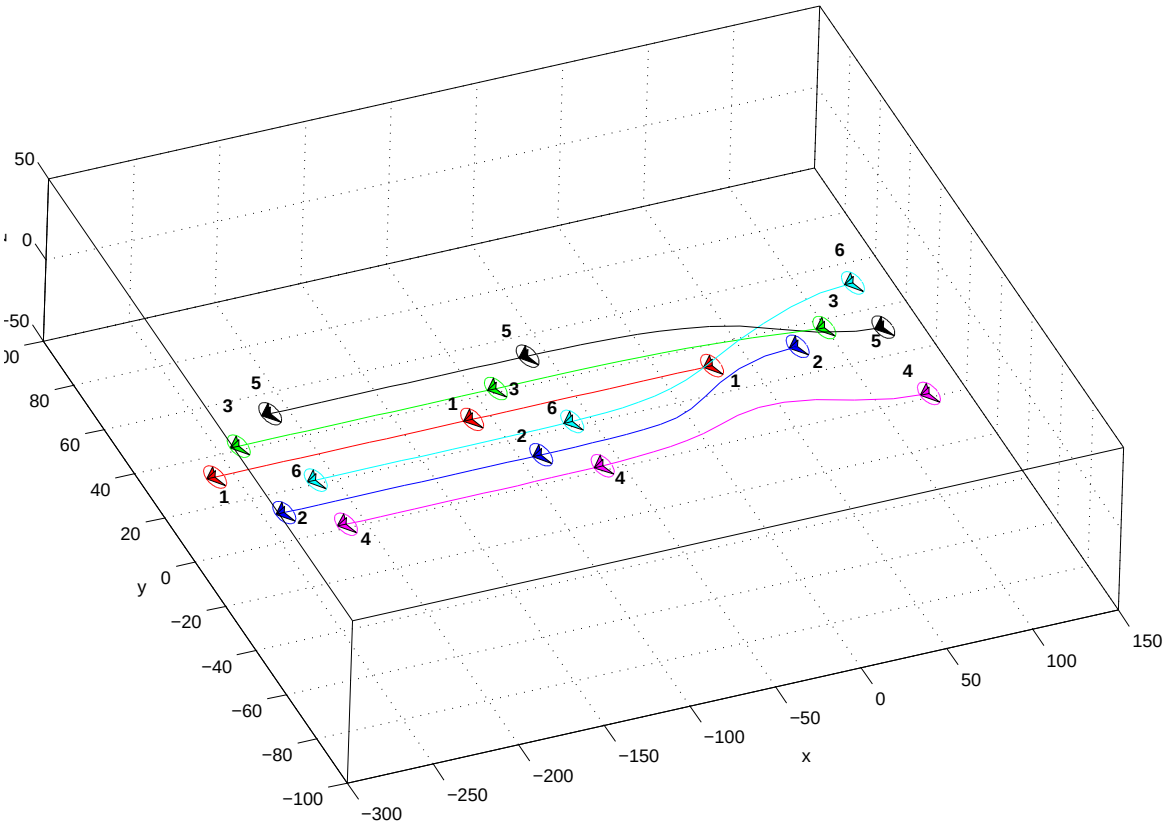


Figure 5.9: Six robots in triangular formation according to a dual-mode MPC with robust formation controller as the terminal controller.

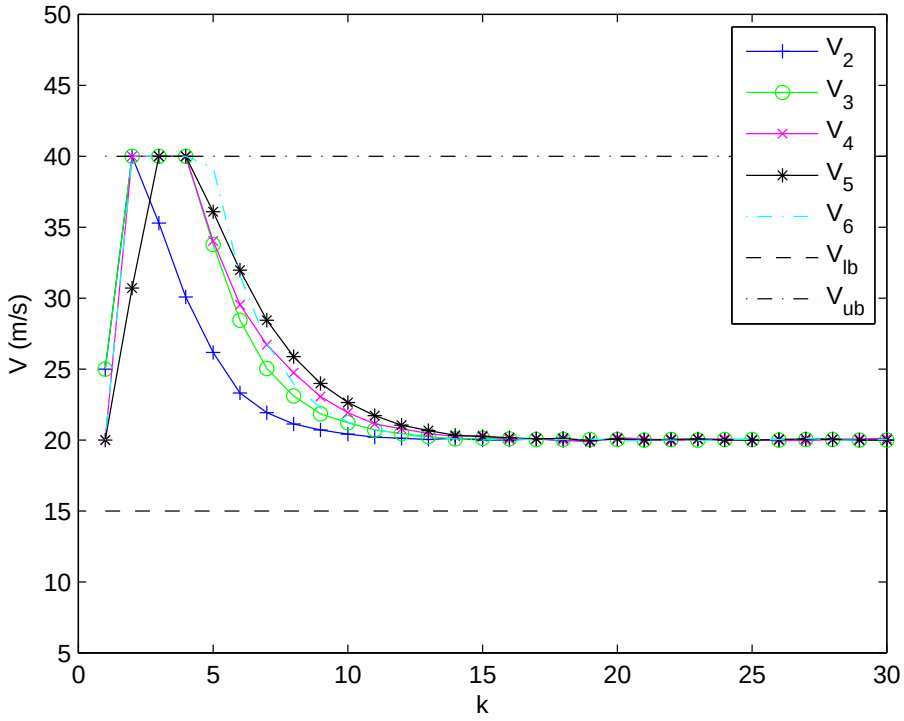


Figure 5.10: Linear velocity control inputs of triangle formation.

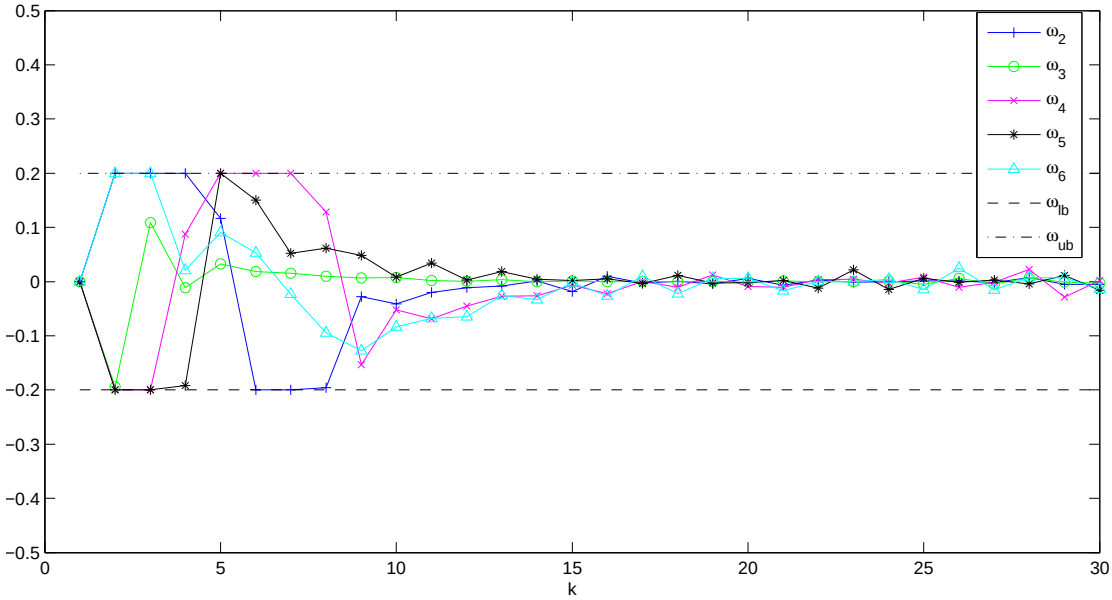


Figure 5.11: Angular velocity control inputs triangle formation.

CHAPTER 6

Fisrt-State Contractive Model Predictive Control of Nonholonomic Mobile Robots

6.1 Introduction

A brief introduction of nonholonomic mobile robots is given in Chapter 2, where a short literature review about the control of nonholonomic mobile robots can also be found. Recently, model predictive control has gained more and more attention in the control community. The inherent ability of MPC to handle constrained systems makes it a promising technique for the control of nonholonomic mobile robots.

In this chapter, we proposed a novel MPC approach for the control of nonholonomic mobile robots. From the literature, most stabilizing MPC methods address stability by adding terminal state penalties in the performance index and/or imposing constraints on the terminal state at the end of the prediction horizon. However, the proposed MPC algorithm guarantees its stability by adding a contractive constraint on the first state at the beginning of the prediction horizon. More specifically, the contributions of this chapter are threefold: (i) The exponential stability of our MPC controller is guaranteed by adding a

first-state contractive constraint. This means that the convergence is faster and no terminal region calculation is required; (ii) tracking a trajectory moving backward is no longer a problem under our MPC controller and (iii), the proposed MPC controller has simultaneous tracking and point stabilization capability, in contrast to most of the existing controllers in the literature.

The rest of the chapter is organized as follows. Section 6.2 introduces the robot kinematic model and the trajectory tracking and point stabilization problems of a nonholonomic mobile robot. A *first-state contractive* MPC algorithm is proposed in Section 6.3. Stability results of the proposed algorithm are found in Section 6.4. In Section 6.5, simulation results are provided to show the effectiveness of the method. Finally, summary is given in Section 6.6.

6.2 Preliminaries

This chapter deals with the problem of designing control laws for the motion control of nonholonomic mobile robots. In this section, a brief introduction of the two fundamental classes of problems, trajectory tracking and point stabilization, are given.

6.2.1 Kinematic Model

Consider the planar motion of mobile robots under the nonholonomic constraint of *pure rolling and non-slipping*, the kinematic model has been given in Section 2.1.1. To assist the reader, the model is listed again as follows

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{6.1}$$

where $(x, y) \in \mathbb{R}^2$ denotes the position of the robot in a Cartesian coordinate frame, $\theta \in (-\pi, \pi]$ represents the orientation of the robot with respect to the positive X axis, and $v \in \mathcal{V} \subseteq \mathbb{R}$ and $\omega \in \mathcal{W} \subseteq \mathbb{R}$ are the control inputs representing linear and angular velocities, respectively.

6.2.2 Trajectory Tracking

Let a triplet $\mathbf{z}_c = [x \;\; y \;\; \theta]^T$ describe the position and the orientation of a mobile robot. The reference trajectories can be described by a virtual reference robot with a state vector $\mathbf{z}_r = [x_r \;\; y_r \;\; \theta_r]^T$, an input vector $\mathbf{u}_r = [v_r \;\; \omega_r]^T$ and the kinematic model (see Figure 6.1)

$$\dot{\mathbf{z}}_r = \begin{bmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{bmatrix} = \begin{bmatrix} \cos\theta_r & 0 \\ \sin\theta_r & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_r. \tag{6.2}$$

Then the trajectory tracking problem can be defined [10].

Definition 6.1 *The trajectory tracking problem, under the assumption that the virtual reference robot is not at rest ($v_r = \omega_r = 0$) when $t \to +\infty$, is to find a feedback control law $\mathbf{u} = [v \;\; \omega]^T$, such that*

$$\lim_{t \to \infty} (\mathbf{z}_r - \mathbf{z}_c) = 0,$$

with any initial robot posture $\mathbf{z}_c(0)$.

By transforming the reference state $\mathbf{z}_r$ in a local coordinate system attached to the tracking robot, an error state $\mathbf{z}_e$ can be defined [16]

$$\mathbf{z}_e := \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} (\mathbf{z}_r - \mathbf{z}_c). \tag{6.3}$$

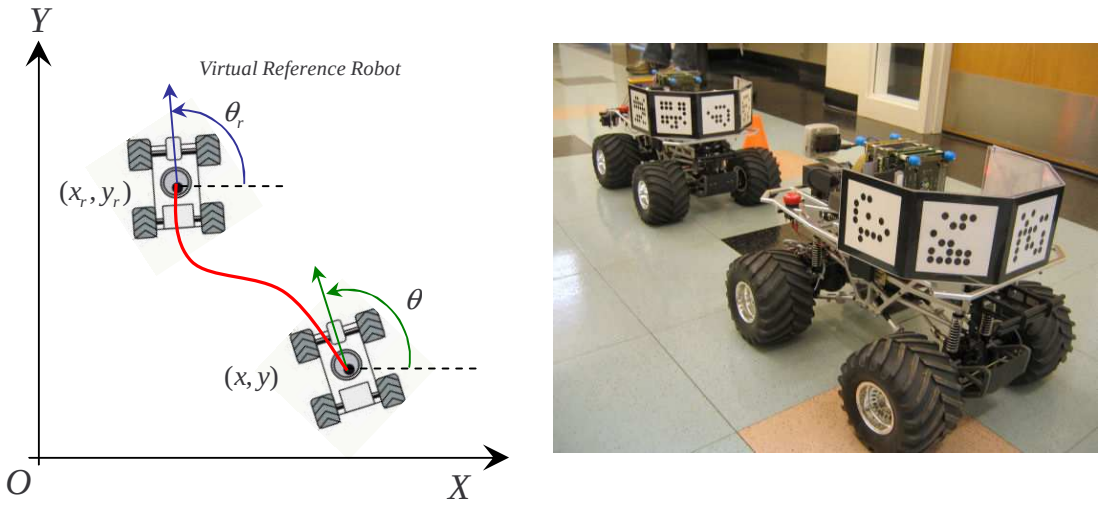


Figure 6.1: Trajectory tracking.

Taking the derivative of (6.3) and rearranging with (6.1), (6.2), the error model becomes

$$\begin{aligned} \dot{x}_e &= \omega y_e - v + v_r \cos\theta_e, \\ \dot{y}_e &= -\omega x_e + v_r \sin\theta_e, \\ \dot{\theta}_e &= \omega_r - \omega. \end{aligned} \tag{6.4}$$

Let us define $\mathbf{u}_e$,

$$\mathbf{u}_e := \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} -v + v_r \cos\theta_e \\ \omega_r - \omega \end{bmatrix}, \tag{6.5}$$

then the error model (6.4) can be rewritten as follows

$$\dot{\mathbf{z}}_e = \begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{z}_e + \begin{bmatrix} 0 \\ v_r \sin\theta_e \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_e. \tag{6.6}$$

By linearizing system (6.6) about the equilibrium point ($\mathbf{z}_e = 0$, $\mathbf{u}_e = 0$), we obtain

$$\dot{\mathbf{z}}_e = \begin{bmatrix} 0 & \omega_r & 0 \\ -\omega_r & 0 & v_r \\ 0 & 0 & 0 \end{bmatrix} \mathbf{z}_e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{u}_e. \tag{6.7}$$

The controllability of system (6.7) can be easily checked. However, when the virtual reference robot stops ($v_r = \omega_r = 0$), the controllable property is lost.

6.2.3 Point Stabilization

For the point stabilization problem, one can have the following definition.

Definition 6.2 *Given an arbitrary constant reference position and orientation* $\mathbf{z}_d = [x_d \;\; y_d \;\; \theta_d]^T$, *the point stabilization problem is to find a feedback control law* $\mathbf{u} = [v \;\; \omega]^T$, *such that*

$$\lim_{t\to\infty} (\mathbf{z}_d - \mathbf{z}_c) = 0,$$

with any initial robot posture $z_c(0)$.

Without loss of generality, we use $\mathbf{z}_d = [0 \;\; 0 \;\; 0]^T$ as the constant reference posture (since by coordinate transforming, any arbitrary posture can be transformed to $[0 \;\; 0 \;\; 0]^T$). Then the problem becomes to find a feedback control law which drives the system (6.1) back to the origin aligning with the X axis.

It is a well-known result that a smooth time-invariant feedback control law does not exist for the point stabilization problem [5]. However, with the analysis in Section 6.2.1, system (6.1) is still controllable. The price is that non-smooth or time-variant control laws have to be applied.

Consider a Cartesian to polar coordinate transformation [48] (see Figure 6.2), a polar state $\mathbf{z}_q = [l \;\; \phi \;\; \alpha]^T$ can be defined

$$\begin{aligned} l &= \sqrt{x^2 + y^2} \\ \phi &= \arctan2(-y, -x) \\ \alpha &= \phi - \theta \end{aligned} \tag{6.8}$$

Then the kinematic model (6.1) becomes

$$\begin{aligned} \dot{l} &= -v\cos\alpha \\ \dot{\phi} &= \frac{v\sin\alpha}{l} \\ \dot{\alpha} &= -\omega + \frac{v\sin\alpha}{l} \end{aligned} \tag{6.9}$$

Note, when $l = 0$, which means that robot reaches the origin, the new kinematic model is not defined.

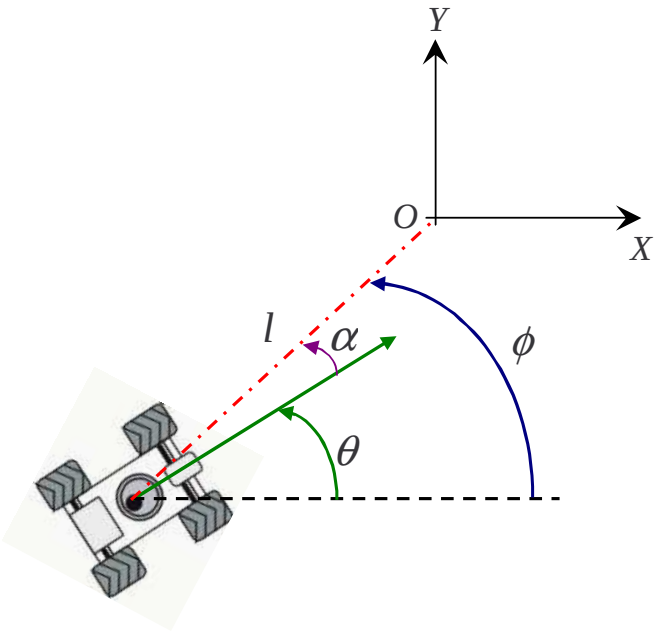


Figure 6.2: Coordinate transformation.

6.3 First-State Contractive MPC

Without considering disturbances and model uncertainties, systems like (6.4) and (6.9) can be generally expressed by the following nonlinear set of differential equations

$$\dot{\mathbf{z}}(t) = \mathbf{h}(\mathbf{z}(t), \mathbf{u}(t)), \quad \mathbf{z}(0) = \mathbf{z}_0, \tag{6.10}$$

with a state vector $\mathbf{z}(t) \in \mathbb{R}^m$ and an input vector $\mathbf{u}(t) \in \mathbb{R}^n$, $m, n \in \mathbb{N}$. Function $\mathbf{h} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^m$ is assumed to be continuous.

Since usually the control system is implemented on a computer in discrete time, (6.10) can be converted into the following set of difference equations

$$\mathbf{z}(k+1) = \mathbf{f}(\mathbf{z}(k), \mathbf{u}(k)), \quad \mathbf{z}(0) = \mathbf{z}_0, \tag{6.11}$$

with a state vector $\mathbf{z}(k) \in \mathcal{Z}$ and an input vector $\mathbf{u}(k) \in \mathcal{U}$, $k \in \mathbb{Z}^*$. $\mathcal{Z} \subset \mathbb{R}^m$ is the state constraints which contains the origin in its interior. $\mathcal{U} \subset \mathbb{R}^n$ is the input constraints which is a compact subset of $\mathbb{R}^n$ containing the origin in its interior. Usually, we have $\mathcal{U} = \{\mathbf{u} \in \mathbb{R}^n : \mathbf{u}_{\min} \leq \mathbf{u} \leq \mathbf{u}_{\max}\}$. $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$ are known constants in $\mathbb{R}^n$. Function

$\mathbf{f} : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^m$ is assumed to be continuous.

The control goal is to find $\mathbf{u}(k)$ which drives the system (6.11) toward the equilibrium ($\mathbf{z}(k) = 0$ and $\mathbf{u}(k) = 0$).

To obtain the current control $\mathbf{u}(k)$ at time t_k, where k is a nonnegative integer ($k \in \mathbb{Z}^*$), a finite-horizon optimal control problem

$$\begin{aligned} \min_{\mathbf{u}} \quad & J_H(\mathbf{z}, k, \mathbf{u}), \\ \text{subject to:} \quad & \mathbf{z}(k+1) = \mathbf{f}(\mathbf{z}(k), \mathbf{u}(k)), \\ & \mathbf{z}(k) \in \mathcal{Z}, \\ & \mathbf{u}(k) \in \mathcal{U}, \end{aligned} \tag{6.12}$$

must be solved online for an MPC algorithm. The performance index $J_H(\mathbf{z}, k, \mathbf{u})$ is defined as

$$J_H(\mathbf{z}, k, \mathbf{u}) := \sum_{i=1}^{H} L(\mathbf{z}(k+i; k), \mathbf{u}(k+i-1; k)), \tag{6.13}$$

where $H \in \mathbb{N}$ is the horizon length (for simplicity, the prediction horizon equals the control horizon in this paper). The incremental cost is defined as

$$L(\mathbf{z}, \mathbf{u}) := \|\mathbf{z}\|_Q^2 + \|\mathbf{u}\|_R^2, \tag{6.14}$$

where $\|\mathbf{z}\|_Q$ and $\|\mathbf{u}\|_R$ denote the weighted 2-norm, which are defined as $\|\mathbf{z}\|_Q^2 := \mathbf{z}^T Q \mathbf{z}$ and $\|\mathbf{u}\|_R^2 := \mathbf{u}^T R \mathbf{u}$. Q and R are positive definite symmetric matrices of appropriate dimensions.

Since a finite horizon is used, the controller found in (6.12) is not guaranteed to be stable. Many researchers have contributed to the stability of nonlinear MPC with some important methods. In [102], authors propose that by adding a *terminal state equality constraint* $\mathbf{z}(k+H) = 0$ to the optimal control problem, the stability can be guaranteed. However, strong assumptions of the optimal control problem are required and the computational load is huge in the nonlinear case. In order to overcome those difficulties, a *terminal state inequality constraint* $\mathbf{z}(k+H) \in \Omega$ can take the place of *terminal state equality constraint*

[103]. Ω, a subset of $\mathbb{R}^m$ containing the origin in its interior, is called *terminal region*. A *terminal controller* is switched in if the states is inside the *terminal region*. This method is denoted as *dual-mode* MPC. A similar approach, *quasi-infinite* MPC, is proposed in [31]. In addition to the *terminal inequality constraint*, a terminal penalty $g(\mathbf{z}(k+H))$ is added to the performance index to assure the stability. A *terminal controller* is still required, but it is never applied to the system in the *quasi-infinite* MPC scheme. Though the computational load can be reduced, the difficulty of *dual-mode* MPC and *quasi-infinite* MPC algorithms lies in calculating the *terminal region* (upper bounded by the *region of attraction* of the *terminal controller*). An MPC scheme without *terminal region* requirements is proposed by authors in [105]. A *terminal contractive constraint* $g(\mathbf{z}(k+H)) \leq \rho g(\mathbf{z}(k))$, where $\rho \in (0,\ 1)$ is called *contractive parameter*, takes the place of *terminal inequality constraint*. Global exponential stability of the closed-loop system can be guaranteed. See [89] for a more detailed discussion on stability of MPC.

To achieve stability, the core idea behind the methods mentioned above is to add terminal state penalties in the performance index and impose constraints on the terminal state at the end of the prediction horizon. Therefore, those methods can be denoted as *terminal-state constrained* MPC (TSC-MPC). However, in the implementation of most MPC schemes, only the first control of the control sequence yield by optimization is applied to the plant at each sampling instance. All the other controls are discarded. Only the first state at the beginning of the prediction horizon is directly affected by this implementation.

Motivated by this observation and the *contractive* MPC scheme developed in [105], a new MPC algorithm is proposed here. To be specific, we obtain the current control $\mathbf{u}(k)$ at

time t_k by solving the following finite-horizon optimal control problem online

$$\begin{aligned} \min_{\mathbf{u}} \quad & J_H(\mathbf{z}, k, \mathbf{u}), \\ \text{subject to:} \quad & \mathbf{z}(k+1) = \mathbf{f}(\mathbf{z}(k), \mathbf{u}(k)), \\ & \mathbf{z}(k) \in \mathcal{Z}, \\ & \mathbf{u}(k) \in \mathcal{U}, \\ & \|\mathbf{z}(k+1)\|_{\hat{P}} \leq \rho \|\mathbf{z}(k)\|_{\hat{P}}, \end{aligned} \tag{6.15}$$

where $J_H(\mathbf{z}, k, \mathbf{u})$ is defined in (6.13) and (6.14). $\hat{P}$ is a positive definite symmetric matrix and $\rho \in (0,\ 1)$.

Note, the last inequality constraint in (6.15) can be called *first-state contractive* constraint. This means the first state at the beginning of the prediction horizon, $\mathbf{z}(k+1)$ is contracted in norm with respect to the current state, $\mathbf{z}(k)$. Therefore, the proposed MPC algorithm can be denoted as *first-state contractive* MPC (FSC-MPC). The user adjustable parameter ρ is called *contractive parameter*, which addresses the contraction rate.

The FSC-MPC controller can be implemented as follows.

FSC-MPC Algorithm

Data: prediction horizon $H \in \mathbb{N}$; sampling time $\delta_T \in \mathbb{R}^+$; weights Q, R, $\hat{P} > 0$; constraints $\mathbf{u}_{\min}$, $\mathbf{u}_{\max} \in \mathbb{R}^m$; contractive parameter $\rho \in (0,\ 1)$; $k \in \mathbb{Z}^*$.

Step 0: set $k = 0$; set initial control prediction $\hat{\mathbf{u}}(i; k) = 0$, $i \in [1, \ldots, H-1]$.

Step 1: measure the states $\mathbf{z}(k)$ at time t_k; with control prediction $\hat{\mathbf{u}}(i; k)$, solve the optimal control problem (6.15) and obtain a control sequence $\mathbf{u}^*(i; k)$.

Step 2: apply the first control $\mathbf{u}^*(1; k)$ in the control sequence $\mathbf{u}^*(i; k)$ to system (6.11) for the time interval $[t_k,\ t_{k+1}]$, where $t_{k+1} = t_k + \delta_T$.

Step 3: update the control prediction as follows

$$\hat{\mathbf{u}}(i; k) = \begin{cases} \mathbf{u}^*(i+1; k) & i \in [1, \ldots, H-2] \\ \mathbf{u}^*(i; k) & i = H-1 \end{cases};$$

set $k = k + 1$; go back to **Step 1**.

Note, in **Step 1**, an assumption is that for all $k \in \mathbb{Z}^*$, a feasible solution of the optimal control problem (6.15), satisfying all the constraints, always exists. However, global optimal solution is not strictly required here. Any feasible or local optimal solutions is acceptable. This approach might compromise the performance, but the stability property of the algorithm will not be affected.

6.4 Stability Results

In this section, the stability of FSC-MPC algorithm will be proven. Before we give the mean results, let us make the following assumptions.

Assumption 6.1 *There exists a constant $\beta \in (0,\ \infty)$ such that for all $\mathbf{z}(k) \in B_\beta := \{\mathbf{z} \in \mathcal{Z} | \|\mathbf{z}\|_{\hat{P}} \leq \beta\}$, a contractive parameter $\rho \in (0,\ 1)$ can be found so that at time t_k, a feasible solution of the optimal control problem (6.15), satisfying all the constraints, always exists for all $k \in \mathbb{Z}^*$.*

Note, Assumption 6.1 means that if the optimal control problem is feasible at time t_0, then all the subsequent optimal control problems are feasible. Since the optimal control problem is feasible at time t_0, state $\mathbf{z}(0) \in B_\beta$. By solving the optimal control problem, we have $\|\mathbf{z}(1)\|_{\hat{P}} \leq \rho \|\mathbf{z}(0)\|_{\hat{P}}$, which means $\mathbf{z}(1) \in B_{\rho\beta} \subset B_\beta$. Therefore, according to Assumption 6.1, the optimal control problem is again feasible at time t_1. By repeating this, we can have $\mathbf{z}(k) \in B_{\rho^k\beta} \subset B_\beta$. This concludes that the optimal control problem is feasible for all $k \in \mathbb{Z}^*$.

Assumption 6.2 *For all $t \in [t_k,\ t_{k+1}]$, $k \in \mathbb{Z}^*$, there exists a constant $\kappa \in (0,\ \infty)$, such that the transient state, $\mathbf{z}(t)$, satisfies $\|\mathbf{z}(t)\|_{\hat{P}} \leq \kappa \|\mathbf{z}(k)\|_{\hat{P}}$.*

Note, Assumption 6.2 means that systems with finite escape time are not under consideration. Since $\mathbf{u}$ is constrained, this assumption is always satisfied.

Theorem 6.1 *Suppose that the optimal control problem is feasible at time t_0 and Assumptions 6.1 and 6.2 are satisfied. The FSC-MPC algorithm described in Section 6.3 for system (6.10), (6.11) is exponentially stable in the sense that the state trajectory of the closed-loop system satisfies the following inequality*

$$\|\mathbf{z}(t)\|_{\hat{P}} \leq \kappa \|\mathbf{z}(0)\|_{\hat{P}} e^{-\frac{(1-\rho)}{\delta_T}(t-t_0)}, \tag{6.16}$$

where δ_T is the sampling time.

Proof: Since the optimal control problem is feasible at time t_0, from Assumption 6.1, the optimal control problem is feasible at time t_k, $k \in \mathbb{Z}^*$. Therefore, we have

$$\|\mathbf{z}(k)\|_{\hat{P}} \leq \rho \|\mathbf{z}(k-1)\|_{\hat{P}} \leq \cdots \leq \rho^k \|\mathbf{z}(0)\|_{\hat{P}}. \tag{6.17}$$

Now with Assumption 6.2 and (6.17), $\mathbf{z}(t)$ satisfies the following inequality

$$\|\mathbf{z}(t)\|_{\hat{P}} \leq \kappa \rho^k \|\mathbf{z}(0)\|_{\hat{P}}, \tag{6.18}$$

where $t \in [t_k,\ t_{k+1}]$, for all $k \in \mathbb{Z}^*$.

Since $\rho \in (0,\ 1)$, we have $e^{(\rho-1)} - \rho \geq 0$, which means $e^{(\rho-1)k} \geq \rho^k \geq 0$, for all $k \in \mathbb{Z}^*$. Inequality (6.18) can be rewritten as follows

$$\|\mathbf{z}(t)\|_{\hat{P}} \leq \kappa \|\mathbf{z}(0)\|_{\hat{P}} e^{-(1-\rho)k}. \tag{6.19}$$

Since $k = (t_k - t_0)/\delta_T$ and $(t - t_0)/\delta_T \leq (t_k - t_0)/\delta_T = k$, for all $t \in [t_0,\ t_k]$, we have

$$e^{-(1-\rho)k} \leq e^{-\frac{(1-\rho)}{\delta_T}(t-t_0)} \tag{6.20}$$

Therefore, from inequalities (6.19) and (6.20), we conclude

$$\|\mathbf{z}(t)\|_{\hat{P}} \leq \kappa \|\mathbf{z}(0)\|_{\hat{P}} e^{-\frac{(1-\rho)}{\delta_T}(t-t_0)}.$$

According to [135], the closed-loop system is exponentially stable, so does the FSC-MPC algorithm. ■

6.5 Simulation Results

The effectiveness of the FSC-MPC algorithm presented in Section 6.3 is investigated by numerical simulations. In the figures, each robot is depicted by an arrow within a circle (dotted circle for virtual reference robot). The orientation of the robot is shown by the orientation of the arrow.

The units used in the simulations are, for position, meter (m), for orientation, radian (rad), for linear velocity, meter/second (m/s), and for angular velocity, radian/second (rad/s).

6.5.1 Trajectory Tracking

In this section, the simulation results of our FSC-MPC controller, Kanayama's controller proposed in [16] and Samson's controller proposed in [10] are compared. Specifically, the controllers proposed in [16] and [10] are

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_r \cos\theta_e + K_x x_e \\ \omega_r + v_r(K_y y_e + K_\theta \sin\theta_e) \end{bmatrix}, \tag{6.21}$$

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_r \cos\theta_e + K_1 x_e \\ \omega_r + K_2 v_r \frac{\sin\theta_e}{\theta_e} + K_3 \theta_e \end{bmatrix}, \tag{6.22}$$

respectively.

The reference trajectory starts from posture $\mathbf{z}_r(0) = [0 \;\; 0 \;\; 0]^T$ with constant control inputs $[v_r \;\; \omega_r]^T = [1 \;\; 0]^T$. In addition, we assume that two perturbations occur at time 10 s and 20 s which change the orientation of the virtual reference robot from 0 rad to $\pi/2$ rad and from $\pi/2$ rad back to 0 rad, respectively. Total simulation time is 30 s.

The initial condition of tracking robot is $\mathbf{z}_c(0) = [0 \;\; 3 \;\; 0]^T$. Controller parameters are selected as follows. For Kanayama's controller, $K_x = 1$, $K_y = 4$ and $K_\theta = 4$. For Samson's controller, $K_1 = 1$, $K_2 = 4$ and $K_3 = 4$. Sampling time for these two controllers

is $\delta_T = 0.1$ s. For FSC-MPC controller, $H = 6$, $\rho = 0.95$,

$$Q = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \hat{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Sampling time is $\delta_T = 0.5$ s. Control input constraints are

$$-4(\text{m/s}) \leq v \leq 4(\text{m/s}), \quad -0.8(\text{rad/s}) \leq \omega \leq 0.8(\text{rad/s}).$$

The system responses of the three controllers are shown in Figure 6.3. Control inputs and errors of each controller are illustrated in Figure 6.4 and 6.5. The FSC-MPC controller successfully stabilizes all the tracking errors to 0. Noticing the high peaks on control inputs of Kanayama's controller and Samson's controller in Figure 6.4, the FSC-MPC controller exhibits a comparable performance while requires much less control energy.

Specifically, we use the integral of norm squared actual control inputs ($\sum_1^k \|\mathbf{u}\|^2 \delta_T$) as a metric to evaluate the control energy. The result is shown in Table 6.1.

Kanayama's Controller	596.1606
Samson's Controller	621.0847
FSC-MPC Controller	87.5021

Table 6.1: The integral of norm squared actual control inputs for tracking.

Note, in Figure 6.5 the x_θ control errors of Kanayama and Samson controllers are much larger than that of our FSC-MPC controller during the time $t = [0,\ 5]$. The reason is that with larger angular velocity inputs of Kanayama and Samson controllers during that time (see Figure 6.4), the robot rotates and points to the reference robot much faster than the robot under our FSC-MPC controller (see Figure 6.3). Since we express the tracking error in a local frame work attached to the tracking vehicle (see equation 6.3), the error is related to orientation of the tracking robot.

6.5.2 Point Stabilization

In this section, the simulation results of our FSC-MPC controller and Aicardi's controller proposed in [48] are compared. Specifically, the controllers proposed in [48] is

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} K_1 e \cos\alpha \\ k_2\alpha + K_1 \frac{\cos\alpha \sin\alpha}{\alpha}(\alpha + K_3\phi) \end{bmatrix} \tag{6.23}$$

Three initial robot postures are used in the simulation. They are

$$\mathbf{z}_c(0) = \left\{ \begin{bmatrix} 1 \\ 0 \\ \pi/2 \end{bmatrix}, \begin{bmatrix} -0.5 \\ 0.867 \\ \pi/2 \end{bmatrix}, \begin{bmatrix} -0.5 \\ -0.867 \\ \pi/2 \end{bmatrix} \right\}.$$

The final posture is $\mathbf{z}_d = [0 \quad 0 \quad 0]^T$. Controller parameters are selected as follows. For Aicardi's controller, $K_1 = 3$, $K_2 = 6$ and $K_3 = 1$. Sampling time is $\delta_T = 0.05$ s and the simulation lasts 4 s. For FSC-MPC controller, controller parameters, sampling time and control input constraints are the same as those in Section 6.5.1. Total simulation time is 10 s.

The trajectories generated by Aicardi's controller and our FSC-MPC controller from different initial postures are shown in Figure 6.6. Control inputs and errors for different initial postures are illustrated in Figures 6.7-6.12. The FSC-MPC controller successfully stabilizes the robot at the desired final posture.

The control energy expended by each controller from different initial postures are shown in Table 6.2. The FSC-MPC controller requires much less control energy in comparison with the Aicardi's controller.

6.5.3 Simultaneous Tracking and Stabilization

Two simulations are illustrated in this section which show that our FSC-MPC controller has the ability of simultaneous tracking and stabilization. Usually, simultaneous tracking and stabilization is not considered under a single controller approach. Most of the existing

Initial Posture	Aicardi's Controller	FSC-MPC Controller
$[1 \;\; 0 \;\; \pi/2]^T$	1772.7122	10.4999
$[-0.5 \;\; 0.867 \;\; \pi/2]^T$	586.2641	6.1507
$[-0.5 \;\; -0.867 \;\; \pi/2]^T$	69.7075	3.7015

Table 6.2: The integral of norm squared actual control inputs for stabilization.

controllers for trajectory tracking of nonholonomic mobile robots will fail when the virtual reference robot stops or moves backward. However, in realistic scenarios, we do need to handle those situations.

In Case 1, the virtual reference robot starts moving backward from posture $\mathbf{z}_r(0) = [0 \;\; 0 \;\; \pi/2]^T$ with constant control inputs $[v_r \;\; \omega_r]^T = [-1 \;\; 0.1]^T$. Then, it stops at time $t = 5$ s. The initial condition of the tracking robot is $\mathbf{z}_c(0) = [10 \; 10 \; \pi/2]^T$. We compare our FSC-MPC controller with Samson's controller (6.22).

Controller parameters, sampling time, simulation time and control input constraints are the same as those in Section 6.5.1.

The results are shown in Figures 6.13-6.15. The FSC-MPC controller successfully stabilizes the tracking robot to the final posture where the reference robot stops. Meanwhile, Samson's controller experiences some extreme maneuvers and only stops the tracking robot to a neighbor position.

In Case 2, the virtual reference robot starts from posture $\mathbf{z}_r(0) = [0 \;\; 0 \;\; 0]^T$ with constant control inputs $[v_r \;\; \omega_r]^T = [1 \;\; 0]^T$. Then, it stops at time $t = 15$ s. Finally, at time $t = 20$ s, it starts to move backward with constant control inputs $[v_r \;\; \omega_r]^T = [-1 \;\; -0.1]^T$. This scenario could happen when the reference trajectory is generated to avoid obstacles. The initial condition of tracking robot is $\mathbf{z}_c(0) = [0 \;\; 3 \;\; 0]^T$.

Controller parameters, sampling time and simulation time are the same as those in Section 6.5.1 except for the control input constraints

$$-5(\text{m/s}) \leq v \leq 5(\text{m/s}), \quad -0.8(\text{rad/s}) \leq \omega \leq 0.8(\text{rad/s}).$$

The results are shown in Figures 6.16 and 6.17. We can see during time $t \in [15,\ 20]$, the FSC-MPC controller does stop the tracking robot $(v = 0, \omega = 0)$. Tracking errors converge to zero.

6.6 Summary

In this chapter, a *first-state contractive* model predictive control (FSC-MPC) algorithm is developed for the trajectory tracking and point stabilization problems of nonholonomic mobile robots. Stability of the proposed MPC scheme is guaranteed by adding a first-state contractive constraint. Simulation results show that the proposed FSC-MPC controller can generate satisfactory system responses while requires much less control energy in comparison with other well-known controllers. In addition, the proposed FSC-MPC algorithm has the ability of simultaneous tracking and stabilization, in contrast to controllers available in the literature.

For all simulations, an initial feasible solution is required for the proposed FSC-MPC controller. Like most of the MPC schemes, a trial-and-error approach is used. The choice of the contractive parameter is critical for the initial feasible solution. A value close to 1 is preferred. However, a small value will give faster convergence rate when the system approaches the equilibrium point. As part of our future work, we are investigating adaptive or time-varying schemes of the contractive parameter, and experimental verifications of the FSC-MPC on the MARHES [136] testbed.

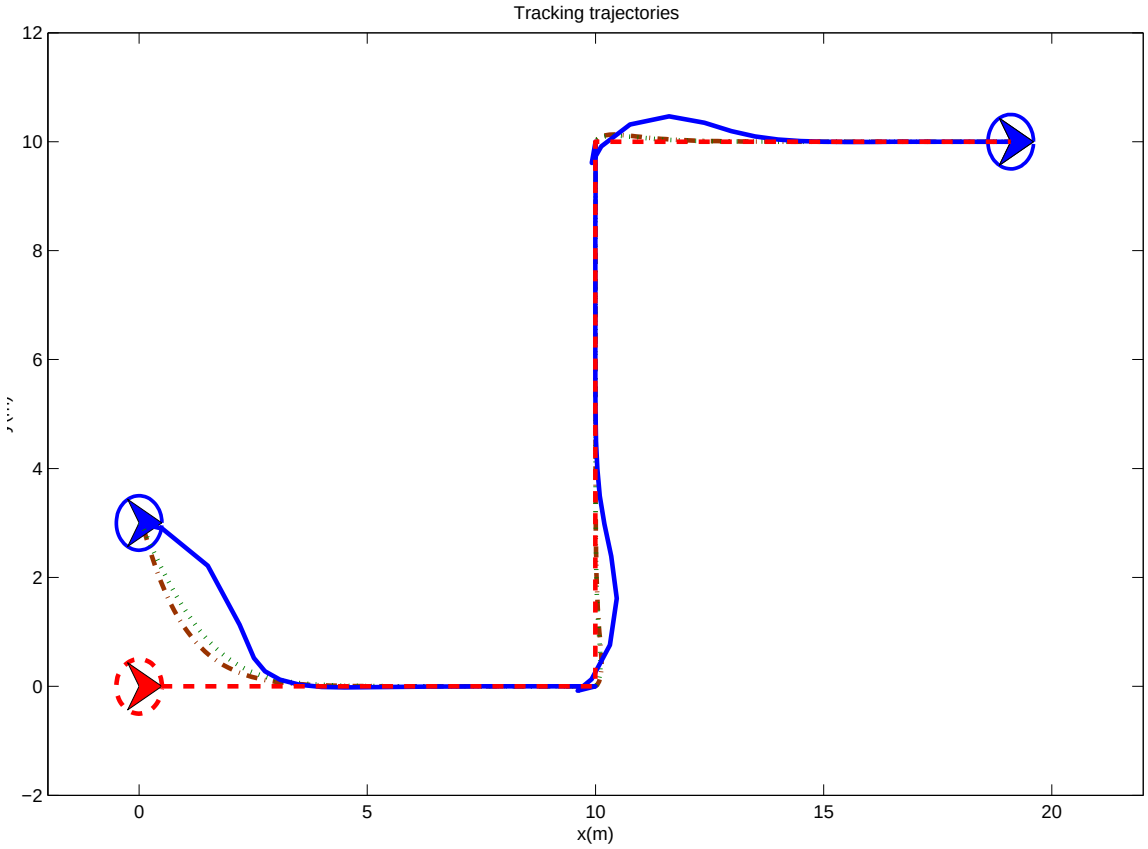


Figure 6.3: Tracking trajectories of three controllers. Dashed: Reference. Solid: FSC-MPC. Dotted: Samson. Dash-dot: Kanayama.

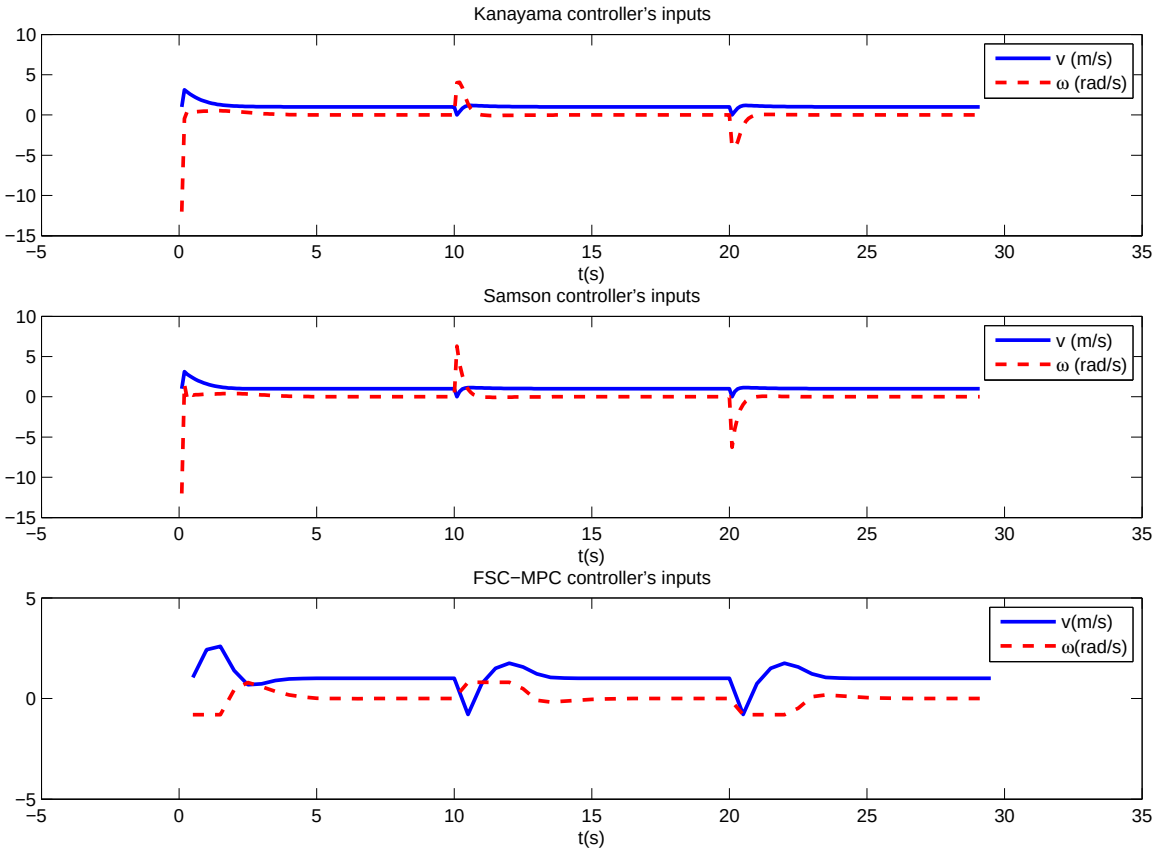


Figure 6.4: Control inputs of three controllers.

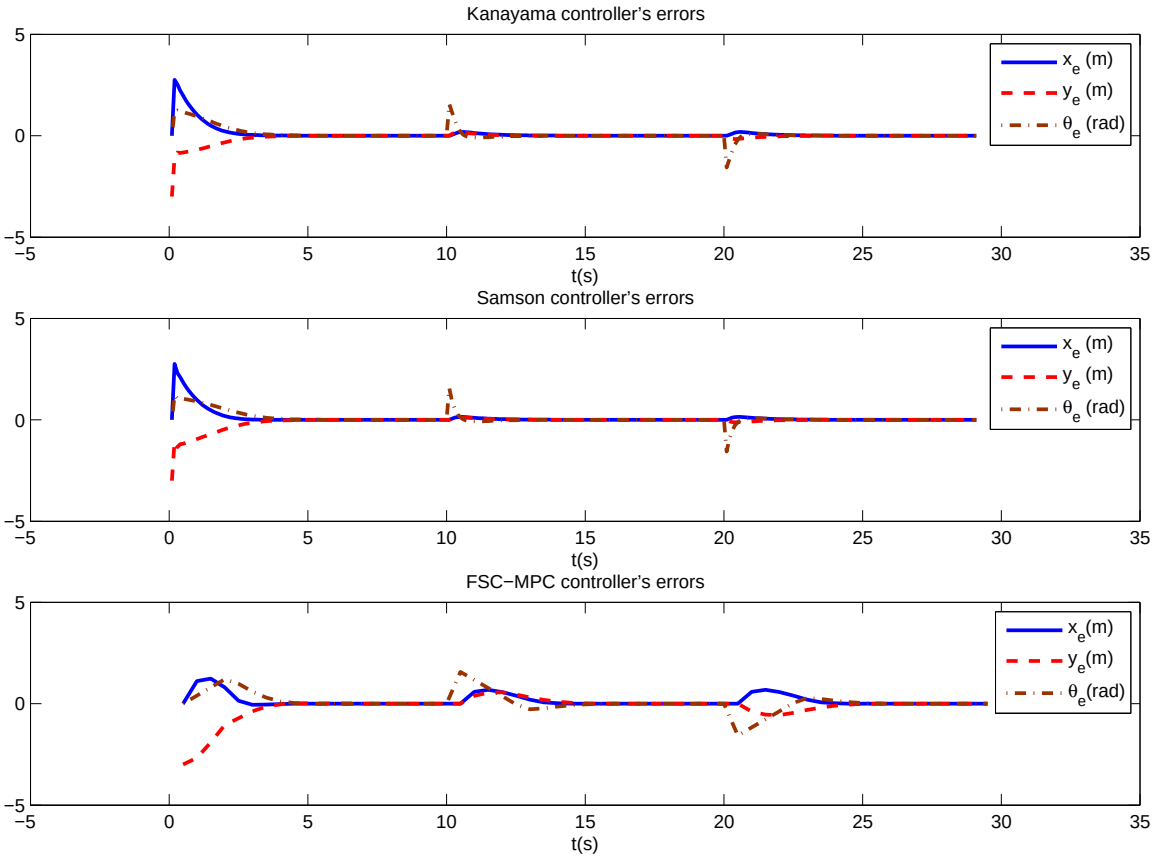


Figure 6.5: Control errors of three controllers.

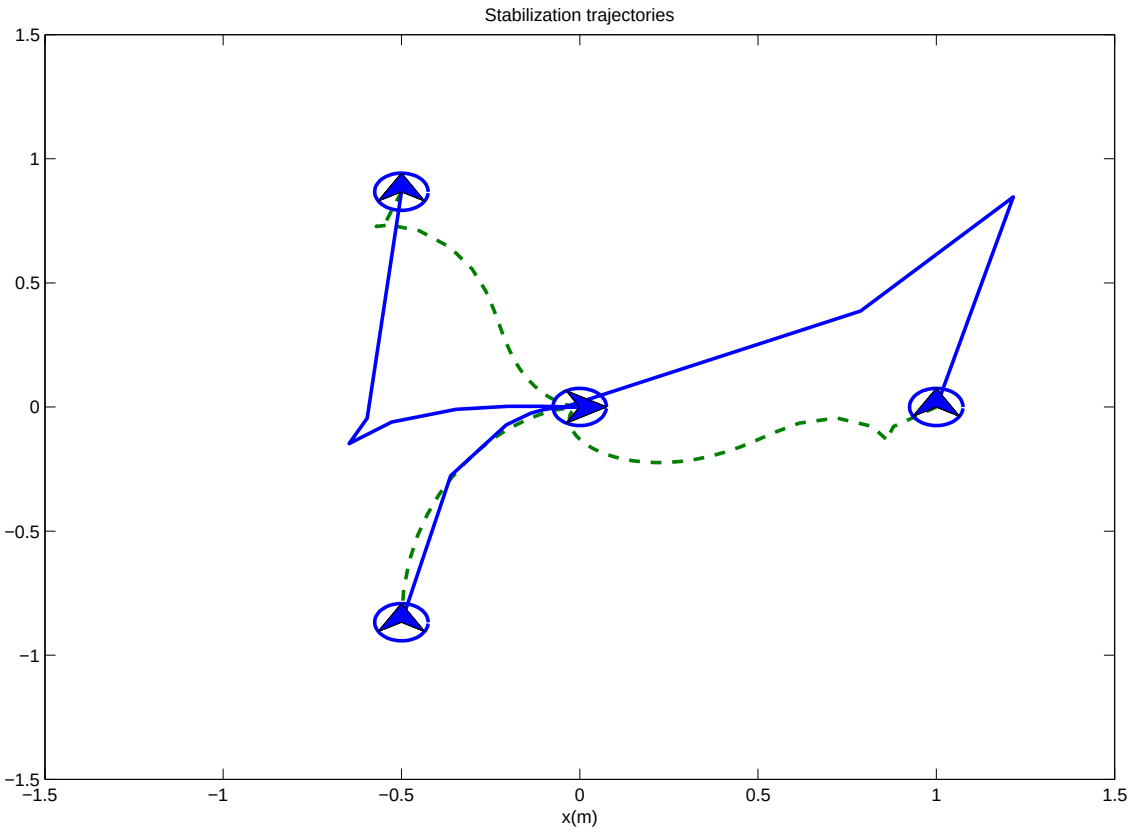


Figure 6.6: Stabilizing trajectories of two controllers. Solid: FSC-MPC. Dashed: Aicardi

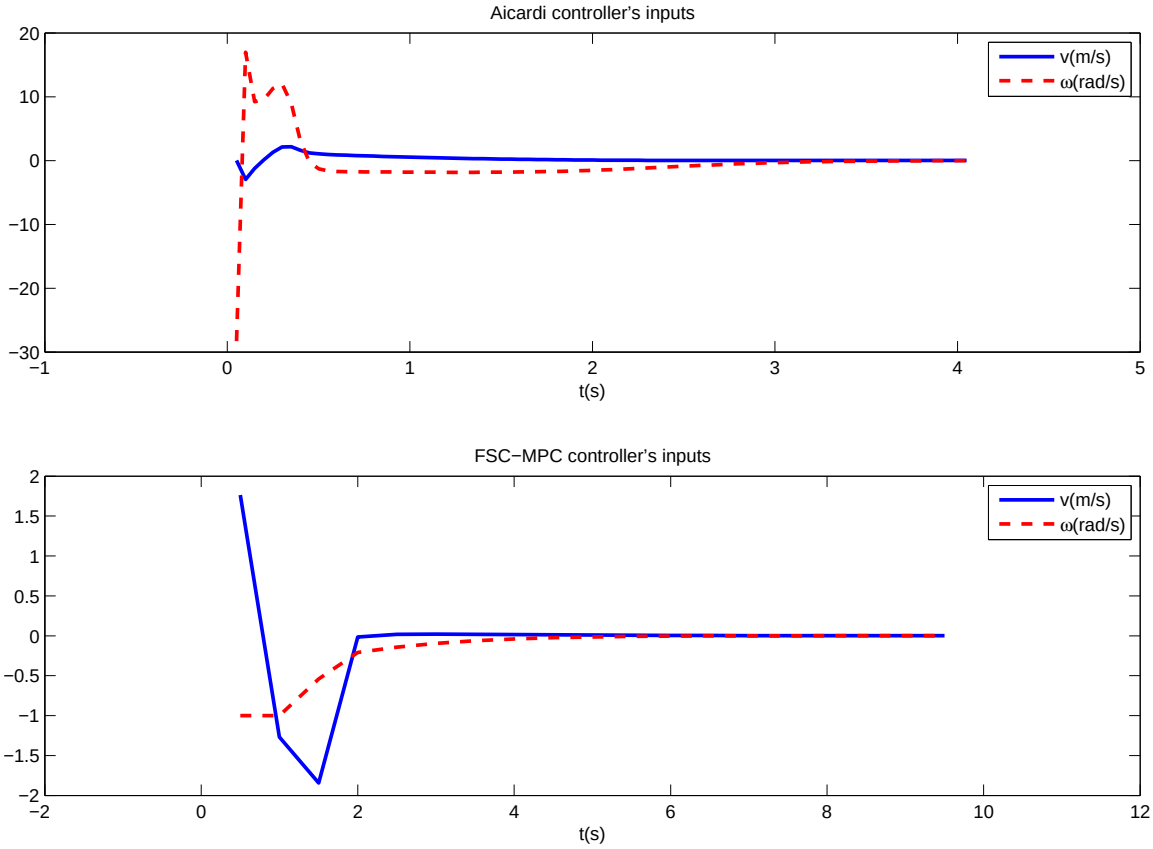


Figure 6.7: Control inputs of two controllers with $[1 \;\; 0 \;\; \pi/2]^T$ as initial posture.

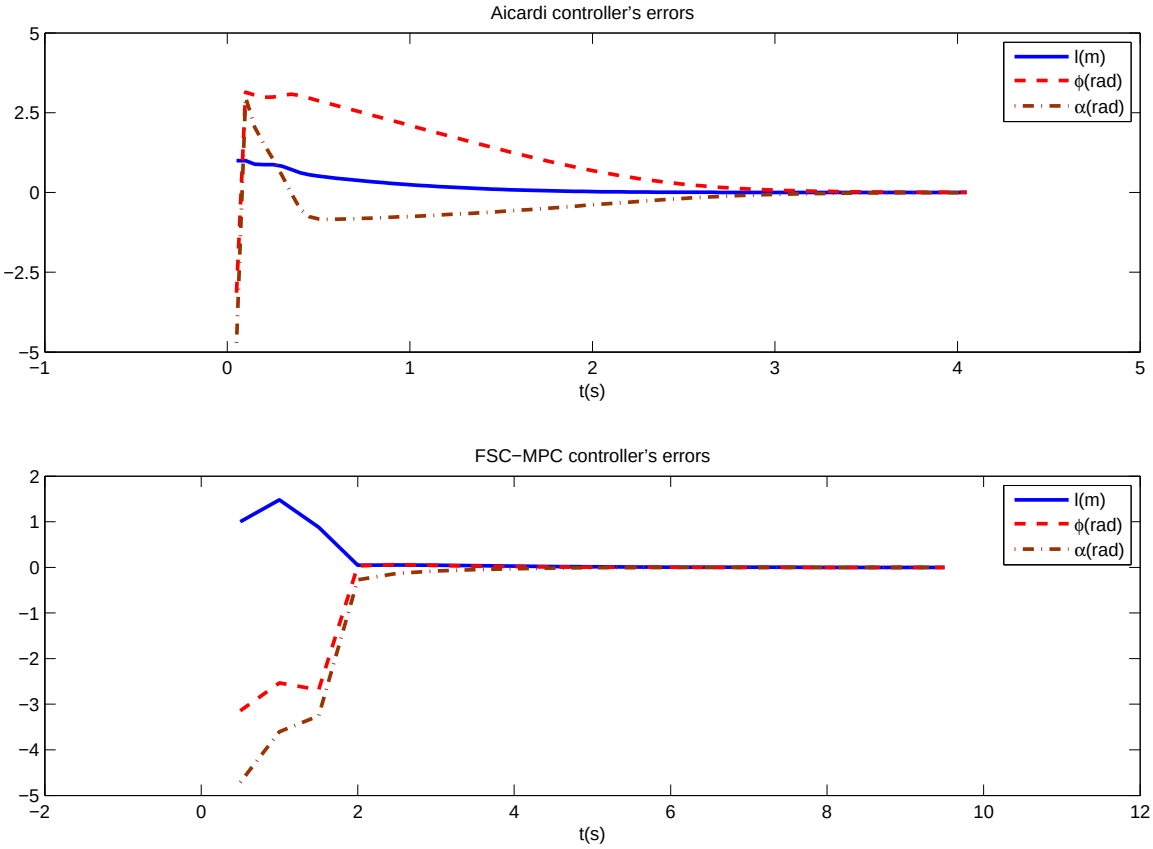


Figure 6.8: Control errors of two controllers with $[1 \;\; 0 \;\; \pi/2]^T$ as initial posture.

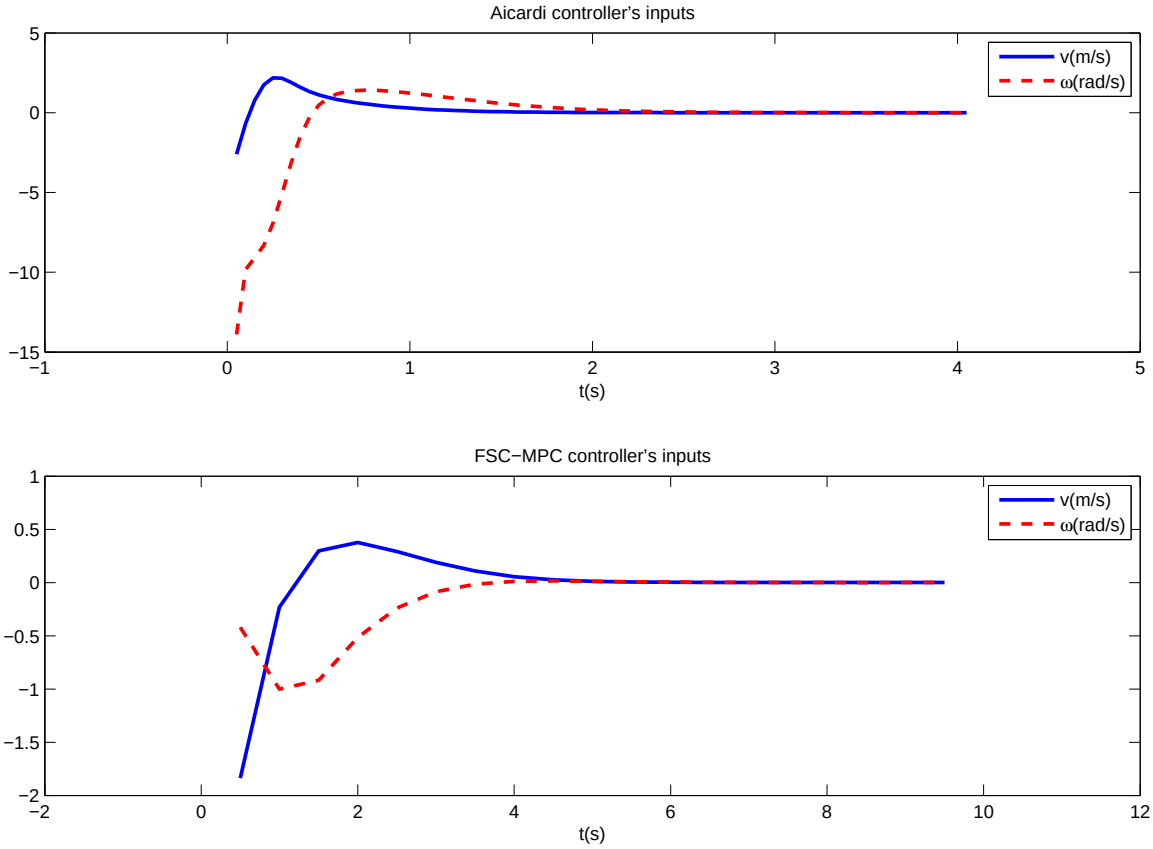


Figure 6.9: Control inputs of two controllers with $[-0.5 \;\; 0.867 \;\; \pi/2]^T$ as initial posture.

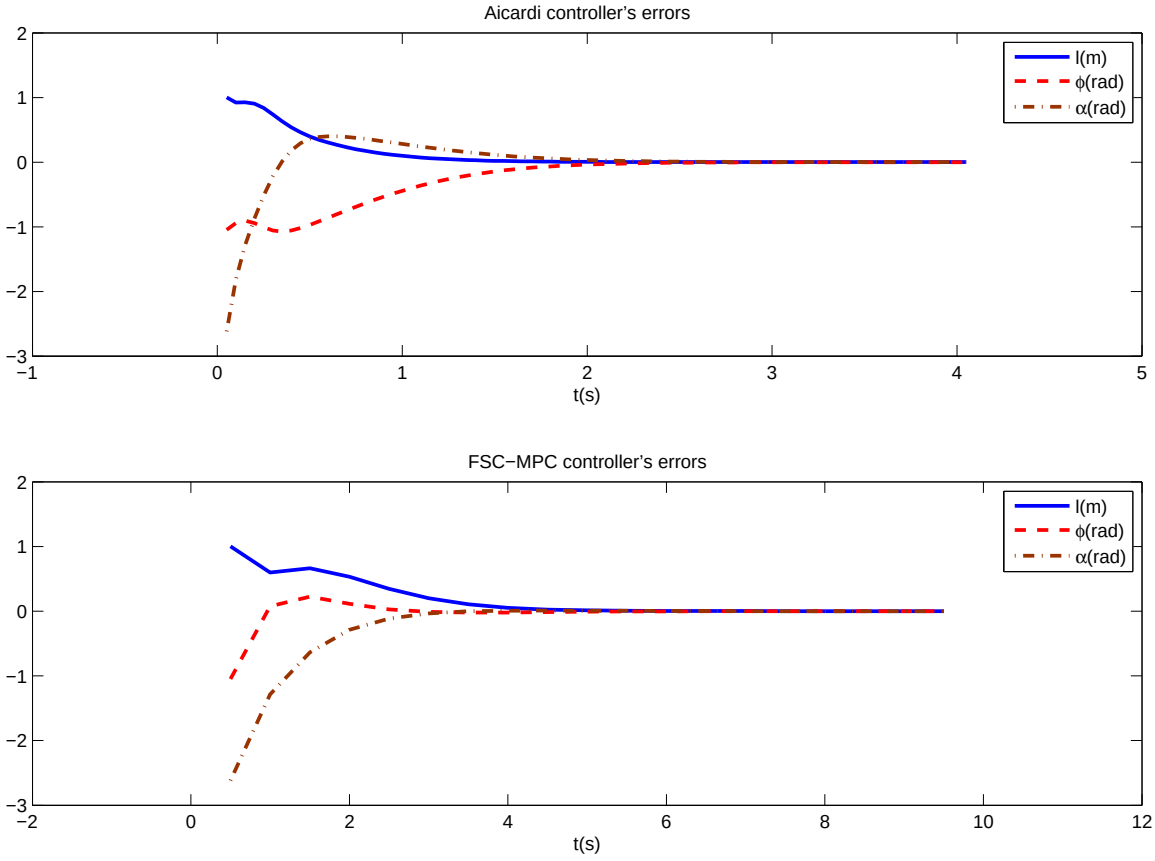


Figure 6.10: Control errors of two controllers with $[-0.5 \;\; 0.867 \;\; \pi/2]^T$ as initial posture.

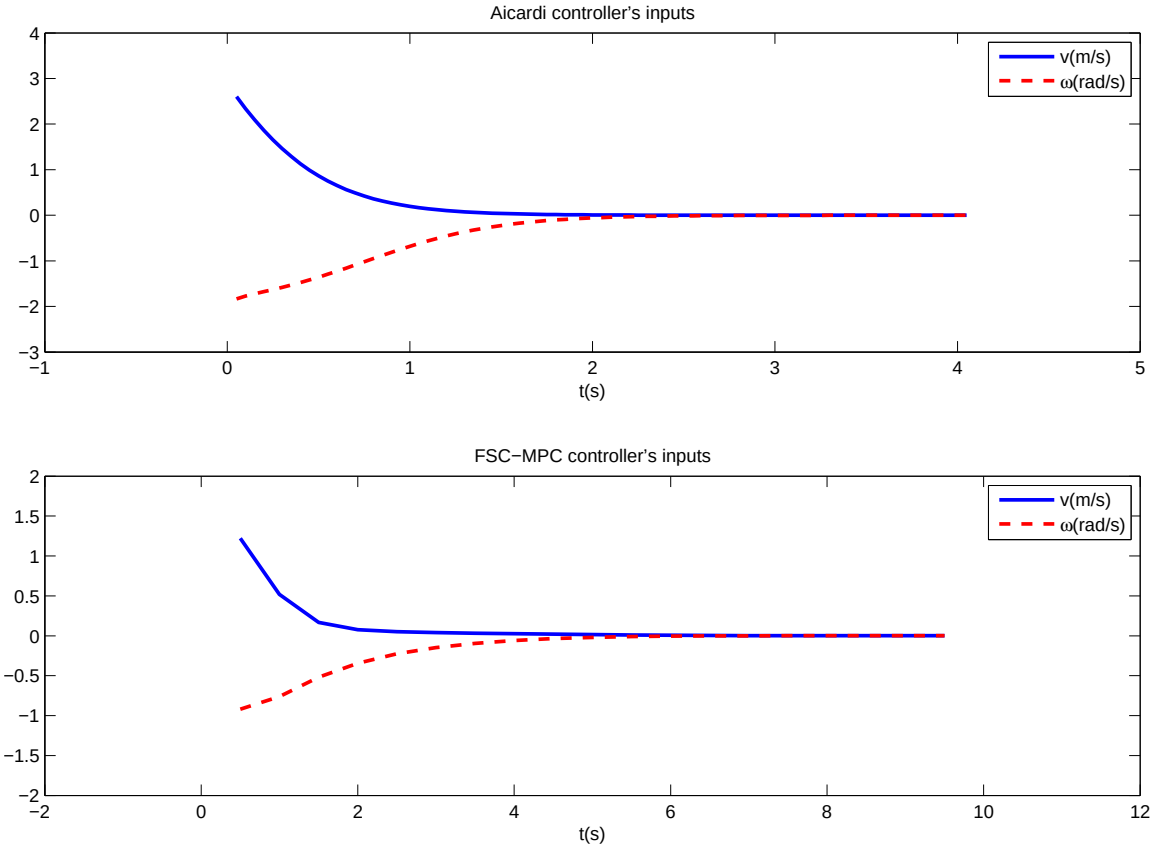


Figure 6.11: Control inputs of two controllers with $[-0.5 \quad -0.867 \quad \pi/2]^T$ as initial posture.

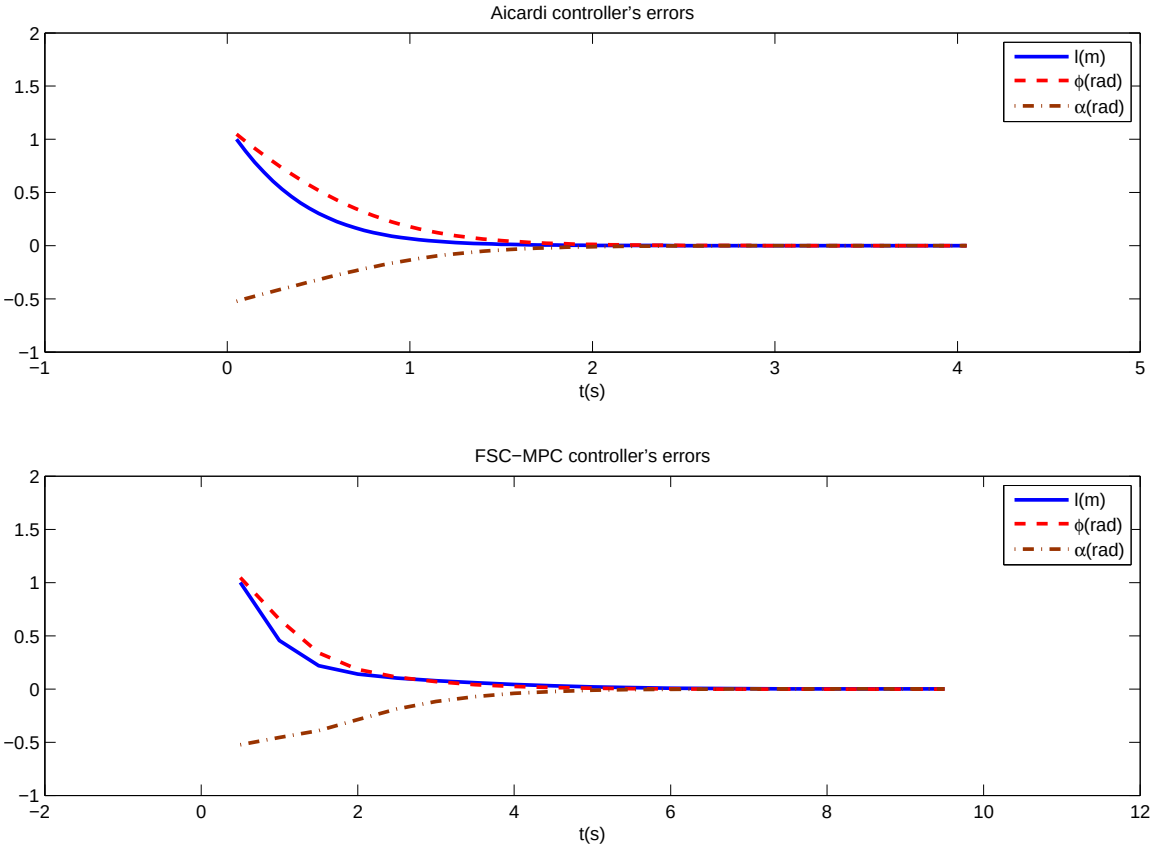


Figure 6.12: Control errors of two controllers with $[-0.5 \quad -0.867 \quad \pi/2]^T$ as initial posture.

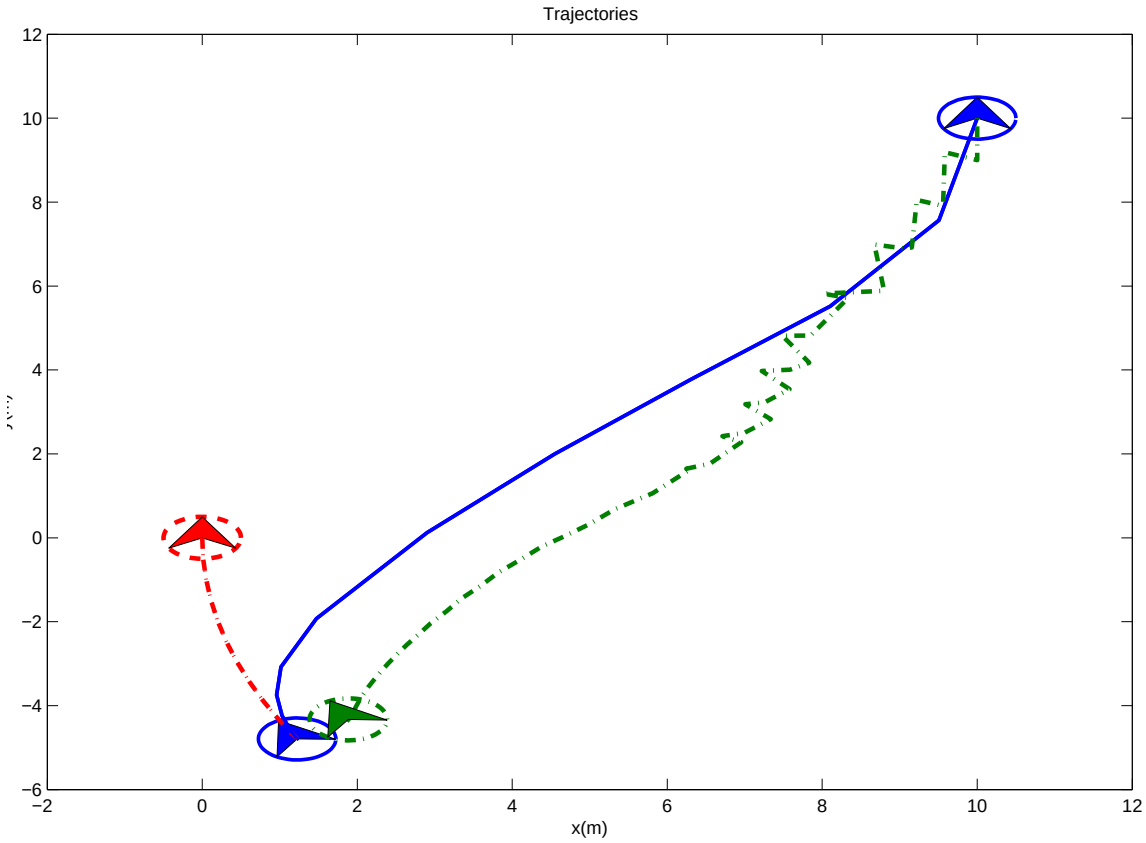


Figure 6.13: Case 1, trajectories of simultaneous tracking and stabilization. Dashed: Reference. Solid: FSC-MPC. Dash-Dot: Samson.

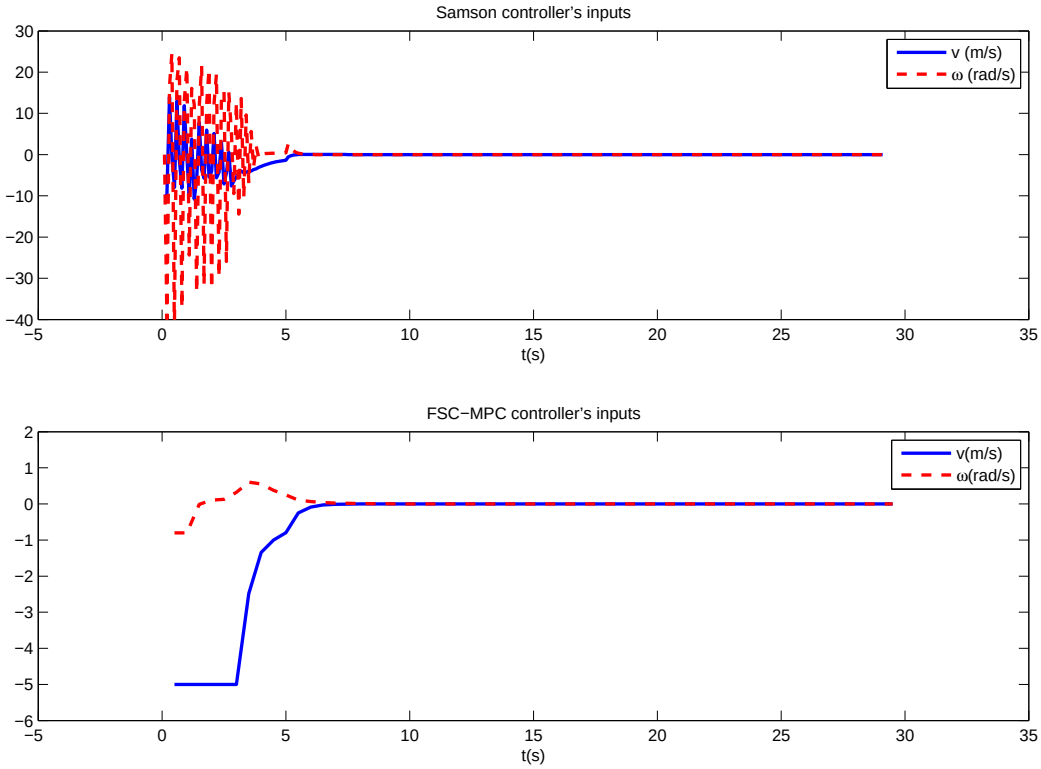


Figure 6.14: Case 1, control inputs of two controllers.

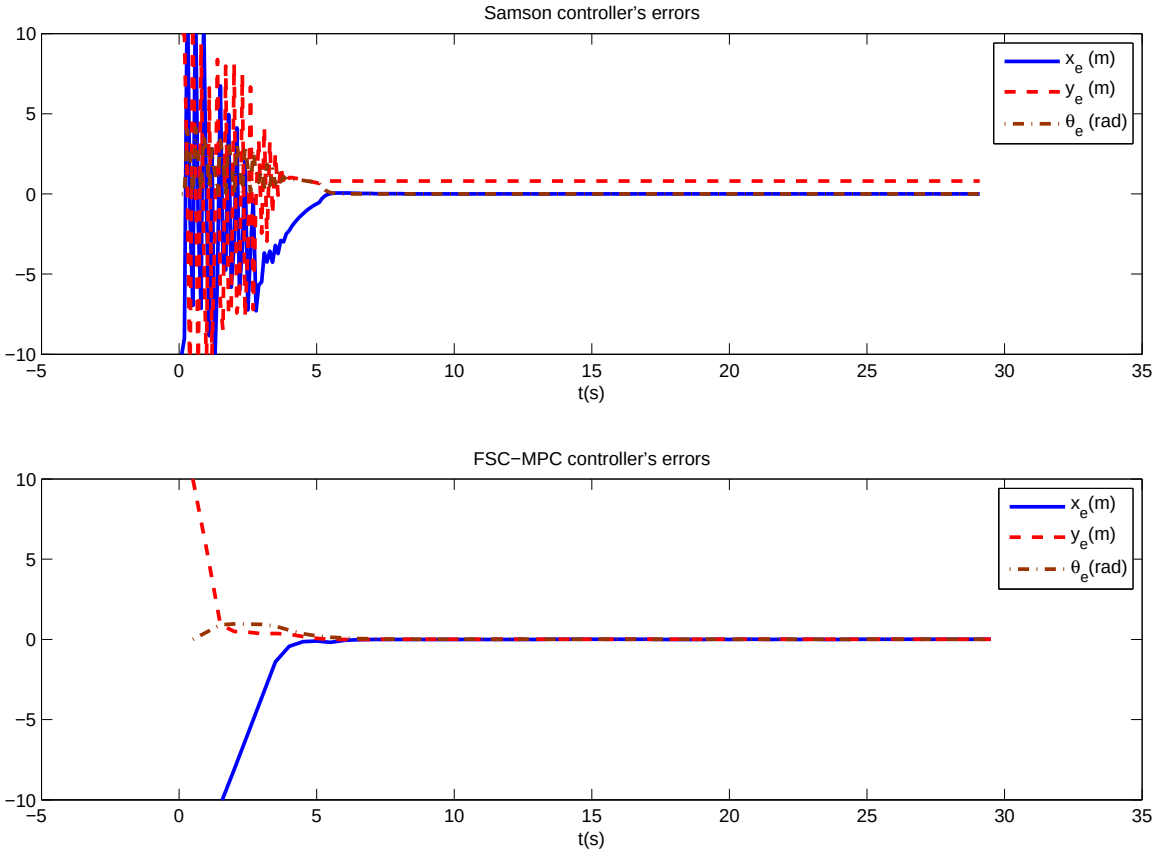


Figure 6.15: Case 1, control errors of two controllers.

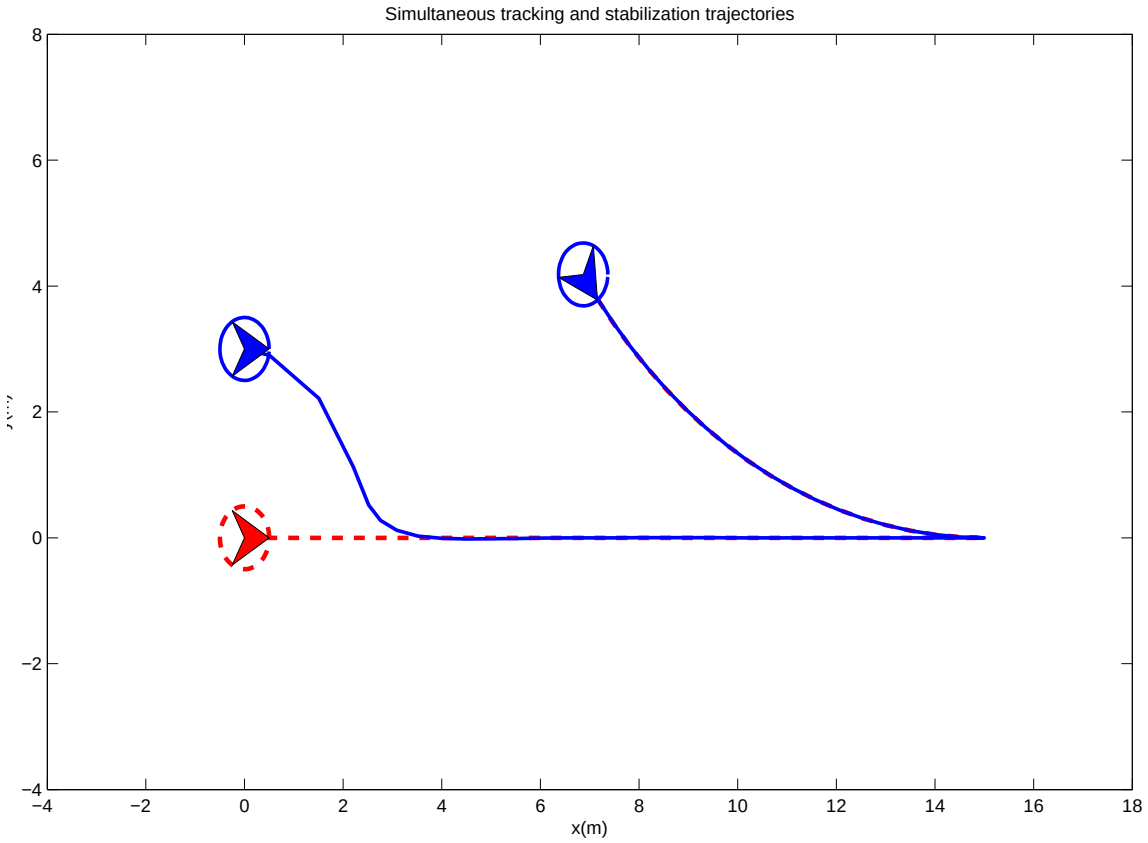


Figure 6.16: Case 2, trajectory of simultaneous tracking and stabilization. Dashed: Reference. Solid: FSC-MPC

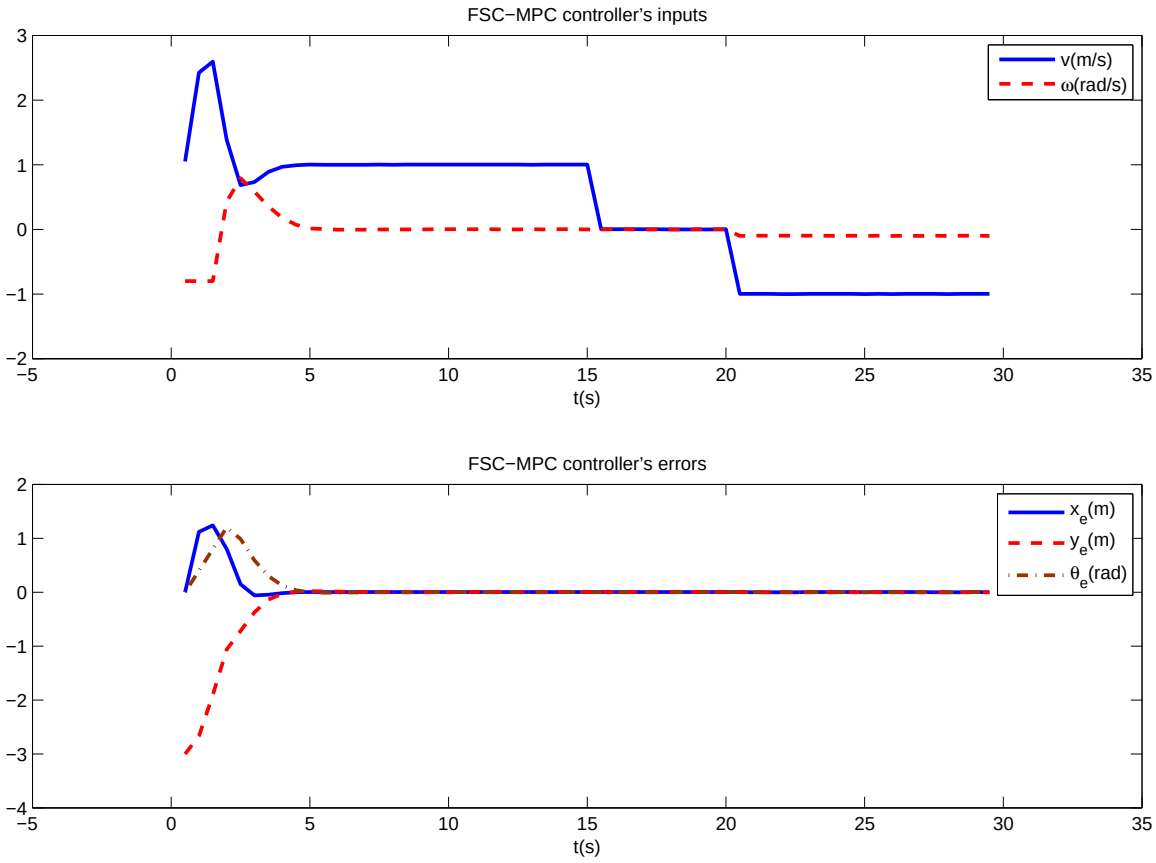


Figure 6.17: Case 2, control inputs and errors.

CHAPTER 7

FSC-MPC Formation Control

In this chapter, we consider using the FSC-MPC algorithm proposed in Chapter 6 to solve the problem of controlling a team of mobile robots with nonholonomic constraints to leader-following formations. Since the FSC-MPC algorithm has simultaneous tracking and point stabilization capability, we expect that it can keep the formation in more sophisticated real world scenarios. As before, the effectiveness of the method is investigated by numerical simulations.

7.1 Formation Error

Let a triplet $p_i = [x_i \;\; y_i \;\; \theta_i]^T$ describe the position and the orientation of the i^{th} mobile robot. A leader-following formation configuration F_{ij}^d between robot i and j can be defined by the desired relative distances Δx_{ij}^d and Δy_{ij}^d in a local coordinate frame attached to the leader robot i, where $F_{ij}^d = \begin{bmatrix} \Delta x_{ij}^d & \Delta y_{ij}^d \end{bmatrix}^T$.

However, to directly use the results in Chapter 6, we need to convert the formation control problem to a trajectory tracking problem. Let us define a reference robot, whose

position and orientation are as follows

$$\begin{aligned} x_r &= x_i + l\cos(\eta), \\ y_r &= y_i + l\sin(\eta), \\ \theta_r &= \theta_i, \end{aligned} \tag{7.1}$$

where

$$\begin{aligned} l &= \sqrt{{\Delta x_{ij}^d}^2 + {\Delta y_{ij}^d}^2}, \\ \eta &= \arctan2(\Delta y_{ij}^d, \Delta x_{ij}^d) + \theta_i. \end{aligned} \tag{7.2}$$

Now we can define the formation error for the j^{th} robot, which is a tracking error defined by

$$x_j^e = \begin{bmatrix} x_r - x_j \\ y_r - y_j \\ \theta_r - \theta_j \end{bmatrix}. \tag{7.3}$$

Figure 7.1 shows this conversion.

7.2 FSC-MPC Formation Controller

Let us define the optimal control problem for robot j, we have

$$\begin{aligned} \min_{\mathbf{u}} \quad & J_j^H(\mathbf{x}_j^e, k, \mathbf{u}_j), \\ \text{subject to:} \quad & \mathbf{x}_j^e(k+1) = \mathbf{f}(\mathbf{x}_j^e(k), \mathbf{u}_j(k)), \\ & \mathbf{x}_j^e(k) \in \mathcal{X}, \\ & \mathbf{u}_j(k) \in \mathcal{U}, \\ & \|\mathbf{x}_j^e(k+1)\|_{\hat{P}} \leq \rho \|\mathbf{x}_j^e(k)\|_{\hat{P}}, \end{aligned} \tag{7.4}$$

where $J_j^H(\mathbf{x}_j^e, k, \mathbf{u}_j)$ is defined as

$$J_j^H(\mathbf{x}_j^e, k, \mathbf{u}_j) := \sum_{i=1}^{H} L(\mathbf{x}_j^e(k+i;k), \mathbf{u}_j(k+i-1;k)), \tag{7.5}$$

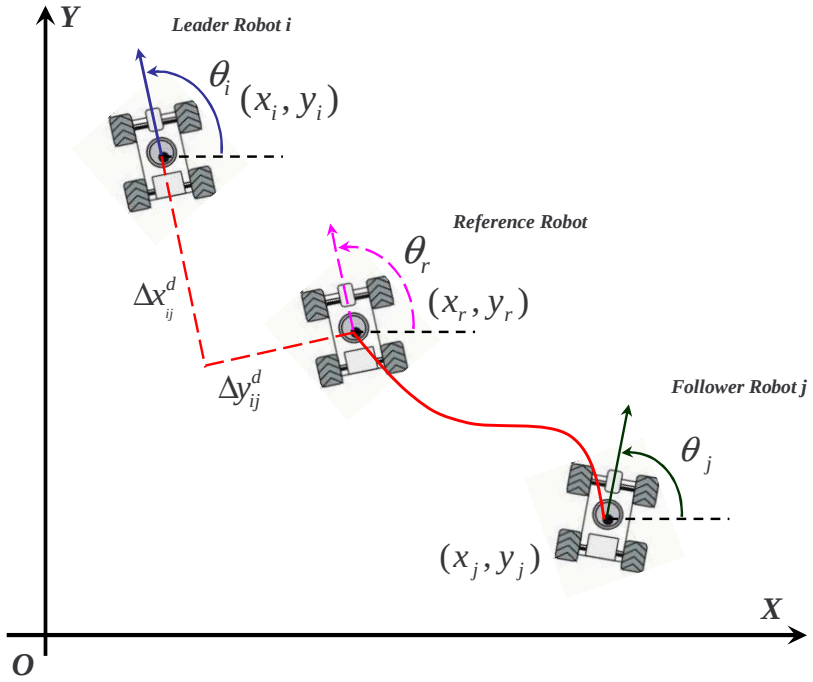


Figure 7.1: Convert formation control to trajectory tracking.

where

$$L(\mathbf{x}_j^e, \mathbf{u}_j) := \|\mathbf{x}_j^e\|_Q^2 + \|\mathbf{u}_j\|_R^2, \tag{7.6}$$

The FSC-MPC formation controller can be implemented as follows.

Data: prediction horizon $H \in \mathbb{N}$; sampling time $\delta_T \in \mathbb{R}^+$; weights Q, R, $\hat{P} > 0$; constraints $\mathbf{u}_{\min}$, $\mathbf{u}_{\max} \in \mathbb{R}^m$; contractive parameter $\rho \in (0,\ 1)$; $k \in \mathbb{Z}^*$; desired formation F_{ij}^d.

Step 0: set $k = 0$; set initial control prediction $\hat{\mathbf{u}}_i(s;k) = \hat{\mathbf{u}}_j(s;k) = 0$, $s \in [1, \ldots, H-1]$.

Step 1: receive leader states $\mathbf{p}_i(k)$, control prediction $\hat{\mathbf{u}}_i(s;k)$ and measure the states $\mathbf{p}_j(k)$ at time t_k; with control prediction $\hat{\mathbf{u}}_i(s;k)$ and $\hat{\mathbf{u}}_j(s;k)$, desired formation F_{ij}^d and relations (7.1), (7.2) and (7.3), solve the optimal control problem (7.4) and obtain a control sequence $\mathbf{u}_j^*(s;k)$.

Step 2: apply the first control $\mathbf{u}_j^*(1;k)$ in the control sequence $\mathbf{u}_j^*(s;k)$ to robot j for the time interval $[t_k,\ t_{k+1}]$, where $t_{k+1} = t_k + \delta_T$.

Step 3: update the control prediction as follows

$$\hat{\mathbf{u}}_j(s;k) = \begin{cases} \mathbf{u}_j^*(s+1;k) & s \in [1,\ldots,H-2] \\ \mathbf{u}_j^*(s;k) & s = H-1 \end{cases};$$

set $k = k+1$; go back to **Step 1**.

7.3 Simulation Results

The effectiveness of the FSC-MPC formation control algorithm presented in Section 7.2 is investigated by numerical simulations. In the figures, each robot is depicted by an arrow within a circle (dotted circle for virtual reference robot). The orientation of the robot is shown by the orientation of the arrow.

The units used in the simulations are, for position, meter (m), for orientation, radian (rad), for linear velocity, meter/second (m/s), and for angular velocity, radian/second (rad/s).

7.3.1 Reconfiguration

In this simulation, we consider a formation reconfiguration scenario. We assume that the leader robot has the ability to sense the environment and communicate with each robots in the team. During the navigation, the leader detects a wall and sends commands to follower robots to change the formation to a line such that they could pass a small gate. After all the robots pass the gate, the leader sends commands to resume the previous formation.

The leader robot starts moving from posture $\mathbf{p}_1(0) = [0 \;\; 0 \;\; 0]^T$ with constant control inputs $[v_1 \;\; \omega_1]^T = [1 \;\; 0]^T$. Robot 2 and 3 start from $\mathbf{p}_2(0) = [-8 \;\; -1 \;\; \pi/4]^T$ and $\mathbf{p}_3(0) = [-15 \;\; -3 \;\; -\pi/4]^T$. The desired formation is $F_{12}^d = [-5 \;\; 2]^T$ and $F_{23}^d = [-5 \;\; -4]^T$. At time $t = 15$ s, the leader detects the wall and sends commands to team robots to change to a new formation $F_{12}^d = [-5 \;\; 0]^T$ and $F_{23}^d = [-5 \;\; 0]^T$, which is a line formations. At

time $t = 30$ s, as all the robots have passed the gate, the leader sends commands again and changes the formation back the original one $F_{12}^d = [-5 \;\; 2]^T$ and $F_{23}^d = [-5 \;\; -4]^T$.

The control parameters for the FSC-MPC formation controller are $H = 3$, $\rho = 0.999$,

$$Q = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \hat{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Control input constraints are

$$-4(\text{m/s}) \leq v \leq 4(\text{m/s}), \quad -0.5(\text{rad/s}) \leq \omega \leq 0.5(\text{rad/s}).$$

Sampling time is $\delta_T = 0.5$ s and the total simulation time is 50 s.

Figure 7.4 shows the trajectories of the robot team. We can see that the triangle formation is kept before and after the change to a line formation and the line formation is achieved when the team pass the gate.

Figure 7.5 and 7.6 show the control inputs and the relative position of robot 2. From 7.6, we can see that the actual relative position of robot 2 converges to the desired formation $F_{12}^d = [-5 \;\; 2]^T$ before $t = 15$ s and after $t = 30$ s and the desired formation $F_{12}^d = [-5 \;\; 0]^T$ during time $[15 \;\; 30]$. From 7.5 we can see that, when the formation is achieved, the control inputs converge to $[1 \;\; 0]^T$, which are the velocities of the leader robot.

Figure 7.7 and 7.8 show the control inputs and the relative position of robot 3. From Figure 7.8, we can see that the actual relative position of robot 3 converges to the desired formation $F_{23}^d = [-5 \;\; -4]^T$ before $t = 15$ s and after $t = 30$ s and the desired formation $F_{23}^d = [-5 \;\; 0]^T$ during time $[15 \;\; 30]$. Figure 7.7 shows that, when the formation is achieved, the control inputs converge to $[1 \;\; 0]^T$, which are the velocities of the leader robot. However, robot 3 requires more control energy than robot 2 does by comparing Figure 7.5 and 7.7.

7.3.2 Obstacle Avoidance

In this simulation, we consider a obstacle avoidance scenario. We assume that the leader robot has the ability to sense the environment. During navigation, the leader robot detects an obstacle. It stops and then moves backward, stops and moves forward for several times to make sure the whole team will not collide with the obstacle. Finally, the team moves forward again in formation.

The leader robot starts moving from posture $\mathbf{p}_1(0) = [0 \;\; 0 \;\; 0]^T$. The leader robot's velocity inputs are

$$\begin{bmatrix} v_1 \\ \omega_1 \end{bmatrix} = \begin{cases} [1 \;\; 0]^T & t = [0 \;\; 15], \\ [0 \;\; 0]^T & t = [15.5 \;\; 20], \\ [-1 \;\; -0.15]^T & t = [20.5 \;\; 25], \\ [0 \;\; 0]^T & t = [25.5 \;\; 30], \\ [1 \;\; 0.15]^T & t = [30.5 \;\; 35], \\ [0 \;\; 0]^T & t = [35.5 \;\; 39.5], \\ [1 \;\; 0]^T & t = 40, \\ [-1 \;\; -0.15]^T & t = [40.5 \;\; 45], \\ [0 \;\; 0]^T & t = [45.5 \;\; 50], \\ [1 \;\; 0]^T & t = [50 \;\; 60], \end{cases}$$

Robot 2 and 3 start from $\mathbf{p}_2(0) = [-8 \;\; -1 \;\; \pi/4]^T$ and $\mathbf{p}_3(0) = [-15 \;\; -3 \;\; -\pi/4]^T$. The desired formation is $F_{12}^d = [-5 \;\; 2]^T$ and $F_{23}^d = [-5 \;\; -4]^T$.

Control parameters for the FSC-MPC formation controller are, $H = 4$, $\rho = 0.999$,

$$Q = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \hat{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Control input constraints are

$$-4(\text{m/s}) \leq v \leq 4(\text{m/s}), \quad -0.5(\text{rad/s}) \leq \omega \leq 0.5(\text{rad/s}).$$

Sampling time is $\delta_T = 0.5$ s and the total simulation time is 60 s.

Figure 7.9 shows the trajectories of the robot team. We can see that the triangle formation is kept.

Figure 7.10 and 7.11 depict the control inputs and the relative position of robot 2. From Figure 7.11, we can see that the actual relative position of robot 2 converges to the desired formation $F_{12}^d = [-5 \;\; 2]^T$. Figure 7.10 shows that, when the formation is achieved, the control inputs converge to the velocities of the leader robot.

Figure 7.12 and 7.13 show the control inputs and the relative position of robot 3. From 7.13, we can see that the actual relative position of robot 3 converges to the desired formation $F_{23}^d = [-5 \;\; -4]^T$. However, by comparing Figure 7.10 and 7.12, we can see that robot 3 requires more control energy than robot 2.

7.3.3 Discussion

From the simulation results in Section 7.3, we notice a problem, the lower *level* number the robot has in a formation, the more control energy it needs to keep the formation.

Here the *level* number can be defined as

- in a formation which has N robots, if robot l is the leader, than its *level* number is 1,
- for robot i, $i \in [1 \cdots N] \backslash l$, if it has a formation relationship with leader l, F_{li}^d, then robot i has a *level* number 2,
- for robot j, if it has a formation relationship with robot i, which has a *level* of n, then robot j's *level* number is $n + 1$,
- the bigger the *level* number is, the lower *level* rank it stands for.

If we redo the obstacle avoidance simulation with five robots, where the formations are defined as $F_{12}^d = [-5 \;\; 2]^T$, $F_{23}^d = [-5 \;\; -4]^T$, $F_{35}^d = [-5 \;\; 2]^T$ and $F_{14}^d = [-5 \;\; -3]^T$ and the *levels* for robots 1, 2, 3, 4, and 5 are 1, 2, 3, 2, and 4, respectively (see Figure 7.2), from

the results in Figure 7.14, it is very clear that robots 3 and 5 requires more control energy than robot 2 and 4.

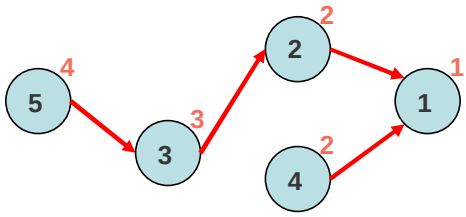


Figure 7.2: Formation *level* of five robots.

This problem can be explained with the help of Figure 7.3. Three robots have a line formation, which is defined by $F_{12}^d = F_{23}^d = [d \quad 0]^T$. Now the leader robot 1 changes its orientation with a small angle θ. To keep the formation, robot 2 at least needs to travel a distance of $d\theta$. For robot 3, that number will be $2d\theta$. This reveals that, in a formation, robots with lower *level* numbers need to travel more distance (which means more control energy) to keep the formation. Therefore, an advise from this discussion is that, for a robot team, too many *levels* in a formation will deteriorate the formation control quality.

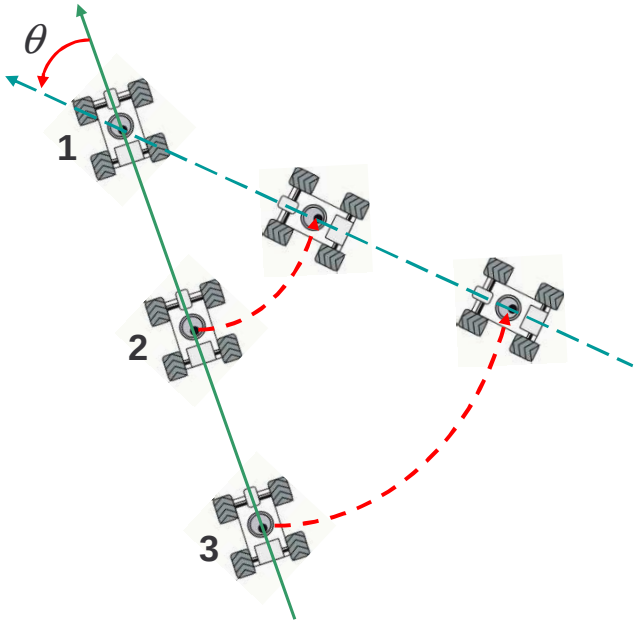


Figure 7.3: Position change in formation.

7.4 Summary

In this chapter, we extend the results of Chapter 6 to formation control. To directly apply the results, the formation control problem is firstly converted to a trajectory tracking problem. Then a FSC-MPC formation controller is constructed. Two real world sophisticated scenarios are simulated and the FSC-MPC formation controller shows its effectiveness. In addition, a discussion about the formation *level* to which the formation control quality relates, is given.

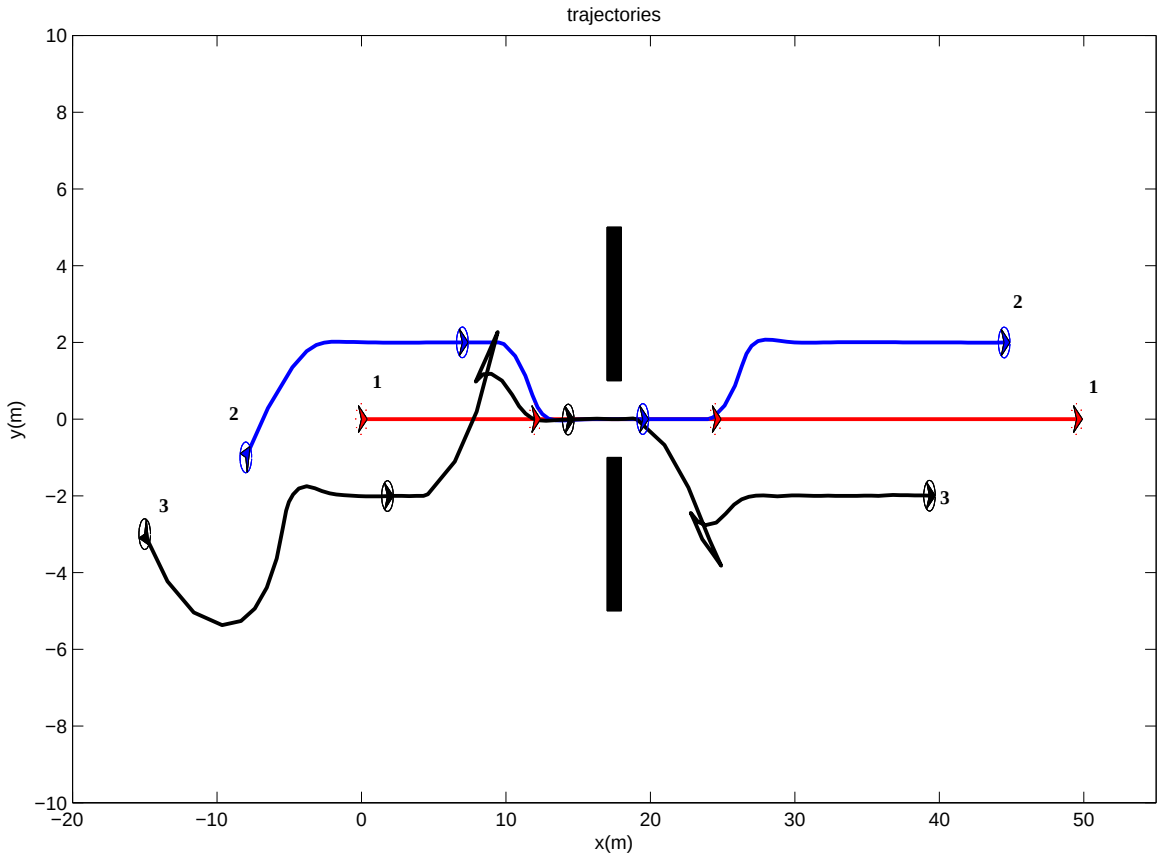


Figure 7.4: Reconfiguration during navigation.

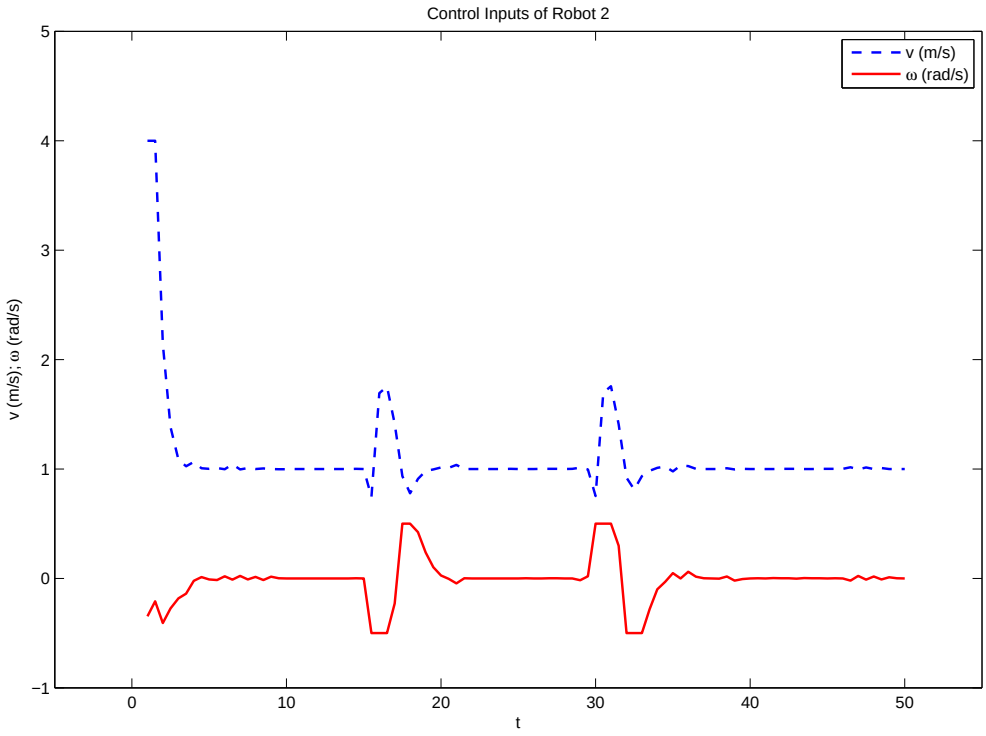


Figure 7.5: Control inputs of Robot 2.

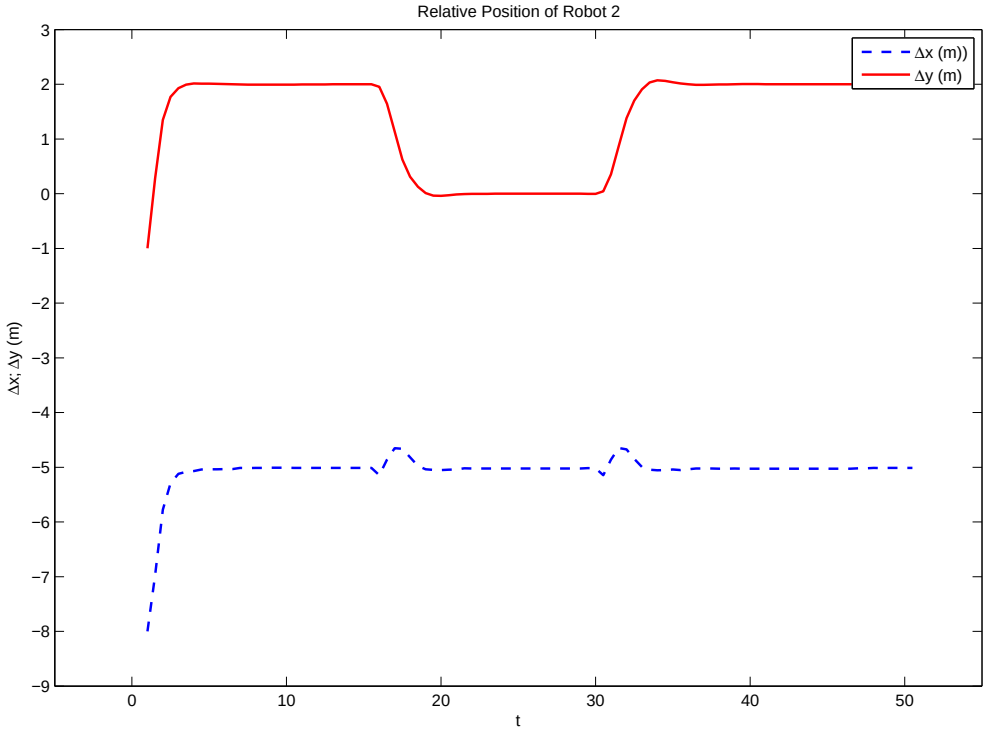


Figure 7.6: Relative position of Robot 2.

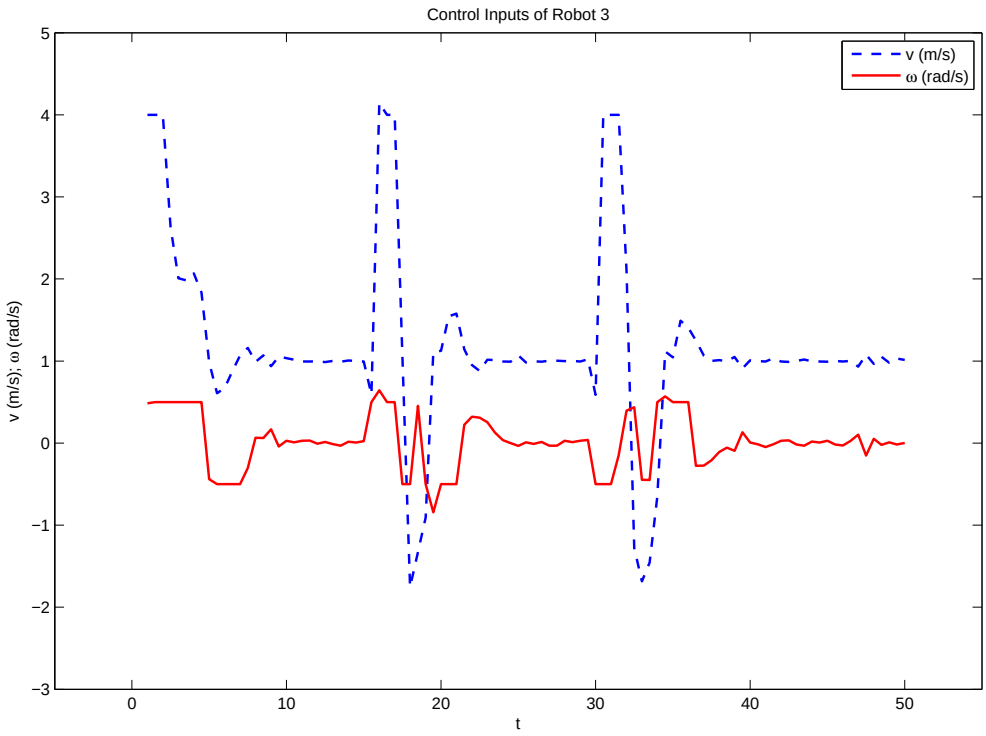


Figure 7.7: Control inputs of Robot 3.

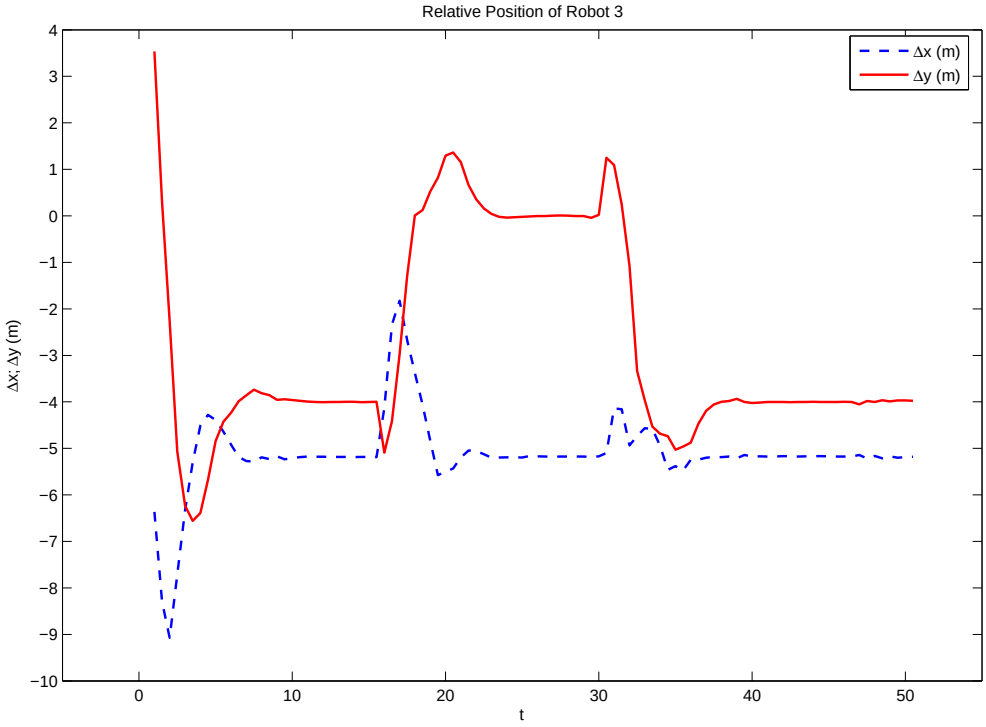


Figure 7.8: Relative position of Robot 3.

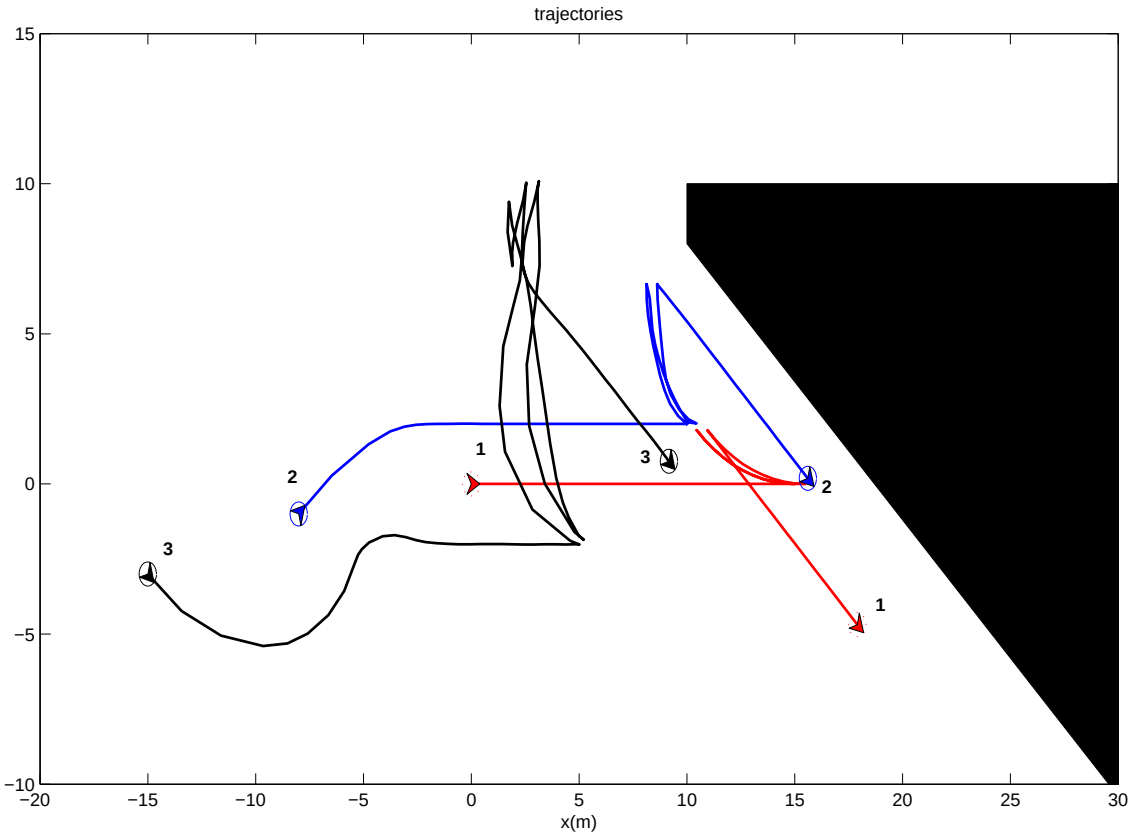


Figure 7.9: Obstacle avoidance during navigation - three robots.

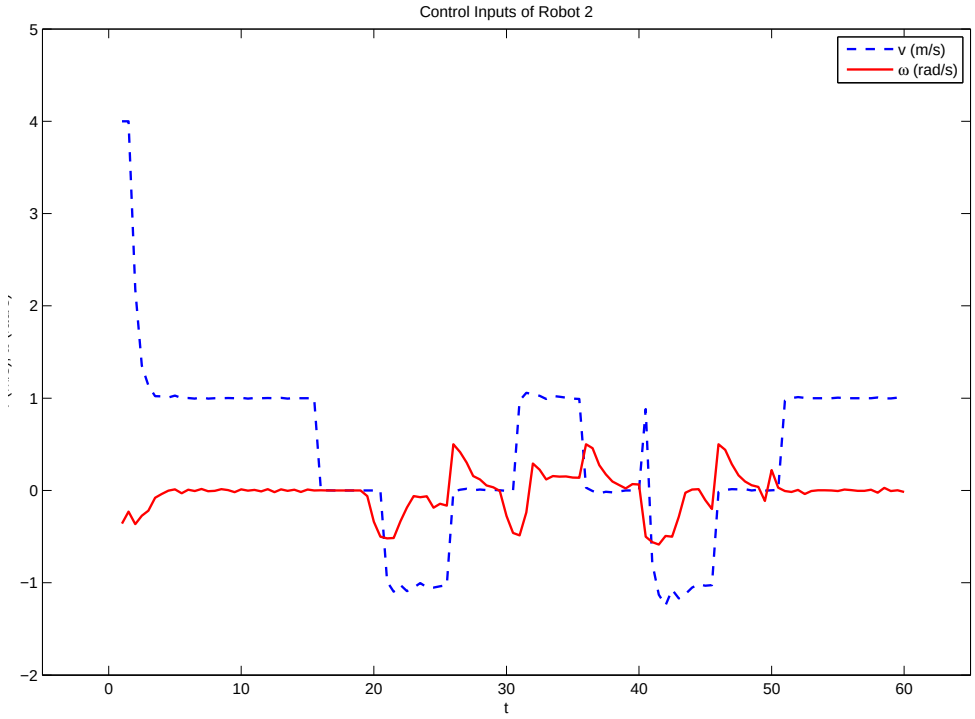


Figure 7.10: Control inputs of Robot 2.

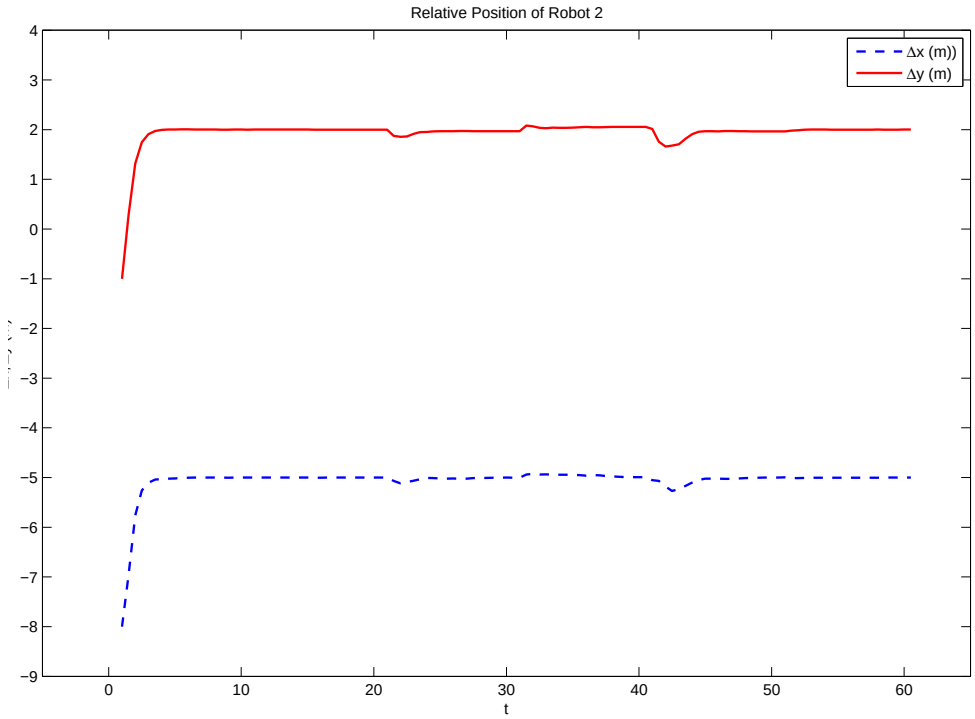


Figure 7.11: Relative position of Robot 2.

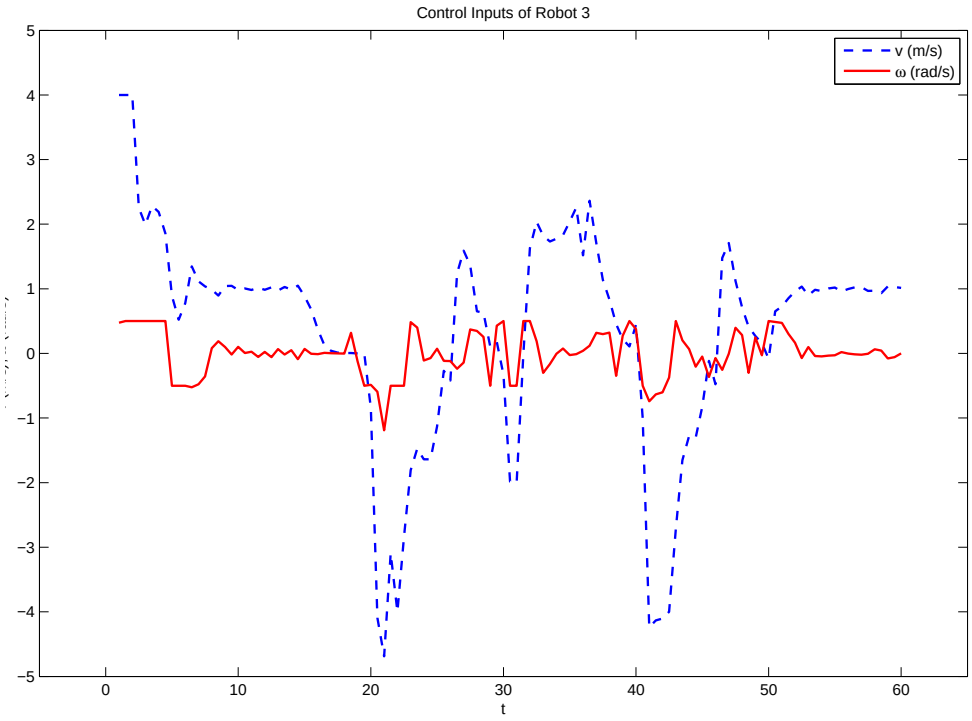


Figure 7.12: Control inputs of Robot 3.

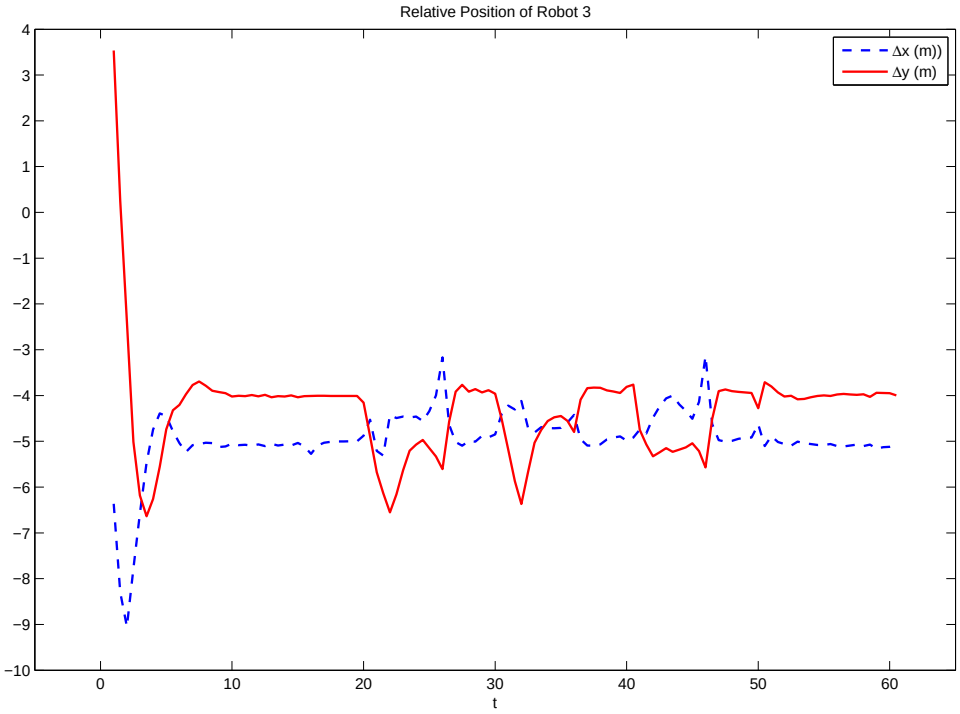


Figure 7.13: Relative position of Robot 3.

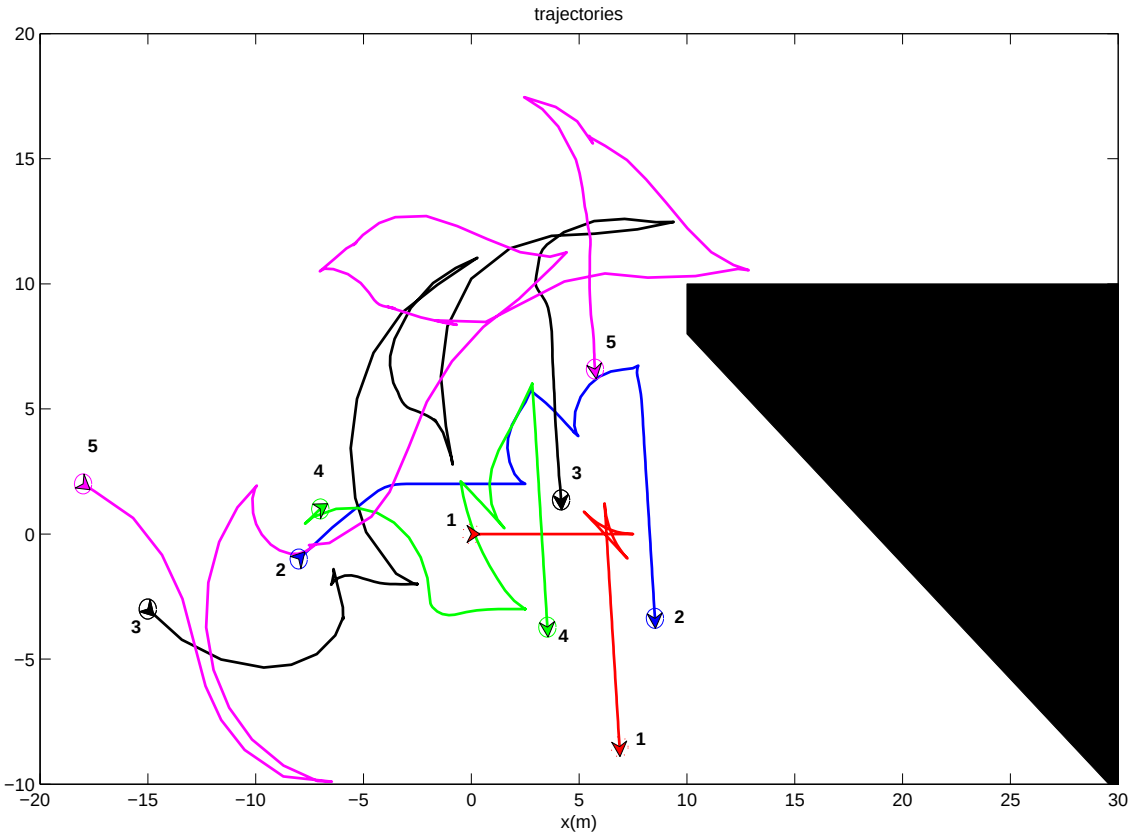


Figure 7.14: Obstacle avoidance during navigation - five robots.

CHAPTER 8

Conclusions

8.1 Summary of Main Results

In this dissertation, two problems are addressed. The first problem is stabilizing a group of nonholonomic mobile robots into formations. The second problem is the trajectory tracking and point stabilization problem of nonholonomic mobile robots. Three control algorithms are proposed step by step each solving some the drawbacks of the preceding algorithms. As pointed out in Chapter 1 and 6, most of the controllers in the literature will fail when face a trajectory/leader moving backward or a stationary trajectory/leader. The first algorithm we proposed is a robust nonlinear formation controller which can handle the situation of leader moving backward. However, it still will fail when the leader stops. In addition, control input constraints are not considered. The second algorithm is a dual-mode MPC controller which explicitly considers the constraints. The third algorithm is a *first-state-contractive* MPC (FSC-MPC) controller with the simultaneous tracking and stabilization capability. With this controller, we solve the problem of handling a trajectory/leader moving backward or a stationary trajectory/leader.

In Chapter 4, we consider the formation control problem of UAVs by Lyapunov-based nonlinear controller design techniques. A two-layered hierarchical control scheme is presented. At the low-layer, we assume that an autopilot operating in *holding mode* controls

the UAV dynamics. With this assumption, a simplified nonholonomic model is constructed for the higher layer formation controller design. The formation is considered in leader-following relation and we use the relative distance and bearing angle between a UAV and its designated leader to define the control law. Using dynamic extension, three different controllers are presented. While the first two controllers, a feedback linearization controller and a sliding mode controller, assume full states information of the leader, the third robust controller only requires the knowledge of leader's position. By eliminating the requirement of leader's velocity and acceleration information, the robust controller reduces the inter-vehicle communication overhead and increases the reliability of the overall system. Stability properties of the controllers are proven using Lyapunov theory. Simulations validate the performance of the algorithms.

The formation control problem is addressed again in the context of MPC in Chapter 5. We propose that it is more convenient to put the vehicles's nonholonomic constraints inside the MPC framework. The formation is defined with graph theory. Since a finite horizon optimal control problem is solved in a generic MPC algorithm, the control obtained is not guaranteed to be stable. We proposed a dual-mode MPC formation controller. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. When the state enters the terminal region, it will be driven to the origin according to the stability properties of the terminal controller. The stability of the system is guaranteed if the system, starting from a state outside of the terminal region, will reach the boundary of the terminal region within finite time under the dual-mode MPC algorithm. This result is proven by contradiction and the monotonicity property of the performance index function. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed. For the choice of stabilizing terminal controller, a comparison between an input-output feedback linearization controller and a robust formation controller is given.

The problem of trajectory tracking and point stabilization of nonholonomic mobile

robots is addressed in Chapter 6. We proposed a novel FSC-MPC approach for the control of nonholonomic mobile robots. Different from most stabilizing MPC methods, which address stability by adding terminal state penalties in the performance index and/or imposing constraints on the terminal state at the end of the prediction horizon, the stability of the FSC-MPC algorithm is guarantees by adding a contractive constraint on the first state at the beginning of the prediction horizon. With this first-state contractive constraint, the proposed FSC-MPC algorithm is exponentially stable. The convergence is faster and no terminal region calculation is required. Tracking a trajectory moving backward is no longer a problem under this FSC-MPC controller. Furthermore, the proposed FSC-MPC controller has simultaneous tracking and point stabilization capability. Simulations also show that, the FSC-MPC controller exhibits a comparable performance while requires much less control energy in comparison with other controllers available in the literature.

8.2 Future Work

Distributed control algorithms are highly desired in multi-vehicle coordination. How to formulate the MPC algorithm in a distributed way, how to guarantee the stability of a local system with limited information from neighboring systems, and how to generate a comparable result with a centralized algorithm are problems that need further exploration.

The current work in this report only focuses on robots' kinematics. In real world applications, the dynamics of the mobile robots cannot be ignored. Assuming a lower level controller, usually a proportional-integral-derivative (PID) controller, handling the dynamics might not be the optimal solution. Designing MPC controllers to handle robot dynamics to form a two-layer MPC approach could be a promising future work.

In addition, model uncertainties and disturbances are not considered in our current work. Though MPC is a algorithm combined with feedforward and feedback properties, the robustness of MPC needs further investigation.

BIBLIOGRAPHY

[1] R. M. Murray, "Recent research in cooperative control of multi-vehicle systems," *ASME Journal of Dynamic Systems, Measurement, and Control, submitted*, August 2006.

[2] R. Fierro, L. Chaimowicz, and V. Kumar, "Multi-robot cooperation," in *Autonomous Mobile Robots: Sensing, Control, Decision Making and Applications*, S. S. Ge and F. L. Lewis, Eds. CRC Press - Taylor and Francis Group, May 2006, ch. 11, pp. 417–459.

[3] M. J. Vachon, R. J. Ray, K. R. Walsh, and K. Ennix, "F/a-18 performance benefits measured during the autonomous formation flight project," NASA Dryden Flight Research Center, Edwards, California., Tech. Rep. NASA/TM-2003-210734, 2003.

[4] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1993.

[5] R. W. Brockett, "Asymptotic stability and feedback stabilization," in *Diferential Geometric Control Theory*, R. W. Brockett, R. S. Millman, and H. J. Sussmann, Eds., Brrkhuser, Boston, MA, 1983, pp. 181–191.

[6] F. Xie, X. Zhang, R. Fierro, and M. Motter, "Autopilot-based nonlinear uav formation controller with extremum-seeking," in *Proc. IEEE Conf. on Decision and Control*, December 2005, pp. 4933–4938.

[7] F. Xie and R. Fierro, "On motion coordination of multiple vehicles with nonholonomic constraints," in *Proc. American Control Conference*, July 2007, pp. 1888–1893.

[8] ——, "First-state contractive model predictive control of nonholonomic mobile robots," *IEEE/ASME Trans. Mechatron.*, July 2007, submitted.

[9] J. C. Alexander and J. H. Maddocks, "On the kinematics of wheeled mobile robots," *Int. J. Robot. Res.*, vol. 8, no. 5, pp. 15–27, 1989.

[10] C. C. de Wit, H. Khennouf, C. Samson, and O. J. Sordalen, "Nonlinear control design for mobile robots," in *Recent Trends in Mobile Robot*, Y. F. Zheng, Ed. World Scientific, 1993, ch. 5, pp. 121–156.

[11] C. Samson, "Velocity and torque feedback control of a nonholonomic cart," in *Proc. in Advanced Robot Control*, C. de Wit, Ed. Springer Verlag, 1990, vol. 162, ch. Int. Workshop in Adaptive and Nonlinear Control: Issue in Robotics.

[12] R. M. Murray and S. S. Sastry, "Steering nonholonomic systesms in chained form," in *Proc. IEEE Conf. on Decision and Control*, Brighton, England, 1991, pp. 1121–1126.

[13] R. W. Brockett, "Nonlinear control theory and differential geometry," in *Proc. International Congress of Mathematicians*, 1983, pp. 1357–1368.

[14] A. M. Bloch, *Nonholonomic Mechanics and Control*. New York, NY: Springer, 2003.

[15] I. Kolmanovsky and N. H. McClamroch, "Developments in nonholonomic control problems," *IEEE Control Syst. Mag.*, vol. 15, no. 6, pp. 20–36, 1995.

[16] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1990, pp. 384–389.

[17] C. Samson and K. Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in Cartesian space," in *Proc. IEEE Int. Conf. Robot. Automat.*, Sacramento, CA, 1991, pp. 1136–1141.

[18] N. Sarkar, X. P. Yun, and V. Kumar, "Control of mechanical systems with rolling constraints: Application to dynamic control of mobile robots," *Int. J. Robot. Res.*, vol. 13, pp. 55–69, 1994.

[19] B. d'Andrea Novel, G. Campion, and G. Bastin, "Control of noholonomic wheeled mobile robots by state feedback linearization," *Int. J. Robot. Res.*, vol. 14, no. 6, pp. 543–559, 1995.

[20] G. Oriolo, A. D. Luca, and M. Vendittelli, "WMR control via dynamic feedback linearization: Design, implementaion, and experimental validation," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 6, pp. 835–852, 2002.

[21] G. C. Walsh, D. Tilbury, S. Sastry, R. Murray, and J. P. Laumond, "Stabilization of trajectory for systems with nonholonomic constraints," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 216–222, Jan. 1994.

[22] J. Yang and J. Kim, "Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots," *IEEE Trans. Robot. Automat.*, vol. 15, no. 3, pp. 578–587, 1999.

[23] D. Chwa, "Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates," *IEEE Trans. Contr. Syst. Technol.*, vol. 12, no. 4, pp. 637–644, 2004.

[24] R. Fierro and F. L. Lewis, "Control of a nonholonomic mobile robot: backstepping kinematics into dynamics," in *Proc. IEEE Conf. on Decision and Control*, New Orleans, LA, December 1995, pp. 3805–3810.

[25] Z. Jiang and H. Nijmeijer, "Tracking control of mobile robots: A case study in backstepping," *Automatica*, vol. 33, no. 7, pp. 1393–1399, 1997.

[26] T. Lee, K. Song, C. Lee, and C. Teng, "Tracking control of unicycle-modeled mobile robots using a saturation feedback controller," *IEEE Trans. Contr. Syst. Technol.*, vol. 9, no. 2, pp. 305–318, 2001.

[27] Z. Jiang and H. Nijmeijer, "A recursive technique for tracking control of nonholonomic systems in chained form," *IEEE Trans. Automat. Contr.*, vol. 44, no. 2, pp. 265–279, 1999.

[28] X. Yun and Y. Yamamoto, "Stability analysis of the internal dynamics of a wheeled mobile robot," *Journal of Robotic Systems*, vol. 14, no. 10, pp. 697–709, 1997.

[29] D. Wang and G. Xu, "Full-state tracking and internal dynamics of nonholonomic wheeled mobile robots," *IEEE/ASME Trans. Mechatronics*, vol. 8, no. 2, pp. 203–214, 2003.

[30] D. Gu and H. Hu, "A stabilizing receding horizon regulator for nonholonomic mobile robots," *IEEE Trans. Robot. Automat.*, vol. 21, no. 5, pp. 1022–1028, October 2005.

[31] H. Chen and F. Allgower, "A quasi-infinite horizon nonlinear model predictive scheme with guaranteed stability," *Automatica*, vol. 14, no. 10, pp. 1205–1217, 1998.

[32] C. Samson, "Time-varying feedback stabilization of car-like wheeled mobile robots," *Int. J. Robotics Research*, vol. 12, no. 1, pp. 55–64, 1993.

[33] ——, "Control of chained systems. Application to path following and time-varying point-stabilizaiton of mobile robots," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 64–77, Jan. 1995.

[34] J. B. Pomet, "Explicit design of time-varying stabilizing control laws for a class of controllable systems withou drift," *Systems and Control Letters*, vol. 18, pp. 147–158, 1992.

[35] A. Teel, R. Murray, and C. Walsh, "Nonholonomic control systems: From steering to stabilization with sinusoids," in *Proc. IEEE Conf. on Decision and Control*, Tucson, AZ, 1992, pp. 1603–1609.

[36] R. Murray and S. Sastry, "Nonholonomic motion planning: steering using sinusoids," *IEEE Trans. Automat. Contr.*, vol. 38, no. 5, pp. 700–716, May 1993.

[37] A. M. Bloch, M. Reyhanoglu, and N. H. McClamroch, "Control and stabilization of nonholonomic dynamic systems," *IEEE Trans. Automat. Contr.*, vol. 37, no. 11, pp. 1746–1757, Nov. 1992.

[38] C. C. de Wit and O. J. Sordalen, "Exponential stabilization of mobile robots with nonholonomic constraints," *IEEE Trans. Automat. Contr.*, vol. 37, no. 11, pp. 1791–1797, 1992.

[39] H. Khennouf and C. C. de Wit, "On the construction of stabilizing discontinuous controllers for nonholonomic systems," in *IFAC Nonlinear Control Syst. Design Symp.*, Tahoe City, CA, 1995, pp. 747–752.

[40] A. Astolfi, "Discontinuous control of nonholonomic systems," *System and Control Letters*, no. 27, pp. 37–45, 1996.

[41] A. Bloch and S. Drakunov, "Stabilization and tracking in the nonholonomic integrator via sliding modes," *Syst. Control Lett.*, vol. 29, pp. 91–99, 1996.

[42] A. Astolfi, "Exponential stabilization of a wheeeled mobile robot via discontinuous control," *J. Dynam. Syst. Meas.*, vol. 121, pp. 121–126, 1999.

[43] R. T. M'Closkey and R. M. Murray, "Exponential stabilization of driftless nonlinear control systems using homogeneous feedback," *IEEE Trans. Automat. Contr.*, vol. 42, no. 5, pp. 614–628, May 1997.

[44] B. Kim and P. Tsiotras, "Controllers for unicycle-type wheeled robots: Theoretical results and experimental validation," *IEEE Trans. Robot. Automat.*, vol. 18, no. 3, pp. 294–307, 2002.

[45] D. Gu and H. Hu, "Receding horizon tracking control of wheeled mobile robots," *IEEE Trans. Contr. Syst. Technol.*, vol. 14, no. 4, pp. 743–749, July 2006.

[46] R. Colbaugh, E. Barany, and K. Glass, "Adaptive control of nonholonomic robotic system," *J. Robotic Systems*, vol. 15, no. 7, pp. 365–393, 1998.

[47] R. Fierro and F. L. Lewis, "Robust practical point stabilization of a nonholonomic mobile robot using neural networks," *Journal of Intelligent Robotic Systems*, no. 20, pp. 295–317, 1997.

[48] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle-like vehicles via Lyapunov techniques," *IEEE Robot. Automat. Mag.*, pp. 27–35, March 1995.

[49] G. Indiveri, "Kinematic time-invariant control of a 2D nonholonomic vehicle," in *Proc. IEEE Conf. on Decision and Control*, Phoenix, AZ, December 1999, pp. 2112–2117.

[50] W. Dixon, D. Dawson, F. Zhang, and E. Zergeroglu, "Global exponential tracking control of mobile robot system via a PE condition," *IEEE Trans. Syst., Man, Cybern. B*, vol. 30, pp. 129–142, Feb. 2000.

[51] W. E. Dixon, D. M. Dawson, E. Zergeroglu, and A. Behal, *Nonlinear control of wheeled mobile robots*. London, UK: Springer, 2001.

[52] P. Morin and C. Samson, "Practical stabilization of a class of nonlinear rsystems: Application to chain systems and mobile robots," in *Proc. IEEE Conf. on Decision and Control*, Sydney, Australia, 2000, pp. 2144–2150.

[53] H. G. Tanner and K. J. Kyriakopoulos, "Discontinuous backstepping for stabilization of nonholonomic mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2002, pp. 3948–3953.

[54] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Trans. Robot. Automat.*, vol. 14, no. 6, pp. 926–939, 1998.

[55] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski, J. Spletzer, and C. J. Taylor, "A vision-based formation control framework," *IEEE Trans. on Robotics and Automation*, vol. 18, no. 5, pp. 813–825, October 2002.

[56] J. P. Desai, J. P. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, Leuven, Belgium, May 1998, pp. 2864–2869.

[57] ——, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Trans. Robot. Automat.*, vol. 17, no. 6, pp. 905–908, 2001.

[58] M. Mesbahi and F. Y. Hadaegh, "Formation flying control of multiple spacecraft via graphs, matrix inequalities, and switching," *AIAA J. of Guidance, Control, and Dynamics*, vol. 24, pp. 369–377, 2001.

[59] H. Tanner, G. Pappas, and V. Kumar, "Leader-to-formation stability," *IEEE Trans. Robot. Automat.*, vol. 20, pp. 443–455, 2004.

[60] P. K. C. Wang and F. Y. Hadaegh, "Coordination and control of multiple microspacecraft moving in formation," *J. Astronautical Sciences*, vol. 44, no. 3, pp. 315–355, 1996.

[61] P. K. C. Wang, "Navigation strategies for multiple autonomous mobile robots moving in formation," *J. of Robotic Systems*, vol. 8, no. 2, pp. 177–195, 1991.

[62] D. Swaroop and J. K. Hedrick, "String stability for a class of nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 41, pp. 349–357, March 1996.

[63] A. Pant, P. Seiler, and K. hedrick, "Mesh stability of look-ahead interconnected systems," *IEEE Trans. Automat. Contr.*, vol. 47, no. 2, pp. 403–407, February 2002.

[64] W. Ren and R. W. Beard, "Formation feedback control for multiple spacecraft via virtual structures," *IEE Proc. on Control Theory and Applications*, vol. 151, no. 3, pp. 357–368, May 2004.

[65] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," in *Proc. IEEE Conf. on Decision and Control*, Orlando, FL, December 2001, pp. 2968–2973.

[66] R. Skjetne, S. Moi, and T. I. Fossen, "Nonlinear formation control of marine craft," in *Proc. IEEE Conf. on Decision and Control*, Las Vegas, NV, December 2002, pp. 1699–1704.

[67] M. A. Lewis and K. H. Tan, "High precision formation control of mobile robots using virtual structures," *Auton. Robots*, vol. 4, pp. 387–403, 1997.

[68] R. W. B. adn J. Lawton and F. Y. Hadaegh, "A feedback architecture for formation control," *Proc. American Control Conference*, vol. 6, pp. 4087–4091, June 2000.

[69] W. Kang, N. Xi, and A. Sparks, "Formation control of autonomous agents in 3D workspace," in *Proc. IEEE Int. Conf. Robot. Automat.*, San Francisco, CA, April 2000, pp. 1755–1760.

[70] R. W. Beard, J. Lawton, and F. Y. Hadaegh, "A coordination architecture for formation control," *IEEE Trans. Contr. Syst. Technol.*, no. 9, pp. 777–790, 2001.

[71] W. Ren and R. W. Beard, "A decentralized scheme for spacecraft formation flying via the virtual structure approach," in *Proc. American Control Conference*, vol. 2, Denver, CO, June 2003, pp. 1746–1751.

[72] E. Lalish, K. A. Morgansen, and T. Tsukamaki, "Formation tracking control using virtual structures and deconfliction," in *Proc. IEEE Conf. on Decision and Control*, San Diego, CA, December 2006, pp. 5699–5705.

[73] Q. Chen and J. Y. S. Luh, "Coordination and control of a group of small mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 1994, pp. 2315–2320.

[74] K. Sugihara and I. Suzuki, "Distributed algorithms for formation of geometric patterns with many mobile robots," *J. Robot. Syst.*, vol. 13, no. 3, pp. 127–139, 1996.

[75] M. Schneider-Fontan and M. J. Han, "Territorial multirobot task division," *IEEE Trans. Robot. Automat.*, vol. 14, pp. 815–822, April 1998.

[76] L. E. Parker, "ALLIANCE: An architecture for fault-tolerant multirobot cooperation," *IEEE Trans. Robot. Automat.*, vol. 14, pp. 220–240, April 1998.

[77] J. Lawton and R. W. Beard, "Synchronized multiple spacecraft rotations," *Automatica*, vol. 38, no. 8, pp. 1359–1364, 2000.

[78] J. R. T. Lawton, R. W. Beard, and B. J. Young, "A decentralized approach to formation maneuvers," *IEEE Trans. Robot. Automat.*, vol. 19, no. 6, pp. 933–941, December 2003.

[79] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Computer Graphics*, vol. 21, no. 4, pp. 25–34, July 987.

[80] A. K. Das, R. Fierro, and V. Kumar, "Control graphs for robot networks," in *Cooperative Control: Models, Applications and Algorithms*, S. Butenko, R. Murphey, and P. Pardalos, Eds. Kluwer Academic Press, January 2003, vol. 1, ch. 4, pp. 55–73.

[81] T. Eren, P. N. Belhumeur, and A. S. Morse, "Closing ranks in vehicle formations based on rigidity," in *Proc. IEEE Conf. on Decision and Control*, Las Vegas, NV, December 2002, pp. 2959–2964.

[82] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Automat. Contr.*, vol. 49, no. 9, pp. 1465–1476, September 2004.

[83] G. Lafferriere, J. Caughman, and A. Williams, "Graph theoretic methods in the stability of vehicle formations," in *Proc. American Control Conference*, vol. 4, Boston, MA, June 2004, pp. 3729–3734.

[84] R. Olfati-Saber and R. M. Murray, "Distributed structural stabilization and tracking for fromations of dynamic multi-agents," in *Proc. IEEE Conf. on Decision and Control*, Las Vegas, NV, December 2002, pp. 209–215.

[85] Z. Jin and R. M. Murray, "Double-graph control strategy of multi-vehicle formations," in *Proc. IEEE Conf. on Decision and Control*, vol. 2, Atlantis, Paradise Island, Bahamas, December 2004, pp. 1988–1994.

[86] W. Dong and Y. Guo, "Formation control of nonholonomic mobile robots using graph theoretical methods," in *Cooperative Systems Control and Optimization*, D. Grundel, R. Murphey, P. Pardalos, and O. Prokopyev, Eds. Springer, 2007, pp. 369–386.

[87] W. B. Dunbar, "Distributed receding horizon control of multiagent systems," Ph.D. dissertation, California Institute of Technology, Pasadena, CA, April 2004.

[88] M. Morari and J. H. Lee, "Model predictive control: past, present and future," *Computers and Chemical Engineering*, vol. 23, pp. 667–682, 1999.

[89] D. Q. Mayne, J. B. Rawings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, June 2000.

[90] J. Richalet, A. Rault, J. L. Testud, and J. Papon, "Algorithmic control of industrial processes," in *Proc. of the 4th IFAC symposium on identification and system parameter estimation*, 1976, pp. 1119–1167.

[91] ——, "Model predictive heuristic control: applications to industrial process," *Automatica*, vol. 14, no. 5, pp. 413–428, 1978.

[92] C. R. Cutler and B. L. Ramaker, "Dynamic matrix control–a computer control algorithm," in *AICHE national meeting*, Houston, TX, April 1979.

[93] ——, "Dynamic matrix control–a computer control algorithm," in *Joint Acutomatic Control Conference*, 1980.

[94] C. R. Cutler, A. Morshedi, and J. haydel, "An industrial perspective on advanced control," in *AICHE national meeting*, Washington, DC, October 1983.

[95] C. E. Garćia and A. M. Morshedi, "Quadratic programming solution of dynmaic marix control (QDMC)," *Chemical Engineering Communications*, vol. 46, pp. 73–87, 1986.

[96] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, pp. 733–764, 2003.

[97] S. Keerthi and E. Gilbert, "Optimal ininite-horizon feedback laws for a general class of constrained discrete-time systems: stability and moving-horizon approximations," *Optimzation Theory and Applications*, pp. 265–293, 1988.

[98] A. Benporad, L. Chisci, and E. Mosca, "On the stabilizing property of the zero terminal state receding horizon regulation," *Automatica*, vol. 30, no. 12, pp. 2013–2015, 1994.

[99] J. B. Rawlings and K. R. Muske, "The stability of constrained receding horizon control," *IEEE transactions on Automatic Control*, vol. 38, no. 10, pp. 1512–1516, 1993.

[100] W. H. Kwon and A. E. Pearson, "A modified quadratic cost problem and feedback stabilization of a linear system," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 838–842, 1977.

[101] P. O. M. Scokaert and J. B. Rawlings, "Constrained linear quadratic regulation," *IEEE Transactions on Automatic Control*, vol. 43, no. 8, pp. 1163–1169, 1998.

[102] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 35, no. 7, pp. 814–824, 1990.

[103] H. Michalska and D. Q. Mayne, "Robust receding horizon control of constrained nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 38, no. 11, pp. 1623–1633, November 1993.

[104] T. H. Yang and E. Polak, "Moving horizon control of nonlinear systems with input saturation, disturbances and plant uncertainty," *International Journal of Control*, vol. 58, no. 4, pp. 875–903, 1993.

[105] S. L. de Oliveira Kothare and M. Morari, "Contractive model predictive control for constrained nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 45, no. 6, pp. 1053–1071, 2000.

[106] V. Nevistic and M. Morari, "Constrained control of feedback linearizable systems," in *Proceeding of the European Control Conference*, Rome, Italy, 1995, pp. 1726–1731.

[107] A. Zheng, "A computationally efficient nonlinear model predictive control algorithm," in *Proc. American Control Conference*, vol. 3, June 1997, pp. 1623–1627.

[108] ——, "Nonlinear model predictive control of the tennessee-eastman process," in *Proc. American Control Conference*, vol. 3, June 1998, pp. 1700–1704.

[109] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra, "Forest fire monitoring with multiple small uavs," in *Proc. American Control Conference*, Portland, OR, June 2005, pp. 3530–3535.

[110] A. Ryan, M. Zennaro, A. Howell, R. Sengupta, and J. K. Hedrick, "An overview of emerging results in cooperative uav control," in *Proc. IEEE Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, December 14-17 2004, pp. 602–607.

[111] J. D. Boskovic, S.-M. Li, and R. K. Mehra, "Semi-globally stable formation flight control design in three dimensions," *Proc. IEEE Conf. on Decision and Control*, pp. 1059–1064, December 2001.

[112] R. Fierro, C. Belta, J. Desai, and V. Kumar, "On controlling aircraft formations," in *Proc. IEEE Conf. on Decision and Control*, Orlando, FL, December 2001, pp. 1065–1070.

[113] M. Pachter, J. J. D'Azzo, and A. Proud, "Tight formation flight control," *Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 246–254, 2001.

[114] J. M. Fowler and R. D'Andrea, "A formation flight experiment," in *IEEE Control Systems Magazine*, vol. 23, Oct 2003, pp. 35–43.

[115] R. C. Nelson, *Flight Stability and Automatic Control*, 2nd ed. McGraw-Hill, 1998.

[116] B. L. Stevens and F. L. Lewis, *Aircraft Control and Simulation*, 2nd ed. Wiley, 2003.

[117] S. Bayraktar, G. E. Fainekos, and G. J. Pappas, "Experimental cooperative control of fixed-wing unmanned aerial vehicles," in *Proc. IEEE Conf. on Decision and Control*, Atlantis, Paradise Island, Bahamas, December 2004, pp. 4292–4298.

[118] I. Cloud Cap Technology, *Piccolo User's Guide*, 2005.

[119] A. Isidori, *Nonlinear Control Systems*, 3rd ed. London: Springer-Verlag, 2001.

[120] M. de Queiroz, D. Dawson, S. Nagarkatti, and F. Zhang, *Lyapunov-Based Control of Mechanical Systems*. Birkhaeuser, Boston: Springer-Verlag, 2000.

[121] R. K. Mehra, J. D. Boskovic, and S. Li, "Autonomous formation flying of multiple ucavs under communication failure," in *Position Location and Navigation Symposium, IEEE 2000*, March 2000, pp. 371–378.

[122] L. Pollini, F. Giulietti, and M. Innocenti, "Robustness to communication failures within formation flight," in *Proc. American Control Conference*, vol. 4, 2002, pp. 2860–2866.

[123] L. Cheng and Y. Wang, "Fault tolerance for communication-based multirobot formation," in *Int. Conf. on Machine Learing and Cybernetics*, vol. 1, 2004, pp. 127–132.

[124] D. Hristu and K. Morgansen, "Limited communication control," *Systems and Control Letters*, vol. 37, pp. 193–205, July 1999.

[125] B. Xian, D. M. Dawson, M. S. de Queiroz, and J. Chen, "A continuous asymptotic tracking control strategy for uncertain nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 49, no. 7, pp. 1206–1211, July 2004.

[126] R. Vidal, O. Shakernia, and S. Sastry, "Formation control of nonholonomic mobile robots with omnidirectional visual servoing and motion segmentation," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 1, Sept. 2003, pp. 584–589.

[127] W. B. Dunbar and R. M. Murray, "Distributed receding horizon control for multi-vehicle formation stabilization," *Automatica*, vol. 2, no. 4, pp. 549–558, 2006.

[128] F. Borrelli, T. Keviczky, K. Fregene, and G. J. Balas, "Decentralized receding horizon control of cooperative vehicle formations," in *Proc. IEEE Conf. on Decision and Control*, Seville, Spain, December 2005, pp. 3955–3960.

[129] W. Li and C. G. Cassandras, "A cooperative receding horizon controller for multi-vehicle uncertain environments," *IEEE Trans. Automat. Contr.*, vol. 51, no. 2, pp. 242–257, Feb. 2006.

[130] A. N. Venkat, J. B. Rawlings, and S. J. Wright, "Stability and optimality of distributed model predictive control," in *Proc. IEEE Conf. on Decision and Control*, Seville, Spain, December 2005, pp. 6680–6685.

[131] K. Wesselowski and R. Fierro, "A dual-mode model predictive controller for robot formations," in *Proc. IEEE Conf. on Decision and Control*, December 2003, pp. 3615–3620.

[132] O. A. A. Orqueda and R. Fierro, "Vision-based nonlinear decentralized controller for unmanned vehicles," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2006, pp. 1–6.

[133] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *IEEE Trans. Automat. Contr.*, vol. 44, no. 3, pp. 648–654, March 1999.

[134] R. Fierro, P. Song, A. K. Das, and V. Kumar, "Cooperative control of robot formations," in *Cooperative Control and Optimization*, ser. Applied Optimization, R. Murphey and P. Pardalos, Eds. Dordrecht: Kluwer Academic Press, 2002, vol. 66, ch. 5, pp. 73–93.

[135] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2002.

[136] D. Cruz, J. McClintock, B. Perteet, O. Orqueda, Y. Cao, and R. Fierro, "Decentralized cooperative control: A multivehicle platform for research in networked embedded systems," *IEEE Control Syst. Mag.*, vol. 27, no. 3, pp. 58–78, June 2007.

VITA

Feng Xie

Candidate for the Degree of

Doctor of Philosophy

Thesis: MODEL PREDICTIVE CONTROL OF NONHOLONOMIC MOBILE ROBOTS

Major Field: Electrical Engineering

Biographical:

Education:

Received the B.S. degree from Zhejiang University, Hangzhou, China, 1997, in Testing Technology and Instrumentation.
Received the M.S. degree from Zhejiang University, Hangzhou, China, 2000, in Control Science and Engineering.
Received the M.S. degree from Oklahoma State University, Stillwater, USA, 2004, in Chemical Engineering.
Completed the requirements for the degree of Doctor of Philosophy with a major in Electrical Engineering Oklahoma State University in December, 2007.

Experience:

Research Assistant, Teaching Assistant, Department of Electrical and Computer Engineering , Oklahoma State University, August 2004 to December 2007.
Research Assistant, Department of Chemical Engineering, Oklahoma State University, August 2002 to July 2004.
Lecturer, Zhejiang University, June 2000 to June 2002.
Research Assistant, Department of Control Science and Engineering, Zhejiang University, September, 1997 to March, 2000.

Professional Memberships: Institute of Electrical and Electronics Engineers (IEEE)

Name: Feng Xie Date of Degree: December, 2007

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: MODEL PREDICTIVE CONTROL OF NONHOLONOMIC MOBILE ROBOTS

Pages in Study: 153 Candidate for the Degree of Doctor of Philosophy

Major Field: Electrical Engineering

In this work, we investigate the possibility of using model predictive control (MPC) for the motion coordination of nonholonomic mobile robots. The contributions of this dissertation can be summarized as follows.

- A robust formation controller is developed for the leader-following formation of unmanned aerial vehicles (UAVs). With the assumption that an autopilot operating in *holding mode* at the low-layer, we present a two-layered hierarchical control scheme which allows a team of UAVs to perform complex navigation tasks under limited inter-vehicle communication. Specifically, the robust control law eliminates the requirement of leader's velocity and acceleration information, which reduces the communication overhead.
- A dual-mode MPC algorithm that allows a team of mobile robots to navigate in formations is developed. The stability of the formation is guaranteed by constraining the terminal state to a terminal region and switching to a stabilizing terminal controller at the boundary of the terminal region. With this dual-mode MPC implementation, stability is achieved while feasibility is relaxed.
- A *first-state contractive* model predictive control (FSC-MPC) algorithm is developed for the trajectory tracking and point stabilization problems of nonholonomic mobile robots. The stability of the proposed MPC scheme is guaranteed by adding a first-state contractive constraint and the controller is exponentially stable. The convergence is faster and no terminal region calculation is required. Tracking a trajectory moving backward is no longer a problem under this MPC controller. Moreover, the proposed MPC controller has simultaneous tracking and point stabilization capability.

Simulation results are presented to verify the validity of the proposed control algorithms and demonstrate the performance of the proposed controllers.

ADVISOR'S APPROVAL: Dr. Rafael Fierro