VISION-BASED CONTROL OF MULTI-AGENT SYSTEMS

By

OMAR ARMANDO ADRIÁN ORQUEDA

Electronic Engineer
Universidad Nacional del Sur
Bahía Blanca, Argentina
1996

Doctor on System Control
Universidad Nacional del Sur
Bahía Blanca, Argentina
2006

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, 2006

VISION-BASED CONTROL OF

MULTI-AGENT SYSTEMS

Dissertation Approved:

Prof. Martin Hagan

Committee Chair

Prof. Rafael Fierro

Thesis Adviser

Prof. Carlos Oliveira

Prof. Weihua Sheng

Dr. Gordon Emslie

Dean of the Graduate College

To my children Paulina and Tomás.

Acknowledgements

It has been a long journey since I started doing research. During this journey, I have had the pleasure of working with several researchers and learning a lot from all of them. Consequently, the list of acknowledgements is very extensive. Nevertheless, I would like to express my special gratitude to two professors from my home university, Alfredo Desages (in memoriam) and Sylvia Padin. Both strongly influenced my decision for doing research when still I was an undergraduate student.

I would like to thank my advisor Dr. Rafael Fierro for his support and encouragement. In particular, I am grateful for the freedom and flexibility I had to look for an approach on formation control in which I was especially interested. I am also extremely thankful for giving me the opportunity of participating in this project at Oklahoma State University.

I would like to thank the members of my thesis committee Profs. Hagan (chair), Oliveira, and Sheng, for reading my dissertation and suggesting very constructive comments which help improve the contents of this work.

I would like to thank Dr. Tony Zhang for his friendship and helpful discussions on the first formulation of the output feedback controller presented in Chapter 5. Many thanks to current and past members of MARHES lab, James, Brent, Carlo, Feng, Eric, Ugur, Colby, Kyle, Chris, Lorne, Justin, Kenny, José, my friends in Stillwater and in Argentina.

I would like to thank my family, specially, my two children Paulina and Tomás. They encourage me everyday to do my best work.

Last but not least, I gratefully acknowledge the sources of my financial support: This research was supported by the National Science Foundation (NSF) under grants #0311460 and CAREER #0348637, and by the U.S. Army Research Office under Grant DAAD19-03-1-0142 (through the University of Oklahoma).

Preface

Creating systems with multiple autonomous vehicles places severe demands on the design of cooperative control schemes and communication strategies. In last years, several approaches have been proposed in the literature. Most of them solve the vehicle coordination problem assuming some kind of communications between team members. Communications are used either to obtain information about neighbors' velocities or relative positions, or to acquire the control law to apply. In the former case, the control computation is decentralized and the later the control is centralized. However, communications make the group sensitive to failure, or restrict the applicability of the controllers to teams of friendly robots.

This dissertation is a step forward toward the design of sensor-based decentralized architectures for stabilization of *unmanned vehicles moving in formation*. The architecture consists of two main components: (*i*) a model-based *vision system*, and (*ii*) *a control algorithm*. The model-based vision system can recognize and relatively localize neighbor robots using fiducial markers on a truncated octagon shape mounted on each vehicle. It is composed of three main components: image acquisition and processing, robot identification, and pose estimation.

The control algorithm uses vision information to stabilize a group of mobile robots in formation. Two main stabilization problems are addressed: stabilization in leader-follower and in two-leader-follower formations. Several control strategies using relative pose between a robot and its designated leader or leaders are presented. These strategies ensure asymptotic coordinated motion using different information levels to implement the controllers.

The first strategy is a *partial state feedback nonlinear approach* that requires full knowledge of leader's velocities and accelerations. The second strategy is a *robust partial state feedback nonlinear approach* that requires knowledge of the rate of change of the relative position error. Finally, the third strategy is an *output feedback approach* that uses high-gain observers to estimate the derivative of the unmanned vehicles' relative position. In consequence, this last algorithm only requires knowledge of

the leader-follower relative distance and bearing angle. Both data are computed using measurements from a single camera, eliminating sensitivity to information flow between vehicles. Furthermore, because the leader's exact trajectory is uncertain, this approach can be applied to both the problem of tracking a given trajectory maintaining a desired formation shape or to pursuit-evasion problems, where the evader trajectory is assumed unknown. This is a distinctive aspect of this vision-based architecture with respect to the current state-of-the-art.

Lyapunov's stability theory-based analysis and numerical simulations in a realistic tridimensional environment show the stability properties of the control approaches.

Finally, we describe our ongoing implementation on virtual and real platforms and show simulation results to verify the validity of the designed methodologies.

Contents

Chapter **Page**

List of Figures **xiv**

I Introduction **1**

1 Motivation **2**
1.1 Introduction 2
1.2 Formations 6
1.3 Related work 9
1.4 Outline 11
1.5 Statement of contributions 12

2 Multi-robot formations **13**
2.1 Mathematical model of the robots 13
2.1.1 Leader-follower polar model 14
2.2 Graph theory 16
2.3 Formal definition of formation 17

3 Nonlinear estimation **19**
3.1 Optimal recursive estimation 20
3.2 The extended Kalman filter (EKF) 21
3.2.1 Simulation example 22
3.3 The unscented Kalman filter (UKF) 24
3.3.1 Simulation example 28
3.4 UKF for parameter estimation 30
3.4.1 Simulation example 30
3.5 High-gain observers (HGO) 32

3.5.1 Simulation example . . . 37
3.6 Higher-order sliding mode (HOSM) observers . . . 37
3.6.1 Simulation example . . . 43

II Main Contributions **49**

4 Mobile robot vision **50**
4.1 System overview . . . 52
4.2 Marker and ID detection . . . 57
4.2.1 Image capture . . . 57
4.2.2 Grayscale conversion . . . 57
4.2.3 Thresholding: The Otsu method . . . 57
4.2.4 Contour extraction and selection . . . 61
4.2.5 ID recognition . . . 61
4.3 Pose estimation . . . 64
4.3.1 Camera model . . . 64
4.3.2 OpenGL camera model . . . 67
4.3.3 Pose estimation algorithms . . . 68
4.3.3.1 Pose from orthography and scaling with iteration (POSIT) method . . . 68
4.3.3.2 Projection ray attraction (PRA) method . . . 72
4.3.3.3 Lowe's method . . . 73
4.4 Nonlinear filtering . . . 77
4.4.1 Mathematical model of the robots . . . 77
4.4.2 Nonlinear observability . . . 78
4.4.3 The dual unscented Kalman filter (DUKF) . . . 80
4.5 Simulation results . . . 81
4.6 Summary . . . 94

5 Vision-based formation control **99**
5.1 Formation mathematical models . . . 100
5.1.1 Leader-follower mathematical model revisited . . . 100
5.1.1.1 Second order leader-follower model with $d = 0$ . . . 102
5.1.1.2 Second order leader-follower model with $d \neq 0$ . . . 104
5.1.2 Two leader-follower mathematical model . . . 105
5.1.2.1 Second order two-leader-follower model with $d = 0$ . . . 108

5.1.2.2 Second order two-leader-follower model with $d \neq 0$. 111
5.1.3 Generic model formulation 113
5.2 The formation control problem 114
5.2.1 Leader-follower control problem 114
5.2.2 Two-leader-follower control problem 116
5.2.3 Formation dynamics, communication and sensing models 118
5.3 Formation control algorithms 118
5.3.1 Partial state feedback formation control (PSFB) 121
5.3.2 Robust state feedback formation control (RSFB) 121
5.3.3 Output feedback formation control algorithm (OFB) 122
5.4 Backstepping 124
5.5 Simulation results 126
5.6 Summary 149

III Conclusions **150**

6 Summary and future work **151**

Bibliography **152**

IV Appendices **167**

A Projective geometry **168**

B Proofs of lemmas and theorems **171**
B.1 Proof of Theorem 5.3.1 171
B.2 Lemma B.2.1 172
B.3 Proof of Theorem 5.3.3 174

C Definitions and mathematical background **178**
C.1 Preliminaries 178
C.2 Controllability and observability 181
C.2.1 Observability 181
C.3 Input-to-state stability 182

D Perturbations, relative dynamics and degrees **184**
D.1 Leader-follower model with uncertain transformation 184
D.2 Internal dynamics 185

List of Figures

Figure **Page**

1.1 Bats, locusts, and ants. 3
1.2 Aircraft formations. 4
1.3 Typical formation shapes: (a) line, (b) column, (c) diamond, (d) wedge [4]. 6
1.4 Formation positions: (a) unit-center-referenced, (b) leader-referenced, (c) neighbor-referenced [4]. 7

2.1 Unicycle-type robots. (a) ERSP Scorpion (Evolution Robotics), (b) MARHES TXT platform. 14
2.2 Formation geometry. 15

3.1 State estimation with the EKF. 24
3.2 State estimation and control with the EKF. 25
3.3 State estimation with the UKF. 28
3.4 State estimation and control with the UKF. 29
3.5 Parameter estimation with the UKF. 32
3.6 Parameter estimation with the UKF and noisy output. 33
3.7 HGO with exact control law. 38
3.8 HGO behavior with unsaturated inputs. 39
3.9 HGO behavior with saturated inputs. 40
3.10 HGO behavior with saturated inputs and noise. 41
3.11 Simulation results with a first order HOSM. 44
3.12 Simulation results with a second order HOSM. 46
3.13 Simulation results with a second order HOSM and measurement noise. 47
3.14 Closed-loop simulation results with a second order HOSM. 48

4.1 Scheme of the TXT platform in MPSLab. 53
4.2 Synthetic camera screen shot. 53

4.3 Logitech 3000 USB camera (bottom-left), PointGrey's Bumblebee (top), and Unibrain's Fire-I (bottom-right) FireWire IEEE-1394 cameras. . 54
4.4 Vision processing flow chart. 55
4.5 Camera and robot frames. 56
4.6 Processing sequence of the octagon shape. 58
4.7 Grayscale conversion. (a) Color image, (b) red channel, (c) green channel, (d) blue channel, and (e) mean value image. 59
4.8 Fiducial marker ID computation. 61
4.9 Projective transformation. 63
4.10 Pinhole camera model. 65
4.11 OpenGL camera frustum [119]. 67
4.12 POSIT projections. 69
4.13 Dual estimation problem. 80
4.14 Identifications: Robot 1 (left) IDs 2, 3, 4, 5; robot 2 (right) IDs 1, 2, 3. 81
4.15 Identifications: Robot 1 (right) IDs 2, 3, 4; robot 2 (left) IDs 2, 3, 4. . 81
4.16 Identifications: Robot 1 (right) IDs 2, 3, 4; robot 2 (left) IDs 1, 2. . . 82
4.17 Pure translation using Lowe's method. 82
4.18 Pure rotation using Lowe's method. 82
4.19 Pure translation. Error and number of iterations using PRA method. 83
4.20 Pure translation. Error and number of iterations using PRA method and a priori information. 84
4.21 Pure rotation. Error and number of iterations using PRA method and a priori information. 84
4.22 Pure translation. Error and number of iterations using Lowe's method. 85
4.23 Pure rotation. Error and number of iterations using Lowe's method. . 85
4.24 Pure translation. Error and number of iterations using POSIT algorithm. 86
4.25 Pure rotation. Error and number of iterations using POSIT algorithm. 86
4.26 Pose estimation with $q_1 = [1, -0.25, 0]^T$, $q_2 = [2, 0.5, 0]^T$. 87
4.27 Pose estimation with $q_1 = [2.5, -0.5, 0]^T$, $q_2 = [3, 0.5, 0]^T$. 87
4.28 Pose estimation with $q_1 = [1.5, -0.5, 0]^T$, $q_2 = \left[4, 0.5, -\frac{\pi}{2}\right]^T$. 88
4.29 Fog effect on pose estimation. 89
4.30 Computational load with one robot in the field of view (blue line) and three robots plus another entering at $t = 1.25$ red line). 90
4.31 Relative distance and bearing estimation using a DUKF. 91
4.32 Velocity estimation using a DUKF. 92
4.33 Velocity estimation using a DUKF with noise injection. 93

4.34 Control, relative distance and bearing estimation using a DUKF. . . . 95
4.35 Control and velocity estimation using a DUKF. 96
4.36 Control, relative distance and bearing estimation using a DUKF with noise injection. 97
4.37 Control and velocity estimation using a DUKF with noise injection. . 98

5.1 Leader-follower formation geometry. 100
5.2 Two leader-follower formation geometry. 106
5.3 Leader-follower dynamic system: (a) complete system, (b) compact form. 119
5.4 Leader-follower (PSFB controller). 119
5.5 Leader-follower (RSFB controller). 120
5.6 Leader-follower (OFB controller). 120
5.7 Robot trajectories for a straight line path and $d = 0$. 127
5.8 Velocity comparison between controllers (5.38), (5.40), and (5.52) for a straight line path and $d = 0$. 128
5.9 Error comparison between controllers (5.38), (5.40), and (5.52) for a straight line path and $d = 0$. 129
5.10 Control effort comparison with $\varepsilon = 0.1$, 0.001 for a straight line path and $d = 0$. 130
5.11 Output error comparison with $\varepsilon = 0.1$, 0.001 for a straight line path and $d = 0$. 131
5.12 Snake view of a simulation with 3 robots for a straight line path and $d = 0$. 132
5.13 Trajectories of the leader robot and two followers for a circular path and $d = 0$. 133
5.14 Velocity comparison between controllers (5.38), (5.40), and (5.52) for a circular path and $d = 0$. 134
5.15 Output error comparison between controllers (5.38), (5.40), and (5.52) for a circular path and $d = 0$. 135
5.16 Control effort comparison with $\varepsilon = 0.1$, 0.001 for a circular path and $d = 0$. 136
5.17 Output error comparison with $\varepsilon = 0.1$, 0.001 for a circular path and $d = 0$. 137
5.18 Snake view of a simulation with 3 robots for a circular path and $d = 0$. 138
5.19 Trajectories of the robots with $d = 0$. 139
5.20 Velocities of robot 5 using the two-leader algorithm with $d = 0$. . . . 140

5.21 Output error of robot 5 using the two-leader algorithm with $d = 0$. . . 141
5.22 Snake view of a simulation with 7 robots with $d = 0$. 142
5.23 Specifications for 7 robots. 142
5.24 Trajectories of the robots for a straight line path and $d \neq 0$. 143
5.25 Error comparison for a straight line path and $d \neq 0$. 144
5.26 Snake view of a simulation with 7 robots for a straight line path with $d \neq 0$. 145
5.27 Trajectories of the robots for a circular path and $d \neq 0$. 146
5.28 Error comparison for a circular path and $d \neq 0$. 147
5.29 Snake view of a simulation with 7 robots for a circular path with $d \neq 0$. 148

A.1 Cross-ratio invariance. 169
A.2 Vanishing points. 170

Part I

Introduction

Chapter 1

Motivation

1.1 Introduction

In the last decade, cooperative multi-robot systems have been increasingly being considered as a mean of performing complex tasks within dynamic environments. These complex and extremely diverse tasks include applications such as automated transportation, spacecraft interferometry, surveillance, mapping, border patrol, search and rescue, disaster relief, unmanned air combat, multi-targeting and multi-platform operations, military battle systems, highway and air transportation systems, mobile sensor networks, and wireless surveillance. The main reason for engaging multi-robot systems in collective behavior is their expected outperformance over single-robot systems in size, cost, flexibility, and fault tolerance [43, 90].

The design of multi-robot formations was initially inspired by the emergent self-organization from social interaction observed in nature. For example, bees, birds, fish, wildebeests, and human beings flock or swarm in particular directions [120], see Figure 1.1. Moreover, it is well-known that aircraft formations fly in a 'V' to decrease drag, save energy, and increase safety, Figure 1.2.

As Camazine et al. pointed out [13]:

> "Self-organization is a process in which pattern at the global level of a system emerges solely from numerous interactions among the lower-level components of the system. Moreover, the rules specifying interactions among the system's components are executed using only local information, without reference to the global pattern. In short, the pattern is an emergent property of the system, rather than a property imposed on the system by an external ordering influence ... pattern is a particular,

Figure 1.1: Bats, locusts, and ants.

Figure 1.2: Aircraft formations.

organized arrangement of object in space or time."

Single individuals may use relatively simple behavioral rules to generate structures and patterns at the collective level that are more complex than the components and processes from which they emerge. For instance, in the mid-40s Walter, Wiener and Shannon observed complex social behavior in the interaction between simple turtle robots equipped with light and touch sensors [14, 112].

Therefore, to understand self-organization and use it in a productive fashion, two main questions have to be answered: The first question is related to the problem of how global behaviors can be fostered using local and relatively simple rules. The second question is related to the problem of which sensing capability needs each individual.

Regarding to the first question, studies on flocking and herding mechanisms indicate that they emerge as a combination of a desire to stay in the group and yet simultaneously keep a minimum separation distance from other members of the flock [96]. In [52], Tu and Toner prove that this combination has to be complemented with motion, by proving that group alignment is not possible in flocks and herds with local perception in the absence of movement. According to recent models from theoretical physics [51, 137], a key factor is the density of animals in the group. If this density increases, a rapid transition occurs from disordered movement of individuals to highly aligned collective motion. Two of the most significant advances in collective motion and synchronization are the models conceived by Vicsek et al. [137] and the Kuramoto [63, 64, 111]. Vicsek designed a discrete-time and stochastic model that analyzes with simple interactions the emergence of different behavior patterns. Kuramoto's model was originally developed for oscillator synchronization, it is a continuous-time and deterministic approach very suitable for analysis of collective behavior emergence without noise.

Regarding to the second question, nature also give guidelines on which and how many sensors are necessary to achieve coordinate behavior. For example, bats sense their surroundings using ultrasound. Birds, mammals, and most insects use vision as their main sensor. In Figure 1.2, the pilot can maintain formation by knowing the relative distance, heading, and bearing between airplanes.

In summary, it can be stressed that members of flocks of birds or schools of fish do not use explicit communication but local sensing, usually vision, in order to maintain a coherent formation or a coordinated motion, even when they have to move around obstacles or avoid predators. This dissertation finds its main motivation in solving the local control problem using vision-based sensing, showing that a stable local behavior

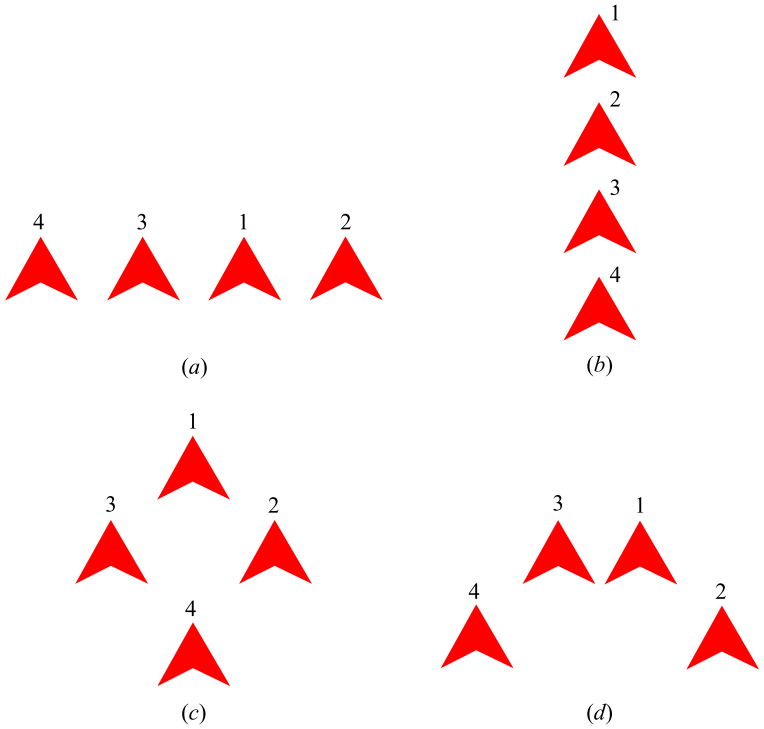


Figure 1.3: Typical formation shapes: (a) line, (b) column, (c) diamond, (d) wedge [4].

can be achieved and maintained without using communications.

1.2 Formations

Intuitively, a *formation is a group of agents maintaining a determined relationship*. The two most important characteristics of a formation are its *shape* and its *position*. The most common formation shapes are *line*, *column*, *diamond*, and *wedge* [4], as shown in Figure 1.3.

Common definitions of formation positions are *unit-center referenced*, *leader referenced*, and *neighbor referenced* [4]. In a *unit-center-referenced* position, all the robots maintain a relative position to a center point given by the average of the coordinates p_x and p_y of each member of the formation. In a *leader-referenced* position the robots maintain a relative position with respect to a leader robot. Finally, in a *neighbor-referenced* position, each robot maintains a position relative to one other predetermined robot. Figure 1.4 depicts these three types of formation position definitions.

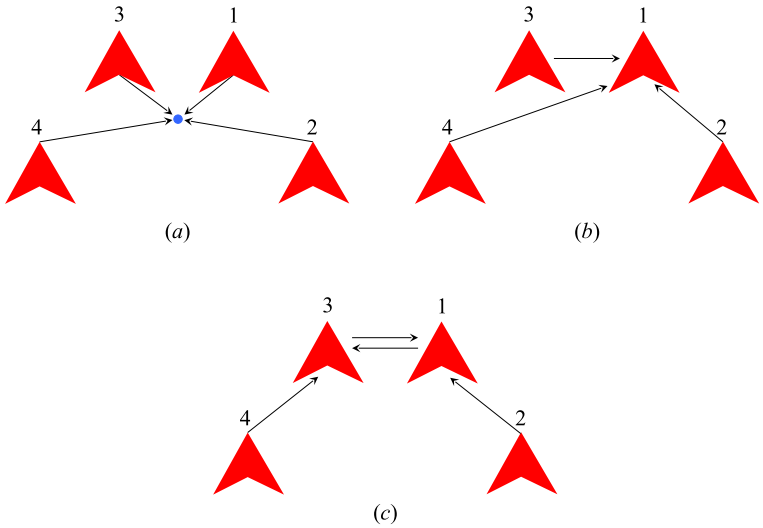


Figure 1.4: Formation positions: (a) unit-center-referenced, (b) leader-referenced, (c) neighbor-referenced [4].

Types of formations

There exist several types of formations. Here we resume some of the more relevant to our work.

Flocking

The earliest application of artificial formation behavior starts with the pioneering work of Craig Reynolds [115]. In his work, the author addresses the problem of simulating flocks of birds and schools of fish with a simple egocentric behavioral model. This model consists of collision avoidance, velocity matching, and formation keeping components. Also, three heuristic rules are defined:

- *cohesion rule* - aim of keeping proximity to nearby flock-mates,
- *separation rule* - desire of collision avoidance,
- *alignment rule* - intention of velocity matching with neighbors.

These three rules are inherently local and give to each member the possibility of navigating using only its sensing capabilities. From a mathematical point of view, they allow to pose the flocking problem as a decentralized control problem. Moreover, the superposition of these three rules results in all agents moving in a loose (as opposite to rigid) formation, with a common heading while avoiding collisions [128, 129].

The work of Reynolds led to the creation of the first computer animated flocking. Several films and video have been produced using flocking, such as "Stanley and Stella in: Breaking the Ice" (1987), "Batman Returns" (1992), "Cliffhanger" (1993), "The Lion King" (1994), "From Dusk Till Dawn" (1996), "The Hunchback of Notre Dame" (1996), and "Finding Nemo" (2003).

Behavior-based formation

The main idea of behavior-based formations is to integrate several goal oriented behaviors with a given arbiter that chooses between them. It is a highly decentralized reactive framework. In [4] for example, the authors define two steps for formation maintenance: (*i*) determination of the robot's position in formation; and (*ii*) generation of motor commands to direct the robot toward the correct location. A global arbiter gives priorities to each behavior depending on environmental conditions.

The method performs well in simulations and experiments, but it lacks mathematical analysis of convergence and robustness.

Leader-following formation

Leader following can be centralized or decentralized. When a decentralized approach is used, each robot can drive itself keeping a desired distance to some of its neighbors. This scheme typically has one leader which takes care of the assignment of followers' relative positions and of the global mission objective, for instance, obstacle avoidance [33, 34, 125].

The major weakness of single-leader architectures is that they have a single point of failure [73]. Therefore, stability properties of these architectures under changes of leaders is a major concern. Some attempts have been made in this field. For instance, in [27, 28], the architecture has a high level layer that detects leader failures and redefines the formation control graph according to some given strategy. Failures include robot malfunctioning, communication errors (in communication-based formations), and information flow breaches.

Virtual structure

Virtual structure approaches describe the entire formation as one single rigid body or pattern. In [3], for example, a *virtual structure* or *pattern formation* is defined as the problem of coordination a group of robots to get into and maintain a formation with

a certain shape such as a wedge or a line, assuming the existence of a communication channel between the central unit and the individual robots.

These approaches are centralized and, consequently, they are very susceptible to operation errors.

1.3 Related work

Creating systems with multiple autonomous vehicles working together to achieve a common mission within a changing environment places severe demands on the design of decision-making supervisors, cooperative control schemes, and communication strategies. Research on multi-vehicle system coordination has been focused both on *centralized* and *decentralized* control strategies. Centralized control strategies have the advantage of being able to reach a global optimum solution for tasks such as path planning and reconfiguration [8, 12, 108, 143]. However, centralized algorithms become infeasible when the number of vehicles and constraints increase, preventing their implementation in real-time. On the other hand, decentralized control approaches only require local information and can effectively achieve multi-vehicle coordination behaviors as the ones observed in nature [28, 27, 63, 72, 73, 88].

Many approaches for solving multi-robot coordination reduce the general control problem to a single-agent control one by assuming that global communication of some coordination information is available. However, a coordination mechanism that does not rely on global communication ensures flexibility and mission safety, because reference trajectories and mission objectives should not be shared among all agents but to some leaders [20]. Of course, this poses the challenge of designing new formation control paradigms, robust and computationally simple, that can perform well in the presence of uncertainty in leaders' current and future states.

Other authors have addressed the coordination problem of multiple unmanned vehicles using optimization techniques [10]. Contributions in this area include work focused on autonomous distributed sensing tasks [25], decentralized optimization-based control algorithms [41], optimal motion planning [9], and formation reconfiguration planning [143]. More recently, the use of model predictive control (MPC) or receding-horizon control (RHC) is becoming popular in the multi-robot system literature [12, 36, 46, 68, 105]. Generally speaking, MPC algorithms rely on the optimization of a predicted model response with respect to the system input to determine the best input changes for a given state. Either hard constraints (that cannot be violated) or soft constraints (that can be violated but with some penalty) can be

incorporated into the optimization problem, giving to MPC a potential advantage over passive state feedback control laws. However, there are possible disadvantages to be considered. For instance, in its traditional use for process control, the primary disadvantage of MPC is the need of a good model of the plant, but such model is not always available. In robotics applications, the foremost disadvantage is the computational cost, negligible for slow-moving systems in the process industry, but very important in real-time applications. In [105], a path-space iteration (PSI) algorithm is designed to solve the motion coordination problem. In this work, the authors present centralized and decentralized algorithms based on an MPC/PSI formulation to solve nonholonomic multi-robot coordination problems in dynamic, unknown environments.

Chen *et al.* [20, 18, 19] develop a decentralized control architecture that employs local sensor information and an internal model approach [60, 61] to handle uncertainties in the leader reference trajectory. The information needed to implement the controller is relative position and velocity between a robot and its leader. This approach uses system immersion to model the known leader's movements and communications to acquire leader's heading. Leader velocities are piecewise linear functions of time over finite intervals of variable length. Inter-vehicle collision avoidance is assured by confining vehicles to specific sectors during transients.

In the last few years, some vision-based motion coordination algorithms have been developed in the literature. In [138], authors design a formation control algorithm based on omni-directional visual servoing and motion segmentation. Vision-based formation controllers are described in [26]. The algorithms use input-output linearization and require the estimation of leader-follower relative angle and leader's linear and angular velocities. Two algorithms are described for image-based leader-follower formation control, one is based on feedback linearization and the other combines Luenberger observers with a linear controller. In [85], the authors give a sufficient condition for observability using a vision-based centralized controller. The control law is based on input-output feedback linearization and assumes that the robots have omni-directional cameras and that the leader can transmit the velocity control to each follower and estimate their states with an extended Kalman filter.

Another vision-based formation controller has been recently developed in [88, 89]. This distributed coordination approach is based on nearest-neighbor interactions, assuming that robots move with constant linear speed and achieve flocking after a given time. In general, flocking algorithms do not maintain a strict formation shape. Such formation maintenance is critical in applications such as cooperative payload

transport [126], cooperative object pushing [50], and distributed sensor deployment when robots move forming certain geometric pattern [142].

In this dissertation, we present different approaches for solving a multi-robot coordination problem using monocular vision. The first part of the dissertation presents an artificial vision system for pose estimation using fiducial marks [103, 104]. The second part presents several vision-based formation control algorithms for leader following and two-leaders following architectures [106, 102, 101]. The first two are state feedback-based controllers that require total or partial knowledge of the state variables, respectively. The third is a robust output feedback decentralized controller based only on monocular vision information about the relative motion between a robot and its designated leader or leaders. A high-gain observer is used to estimate the derivatives of the measurements. This last algorithm eliminates the need of inter-vehicle communications which increases the reliability of the overall system, making this approach suitable for pursuit-evasion problems, when the pursued is an unfriendly robot. Finally, we also describe our ongoing implementation on virtual and real platforms.

1.4 Outline

The rest of the dissertation is organized as follows. In Chapter 2, we review some definitions on graph theory and robot formations that are used on the main two chapters of this dissertation. Chapter 3 resumes four nonlinear estimation algorithms and compare their properties through simple examples. Chapter 4 presents the implementation of real and virtual vision systems for robot identification and localization. Chapter 5 describes and analyzes vision-based decentralized formation control algorithms. Finally, we present our concluding remarks and future work in Chapter 6.

1.5 Statement of contributions

This dissertation is a step forward toward the design of sensor-based decentralized architectures for stabilization of *unmanned vehicles moving in formation*. The contributions can be divided into two main parts:

- A vision system for robot identification and localization.
- A set of robust vision-based formation controllers.

The vision system, composed of image acquisition and processing, robot identification, and pose estimation, can recognize and relatively localize neighbor robots using fiducial markers.

The set of robust vision-based formation controllers allows to maintain relative distance and bearing in leader-follower and two-leader-follower formations. These controllers ensure asymptotic coordinated motion coordination using different information and sensing levels.

To the author's knowledge, the whole architecture is the first framework that applies a realistic sensor system and control algorithms to the formation control problem with guaranteed stability and without communications.

Chapter 2

Multi-robot formations

In this Chapter, some basic concepts relevant to multi-robot formations are summarized. We briefly review the dynamical model of the robot and the polar relationship between a leader and its follower, that is, the *basic cell* of a formation. We resume some tools for formation characterization using graph theory and the formal definition of a formation. The reader is referred to the bibliography for a more detailed treatment [35, 133].

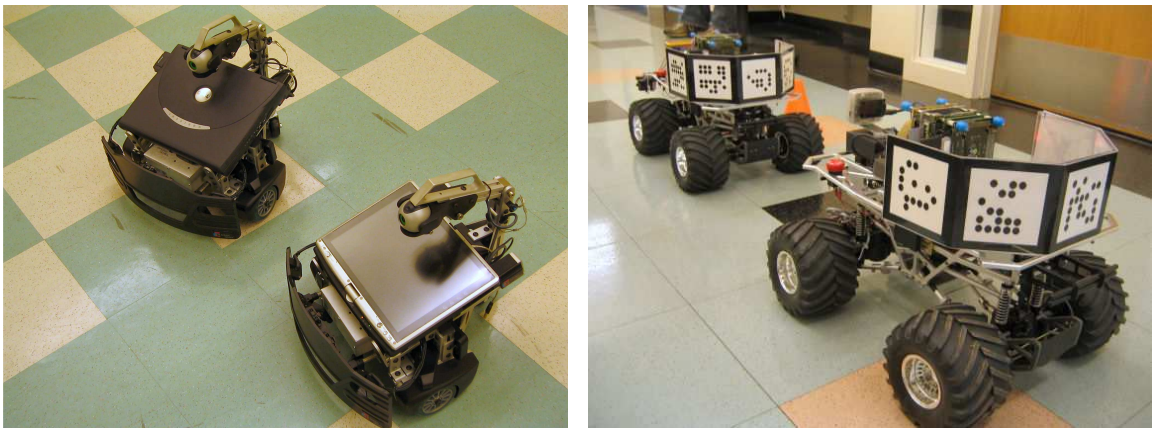
2.1 Mathematical model of the robots

Let us consider a multi-robot system composed of N_a agents, as the ones shown in Figure 2.1, modelled as unicycle-like vehicles[1] Σ_i moving on the plane, with the dynamic model of the ith robot given by [20]

$$\begin{aligned} \dot{x}_i(t) &= v_i(t)\cos\theta_i(t), \\ \dot{y}_i(t) &= v_i(t)\sin\theta_i(t), \\ \dot{\theta}_i(t) &= \omega_i(t), \\ m_i\dot{v}_i(t) &= F_i(t), \\ J_i\dot{\omega}_i(t) &= T_i(t), \end{aligned} \tag{2.1}$$

where $q_i(t) = [x_i(t), y_i(t), \theta_i(t)]^T \in SE(2)$, is the configuration vector with respect to an inertial reference frame, $(x_i, y_i) \in \mathbb{R}^2$ denotes the position in Cartesian coordinates and is the intersection of the axis of symmetry with the driving wheels axis, $\theta_i \in \mathbb{S} = (-\pi, \pi]$ is the heading angle, $u_i(t) := [v_i(t), \omega_i(t)]^T \in \mathcal{U}_i \subseteq \mathbb{R}^2$, $v_i(t)$,

[1] Note that other mathematical models can be adapted to this framework.



(a) (b)

Figure 2.1: Unicycle-type robots. (a) ERSP Scorpion (Evolution Robotics), (b) MARHES TXT platform.

$\omega_i(t)$ are the linear and the angular velocities, respectively, $\mathcal{U}_i$ is a compact set of admissible velocities containing the origin, and $F_i(t)$, $T_i(t) \in \mathbb{R}$ are the admissible control signals. Without lost of generality, we assume that all sets $\mathcal{U}_i$ are equal, then

$$\mathcal{U} = \mathcal{U}_i := \{(v_i, \omega_i) | \, |v_i| \leq v_{\max}, \, |\omega_i| \leq \omega_{\max}\},$$

and that the masses and inertia moments are equal to one $m_i = J_i = 1$, with $i = 1, \ldots, N_a$.

As can be seen in Figure 2.1, a monocular camera is mounted on each robot. In an ERSP Scorpion robot, the camera is mounted on the intersection of the axis of symmetry with the driving wheels axis, see Figure 2.1a. In a MARHES TXT platform, the camera is mounted d units ahead on the axis of symmetry, see Figure 2.1b.

2.1.1 Leader-follower polar model

Let the Euclidean distance $\ell_{ik}(t) \in \mathbb{R}_{\geq 0}$ and the angles $\alpha_{ik}(t)$, $\theta_{ik}(t) \in \mathbb{S} = (-\pi, \pi]$ between robots i and k be defined as

$$\ell_{ik}(t) = \sqrt{(x_i(t) - x_k^c(t))^2 + (y_i(t) - y_k^c(t))^2}, \tag{2.2}$$

$$\alpha_{ik}(t) = \zeta_{ik}(t) - \theta_i(t), \tag{2.3}$$

$$\theta_{ik}(t) = \theta_i(t) - \theta_k(t), \tag{2.4}$$

where $\zeta_{ik}(t) = \text{atan2}(y_i(t) - y_k^c(t), x_i(t) - x_k^c(t))$, and $x_k^c(t) = x_k(t) + d\cos\theta_k(t)$, $y_k^c(t) = y_k(t) + d\sin\theta_k(t)$ are the coordinates of the camera, as shown in Figure 2.2.

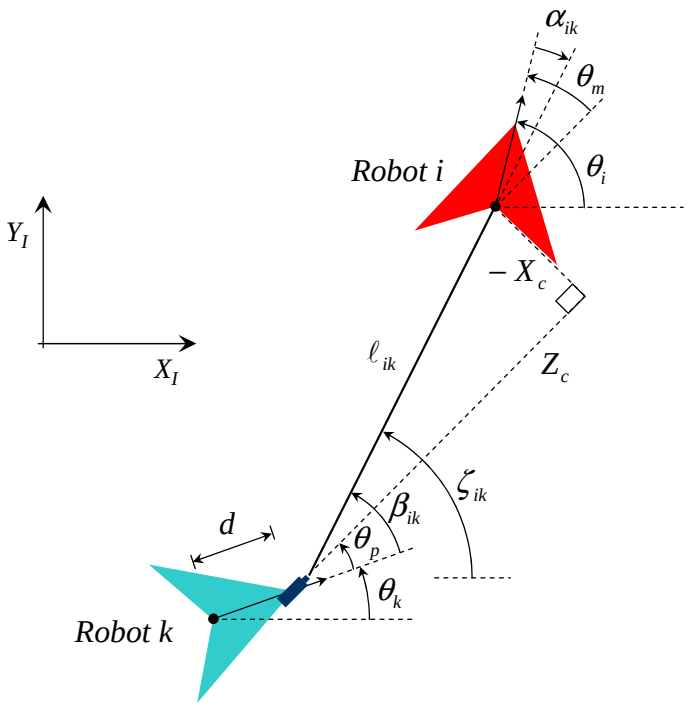


Figure 2.2: Formation geometry.

Then, assuming that both robots are described by the model (2.1), the leader-follower system can be described in polar form by the set of equations

$$\begin{bmatrix} \dot{x}_i(t) \\ \dot{y}_i(t) \\ \dot{\theta}_i(t) \\ \dot{\ell}_{ik}(t) \\ \dot{\alpha}_{ik}(t) \\ \dot{\theta}_{ik}(t) \end{bmatrix} = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \\ \cos\alpha_{ik} & 0 \\ -\frac{\sin\alpha_{ik}}{\ell_{ik}} & -1 \\ 0 & 1 \end{bmatrix} u_i(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\cos(\alpha_{ik}+\theta_{ik}) & -d\sin(\alpha_{ik}+\theta_{ik}) \\ \frac{\sin(\alpha_{ik}+\theta_{ik})}{\ell_{ij}} & -\frac{d\cos(\alpha_{ik}+\theta_{ik})}{\ell_{ik}} \\ 0 & -1 \end{bmatrix} u_k(t). \tag{2.5}$$

As mentioned, each agent is equipped with a vision sensor and a pan controller, as shown in Figure 2.2. Then, the measured variables are given by

$$\begin{aligned} \theta_t(t) &= \arctan\left(\frac{-x_c(t)}{z_c(t)}\right), \\ \alpha_{ik}(t) &= \theta_t(t) - \theta_m(t), \\ \theta_{ik}(t) &= \theta_p(t) + \theta_m(t). \end{aligned}$$

2.2 Graph theory

Graph theory provides a convenient framework for modeling multi-vehicle coordination problems. Particularly, it is very powerful to generalize individual properties to group properties [133]. For example, graphs have been used to capture the interconnection topology [37, 97], describe control structure [28, 27], test constraint feasibility [125], characterize information flow [40], and analyze error propagation [133]. Graphs can be undirected to model position constraints [37, 97], or directed to describe information flow [40] or leader following inter-agent control specifications [34, 127, 132]

In this Section, we resume some concepts from graph theory, the reader is referred to the literature for a detailed treatment [35, 96, 133].

A *directed graph* $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{E})$ of the set of vertices $\mathcal{V} \in \{1, \ldots, N\}$ and directed edges $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$. An edge $(i, j) \in \mathcal{E}$ is an ordered pair of distinct vertices in $\mathcal{V}$ that it is said to be incoming with respect to the *head* j and outgoing with respect to the *tail* i. The *in degree* of a vertex is defined as the number of edges that have this vertex as head. If i, $j \in \mathcal{V}$ and $(i, j) \in \mathcal{E}$, then i and j are said to be *adjacent*, or *neighbors*, and are denoted by $i \sim j$. The *adjacency matrix* $A(\mathcal{G}) = \{a_{ij}\}$ of a graph $\mathcal{G}$ is a matrix with nonzero elements such that $a_{ij} \neq 0 \Leftrightarrow i \sim j$. The set of *neighbors* of node i is defined by

$$\mathcal{N}_i := \{ j \in \mathcal{V} \backslash \{i\} | \, a_{ij} \neq 0 \} . \tag{2.6}$$

The number of neighbors of each vertex is its *valence*. A *path* of length r_p from vertex $i \in \mathcal{V}$ to vertex $j \in \mathcal{V}$ is a sequence of r_p+1 distinct vertices, $v_1 = i, \ldots, v_k, \ldots, v_{r_p+1} = j$, such that for all $k \in [1, r_p]$, $(v_k, v_{k+1}) \in \mathcal{E}$. A *weak path* of length r_p from vertex $i \in \mathcal{V}$ to vertex $j \in \mathcal{V}$ is a sequence of r_p+1 distinct vertices, $v_1 = i, \ldots, v_k, \ldots, v_{r_p+1} = j$, such that for all $k \in [1, r_p]$, $(v_k, v_{k+1}) \in \mathcal{E}$ or $(v_{k+1}, v_k) \in \mathcal{E}$. A directed graph $\mathcal{G}$ is *weakly connected* or simply *connected* if any two vertices can be joined with a weak path. The distance between two vertices i and j in a graph $\mathcal{G}$ is the length of the shortest path between both vertices. The *diameter* of a graph is the maximum distance between two distinct vertices.

The *incidence matrix* $B(\mathcal{G}^\sigma)$ of a graph $\mathcal{G}$ with orientation σ, $\mathcal{G}^\sigma$, is a matrix whose rows and columns are indexed by the vertices and edges of $\mathcal{G}$, respectively, such that the (i, j) entry of $B(\mathcal{G}^\sigma)$ is equal to 1 if edge (i, j) is incoming to vertex i, -1 if edge (i, j) is out coming from vertex i, and 0 otherwise.

The symmetric matrix defined as

$$L(\mathcal{G}) = B(\mathcal{G}^\sigma) B(\mathcal{G}^\sigma)^T$$

is called the *Laplacian* of $\mathcal{G}$ and is independent of the choice of orientation σ. For a connected graph, L has a single zero eigenvalue and the associated eigenvector is the n-dimensional vector of ones, 1_n.

A *cycle* is a connected graph where each vertex is incident with one incoming and one outgoing edge. An *acyclic graph* is a graph with no cycles.

Let $p_i = (p_i^x, p_i^y) \in \mathbb{R}^2$ denote the position of robot i, and $r_i > 0$ denote the *interaction range* between agent i and the other robots. A *spherical neighborhood* (or *shell*) of radius r_i around p_i is defined as

$$\mathcal{B}(p_i, r_i) := \left\{ q \in \mathbb{R}^2 : \|q - p_i\| \leq r_i \right\}.$$

Let us define $p = \mathrm{col}(p_i) \in \mathbb{R}^{2n}$, where $n = |\mathcal{V}|$ is the number of nodes of graph $\mathcal{G}$, and $r = \mathrm{col}(r_i)$. We refer to the pair (p, r) as a *cluster* with *configuration* p and vector of radii r. A *spatial adjacency matrix* $A(p) = [a_{ij}(p)]$ induced by a cluster is given by

$$a_{ij}(p) = \begin{cases} 1, & \text{if } p_j \in \mathcal{B}(p_i, r_i),\ j \neq i \\ 0, & \text{otherwise.} \end{cases}$$

The spatial adjacency matrix $A(p)$ defines a *spatially induced graph* or *net* $\mathcal{G}(p)$. A node $i \in \mathcal{V}$ with a spherical neighborhood define a *neighboring graph* $\mathcal{N}_i$ as

$$\mathcal{N}_i(p) := \{ j \in \mathcal{V} : a_{ij}(p) > 0 \}. \tag{2.7}$$

An *α-lattice* [96] is a configuration p satisfying the set of constraints

$$\|p_j - p_i\| = d,\ \forall j \in \mathcal{N}_i(p).$$

A *quasi α-lattice* [96] is a configuration p satisfying the set of inequality constraints

$$-\delta \leq \|p_j - p_i\| - d \leq \delta,\ \forall (i,j) \in \mathcal{E}(p).$$

2.3 Formal definition of formation

Now, it is possible to give a formal definition of formation [133]:

Definition 2.3.1 (Formation). *A* formation *is a network of vehicles interconnected via their controller* specifications *that dictate the relationship each agent must maintain with respect to its leader or leaders. The interconnections between agents are modeled as edges in an acyclic directed graph, labeled by a given relationship.*

Using the robot model presented in (2.5), the *relationship* between two robots could be distance and bearing angle or just distance. In the former case, the in-degree of the follower is one. Whereas in the latter case, the in-degree of the follower becomes two.

Definition 2.3.2 (Formation Control Graph). *A formation control graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{S})$ *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 to each vertex* v_k *a control system (4.52).*
- *An edge set* $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ *encoding leader-follower relationships between agents. The ordered pair* $(v_i, v_k) := e_{ik}$ *belongs to* $\mathcal{E}$ *if* u_k *depends on the state of agent* i, q_i.
- *A collection* $\mathcal{S} = \{s_k\}$ *of node specifications defining control objectives, or* set points*, for each node* k :(v_i, v_k), $(v_j, v_k) \in \mathcal{E}$ *and for some* v_i, v_j, $v_k \in \mathcal{V}$.

Remark 2.3.3. Note that $j \equiv i$ when the relationship between robots i and k is given by distance and bearing.

Remark 2.3.4. Some examples of control objectives are collision avoidance, or maintenance of communication or sensing links. In the first case, this objective can be defined by the minimum distance two robots should keep. The second and third cases can be specified by a maximum distance between robots and a given angular range to be achieved.

The tails of all incoming edges to a vertex k represent leaders of robot k. Vertices of in-degree zero represent *formation leaders*. Leader agents have no incoming edges; because they regulate their behavior such that the formation may achieve some group objectives, such as navigation avoiding obstacles or tracking reference paths.

Therefore, the *formation control problem* can be divided into two parts: *graph assignment* and *controller design*. The first problem consists in designing the formation control graph $\mathcal{G}$ and the specification collection $\mathcal{S}$. The second problem consists in maintaining the formation described by the pair $\mathcal{G}$ and $\mathcal{S}$ in the presence of constrains, obstacles, or limited sensing capabilities.

In this dissertation, we assume that $\mathcal{G}$ and $\mathcal{S}$ are preassigned and mainly focus on the second problem.

Chapter 3

Nonlinear estimation

Many control algorithms rely on the basic design assumption of exact knowledge of the system model and availability of measurements of every state. For example, the widely used *state feedback* control techniques [27, 34, 44, 58, 69]. However, in real applications, measurements acquired with one or several sensors are noisy and uncertain and some states are not available. For instance, mobile robots carry sensors such as wheel encoders, inertial units, accelerometers, gyroscopes, and cameras. The data of all these sensors should be smoothened and/or fused to estimate the real values of the system variables using estimation filters [92, 109, 113, 116]. On the other hand, observers can be used to estimate unavailable variables [58, 70, 93, 122]. In particular, output differentiators play a central role in the estimation of the derivatives of some states.

Two of the most well-known estimation tools available nowadays are the extended Kalman filter (EKF) [23, 86] and the unscented Kalman filter (UKF) [65, 66, 140, 141]. The EKF relies on a linearization using a Taylor series expansion of a nonlinear sensor model. Because the linearization is performed using uncertain states, it can produce the divergence of the filter.

The UKF performs the linearization process in the Kalman filter using the unscented transform. The unscented transform is a deterministic sampling approach that propagates through the true non-linear system a set of chosen sample points to completely describes the true mean and covariance of a Gaussian random variable. The propagated points can capture the posterior mean and covariance accurately up to the 3rd order of a Taylor series expansion. The UKF outperforms the EKF in nonlinear system applications.

Both filters, rely on an accurate model description of the system. In the last few years, two state observers have emerged: the high-gain observer (HGO) [1, 2, 55,

69, 70, 84], and the high-order sliding mode (HOSM) observer [5, 6, 7, 49, 76, 78, 77]. The advantage of both observers is that they do not need an accurate system model, because they are particularly appropriate to deal with unmodeled dynamics and perturbations.

The rest of this Chapter is organized as follows. In Section 3.1, we resume basic concepts on recursive estimation. In Section 3.2, we review the extended Kalman algorithm and analyze a simple nonlinear example. In Section 3.3, we describe the unscented Kalman filter and compare it with the EKF. In Section 3.5, we study the performance of high-order observers. In Section 3.6, we survey the high-order sliding model observer and its use as differentiator.

3.1 Optimal recursive estimation

Let the state vector be given by $\mathbf{x} \in \mathbb{R}^n$, and let the system be described by the non-linear stochastic difference equation

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k, \mathbf{v}_k\right), \\ \mathbf{y}_k &= \mathbf{h}\left(\mathbf{x}_k, \mathbf{n}_k\right), \end{aligned} \tag{3.1}$$

where $\mathbf{x}_k$ is the unobserved state of the system, $\mathbf{u}_k \in \mathbb{R}^m$ is a known exogenous input, $\mathbf{y}_k \in \mathbb{R}^p$ is the observed measurement signal, $\mathbf{v}_k$ and $\mathbf{n}_k$ are process and measurement noise, respectively. Variables $\mathbf{v}_k$ and $\mathbf{n}_k$ are assumed independent, white, and with Gaussian probability distributions $p\left(\mathbf{v}\right) \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_\mathbf{v}\right)$ and $p\left(\mathbf{n}\right) \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_\mathbf{n}\right)$.

The goal is to estimate the state $\mathbf{x}_k$ using observations $\mathbf{y}_k$. The optimal estimate in the minimum mean-squared error (MMSE) sense is given by the conditional mean [140, 141]

$$\hat{\mathbf{x}}_k = \mathbb{E}\left[\mathbf{x}_k \left| \mathbf{Y}_0^k \right.\right], \tag{3.2}$$

where $\mathbf{Y}_0^k$ is the sequence of observations up to time k. In order to solve equation (3.2) it is necessary the knowledge of the *a posteriori* density $p\left(\mathbf{x}_k \left| \mathbf{Y}_0^k \right.\right)$. The a posteriori density can be computed using the Bayesian approach

$$p\left(\mathbf{x}_k \left| \mathbf{Y}_0^k \right.\right) = \frac{p\left(\mathbf{x}_k \left| \mathbf{Y}_0^{k-1} \right.\right) p\left(\mathbf{y}_k \left| \mathbf{x}_k \right.\right)}{p\left(\mathbf{y}_k \left| \mathbf{Y}_0^{k-1} \right.\right)}, \tag{3.3}$$

where

$$p\left(\mathbf{x}_k \left| \mathbf{Y}_0^{k-1} \right.\right) = \int p\left(\mathbf{x}_k \left| \mathbf{x}_{k-1} \right.\right) p\left(\mathbf{x}_{k-1} \left| \mathbf{Y}_0^{k-1} \right.\right) d\mathbf{x}_{k-1}, \tag{3.4}$$

and the normalizing constant $p\left(\mathbf{y}_k \left| \mathbf{Y}_0^{k-1} \right.\right)$ is given by

$$p\left(\mathbf{y}_k \left| \mathbf{Y}_0^{k-1} \right.\right) = \int p\left(\mathbf{x}_k \left| \mathbf{Y}_0^{k-1} \right.\right) p\left(\mathbf{y}_k \left| \mathbf{x}_k \right.\right) d\mathbf{x}_k. \tag{3.5}$$

Equation (3.3) specifies the current state density as a function of the previous density and the most recent measurement data. Knowledge of the initial condition $p\left(\mathbf{x}_0 \left| \mathbf{y}_0 \right.\right) = \frac{p(\mathbf{y}_0|\mathbf{x}_0)p(\mathbf{x}_0|\mathbf{y}_0)}{p(\mathbf{y}_0)}$ determines $p\left(\mathbf{x}_k \left| \mathbf{Y}_0^k \right.\right)$ for all k. State-transition probability $p\left(\mathbf{x}_k \left| \mathbf{x}_{k-1} \right.\right)$ and measurement probability $p\left(\mathbf{y}_k \left| \mathbf{x}_k \right.\right)$ are specified by the state-space model (3.1) and the densities $p\left(\mathbf{v}_k\right)$, $p\left(\mathbf{n}_k\right)$. However, the multi-dimensional integrations given by (3.4) and (3.5) cannot not be computed in closed-form for most systems. As performed in the *particle filter* approach, the integral can be approximated by finite sums by applying Monte-Carlo sampling techniques. These finite sums converge to the true solution in the limit.

Assuming that all densities remain Gaussian, then the Bayesian recursion can be greatly simplified because only the conditional mean $\hat{\mathbf{x}}_k = \mathbb{E}\left[\mathbf{x}_k \left| \mathbf{Y}_0^k \right.\right]$ and covariance $\mathbf{P}_{\mathbf{x}_k}$ need to be evaluated. This lead to the recursive estimation

$$\hat{\mathbf{x}}_k^- = \mathbb{E}\left[\mathbf{f}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}\right)\right], \tag{3.6}$$

$$\mathcal{K}_k = \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}^{-1}, \tag{3.7}$$

$$\hat{\mathbf{y}}_k^- = \mathbb{E}\left[\mathbf{h}\left(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}\right)\right], \tag{3.8}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathcal{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k^-\right), \tag{3.9}$$

$$\mathbf{P}_{\mathbf{x}_k} = \mathbf{P}_{\mathbf{x}_k}^- - \mathcal{K}_k\mathbf{P}_{\tilde{\mathbf{y}}_k}\mathcal{K}_k^T, \tag{3.10}$$

where $\hat{\mathbf{x}}_k^-$ and $\hat{\mathbf{y}}_k^-$ are optimal predictions $\mathbf{x}_k$ and $\mathbf{y}_k$, respectively, $\mathcal{K}_k$ is the optimal gain term, $\mathbf{P}_{\mathbf{x}_k}^-$ is the prediction of the covariance of $\mathbf{x}_k$ and $\mathbf{P}_{\tilde{\mathbf{y}}_k}$ is the covariance of $\tilde{\mathbf{y}}_k$, with $\tilde{\mathbf{y}}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k^-$.

3.2 The extended Kalman filter (EKF)

The extended Kalman filter (EKF) is an estimation algorithm that uses a predictor-corrector mechanism to estimate the current state of a system with its nonlinear model and to correct this estimate using any available sensor measurements. The

EKF approximates the optimal nonlinear terms in equations (3.6) to (3.10) as

$$\begin{aligned}\hat{\mathbf{x}}_k^- &\approx \mathbf{f}\left(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \bar{\mathbf{v}}\right),\\ \mathcal{K}_k &\approx \hat{\mathbf{P}}_{\mathbf{x}_k\mathbf{y}_k}\hat{\mathbf{P}}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}^{-1},\\ \hat{\mathbf{y}}_k^- &\approx \mathbf{h}\left(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{v}_{k-1}\right),\end{aligned}$$

where noise random variables with distributions $p(\mathbf{v}_k)$, $p(\mathbf{n}_k)$ are approximated by their prior mean $\bar{\mathbf{v}} = \mathbb{E}[\mathbf{v}]$, $\bar{\mathbf{n}} = \mathbb{E}[\mathbf{n}]$, respectively. The covariances are computed by linearizing model (3.1) as follows

$$\begin{aligned}\mathbf{x}_{k+1} &\approx \mathbf{A}_k\mathbf{x}_k + \mathbf{G}_k\mathbf{u}_k + \mathbf{B}_k\mathbf{v}_k,\\ \mathbf{y}_k &\approx \mathbf{C}_k\mathbf{x}_k + \mathbf{D}_k\mathbf{n}_k,\end{aligned}$$

where

$$\begin{aligned}&\mathbf{A}_k = \left.\frac{\partial \mathbf{f}\left(\mathbf{x}, \mathbf{u}_k, \bar{\mathbf{v}}\right)}{\partial \mathbf{x}}\right|_{\mathbf{x}_k}, \ \mathbf{G}_k = \left.\frac{\partial \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}, \bar{\mathbf{v}}\right)}{\partial \mathbf{u}}\right|_{\mathbf{u}_k}, \ \mathbf{B}_k = \left.\frac{\partial \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k, \mathbf{v}\right)}{\partial \mathbf{v}}\right|_{\bar{\mathbf{v}}},\\ &\mathbf{C}_k = \left.\frac{\partial \mathbf{h}\left(\mathbf{x}, \bar{\mathbf{n}}\right)}{\partial \mathbf{x}}\right|_{\mathbf{x}_k}, \text{ and } \mathbf{D}_k = \left.\frac{\partial \mathbf{h}\left(\mathbf{x}_k, \mathbf{n}\right)}{\partial \mathbf{n}}\right|_{\bar{\mathbf{n}}}.\end{aligned}$$

Algorithm 1 resumes the equations for the EKF.

3.2.1 Simulation example

The following model is used through all the examples in the rest of this Chapter. Let the system be given by

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= x_2^3 + u,\\ y &= x_1.\end{aligned} \tag{3.11}$$

It can be easily proved that the feedback law

$$u = -x_2^3 - x_1 - x_2, \tag{3.12}$$

stabilizes system (3.11).

Figures 3.1 and 3.2 show results of state estimation using the EKF for model (3.11), with additive Gaussian noise on the output $p(\mathbf{n}) \sim \mathcal{N}(\mathbf{0}, 3 \times 10^{-6})$. The

Algorithm 1 Extended Kalman filter

1: Initialization.

$$\hat{\mathbf{x}}_0 = \mathbb{E}\left[\mathbf{x}_0\right], \mathbf{P}_{\mathbf{x}_0} = \mathbb{E}\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)^T\right].$$

2: **for** $k \in \mathbb{N}$ **do**

3: Time update:

$$\begin{aligned}\hat{\mathbf{x}}_k^- &= \mathbf{f}\left(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \bar{\mathbf{v}}\right), \\ \mathbf{P}_{\mathbf{x}_k}^- &= \mathbf{A}_{k-1}\mathbf{P}_{\mathbf{x}_{k-1}}\mathbf{A}_{k-1}^T + \mathbf{B}_k\mathbf{R}_\mathbf{v}\mathbf{B}_k^T.\end{aligned}$$

4: Measurement update:

$$\begin{aligned}\mathcal{K}_k &= \mathbf{P}_{\mathbf{x}_k}^-\mathbf{C}_k^T\left(\mathbf{C}_k\mathbf{P}_{\mathbf{x}_k}^-\mathbf{C}_k^T + \mathbf{D}_k\mathbf{R}_\mathbf{n}\mathbf{D}_k^T\right)^{-1}, \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \mathcal{K}_k\left(\mathbf{y}_k - \mathbf{h}\left(\hat{\mathbf{x}}_k^-, \bar{\mathbf{n}}\right)\right), \\ \mathbf{P}_{\mathbf{x}_k} &= \left(\mathbf{I} - \mathcal{K}_k\mathbf{C}_k\right)\mathbf{P}_{\mathbf{x}_k}^-,\end{aligned}$$

5: **end for**

where $\mathbf{R}_\mathbf{v}$ is the process noise covariance, $\mathbf{R}_\mathbf{n}$ is the measurement noise covariance, and

$$\begin{aligned}\mathbf{A}_k &= \left.\frac{\partial \mathbf{f}\left(\mathbf{x}, \mathbf{u}_k, \bar{\mathbf{v}}\right)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k}, \mathbf{B}_k = \left.\frac{\partial \mathbf{f}\left(\hat{\mathbf{x}}_k^-, \mathbf{u}_k, \mathbf{v}\right)}{\partial \mathbf{v}}\right|_{\bar{\mathbf{v}}}, \\ \mathbf{C}_k &= \left.\frac{\partial \mathbf{h}\left(\mathbf{x}, \bar{\mathbf{n}}\right)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k}, \mathbf{D}_k = \left.\frac{\partial \mathbf{h}\left(\hat{\mathbf{x}}_k^-, \mathbf{n}\right)}{\partial \mathbf{n}}\right|_{\bar{\mathbf{n}}}.\end{aligned}$$

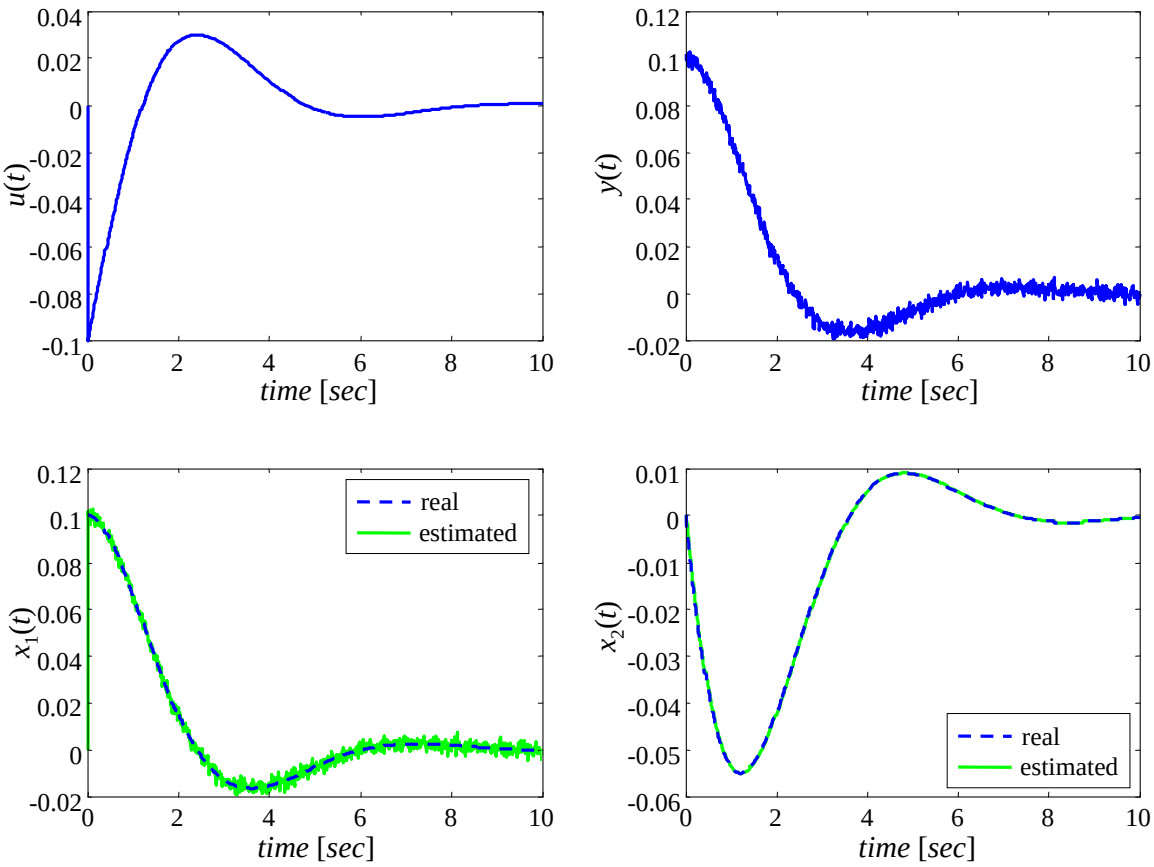


Figure 3.1: State estimation with the EKF.

initial state is given by $\mathbf{x}_0 = [x_1(0), x_2(0)]^T = [0.1, 0]^T$, the filter is initialized with expected initial state $\hat{\mathbf{x}}_0 = \mathbb{E}[\mathbf{x}_0] = [0, 0]^T$, and state covariance $\mathbf{P}_{\mathbf{x}_0} = \text{diag}(1, 1)$.

Figure 3.1 shows state estimation results for model (3.11) using control law (3.12). Figure 3.2 shows similar results with the control law

$$u = -\hat{x}_2^3 - \hat{x}_1 - \hat{x}_2. \tag{3.13}$$

Both figures show similar behavior due to the fast convergence of the EKF.

3.3 The unscented Kalman filter (UKF)

The unscented Kalman filter (UKF) uses the unscented transform (UT) to capture the mean and covariance estimates with a minimal set of sample points. It provides 3rd order accuracy of the Taylor series expansion of the Gaussian error distribution for any non-linear system. For non-Gaussian inputs, approximations are accurate to at least the 2nd order [65, 66, 140, 141].

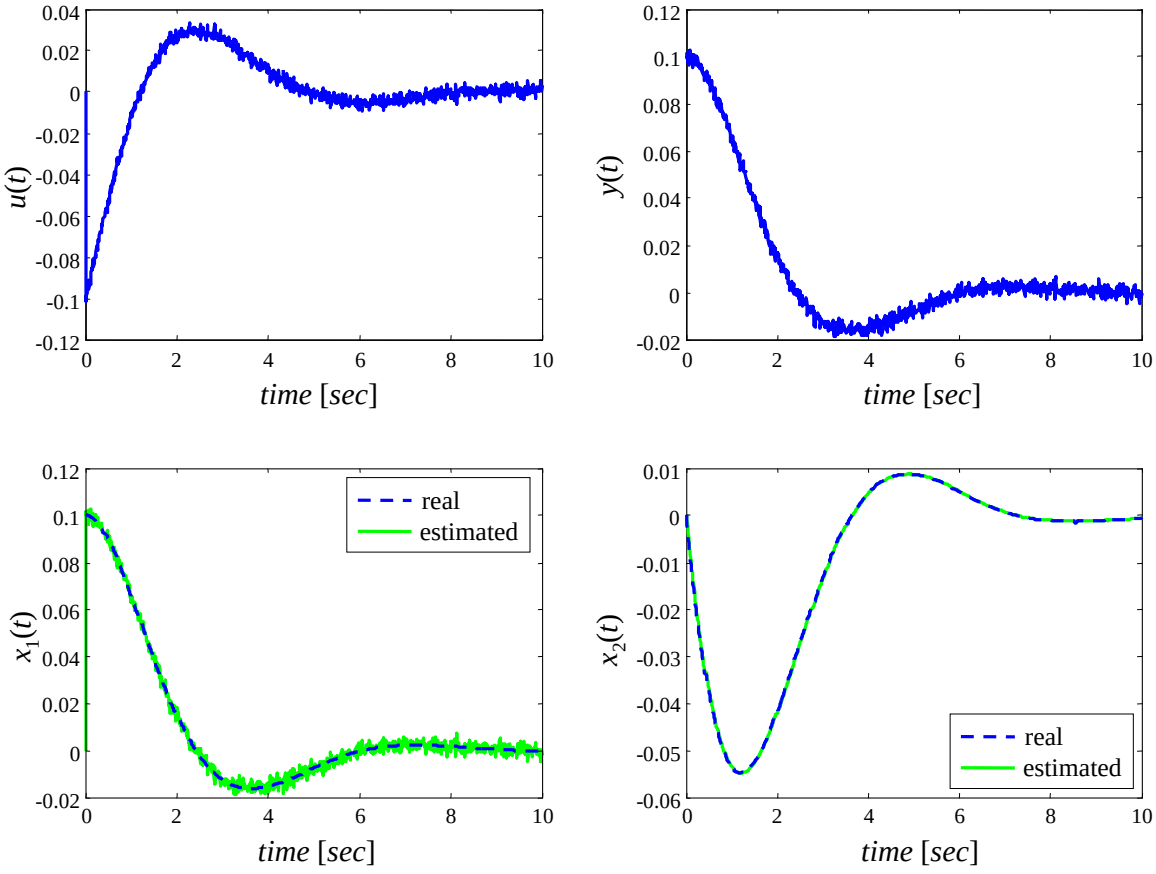


Figure 3.2: State estimation and control with the EKF.

The UT is a deterministic sampling approach that can be summarized as follows [141]: Let $\mathbf{x} \in \mathbb{R}^n$ be a n-dimensional random variable with mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P_x}$. The propagation of the random variable $\mathbf{x}$ through a nonlinear function $\mathbf{y} = \mathbf{g}(\mathbf{x})$ can be approximated by $2n+1$ weighted points $\chi_i \in \mathbb{R}^n$, $i = 0, \ldots, 2n$,

$$\begin{aligned}
\chi_0 &= \bar{\mathbf{x}}, \\
\chi_i &= \bar{\mathbf{x}} + \left(\sqrt{(n+\kappa)\,\mathbf{P_x}}\right)_i, \quad i = 1, \ldots, n, \\
\chi_i &= \bar{\mathbf{x}} - \left(\sqrt{(n+\kappa)\,\mathbf{P_x}}\right)_i, \quad i = n+1, \ldots, 2n, \\
W_0^m &= \frac{\kappa}{n+\kappa}, \\
W_0^c &= \frac{\kappa}{n+\kappa} + a - \alpha^2 + \beta, \\
W_i^c &= W_i^m = \frac{1}{2\,(n+\kappa)}, \quad i = 1, \ldots, 2n,
\end{aligned}$$

where $\kappa = \alpha^2\,(n+\lambda) - n$ is a scaling parameter, α determines the spread of the sigma points around $\bar{\mathbf{x}}$ and is usually set to a small positive value, λ is a scaling parameter of value 0 or $3-n$, β is a parameter used to incorporate any prior knowledge about the distribution of $\mathbf{x}$ ($\beta = 2$ is optimal for Gaussian distributions), $\left(\sqrt{(n+\kappa)\,\mathbf{P_x}}\right)_i$ is the ith column of the matrix square root, and W_i is a weight associated with the ith point.

The sigma points χ_i are then propagated through the nonlinear function representing the system

$$\mathcal{Y}_i = \mathbf{g}(\chi_i), \quad i = 0, \ldots, 2n,$$

and the mean and covariance for $\mathbf{y}$ are approximated using a weighted sample mean and covariance of the posterior sigma points,

$$\begin{aligned}
\bar{\mathbf{y}} &\approx \sum_{i=0}^{2n} W_i^m \mathcal{Y}_i, \\
\mathbf{P_y} &\approx \sum_{i=0}^{2n} (\mathcal{Y}_i - \bar{\mathbf{y}})\,(\mathcal{Y}_i - \bar{\mathbf{y}})^T.
\end{aligned}$$

The unscented Kalman filter is a straightforward extension of the UT to recursive estimation of system (3.1). Algorithm 2 resumes UKF equations for parameter estimation for process and measurement noise purely additive [141].

Algorithm 2 Unscented Kalman filter for state estimation.

1: Initialization.

$$\hat{\mathbf{x}}_0 = \mathbb{E}\left[\mathbf{x}_0\right], \mathbf{P}_{\mathbf{x}_0} = \mathbb{E}\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0\right)^T\right].$$

2: **for** $k \in \mathbb{N}$ **do**

3: Compute sigma points:

$$\chi_{k-1} = \left[\hat{\mathbf{x}}_{k-1},\, \hat{\mathbf{x}}_{k-1} \pm \gamma\sqrt{\mathbf{P}_{k-1}}\right].$$

4: Time update:

$$\begin{aligned}
\chi^*_{k|k-1} &= \mathbf{f}\left(\chi_{k-1}, \mathbf{u}_{k-1}\right), \\
\hat{\mathbf{x}}^-_k &= \sum_{i=0}^{2N_x} W_i^m \chi^*_{i,k|k-1}, \\
\mathbf{P}^-_k &= \sum_{i=0}^{2N_x} W_i^c \left(\chi^*_{i,k|k-1} - \hat{\mathbf{x}}^-_k\right)\left(\chi^*_{i,k|k-1} - \hat{\mathbf{x}}^-_k\right)^T + \mathbf{R}_\mathbf{v}, \\
\chi_{k|k-1} &= \left[\hat{\mathbf{x}}^-_{k-1},\, \hat{\mathbf{x}}^-_{k-1} \pm \gamma\sqrt{\mathbf{P}^-_k}\right], \\
\mathcal{Y}_{k|k-1} &= \mathbf{h}\left(\chi_{k|k-1}\right), \\
\hat{\mathbf{y}}^-_k &= \sum_{i=0}^{2N_x} W_i^m \mathcal{Y}_{i,k|k-1}.
\end{aligned}$$

5: Measurement update:

$$\begin{aligned}
\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k} &= \sum_{i=0}^{2n} W_i^c \left(\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}^-_k\right)\left(\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}^-_k\right)^T + \mathbf{R}_\mathbf{n}, \\
\mathbf{P}_{\mathbf{x}_k\mathbf{y}_k} &= \sum_{i=0}^{2n} W_i^c \left(\mathcal{X}_{i,k|k-1} - \hat{\mathbf{x}}^-_k\right)\left(\mathcal{Y}_{i,k|k-1} - \hat{\mathbf{y}}^-_k\right)^T, \\
\mathcal{K}_k &= \mathbf{P}_{\mathbf{x}_k\mathbf{y}_k}\mathbf{P}^{-1}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}, \\
\hat{\mathbf{x}}_k &= \hat{\mathbf{x}}^-_k + \mathcal{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}^-_k\right), \\
\mathbf{P}_{\mathbf{x}_k} &= \mathbf{P}_{\mathbf{x}_{k-1}} - \mathcal{K}_k\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}\mathcal{K}_k^T,
\end{aligned}$$

6: **end for**

where n is the dimension of the original state $\mathbf{x}_k$, $\gamma = \sqrt{n + \lambda}$, $\mathbf{R}_\mathbf{v}$ is the process noise covariance, and $\mathbf{R}_\mathbf{n}$ is the measurement noise covariance.

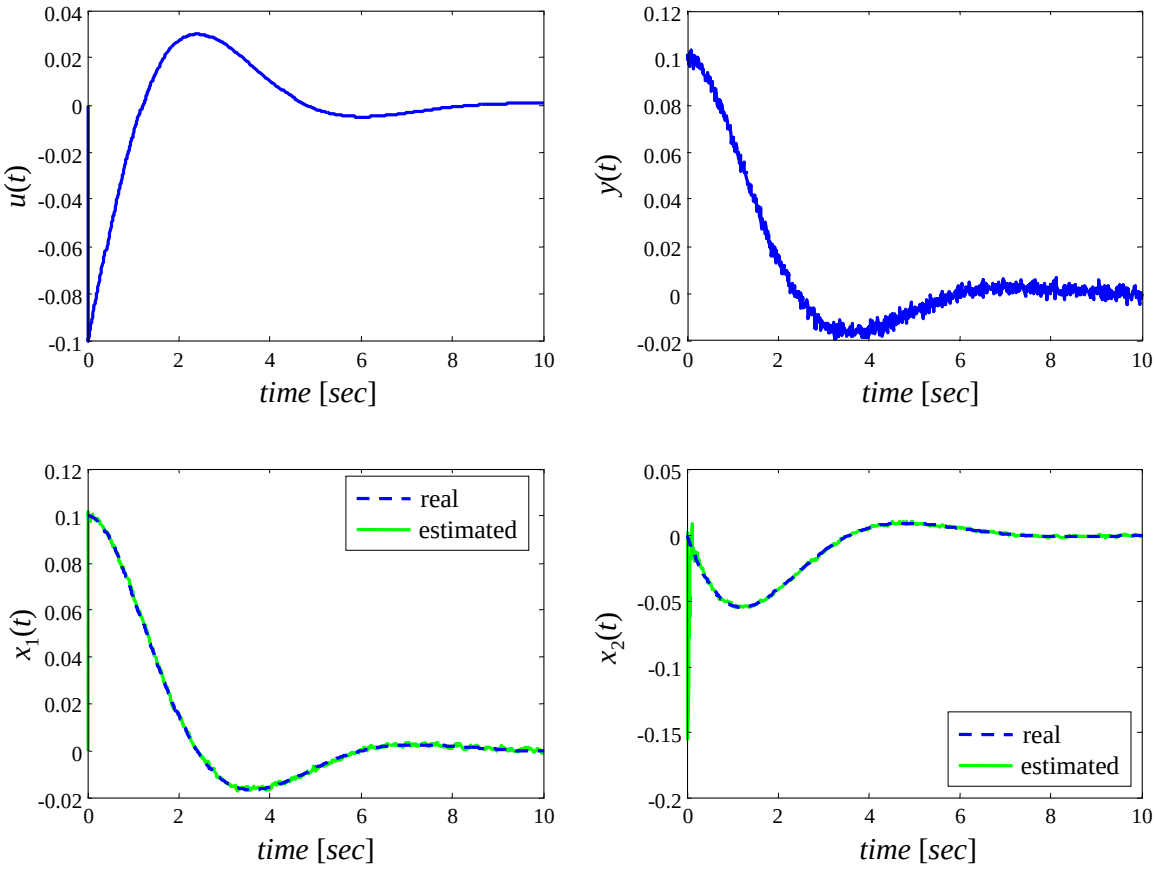


Figure 3.3: State estimation with the UKF.

3.3.1 Simulation example

Figures 3.3 and 3.4 show results of state estimation using the UKF for the model given by equation (3.11), as was presented in Section 3.2.1.

Figure 3.3 shows state estimation results for model (3.11) using control law (3.12). As can be seen in figures 3.1 and 3.3, the estimation results are similar except for smoother estimates using the UKF.

The smoothing effect is more clear in the computation of control law (3.13), as can be seen comparing figures 3.2 and 3.4. The smoothing effect is due to the better approximation that the UKF does of the nonlinear system function.

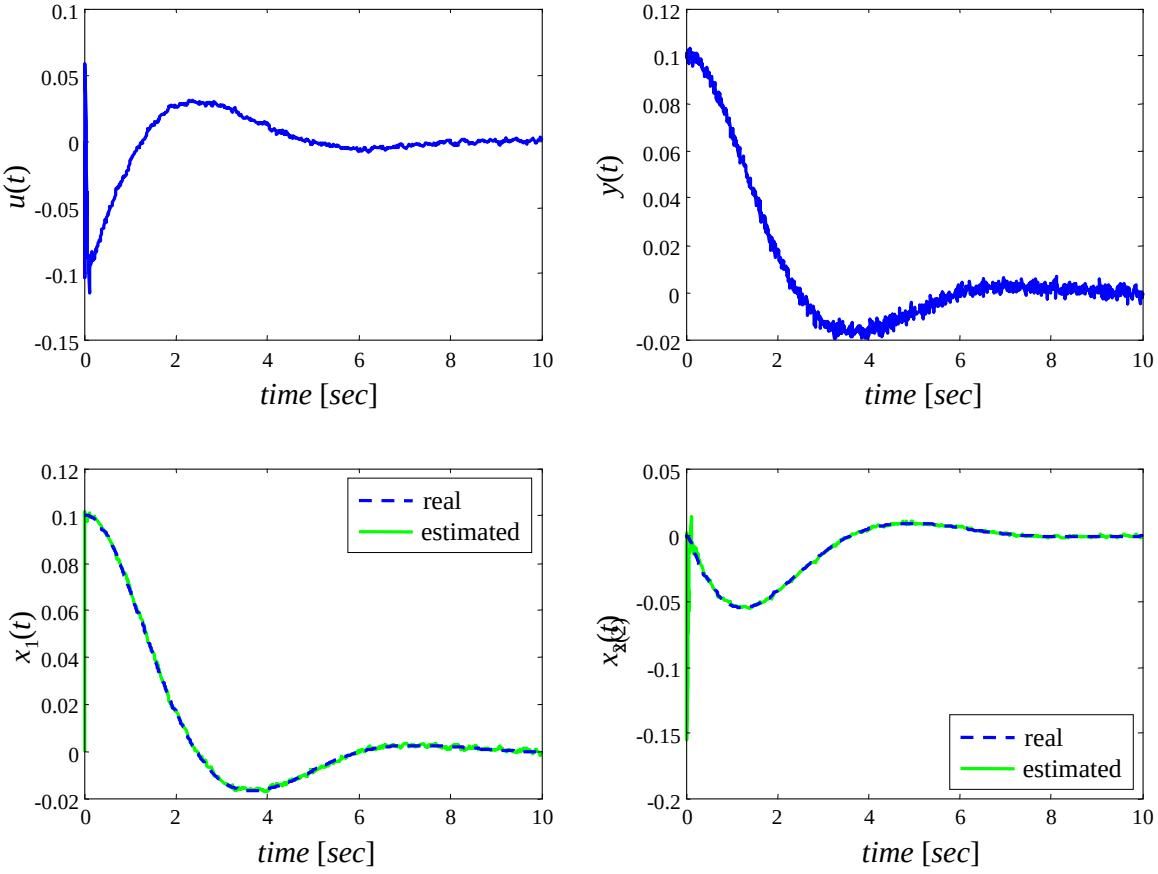


Figure 3.4: State estimation and control with the UKF.

3.4 UKF for parameter estimation

An application of the EKF or UKF is learning a nonlinear mapping. This application is used in Chapter 4 to obtain leader's velocities. Let the mapping be given by

$$\mathbf{y}_k = \mathbf{g}\left(\mathbf{x}_k, \mathbf{w}\right),$$

where $\mathbf{w} \in \mathbb{R}^{n_w}$ corresponds to a set of unknown parameters. The UKF may be used to estimate the parameters $\mathbf{w}$ by writing a new state-space representation

$$\begin{aligned} \mathbf{w}_{k+1} &= \mathbf{w}_k + \mathbf{v}_k, \\ \mathbf{y}_k &= \mathbf{g}\left(\mathbf{x}_k, \mathbf{w}_k\right) + \mathbf{n}_k. \end{aligned} \tag{3.14}$$

This model (3.14) solves the optimization problem [141]

$$\min_{\mathbf{w}} J\left(\mathbf{w}\right) = \min_{\mathbf{w}} \mathbb{E}\left\{\sum_{m=1}^{k} \left[\mathbf{y}_m - \mathbf{g}\left(\mathbf{x}_m, \mathbf{w}\right)\right]^T \left[\mathbf{y}_m - \mathbf{g}\left(\mathbf{x}_m, \mathbf{w}\right)\right]\right\}. \tag{3.15}$$

Algorithm 3 resumes the equations of the UKF for parameter estimation.

In this work, we have chosen the innovation covariance $\mathbf{R}_{\mathbf{v},k}$ in Algorithm 3 as $\mathbf{R}_{\mathbf{v},k} = \mathbb{E}\left[\mathbf{v}_k \mathbf{v}_k^T\right] = \left(\lambda_v^{-1} - 1\right)\mathbf{R}_{\mathbf{w}}$, with $\lambda_v \in (0, 1]$, to provide an exponentially decaying weighting term as a forgetting factor to allow adaptation to changes in $\mathbf{w}$.

3.4.1 Simulation example

Let the system be given by

$$\begin{aligned} \dot{x}_1 &= w_1 x_1 + w_2 x_2, \\ \dot{x}_2 &= x_2^3 + u, \\ y &= 10 x_1, \end{aligned} \tag{3.16}$$

where $\mathbf{w} = [1.5, 1.0]^T$ represents the vector of unknown parameters. The control $u = -x_2^3 - 5x_1 - 3x_2$ stabilizes system (3.16), with the closed loop matrix given by

$$A_{cl} = \begin{bmatrix} 1.5 & 1 \\ -5 & -3 \end{bmatrix},$$

then, the eigenvalues of this system are given by $\lambda_1\left(A_{cl}\right) = -0.5$ and $\lambda_2\left(A_{cl}\right) = -1$.

Figures 3.5 and 3.6 show results of parameter estimation using the UKF for

Algorithm 3 Unscented Kalman filter for parameter estimation.

1: Initialization.

$$\hat{\mathbf{w}}_0 = \mathbb{E}\left[\mathbf{w}\right], \, \mathbf{P}_{\mathbf{w}_0} = \mathbb{E}\left[\left(\mathbf{w} - \hat{\mathbf{w}}_0\right)\left(\mathbf{w} - \hat{\mathbf{w}}_0\right)^T\right].$$

2: **for** $k \in \mathbb{N}$ **do**

3: Time update and sigma points computation:

$$\begin{aligned}
\hat{\mathbf{w}}_k^- &= \hat{\mathbf{w}}_{k-1}, \\
\mathbf{P}_{\mathbf{w}_k}^- &= \mathbf{P}_{\mathbf{w}_{k-1}} + \mathbf{R}_{\mathbf{v},k-1}, \\
\mathcal{W}_{k|k-1} &= \left[\hat{\mathbf{w}}_k^-, \, \hat{\mathbf{w}}_k^- \pm \gamma\sqrt{\mathbf{P}_{\mathbf{w}_k}^-}\right], \\
\mathcal{D}_{k|k-1} &= \mathbf{g}\left(\mathbf{x}_k, \mathcal{W}_{k|k-1}\right), \\
\hat{\mathbf{y}}_k &= \mathbf{g}\left(\mathbf{x}_k, \hat{\mathbf{w}}_k^-\right).
\end{aligned}$$

4: Measurement update:

$$\begin{aligned}
\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k} &= \sum_{i=0}^{2n_w} W_i^c \left(\mathcal{D}_{i,k|k-1} - \hat{\mathbf{y}}_k\right)\left(\mathcal{D}_{i,k|k-1} - \hat{\mathbf{y}}_k\right)^T + \mathbf{R}_{\mathbf{n},k}, \\
\mathbf{P}_{\mathbf{w}_k\mathbf{y}_k} &= \sum_{i=0}^{2n_w} W_i^c \left(\mathcal{W}_{i,k|k-1} - \hat{\mathbf{w}}_k^-\right)\left(\mathcal{D}_{i,k|k-1} - \hat{\mathbf{y}}_k\right)^T, \\
\mathcal{K}_k &= \mathbf{P}_{\mathbf{w}_k\mathbf{y}_k}\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}^{-1}, \\
\hat{\mathbf{w}}_k &= \hat{\mathbf{w}}_k^- + \mathcal{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k\right), \\
\mathbf{P}_{\mathbf{w}_k} &= \mathbf{P}_{\mathbf{w}_k}^- - \mathcal{K}_k\mathbf{P}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}\mathcal{K}_k^T,
\end{aligned}$$

5: **end for**

where n_w is the dimension of parameter space, $\gamma = \sqrt{n_w + \lambda}$, $\mathbf{R}_{\mathbf{v}}$ is the process noise covariance, and $\mathbf{R}_{\mathbf{n}}$ is the measurement noise covariance.

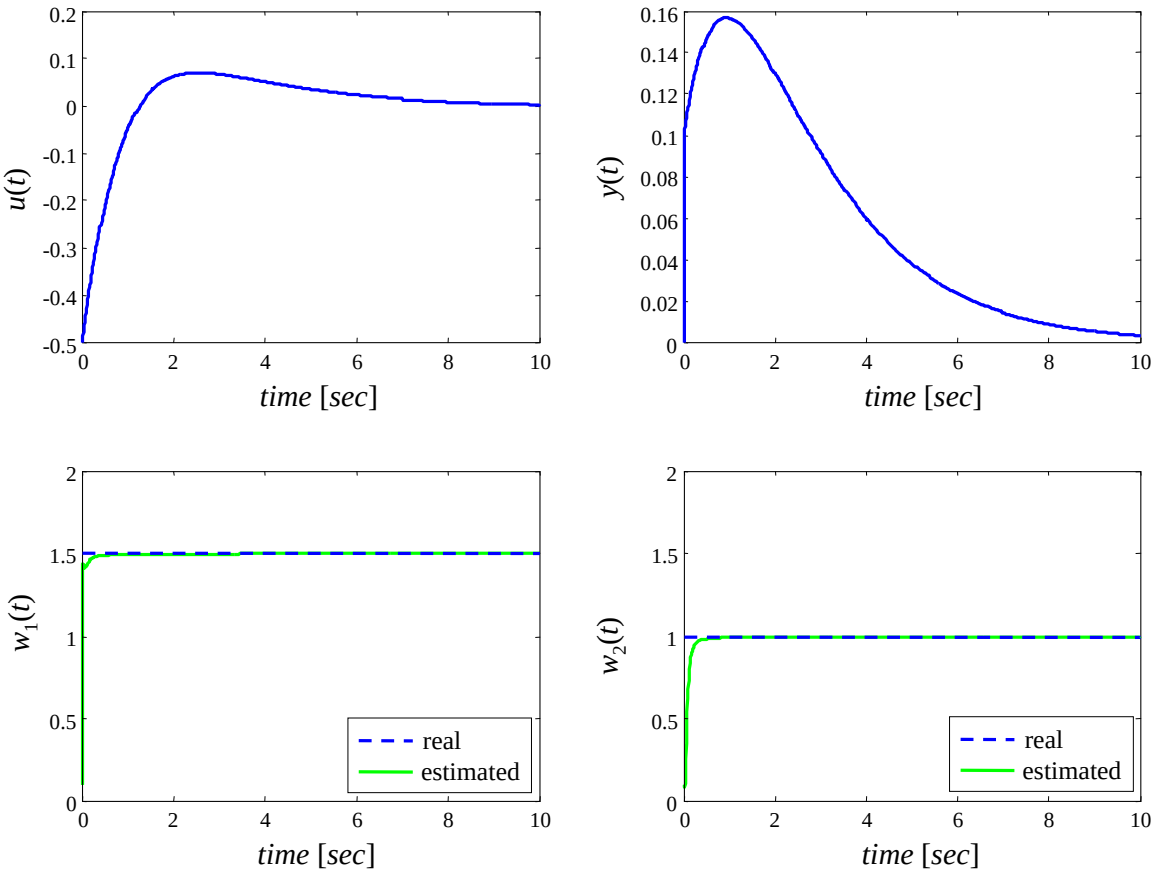


Figure 3.5: Parameter estimation with the UKF.

model (3.16). The filter is initialized with expected parameter values $\hat{\mathbf{w}}_0 = \mathbb{E}[\mathbf{w}_0] = [0.1, 0.1]^T$, and parameter covariance $\mathbf{P}_{\mathbf{w}_0} = \text{diag}(50, 50)$.

Figures 3.5 shows that the parameters converge to true values with a noiseless output signal. Figure 3.6 shows the results of estimating parameters when the output is corrupted by additive Gaussian noise $p(\mathbf{n}) \sim \mathcal{N}(\mathbf{0}, 3 \times 10^{-6})$. As can be seen, the parameters exhibit polarization, converging to the values $w_1(\infty) = 1.4083$ and $w_2(\infty) = 0.9318$. This characteristic is typical in system without enough input richness.

3.5 High-gain observers (HGO)

In the last years, the combination between *high-gain* observers and globally bounded state feedback control has gained the attention of many researchers [2, 55, 69, 70, 84]. This interest is due to robustness properties exhibited by the observer to uncertainties, unmodeled sensor and actuator dynamics, and to the ability of the observer-controller

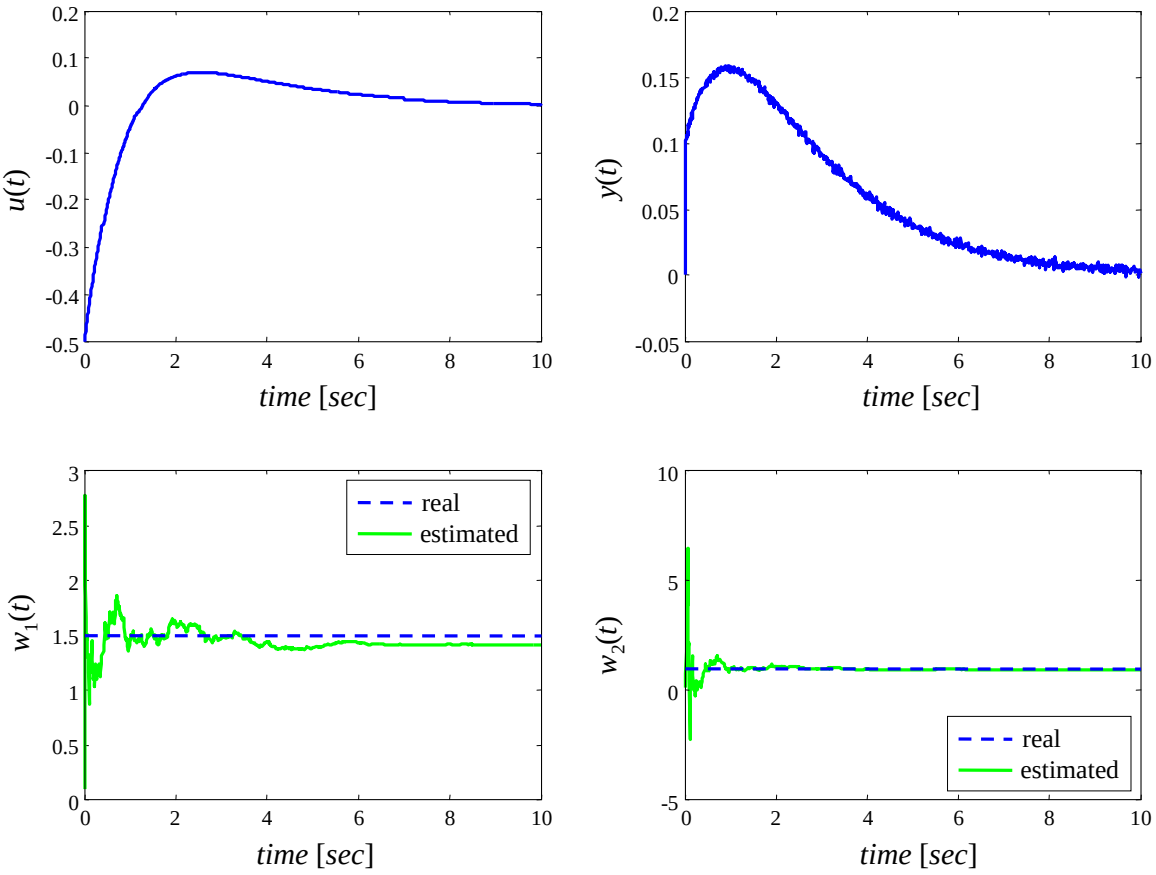


Figure 3.6: Parameter estimation with the UKF and noisy output.

pair to recover the state feedback controller performance when the gain of the observer is high enough.

An additional advantage of high-gain observers is the possibility of using the *separation principle* to design controllers for a class of nonlinear systems [2, 62]. The *separation principle* means that a controller and an observer can be designed in two steps:

- Design of a globally bounded state feedback controller based on the state variable x that stabilizes the system and meets given specifications.
- Development of an output feedback controller by replacing the state x by its estimate $\hat{x}$.

The combined observer-controller output feedback preserve the main features of the controller with the full state available, in spite of the non-correctness of high-gain observers with any fixed finite gain values.

One of the main drawbacks of high-gain observers is that they extend the bandwidth of the control design to unmodeled fast dynamics. The extended bandwidth does not present a problem for the actuator and sensor dynamics, preventing that their dynamics are sufficiently fast relative to the dynamics of the nominal closed-loop system [84, 70]. However, it is a problem when the observer estimates derivatives of noisy signals.

In the following, we resume an example from [69] to show some properties of the design of output feedback controllers using high-gain observers.

Let a second order nonlinear system have the form

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= \phi(x, u), \\ y &= x_1. \end{aligned} \tag{3.17}$$

Let $u = \gamma(x, y)$ be a locally Lipschitz state feedback control law that stabilizes the origin $x = 0$.

Let the high-gain observer be defined using only measurements of the output y

$$\begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 + \frac{\alpha_1}{\varepsilon}(y - \hat{x}_1), \\ \dot{\hat{x}}_2 &= \phi_0(\hat{x}, u) + \frac{\alpha_1}{\varepsilon^2}(y - \hat{x}_1), \\ u &= \gamma(\hat{x}, y), \end{aligned} \tag{3.18}$$

where $\phi_0(x,u)$ nominal model of the nonlinear function $\phi(x,u)$.

To satisfy the conditions of the separation principle in [2], the state feedback control and the observer must fulfill the following assumptions [70]:

Assumption 3.5.1. $\gamma(0,0)=0$, $\gamma(x,y)$ is a twice-continuously differentiable function over the domain of interest and a globally bounded function of x, and the origin $(x=0)$ is an exponentially stable equilibrium point of the closed-loop system

$$\begin{aligned}\dot{x}_i &= x_{i+1}, \quad 1 \le i \le r-1,\\ \dot{x}_r &= \phi(x, \gamma(x,y)),\end{aligned}$$

where r is the relative degree of the system.

Assumption 3.5.2. $\phi_0(0,0)=0$, $\phi_0(x,u)$ is a twice-continuously differentiable function over the domain of interest and a globally bounded function of x, ε is a positive constant, and α_1 to α_r are positive constants chosen such that the roots of

$$s^r + \alpha_1 s^{r-1} + \ldots + \alpha_{r-1} s + \alpha_r = 0$$

have negative real parts.

It follows from the separation principle that, for sufficiently small ε, the output feedback controller stabilizes the origin of the closed-loop system and recovers the performance of the state feedback controller in the sense that the trajectories of x under output feedback approach those under state feedback as ε tends to zero.

Let the estimation error be given by

$$\tilde{x} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix} = \begin{bmatrix} x_1 - \hat{x}_1 \\ x_2 - \hat{x}_2 \end{bmatrix},$$

then the error dynamics are

$$\begin{aligned}\dot{\tilde{x}}_1 &= -h_1\tilde{x}_1 + \tilde{x}_2\\ \dot{\tilde{x}}_2 &= -h_2\tilde{x}_1 + \delta(x,\tilde{x})\end{aligned}$$

with $\delta(x,\tilde{x}) = \phi(x,\gamma(\hat{x})) - \phi_0(\hat{x},\gamma(\hat{x}))$. Let the *scaled estimation errors* be defined

as

$$\eta_1 = \frac{\tilde{x}_1}{\varepsilon},$$
$$\eta_2 = \tilde{x}_2.$$

Using the scaled estimation errors, we can define the singularly perturbed system

$$\begin{aligned} \varepsilon\dot{\eta}_1 &= -\alpha_1\eta_1 + \eta_2, \\ \varepsilon\dot{\eta}_2 &= -\alpha_2\eta_1 + \varepsilon\delta\left(x, \tilde{x}\right). \end{aligned} \tag{3.19}$$

In (3.19), note that small values of ε reduce the effects of the error model $\delta(\cdot)$, make η much faster than x, and produce *peaking phenomenon* given by a response of the form

$$\frac{1}{\varepsilon}\exp\left(-\frac{\alpha t}{\varepsilon}\right).$$

This peaking phenomenon can be avoided by saturating the control input

$$u = \text{sat}\left(\gamma\left(x\right)\right).$$

Finally, the stability analysis is performed over the complete system

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= \phi\left(x, \gamma\left(\hat{x}\right)\right), \\ \varepsilon\dot{\eta}_1 &= -\alpha_1\eta_1 + \eta_2, \\ \varepsilon\dot{\eta}_2 &= -\alpha_2\eta_1 + \varepsilon\delta\left(x, \tilde{x}\right). \end{aligned}$$

Due to the separation principle, the stability analysis can be performed into two parts:

- The stability analysis of the slow motion system (x_1, x_2), with $\varepsilon = 0$, using an appropriate Lyapunov function $V(x)$.
- The stability analysis of the fast motion system $\varepsilon\dot{\eta} = A_0\eta$, with $A_0 = \begin{bmatrix} -\alpha_1 & 1 \\ -\alpha_2 & 0 \end{bmatrix}$, using the Lyapunov function $W(\eta) = \eta^T P_0 \eta$, with P_0 satisfying

$$P_0 A_0 + A_0^T P_0 = -I_4.$$

3.5.1 Simulation example

Let us consider the nonlinear system system (3.11) and the high-gain observer

$$\begin{aligned}\dot{\hat{x}}_1 &= \hat{x}_2 + \frac{2}{\varepsilon}(y - \hat{y}),\\ \dot{\hat{x}}_2 &= \frac{1}{\varepsilon^2}(y - \hat{x}_1),\\ \hat{y} &= \hat{x}_1.\end{aligned}$$

Figure 3.7 shows simulation results using control law (3.12) and different values of gain ε. As can be seen, smaller values of ε increase the convergence rate and the overshoot. This effect is clearly noticed in Figure 3.8. The plot at the top right corner shows that the control input increases significantly when ε decreases. Despite their short time duration, these large values of the control input may cause closed-loop instability.

Figure 3.9 shows same results for the case of input saturation

$$u = \text{sat}\left(-\hat{x}_2^3 - \hat{x}_1 - \hat{x}_2\right),$$

as can be seen, the performance has been improved significantly.

Figure 3.10 shows results with saturated inputs and noise. Because the HGO increases control bandwidth, the results are notoriously degraded.

3.6 Higher-order sliding mode (HOSM) observers

Sliding mode control has been a key approach to solve problems under heavy uncertainty conditions [70, 121, 135, 136]. To solve these problems, the sliding mode approach keeps a properly chosen constraint by means of high-frequency control switching. The main features of the sliding mode approach are insensitivity to external and internal disturbances, ultimate accuracy and finite-time transient. Notwithstanding, the sliding has some restrictions: If the sliding mode surface, or constraint, is defined by an equality $s = 0$, where s is a system output described by a smooth function, the standard sliding mode may be implemented only if the relative degree of s is 1. Moreover, high frequency control switching leads to the so-called *chattering effect* which can be dangerous in real applications.

In recent years, a new generation of controllers based on higher-order sliding-mode (HOSM) theory has been developed [5, 6, 7, 30, 49, 76, 78]. HOSM generalizes

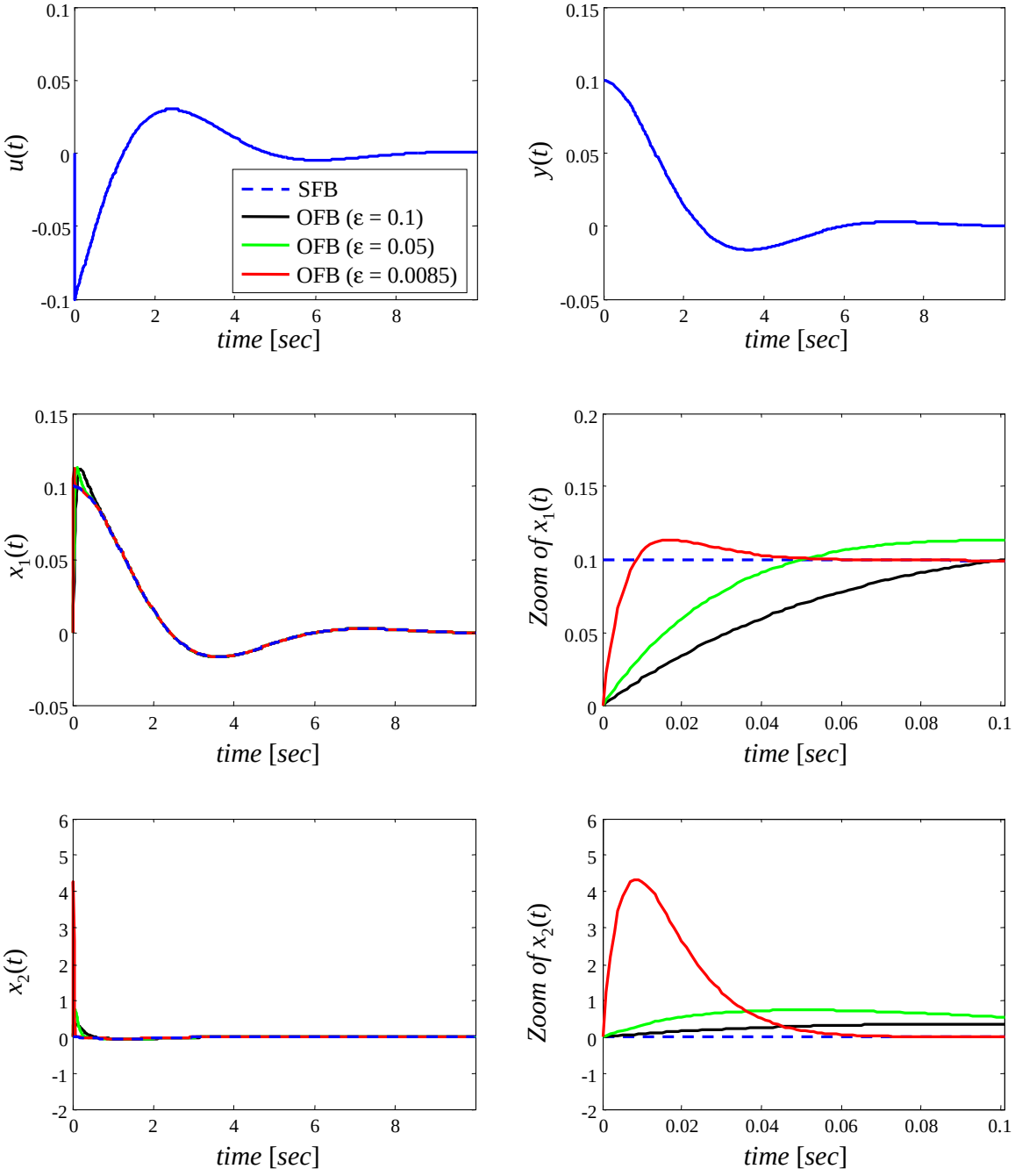


Figure 3.7: HGO with exact control law.

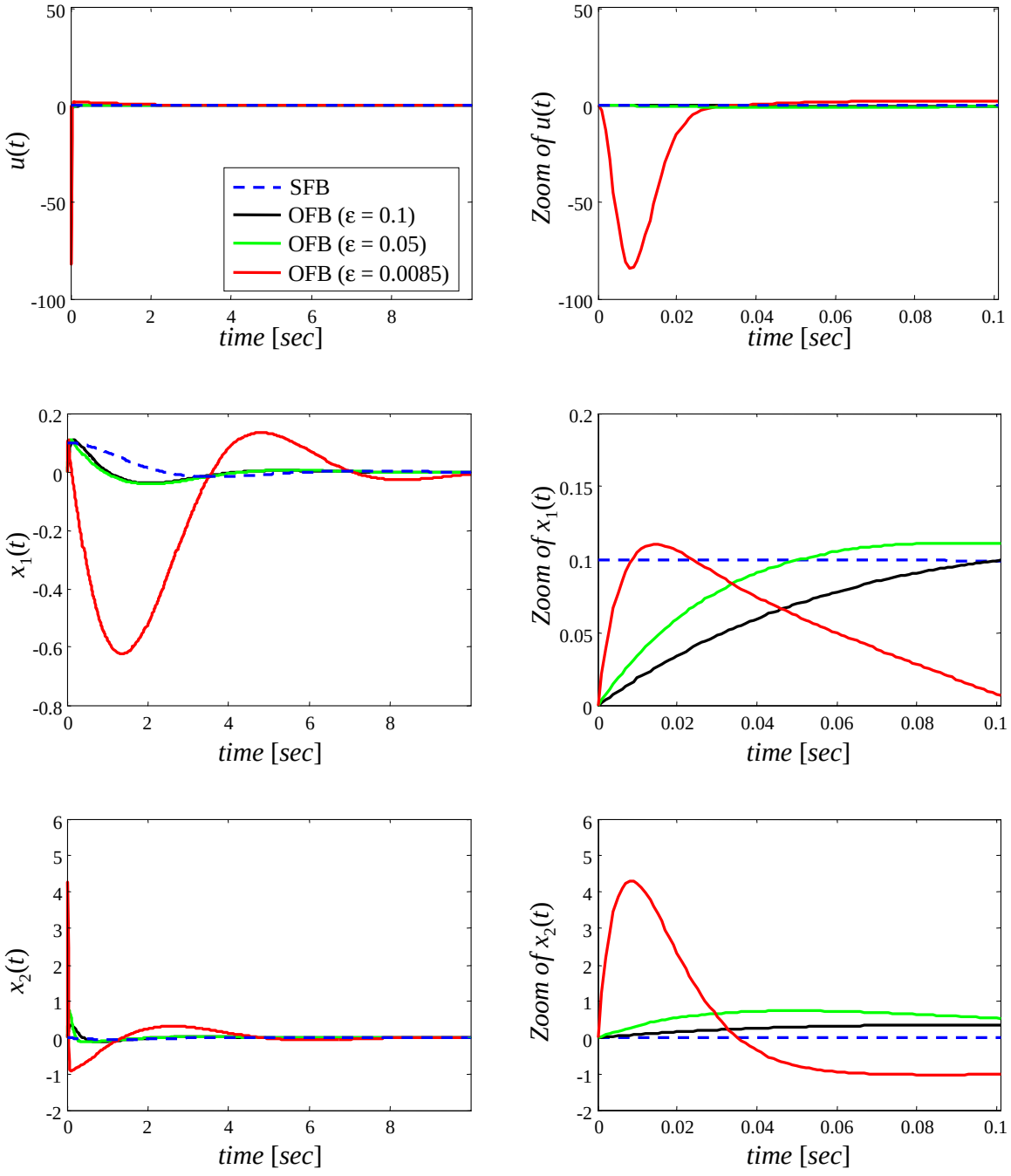


Figure 3.8: HGO behavior with unsaturated inputs.

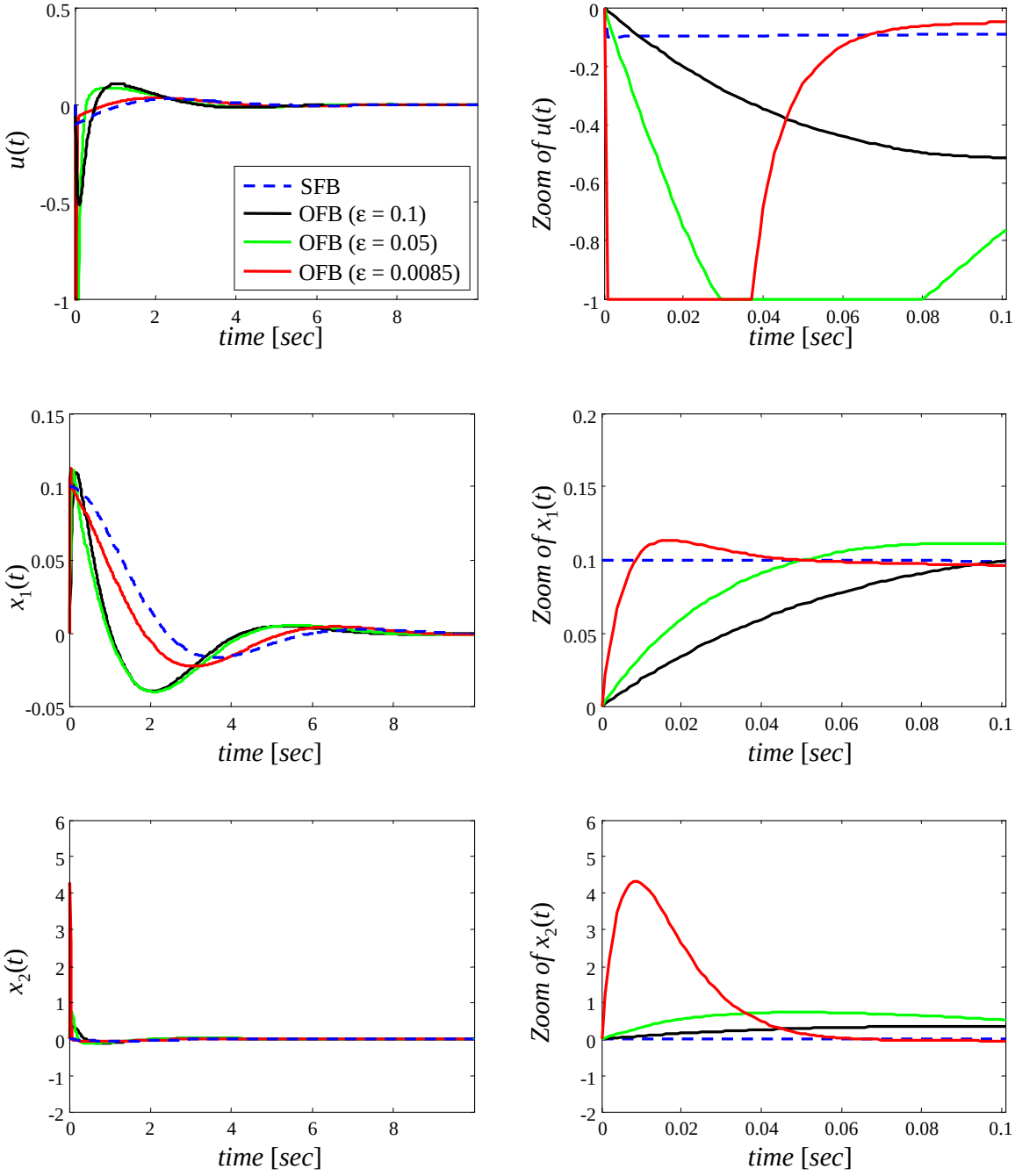


Figure 3.9: HGO behavior with saturated inputs.

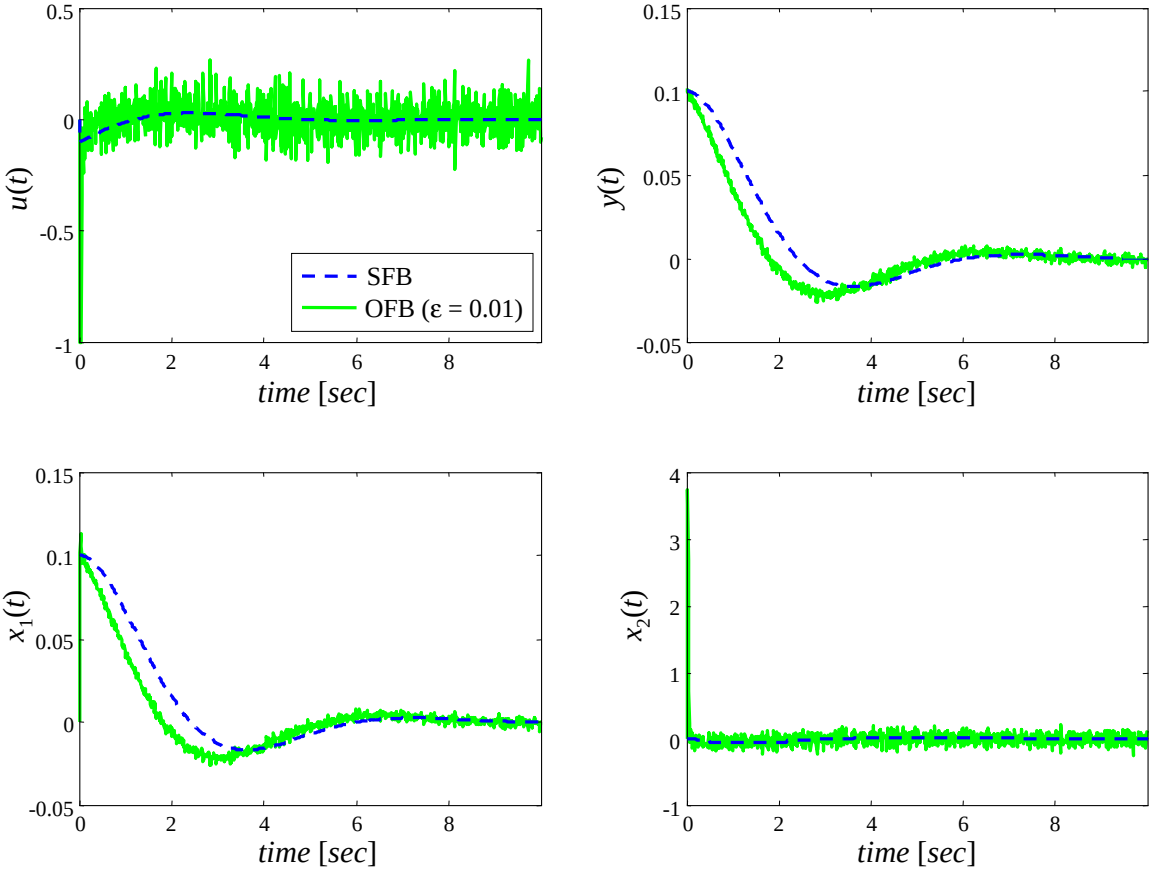


Figure 3.10: HGO behavior with saturated inputs and noise.

the basic sliding mode idea, acting on higher order time derivatives of the system deviation from the sliding surface instead of affecting the first deviation derivative, as standard sliding modes do.

The rth order sliding mode is determined by the equalities

$$s = \dot{s} = \ddot{s} = \ldots = s^{(r-1)} = 0,$$

where s, $\dot{s}$, $\ddot{s}$, ..., $s^{(r-1)}$ are continuous functions. In fact, the sliding mode motions also satisfy the equality $s^{(r)} = 0$, but as a result of some averaging process.

In particular, HOSM theory has been applied to solve the differentiation problem in [75, 76]. The problem can be stated as follows: Let input signal $f(t)$ be a function defined on $[0, \infty)$ consisting of a bounded Lebesgue-measurable noise with unknown features and an unknown base signal $f_0(t)$ with the nth derivative having a known Lipschitz constant $L > 0$. The problem is to find real-time robust estimations of $\dot{f}_0(t)$, $\ddot{f}_0(t)$, ..., $f_0^{(n)}(t)$ which should be exact in the absence of measurement noises.

The differentiator based on HOSM designed in [76] can be resumed as

$$\begin{aligned}
\dot{z}_0 &= v_0, \quad v_0 = -\lambda_0 |z_0 - f(t)|^{n/(n+1)} \operatorname{sign}(z_0 - f(t)) + z_1, \\
\dot{z}_1 &= v_1, \quad v_1 = -\lambda_1 |z_1 - v_0|^{(n-1)/n} \operatorname{sign}(z_1 - v_0) + z_2, \\
&\vdots \\
\dot{z}_{n-1} &= v_{n-1}, \quad v_{n-1} = -\lambda_{n-1} |z_{n-1} - v_{n-2}|^{1/2} \operatorname{sign}(z_{n-1} - v_{n-2}) + z_n, \\
\dot{z}_n &= -\lambda_n \operatorname{sign}(z_n - v_{n-1}).
\end{aligned} \tag{3.20}$$

The main properties of this differentiator are summarized by the following two theorems and a lemma.

Theorem 3.6.1 ([76]). *Given differentiator (3.20). If the parameters are properly chosen, then the following equalities are true in the absence of input noises after a finite transient process*

$$\begin{aligned}
z_0 &= f_0(t), \\
z_i &= f_0^{(i)}(t), \quad i = 1, \ldots, n.
\end{aligned}$$

Theorem 3.6.2 ([76]). *Let input noise satisfy the inequality $|f(t) - f_0(t)| \leq \varepsilon$. Then the following inequalities are established in finite time for some positive con-*

stants μ_i, ν_i depending on the parameters of the differentiator

$$\left|z_i - f_0^{(i)}(t)\right| \leq \mu_i \varepsilon^{(n-i+1)/(n+1)}, \quad i = 0, \ldots, n,$$
$$\left|v_i - f_0^{(i+1)}(t)\right| = \nu_i \varepsilon^{(n-i)/(n+1)}, \quad i = 0, \ldots, n-1.$$

Lemma 3.6.3 ([94]). *Consider the observer (3.20). If $\left|f^{(n+1)}(t)\right| \leq k_{n+1}$, for all $t < \infty$, with $k_{n+1} \in \mathbb{R}_{\geq 0}$, Then the observed variables z_i, $i = 1, \ldots, n$, cannot diverge in finite time.*

There are two main differences between observers (3.18) and (3.20): The first difference is the finite-time convergence property of the HOSM observer, in contrast to the asymptotic convergence property of the HGO observer. This property implies that the separation principle is no longer necessary. The second difference is the boundedness of the control law. The HOSM observer is bounded by design. However, the input of the HGO observer might be saturated to avoid peaking phenomenon. It should be noticed that the only requirements for the implementation of the HOSM differentiator is the boundedness of some higher-order derivative of its input.

3.6.1 Simulation example

Let us consider the nonlinear system system (3.11) and the HOSM (3.20). Figures 3.11 to 3.14 show simulation results with the HOSM. Figure 3.11 shows simulation results using control law (3.12), and the HOSM observer

$$\begin{aligned}
\dot{\hat{x}}_1 &= v_1, \quad v_1 = \hat{x}_2 - \lambda_0 \left|\hat{x}_1 - y\right|^{1/2} \operatorname{sign}(\hat{x}_1 - y), \\
\dot{\hat{x}}_2 &= -\lambda_1 \operatorname{sign}(\hat{x}_2 - v_1), \\
\hat{y} &= \hat{x}_1
\end{aligned} \tag{3.21}$$

with $\lambda_0 = 2.8$ and $\lambda_1 = 0.75$. There exist a tradeoff between convergence and sensitivity to input noise: the larger the parameters, the faster the convergence and the higher the sensitivity to input noise and sampling step.

As can be seen in Figure 3.11, the estimated derivative of the output $\hat{x}_2$ presents a notorious chattering effect. This effect is smoothened in its integral value $\hat{x}_1$. Consequently, an option to deal with the chattering problem is to computed a higher order

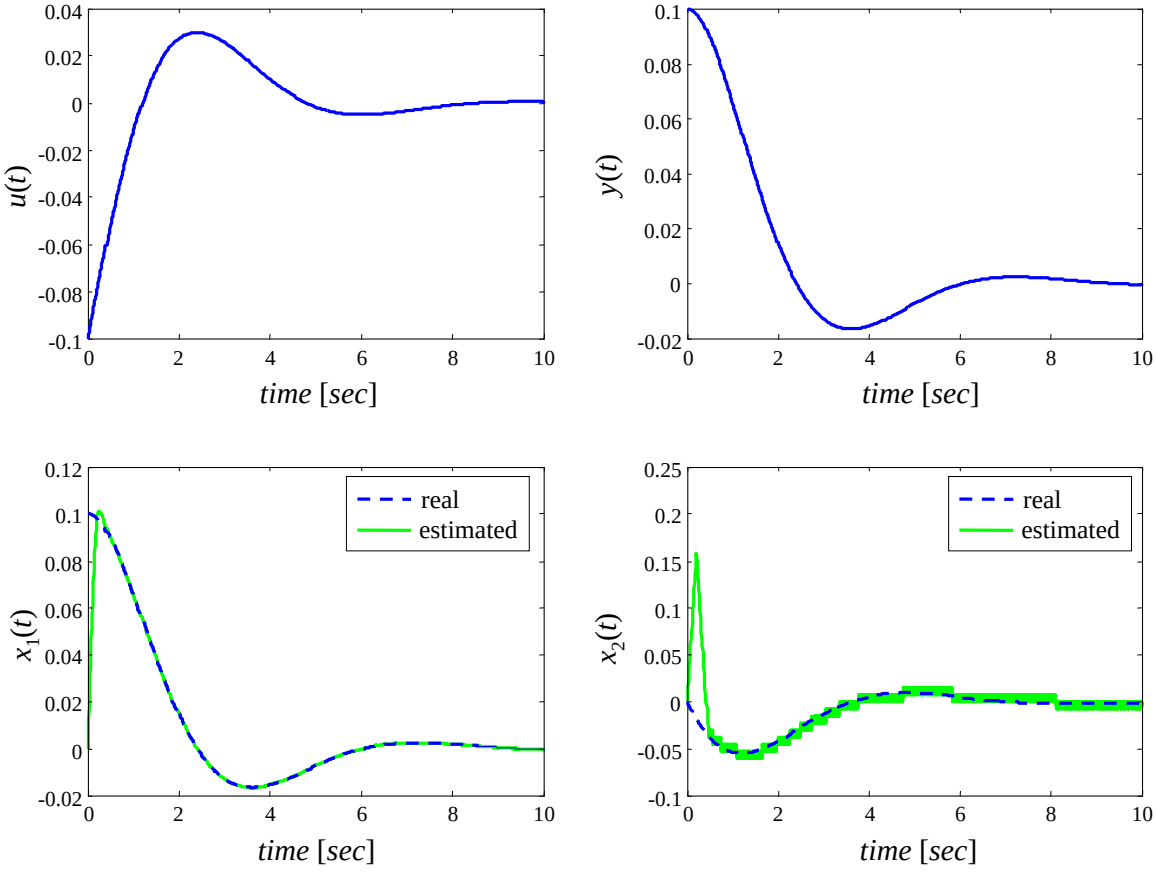


Figure 3.11: Simulation results with a first order HOSM.

derivative using observer (3.20), that is

$$\begin{aligned}
\dot{\hat{x}}_1 &= v_1, \quad v_1 = -\lambda_0 \left|\hat{x}_1 - y\right|^{2/3} \operatorname{sign}\left(\hat{x}_1 - y\right) + \hat{x}_1, \\
\dot{\hat{x}}_2 &= v_2, \quad v_2 = -\lambda_1 \left|\hat{x}_2 - v_1\right|^{1/2} \operatorname{sign}\left(\hat{x}_2 - v_1\right) + \hat{x}_2, \\
\dot{\hat{x}}_3 &= -\lambda_3 \operatorname{sign}\left(\hat{x}_3 - v_2\right).
\end{aligned} \tag{3.22}$$

Figure 3.12 shows the results of this differentiator with $\lambda_0 = 7.5$, $\lambda_2 = 5.0$, and $\lambda_3 = 1.5$. Clearly, this version outperforms the previous one.

Figure 3.13 shows results of HOSM (3.22) with measurement noise on the output. As can be seen, the results are similar to the EKF but without using a priori information about the system.

Figure 3.14 shows results with the control law (3.13), the results are quite satisfactory.

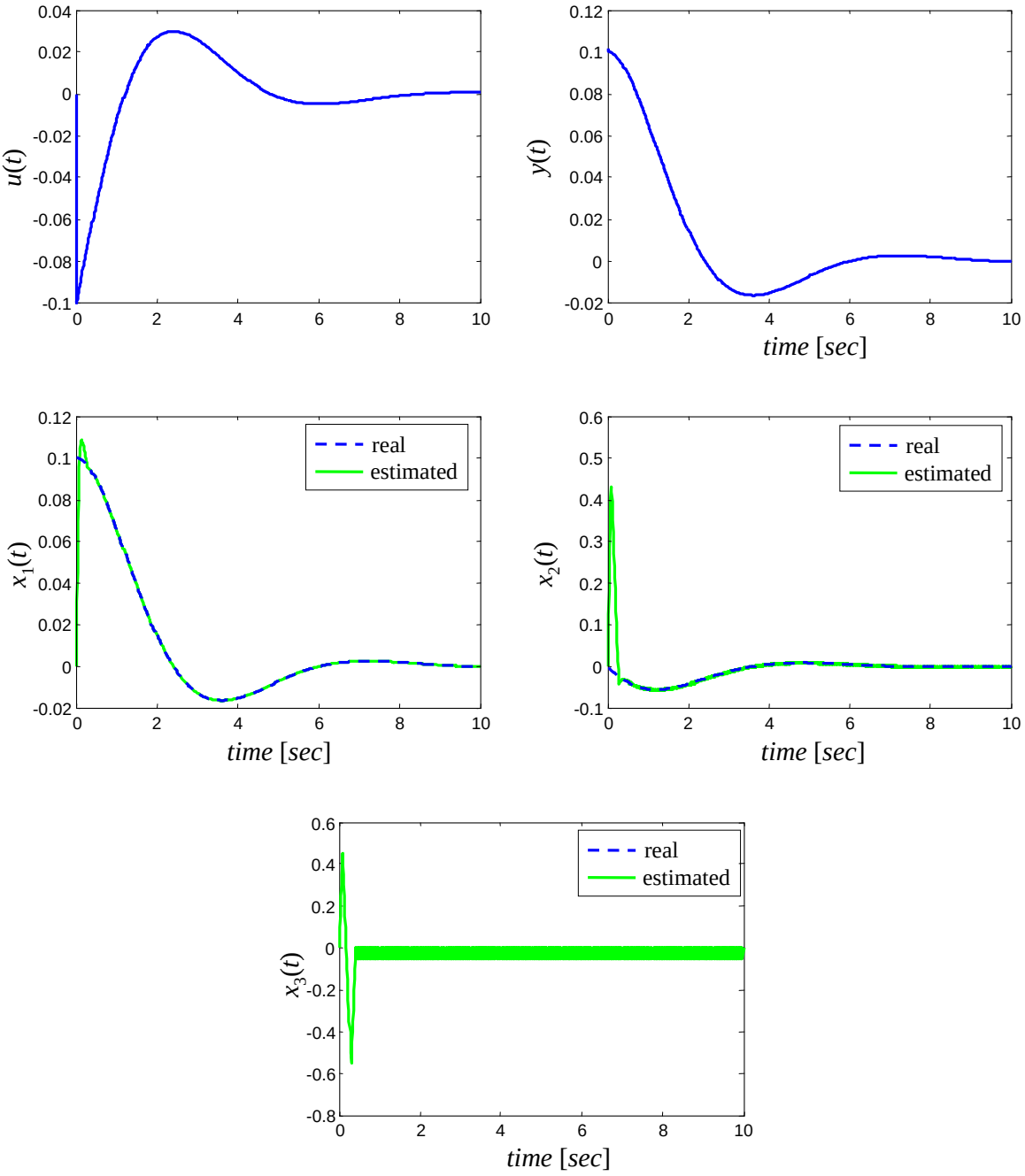


Figure 3.12: Simulation results with a second order HOSM.

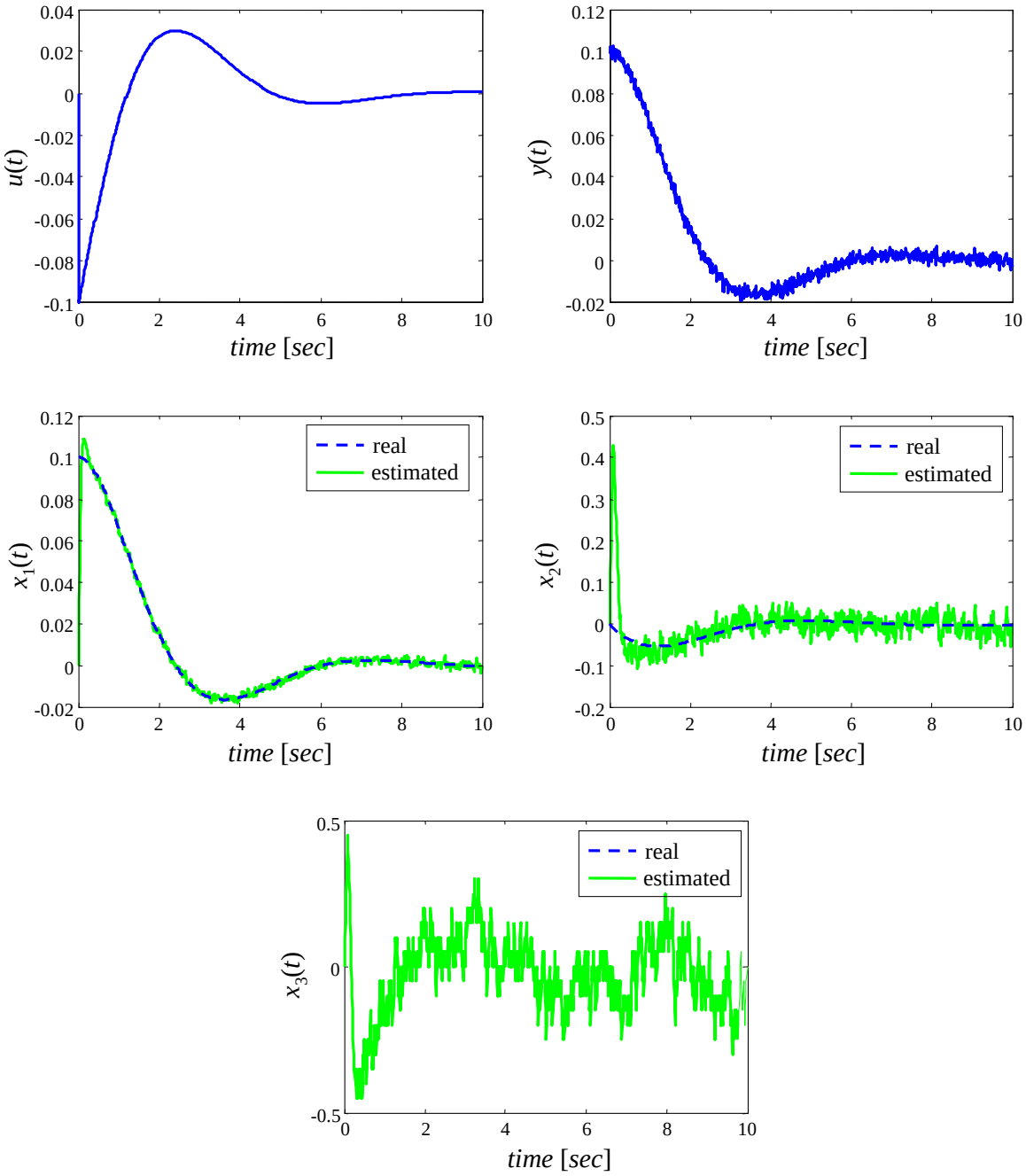


Figure 3.13: Simulation results with a second order HOSM and measurement noise.

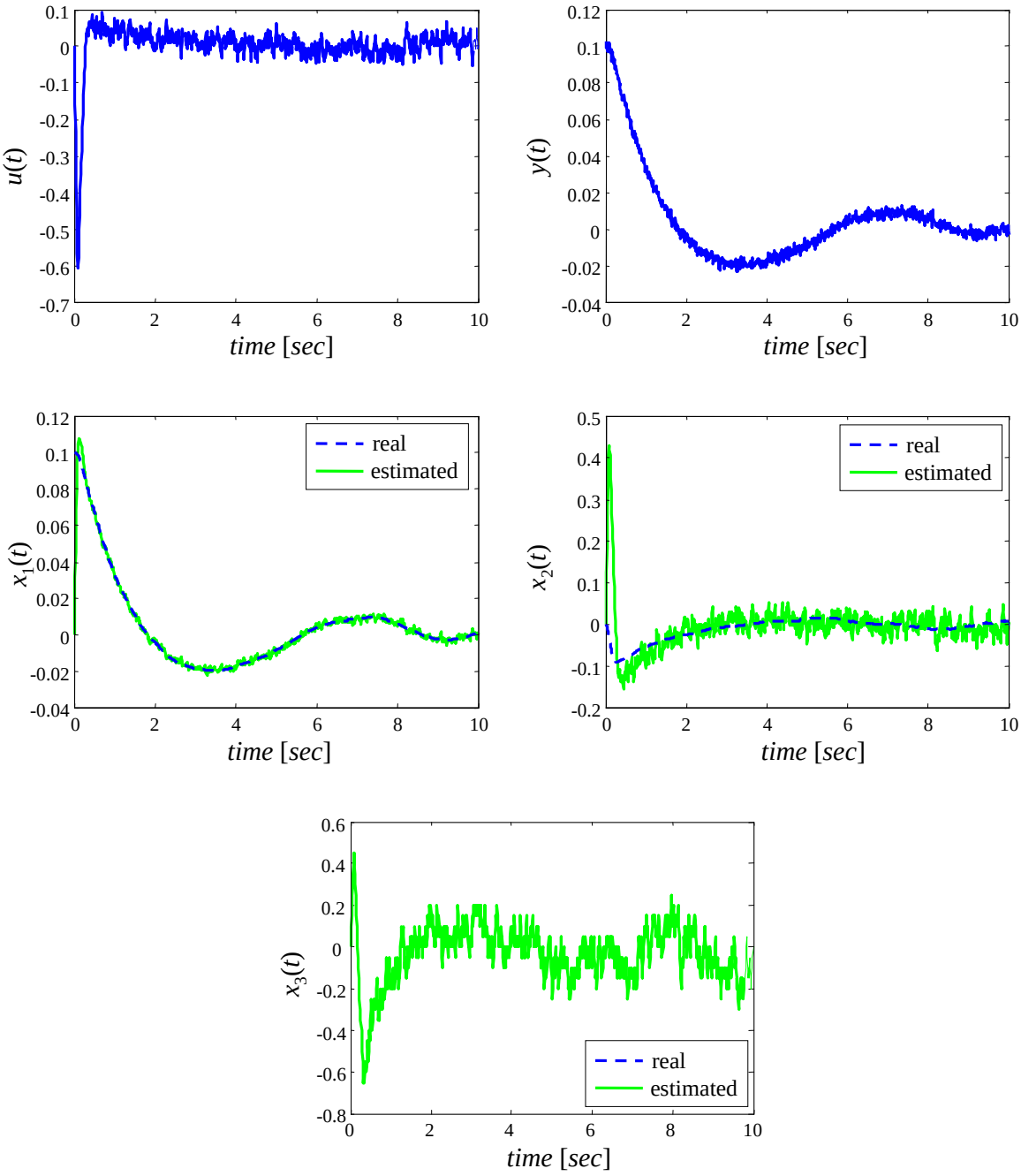


Figure 3.14: Closed-loop simulation results with a second order HOSM.

Part II

Main Contributions

Chapter 4

Mobile robot vision

Localization in mobile robot formations has drawn the attention of many researchers during the last few years. Traditionally, the control system design relies on measurements from dead reckoning sensors. However, these measurements are completely unreliable after few meters of navigation due to encoder's low accuracy and drift.

Due to reduced cost and flexibility, the current trend is the design of systems using a single or a pair of cameras to determine the relative position of a robot with respect to another robot or object [15, 24, 39, 42, 53, 74, 83, 87]. For example, cameras can recognize and distinguish between objects of similar shapes, types, and colors, a task impossible to perform with range or acoustic sensors. Nonetheless, vision-based control poses new particular challenges. For instance, the controller has to be robust to camera calibration errors, nonlinearities arising from camera models, and intrinsic ambiguities of vision sensors.

Localization in mobile robot formations using machine vision is part of what is known as the *visual target tracking* problem. This tracking can be realized with a fixed camera or with a mobile camera. The visual target tracking problem can be divided into two main components: *target detection*, or target *segmentation*, and *pose* (distance and orientation) *estimation* [53].

There are several approaches for target detection in the literature. For example, in [83], the author presents a method that consists in extracting distinctive features invariant to image scale and rotation. The recognition process is divided in three steps: The first step is a matching process between individual features and a feature database of known objects using a fast nearest-neighbor algorithm. The second step identifies clusters belonging to a single object through the computation of the Hough transform. Finally, the third step verifies consistency of pose parameters using a least-squares solution. In [117], the method presented in the previous paper is used

for global localization using landmarks. An alternative approach that uses feature points is the well-known and very robust SIFT algorithm [83, 87]. This method is invariant to scale, robust to noise and occlusions, but very computationally intensive. It takes around 3s to compute features with a Pentium IV, making it inadequate for online applications.

To cope with this problem, robust target detection is often achieved through the use of artificial fiducial markers. Methods using markers are referred as *model-based pose estimation* methods, as classified in [57]. In practice, three-dimensional positions of features points on the object are measured a priori and stored in a database. Then, the online detection problem consists in detecting and relating image points with the a priori selected features in order to compute the pose of the object.

The problem of pose estimation using fiducial markers has been mainly addressed by tangible augmented reality system applications [24, 29, 67]. The goal of these methods is to combine virtual 3D representations with the real world for active interaction. In these systems, markers are distributed around a synthetic environment to help localize and relate the virtual camera position with the real position in the world. In [42], the author presents a vision-based system for controlling multiple robot platforms in real time using planar markers and a top-view video camera for on-line pose estimation. In [16, 17, 38], the authors develop a monocular-camera-based visual servo tracking controller for a mobile robot subject to nonholonomic motion constraints. The algorithm defines a desired trajectory for the vehicle with a sequence of prerecorded images of three target points. In [53], the authors present a visual tracking scheme that detects the target contour using a shape adaptive sum-of-squared difference algorithm. The target velocity is decomposed into a component caused by the target motion and a component caused by the camera motion. The latter component is computed using the image Jacobian, allowing an accurate estimation of the target position in the following image.

There exist few applications of machine vision to the formation control problem. In [22, 114], follower robots estimate position and orientation of their leaders using a color-tracking algorithm using off-the-shelf cameras. Each robot is equipped with a color pattern consisting of a central purple rectangle and two lateral green rectangles. The central rectangle provides an estimate of the distance based on measured and real heights. The difference between the perceived heights of the lateral rectangles provides an estimate of the orientation of the pattern with respect to the observing robot. This algorithm is not robust to noise or pixel vibrations. In [27, 28], an on-board catadioptric camera system is used. The omni-directional images obtained are used

to input data to state observers. In [26, 138, 139], the authors use segmentation from multiple central views to keep track of which pixels correspond to each leader from pictures taken with omni-directional cameras. Pose estimation of each leader in the image plane of the follower is performed by rank constraint on the central panoramic optical flow across multiple frames. It is well-know the massive computational power requirements of this technique. In [88, 89], the authors present an algorithm for flocking based on the measurement of time-to-collision and optical flow each robot does. In these works there is no stability analysis including the effects of measurement errors and robust methods for image interpretation.

This Chapter presents a vision-based framework for mobile robot detection and tracking using off-the-shelf cameras mounted on mobile robots. Target detection and pose estimation are performed from single frames using fiducial markers as key elements. The method consists in distributing an octagon shaped structure on the back part of each robot. These shapes are easy to extract from the images and posses unique ID codes to facilitate pose estimation. Three pose estimation methods are programmed and compared: the PRA algorithm [98], Lowe's algorithm [82, 15], and a modified POSIT algorithm [31, 95]. Finally, a dual Unscented Kalman filter (DUKF) is implemented to smooth measured data and estimate unknown leader's velocities.

The rest of the Chapter is organized as follows: Section 4.1 summarizes the visual system and the pose estimation problem. Section 4.2 presents the ID detection approach. Section 4.3 describes our pose estimation algorithm. Section 4.4 reviews the leader-follower model and the dual unscented Kalman filter. Section 4.5 shows simulation results for the different steps involved in this framework. Finally, we present our concluding remarks and future work in Section 4.6.

4.1 System overview

The visual tracking problem is divided in *target detection* and *pose estimation*. Target detection is related to image processing, whereas pose estimation is related to vision and nonlinear filtering.

The detection process is the most time consuming, but it can be simplified by using fiducial markers. In our vision system, MRVision for mobile robot vision[1], markers are distributed on the back part of each robot on a truncated octagon shaped structure, as can be seen in Figure 4.1. Each face of this shape has a code that not only identifies

[1] http://orqueda.net/research.aspx

the face, but also its position on the robot, as it is explained in the next Section. The ink patter is black and white to reduce lighting and camera sensitivity requirements.

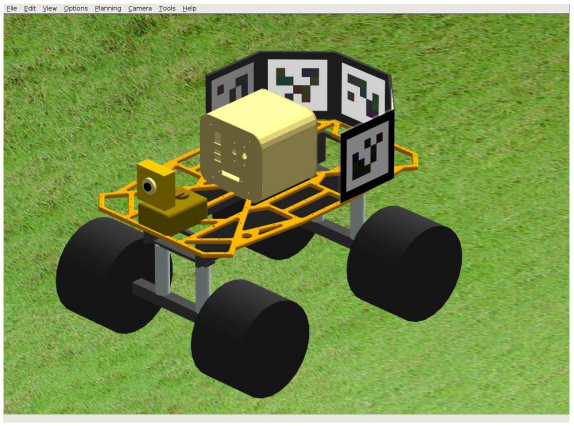


Figure 4.1: Scheme of the TXT platform in MPSLab.

The vision library consists of synthetic and real cameras and depends on the vision library OpenCV [11] and on MPSLab, a motion planning, simulation, and virtual perception library for simulating systems in 3D environments [99, 100]. Synthetic cameras are used for tuning and testing purposes, see Figure 4.2, and real cameras are used with a PC104 for the on-board image capture and processing system. Currently, Firewire IEEE-1394 and USB 2.0 cameras are supported, in particular, Unibrain's Fire-I, Point Grey's Bumblebee, and Logitech 3000 cameras, see Figure 4.3.

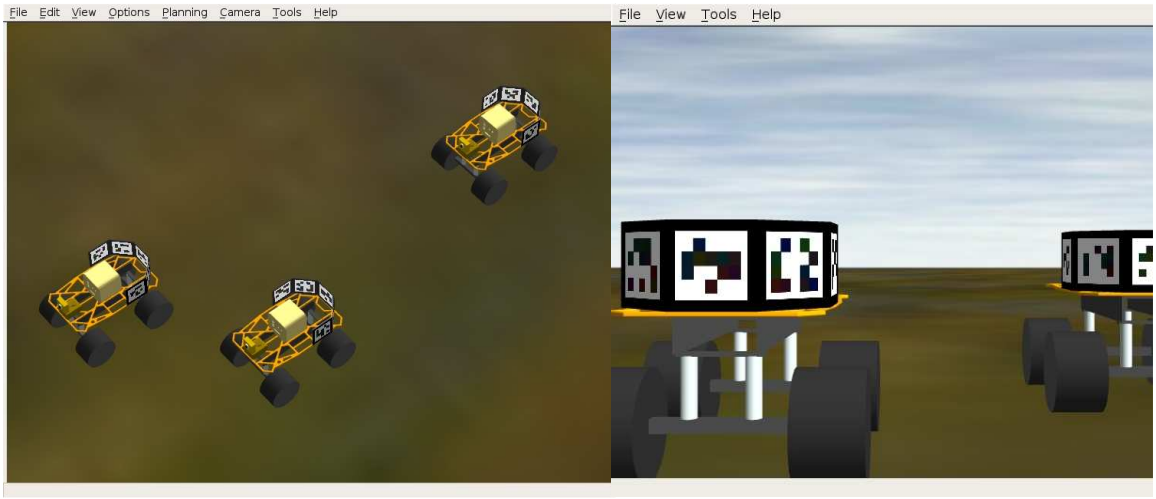


Figure 4.2: Synthetic camera screen shot.

Vision processing can be divided in the tasks resumed in the flow chart of Figure 4.4:

Figure 4.3: Logitech 3000 USB camera (bottom-left), PointGrey's Bumblebee (top), and Unibrain's Fire-I (bottom-right) FireWire IEEE-1394 cameras.

1. *Video capture and transformation:* Images from several sources with different formats (YUV, RGB, etc.) are transformed into OpenCV's IplImage RGB format.

2. *Filtering and thresholding:* The IplImage image is converted to grayscale. Then, a thresholding based on a modified Otsu method is performed on the grayscale image.

3. *Point selection/face labeling:* A search and identification of each face marker in the binary image is performed. First, all the contours on the binary image are extracted. Second, angular changes in each contour close to 90° are computed to determine the square enclosing each marker. After estimating the four corners of each square, a search for markers is performed. If a valid marker is found, the corresponding four corners are stored to be used for pose estimation.

4. *Pose estimation:* Using the algorithms presented in Section 4.3, the position of the leader robot with respect to the camera position (x_c, y_c, z_c) and the relative rotation of the leader robot with respect to the camera plane θ_m are computed, see figures 2.2 and 4.5 for the definition of the variables.

5. *Transformation*: The desired values are computed using (x_c, y_c, z_c), θ_m, and

the value of the camera pan θ_p, as

$$\ell_{ij} = \sqrt{x_c^2 + y_c^2 + z_c^2}, \tag{4.1}$$

$$\theta_t = \arctan\left(\frac{-x_c}{z_c}\right), \tag{4.2}$$

$$\alpha_{12} = \theta_t - \theta_m, \tag{4.3}$$

$$\theta_{12} = \theta_p + \theta_m. \tag{4.4}$$

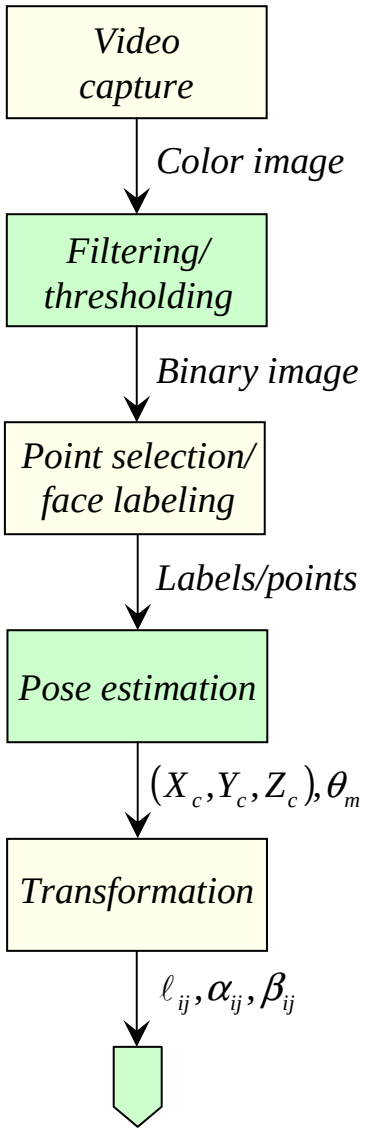


Figure 4.4: Vision processing flow chart.

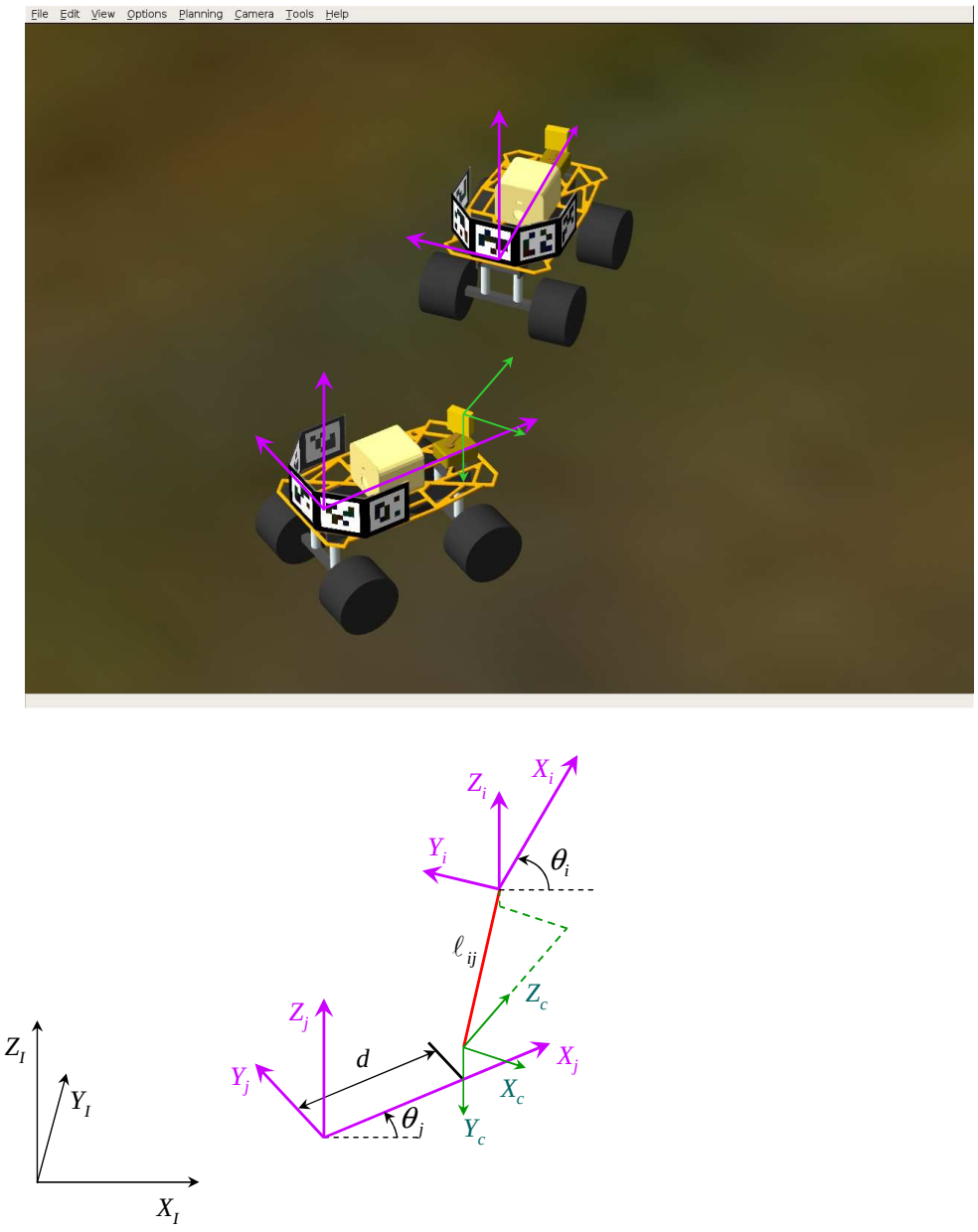


Figure 4.5: Camera and robot frames.

4.2 Marker and ID detection

The steps involved in marker and ID detection processes are:

1. Image capturing.
2. Grayscale conversion.
3. Thresholding.
4. Contour extraction and selection.
5. ID recognition.

Figure 4.6 shows steps 1 to 5 applied to a synthetic image. In the following, we will resume the main characteristics of these steps.

4.2.1 Image capture

Image capture is performed using standard drivers. MRVision is programmed in C++ and works on Linux and Windows platforms. It can capture images from synthetic cameras, Firewire IEEE-1394 and USB 2.0 compatible cameras. After capture, the image is converted to OpenCV's IplImage format [11].

4.2.2 Grayscale conversion

Grayscale conversion is necessary to reduce the amount of data. This conversion is performed by simply extracting the green channel of the original color image. The use of just one channel is justified by the low-level of information loss. In Figure 4.7, it can be seen a comparison of different methods of grayscale conversion. Figure 4.7a shows the color image, Figure 4.7b, c, and d, show the red, green, blue channel images respectively. As can be seen, the blue channel image loss the most information. Figure 4.7e if the mean value image. This image does not justify the computational burden.

4.2.3 Thresholding: The Otsu method

Thresholding, or *bilevel thresholding*, segments an image into two brightness regions: background and object. That is, for a gray level image $I(x, y)$, with m gray levels

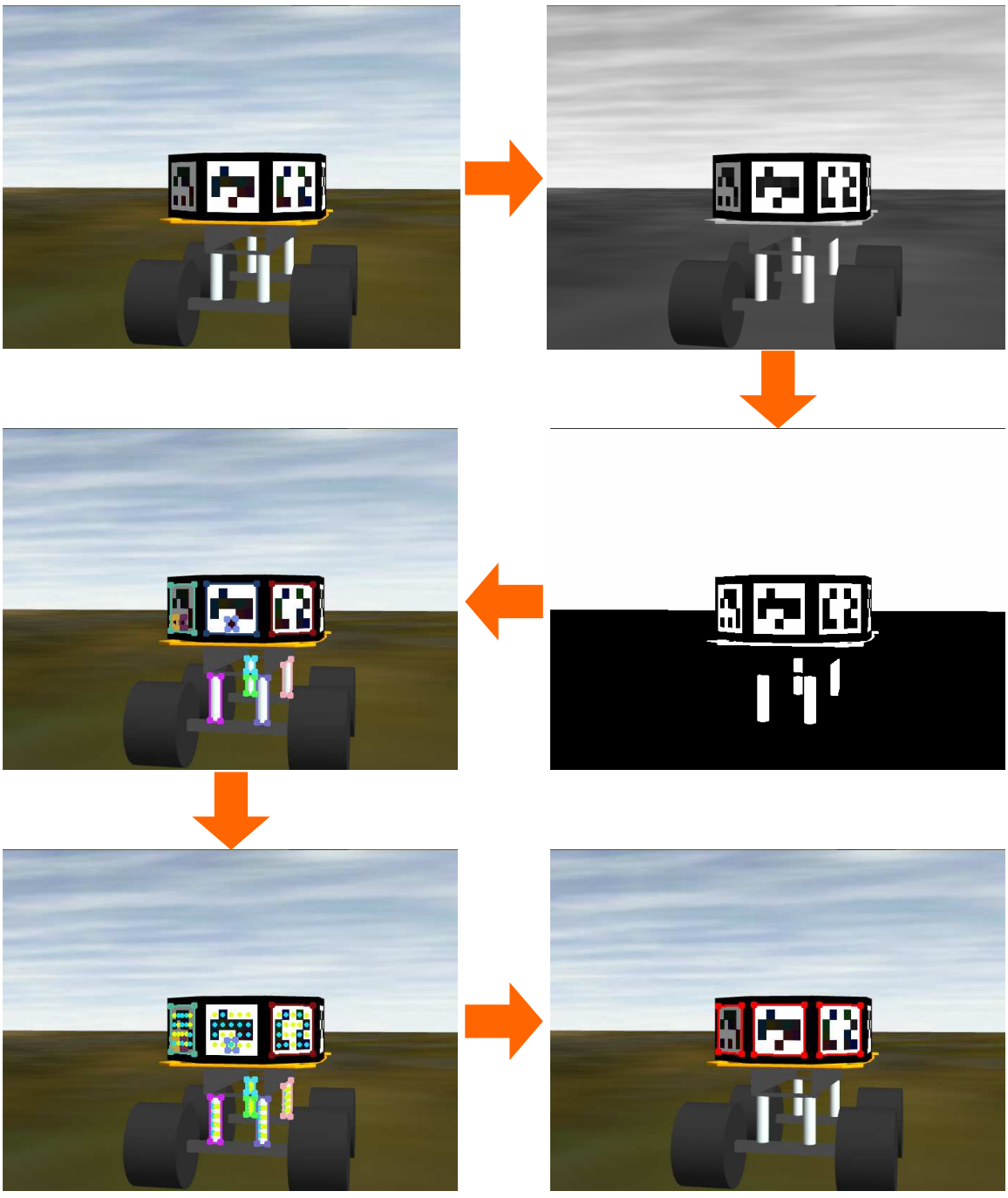


Figure 4.6: Processing sequence of the octagon shape.

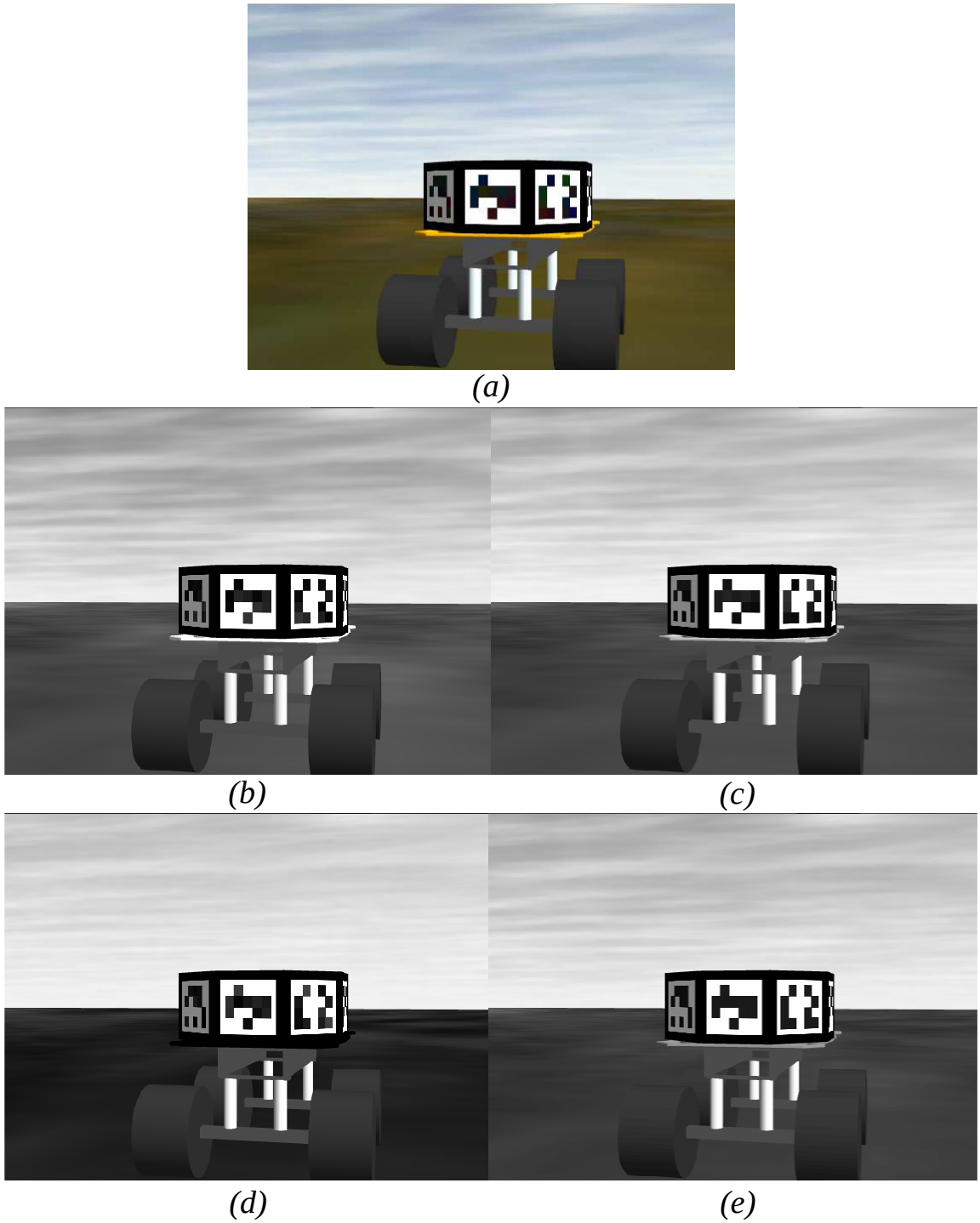


Figure 4.7: Grayscale conversion. (a) Color image, (b) red channel, (c) green channel, (d) blue channel, and (e) mean value image.

$0, 1, \ldots, m-1$, bilevel thresholding is to transform $I(x,y)$ into a binary image $I_b(x,y)$ by a threshold j, $0 \leq t \leq m-1$, such that

$$I_b(x,y) = \begin{cases} 0, & \text{if } I(x,y) \leq t, \\ 1, & \text{if } I(x,y) > t. \end{cases}$$

Traditionally, a fixed thresholding value is used. However, in our application, lighting variations, different camera properties and environmental conditions make the selection of this value very difficult. For this reason, we use a modified version of the method due to Otsu [21, 56, 107]. This method formulates the threshold selection problem as a discriminant analysis. The method divides the gray level image histogram in two groups, A and B, then selects the threshold as the point of maximum variance between both groups.

Let n_i be the number of pixels with a gray level value i and $n = \sum_{i=0}^{m-1} n_i$ be the total number of pixels in the image $I(x,y)$. Let $P_i = \frac{n_i}{n}$ be the probability of occurrence of grey-level i. Let $\omega_1(j)$, $M_1(j)$, $\omega_2(j)$, $M_2(j)$ be the number of pixels and the average gray level value in group A and group B, respectively. Then

$$\omega_1(j) = \sum_{i=0}^{j} n_i, \quad M_1(j) = \frac{\sum_{i=0}^{j} i \cdot n_i}{\omega_1(j)},$$
$$\omega_2(j) = \sum_{i=j+1}^{m-1} n_i, \quad M_2(j) = \frac{\sum_{i=j+1}^{m-1} i \cdot n_i}{\omega_2(j)},$$

Expressing the average gray level value M_T of all pixels in image $f(x,y)$ as

$$M_T = \frac{\omega_1(j) M_1(j) + \omega_2(j) M_2(j)}{\omega_1(j) + \omega_2(j)},$$

the variance between the two groups, denoted as $\sigma_B^2(j)$, is

$$\begin{aligned} \sigma_B^2(j) &= \omega_1(j) [M_1(j) - M_T]^2 + \omega_2(j) [M_2(j) - M_T]^2 \\ &= \frac{\omega_1(j) \omega_2(j) [M_1(j) - M_2(j)]^2}{\omega_1(j) + \omega_2(j)}. \end{aligned}$$

The optimal threshold $t = t^*$ is the value j for which $\sigma_B^2(j)$ is maximum.

The modification of Otsu's method presented in this work consists in dividing the image in 9 disjoint and 4 overlapped regions. A threshold value is computed for each region using the Otsu algorithm. Then, the values used are the average threshold

values of disjoint and overlapped regions.

4.2.4 Contour extraction and selection

Contour extraction from the binary image is performed using the OpenCV implementation of Suzuki and Abe's method [123]. This method generates a collection of external contours. Then, just the contours with four strong corners are selected. The corners are chosen by exploring changes in the derivative of a given contour. When this change in the derivative is big enough, the point is classified as strong corner point.

4.2.5 ID recognition

ID recognition is based on an unique code printed on each face of the octagon shape on the back part of the robot. After having the squares computed with the previous *contour extraction and selection* method, the next step is to determine which contours are related to valid robot codes. To perform this task, first a training step creates a database with square markers of $m \times n$ cells, and assigns an ID number to each marker. This ID number is computed by a binary string whose position number is 1 if the square is filled and 0 otherwise, as shown in Figure 4.8 for $m = 4$ and $n = 5$.

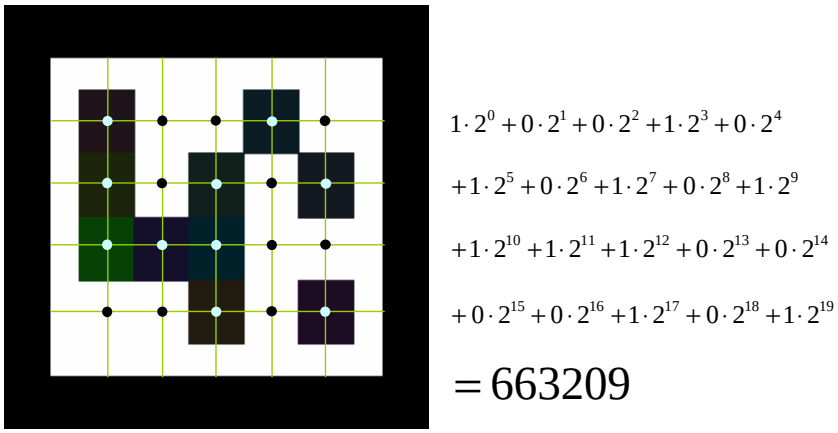


Figure 4.8: Fiducial marker ID computation.

Second, the identification step consists in recovering each face's ID from the perspective transformed image. To explain the method developed in this work, we introduce some basic concepts on Projective Geometry in Appendix A.

To compute the ID we use the *cross ratio*, an invariant to a *projective transformation* [39, 47]. The cross ratio is defined as

$$C_r\left(p_1, p_2; p_3, p_4\right) = \frac{d_{13} d_{24}}{d_{14} d_{23}},$$

where d_{ij} is the Euclidean distance between two points $p_i = \left[x_1^i, x_2^i, x_3^i\right]^T$ and $p_j = \left[x_1^j, x_2^j, x_3^j\right]^T$, given by

$$d_{ij} = \sqrt{\left(\frac{x_1^i}{x_3^i} - \frac{x_1^j}{x_3^j}\right)^2 + \left(\frac{x_2^i}{x_3^i} - \frac{x_2^j}{x_3^j}\right)^2}.$$

Therefore, if q_i, $i = 1, \ldots, 4$, denote the projective transformed points p_i, $i = 1, \ldots, 4$, then

$$C_r\left(p_1, p_2; p_3, p_4\right) = C_r\left(q_1, q_2; q_3, q_4\right).$$

Using the previous contour analysis, we obtain points q_{NW}, q_{SW}, q_{NE}, and q_{SE}, see Figure 4.9. The point q_c is given by the intersection of the segments $s_{NW,SE} = (q_{NW}, q_{SE})$ and $s_{SE,NE} = (q_{SW}, q_{NE})$, then

$$q_c = \left(q_{NW} \times q_{SE}\right) \times \left(q_{SW} \times q_{NE}\right). \tag{4.5}$$

Points v_1 and v_2 are called *vanishing points*, they could be at infinity and are given by

$$v_1 = \left(q_{SE} \times q_{SW}\right) \times \left(q_{NE} \times q_{NW}\right),$$
$$v_2 = \left(q_{NE} \times q_{SE}\right) \times \left(q_{NW} \times q_{SW}\right).$$

Transformed points q_W, q_E, q_N, and q_S are given by

$$q_W = \left(q_{NW} \times q_{SW}\right) \times \left(v_1 \times q_c\right), \tag{4.6}$$
$$q_E = \left(q_{NE} \times q_{SE}\right) \times \left(v_1 \times q_c\right), \tag{4.7}$$
$$q_N = \left(q_{NW} \times q_{NE}\right) \times \left(v_2 \times q_c\right), \tag{4.8}$$
$$q_S = \left(q_{SW} \times q_{SE}\right) \times \left(v_2 \times q_c\right). \tag{4.9}$$

Due to the invariability of the cross ratio, a point $p = (\alpha, \beta)$ belonging to the fiducial marker on the original space can be univocally mapped into the point $q = (\alpha_S, \beta_S)$ on the collineated space, see Figure 4.9. Let the following invariants be

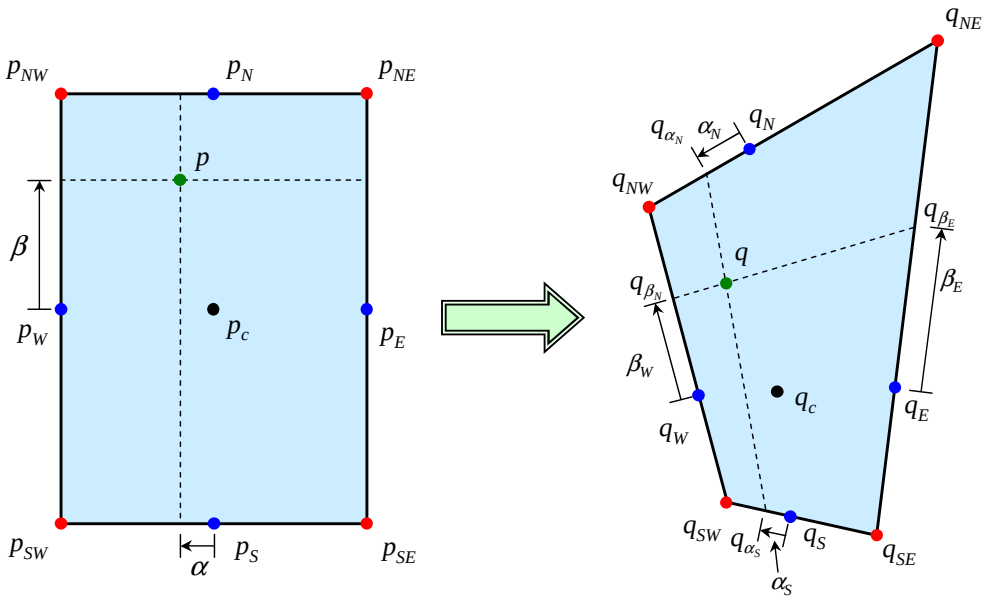


Figure 4.9: Projective transformation.

defined as

$$\rho_\alpha = \frac{1}{C_r\left(p_{NW}, \alpha; p_N, p_{NE}\right)} = \frac{1}{C_r\left(q_{NW}, \alpha; q_N, q_{NE}\right)} = \frac{2\left|\alpha\right|}{1+\left|\alpha\right|}, \tag{4.10}$$

$$\rho_\beta = \frac{1}{C_r\left(p_{NW}, \beta; p_W, p_{SW}\right)} = \frac{1}{C_r\left(q_{NW}, \beta; q_W, q_{SW}\right)} = \frac{2\left|\beta\right|}{1+\left|\beta\right|}, \tag{4.11}$$

where α and β are defined with the generation of the ID. Then, the displacement α_N, α_S, β_W, and β_E from q_N, q_S, q_W, and q_E, respectively, can be computed as

$$\alpha_N = \begin{cases} \frac{d_{NW,N}\rho_\alpha}{d_{NW,NE}-d_{N,NE}\rho_\alpha}, & \text{if } \alpha \geq 0, \\ \frac{d_{N,NE}\rho_\alpha}{d_{NW,NE}-d_{NW,N}\rho_\alpha}, & \text{if } \alpha < 0, \end{cases} \tag{4.12}$$

$$\alpha_S = \begin{cases} \frac{d_{SW,S}\rho_\alpha}{d_{SW,SE}-d_{S,SE}\rho_\alpha}, & \text{if } \alpha \geq 0, \\ \frac{d_{S,SE}\rho_\alpha}{d_{SW,SE}-d_{SW,S}\rho_\alpha}, & \text{if } \alpha < 0, \end{cases} \tag{4.13}$$

$$\beta_W = \begin{cases} \frac{d_{NW,W}\rho_\beta}{d_{NW,SW}-d_{W,SW}\rho_\beta}, & \text{if } \beta \geq 0, \\ \frac{d_{W,SW}\rho_\beta}{d_{NW,SW}-d_{NW,W}\rho_\beta}, & \text{if } \beta < 0, \end{cases} \tag{4.14}$$

$$\beta_E = \begin{cases} \frac{d_{NE,E}\rho_\beta}{d_{NE,SE}-d_{E,SE}\rho_\beta}, & \text{if } \beta \geq 0, \\ \frac{d_{E,SE}\rho_\beta}{d_{NE,SE}-d_{NE,E}\rho_\beta}, & \text{if } \beta < 0. \end{cases} \tag{4.15}$$

Finally, any position on the transformed space q is found as intersection of the seg-

ments $s_\alpha = (q_{\alpha_N}, q_{\alpha_S})$ and $s_\beta = (q_{\beta_w}, q_{\beta_E})$ as

$$q = (q_{\alpha_N} \times q_{\alpha_S}) \times (q_{\beta_W} \times q_{\beta_E}) \,. \tag{4.16}$$

In summary, the identification step transforms the center point of each square region using (4.16) to the collineated space and detect if the square is filled. Then, the fiducial marker is obtained using the on-line recovered ID and the database previously created.

To increase the robustness of the method, several modifications have been performed:

- The vertical size of the square is bigger than the horizontal to take into account lower vertical camera resolutions.
- In the classification of a region as filled, average or Gaussian filters are applied to avoid false readings due to noise.
- The IDs in the database have Hamming distances[2] greater or equal than $d_{\min}$ ($d_{\min} = 6$ in our application). Then, the marker is classified with a given ID in the database when its Hamming distance is less than $d_{\min}/2$.

4.3 Pose estimation

Pose estimation refers to the issue of obtaining relative position and orientation between two or more mobile robots using a camera. This Section reviews the camera models and resumes the key points of the pose estimation methods used in this work.

4.3.1 Camera model

The cameras used in this work are modeled with the well-know pinhole camera model. This model can be viewed as a box with a hole, or *aperture*, on one of its sides. This aperture allows light to enter into the box and reflect on the opposite side of the box, where light intensity can be measured, Figure 4.10 shows an schematic representation [47].

Let $p_i = [x_i, y_i, z_i, 1]^T$ and ${}^c p_i = [{}^c x_i, {}^c y_i, {}^c z_i, 1]^T$ denote the homogeneous coordinate vectors of a point p_i in the world coordinate and the camera systems, respectively.

[2] The *Hamming distance* is defined as the number of bits which differ between two binary strings.

Let the coordinate transformation $\mathcal{T}$ between points p_i and ${}^c p_i$ be given by a rotation $\mathcal{R}$ and a translation t, then

$$ {}^c p_i = \mathcal{T} p_i. \tag{4.17}$$

where $\mathcal{T} = [\mathcal{R}, t]$. The parameters of matrix $\mathcal{R}$ and vector t are known as *extrinsic parameters*.

Using the pinhole camera model, the homogeneous image coordinate vector $m_i = [u_i, v_i, 1]^T$ of point p_i is given by

$$ u_i = \alpha \frac{{}^c x_i}{{}^c z_i} - \alpha \cot\theta_s \frac{{}^c y_i}{{}^c z_i} + u_0, \quad v_i = -\frac{\beta}{\sin\theta_s} \frac{{}^c y_i}{{}^c z_i} + v_0, \tag{4.18}$$

where θ_s is the skew angle, $\alpha = kf$, $\beta = \ell f$ are magnification parameters, f is the focal length, and k, ℓ are scale parameters. Note that $k = \ell = 1$, and $\theta_s = 90°$ in an ideal camera. These parameters are known as *intrinsic parameters*. From (4.18) and (4.17), the projection of point p_i on the image is given by

$$ m_i = \frac{1}{{}^c z_i} \mathcal{M} p_i, \tag{4.19}$$

where $\mathcal{M} = \mathcal{K} \begin{bmatrix} \mathcal{R} & t \end{bmatrix}$ and $\mathcal{K}$ denotes the *intrinsic* matrix transformation given by

$$ \mathcal{K} = \begin{bmatrix} \alpha & \alpha \cot\theta_s & u_0 \\ 0 & -\frac{\beta}{\sin\theta_s} & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.20}$$

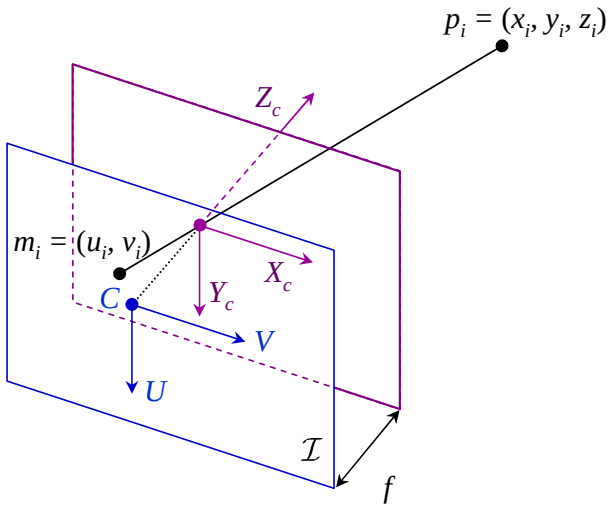


Figure 4.10: Pinhole camera model.

The *extrinsic* matrix transformation can be represented in various forms. The most common ones are Euler angles, quaternions, and exponential maps [91]. In this work, we have chosen the exponential map representation; because it only requires three parameters to describe a rotation $\mathcal{R}$ and other three parameters to describe a translation t.

Let $\omega = [\omega_x, \omega_y, \omega_z]^T$ be the rotation axis unity vector, and θ the rotation angle. Then the rotation matrix can be represented by

$$\mathcal{R}(\Omega) = \exp(\Omega\theta), \tag{4.21}$$

where Ω is the skew-symmetric matrix

$$\Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_z & 0 \end{bmatrix}. \tag{4.22}$$

Equation (4.21) can be evaluated using Rodrigues' formula [91]

$$\mathcal{R}(\Omega) = \exp(\Omega\theta) = I + \Omega\frac{\sin\theta}{\theta} + \Omega^2\frac{(1-\cos\theta)}{\theta^2}. \tag{4.23}$$

Therefore, because Euler's theorem says that any orientation $\mathcal{R} \in SO(3)$ is equivalent to a rotation about a fixed axis $\omega \in \mathbb{R}^3$ through an angle $\theta \in [0, 2\pi)$, a given a rotation matrix can be evaluated by an exponential map. Let $\mathcal{R}$ be such rotation matrix

$$\mathcal{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix},$$

then

$$\theta = \cos^{-1}\left(\frac{\text{trace}(\mathcal{R}) - 1}{2}\right).$$

If $\mathcal{R} \neq I$, then

$$\omega = \frac{1}{2\sin\theta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}.$$

If $\mathcal{R} = I$, then $\theta = 0$ and ω can be chosen arbitrarily.

4.3.2 OpenGL camera model

The main advantage of using a synthetic camera is the possibility of having its internal parameters perfectly known. OpenGL performs several transformations before drawing a point on the screen. This Section describes these transformations and obtains a synthetic intrinsic matrix transformation $\mathcal{K}_{OGL}$.

Let ${}^{c}p_i = [{}^{c}x_i, {}^{c}y_i, {}^{c}z_i, 1]^T$ denote the homogeneous coordinate vector of a point p_i in the camera system. Let the projection matrix $\mathcal{P}$ be defined as

$$\mathcal{P} = \begin{bmatrix} \frac{2d_n}{d_r - d_\ell} & 0 & \frac{d_r + d_\ell}{d_r - d_\ell} & 0 \\ 0 & \frac{2d_n}{d_t - d_b} & \frac{d_t + d_b}{d_t - d_b} & 0 \\ 0 & 0 & -\frac{d_f + d_n}{d_f - d_n} & -\frac{2d_f d_n}{d_f - d_n} \\ 0 & 0 & -1 & 0 \end{bmatrix},$$

where d_n, d_f, d_ℓ, d_r, d_t, and d_b are the parameters of the OpenGL camera frustum shown in Figure 4.11 [119]. In this work, we use $d_r = -d_\ell = \frac{w}{2}$, $d_t = -d_b = \frac{h}{2}$, where w and h are the image width and height, respectively.

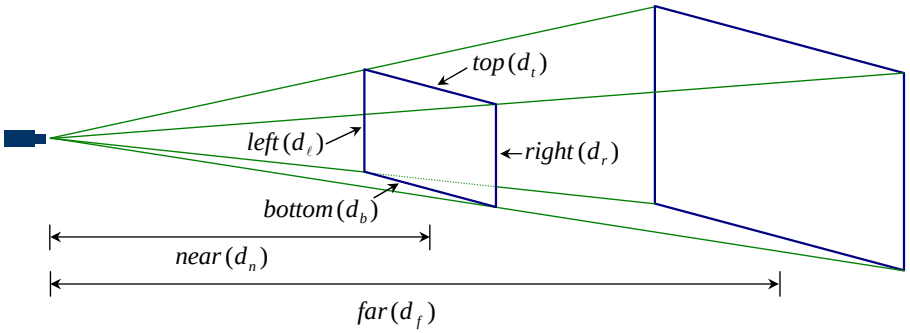


Figure 4.11: OpenGL camera frustum [119].

Applying the projection matrix to the point in eye coordinates, we obtain the point expressed in *clip coordinates*

$$\begin{bmatrix} {}^{e}x_i \\ {}^{e}y_i \\ {}^{e}z_i \\ {}^{e}w_i \end{bmatrix} = \mathcal{P} \begin{bmatrix} {}^{c}x_i \\ {}^{c}y_i \\ {}^{c}z_i \\ 1.0 \end{bmatrix}, \tag{4.24}$$

Points in clip coordinates are transformed to *normalized device coordinates* by

$$\begin{bmatrix} {}^{d}x_i \\ {}^{d}y_i \\ {}^{d}z_i \end{bmatrix} = \begin{bmatrix} 1/{}^{e}w_i & 0 & 0 \\ 0 & 1/{}^{e}w_i & 0 \\ 0 & 0 & 1/{}^{e}w_i \end{bmatrix} \begin{bmatrix} {}^{e}x_i \\ {}^{e}y_i \\ {}^{e}z_i \end{bmatrix}. \tag{4.25}$$

Finally, points in normalized device coordinates are transformed to *windows* or *image coordinates* using

$$\begin{bmatrix} u_i \\ v_i \\ z_i \end{bmatrix} = \frac{1}{2} \begin{bmatrix} w & 0 & 0 \\ 0 & h & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{d}x_i \\ {}^{d}y_i \\ {}^{d}z_i \end{bmatrix} + \frac{1}{2} \begin{bmatrix} w \\ h \\ 1 \end{bmatrix}. \tag{4.26}$$

Using (4.24), (4.25), (4.26) the OpenGL intrinsic matrix transformation $\mathcal{K}_{OGL}$ is given by

$$\mathcal{K}_{OGL} = \begin{bmatrix} \frac{d_n w}{d_r - d_\ell} & 0 & \frac{d_\ell w}{d_r - d_\ell} & 0 \\ 0 & \frac{d_n h}{d_t - d_b} & \frac{d_b h}{d_t - d_b} & 0 \\ 0 & 0 & -\frac{d_f}{d_f - d_n} & -\frac{d_f d_n}{d_f - d_n} \end{bmatrix}.$$

Therefore, by simple comparison with (4.20), we can obtain the intrinsic parameters

$$\theta = \frac{\pi}{2}, \quad \alpha = \frac{d_n w}{d_r - d_\ell}, \quad \beta = -\frac{d_n h}{d_t - d_b},$$
$$u_0 = \frac{d_\ell w}{d_r - d_\ell}, \quad v_0 = \frac{d_b h}{d_t - d_b}.$$

4.3.3 Pose estimation algorithms

This section summarizes and compares three model-based pose estimation algorithms from single images:

1. Pose from orthography and scaling with iteration (POSIT) algorithm.
2. Projection ray attraction (PRA) method.
3. Lowe's method.

4.3.3.1 Pose from orthography and scaling with iteration (POSIT) method

POSIT algorithm assumes that at least four feature points on the object can be detected and matched in the image. Moreover, as every model-based pose estimation

algorithm, assumes that the geometry of the object is known [31, 32, 95]. The method is an iterated version of POS (pose from orthography and scaling), which approximates a perspective projection with a scaled orthographic projection and then finds the transformation matrix of the object by solving a set of linear equations.

The algorithm computes point projections on a plane passing through the origin of the object, as shown in Figure 4.12. Let ${}^c p_i = [{}^c x_i, {}^c y_i, {}^c z_i]^T$, $i = 1, \ldots, N$, be points in the camera coordinate frame. Let $\mathcal{R}$ denote the rotation matrix, with

$$\mathcal{R} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{4.27}$$

and t the translation vector

$$t = [t_x, t_y, t_z]^T. \tag{4.28}$$

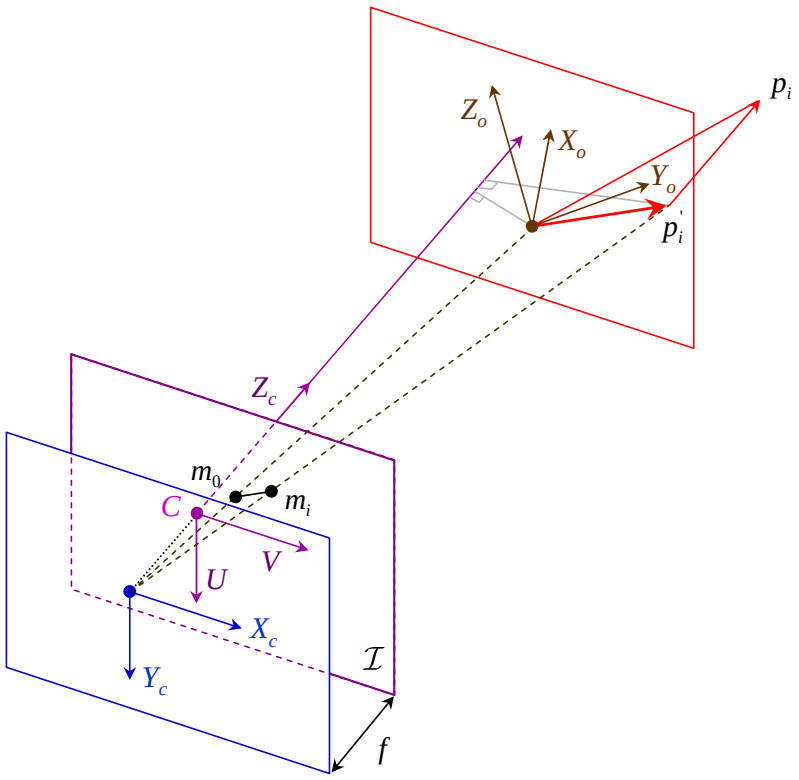


Figure 4.12: POSIT projections.

Let p_i, $i = 1, \ldots, N$, be the points ${}^c p_i$ in the object coordinate frame. Then

$$ {}^c x_i = r_1 p_i + t_x, \tag{4.29} $$

$$ {}^c y_i = r_2 p_i + t_y, \tag{4.30} $$

$$ {}^c z_i = r_3 p_i + t_z. \tag{4.31} $$

The perspective projection of ${}^c p_i$ with a simplified internal model of the camera, $\theta_s = \frac{\pi}{2}$, $\alpha = \beta = f$ in (4.20), is given by

$$ u_i = f \frac{r_1 p_i + t_x}{r_3 p_i + t_z}, $$

$$ v_i = -f \frac{r_2 p_i + t_y}{r_3 p_i + t_z}. $$

These projections can be written

$$ u_i = \frac{i_0 p_i + u_0}{\epsilon_i + 1}, \tag{4.32} $$

$$ v_i = \frac{j_0 p_i + v_0}{\epsilon_i + 1}, \tag{4.33} $$

with

$$ u_0 = f \frac{t_x}{t_z}, \quad v_0 = -f \frac{t_y}{t_z}, \tag{4.34} $$

$$ i_0 = \frac{f}{t_z} r_1, \quad j_0 = -\frac{f}{t_z} r_2, \tag{4.35} $$

$$ \epsilon_i = \frac{1}{t_z} r_3 p_i, \quad i = 1, \ldots, N. \tag{4.36} $$

Let $u_i^{'} = u_i \, (\epsilon_i + 1)$ and $v_i^{'} = v_i \, (\epsilon_i + 1)$ be the coordinates of the orthographic projection, then

$$ u_i^{'} = \frac{f}{t_z} \, {}^c x_i, $$

$$ v_i^{'} = -\frac{f}{t_z} \, {}^c y_i. $$

From (4.32), (4.33), we have

$$ i_0 p_i = u_i \, (\epsilon_i + 1) - u_0, \tag{4.37} $$

$$ j_0 p_i = v_i \, (\epsilon_i + 1) - v_0. \tag{4.38} $$

When the values ϵ_i are known, equations (4.37) and (4.38) form a linear system of equations in which the unknown are the coordinates i_0 and j_0. If the values given to ϵ_i are inexact, the solution of POS algorithm is just an approximation. However, after computing the unknowns r_1 and r_2, better approximation of ϵ_i can be computed using (4.36).

For each measured image point m_i and corresponding object point p_i, $i = 1, \ldots, N$, a set of equations (4.37)-(4.38) can be defined. Let these N equations be written in matrix form, then

$$A \begin{bmatrix} i_0^T \\ j_0^T \end{bmatrix} = V_\epsilon, \tag{4.39}$$

where

$$A = \begin{bmatrix} p_1^T & p_1^T \\ \vdots & \vdots \\ p_N^T & p_N^T \end{bmatrix},$$

and

$$V_\epsilon = \begin{bmatrix} u_1 (\epsilon_1 + 1) - u_0 & v_1 (\epsilon_1 + 1) - v_0 \\ \vdots & \vdots \\ u_N (\epsilon_N + 1) - u_0 & v_N (\epsilon_N + 1) - v_0 \end{bmatrix}.$$

Therefore, the solution for i_0 and j_0 is

$$\begin{bmatrix} i_0^T \\ j_0^T \end{bmatrix} = \left(A^T A\right)^{-1} A^T V_\epsilon. \tag{4.40}$$

Algorithm 4 resumes the implemented POSIT algorithm.

Algorithm 4 POSIT.

1: Initialize ϵ_i to some positive value.
2: **while** $\|\epsilon_i - \epsilon_{i-1}\| > tol$ **do**
3: Compute i_0 and j_0 using (4.40).
4: $i_0' = [i_0(0), i_0(1), i_0(2)]$, $j_0' = [j_0(0), j_0(1), j_0(2)]$.
5: $t_z = f \Big/ \sqrt{\|i_0'\| \, \|j_0'\|}$.
6: $r_1 = \frac{i_0'}{\|i_0'\|}$, $r_2 = \frac{j_0'}{\|j_0'\|}$, $r_3 = r_1 \times r_2$.
7: $t_x = \frac{t_z}{f} i_0(3)$, $t_y = -\frac{t_z}{f} j_0(3)$.
8: **for** $i = 1$ **to** N **do**
9: $\epsilon_i = \frac{1}{t_z} r_3 p_i$.
10: **end for**
11: **end while**

4.3.3.2 Projection ray attraction (PRA) method

Projection ray attraction (PRA) is a method presented in [98] to recover motion information from two-dimensional projections of a given three-dimensional point set. It is divided in two linear estimation parts: *depth approximation* and *pose computation*. In the depth approximation part, feature points in the three-dimensional space are estimated. In the pose computation part, rotation and translation parameters of the object are estimated using singular value decomposition (SVD). Both part are iteratively executed until the result is stable.

The main idea is to state pose estimation as a nonlinear optimization problem using a simplified camera model: Given a set of image points m_i, $i = 1, \ldots, N$, and three-dimensional feature points p_i, referred to the object coordinate frame, or ${}^c p_i$ referred to the camera coordinate frame, the pose estimation problem can be solved by minimizing the functional

$$\min_{\mathcal{R},t,d_i} \mathcal{F}(\mathcal{R}, t, d_i) = \sum_{i=1}^{N} \|d_i v_i - (\mathcal{R} p_i + t)\|^2, \tag{4.41}$$

where v_i, $i = 1, \ldots, N$, are normalized three-dimensional point representations of the image point m_i,

$$v_i = \frac{M_i}{\|M_i\|},$$

with

$$M_i = \begin{bmatrix} u_i \\ v_i \\ f \end{bmatrix} = f \begin{bmatrix} {}^c x_i \,/\, {}^c z_i \\ -\,{}^c y_i \,/\, {}^c z_i \\ 1 \end{bmatrix},$$

and d_i is the depth of the object, with

$${}^c p_i = [\,{}^c x_i,\, {}^c y_i,\, {}^c z_i]^T = d_i v_i.$$

Therefore, the depth approximation part of the algorithm in [98] consists in approximate the depth by

$$d_i = v_i^T \, {}^c p_i. \tag{4.42}$$

The minimization problem (4.41) is solved by SVD. Let H be defined as

$$H = \frac{1}{N} \sum_{i=1}^{N} \left\| (M_i - \mu_M)\,(\,{}^c p_i - \mu_p)^T \right\|,$$

where

$$\mu_p = \frac{1}{N}\sum_{i=1}^{N} {}^{c}p_i, \quad \sigma_p^2 = \frac{1}{N}\sum_{i=1}^{N} \| {}^{c}p_i - \mu_p \|^2 ,$$
$$\mu_M = \frac{1}{N}\sum_{i=1}^{N} M_i, \quad \text{and } \sigma_M^2 = \frac{1}{N}\sum_{i=1}^{N} \| M_i - \mu_M \|^2 .$$

Let the SVD of H be given by

$$H = UWV^T.$$

Then,

$$\mathcal{R} = USV^T,$$
$$t = \mu_M - c\mathcal{R}\mu_p,$$

where

$$c = \frac{1}{\sigma_p^2}\,\text{trace}\,(WS)\,.$$

and S is given by

$$S = \begin{cases} I, & \text{if } \det(H) \geq 0, \\ \text{diag}\,(1,1,-1)\,, & \text{if } \det(H) < 0, \end{cases}$$

when $\text{rank}\,(H) = 3$, or

$$S = \begin{cases} I, & \text{if } \det(U)\det(V) = 1, \\ \text{diag}\,(1,1,-1)\,, & \text{if } \det(U)\det(V) = -1, \end{cases}$$

when $\text{rank}\,(H) \neq 3$. See [98] for more details.

4.3.3.3 Lowe's method

This method uses a Newton-type optimization algorithm to estimate the 6 parameter of the external matrix: three for the translation vector, and three for the rotation matrix parameterized by an exponential map. The method has the risk of converging to a false local minimum when the initial values of the unknown parameters are far from the real ones. To avoid this problem, the optimization problem is augmented by a stabilization method that incorporates a prior model of parameter uncertainty range and standard deviation estimation of each image [15, 82].

As in the previous approaches, we assume that the transformation between two frames of a point ${}^{c}p_i$ in the camera coordinate frame is a rotation $\mathcal{R}$ and a translation t. The estimated image projections of these points are given by

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \frac{1}{{}^{c}z_i}\mathcal{K}\,{}^{c}p_i = \frac{1}{r_3 p_i + t_z}\begin{bmatrix} k_{11} & k_{12} \\ 0 & k_{22} \end{bmatrix}\begin{bmatrix} r_1 p_i + t_x \\ r_2 p_i + t_y \end{bmatrix}, \tag{4.43}$$

where p_i and ${}^{c}p_i$ are related by equations (4.17), (4.29) to (4.31).

Let the optimization problem be defined as

$$\vartheta = \arg\min_{\vartheta} \sum_{i=1}^{N} \left\| m_i - \hat{m}_i\left(\hat{\vartheta}\right)\right\|^2, \tag{4.44}$$

where ϑ is the parameter vector

$$\vartheta = \left[\omega_x, \omega_y, \omega_z, t_x, t_y, t_z\right]^T,$$

m_i, $i = 1, \ldots, N$, are measured image points, and $\hat{m}_i$, $i = 1, \ldots, N$, are estimated image points

$$\hat{m}_i = \frac{f}{\hat{r}_3\,{}^{c}q_i + \hat{t}_z}\begin{bmatrix} \hat{r}_1\,{}^{c}q_i + \hat{t}_x \\ -\hat{r}_2\,{}^{c}q_i - \hat{t}_y \end{bmatrix}.$$

Let the error $e_N(\vartheta)$ be defined as

$$e_N(\vartheta) = \begin{bmatrix} u_1 - \hat{u}_1 \\ v_1 - \hat{v}_1 \\ \vdots \\ u_N - \hat{u}_N \\ v_N - \hat{v}_N \end{bmatrix} \in \mathbb{R}^{2N}. \tag{4.45}$$

In general, Newton-type methods compute a vector of corrections $\Delta\vartheta$ based on the local linearity of (4.44) and error (4.45). Then

$$\vartheta_{\tau+1} = \vartheta_\tau + \Delta\vartheta, \tag{4.46}$$

where τ denotes the iteration index. In particular, in the Levenberg-Marquardt algorithm, the correction $\Delta\vartheta$ is computed as

$$\Delta\vartheta = -\left(J^T J + \lambda I\right)^{-1} J^T e_N, \tag{4.47}$$

where $\lambda > 0$ is an adaptive parameter, $J \in \mathbb{R}^{2N\times 6}$ is the Jacobian matrix, given by

$$J = \begin{bmatrix} J_1 \\ \vdots \\ J_N \end{bmatrix}, \tag{4.48}$$

with

$$J_i = \begin{bmatrix} \frac{\partial u_i}{\partial \omega_x} & \frac{\partial u_i}{\partial \omega_y} & \frac{\partial u_i}{\partial \omega_z} & \frac{\partial u_i}{\partial t_x} & \frac{\partial u_i}{\partial t_y} & \frac{\partial u_i}{\partial t_z} \\ \frac{\partial v_i}{\partial \omega_x} & \frac{\partial v_i}{\partial \omega_y} & \frac{\partial v_i}{\partial \omega_z} & \frac{\partial v_i}{\partial t_x} & \frac{\partial v_i}{\partial t_y} & \frac{\partial v_i}{\partial t_z} \end{bmatrix}, \quad i = 1, \ldots, N.$$

To compute the derivatives of the estimated projections with respect to t_x, t_y, t_z, ω_x, ω_y, and ω_z, we need to obtain the derivatives of the rotation matrix $\mathcal{R}$ with respect to these parameters. Using (4.23) and (4.22), we have

$$\mathcal{R}(\Omega) = I + c_1(\theta)\,\Omega + c_2(\theta)\,\Omega^2,$$

with $c_1(\theta) = \frac{\sin\theta}{\theta}$, $c_2(\theta) = \frac{1-\cos\theta}{\theta^2}$, and

$$\Omega^2 = \begin{bmatrix} -\left(\omega_y^2 + \omega_z^2\right) & \omega_x\omega_y & \omega_x\omega_z \\ \omega_x\omega_y & -\left(\omega_x^2 + \omega_z^2\right) & \omega_y\omega_z \\ \omega_x\omega_z & \omega_y\omega_z & -\left(\omega_x^2 + \omega_y^2\right) \end{bmatrix}.$$

Then

$$r_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} + c_1(\theta)\,\Omega_1 + c_2(\theta)\,\Omega_1^2, \tag{4.49}$$

$$r_2 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} + c_1(\theta)\,\Omega_2 + c_2(\theta)\,\Omega_2^2, \tag{4.50}$$

$$r_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} + c_1(\theta)\,\Omega_3 + c_2(\theta)\,\Omega_3^2, \tag{4.51}$$

with

$$\Omega_1 = \begin{bmatrix} 0, & -\omega_z, & \omega_y \end{bmatrix}, \Omega_2 = \begin{bmatrix} \omega_z, & 0, & -\omega_x \end{bmatrix}, \Omega_3 = \begin{bmatrix} -\omega_y, & \omega_z, & 0 \end{bmatrix},$$

$$\Omega_1^2 = \begin{bmatrix} -\left(\omega_y^2 + \omega_z^2\right), & \omega_x\omega_y, & \omega_x\omega_z, \end{bmatrix},$$

$$\Omega_2^2 = \begin{bmatrix} \omega_x\omega_y, & -\left(\omega_x^2 + \omega_z^2\right), & \omega_y\omega_z \end{bmatrix},$$

$$\Omega_3^2 = \begin{bmatrix} \omega_x\omega_z, & \omega_y\omega_z, & -\left(\omega_x^2 + \omega_y^2\right) \end{bmatrix}.$$

Using (4.49) to (4.51), we have

$$\begin{aligned}
\frac{\partial r_1}{\partial \omega_x} &= c_2\left[0,\omega_y,\omega_z\right]+c_3\omega_x\left[0,\text{-}\omega_z,\omega_y\right]+c_4\omega_x\left[\text{-}\omega_y^2\text{-}\omega_z^2,\omega_x\omega_y,\omega_x\omega_z\right],\\
\frac{\partial r_1}{\partial \omega_y} &= c_1\left[0,0,1\right]+c_2\left[\text{-}2\omega_y,\omega_x,0\right]+c_3\omega_y\left[0,\text{-}\omega_z,\omega_y\right]+c_4\omega_y\left[\text{-}\omega_y^2\text{-}\omega_z^2,\omega_x\omega_y,\omega_x\omega_z\right],\\
\frac{\partial r_1}{\partial \omega_z} &= c_1\left[0,\text{-}1,0\right]+c_2\left[\text{-}2\omega_z,0,\omega_x\right]+c_3\omega_z\left[0,\text{-}\omega_z,\omega_y\right]+c_4\omega_z\left[\text{-}\omega_y^2\text{-}\omega_z^2,\omega_x\omega_y,\omega_x\omega_z\right],
\end{aligned}$$

$$\begin{aligned}
\frac{\partial r_2}{\partial \omega_x} &= c_1\left[0,0,\text{-}1\right]+c_2\left[\omega_y,\text{-}2\omega_x,0\right]+c_3\omega_x\left[\omega_z,0,\text{-}\omega_x\right]+c_4\omega_x\left[\omega_x\omega_y,\text{-}\omega_x^2\text{-}\omega_z^2,\omega_y\omega_z\right],\\
\frac{\partial r_2}{\partial \omega_y} &= c_2\left[\omega_x,0,\omega_z\right]+c_3\omega_y\left[\omega_z,0,\text{-}\omega_x\right]+c_4\omega_y\left[\omega_x\omega_y,\text{-}\omega_x^2\text{-}\omega_z^2,\omega_y\omega_z\right],\\
\frac{\partial r_2}{\partial \omega_z} &= c_1\left[1,0,0\right]+c_2\left[0,\text{-}2\omega_z,\omega_y\right]+c_3\omega_z\left[\omega_z,0,\text{-}\omega_x\right]+c_4\omega_z\left[\omega_x\omega_y,-\omega_x^2\text{-}\omega_z^2,\omega_y\omega_z\right],
\end{aligned}$$

$$\begin{aligned}
\frac{\partial r_3}{\partial \omega_x} &= c_1\left[0,1,0\right]+c_2\left[\omega_z,0,\text{-}2\omega_x\right]+c_3\omega_x\left[\text{-}\omega_y,\omega_z,0\right]+c_4\omega_x\left[\omega_x\omega_z,\omega_y\omega_z,\text{-}\omega_x^2\text{-}\omega_y^2\right],\\
\frac{\partial r_3}{\partial \omega_y} &= c_1\left[\text{-}1,0,0\right]+c_2\left[0,\omega_z,\text{-}2\omega_y\right]+c_3\omega_y\left[\text{-}\omega_y,\omega_z,0\right]+c_4\omega_y\left[\omega_x\omega_z,\omega_y\omega_z,\text{-}\omega_x^2\text{-}\omega_y^2\right],\\
\frac{\partial r_3}{\partial \omega_z} &= c_2\left[\omega_x,\omega_y,0\right]+c_3\omega_z\left[-\omega_y,\omega_z,0\right]+c_4\omega_z\left[\omega_x\omega_z,\omega_y\omega_z,-\omega_x^2\text{-}\omega_y^2\right],
\end{aligned}$$

with $c_3\left(\theta\right)=\frac{\theta\cos\theta-\sin\theta}{\theta^3}$, $c_4\left(\theta\right)=\frac{\theta\sin\theta-2(1-\cos\theta)}{\theta^4}$.

Finally, using (4.43), we have

$$\begin{aligned}
\begin{bmatrix}\frac{\partial u_i}{\partial \omega_x}\\ \frac{\partial v_i}{\partial \omega_x}\end{bmatrix} &= \frac{1}{{}^cz_i}\left(\begin{bmatrix}k_{11} & k_{12}\\ 0 & k_{22}\end{bmatrix}\begin{bmatrix}\frac{\partial r_1}{\partial \omega_x}\\ \frac{\partial r_2}{\partial \omega_x}\end{bmatrix}-\begin{bmatrix}u_i\\ v_i\end{bmatrix}\frac{\partial r_3}{\partial \omega_x}\right)p_i,\\
\begin{bmatrix}\frac{\partial u_i}{\partial \omega_y}\\ \frac{\partial v_i}{\partial \omega_y}\end{bmatrix} &= \frac{1}{{}^cz_i}\left(\begin{bmatrix}k_{11} & k_{12}\\ 0 & k_{22}\end{bmatrix}\begin{bmatrix}\frac{\partial r_1}{\partial \omega_y}\\ \frac{\partial r_2}{\partial \omega_y}\end{bmatrix}-\begin{bmatrix}u_i\\ v_i\end{bmatrix}\frac{\partial r_3}{\partial \omega_y}\right)p_i,\\
\begin{bmatrix}\frac{\partial u_i}{\partial \omega_y}\\ \frac{\partial v_i}{\partial \omega_y}\end{bmatrix} &= \frac{1}{{}^cz_i}\left(\begin{bmatrix}k_{11} & k_{12}\\ 0 & k_{22}\end{bmatrix}\begin{bmatrix}\frac{\partial r_1}{\partial \omega_z}\\ \frac{\partial r_2}{\partial \omega_z}\end{bmatrix}-\begin{bmatrix}u_i\\ v_i\end{bmatrix}\frac{\partial r_3}{\partial \omega_z}\right)p_i,
\end{aligned}$$

$$\begin{bmatrix}\frac{\partial u_i}{\partial t_x}\\ \frac{\partial v_i}{\partial t_x}\end{bmatrix}=\frac{1}{{}^cz_i}\begin{bmatrix}k_{11}\\ 0\end{bmatrix},\ \begin{bmatrix}\frac{\partial u_i}{\partial t_y}\\ \frac{\partial v_i}{\partial t_y}\end{bmatrix}=\frac{1}{{}^cz_i}\begin{bmatrix}k_{12}\\ k_{22}\end{bmatrix},\ \begin{bmatrix}\frac{\partial u_i}{\partial t_z}\\ \frac{\partial v_i}{\partial t_z}\end{bmatrix}=-\frac{1}{{}^cz_i}\begin{bmatrix}u_i\\ v_i\end{bmatrix}.$$

4.4 Nonlinear filtering

In this Section, we resume the leader-follower mathematical model and an estimation algorithm using a dual unscented Kalman filter. The filter is used to smooth measured variables and to recover leader's velocities.

4.4.1 Mathematical model of the robots

In general, leader-follower controllers require reliable estimates of the leader robot's speed v_1 and angular velocity ω_1 by the follower robot. These quantities can be estimated using an unscented Kalman filter with the appropriate dynamic model and measurements from the camera range ℓ_{12}, bearing α_{12}, and relative orientation θ_{12}. For this reason, let us consider a leader-follower system composed of two unicycle-type vehicles moving on the plane, with the kinematic model of the ith robot, $i = 1, 2$, given by

$$\dot{q}_i(t) = \begin{bmatrix} \dot{x}_i(t) \\ \dot{y}_i(t) \\ \dot{\theta}_i(t) \end{bmatrix} = \begin{bmatrix} \cos\theta_i(t) & 0 \\ \sin\theta_i(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i(t) \\ \omega_i(t) \end{bmatrix}. \tag{4.52}$$

where $q_i(t) = [x_i(t), y_i(t), \theta_i(t)]^T \in SE(2)$ is the configuration vector, $u_i(t) = [v_i(t), \omega_i(t)]^T \in \mathcal{U}_i \subseteq \mathbb{R}^2$ is the velocity vector, and $\mathcal{U}_i$ is a compact set of admissible inputs.

Let the Euclidean distance $\ell_{12}(t) \in \mathbb{R}_{\geq 0}$ and angles $\alpha_{12}(t)$, $\beta_{12}(t) \in (-\pi, \pi]$ between robots 1 (leader) and 2 (follower) be defined as

$$\ell_{12}(t) = \sqrt{(x_1(t) - x_2^c(t))^2 + (y_1(t) - y_2^c(t))^2}, \tag{4.53}$$

$$\alpha_{12}(t) = \zeta_{12}(t) - \theta_1(t), \tag{4.54}$$

$$\theta_{12}(t) = \theta_1(t) - \theta_2(t), \tag{4.55}$$

where $\zeta_{12}(t) = \text{atan2}(y_1 - y_2^c, x_1 - x_2^c)$, and $x_2^c(t) = x_2 + d\cos\theta_2$, $y_2^c(t) = y_2 + d\sin\theta_2$ are the coordinates of the camera, as shown in Figure 2.2.

Then, the leader-follower system can be described in polar form by the set of

equations

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{y}_1(t) \\ \dot{\theta}_1(t) \\ \dot{\ell}_{12}(t) \\ \dot{\alpha}_{12}(t) \\ \dot{\theta}_{12}(t) \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & 0 \\ \sin\theta_1 & 0 \\ 0 & 1 \\ \cos\alpha_{12} & 0 \\ -\frac{\sin\alpha_{12}}{\ell_{12}} & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1(t) \\ \omega_1(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\cos\beta_{12} & -d\sin\beta_{12} \\ \frac{\sin\beta_{12}}{\ell_{12}} & -\frac{d\cos\beta_{12}}{\ell_{12}} \\ 0 & -1 \end{bmatrix} \begin{bmatrix} v_2(t) \\ \omega_2(t) \end{bmatrix}, \tag{4.56}$$

with $\beta_{12} = \alpha_{12} + \theta_{12}$.

Equation (4.56) can be written in discrete form as

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) + \mathbf{v}_k, \\ \mathbf{y}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) + \mathbf{n}_k, \end{cases} \tag{4.57}$$

where $\mathbf{x}_k = [x_1(k), y_1(k), \theta_1(k), \ell_{12}(k), \alpha_{12}(k), \beta_{12}(k)]^T$, $\mathbf{u}_k = [v_2(k), w_2(k)]^T$, $\mathbf{w}_k = [v_1(k), w_1(k)]^T$, random variables $\mathbf{v}_k$ and $\mathbf{n}_k$ are process and measurement noise, respectively, and $\mathbf{y}_k = [\ell_{12}(k), \alpha_{12}(k), \beta_{12}(k)]^T$ is obtained using measurements from the vision system (x_c, y_c, z_c), θ_m, and the value of the camera pan θ_p. Therefore

$$\ell_{12} = \sqrt{x_c^2 + y_c^2 + z_c^2},\ \theta_t = \arctan\left(\frac{-x_c}{z_c}\right),$$
$$\alpha_{12} = \theta_t - \theta_m,\ \theta_{12} = \theta_p + \theta_m.$$

4.4.2 Nonlinear observability

In this Section, we present some observability results for system (4.56). In this particular case, the proof of observability is direct, but it will serve the purpose of being introductory for more complex systems analyzed in Chapter 2. The supporting theory for the proofs given in this Section is resumed in Appendix C.

Let model (4.56) be written in affine in control form as

$$\Sigma: \begin{cases} \dot{\mathbf{x}}(t) = \sum_{i=1}^{4} \mathbf{f}_i(\mathbf{x}(t)) u_i(t), \\ \mathbf{y}(t) = \mathbf{h}(\mathbf{x}), \end{cases} \tag{4.58}$$

with $\mathbf{x} = [x_1, y_1, \theta_1, \ell_{12}, \alpha_{12}, \beta_{12}]^T \in \mathcal{M} \subseteq \mathbb{R}^2 \times \mathbb{R}_{\geq 0} \times \mathbb{S}^3$, $\mathcal{M}$ is an appropriate manifold, $\mathbf{y} = [\ell_{12}, \alpha_{12}, \beta_{12}]^T \in \mathbb{R}_{\geq 0} \times \mathbb{S}^2$,

$$
\begin{aligned}
\mathbf{f}_1(\mathbf{x}) &= \left[\cos\theta_1,\, \sin\theta_1,\, 0,\, \cos\alpha_{12},\, -\frac{\sin\alpha_{12}}{\ell_{12}},\, 0\right]^T, \\
\mathbf{f}_2(\mathbf{x}) &= [0,\, 0,\, 1,\, 0,\, -1,\, 1]^T, \\
\mathbf{f}_3(\mathbf{x}) &= \left[0,\, 0,\, 0,\, -\cos\beta_{12},\, \frac{\sin\beta_{12}}{\ell_{12}},\, 0\right]^T, \\
\mathbf{f}_4(\mathbf{x}) &= \left[0,\, 0,\, 0,\, -d\sin\beta_{12},\, -\frac{d\cos\beta_{12}}{\ell_{12}},\, -1\right]^T, \\
\mathbf{h}(\mathbf{x}) &= [\ell_{12},\, \alpha_{12},\, \theta_{12}]^T, \\
u_1(t) &= v_1(t),\, u_2(t) = \omega_1(t),\, u_3(t) = v_2(t),\, u_4(t) = \omega_2(t),
\end{aligned}
$$

and we have omitted the dependency with respect to time for brevity.

The observation space $\mathcal{O}$ is the span over $\mathbb{R}$ of

$$
\left\{1,\, \ell_{12},\, \alpha_{12},\, \theta_{12},\, \frac{\sin\alpha_{12}}{\ell_{12}},\, \frac{\sin(\alpha_{12}+\theta_{12})}{\ell_{12}},\, \frac{d\cos(\alpha_{12}+\theta_{12})}{\ell_{12}},\, \cos\alpha_{12},\, \cos(\alpha_{12}+\theta_{12}),\, d\sin(\alpha_{12}+\theta_{12})\right\}.
$$

Then, the observability codistribution $d\mathcal{O}(\mathbf{x})$ is given by

$$
d\mathcal{O}(\mathbf{x}) = \operatorname{span}\{o_1,\, o_2,\, o_3,\, o_4,\, o_5,\, o_6,\, o_7\},
$$

with

$$
\begin{aligned}
o_1 &= [0,0,0,1,0,0], \\
o_2 &= [0,0,0,0,\sin\alpha_{12},0], \\
o_3 &= [0,0,0,0,\sin(\alpha_{12}+\theta_{12}),\sin(\alpha_{12}+\theta_{12})], \\
o_4 &= [0,0,0,0,1,0], \\
o_5 &= \left[0,0,0,\frac{\sin\alpha_{12}}{\ell_{12}^2},\frac{\cos\alpha_{12}}{\ell_{12}},0\right], \\
o_6 &= \left[0,0,0,\frac{\cos(\alpha_{12}+\theta_{12})}{\ell_{12}^2},\frac{\sin(\alpha_{12}+\theta_{12})}{\ell_{12}},\frac{\sin(\alpha_{12}+\theta_{12})}{\ell_{12}}\right], \text{ and} \\
o_7 &= [0,0,0,0,0,1].
\end{aligned}
$$

It is clear that x_1, y_1, and θ_1 are unobservable. However, $\dim(d\mathcal{O}(\mathbf{x})) = 3$ then, by the observability rank condition theorem, see Appendix C, $\ell_{12}, \alpha_{12}, \theta_{12}$ are globally observable.

4.4.3 The dual unscented Kalman filter (DUKF)

The unscented Kalman filter (UKF) uses a deterministic sampling approach, the unscented transform (UT), to capture the mean and covariance estimates with a minimal set of sample points. When the sampling points are propagated through the true non-linear system, the UKF can capture the posterior mean and covariance accurately up to the 3rd order for Taylor series expansion of the Gaussian error distribution with same order of computational complexity as the EKF that can achieve only up to first-order accuracy. It should also be noted that the of the UKF is the same order as that of EKF [65, 66, 140, 141].

In this Chapter, we use the dual unscented Kalman filter (DUKF) [141] described in Chapter 3. This filter runs two parallel filters for state and parameter estimation. The state estimation filter considers the parameter known for state updating, whereas its parameter estimation counterpart considers states known for parameter updating.

In the leader-follower model (4.57) the states are given by the vector $\mathbf{x}_k$ and the parameter by vector $\mathbf{w}_k$. That is, leader's velocities are considered known for state estimation and then updated by the parameter estimation filter, as it is schematically represented in Figure 4.13. Note that there is no convergence guarantee with the DUKF.

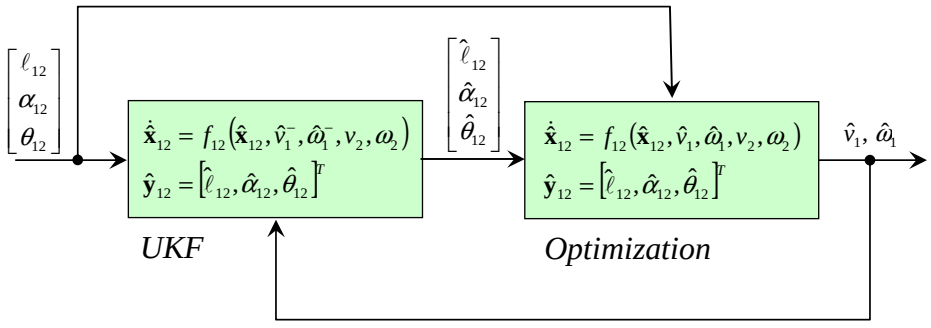


Figure 4.13: Dual estimation problem.

4.5 Simulation results

Figures 4.14-4.16 show identification results for three robots. Figure 4.14 shows faces detected when the robots are in configurations $q_1 = [1.5, .25, 0.0]^T$ and $q_2 = [2.0, -0.45, 0.0]^T$ relative to the follower configuration. Figure 4.15 and 4.16 show same results with $q_1 = [2.5, -0.5, 0.0]^T$, $q_2 = [3.5, 0.5, 0.0]^T$ and $q_1 = [2.5, -0.5, 0.0]^T$, $q_2 = \left[3.5, 0.5, \frac{\pi}{2}\right]^T$, respectively. The algorithm performs very well, even with small surfaces, changes in illumination and increasing distance.

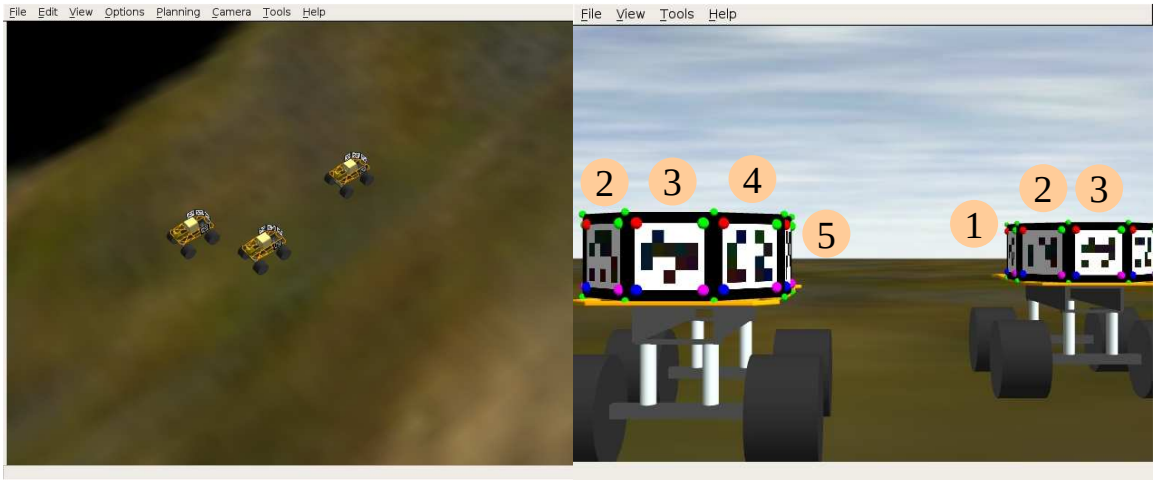


Figure 4.14: Identifications: Robot 1 (left) IDs 2, 3, 4, 5; robot 2 (right) IDs 1, 2, 3.

Figure 4.15: Identifications: Robot 1 (right) IDs 2, 3, 4; robot 2 (left) IDs 2, 3, 4.

Figures 4.17 and 4.18 show estimated pose for pure translation and pure rotation motions using Lowe's algorithm.

Figure 4.16: Identifications: Robot 1 (right) IDs 2, 3, 4; robot 2 (left) IDs 1, 2.

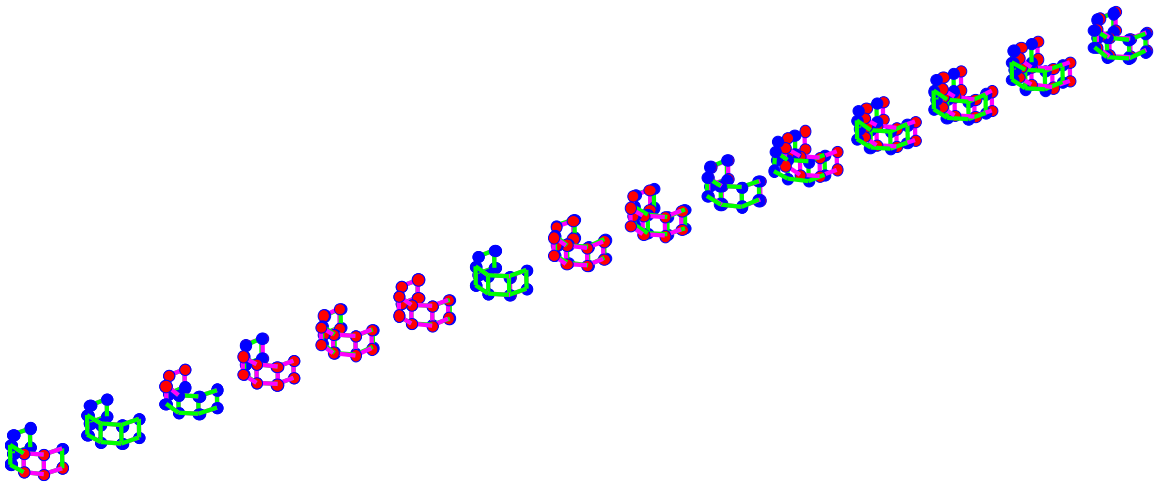


Figure 4.17: Pure translation using Lowe's method.

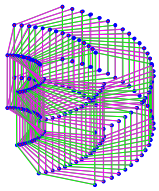


Figure 4.18: Pure rotation using Lowe's method.

Figures 4.19 to 4.25 show simulations using different pose estimation methods for pure translation and pure rotation motions. We compared the final error and the number of iterations to achieve convergence for each algorithm. Figure 4.19 shows the error and number of iteration of PRA algorithm without using information from previous iterations. Figure 4.20 shows same results using previous information as starting configuration. Figure 4.21 shows the error and number of iterations for pure rotations using a priori information. As can be seen, the number of iteration is considerably reduced.

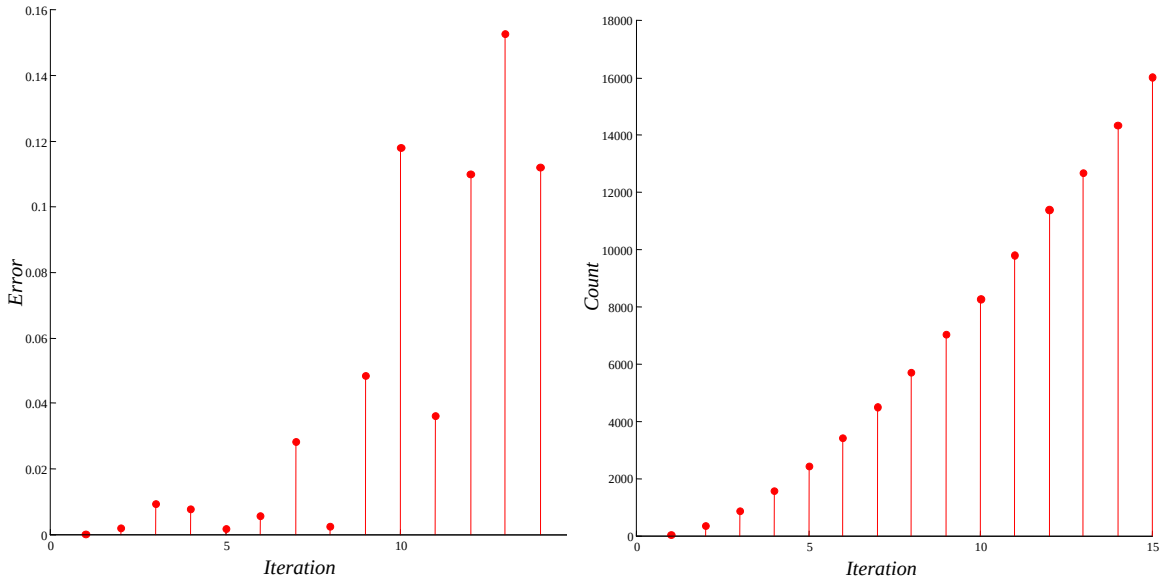


Figure 4.19: Pure translation. Error and number of iterations using PRA method.

Figures 4.22 and 4.23 show the results for pure translation and rotation motions using Lowe's algorithm. It should be noted that the number of iterations is lower and the final error is smaller than in the previous case.

Figures 4.24 and 4.25 show the results for pure translation and rotation motions using POSIT algorithm with a priori information. The number of iterations is even lower than in the previous case, but the final error is bigger. A drawback of POSIT algorithm is that, due to the orthographic approximation, it needs the value of ${}^{c}z_i$ bigger than $|{}^{c}x_i|$ and $|{}^{c}y_i|$ to achieve convergence.

Figures 4.26 to 4.28 show pose estimation results with Lowe's method initialized with POSIT algorithm. Red skeletons show the estimated positions. The distance and angle are slightly biased with accuracies around 3% and 5%, respectively. As can be seen, the estimated pose is very accurate for practical purposes. Figure 4.29 shows the effect of noise on the image. As can be seen, the effect reduces the achievable

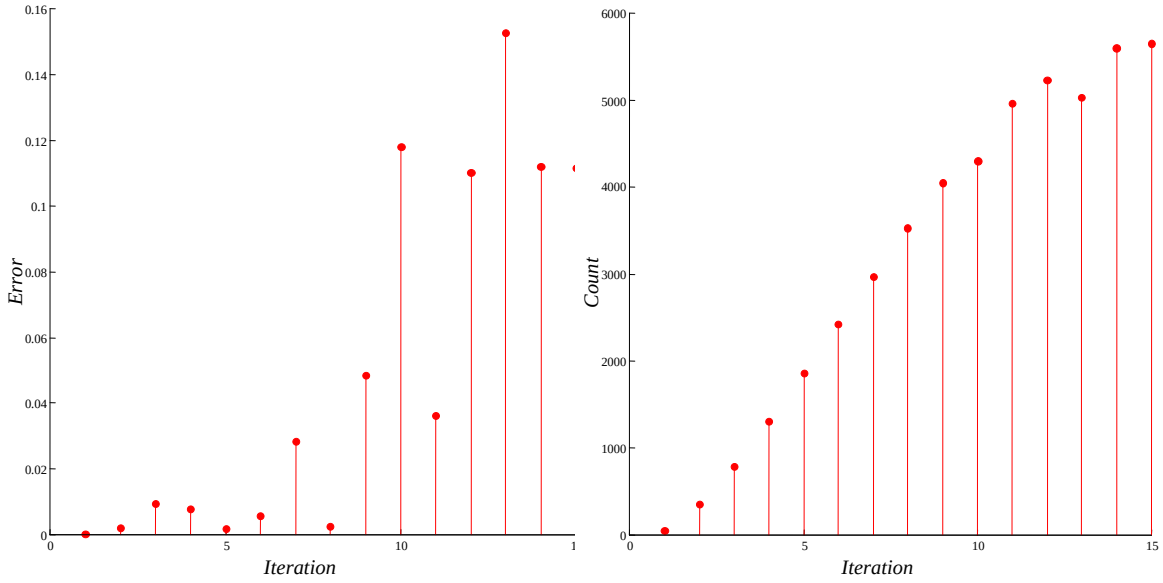


Figure 4.20: Pure translation. Error and number of iterations using PRA method and a priori information.

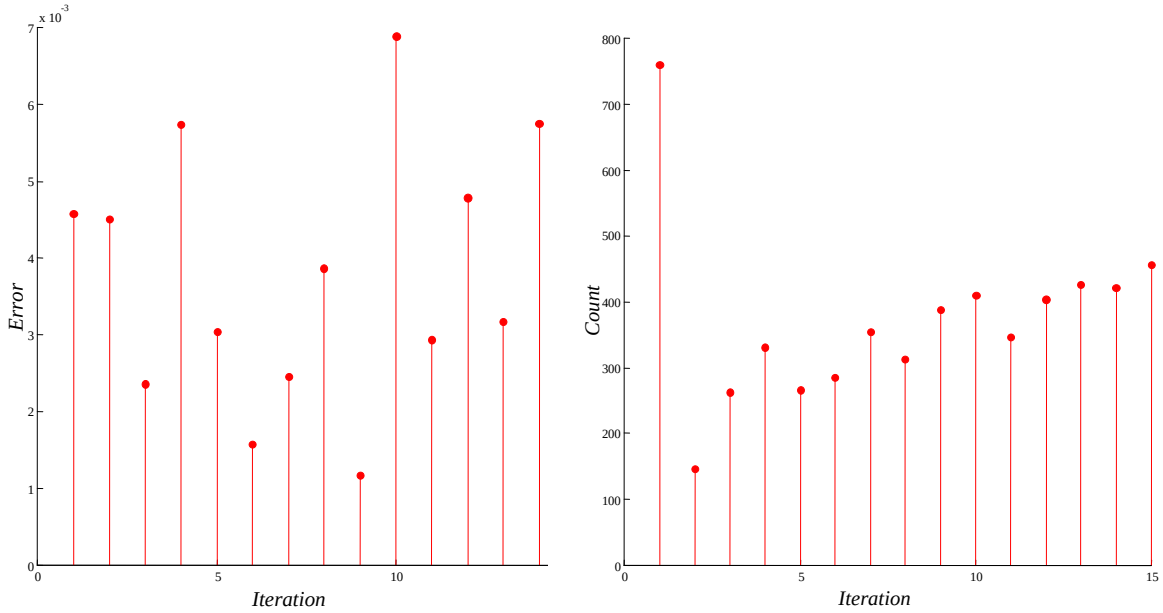


Figure 4.21: Pure rotation. Error and number of iterations using PRA method and a priori information.

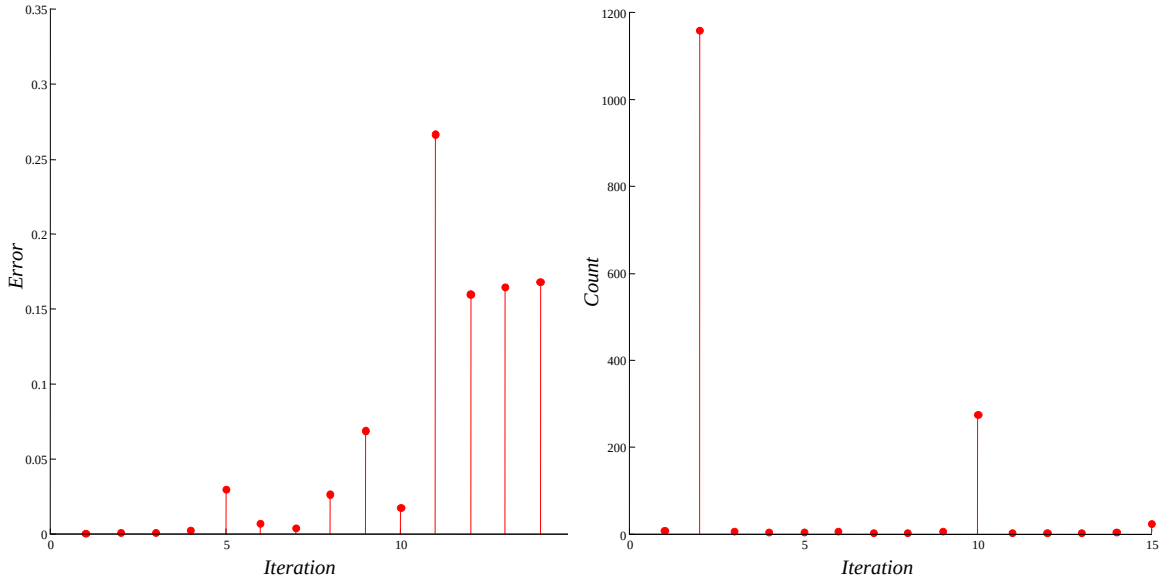


Figure 4.22: Pure translation. Error and number of iterations using Lowe's method.

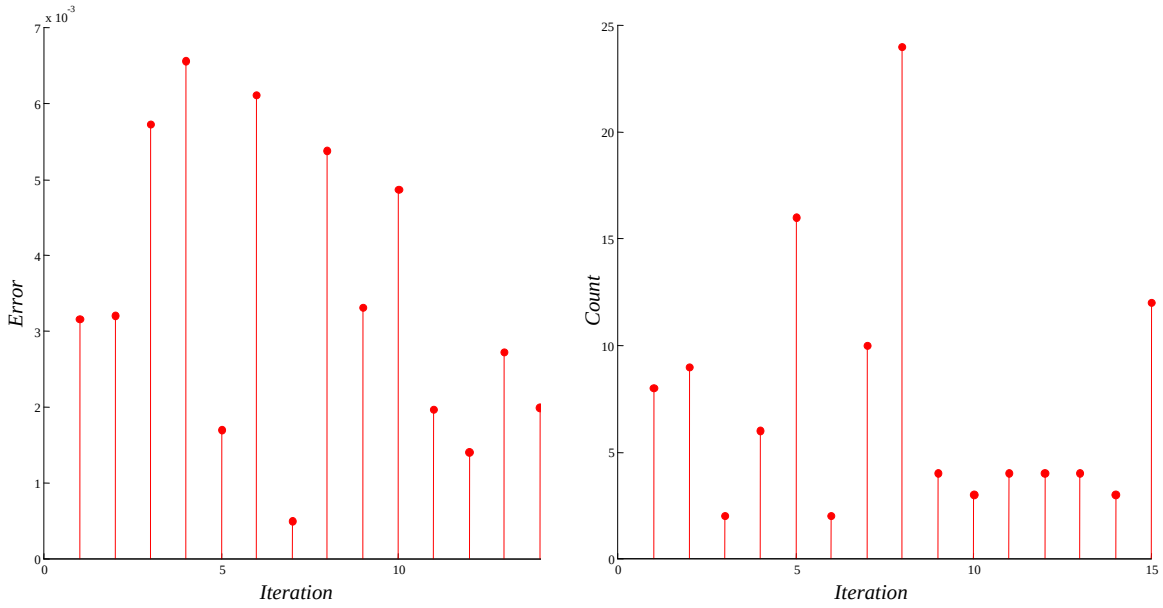


Figure 4.23: Pure rotation. Error and number of iterations using Lowe's method.

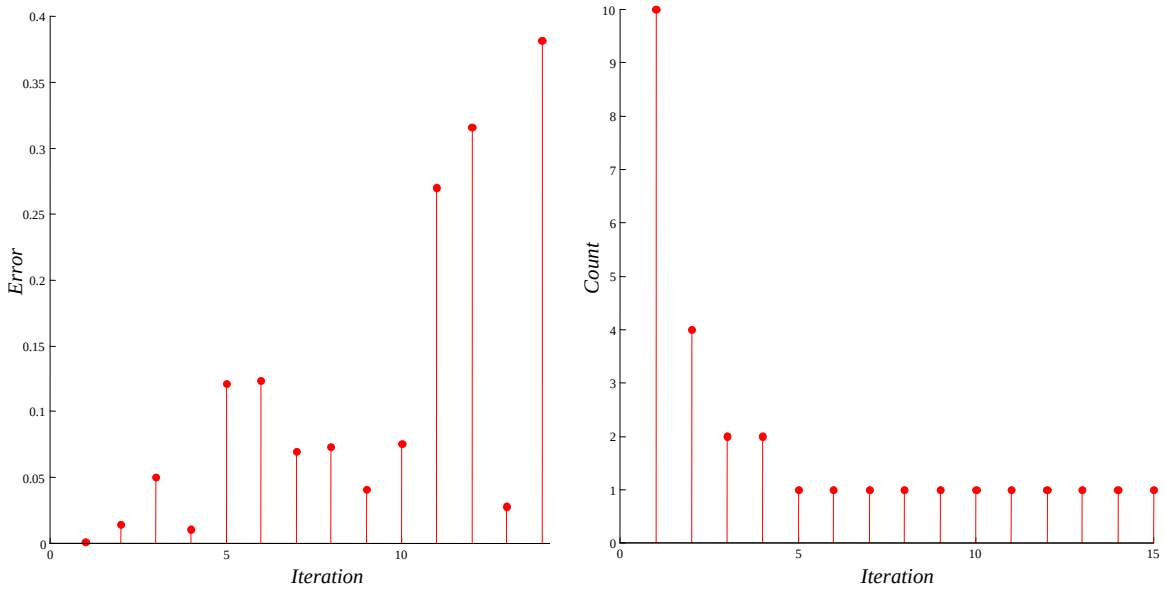


Figure 4.24: Pure translation. Error and number of iterations using POSIT algorithm.

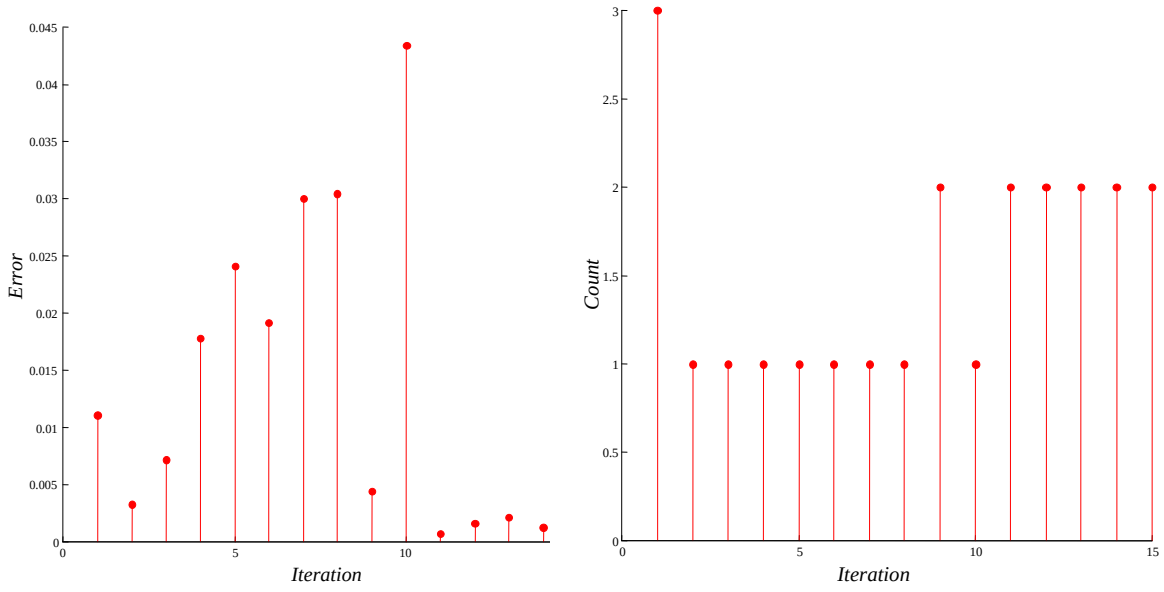


Figure 4.25: Pure rotation. Error and number of iterations using POSIT algorithm.

maximum distance to 3 m.

Figure 4.26: Pose estimation with $q_1 = [1, -0.25, 0]^T$, $q_2 = [2, 0.5, 0]^T$.

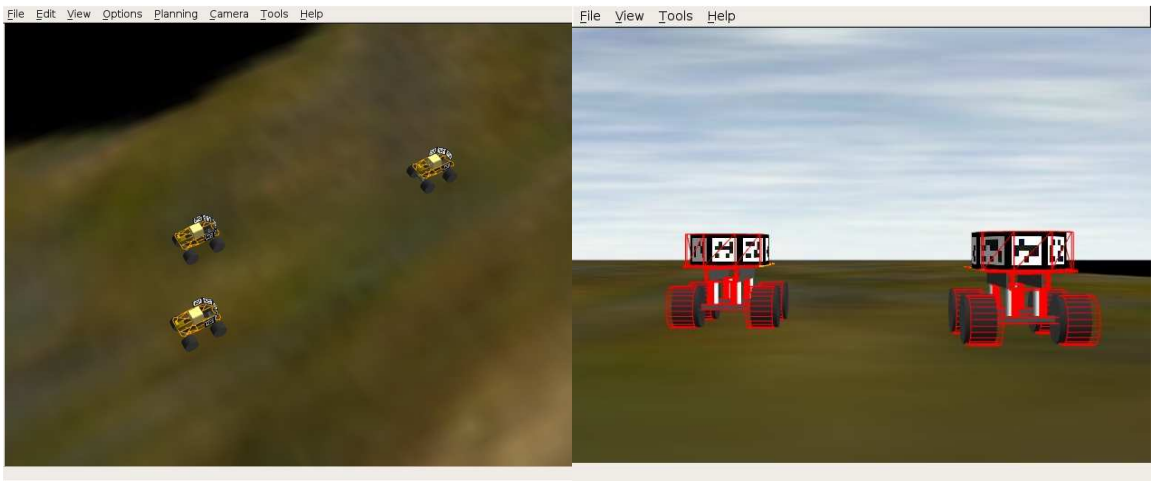


Figure 4.27: Pose estimation with $q_1 = [2.5, -0.5, 0]^T$, $q_2 = [3, 0.5, 0]^T$.

Figure 4.30 shows the computation time of the vision system. The blue line shows the time needed to detect a single robot in the field of view (roughly 10.7 msec). The red line shows the time needed to detect 3 robots plus another robot entering in the field of view at $t = 1.25$ sec.

Figures 4.31 to 4.37 show real, measured, and estimated relative distance ℓ_{12} and

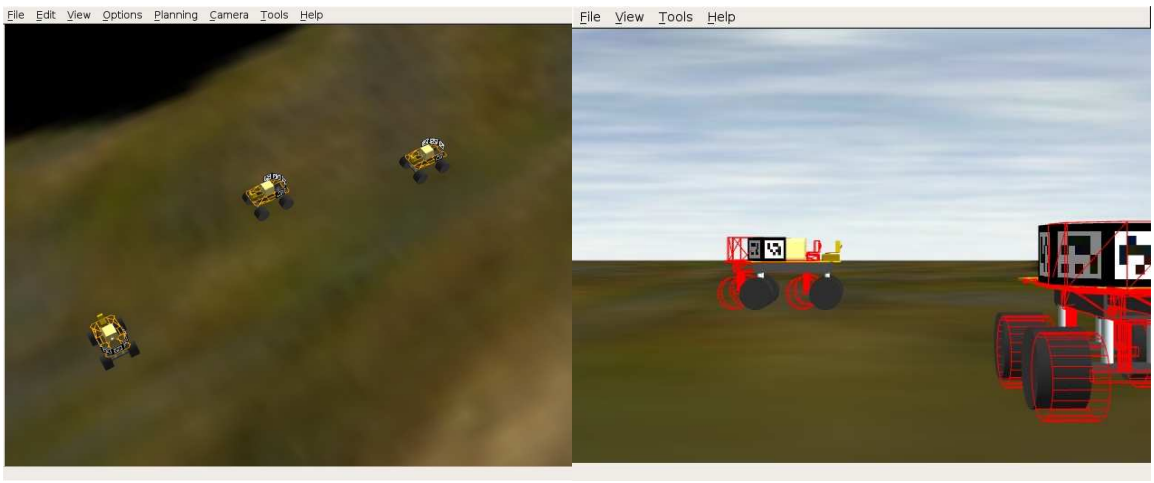


Figure 4.28: Pose estimation with $q_1 = [1.5, -0.5, 0]^T$, $q_2 = \left[4, 0.5, -\frac{\pi}{2}\right]^T$.

bearing α_{12} using the DUKF. The initial values in the simulation are

$$\begin{aligned} \mathbf{x}_0 &= [1.5, 0.25, 0, 1.047, 0.233, 0.232]^T, \\ \mathbf{P}_{\mathbf{x}_0} &= \text{diag}\,(10, 10, 10), \\ \mathbf{w}_0 &= [0, 0]^T, \end{aligned}$$

$\mathbf{P}_{\mathbf{w}_0} = \text{diag}\,(500, 0.01)$ for figures 4.31 to 4.33, and $\mathbf{P}_{\mathbf{w}_0} = \text{diag}\,(10, 0.01)$ for figures 4.34 to 4.37.

It is assumed that there is a bias in the measured variables given by

$$\bar{\mathbf{n}} = [-0.0702,\ 0.0248,\ 0.0236]^T,$$

with measurement noise covariance $\mathbf{R}_n = \text{diag}\,(0.07, 0.03, 0.03)$.

Figures 4.31 and 4.32 show estimation results with the DUKF in open loop. That is, referring to Figure 4.13, the real values of v_1 and ω_1 are sent to the UKF estimator, and real state values $\mathbf{x}_{12}$ are sent to the optimization block. It can be noticed that the optimizer fails to obtain the real leader's velocities due to lack of signal richness, as shown in in Figure 4.32. To improve the richness properties and the convergence, a Gaussian noise signal with covariance $\mathbf{R}_{n'} = \text{diag}\,(0.0035, 0.015, 0.015)$ is added to the measured output vector for the optimization block. The performance is notoriously increased, as shown in Figure 4.33.

Figures 4.34 to 4.37 show the results of the DUKF working in closed loop. It should be noticed the accuracy in the state and velocity estimation. The mean square errors obtained with the estimations are $e_\ell = 1.821 \times 10^{-4}$, $e_\alpha = 5.730 \times 10^{-5}$,

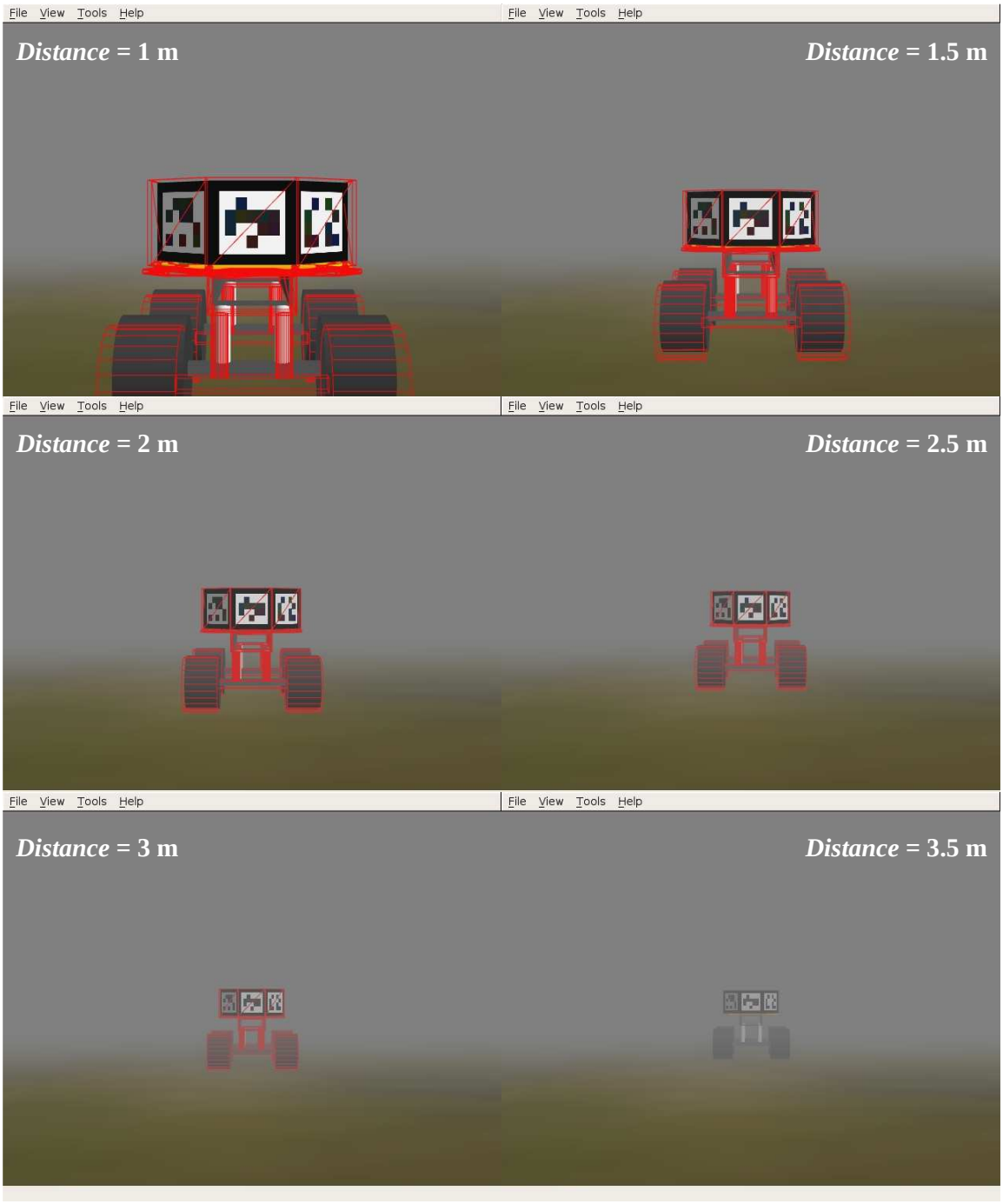


Figure 4.29: Fog effect on pose estimation.

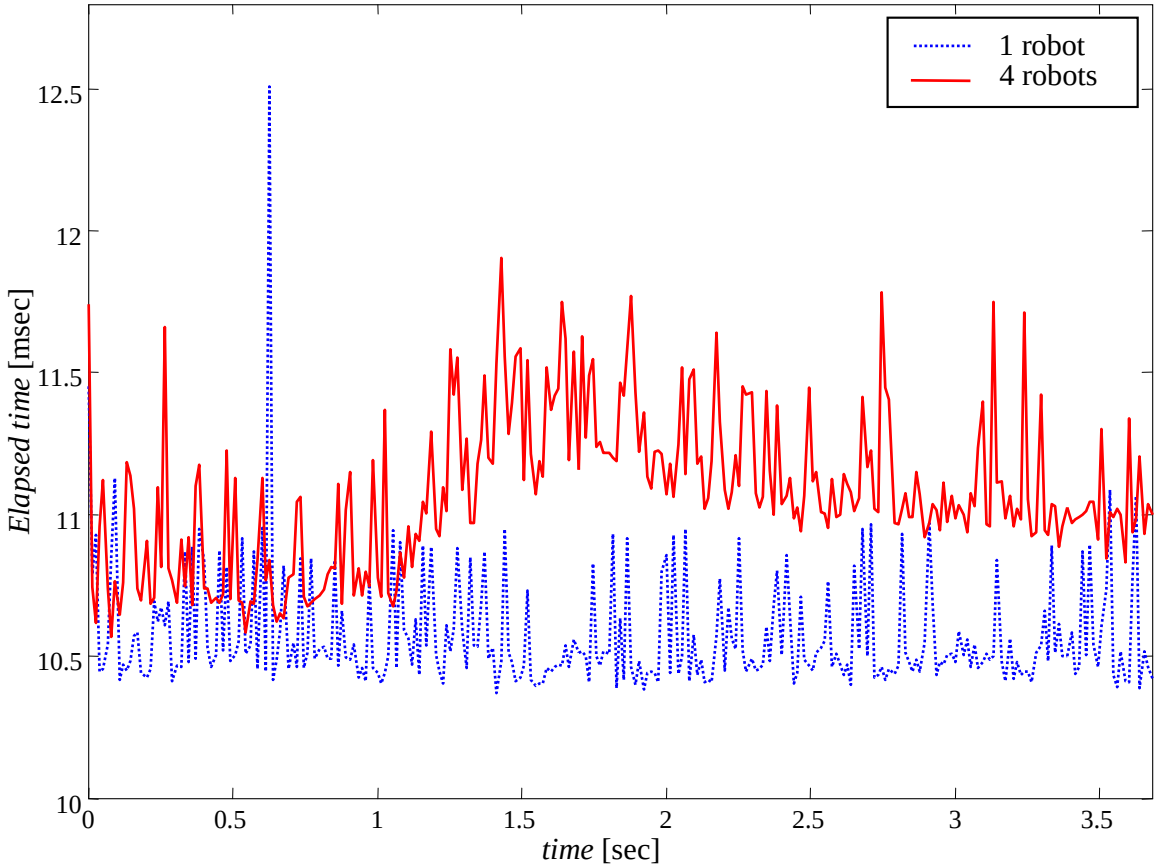


Figure 4.30: Computational load with one robot in the field of view (blue line) and three robots plus another entering at $t = 1.25$ red line).

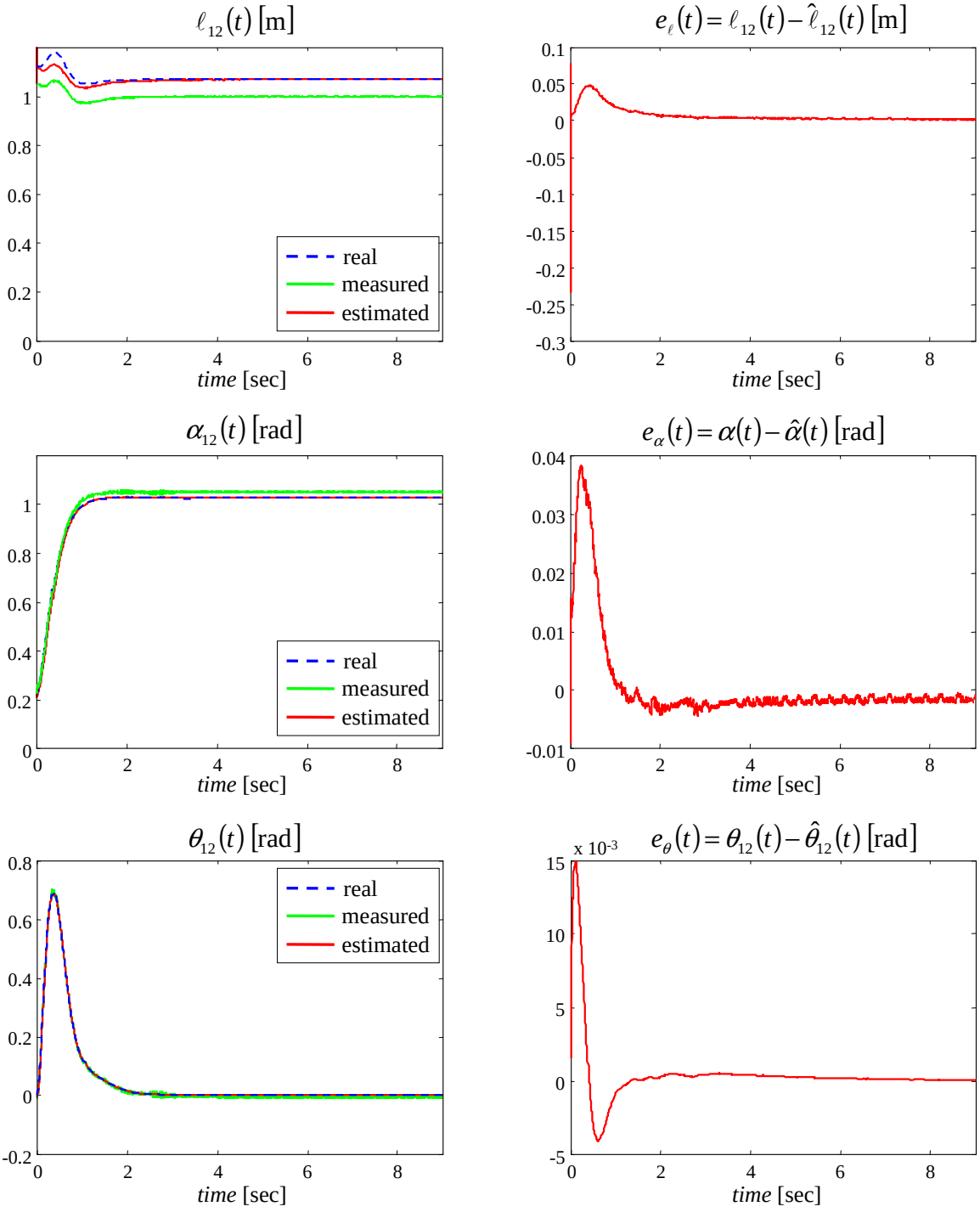


Figure 4.31: Relative distance and bearing estimation using a DUKF.

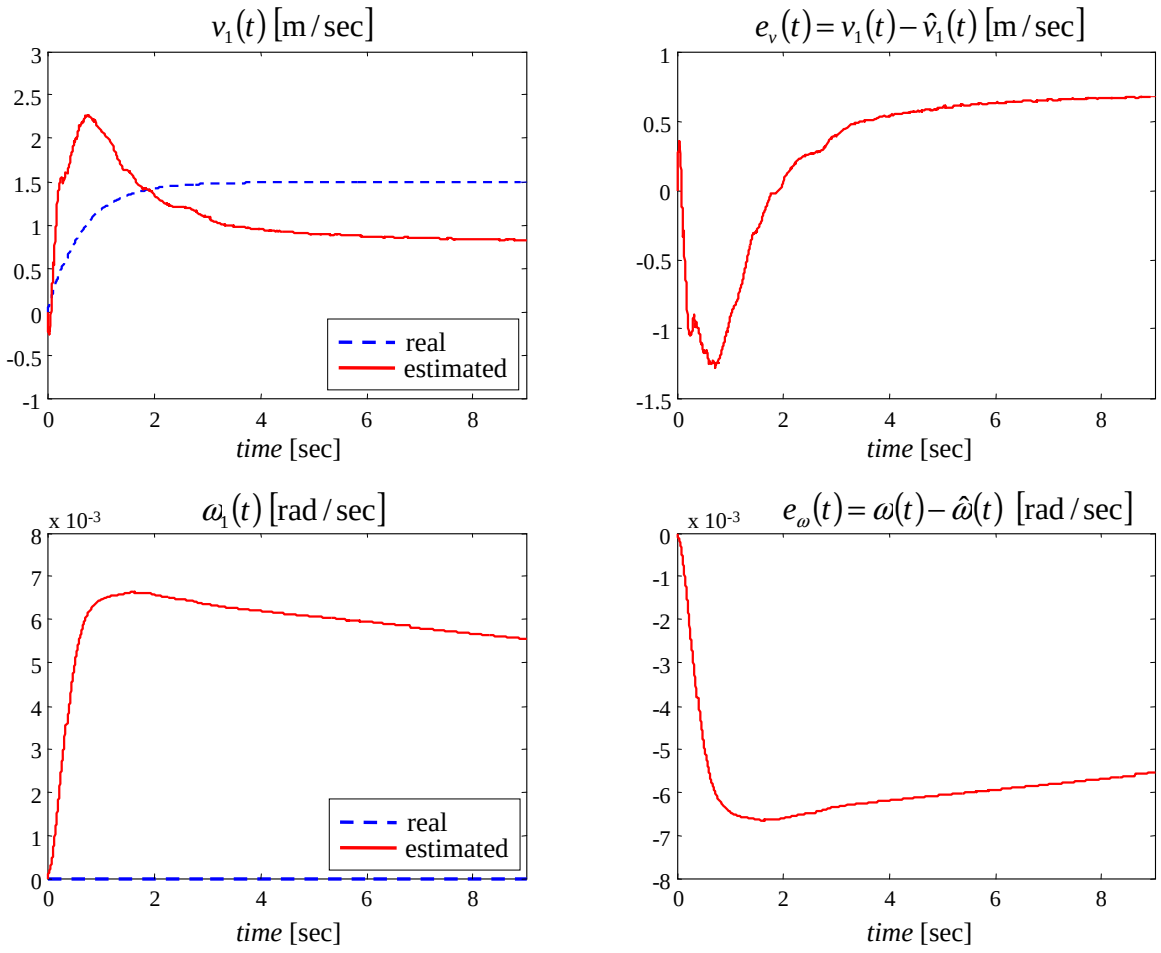


Figure 4.32: Velocity estimation using a DUKF.

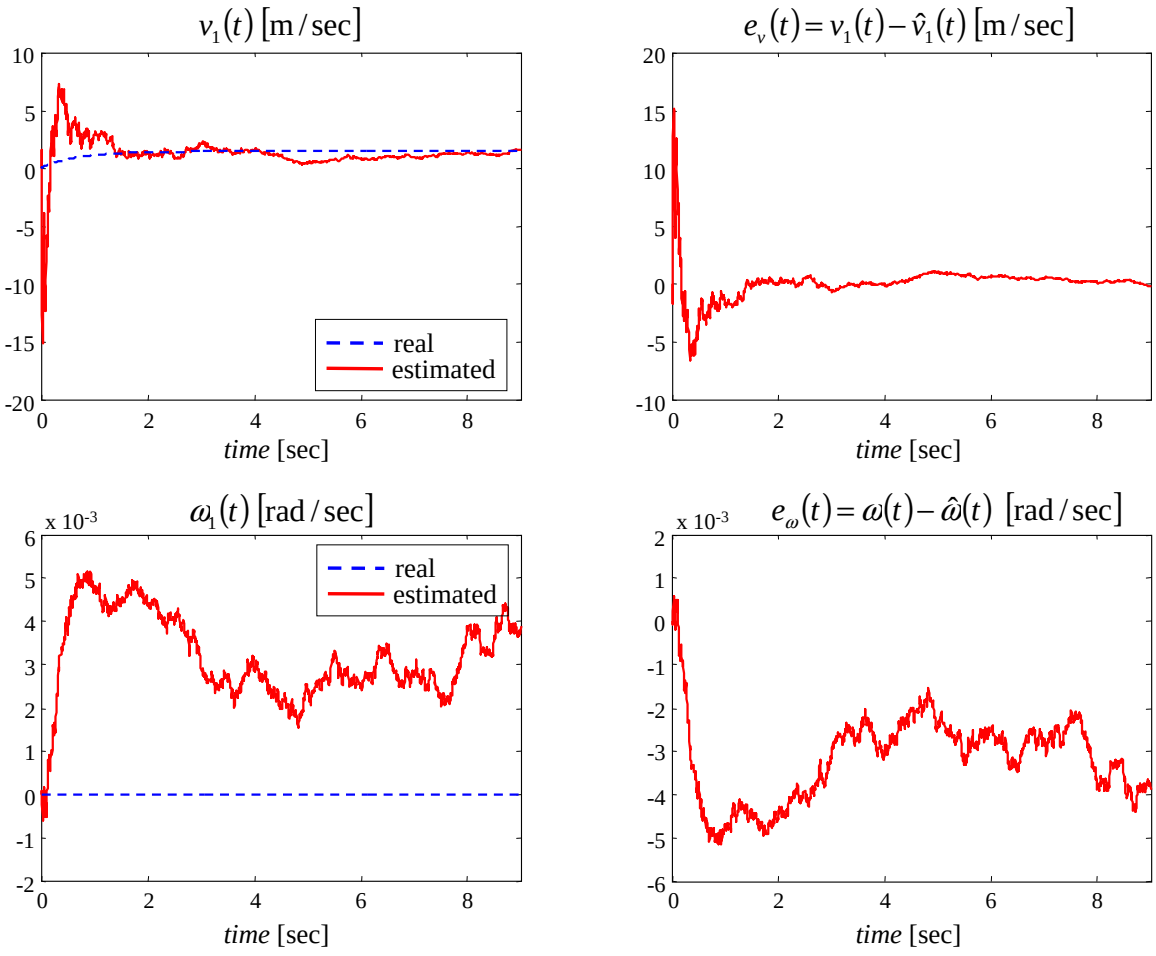


Figure 4.33: Velocity estimation using a DUKF with noise injection.

$e_\theta = 6.029 \times 10^{-6}$, $e_v = 3.648 \times 10^{-4}$, and $e_\omega = 5.272 \times 10^{-7}$.

4.6 Summary

This Chapter presents an architecture for model-based pose estimation for pairs of robots in formation. This architecture combines fiducial-based robot following, three pose estimation algorithms, and nonlinear filtering.

Comparison between pose estimation approaches shows that POSIT presents the fastest convergence rate and simplicity. However, it fails to arrive to a correct result if the value of the coordinate ${}^c z_i$ of the object is smaller than the values of $|{}^c x_i|$ and/or $|{}^c y_i|$ in the camera frame. This problem is not present in the Lowe's or PRA algorithms. PRA algorithm requires notoriously more iterations to achieve convergence. On the other hand, PRA is less complex than Lowe's algorithm. Lowe's algorithm achieves the best results of the three algorithms described here. Its main problem is its complexity and the need of several iterations to achieve convergence.

Fiducial marker detection performs very well in the simulations in a realistic 3D environment for distances between 30 cm and 2 m, and angles between $-\frac{\pi}{2}$ to $\frac{\pi}{2}$. Pose estimation algorithms give accurate measurements of relative distance and bearing. Finally, the implemented dual unscented Kalman filter notoriously improves measurements and is able to estimate unknown velocities.

The main drawback of the current implementation is on feature detection. The method is sensitive to noise and change in lighting, but very fast to compute. Future research will focus results from the dual unscented Kalman filter with feature point extraction to increase robustness of the whole framework.

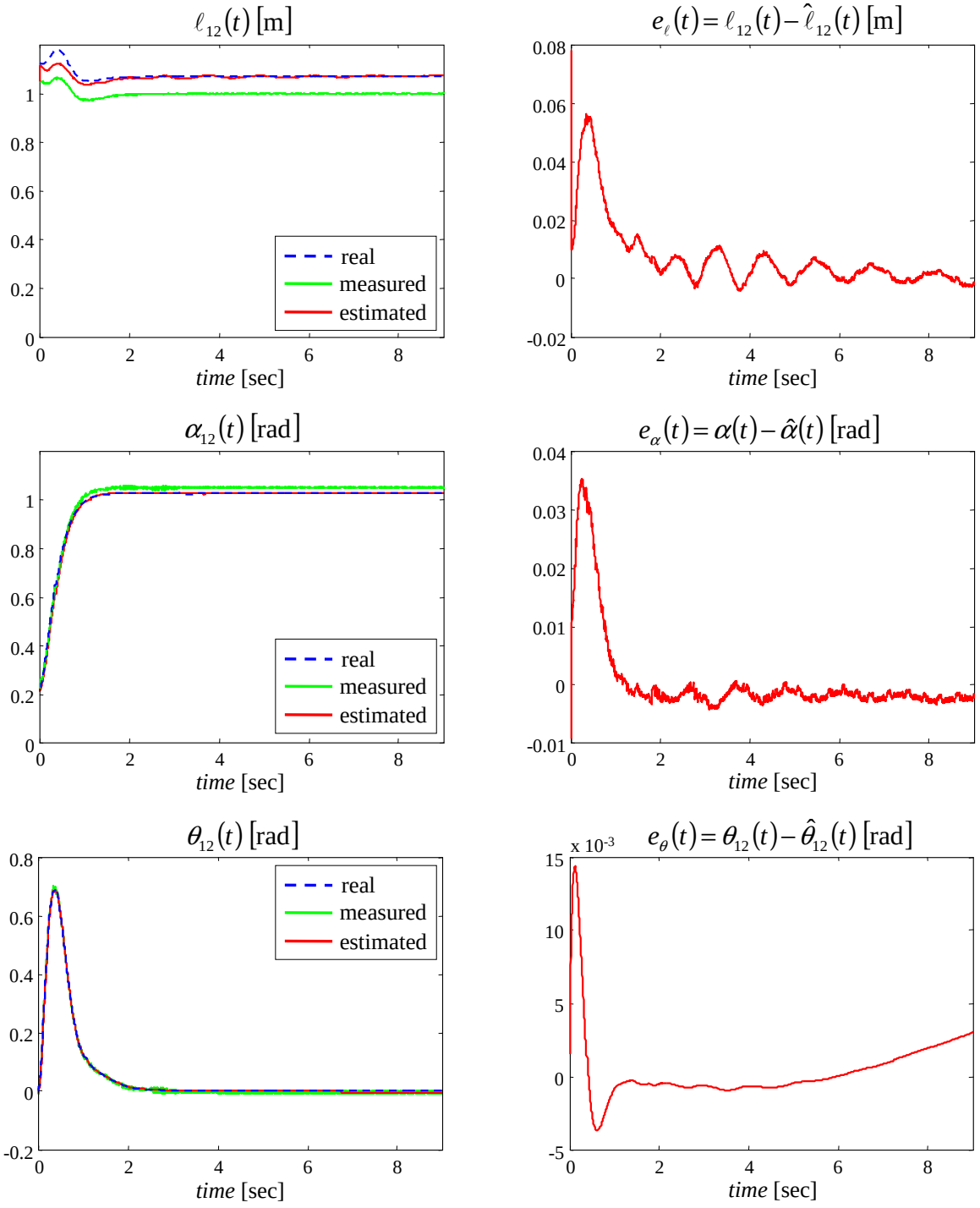


Figure 4.34: Control, relative distance and bearing estimation using a DUKF.

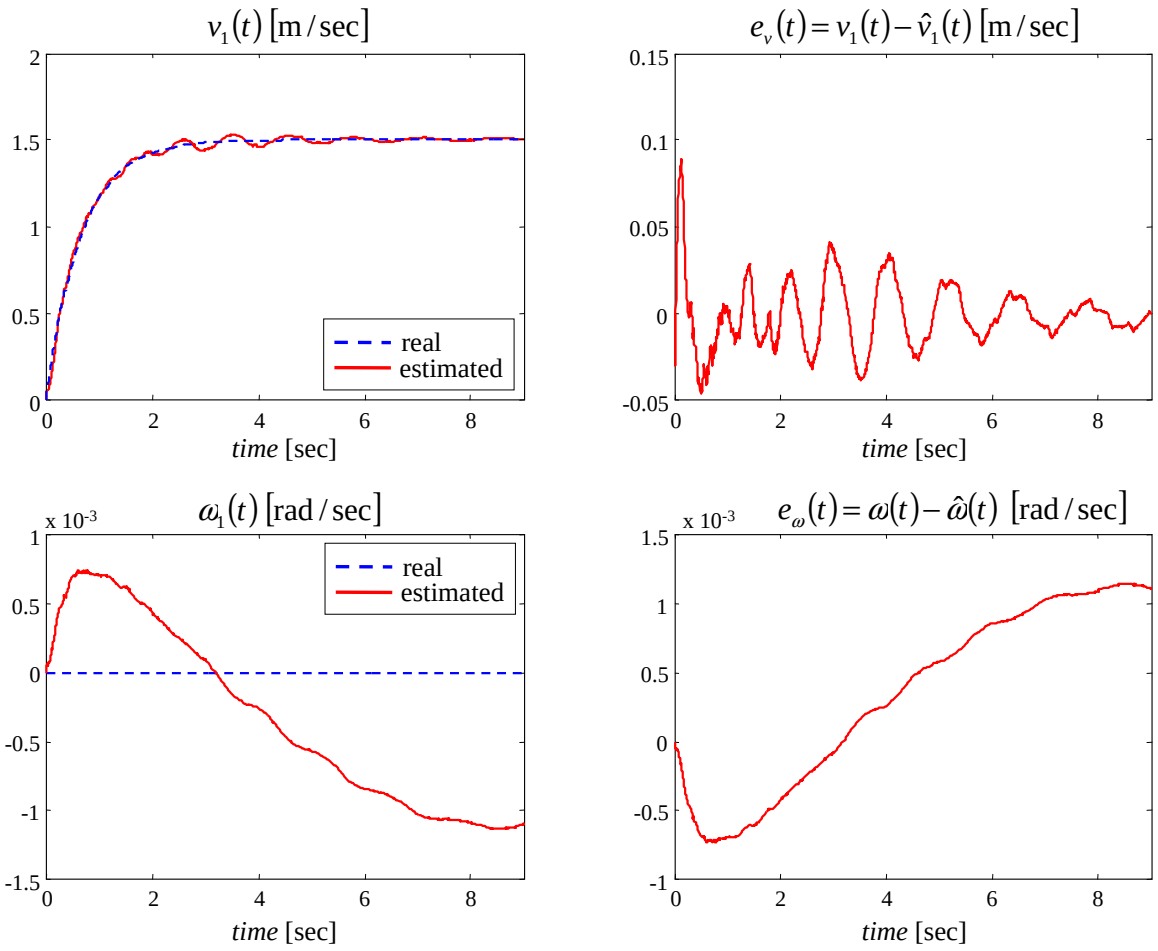


Figure 4.35: Control and velocity estimation using a DUKF.

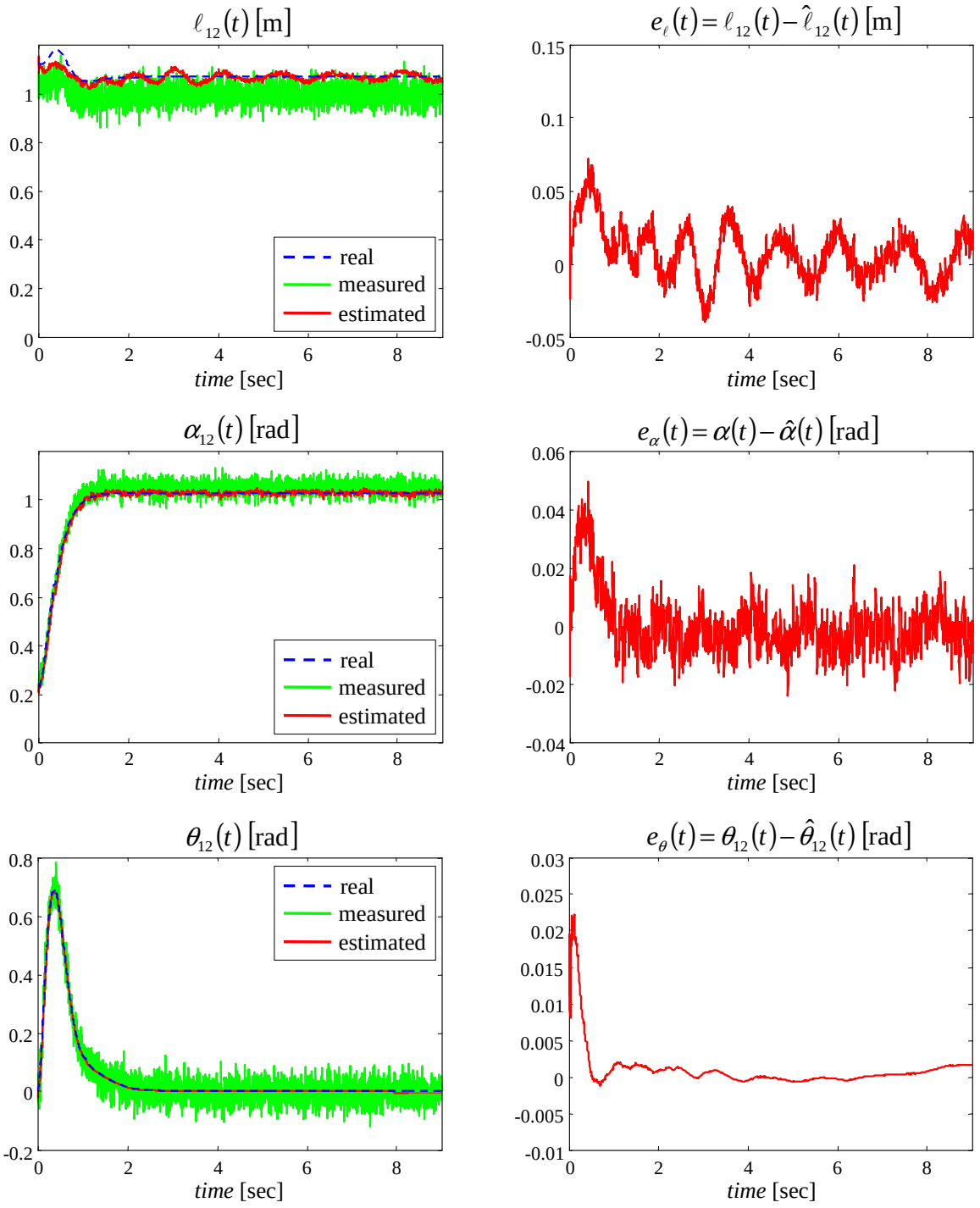


Figure 4.36: Control, relative distance and bearing estimation using a DUKF with noise injection.

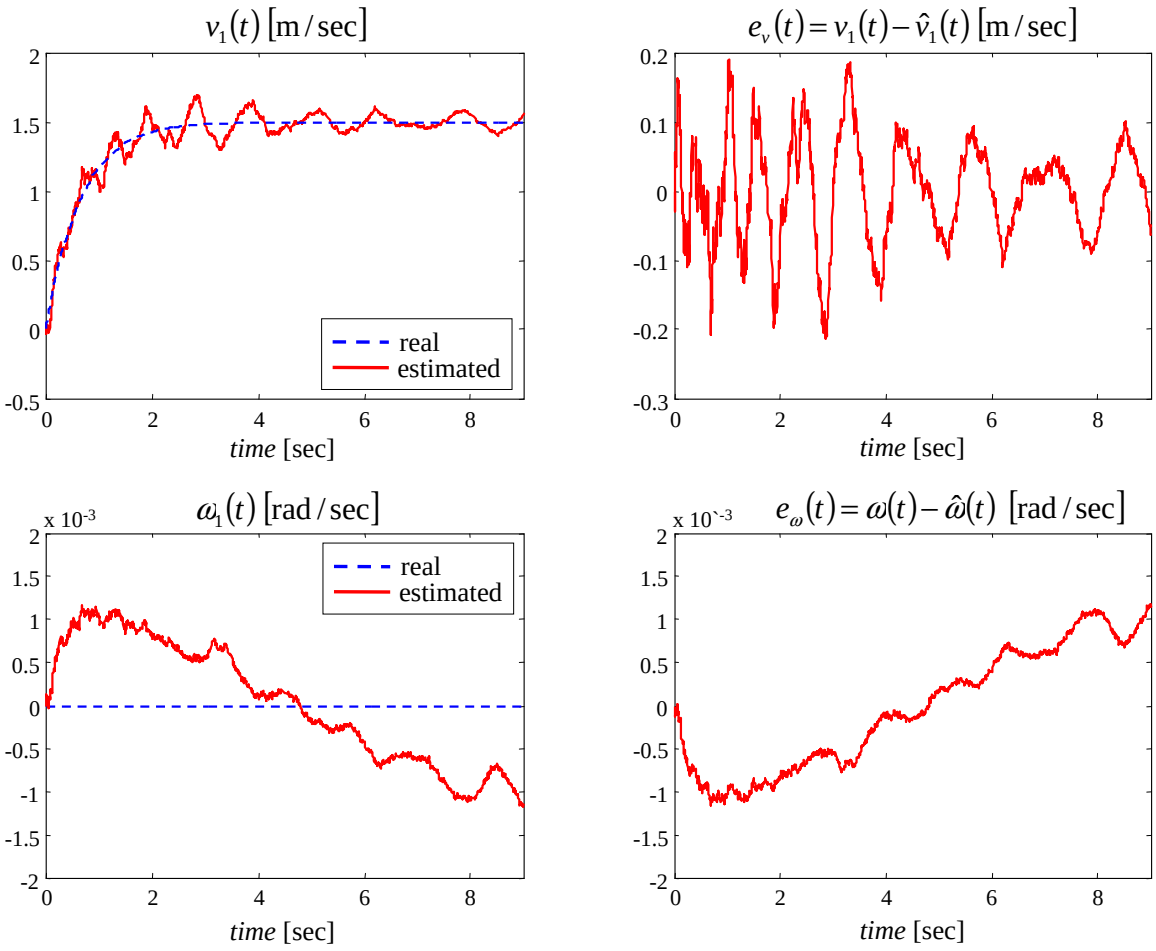


Figure 4.37: Control and velocity estimation using a DUKF with noise injection.

Chapter 5

Vision-based formation control

This Chapter presents several vision-based control strategies for decentralized stabilization of *unmanned vehicle* (UV) formations. The algorithms are designed to solve the leader-follower and two leader-follower coordination problems. In the former case, the robot's node specification is given by the relative distance and bearing to a leader robot. In the latter case, the robot's node specification is given by the relative distances to two leader robots.

The algorithms presented in this chapter can be divided into three main categories depending on the information flow needed for their implementation. The algorithms in the first category are *partial (dynamic) state feedback* nonlinear controllers that require full knowledge of leader's velocities and accelerations. The algorithms in the second category are *robust robust (dynamic) state feedback* nonlinear controllers that requires knowledge of the rate of change of the relative position errors. Finally, the algorithms in the third category are *(dynamic) output feedback* approaches that use *observers* to estimate the derivative of the unmanned vehicles' relative position. Thus, the algorithms in the last category only require knowledge of the leader-follower relative distances and bearings. Both data are computed using measurements from a single monochrome camera, as shown in Chapter 5, eliminating sensitivity to information flow between vehicles and increasing the reliability of the overall system.

The rest of the Chapter is organized as follows. Section 5.1 introduces the different mathematical model for the leader and two-leader-follower systems. The problem statement is given in Section 5.2. Section 5.3 describes and analyzes the three types of nonlinear formation control algorithms presented in this dissertation. Section 5.4 discusses the implementation of a backstepping algorithm using partial information. Section 5.5 provides numerical simulations in the realistic 3D environment MPSLab [99, 100]. A summary in Section 5.6 closes this Chapter.

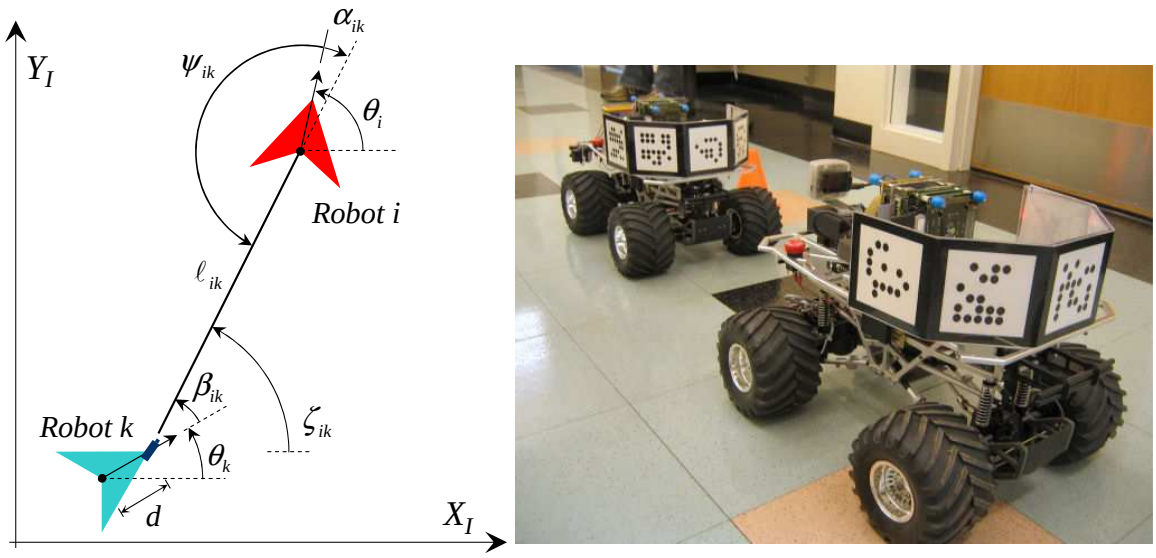


Figure 5.1: Leader-follower formation geometry.

5.1 Formation mathematical models

In this Section, we describe two mathematical models and their variations for the formation control problem. The first model is a leader-follower approach and the second one is a two-leader follower approach. The main purpose of the Section is to arrive to a common mathematical description that allows the design of controllers independently of the problem in consideration.

5.1.1 Leader-follower mathematical model revisited

In Section 2.1.1 we describe the leader-follower polar model as follows, repeated here for clarity

$$\ell_{ik}(t) := \sqrt{(x_i(t) - x_k^c(t))^2 + (y_i(t) - y_k^c(t))^2}, \tag{5.1}$$

$$\alpha_{ik}(t) := \zeta_{ik}(t) - \theta_i(t), \tag{5.2}$$

$$\theta_{ik}(t) := \theta_i(t) - \theta_k(t), \tag{5.3}$$

$$\beta_{ik}(t) := \zeta_{ik}(t) - \theta_k(t) = \alpha_{ik}(t) + \theta_{ik}(t), \tag{5.4}$$

where $\ell_{ik}(t) \in \mathbb{R}_{>0}$, $\alpha_{ik}(t)$, $\beta_{ik}(t)$, $\theta_{ik}(t) \in \mathbb{S}$, $\zeta_{ik}(t) = \text{atan2}(y_i - y_k^c, x_i - x_k^c)$, and $x_k^c(t) = x_k(t) + d\cos\theta_k(t)$, $y_k^c(t) = y_k(t) + d\sin\theta_k(t)$ are the coordinates of the camera, as it is schematized in Figure 5.1

Let $x \in \mathcal{M} \subseteq \mathbb{R}_{\geq 0} \times \mathbb{S}^2$ be defined as

$$x(t) = [x_1, x_2, x_3]^T = [\ell_{ik},\, \alpha_{ik},\, \theta_{ik}]^T . \tag{5.5}$$

Taking time derivative of (5.1), (5.2), and (5.3), the relative model in polar form is given by

$$\dot{x}(t) = g(x)\, u_k(t) + g_i(x)\, u_i(t), \tag{5.6}$$

where

$$g(x) := \begin{bmatrix} -\cos\beta_{ik} & -d\sin\beta_{ik} \\ \frac{\sin\beta_{ik}}{x_1} & -\frac{d\cos\beta_{ik}}{x_1} \\ 0 & -1 \end{bmatrix},$$

$$g_i(x) := \begin{bmatrix} \cos x_2 & 0 \\ -\frac{\sin x_2}{x_1} & -1 \\ 0 & 1 \end{bmatrix},$$

$$\beta_{ik} := x_2 + x_3,$$

and

$$u_k(t) := [v_k(t),\, \omega_k(t)]^T,$$
$$u_i(t) := [v_i(t),\, \omega_i(t)]^T,$$

are the velocities of the follower and the leader, respectively.

Remark 5.1.1. For this model, we consider the follower's velocity as input, the real value of the inputs can be computed using backstepping [45, 69, 130].

Let an input transformation be defined as

$$\begin{bmatrix} v_k(t) \\ \omega_k(t) \end{bmatrix} = \begin{bmatrix} -\cos\beta_{ik} & x_1\sin\beta_{ik} \\ -\frac{1}{d}\sin\beta_{ik} & -\frac{x_1}{d}\cos\beta_{ik} \end{bmatrix} \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.7}$$

where

$$\bar{u}_k(t) := \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.8}$$

is an auxiliary input. Then, system (5.6) takes the form

$$\dot{x}(t) = \bar{g}(x)\,\bar{u}_k(t) + g_i(x)\,u_i(t), \tag{5.9}$$

with

$$\bar{g}(x) := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \frac{\sin\beta_{ik}}{d} & \frac{x_1\cos\beta_{ik}}{d} \end{bmatrix}.$$

Remark 5.1.2. Note that transformation (5.7) is well-defined as long as $d > 0$ and $x_1(t) = \ell_{ik}(t) \geq \ell_{\min} > 0$, where $\ell_{\min}$ is the minimum distance required between robots to avoid collision. However, $d = 0$ in Scorpion robots. This special case is analyzed in the next Subsection.

5.1.1.1 Second order leader-follower model with $d = 0$

For the Scorpion robot, $d = 0$. Then let $x \in \mathcal{M} \subseteq \mathbb{R}_{>0} \times \mathbb{R}^2 \times \mathbb{S}^2$ be redefined as

$$x(t) = [x_{1,1},\, x_{1,2},\, x_{2,1},\, x_{2,2},\, x_{3,1}]^T = \left[\ell_{ik}, \dot{\ell}_{ik}, \alpha_{ik}, \dot{\alpha}_{ik}, \theta_{ik}\right]^T. \tag{5.10}$$

Note that $x_i = x_{i,1}$, $i = 1, \ldots, 3$, coincide with the definition of first order model (5.5). Taking time derivative of (5.10), assuming $d = 0$, we obtain

$$\begin{aligned} \dot{x}(t) = {} & Ax(t) + g(x_{*,1},\, v_k)\,\varpi_k(t) + g_0(x, v_k) \\ & + g_{v,1}(x,\, v_k)\,u_i(t) + g_{v,2}(x_{*,1})\,\dot{u}_i(t), \end{aligned} \tag{5.11}$$

where $A = \begin{bmatrix} \text{blockdiag}(A_1,\, A_2) \\ 0_{1\times 4} \end{bmatrix}$, with $A_i = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $i = 1, 2$, is in Brunowsky general form,

$$g(x_{*,1},\, v_k) := \begin{bmatrix} 0 & 0 \\ -\cos\beta_{ik} & -v_k\sin\beta_{ik} \\ 0 & 0 \\ \frac{\sin\beta_{ik}}{x_{1,1}} & -\frac{v_k\cos\beta_{ik}}{x_{1,1}} \\ 0 & -1 \end{bmatrix},$$

$$g_0(x,\, v_k) := \begin{bmatrix} 0 \\ v_k x_{2,2} \sin\beta_{ik} \\ 0 \\ v_k \frac{x_{1,1} x_{2,2} \cos\beta_{ik} - x_{1,2} \sin\beta_{ik}}{x_{1,1}^2} \\ 0 \end{bmatrix},$$

$$g_{v,1}(x,\, v_k) := \begin{bmatrix} 0 & 0 \\ -x_{2,2} \sin x_{2,1} & v_k \sin\beta_{ik} \\ 0 & 0 \\ -\frac{x_{1,1} x_{2,2} \cos x_{2,1} - x_{1,2} \sin x_{2,1}}{x_{1,1}^2} & \frac{v_k \cos\beta_{ik}}{x_{1,1}} \\ 0 & 1 \end{bmatrix},$$

$$g_{v,2}(x_{*,1}) := \begin{bmatrix} 0 & 0 \\ \cos x_{2,1} & 0 \\ 0 & 0 \\ -\frac{\sin x_{2,1}}{x_{1,1}} & -1 \\ 0 & 0 \end{bmatrix},$$

and

$$\beta_{ik} := x_{2,1} + x_{3,1}.$$

The control signal takes the special form

$$\varpi_k(t) := \left[\dot{v}_k(t)\,,\, \omega_k(t)\right]^T, \tag{5.12}$$

and we have used $x_{*,1}$ to stress dependency on measurable variables.

Remark 5.1.3. In this case, backstepping has to be applied just to $\omega_k(t)$, because the model (5.11) can be used to compute $F_k = m_k \dot{v}_k$, according to equation (2.1).

Using a control transformation similar to (5.7), that is,

$$\begin{bmatrix} \dot{v}_k(t) \\ \omega_k(t) \end{bmatrix} = \begin{bmatrix} -\cos\beta_{ik} & x_{1,1} \sin\beta_{ik} \\ -\frac{1}{v_k} \sin\beta_{ik} & -\frac{x_{1,1}}{v_k} \cos\beta_{ik} \end{bmatrix} \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.13}$$

we obtain

$$\begin{aligned} \dot{x}(t) = {} & Ax(t) + \bar{g}(x_{*,1},\, v_k)\, \bar{u}_k(t) + g_0(x, v_k) \\ & + g_{v,1}(x,\, v_k)\, u_i(t) + g_{v,2}(x_{*,1})\, \dot{u}_i(t), \end{aligned} \tag{5.14}$$

with $\bar{u}_k(t) = [\bar{v}_k(t)\ \bar{\omega}_k(t)]^T$ and

$$\bar{g}(x_{*,1},\, v_k) := \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ \frac{1}{v_k}\sin\beta_{ik} & \frac{x_{1,1}}{v_k}\cos\beta_{ik} \end{bmatrix}.$$

Remark 5.1.4. Note that the transformation (5.13) is defined as long as $|v_k(t)| \geq v_{\min} > 0$ and $x_{1,1}(t) = \ell_{ik}(t) \geq \ell_{\min} > 0$, where $v_{\min}$ is the minimum speed of the follower robot, and $\ell_{\min}$ is the minimum distance required between robots to avoid collision.

5.1.1.2 Second order leader-follower model with $d \neq 0$

Model (5.11) suggests that the control signals can be computed directly without using backstepping by higher-order time differentiation. Let $x \in \mathcal{M} \subseteq \mathbb{R}_{>0} \times \mathbb{R}^3 \times \mathbb{S}^2$ be defined as

$$x(t) = [x_{1,1},\, x_{1,2},\, x_{2,1},\, x_{2,2},\, x_{3,1},\, x_{3,2}]^T = \left[\ell_{ik}, \dot{\ell}_{ik}, \alpha_{ik}, \dot{\alpha}_{ik}, \theta_{ik}, \dot{\theta}_{ik}\right]^T.$$

Note that $x_i = x_{i,1}$, $i = 1, \ldots, 3$, coincide with the definition of first order model (5.5). Taking time derivative of (5.6) and assuming $d \neq 0$, we obtain

$$\dot{x}(t) = Ax(t) + g(x_{*,1})\,\dot{u}_k(t) + g_0(x,\, u_k) + g_v(x_{*,1})\,\dot{u}_i(t), \tag{5.15}$$

where denote all measurable vector components $x_{1,1}$, $x_{2,1}$, ..., $A = \text{blockdiag}(A_1,\, A_2,\, A_3)$ is in Brunowsky canonical form,

$$g(x_{*,1}) := \begin{bmatrix} 0 & 0 \\ -\cos\beta_{ik} & -d\sin\beta_{ik} \\ 0 & 0 \\ \frac{\sin\beta_{ik}}{x_{1,1}} & -\frac{d\cos\beta_{ik}}{x_{1,1}} \\ 0 & 0 \\ 0 & -1 \end{bmatrix},$$

$$\beta_{ik} := x_{2,1} + x_{3,1},$$

$$g_0(x,\, u_k) := \begin{bmatrix} 0 \\ v_k x_{3,2} \sin\beta_{ik} - d\omega_k x_{3,2} \cos\beta_{ik} + x_{1,1} x_{2,2} (\omega_k + x_{2,2} + x_{3,2}) \\ 0 \\ \frac{1}{x_{1,1}} \left[v_k x_{3,2} \cos\beta_{ik} + d\omega_k x_{3,2} \sin\beta_{ik} - x_{1,2} (\omega_k + 2x_{2,2} + x_{3,2}) \right] \\ 0 \\ 0 \end{bmatrix},$$

$$g_v(x_{*,1}) := \begin{bmatrix} 0 & 0 \\ \cos x_{2,1} & 0 \\ 0 & 0 \\ -\frac{\sin x_{2,1}}{x_{1,1}} & -1 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

Using a control transformation similar to (5.7), that is,

$$\begin{bmatrix} \dot{v}_k(t) \\ \dot{\omega}_k(t) \end{bmatrix} = \begin{bmatrix} -\cos\beta_{ik} & x_{1,1}\sin\beta_{ik} \\ -\frac{1}{d}\sin\beta_{ik} & -\frac{x_{1,1}}{d}\cos\beta_{ik} \end{bmatrix} \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.16}$$

we obtain

$$\dot{x}(t) = Ax(t) + \bar{g}(x_{*,1})\, \bar{u}_k(t) + g_0(x, u_k) + g_v(x_{*,1})\, \dot{u}_i(t), \tag{5.17}$$

with $\bar{u}_k(t) = [\bar{v}_k(t)\ \bar{\omega}_k(t)]^T$ and

$$\bar{g}(x_{*,1}) := \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ \frac{1}{d}\sin\beta_{ik} & \frac{x_{1,1}}{d}\cos\beta_{ik} \end{bmatrix}.$$

5.1.2 Two leader-follower mathematical model

The second type of coordination algorithms presented in this dissertation are based on the relative distances between the follower robot k and two leaders i and j. Therefore, we have develop similar models for the case of relative measurements shown in Figure 5.2.

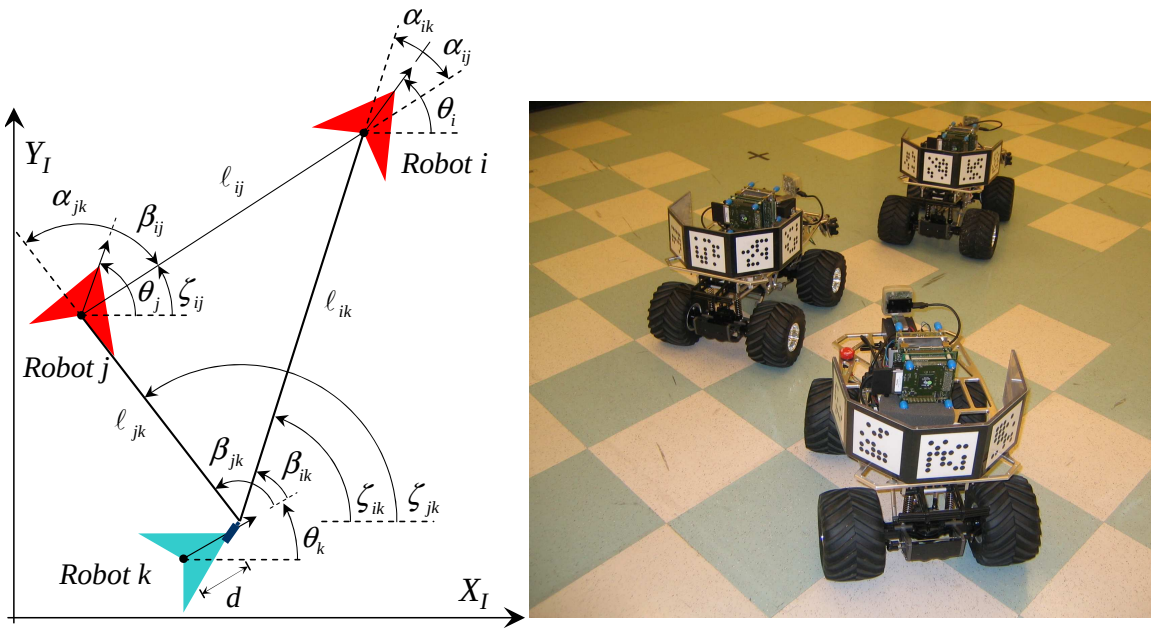


Figure 5.2: Two leader-follower formation geometry.

Let $x(t) \in \mathcal{M} \subseteq \mathbb{R}^2_{\geq 0} \times \mathbb{S}^4$ be defined as

$$x(t) = [x_1,\, x_2,\, x_3,\, x_4,\, x_5,\, x_6]^T := [\ell_{ik},\, \ell_{jk},\, \alpha_{ik},\, \alpha_{jk},\, \theta_{ik},\, \theta_{jk}]^T . \tag{5.18}$$

Then, taking time derivative, we have

$$\dot{x}(t) = g(x)\, u_k(t) + g_i(x)\, u_i(t) + g_j(x)\, u_j(t) , \tag{5.19}$$

with

$$g(x) := \begin{bmatrix} -\cos\beta_{ik} & -d\sin\beta_{ik} \\ -\cos\beta_{jk} & -d\sin\beta_{jk} \\ \frac{\sin\beta_{ik}}{x_1} & -\frac{d\cos\beta_{ik}}{x_1} \\ \frac{\sin\beta_{jk}}{x_2} & -\frac{d\cos\beta_{jk}}{x_2} \\ 0 & -1 \\ 0 & -1 \end{bmatrix},$$

$$g_i(x) := \begin{bmatrix} \cos x_3 & 0 \\ 0 & 0 \\ -\frac{\sin x_3}{x_1} & -1 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \text{ and } g_j(x) := \begin{bmatrix} 0 & 0 \\ \cos x_4 & 0 \\ 0 & 0 \\ -\frac{\sin x_4}{x_2} & -1 \\ 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\beta_{ik} := x_3 + x_5,$$

$$\beta_{jk} := x_4 + x_6.$$

Let the input transformation be given by

$$\begin{bmatrix} v_k(t) \\ \omega_k(t) \end{bmatrix} = \frac{1}{\sin \delta_k} \begin{bmatrix} -\sin \beta_{jk} & \sin \beta_{ik} \\ \frac{\cos \beta_{jk}}{d} & -\frac{\cos \beta_{ik}}{d} \end{bmatrix} \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.20}$$

with $\delta_k := \zeta_{jk} - \zeta_{ik} = x_4 + x_6 - x_3 - x_5$. Then, system (5.19) takes the form

$$\dot{x}(t) = \bar{g}(x)\,\bar{u}_k(t) + g_i(x)\,u_i(t) + g_j(x)\,u_j(t), \tag{5.21}$$

with

$$\bar{g}(x) := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -\frac{\cos \delta_k}{x_1 \sin \delta_k} & \frac{1}{x_1 \sin \delta_k} \\ -\frac{1}{x_2 \sin \delta_k} & \frac{\cos \delta_k}{x_2 \sin \delta_k} \\ -\frac{\cos \beta_{jk}}{d \sin \delta_k} & \frac{\cos \beta_{ik}}{d \sin \delta_k} \\ -\frac{\cos \beta_{jk}}{d \sin \delta_k} & \frac{\cos \beta_{ik}}{d \sin \delta_k} \end{bmatrix},$$

and the auxiliary control input is given by

$$\bar{u}_k(t) := [\bar{v}_k(t), \bar{\omega}_k(t)]^T,$$

as in the leader-follower case.

Remark 5.1.5. Transformation (5.20) if well-defined as long as $\delta_k \neq 0$ and $d > 0$. The case $\delta_k = 0$ can happen when one of the leading robots obstructs the line of sight of the follower or when both robots occupy the same point (impossible due to collision). The former case must be avoided due to the lack of sensor data by switching to a leader-follower controller.

5.1.2.1 Second order two-leader-follower model with $d = 0$

As mentioned, for Scorpion robots $d = 0$. Then, let $x(t) \in \mathcal{M} \subseteq \mathbb{R}^2_{\geq 0} \times \mathbb{S}^4 \times \mathbb{R}^4$ be defined as

$$\begin{aligned} x(t) &:= [x_{1,1},\, x_{1,2},\, x_{2,1},\, x_{2,2},\, x_{3,1},\, x_{3,2},\, x_{4,1},\, x_{4,2},\, x_{5,1},\, x_{6,1}]^T \\ &= \left[\ell_{ik},\, \dot{\ell}_{ik},\, \ell_{jk},\, \dot{\ell}_{jk},\, \alpha_{ik},\, \dot{\alpha}_{ik},\, \alpha_{jk},\, \dot{\alpha}_{jk},\, \theta_{ik},\, \theta_{jk}\right]^T. \end{aligned}$$

Note that $x_i = x_{i,1}$, $i = 1, \ldots, 6$, coincide with the definition of first order model (5.18). Taking time derivative of (5.19) assuming that $d = 0$, we obtain

$$\begin{aligned} \dot{x}(t) = Ax(t) &+ g(x_{*,1}, v_k)\,\varpi(t) + g_0(x, v_k) \\ &+ g_{v,2}(x)\,V_{k,2}(t) + g_{v,1}(x)\,V_{k,1}(t), \end{aligned} \tag{5.22}$$

where

$$A := \begin{bmatrix} \text{blockdiag}(A_1, \ldots, A_4) & 0_{8\times 2} \\ 0_{2\times 8} & 0_{2\times 2} \end{bmatrix},$$

$$g(x_{*,1}, v_k) := \begin{bmatrix} 0 & 0 \\ -\cos\beta_{ik} & -v_k \sin\beta_{ik} \\ 0 & 0 \\ -\cos\beta_{jk} & -v_k \sin\beta_{jk} \\ 0 & 0 \\ \frac{\sin\beta_{ik}}{x_{1,1}} & -\frac{x_{1,1} v_k \cos\beta_{ik}}{x_{1,1}^2} \\ 0 & 0 \\ \frac{\sin\beta_{jk}}{x_{2,1}} & -\frac{x_{2,1} v_k \cos\beta_{jk}}{x_{2,1}^2} \\ 0 & -1 \\ 0 & -1 \end{bmatrix},$$

$$g_0\left(x, v_k\right) := \begin{bmatrix} 0 \\ v_k x_{3,2} \sin \beta_{ik} \\ 0 \\ v_k x_{4,2} \sin \beta_{jk} \\ 0 \\ v_k \frac{x_{1,1} x_{3,2} \cos \beta_{ik} - x_{1,2} \sin \beta_{ik}}{x_{1,1}^2} \\ 0 \\ v_k \frac{x_{2,1} x_{4,2} \cos \beta_{jk} - x_{2,2} \sin \beta_{jk}}{x_{2,1}^2} \\ 0_{2\times 1} \end{bmatrix},$$

$$g_{v,1}\left(x, v_k\right) := \left[g_{v,1}^1\left(x, v_k\right),\, g_{v,1}^2\left(x, v_k\right),\, g_{v,1}^3\left(x, v_k\right),\, g_{v,1}^4\left(x, v_k\right)\right]$$

$$g_{v,1}^1\left(x, v_k\right) := \begin{bmatrix} 0 \\ -x_{3,2} \sin x_{3,1} \\ 0_{3\times 1} \\ \frac{x_{1,2} \sin x_{3,1} - x_{1,1} x_{3,2} \cos x_{3,1}}{x_{1,1}^2} \\ 0_{4\times 1} \end{bmatrix},\; g_{v,1}^2\left(x, v_k\right) := \begin{bmatrix} 0 \\ v_k \sin \beta_{ik} \\ 0_{3\times 1} \\ \frac{x_{1,1} v_k \cos \beta_{ik}}{x_{1,1}^2} \\ 0_{4\times 1} \end{bmatrix},$$

$$g_{v,1}^3\left(x, v_k\right) := \begin{bmatrix} 0_{3\times 1} \\ -x_{4,2} \sin x_{4,1} \\ 0_{3\times 1} \\ \frac{x_{2,2} \sin x_{4,1} - x_{2,1} x_{4,2} \cos x_{4,1}}{x_{2,1}^2} \\ 0_{2\times 1} \end{bmatrix},\; g_{v,1}^4\left(x, v_k\right) := \begin{bmatrix} 0_{3\times 1} \\ v_k \sin \beta_{jk} \\ 0_{3\times 1} \\ \frac{x_{2,1} v_k \cos \beta_{jk}}{x_{2,1}^2} \\ 0_{2\times 1} \end{bmatrix},$$

$$g_{v,2}\left(x_{*,1}\right) := \left[g_{v,2}^1\left(x_{*,1}\right),\, g_{v,2}^2\left(x_{*,1}\right),\, g_{v,2}^3\left(x_{*,1}\right),\, g_{v,2}^4\left(x_{*,1}\right)\right],$$

$$g_{v,2}^1\left(x_{*,1}\right) := \begin{bmatrix} 0 \\ \cos x_{3,1} \\ 0_{3\times 1} \\ -\frac{\sin x_{3,1}}{x_{1,1}} \\ 0_{4\times 1} \end{bmatrix},\; g_{v,2}^2\left(x_{*,1}\right) := \begin{bmatrix} 0_{5\times 1} \\ -1 \\ 0_{4\times 1} \end{bmatrix},$$

$$g_{v,2}^{3}\left(x_{*,1}\right):=\left[\begin{array}{c} 0_{3\times 1} \\ \cos x_{4,1} \\ 0_{3\times 1} \\ -\frac{\sin x_{4,1}}{x_{2,1}} \\ 0_{2\times 1} \end{array}\right],\ g_{v,2}^{4}\left(x_{*,1}\right):=\left[\begin{array}{c} 0_{7\times 1} \\ -1 \\ 0_{2\times 1} \end{array}\right],$$

$$\beta_{ik}:=x_{3,1}+x_{5,1},$$

$$\beta_{jk}:=x_{4,1}+x_{6,1},$$

and

$$\begin{aligned} V_{k,1}\left(t\right) &:= \left[v_i\left(t\right),\, \omega_i\left(t\right),\, v_j\left(t\right),\, \omega_j\left(t\right)\right]^T, \\ V_{k,2}\left(t\right) &:= \left[\dot{v}_i\left(t\right),\, \dot{\omega}_i\left(t\right),\, \dot{v}_j\left(t\right),\, \dot{\omega}_j\left(t\right)\right]^T, \end{aligned} \tag{5.23}$$

depend on leaders' velocities.

Using a control transformation similar to (5.7), that is,

$$\left[\begin{array}{c} \dot{v}_k\left(t\right) \\ \omega_k\left(t\right) \end{array}\right]=\left[\begin{array}{cc} -\frac{\sin\beta_{jk}}{\sin\delta_k} & \frac{\sin\beta_{ik}}{\sin\delta_k} \\ \frac{\cos\beta_{jk}}{v_k\sin\delta_k} & -\frac{\cos\beta_{ik}}{v_k\sin\delta_k} \end{array}\right]\left[\begin{array}{c} \bar{v}_k\left(t\right) \\ \bar{\omega}_k\left(t\right) \end{array}\right], \tag{5.24}$$

we obtain

$$\begin{aligned} \dot{x}\left(t\right) = {} & Ax\left(t\right)+\bar{g}\left(x_{*,1},v_k\right)\bar{u}_k\left(t\right)+g_0\left(x,v_k\right) \\ & +g_{v,2}\left(x\right)V_{k,2}\left(t\right)+g_{v,1}\left(x\right)V_{k,1}\left(t\right), \end{aligned} \tag{5.25}$$

with $\bar{u}_k\left(t\right)=\left[\bar{v}_k\left(t\right)\ \bar{\omega}_k\left(t\right)\right]^T$ and

$$\bar{g}\left(x_{*,1},v_k\right):=\left[\begin{array}{cc} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ -\frac{\cos\delta_k}{x_{1,1}\sin\delta_k} & \frac{1}{x_{1,1}\sin\delta_k} \\ 0 & 0 \\ -\frac{1}{x_{2,1}\sin\delta_k} & \frac{\cos\delta_k}{x_{2,1}\sin\delta_k} \\ -\frac{\cos\beta_{jk}}{v_k\sin\delta_k} & \frac{\cos\beta_{ik}}{v_k\sin\delta_k} \\ -\frac{\cos\beta_{jk}}{v_k\sin\delta_k} & \frac{\cos\beta_{ik}}{v_k\sin\delta_k} \end{array}\right].$$

Remark 5.1.4 also applies to this case.

5.1.2.2 Second order two-leader-follower model with $d \neq 0$

Let $x(t) \in \mathcal{M} \subseteq \mathbb{R}^2_{>0} \times \mathbb{S}^4 \times \mathbb{R}^6$ be defined as

$$\begin{aligned} x(t) &:= [x_{1,1},\, x_{1,2},\, x_{2,1},\, x_{2,2},\, x_{3,1},\, x_{3,2},\, x_{4,1},\, x_{4,2},\, x_{5,1},\, x_{5,2},\, x_{6,1},\, x_{6,2}]^T \\ &= \left[\ell_{ik},\, \dot{\ell}_{ik},\, \ell_{jk},\, \dot{\ell}_{jk},\, \alpha_{ik},\, \dot{\alpha}_{ik},\, \alpha_{jk},\, \dot{\alpha}_{jk},\, \theta_{ik},\, \dot{\theta}_{ik},\, \theta_{jk},\, \dot{\theta}_{jk}\right]^T \end{aligned}$$

As in the previous case, $x_i = x_{i,1}$, $i = 1, \ldots, 6$, coincide with the definition of first order model (5.18). Taking time derivative of (5.19), we obtain

$$\begin{aligned} \dot{x}(t) = Ax(t) &+ g(x_{*,1})\,\dot{u}_k(t) + g_0(x, u_k) \\ &+ g_{v,1}(x)\,V_{k,1}(t) + g_{v,2}(x_{*,1})\,V_{k,2}(t), \end{aligned} \tag{5.26}$$

with $\dot{u}_k(t) := \begin{bmatrix} \dot{v}_k(t) \\ \dot{\omega}_k(t) \end{bmatrix}$,

$$g_0(x, u_k) := \left[g_0^1(x, u_k),\, g_0^2(x, u_k)\right] \begin{bmatrix} v_k(t) \\ \omega_k(t) \end{bmatrix},$$

$$g_0^1(x, u_k) := \begin{bmatrix} 0 \\ (x_{3,2} + x_{5,2}) \sin\beta_{ik} \\ 0 \\ (x_{4,2} + x_{6,2}) \sin\beta_{jk} \\ 0 \\ \frac{x_{1,1}(x_{3,2}+x_{5,2})\cos\beta_{ik} - x_{1,2}\sin\beta_{ik}}{x_{1,1}^2} \\ 0 \\ \frac{x_{2,1}(x_{4,2}+x_{6,2})\cos\beta_{jk} - x_{2,2}\sin\beta_{jk}}{x_{2,1}^2} \\ 0_{4\times 1} \end{bmatrix},$$

$$g_0^2\left(x,u_k\right):=\begin{bmatrix}0\\-d\left(x_{3,2}+x_{5,2}\right)\cos\beta_{ik}\\0\\-d\left(x_{4,2}+x_{6,2}\right)\cos\beta_{jk}\\0\\\frac{dx_{1,1}\left(x_{3,2}+x_{5,2}\right)\sin\beta_{ik}+dx_{1,2}\cos\beta_{ik}}{x_{1,1}^2}\\0\\\frac{dx_{2,1}\left(x_{4,2}+x_{6,2}\right)\sin\beta_{jk}+dx_{2,2}\cos\beta_{jk}}{x_{2,1}^2}\\0_{4\times1}\end{bmatrix},$$

$$g\left(x_{*,1}\right):=\begin{bmatrix}0&0\\-\cos\beta_{ik}&-d\sin\beta_{ik}\\0&0\\-\cos\beta_{jk}&-d\sin\beta_{jk}\\0&0\\\frac{\sin\beta_{ik}}{x_{1,1}}&-\frac{d\cos\beta_{ik}}{x_{1,1}}\\0&0\\\frac{\sin\beta_{jk}}{x_{2,1}}&-\frac{d\cos\beta_{jk}}{x_{2,1}}\\0&0\\0&-1\\0&0\\0&-1\end{bmatrix},$$

$$g_{v,1}\left(x\right):=\begin{bmatrix}0&&0_{3\times1}&\\-x_{3,2}\sin x_{3,1}&&-x_{4,2}\sin x_{4,1}&\\0_{3\times1}&0_{12\times1}&0_{3\times1}&0_{12\times1}\\\frac{x_{1,2}\sin x_{3,1}-x_{1,1}x_{3,2}\cos x_{3,1}}{x_{1,1}^2}&&\frac{x_{2,2}\sin x_{4,1}-x_{2,1}x_{4,2}\cos x_{4,1}}{x_{2,1}^2}&\\0_{6\times1}&&0_{4\times1}&\end{bmatrix},$$

$$g_{v,2}\left(x\right):=\begin{bmatrix}0&0_{5\times1}&0_{3\times1}&\\\cos x_{3,1}&-1&\cos x_{4,1}&0_{7\times1}\\0_{3\times1}&0_{3\times1}&0_{3\times1}&-1\\-\frac{\sin x_{3,1}}{x_{1,1}}&1&-\frac{\sin x_{4,1}}{x_{2,1}}&0_{3\times1}\\0_{6\times1}&0_{2\times1}&0_{4\times1}&1\end{bmatrix},$$

and $V_{k,1}(t)$, $V_{k,2}(t)$ are defined by (5.23).

Using a control transformation similar to (5.20), that is,

$$\begin{bmatrix} \dot{v}_k(t) \\ \dot{\omega}_k(t) \end{bmatrix} = \frac{1}{\sin\delta_k} \begin{bmatrix} -\sin\beta_{jk} & \sin\beta_{ik} \\ \frac{\cos\beta_{jk}}{d} & -\frac{\cos\beta_{ik}}{d} \end{bmatrix} \begin{bmatrix} \bar{v}_k(t) \\ \bar{\omega}_k(t) \end{bmatrix}, \tag{5.27}$$

we obtain

$$\begin{aligned} \dot{x}(t) = {} & Ax(t) + \bar{g}(x_{*,1})\,\bar{u}_k(t) + g_0(x, u_k) \\ & + g_{v,1}(x)\,V_{k,1}(t) + g_{v,2}(x_{*,1})\,V_{k,2}(t), \end{aligned} \tag{5.28}$$

with $\bar{u}_k(t) = [\bar{v}_k(t)\ \bar{\omega}_k(t)]^T$ and

$$\bar{g}(x_{*,1}) := \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ -\frac{\cos\delta_k}{x_{1,1}\sin\delta_k} & \frac{1}{x_{1,1}\sin\delta_k} \\ 0 & 0 \\ -\frac{1}{x_{2,1}\sin\delta_k} & \frac{\cos\delta_k}{x_{2,1}\sin\delta_k} \\ 0 & 0 \\ -\frac{\cos\beta_{jk}}{d\sin\delta_k} & \frac{\cos\beta_{ik}}{d\sin\delta_k} \\ 0 & 0 \\ -\frac{\cos\beta_{jk}}{d\sin\delta_k} & \frac{\cos\beta_{ik}}{d\sin\delta_k} \end{bmatrix}.$$

5.1.3 Generic model formulation

Analyzing equations (5.14), (5.17), (5.25), and (5.28), a generic model can be designed. Therefore, with adequate function vector definitions, it is possible to proof stability properties and design controllers for the leader-follower and two leader-follower problems using an unified model. Let $x \in \mathcal{M}$ be adequately defined according to each case. Then

$$\dot{x}(t) = Ax(t) + \bar{g}(x_{*,1}, v_k)\,\bar{u}_k(t) + g_0(x, u_k) + g_v(x)\,V_k(t), \tag{5.29}$$

- For the second order leader-follower model with $d = 0$:

$$\begin{aligned} g_0(x, u_k) &= g_0(x, v_k), \\ g_v(x) &= [g_{v,1}(x,\ v_k),\ g_{v,2}(x_{*,1})], \\ V_k(t) &= \left[u_i^T(t),\ \dot{u}_i^T(t)\right]^T. \end{aligned}$$

- For the second order leader-follower model with $d \neq 0$:

$$\begin{aligned} \bar{g}(x_{*,1}, v_k) &= \bar{g}(x_{*,1}), \\ V_k(t) &= \dot{u}_i(t). \end{aligned}$$

- For the second order two-leader-follower model with $d = 0$:

$$\begin{aligned} g_0(x, u_k) &= g_0(x, v_k) \\ g_v(x) &= [g_{v,1}(x),\ g_{v,2}(x)], \\ V_k(t) &= \left[V_{k,1}^T(t),\ V_{k,2}^T(t)\right]^T. \end{aligned}$$

- For the second order two-leader-follower model with $d \neq 0$:

$$\begin{aligned} \bar{g}(x_{*,1}, v_k) &= \bar{g}(x_{*,1}), \\ g_0(x, u_k) &= g_0(x, u_k), \\ g_v(x) &= [g_{v,1}(x),\ g_{v,2}(x_{*,1})], \\ V_k(t) &= \left[V_{k,1}^T(t),\ V_{k,2}^T(t)\right]^T. \end{aligned}$$

5.2 The formation control problem

5.2.1 Leader-follower control problem

As mentioned, leader-follower coordination algorithms are based on the relative distance and bearing between a robot k and its *leader* i. Let $s_k(t) \in \mathcal{S} \subseteq \mathbb{R}_{\geq 0} \times \mathbb{S}$ denote the node specification for robot k, then

$$s_k(t) = [\ell_{ik}(t), \alpha_{ik}(t)]^T. \tag{5.30}$$

Let the *tracking error* $e_k(t) \in \mathbb{R} \times \mathbb{S}$ be defined as

$$e_k(t) := s_k^d(t) - s_k(t), \tag{5.31}$$

where the superscript 'd' stands for *desired* value. Let the error in state space form be defined as

$$\begin{aligned} e_1(t) &= e_k(t), \\ e_2(t) &= \dot{e}_1(t) = \dot{e}_k(t). \end{aligned}$$

Taking first and second time derivatives of (5.31) and using the generic model (5.29), it follows

$$\begin{cases} \dot{e}_1(t) = \dot{s}_k^d(t) - \dot{s}_k(t) = e_2(t), \\ \dot{e}_2(t) = \ddot{s}_k^d(t) - \ddot{s}_k = \ddot{s}_k^d(t) - \bar{g}_0(x, u_k) - \bar{g}_v(x) V_k(t) - \bar{u}_k(t), \end{cases} \tag{5.32}$$

with

$$\bar{g}_0(x, u_k) := \begin{bmatrix} [g_0(x, u_k)]_2 \\ [g_0(x, u_k)]_4 \end{bmatrix},$$

and

$$\bar{g}_v(x) := \begin{bmatrix} [g_v(x)]_2 \\ [g_v(x)]_4 \end{bmatrix},$$

where $[\cdot]_i$ indicates the i-th row of a matrix or vector.

Assuming that the leader robot i is stably tracking some desired trajectory $u_i^d(t) := \left[v_i^d(t), \omega_i^d(t)\right]^T \in \mathbb{R}^2$, such that $u_i(t)$, $\dot{u}_i(t)$, $\ddot{u}_i(t) \in \mathcal{L}_\infty$, the *objective* is to design a suitable state-feedback decentralized (dynamic) controller, represented by $\bar{u}_k(t)$, that asymptotically regulates the output error $e_k(t)$ to an arbitrarily small neighborhood of the origin $\mathcal{B}(0; \varepsilon)$, while maintaining the output trajectory $s_k(t)$ for $t \geq 0$ within a given set $\mathcal{S}_k \subseteq \mathbb{R} \times \mathbb{S}$ of the form

$$\mathcal{S}_k = \left\{ \left(s_k^1, s_k^2\right) \in \mathcal{S}_k \subseteq \mathbb{R} \times \mathbb{S} : \underline{s}_k^1 < s_k^1 < \bar{s}_k^1, \underline{s}_k^2 < s_k^2 < \bar{s}_k^2 \right\},$$

where $\mathcal{B}(0; \varepsilon)$ is a ball of radius ε around the origin. For the problem to be well posed, we assume that $s_k(0) \in \mathcal{S}_k$ and $s_k^d(t) \in \mathcal{S}_k$, for all $t \geq 0$.

Remark 5.2.1. Usually $V_k(t)$ is unknown.

Remark 5.2.2. In the literature, the control specification for the leader-follower problem is given in terms of the relative bearings $\psi_{ik}(t)$, $\gamma_{ik}(t) \in \mathbb{S}$, with

$$\psi_{ik}(t) := \pi + \alpha_{ik}(t), \tag{5.33}$$

$$\gamma_{ik}(t) := \pi + \beta_{ik}(t), \tag{5.34}$$

as shown in Figure 5.1. However, given the equivalence between angles, we will use the original angles $\alpha_{ik}(t)$ and $\beta_{ik}(t)$ to define the desired position. The main reason for this choice is to avoid the discontinuity when the specification dictates pure trailing, that is, $\psi_{ik}^d = \pi$ or $\alpha_{ik}^d = 0$.

Remark 5.2.3. Considering the dynamical model of the robot (2.1), the equilibrium point for system (5.11) is given by

$$x^*(t) = \left[\ell_{ik}^d(t),\, 0,\, \alpha_{ik}^d(t),\, 0,\, 0,\, \dot{v}_k^d(t),\, \dot{\omega}_k^d(t)\right]^T,$$

whereas the equilibrium point for system (5.15) is given by

$$x^*(t) = \left[\ell_{ik}^d(t),\, 0,\, \alpha_{ik}^d(t),\, 0,\, 0,\, 0,\, \dot{v}_k^d(t),\, \dot{\omega}_k^d(t)\right]^T.$$

Some pathological situations are excluded, like the leader rotating around the follower with constant velocity.

5.2.2 Two-leader-follower control problem

The second type of coordination algorithms presented in this dissertation are based on the relative distances between the follower robot k and two leaders i and j. Let $s_k(t) \in \mathbb{R}^2_{\geq 0}$ denote the specification of robot k, then

$$s_k(t) := \left[\ell_{ik}(t),\, \ell_{jk}(t)\right]^T. \tag{5.35}$$

Let the *tracking error* $e_k(t) \in \mathbb{R}^2$ be defined as

$$e_k(t) := s_k^d(t) - s_k(t), \tag{5.36}$$

and the error in state space be defined as

$$e_1(t) = e_k(t),$$
$$e_2(t) = \dot{e}_1(t) = \dot{e}_k(t).$$

Following the same steps as in the leader-follower case, it follows

$$\begin{cases} \dot{e}_1(t) = \dot{s}_k^d(t) - \dot{s}_k(t) = e_2(t), \\ \dot{e}_2(t) = \ddot{s}_k^d(t) - \ddot{s}_k = \ddot{s}_k^d(t) - \bar{g}_0(x, u_k) - \bar{g}_v(x) V_k(t) - \bar{u}_k(t), \end{cases} \tag{5.37}$$

with

$$\bar{g}_0(x, u_k) := \begin{bmatrix} [g_0(x, u_k)]_2 \\ [g_0(x, u_k)]_4 \end{bmatrix},$$

and

$$\bar{g}_v(x) := \begin{bmatrix} [g_v(x)]_2 \\ [g_v(x)]_4 \end{bmatrix}.$$

Assuming that the leader robot i is stably tracking some desired trajectory $u_i^d(t) := \left[v_i^d(t), \omega_i^d(t)\right]^T \in \mathbb{R}^2$, such that $u_i(t)$, $\dot{u}_i(t)$, $\ddot{u}_i(t) \in \mathcal{L}_\infty$, and the trajectory of leader robot j is such that $u_j(t)$, $\dot{u}_j(t)$, $\ddot{u}_j(t) \in \mathcal{L}_\infty$, due to the stability properties of the leader-follower algorithm (proved in Section 5.3), the *objective* is to design a suitable state-feedback decentralized (dynamic) controller, represented by $\bar{u}_k(t)$, that asymptotically regulates the output error $e_k(t)$ to an arbitrarily small neighborhood of the origin $\mathcal{B}(0; \varepsilon)$, while maintaining the output trajectory $s_k(t)$ for $t \geq 0$ within a given set $\mathcal{S}_k \subseteq \mathbb{R}^2$ of the form

$$\mathcal{S}_k = \left\{ \left(s_k^1, s_k^2\right) \in \mathcal{S}_k \subseteq \mathbb{R}^2 : \underline{s}_k^1 < s_k^1 < \bar{s}_k^1, \underline{s}_k^2 < s_k^2 < \bar{s}_k^2 \right\},$$

where $\mathcal{B}(0; \varepsilon)$ is a ball of radius ε around the origin. For the problem to be well posed, we assume that $s_k(0) \in \mathcal{S}_k$ and $s_k^d(t) \in \mathcal{S}_k$, for all $t \geq 0$.

As can be seen, the similarities in the structure of equations (5.32) and (5.37) and in problem formulations allow an unified design of different controllers.

Remark 5.2.4. Considering the dynamical model of the robot (2.1), the equilibrium point for system (5.22) is given by

$$x^*(t) = \left[\ell_{ik}^d,\, 0,\, \ell_{jk}^d,\, 0,\, \alpha_{ik},\, 0,\, \alpha_{ik} + \theta_{ij} + \delta_k^d,\, 0,\, \theta_i,\, 2\theta_k - \theta_i,\, \dot{v}_k^d,\, \dot{\omega}_k^d\right]^T,$$

whereas the equilibrium point for system (5.26) is given by

$$x^*(t) = \left[\ell_{ik}^d,\, 0,\, \ell_{jk}^d,\, 0,\, \alpha_{ik},\, 0,\, \alpha_{ik} + \theta_{ij} + \delta_k^d,\, 0,\, \theta_i,\, 0,\, 2\theta_k - \theta_i,\, 0,\, \dot{v}_k^d,\, \dot{\omega}_k^d\right]^T.$$

Some pathological situations are excluded, like the both leaders rotating around the follower.

5.2.3 Formation dynamics, communication and sensing models

Let $x_{*,1}(t)$ represent the measured variables, with $x_{*,1}(t) \in \mathbb{R} \times \mathcal{S}^2$ for the leader-follower problem and $x_{*,1}(t) \in \mathbb{R}^2 \times \mathcal{S}^4$ for the two leader-follower problem. Let $V_k(t)$ be defined as the external, possibly unknown, input function of leader(s) velocities.

It should be noted that there exist three information flow levels in decreasing order of measurement complexity for the controlled system given by equations (5.32) and (5.37): The first level, the most complex from the point of view of information flow, is given by the knowledge of the velocity and acceleration of the leader(s), the specification and its derivative, that is, $V_k(t)$ and $s_k(t)$ and $\dot{s}_k(t)$, respectively. This case, sketched in Figure 5.3 for the leader-follower case, can be solved by using a *partial state feedback controller* (PSFB) [121]. Figure 5.4 shows an PSFB controller including an auxiliary variable-structure controller to reject uncertainties and input noise.

The second information flow level requires the knowledge of the specification and its derivative, $s_k(t)$ and $\dot{s}_k(t)$, respectively, but considers the velocity and acceleration of the leader as perturbations, as it is shown in Figure 5.5. This case can be solved using a *robust state feedback controller* (RSFB). That is, the controller using a variable structure auxiliary input.

Finally, the third, and simplest level from the information flow point of view, only requires the knowledge of the specification $s_k(t)$, as depicted in Figure 5.6. This case can be solved using an *output feedback controller* (OFB) and a high-gain observer.

Remark 5.2.5. The last controller can be used for pursuit applications, because it does not meet any information transmitted to the pursuer (or follower).

In Section 5.3, we describe, analyze, and compare each of the controllers mentioned previously.

5.3 Formation control algorithms

This Section presents the state feedback controllers for the three information flow levels described earlier.

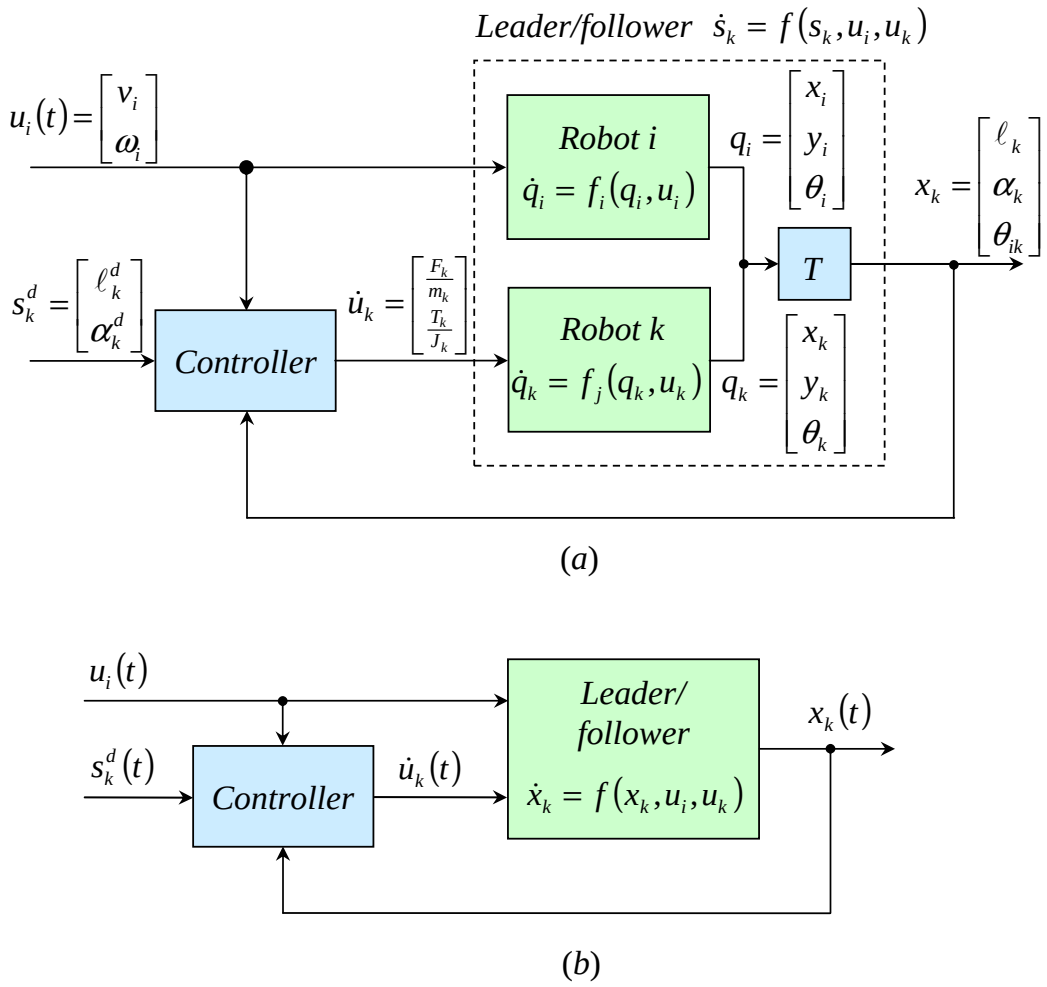


Figure 5.3: Leader-follower dynamic system: (a) complete system, (b) compact form.

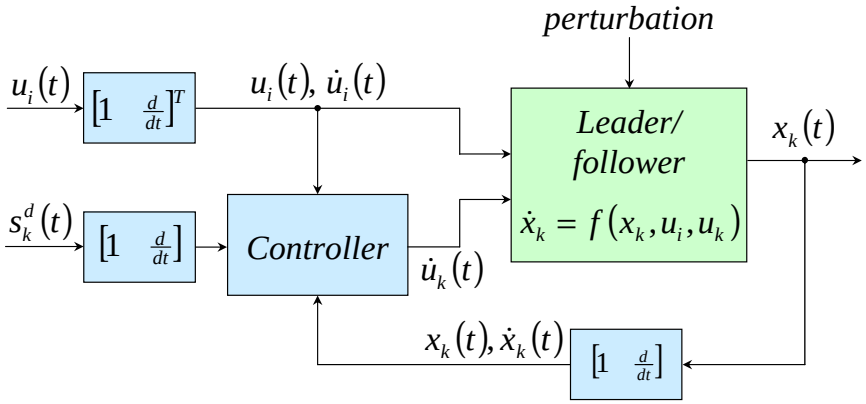


Figure 5.4: Leader-follower (PSFB controller).

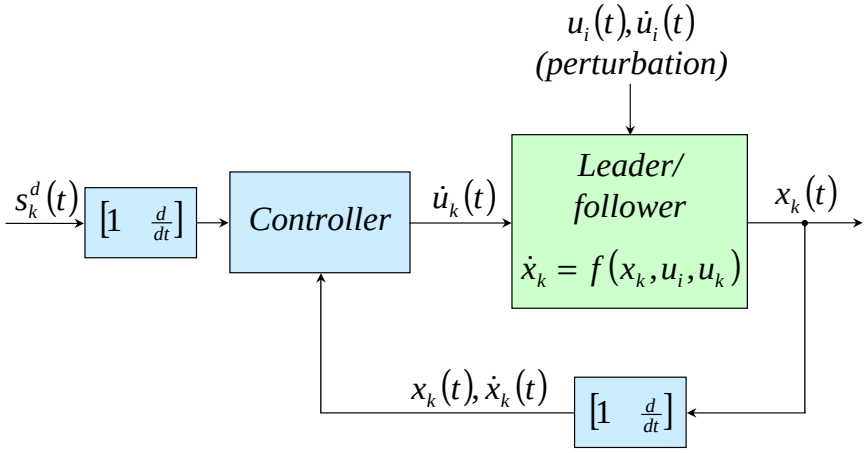


Figure 5.5: Leader-follower (RSFB controller).

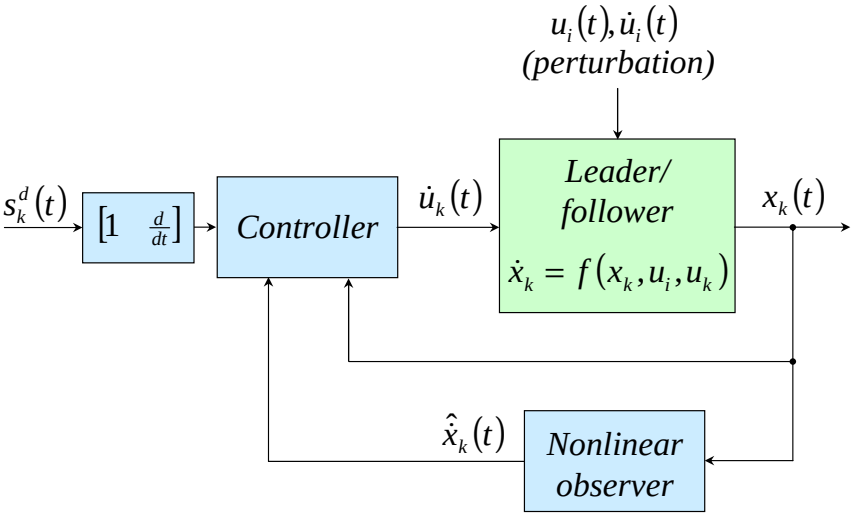


Figure 5.6: Leader-follower (OFB controller).

5.3.1 Partial state feedback formation control (PSFB)

The first control law is equivalent to the one originally presented in [34] for first-order dynamics. It assumes full knowledge of all states of robot j and its leader i in (5.32) and (5.37). The control input is given by

$$\bar{u}_k(t) = \ddot{s}_k^d(t) + 2K\dot{e}_k(t) + K^2 e_k(t) - \bar{g}_0(x) - \bar{g}_v(x) V_k(t), \tag{5.38}$$

where $K = \text{diag}(k_1, k_2)$, $k_1,\ k_2 \in \mathbb{R}_{>0}$ are positive gain constants. Replacing (5.38) into (5.32) or (5.37), the error dynamics of the closed-loop system become

$$\ddot{e}_k(t) + 2K\dot{e}_k(t) + K^2 e_k(t) = 0. \tag{5.39}$$

Therefore, because the characteristic equation (5.39) can be made Hurwitz and critically damped by an adequate choice of k_1 and k_2, the error $e_k(t)$ tends exponentially to zero as $t \to \infty$.

5.3.2 Robust state feedback formation control (RSFB)

This controller assumes that leaders' velocities and accelerations, $V_k(t)$, are unknown. However, the derivative of edge specification $\dot{s}_k(t)$ is perfectly known by measurement or estimation. Then, let the RSFB control law $\bar{u}_k(t)$ be designed as

$$\bar{u}_k(t) = \ddot{s}_k^d(t) + 2K\dot{e}_k(t) + K^2 e_k(t) - \bar{g}_0(x_k) + \bar{u}_k^c(t), \tag{5.40}$$

where $\bar{u}_k^c(t) \in \mathbb{R}^2$ is an auxiliary control law defined as

$$\bar{u}_k^c(t) := \beta\, \text{sign}(e_k(t)), \tag{5.41}$$

with $\beta = \text{diag}(\beta_1,\ \beta_2) \in \mathbb{R}_{>0}^{2\times 2}$, a positive control gain matrix. The function $\text{sign}(\cdot)$ is defined as usual as

$$\text{sign}(x) := \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x = 0 \\ -1, & \text{if } x < 0. \end{cases}$$

The following theorem states the stability of the closed-loop system.

Theorem 5.3.1. *Control law (5.40) and auxiliary control law (5.41) ensure that the system (5.29) is asymptotically stable, all its variables are bounded, and the tracking*

error and its derivative tend to zero, that is

$$e_k(t),\ \dot{e}_k(t) \to 0 \ \textit{as}\ t \to \infty. \tag{5.42}$$

Proof. See Section B.1 in Appendix B. □

Remark 5.3.2. Alternatively, we use the function

$$\bar{u}_k^c(t) := \beta \tanh\left(e_k(t)/\epsilon_0\right), \tag{5.43}$$

to avoid excessive chattering. Although all the stability proofs are given in terms of (5.41), systems (5.32) or (5.37) are ISS stable with respect to $\bar{u}_k^c(t)$. Therefore, it is possible to attract the system to the stability region by choosing correctly K, β, and ϵ_0.

5.3.3 Output feedback formation control algorithm (OFB)

In the previous Subsection, a robust state feedback control law was designed that did not require the knowledge of the leader's speed and acceleration, but the knowledge of $\dot{s}_k(t)$, or $x_{*,2}(t) = \dot{x}_{*,1}(t)$, see equation (5.40). In this subsection, the RFSB controller is extended to an output feedback (OFB) control algorithm by using a high-gain observer (HGO) to avoid this drawback [1, 2, 69, 70, 84].

Let the unknown vector $\dot{s}_k(t)$ be estimated through $x_{*,2}(t)$ using the HGO

$$\dot{\hat{x}}(t) = A\hat{x}(t) + \bar{g}(x_{*,1}, v_k)\,\bar{u}_k(t) + g_0(\hat{x}, u_k) + H\left(x_{*,1}(t) - \hat{x}_{*,1}(t)\right), \tag{5.44}$$

with $H = \text{blockdiag}(H_1, H_2, \ldots)$ depends on the problem,

$$H_i = \begin{bmatrix} \alpha_1/\varepsilon \\ \alpha_2/\varepsilon^2 \end{bmatrix},$$

where $\alpha_1,\ \alpha_2 \in \mathbb{R}_{>0}$ are constants, called HGO gains, and $\varepsilon \in \mathbb{R}_{>0}$ is the HGO constant. The HGO gains are chosen such that the roots of

$$s^2 + \alpha_1 s + \alpha_2 = 0,$$

have negative real parts. The HGO constant has to be designed such that the estimated variable $\hat{x}_{*,2}(t)$ converges to the real values $x_{*,2}(t)$ fast enough to stabilize the whole system. Let the scaled estimation error $\eta(t)$ be defined as $\eta(t) :=$

$[\eta_{1,1},\, \eta_{1,2},\, \eta_{2,1},\, \eta_{2,2},\, \ldots]^T$, with

$$\begin{cases} \eta_{*,1}(t) = \dfrac{1}{\varepsilon}\left(x_{*,1}(t) - \hat{x}_{*,1}(t)\right), \\ \eta_{*,2}(t) = x_{*,2}(t) - \hat{x}_{*,2}(t). \end{cases} \tag{5.45}$$

Using (5.44) and (5.45), the dynamic observer error system is given by

$$\begin{cases} \varepsilon\dot{\eta}_{*,1}(t) = -\alpha_1\eta_{*,1}(t) + \eta_{*,22}(t), \\ \varepsilon\dot{\eta}_{*,2}(t) = -\alpha_2\eta_{*,1}(t) + \varepsilon B\left[g_0(x,u_k) - g_0(\hat{x},u_k) + g_v(x)\,V_k(t)\right], \end{cases} \tag{5.46}$$

where $B = \text{blockdiag}(B_1,\, B_2,\, \ldots)$ depends on the specific problem, with

$$B_i = [0\,1],\ i = 1, 2, \ldots.$$

The dynamics of the scaled error $\eta(t)$ can be written in compact form as

$$\varepsilon\dot{\eta}(t) = A_0\eta(t) + \varepsilon f(t), \tag{5.47}$$

where $A_0 = \text{blockdiag}(a_0,\, a_0,\, \ldots)$, with

$$a_0 = \begin{bmatrix} -\alpha_1 & 1 \\ -\alpha_2 & 0 \end{bmatrix},$$

and

$$f(t) = \begin{bmatrix} 0 \\ g_1(x,u_k)\,\eta_2 + Bg_v(x)\,V_k(t) \end{bmatrix},$$

with $\bar{g}_1(x,u_k)\,\eta_2 = B\left[g_0(x,u_k) - g_0(\hat{x},u_k)\right]$ depends on the problem. If $\varepsilon \to 0$ and $d\tau = \frac{1}{\varepsilon}dt$ in (5.47), then

$$\frac{d\eta(\tau)}{d\tau} = A_0\eta(\tau). \tag{5.48}$$

Equation (5.48) is called the *boundary layer system.*

Let $W(\eta)$ be a non-negative function for the boundary layer system defined as

$$W(\eta) = \eta^T P_0 \eta, \tag{5.49}$$

where P_0 is a positive definite matrix such that $P_0A_0 + A_0^T P_0 = -I$, then

$$\frac{dW}{d\tau} = \frac{d\eta^T}{d\tau}P_0\eta + \eta^T P_0\frac{d\eta}{d\tau} = -\left\|\eta\right\|^2 \leq 0. \tag{5.50}$$

Consequently, the boundary layer system has the properties

$$\begin{aligned} \lambda_{\min}(P_0)\|\eta\|^2 \leq W(\eta) \leq \lambda_{\max}(P_0)\|\eta\|^2, \\ \dot{W} = \frac{\partial W}{\partial \eta}\dot{\eta} \leq -\lambda_\omega \|\eta\|^2, \ 0 \leq \lambda_\omega \leq 1, \\ \left\| \frac{\partial W}{\partial \eta} \right\| \leq 2\|P_0\|\|\eta\|, \ \|P_0\| = \lambda_{\max}(P_0). \end{aligned} \tag{5.51}$$

From (5.51), it is clear that the origin $\eta(t) = 0$ is a globally exponentially stable equilibrium of (5.48).

Theorem 5.3.3. *Let the control law be given by*

$$\bar{u}_k(t) = \ddot{s}_k^d(t) + 2K\left(\dot{s}_k^d(t) - \hat{s}_2(t)\right) + K^2 e_k(t) - \bar{g}_0(\hat{x}, u_k) + \bar{u}_k^c(t), \tag{5.52}$$

with the components of $\hat{s}_2(t)$ extracted from $\hat{x}_{*,2}(t)$. Controller (5.52) with the auxiliary control law (5.41) asymptotically regulates the output error $e_k(t)$ of the combined system (5.29) with the observer (5.47) to an arbitrarily small neighborhood of the origin $\mathcal{B}(0;\varepsilon)$.

Proof. See Chapter B.3 in Appendix B. □

Remark 5.3.4. The RSFB controller (5.40) designed in Subsection 5.3.2 is globally asymptotically stable. Theoretically, the observer error will not cause the system to become unstable using the OFB controller (5.52). However, the solution for the error dynamics $\eta(t)$ (5.46) contains terms of the form $\frac{1}{\varepsilon}e^{-\omega t/\varepsilon}$, for some $\omega > 0$. Note that $\eta(t)$ can be very large if ε is small enough, because its amplitude is $O\left(\frac{1}{\varepsilon}\right)$. Then, the new feedback control design based on HGO has to be saturated to avoid the *peaking phenomenon* due to $\eta(t)$ and prevent over-exceeding the control strength.

5.4 Backstepping

For $d = 0$, control laws (5.38), (5.40) and (5.52) give the desired input vector

$$\varpi_k^d(t) = \begin{bmatrix} \dot{v}_k^d(t) \\ \omega_k^d(t) \end{bmatrix} = T_c(x_{*,1}, v_k)\,\bar{u}_k(t), \tag{5.53}$$

with

$$T_c(x_{*,1}, v_k) = \begin{bmatrix} -\cos\beta_{ik} & x_{1,1}\sin\beta_{ik} \\ -\frac{1}{v_k}\sin\beta_{ik} & -\frac{x_{1,1}}{v_k}\cos\beta_{ik} \end{bmatrix},$$

for the second order leader-follower model, and

$$T_c\left(x_{*,1}, v_k\right) := \frac{1}{\sin\delta_k}\begin{bmatrix} -\sin\beta_{jk} & \sin\beta_{ik} \\ \frac{1}{v_k}\cos\beta_{jk} & -\frac{1}{v_k}\cos\beta_{ik} \end{bmatrix},$$

for the second order two-leader-follower model.

However, while $\dot{v}_k^d\left(t\right)$ is given directly, we need the derivative of $\omega_k^d\left(t\right)$ to implement the control law of the dynamical system (2.5). This derivative is computed using the well-known backstepping technique [69, 130, 131] as follows: Let us add and subtract $\bar{g}_{*,2}\left(x_{*,1},\, v_k\right)\omega_k^d\left(t\right)$ to (5.32), with

$$\bar{g}_{*,2}\left(x_{*,1},\, v_k\right) = \begin{bmatrix} \bar{g}_{12}\left(x_{*,1},\, v_k\right) \\ \bar{g}_{22}\left(x_{*,1},\, v_k\right) \end{bmatrix}$$

and $\omega_k^d\left(t\right)$ is computed in equation (5.53). Then, using the PSFB controller (5.38), the error dynamics is

$$\ddot{e}_k\left(t\right) = -2K\dot{e}_k\left(t\right) - K^2 e_k\left(t\right) - \bar{g}_{*,2}\left(x_{*,1},\, v_k\right)\left[\omega_k\left(t\right) - \omega_k^d\left(t\right)\right].$$

Using the RSFB controller (5.40), the error dynamics is

$$\begin{aligned}\ddot{e}_k\left(t\right) = &-2K\dot{e}_k\left(t\right) - K^2 e_k\left(t\right) - \bar{u}_k^c\left(t\right) \\ &- \bar{g}_v\left(x, v_k\right)V_k\left(t\right) - \bar{g}_{*,2}\left(x_{*,1},\, v_k\right)\left[\omega_k\left(t\right) - \omega_k^d\left(t\right)\right].\end{aligned}$$

Finally, using the OFB controller (5.52), the error dynamics is

$$\begin{aligned}\ddot{e}_k\left(t\right) = &-2K\left(\dot{s}_k^d\left(t\right) - \hat{s}_2\left(t\right)\right) - K^2 e_k\left(t\right) - \left[\bar{g}_0\left(x,\, v_k\right) - \bar{g}_0\left(\hat{x}, u_k\right)\right] \\ &- \bar{g}_v\left(x, v_k\right)V_k\left(t\right) - \bar{u}_k^c\left(t\right) - \bar{g}_{*,2}\left(x_{*,1},\, v_k\right)\left[\omega_k\left(t\right) - \omega_k^d\left(t\right)\right].\end{aligned}$$

Following the design procedure described in [69] by adding the term

$$\frac{1}{2}\left[\omega_k\left(t\right) - \omega_k^d\left(t\right)\right]^2$$

to the original Lyapunov function, the system keeps its stability properties. Then, input $\dot{\omega}_k\left(t\right)$ can be implemented with

$$\dot{\omega}_k\left(t\right) = \dot{\omega}_k^d\left(t\right) - k_z\left[\omega_k\left(t\right) - \omega_k^d\left(t\right)\right] + r^T\left(t\right)\bar{g}_{*,2}\left(x_{*,1},\, v_k\right), \tag{5.54}$$

where $k_z \in \mathbb{R}_{>0}$ is a design constant. Term $\dot{\omega}_k^d(t)$ cannot computed in closed form in the OFB controller, because it depends on unknown variables. Therefore the HGO is extended with

$$\begin{cases} \dot{\hat{w}}_1(t) = \hat{s}_2(t) + \dfrac{\alpha_1}{\varepsilon}\left(\dot{\omega}_k^d(t) - \hat{w}_1(t)\right), \\ \dot{\hat{w}}_2(t) = \dfrac{\alpha_2}{\varepsilon^2}\left(\dot{\omega}_k^d(t) - \hat{w}_1(t)\right), \\ \hat{\dot{\omega}}_k^d(t) = \hat{w}_2, \end{cases} \tag{5.55}$$

to compute the derivative $\dot{\omega}_k^d(t)$.

5.5 Simulation results

In this Section, we present several simulations to validate and analyze the performance of the decentralized control algorithms designed herein. All simulations were written in C++ on a Linux platform using MPSLab[1], a motion planning, simulation, and virtual perception software library designed and programmed by the author of this dissertation [99, 100].

Figures 5.7 to 5.22 show simulation with the leader-follower and two-leader-follower controllers considering $d = 0$ and using backstepping for $\dot{\omega}_k(t)$. Figures 5.24 to 5.29 show simulations with the leader-follower and two-leader-follower controllers considering $d \neq 0$.

In figures 5.7 to 5.12 we show the results simulating a formation of three robots using the OFB leader-follower algorithm, with the leader following a straight line path with constant velocity $u_\ell(t) = [1.0\,\text{m/sec},\, 0.0\,\text{rad/sec}]^T$. Figures 5.13 to 5.18 show the results with the leader following a circular path with constant velocity $u_\ell(t) = [1.0\,\text{m/sec},\, 0.3\,\text{rad/sec}]^T$.

The robot initial positions are $q_\ell(0) = [0,\, 0,\, 0]^T$, $q_1(0) = [-2,\, -1,\, 0]^T$, and $q_2(0) = [-2,\, 1,\, 0]^T$, and their initial velocities are equal to 0.01. The controller parameters are $k_1 = k_2 = \beta_1 = \beta_2 = 5.0$, the sampling time is 10 msec, the edge specifications are $s_1^d = \left[2.0, -\frac{3}{4}\pi\right]^T$ and $s_2^d = \left[2.0, \frac{3}{4}\pi\right]^T$. As can be seen, follower robots start out of formation. Follower velocities are bounded by $|v_j(t)| \leq 1.6$ and $|\omega_j(t)| \leq 11.5$, $j = 1, 2, \ldots$

The decentralized controller is able to drive each robot to the desired relative distance and desired bearing angle in both cases. Figures 5.7 and 5.13 show robot trajectories using the OFB controller with $\varepsilon = 0.001$. Figures 5.8, 5.9, 5.14, and 5.15 show the decentralized control inputs and the tracking error of follower 1 using the

[1] http://orqueda.net/research.aspx

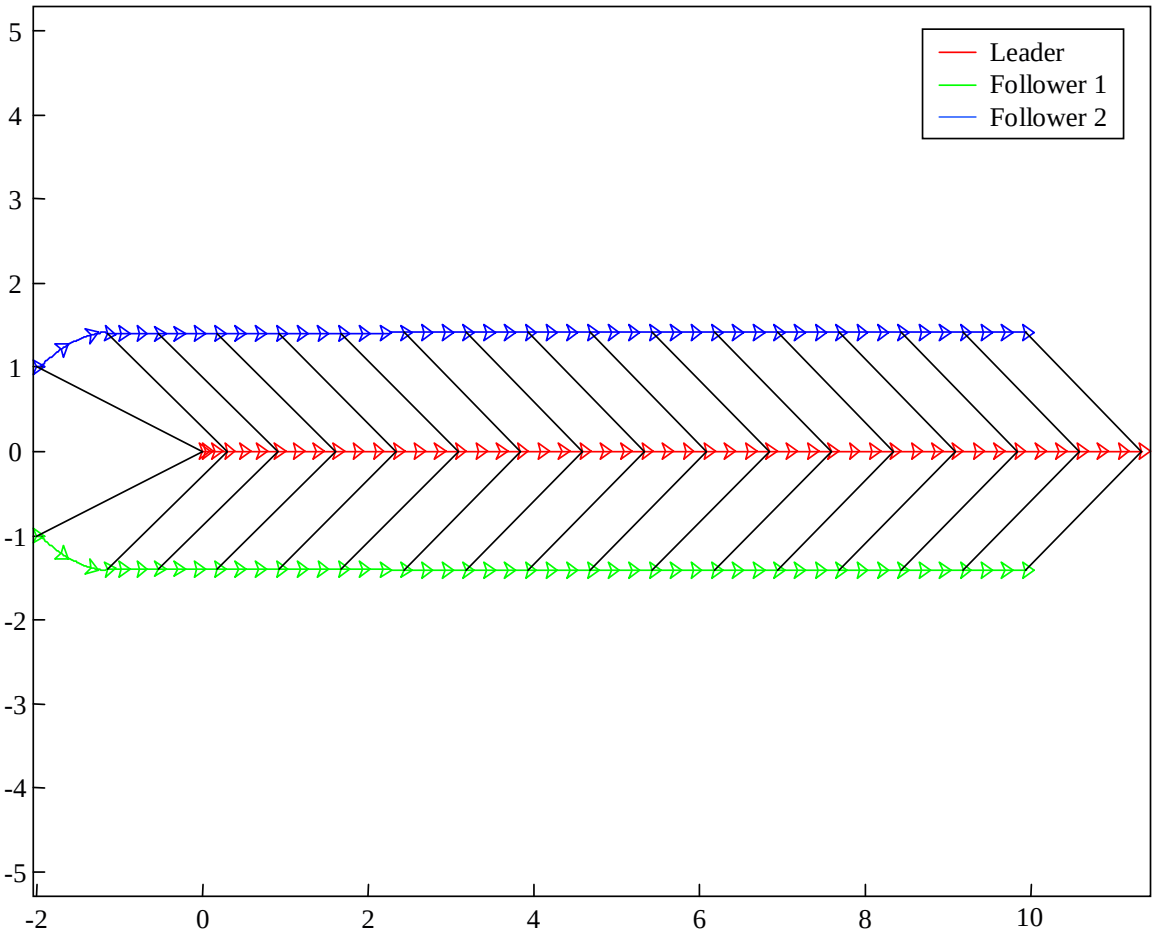


Figure 5.7: Robot trajectories for a straight line path and $d = 0$.

three controllers. It should be noted that not only the tracking error but also the control effort degrade with the loss of information. Notwithstanding, the behavior using the OFB controller is quite satisfactory.

Figures 5.10, 5.11, 5.16, and 5.17 illustrate the effects of the selection of the HGO constant ε on the control input and tracking error, respectively. As ε decreases, tracking errors decrease and control efforts increase during the transient response, when the observer have not converged.

Figures 5.12 and 5.18 show simulations with this three-robot formation in the 3D simulation environment MPSLab. As it can be seen, the results in the environment MPSLab are quite remarkable.

Figures 5.19 to 5.22 show a simulation with the OFB scheme with a 7-robot formation, when the leader follows a circular trajectory. The initial positions of the robots are $q_\ell(0) = [0, -5, 0]^T$, $q_1(0) = [-3, -8, 0]^T$, $q_2(0) = [-3, -2, 0]^T$, $q_3(0) = [-3, -10, 0]^T$, $q_4(0) = [-3, -6, 0]^T$, $q_5(0) = [-3, -4, 0]^T$, $q_6(0) = [-3, 0, 0]^T$, and

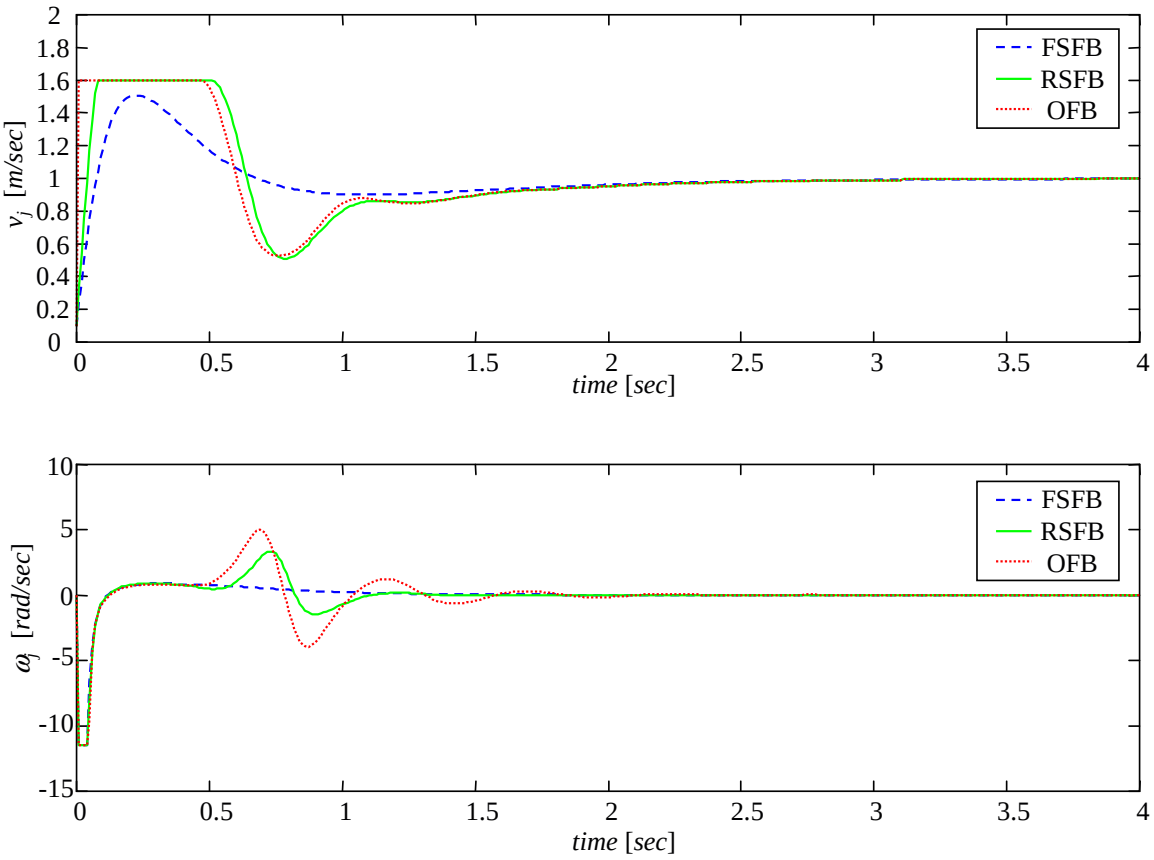


Figure 5.8: Velocity comparison between controllers (5.38), (5.40), and (5.52) for a straight line path and $d = 0$.

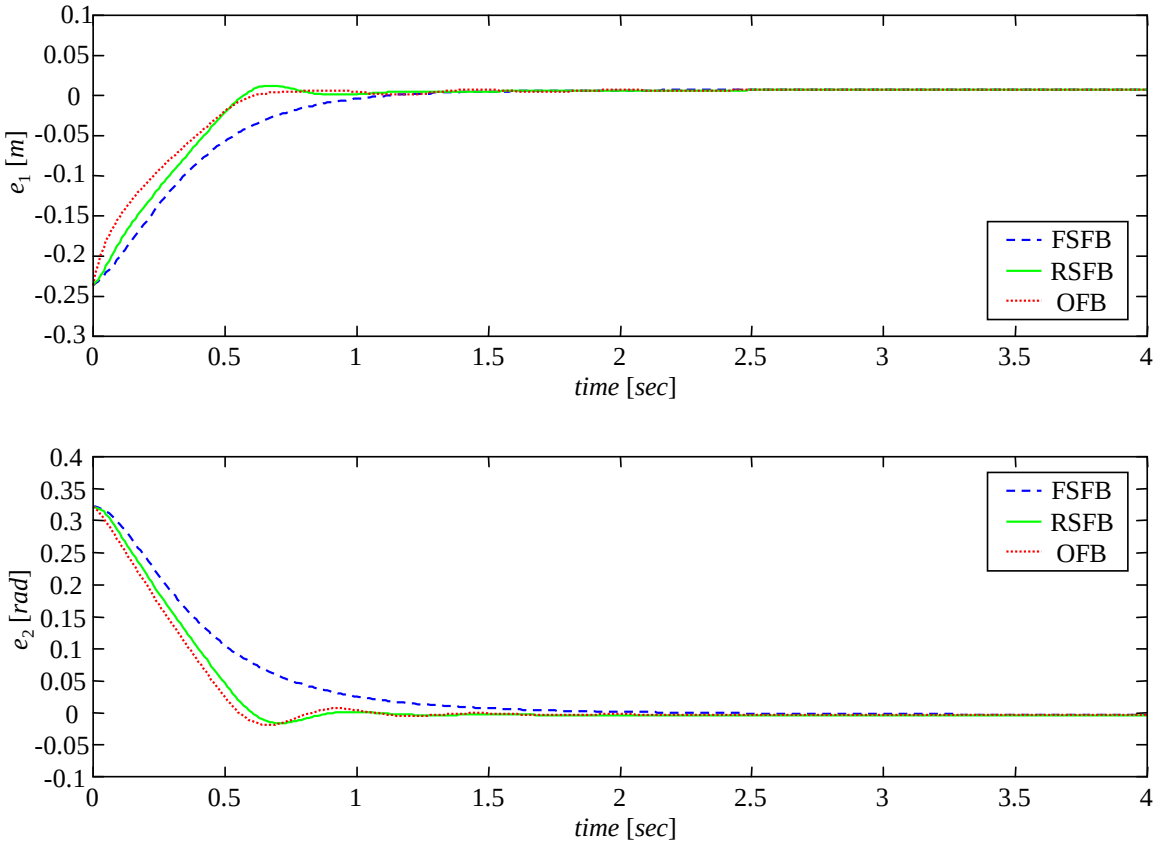


Figure 5.9: Error comparison between controllers (5.38), (5.40), and (5.52) for a straight line path and $d = 0$.

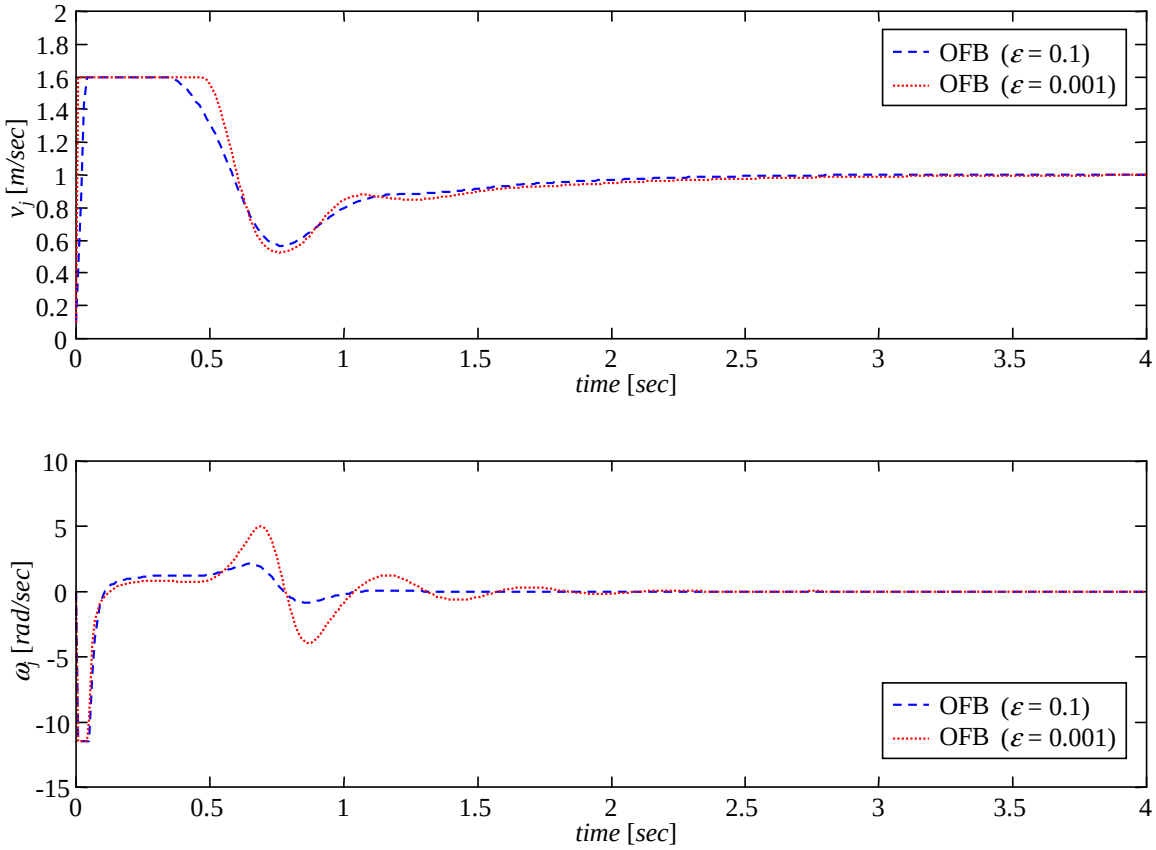


Figure 5.10: Control effort comparison with $\varepsilon = 0.1, 0.001$ for a straight line path and $d = 0$.

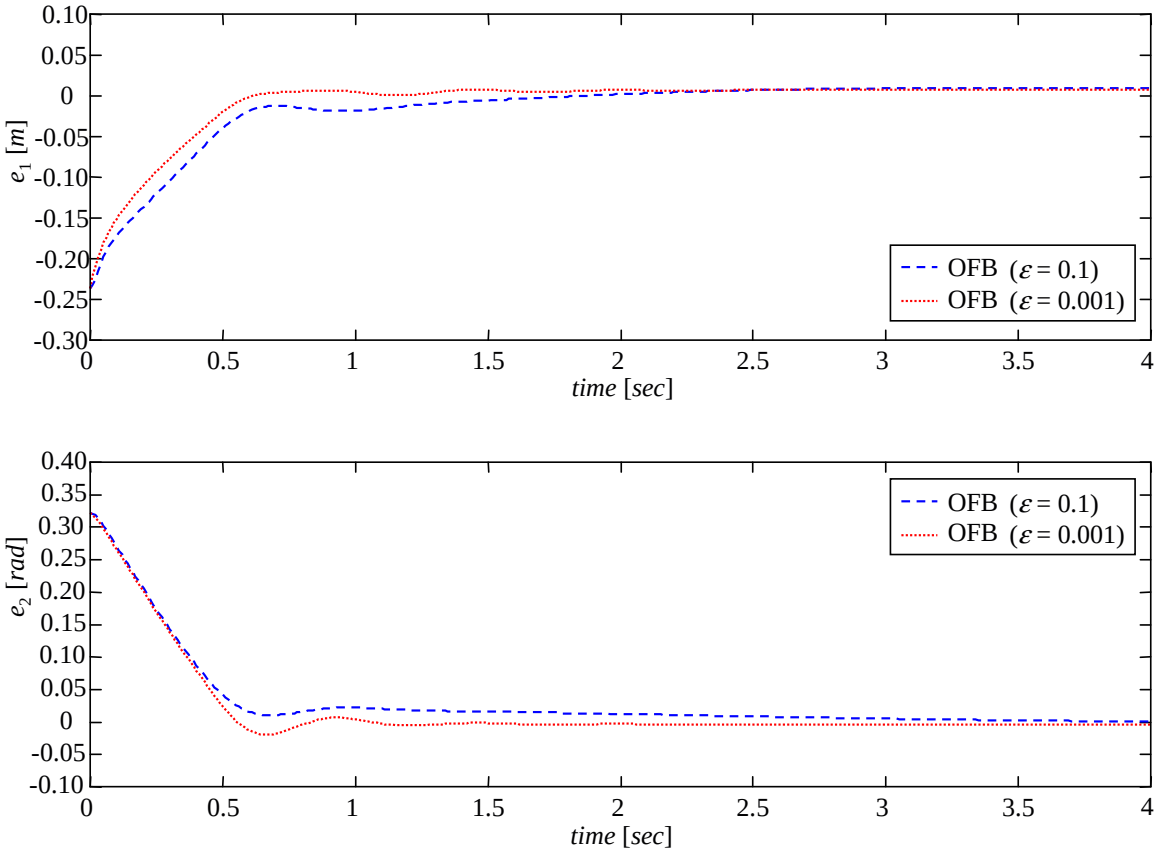


Figure 5.11: Output error comparison with $\varepsilon = 0.1$, 0.001 for a straight line path and $d = 0$.

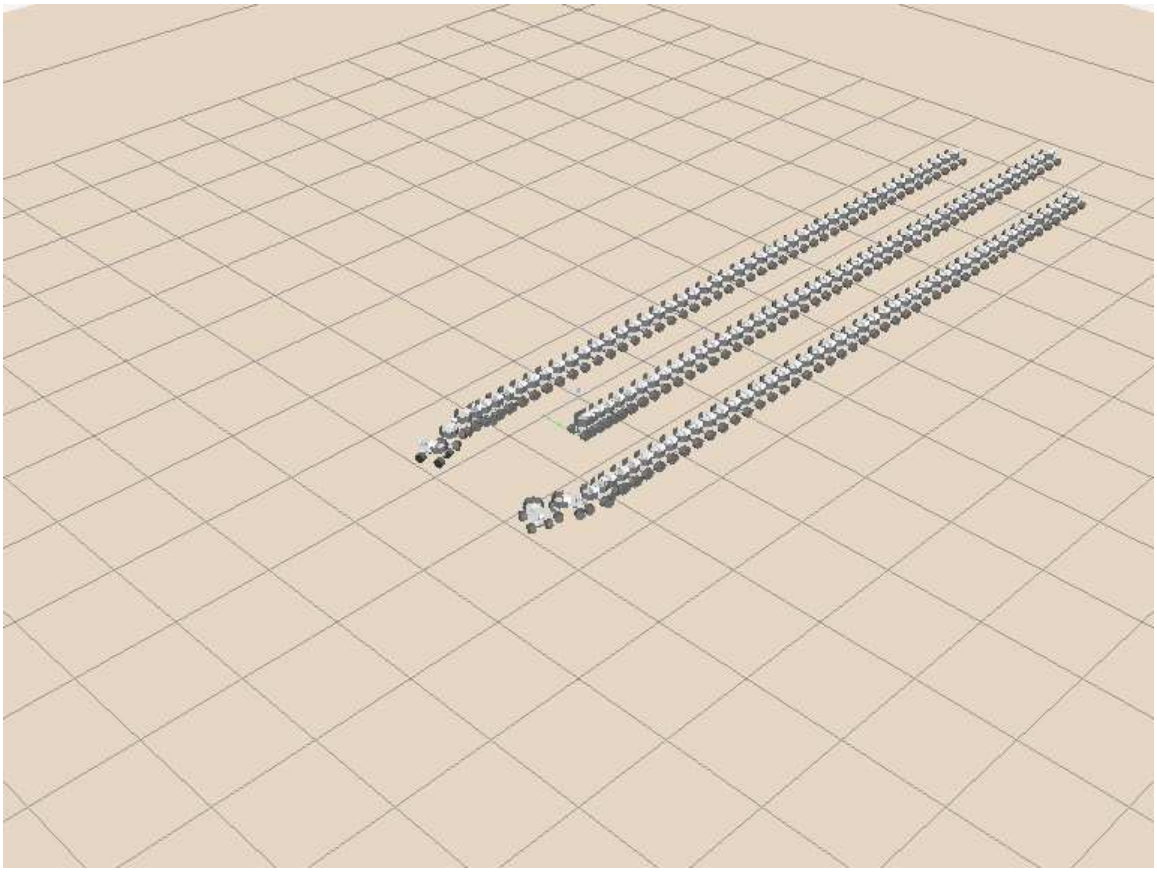


Figure 5.12: Snake view of a simulation with 3 robots for a straight line path and $d = 0$.

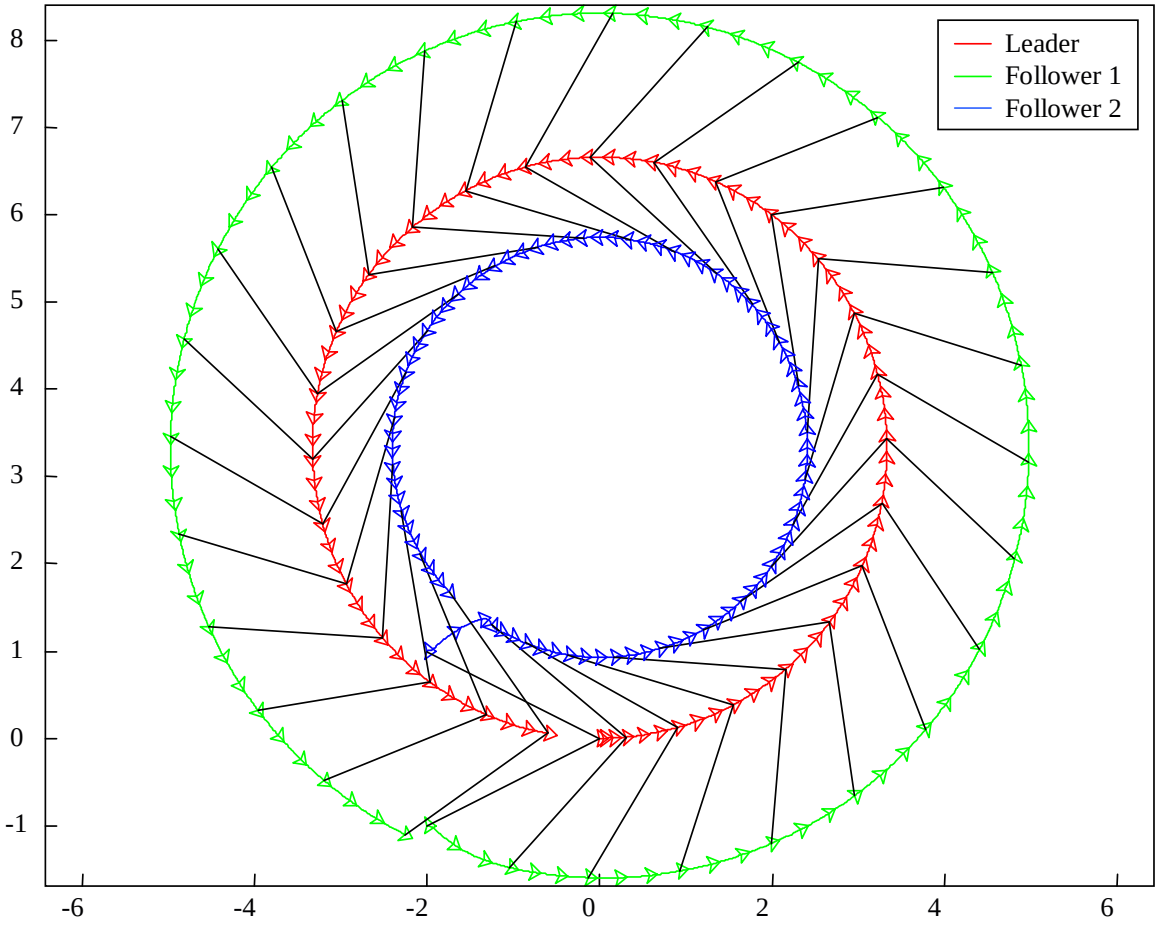


Figure 5.13: Trajectories of the leader robot and two followers for a circular path and $d = 0$.

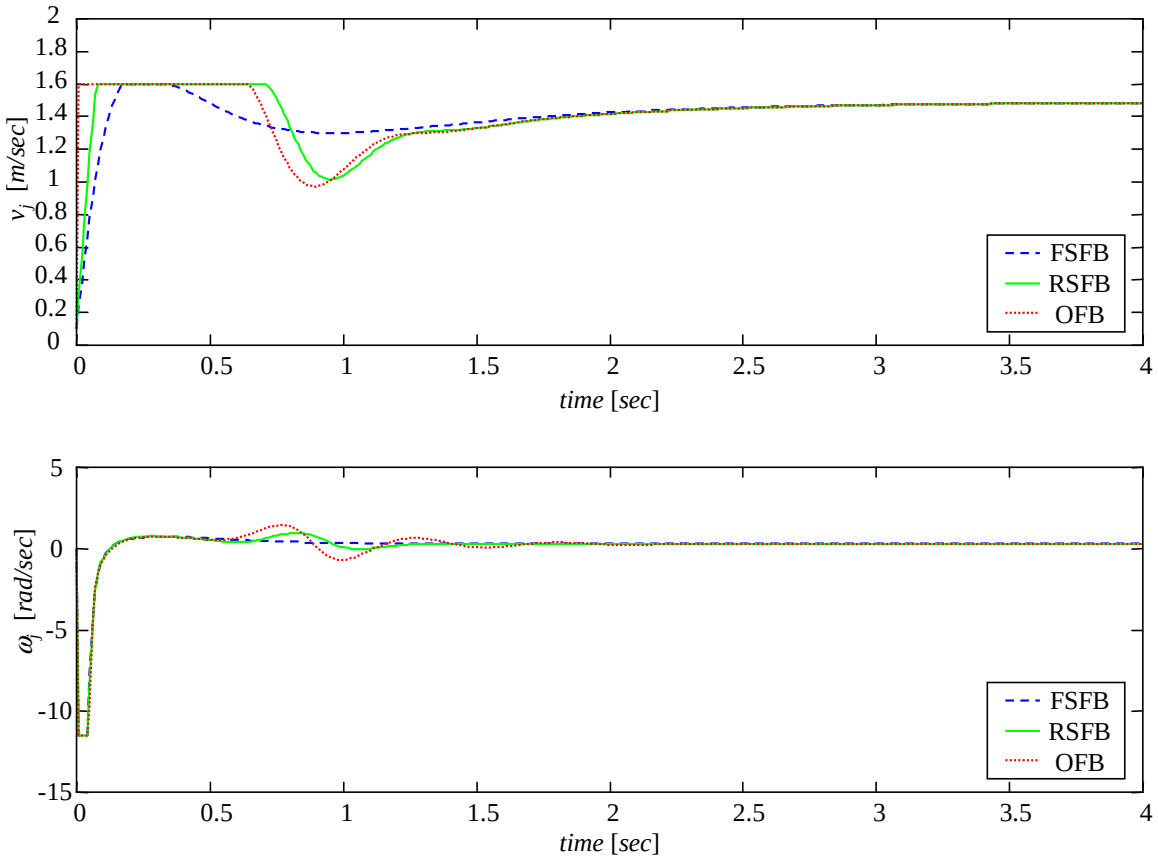


Figure 5.14: Velocity comparison between controllers (5.38), (5.40), and (5.52) for a circular path and $d = 0$.

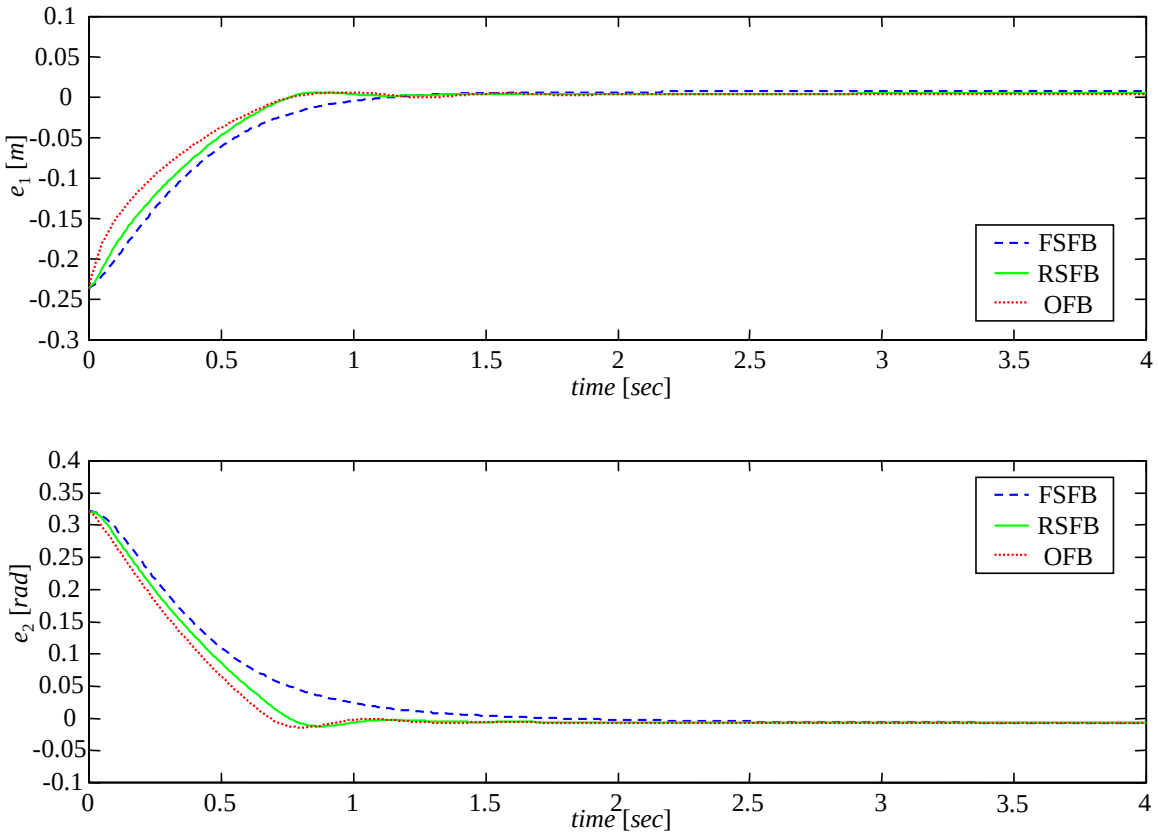


Figure 5.15: Output error comparison between controllers (5.38), (5.40), and (5.52) for a circular path and $d = 0$.

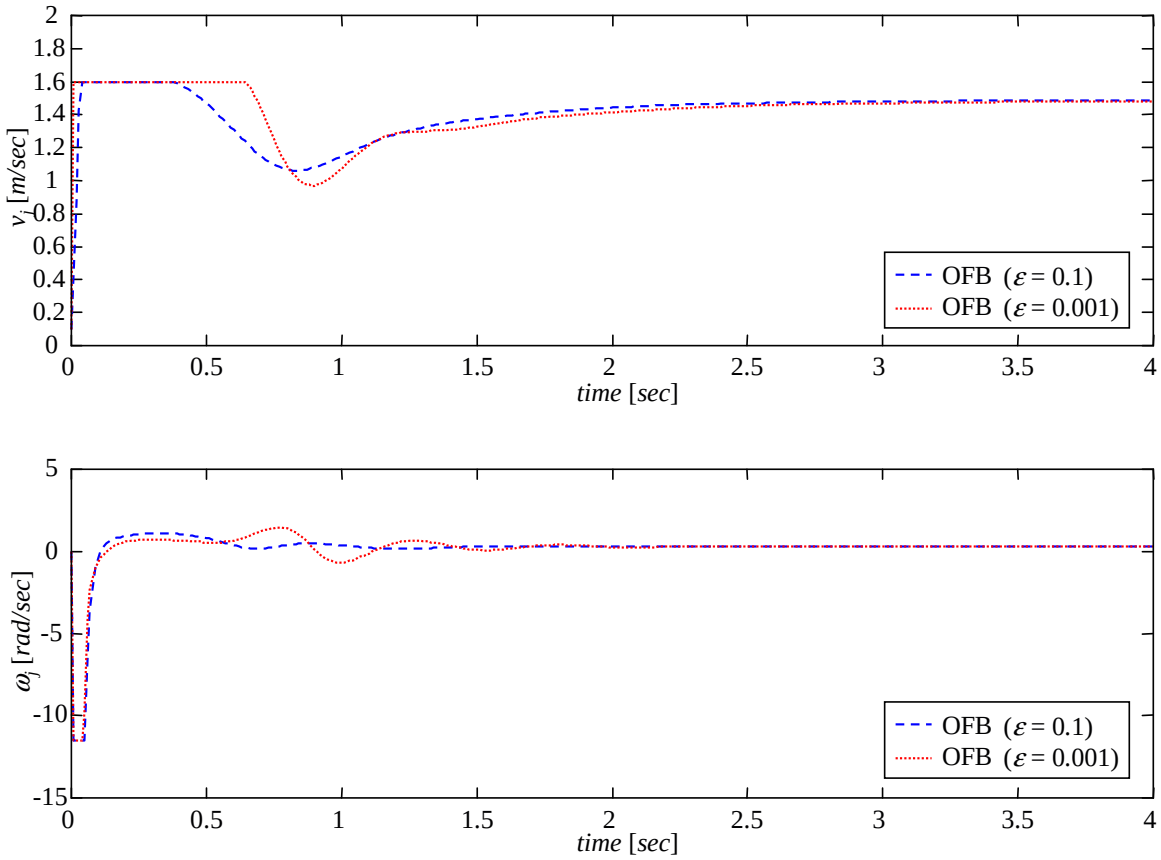


Figure 5.16: Control effort comparison with $\varepsilon = 0.1, 0.001$ for a circular path and $d = 0$.

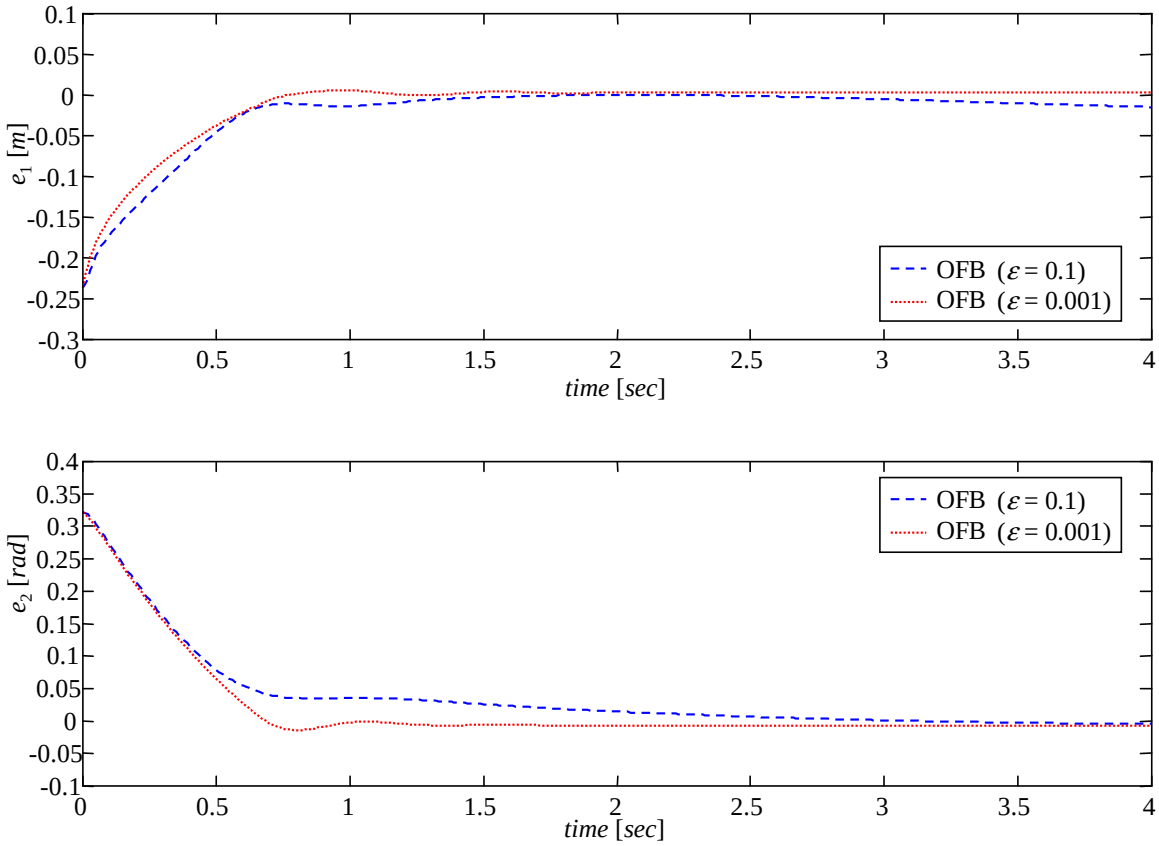


Figure 5.17: Output error comparison with $\varepsilon = 0.1, 0.001$ for a circular path and $d = 0$.

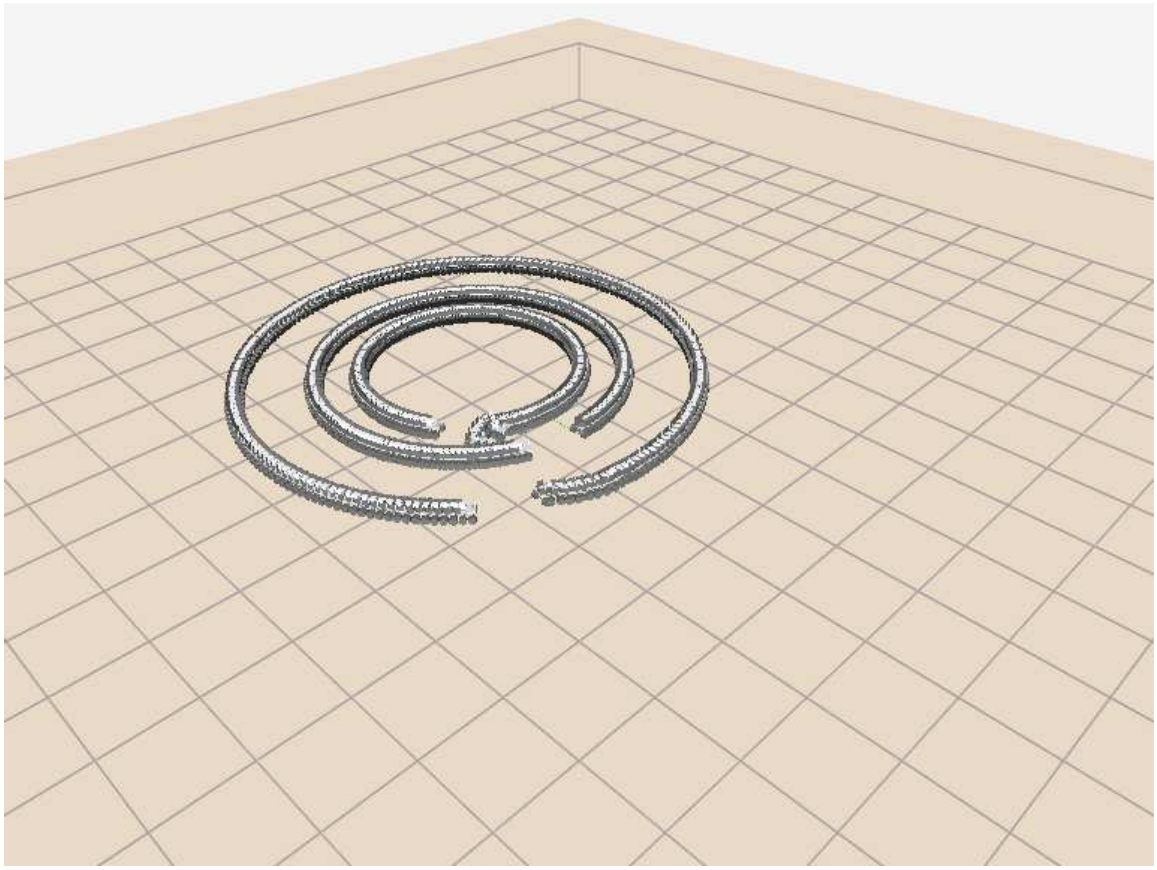


Figure 5.18: Snake view of a simulation with 3 robots for a circular path and $d = 0$.

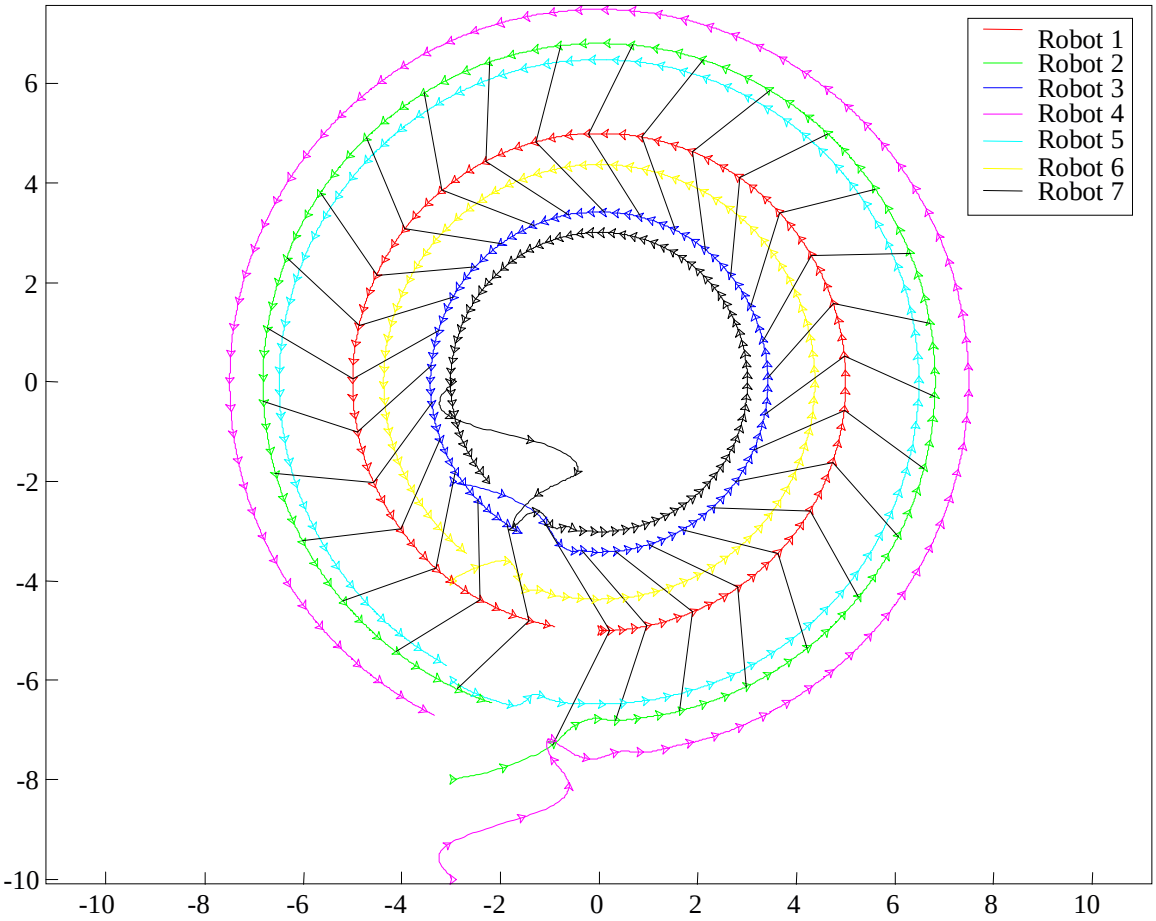


Figure 5.19: Trajectories of the robots with $d = 0$.

their initial velocities are equal to 0.01 m/sec. The formation control graph is defined by the specifications shown in Figure 5.23. It should be noted that robots 2, 3, 4, and 7 use leader-follower algorithm, whereas robots 5 and 6 use two leader-follower algorithms.

Figures 5.20 and 5.21 show the control inputs and the specification error of robot 5 using a two leader-follower algorithm. Figure 5.22 shows simulations of the OFB controller with seven-robot formation in the 3D simulation environment MPSLab. As it can be seen, despite of the lack of knowledge about the leader or leaders state, the behavior is quite satisfactory using any of the OFB observer/controller pairs.

Figure 5.24 to 5.29 show simulations using $d \neq 0$. It can be easily seen that this controller outperforms the previous one for straight line and circular paths. The main reasons for this better performance are the straight computation of $\dot{\omega}_k(t)$ and the smoothing effect of the integral term on $\omega_k(t)$. For $d = 0$, $\omega_k(t)$ contains a variable structure signal. This signal affects only $\dot{\omega}_k(t)$ for $d \neq 0$.

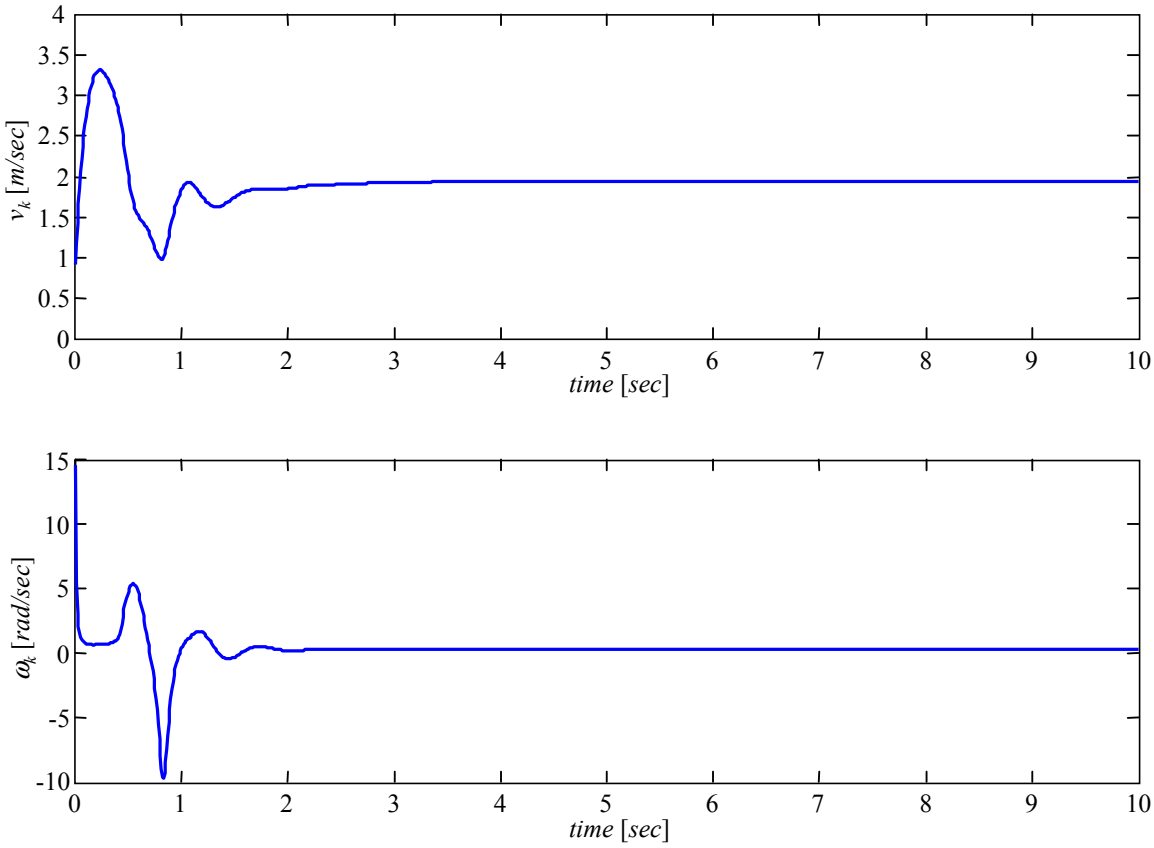


Figure 5.20: Velocities of robot 5 using the two-leader algorithm with $d = 0$.

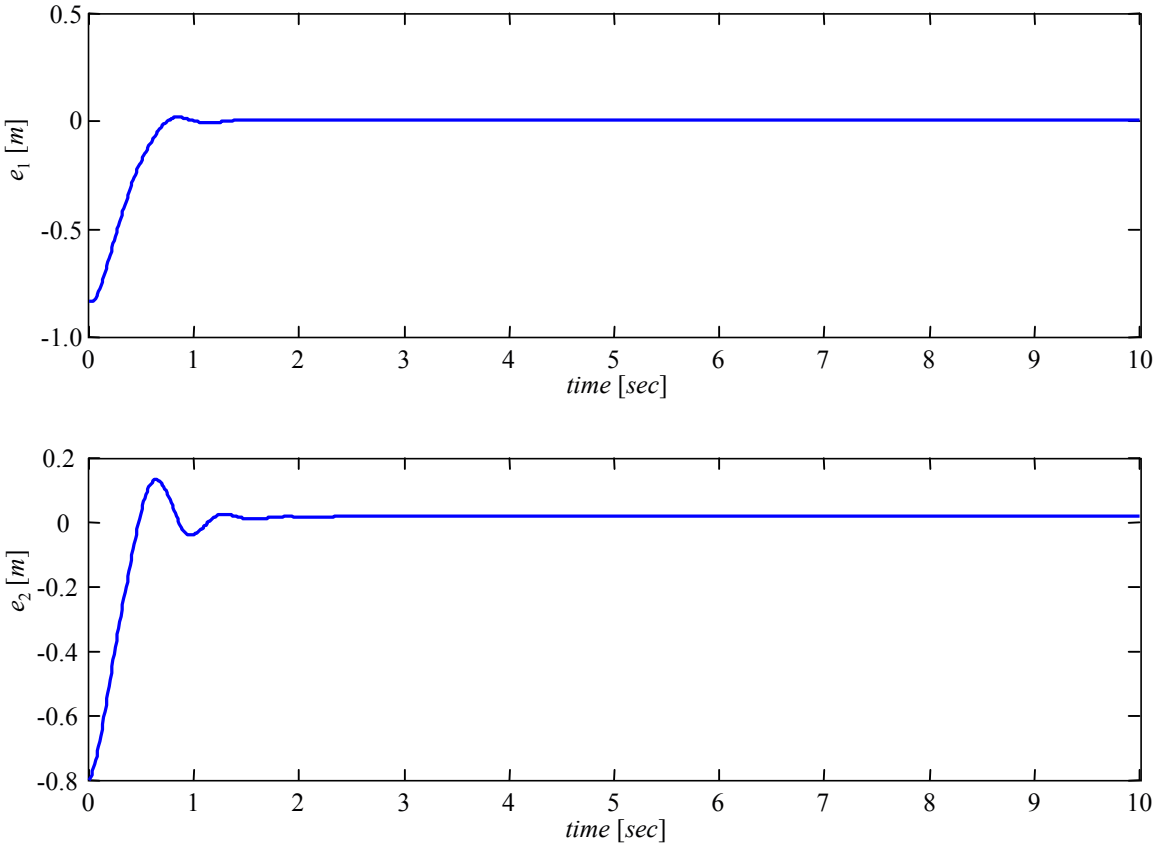


Figure 5.21: Output error of robot 5 using the two-leader algorithm with $d = 0$.

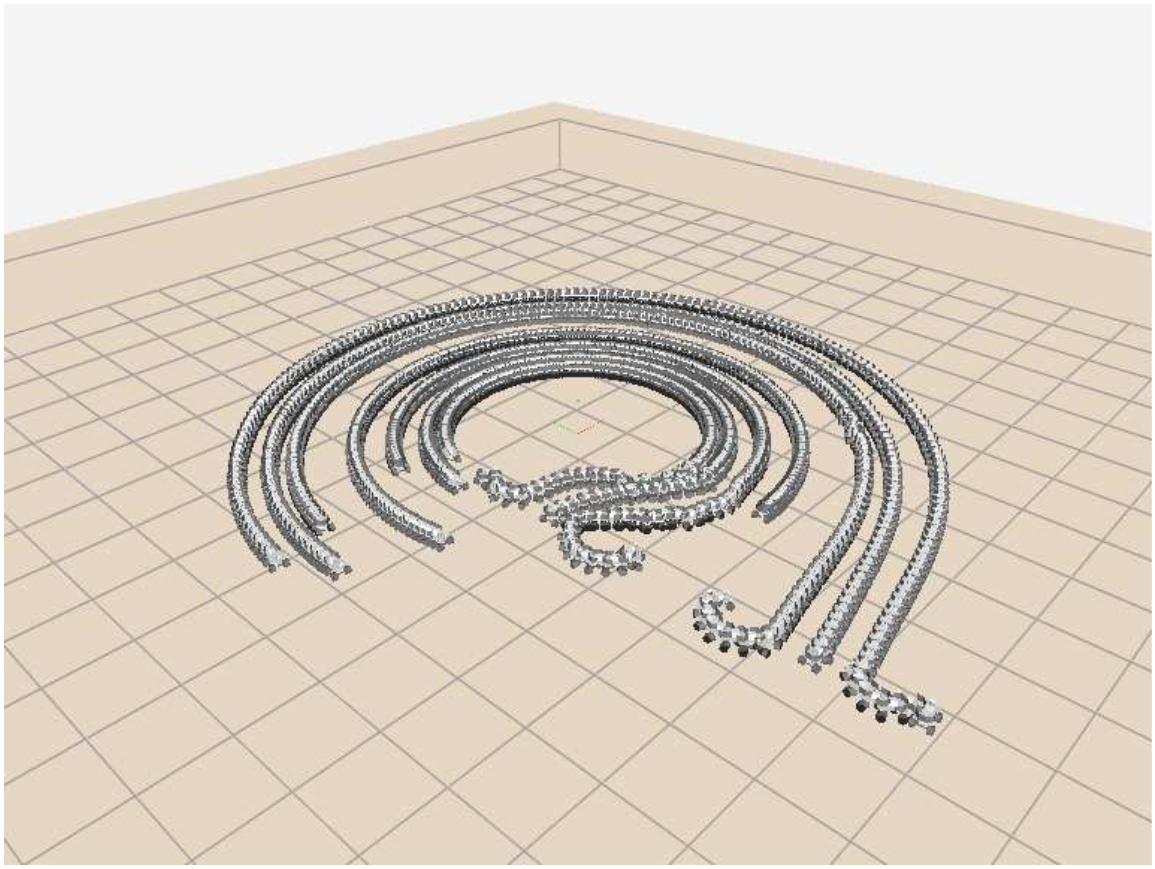


Figure 5.22: Snake view of a simulation with 7 robots with $d = 0$.

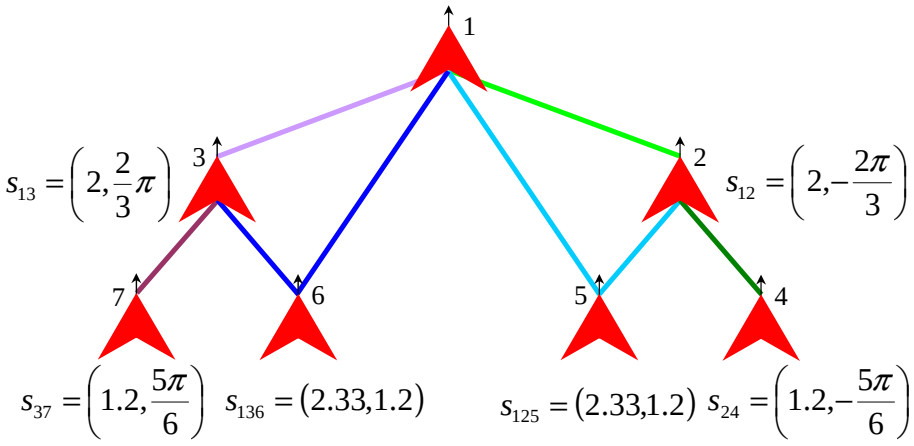


Figure 5.23: Specifications for 7 robots.

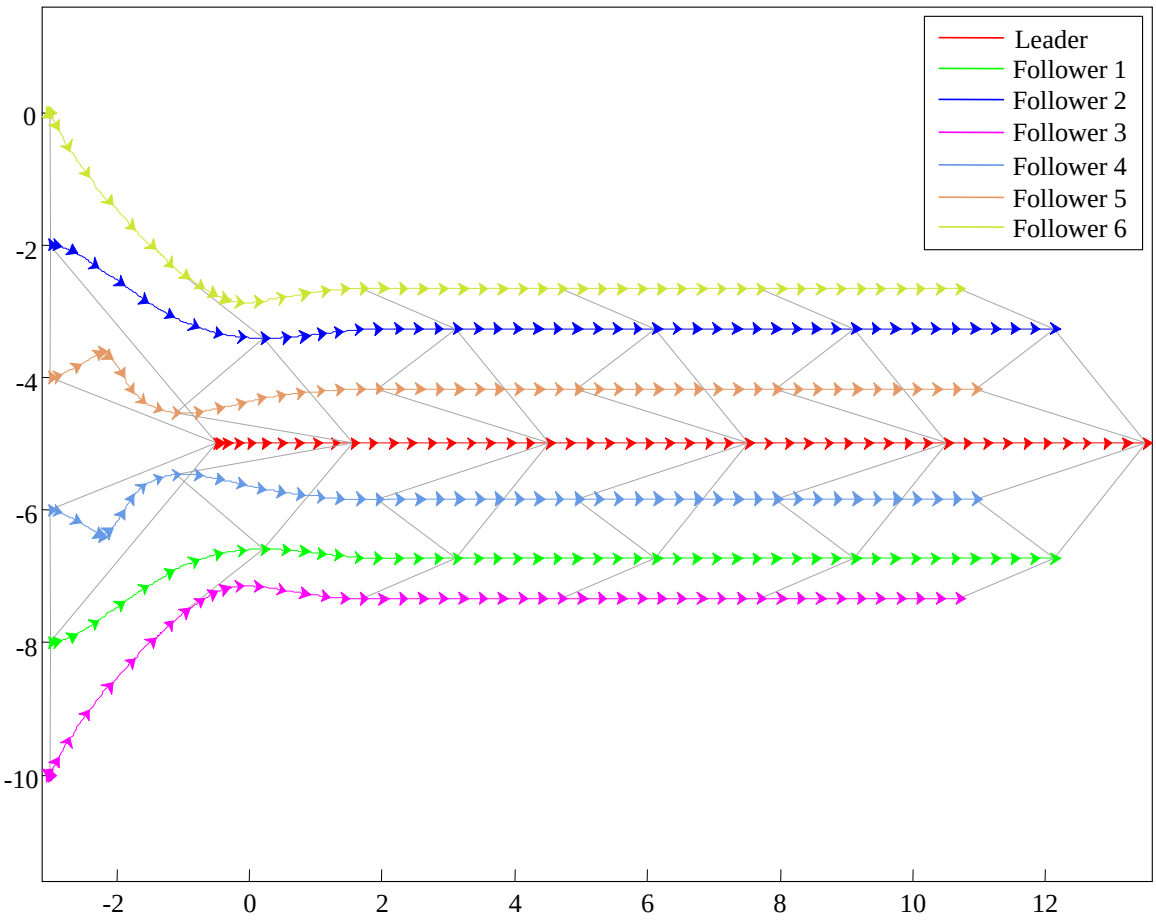


Figure 5.24: Trajectories of the robots for a straight line path and $d \neq 0$.

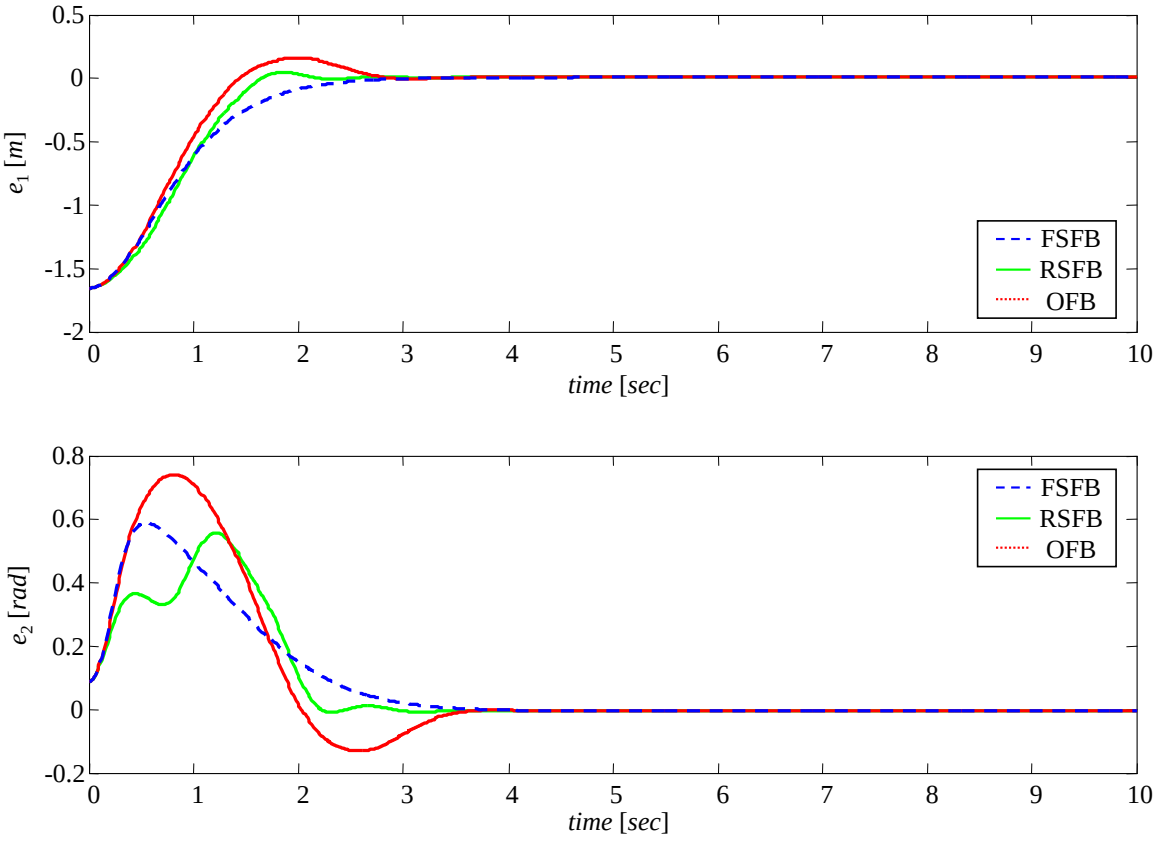


Figure 5.25: Error comparison for a straight line path and $d \neq 0$.

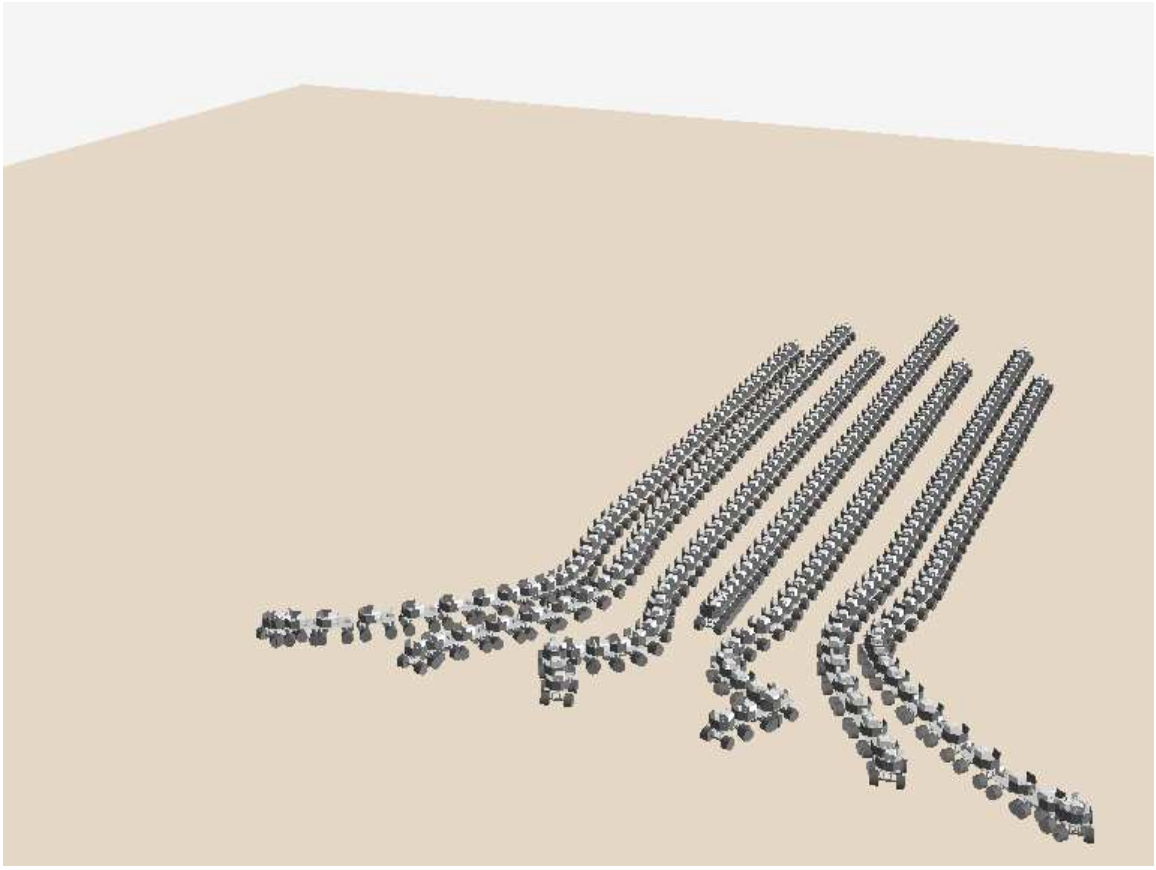


Figure 5.26: Snake view of a simulation with 7 robots for a straight line path with $d \neq 0$.

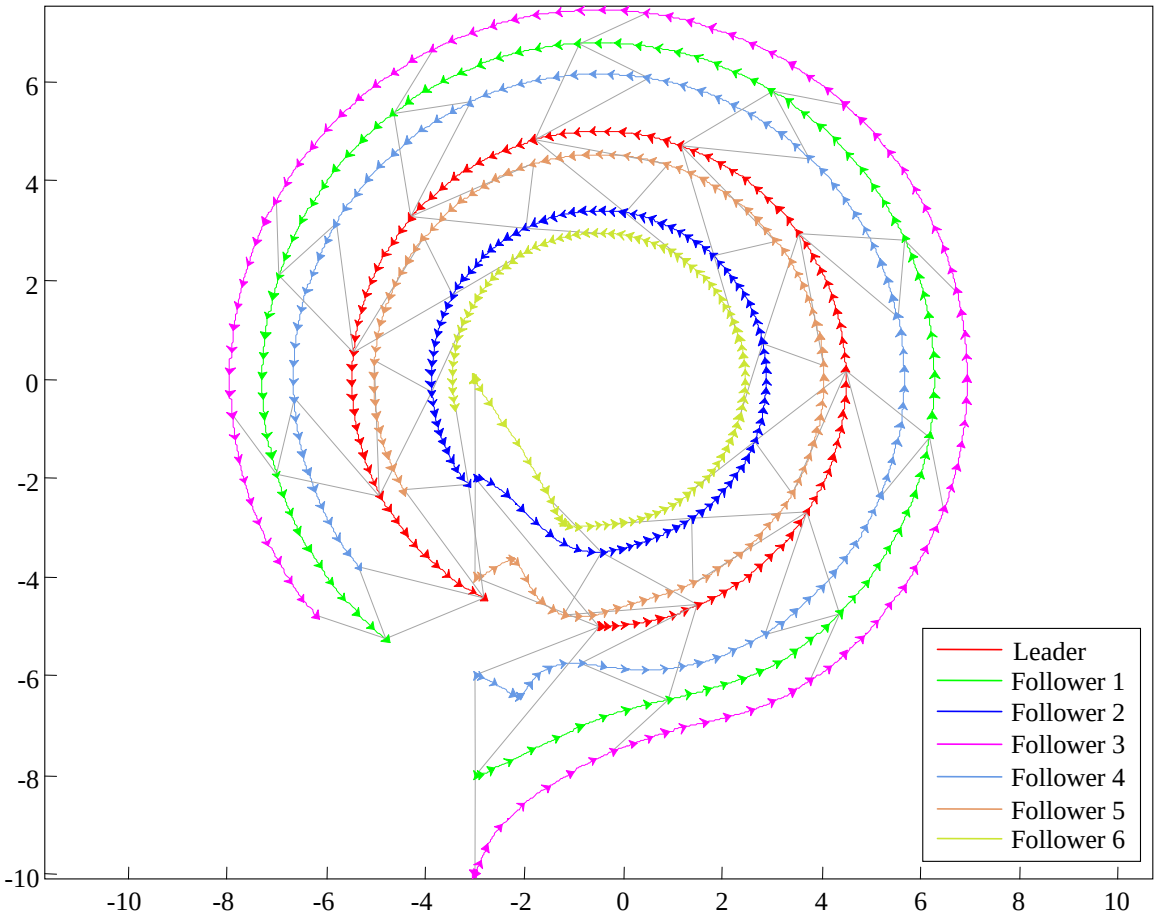


Figure 5.27: Trajectories of the robots for a circular path and $d \neq 0$.

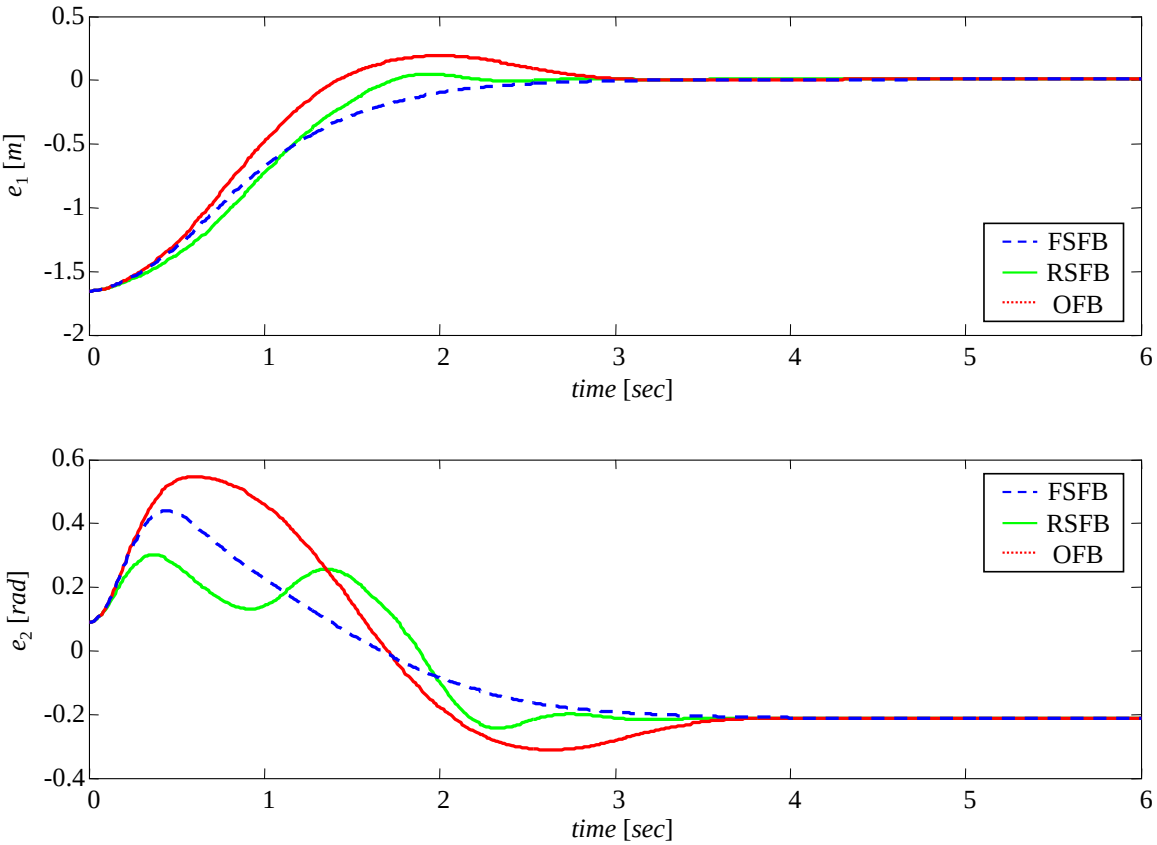


Figure 5.28: Error comparison for a circular path and $d \neq 0$.

Figure 5.29: Snake view of a simulation with 7 robots for a circular path with $d \neq 0$.

5.6 Summary

This Chapter presents robust multi-vehicle output feedback decentralized controllers. Using the main result of this work, an output feedback controller, each robot only requires a single camera to maintain a specified formation shape. The formation specification is given by the relative distance and bearing angle between each robot and its leader. A high gain observer is used to estimate the states of the robot and its neighbor(s).

A theoretical analysis based on Lyapunov stability theory has been performed to prove asymptotic stability of the robust state feedback formation controller (Subsection 5.3.2), and globally uniformly ultimately bounded stability of the output feedback formation controller (Subsection 5.3.3). Simulations in a realistic 3D environment verified the formation controller performances.

Part III

Conclusions

Chapter 6

Summary and future work

This dissertation presents a framework for vision-based control for multi-vehicle coordination. This framework consists of two main parts: (*i*) a model-based vision system, and (*ii*) vision-based formation control algorithms.

The model-based vision system, presented in Chapter 4, combines fiducial-based robot identification and pose estimation algorithms. It can estimate relative pose (distance and angle) between two or more robots from single images. The system is divided in two main components, image processing and pose estimation. For pose estimation, three approaches are analyzed and compared: POSIT algorithm, PRA approach, and Lowe's method. POSIT presents the fastest convergence rate and highest simplicity. However, it fails to arrive to correct results in specific situations. The PRA algorithm requires more iterations to achieve convergence. But, it is less complex than Lowe's algorithm. Lowe's algorithm achieves the best results of the three algorithms described here. Its main problem is its complexity and need of several iterations to achieve convergence.

The formation control algorithms presented in Chapter 5 are robust vision-based output feedback decentralized controllers that use relative position between a robot and its designated leader(s) to define the control law. The term robust is understood in the sense that the algorithms do not need the knowledge of leader's velocity. Three different controllers are presented for the cases of leader following and two-leaders following. The first controller assumes complete knowledge of relative positions, velocities, and accelerations. The second controller assumes knowledge of relative positions and their time-derivatives. Finally, the third controller assumes knowledge of just relative positions, eliminating the need of inter-vehicle communications and, consequently, increasing the reliability of the overall system. In the last approach, high-gain observers are used to estimate derivatives from visual data. Stability properties of

the three approaches are proved using Lyapunov theory. Simulations in a realistic 3D environment validate the performance of the designed architecture and algorithms.

The deployment of multi-robot systems relies on communication networks operating in a dynamic and often hostile environment. These systems are expected to operate autonomously with minimal human intervention, reduced decision-making time, and limited sensors. Therefore, to help establish correctness and robustness of coordination algorithms, it is of critical importance the analysis of potential failures in the execution of coordination algorithms in the presence of dynamic changes in the communication topology, communication malfunctions, addition and removal of agents, cuts on communication links, and reduced sensor capabilities. Moreover, a key issue in multi-vehicle systems is the interplay between control and communication in the robotic network. Additionally, the effects of control interconnections and information flow on the stability properties of the group are a major concern.

Future research will focus on the analysis of input-to-state [127, 133] and string stability [110, 124] properties of the controllers presented in this work. Moreover, another key problem is the synchronization between systems. A typical example, is the synchronization between a system and its observer, but, more interestingly, it is the search of new techniques capable of modelling complex behaviors found in nature, like flocking, swarming, or schooling. Future work will address this issue.

Bibliography

[1] J.H. Ahrens and H.K. Khalil. Asymptotic properties of extended kalman filters for a class of nonlinear systems. In *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05*, pages 6152–6157, 2005.

[2] A.N. Atassi and H.K Khalil. A separation principle for the stabilization of a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 44(9):1672–1687, September 1999.

[3] E. Bahceci, O. Soysal, and E. Sahin. A review: Pattern formation and adaptation in multi-robot systems. Technical Report CMU-RI-TR-03-43, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, October 2003.

[4] T. Balch and R.C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Automatic Control*, 14(6):926–939, December 1998.

[5] G. Bartolini, A. Ferrara, and E. Usai. Chattering avoidance by second-order sliding mode control. *IEEE Transactions on Automatic Control*, 43(2):241–246, 1998.

[6] G. Bartolini, A. Ferrara, E. Usai, and V.I. Utkin. On multi-input chattering-free second-order sliding mode control. *IEEE Transactions on Automatic Control*, 45(9):1711–1717, 2000.

[7] G. Bartolini, A. Pisano, and E. Usai. An improved second-order sliding-mode control scheme robust against the measurement noise. *IEEE Transactions on Automatic Control*, 49(10):1731–1737, 2004.

[8] J.S. Bellingham, M. Tillerson, M. Alighanbari, and J.P. How. Cooperative path planning for multiple UAVs in dynamic and uncertain environment. In

Proceedings of the IEEE Conference on Decision and Control, number 3, pages 2816–2822, Las Vegas, NV, December 2002.

[9] C. Belta and V. Kumar. Optimal motion generation for groups of robots: A geometric approach. *ASME Journal of Mechanical Design*, 126:63–70, 2004.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, March 2004.

[11] G. Bradski. The OpenCV library. *Dr. Dobb's Journal*, 25(11):120, 122–125, November 2000.

[12] J. Bullingham, A. Richards, and J.P. How. Receding horizon control of autonomous aerial vehicles. In *Proceedings of the American Control Conference*, pages 3741–3746, Anchorage, AK, May 2002.

[13] S. Camazine, J.L. Deneubourg, N. Franks, J. Sneyd, E. Bonabeau, and G. Theraulaz. *Self-Organization in Biological Systems*. Princeton University Press, 2001.

[14] Y.U. Cao, A.S. Fukunaga, and A. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.

[15] M.M.-Y. Chang and K.H. Wong. Model reconstruction and pose acquisition using extended Lowe's method. *Multimedia, IEEE Transactions on*, 7(2):253–260, 2005.

[16] J. Chen, D.M. Dawson, W.E. Dixon, and A. Behal. Adaptive homography-based visual servo tracking for fixed and camera-in-hand configurations. *IEEE Transactions on Control Systems Technology*, 13(5):814–825, September 2005.

[17] J. Chen, W.E. Dixon, D.M. Dawson, and M. McIntyre. Homography-based visual servo tracking control of a wheeled mobile robot. *IEEE Transactions on Robotics*, 22(2):406–415, April 2006.

[18] X. Chen and A. Serrani. Remarks on ISS and formation control. In *Proceedings of the IEEE Conference on Decision and Control*, number 1, pages 177–182, Atlantis, Paradise Island, Bahamas, dec 2004.

[19] X. Chen and A. Serrani. An internal model approach to autonomous leader/follower trailing for non-holonomic vehicles. *International Journal of Robust and Nonlinear Control*, 16(14):641–670, 2006.

[20] X. Chen, A. Serrani, and H. Özbay. Control of leader-follower formations of terrestrial UAVs. In *Proceedings of the 42nd IEEE Conference on Decision and Control*, number 1, pages 498–503, Maui, Hawaii USA, December 2003.

[21] M. Cheriet, J.N. Said, and C.Y. Suen. A recursive thresholding technique for image segmentation. *IEEE Transactions on Image Processing*, 7(6):918–921, 1998.

[22] S.Y. Chiem and E. Cervera. Vision-based robot formations with Bézier trajectories. In F. Groen et al., editor, *Intelligent Autonomous Systems*, volume 8, pages 191–198. IOS Press, 2004.

[23] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun. *Principles of Robot Motion - Theory, Algorithms, and Implementations*. Intelligent Robotics and Autonomous Agents. MIT Press, June 2005.

[24] D. Claus and A.W. Fitzgibbon. Polyhedral reliable automatic calibration of a marker-based position tracking system. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 300–305, January 2005.

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

[26] N. Cowan, O. Shakerina, R. Vidal, and S. Sastry. Vision-based follow-the-leader. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, number 2, pages 1796–1801, Las Vegas, Nevada USA, October 2003.

[27] A.K. Das. *A framework for control of formations of mobile robots: Theory and experiments*. PhD thesis, Mechanical Engineering and Applied Mechanics, University of Pennsylvania, 2003.

[28] A.K. Das, R. Fierro, V. Kumar, J.P. Ostrowski, J. Spletzer, and C.J. Taylor. A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, 18(5):813–825, 2002.

[29] P. David, D. DeMenthon, R. Duraiswami, and H. Samet. SoftPOSIT: Simultaneous pose and correspondence determination. *International Journal of Computer Vision*, 59(3):259–284, September 2004.

[30] J. Davila, L. Fridman, and A. Levant. Second-order sliding-mode observer for mechanical systems. *IEEE Transactions on Automatic Control*, 50(11):1785–1789, 2005.

[31] D. DeMenthon and L.S. Davis. Recognition and tracking of 3D objects by 1D search. In *Proceedings of the Image Understanding Workshop*, pages 653–659, Washington, DC, April 1993.

[32] D. DeMenthon and L.S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141, 1995.

[33] J.P. Desai, J. Ostrowski, and V. Kumar. Controlling formations of multiple mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2864–2869, Leuven, Belgium, May 1998.

[34] J.P. Desai, J.P. Ostrowski, and V. Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17(6):905–908, December 2001.

[35] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, 3rd edition, 2005.

[36] W. B. Dunbar and R. M. Murray. Model predictive control of coordinated multi-vehicle formations. In *Proceedings of the IEEE Conference on Decision and Control*, pages 4631–4636, Las Vegas, NV, Dec. 10-13 2002.

[37] T. Eren, P.N. Belhumeur, and A.S. Morse. Closing ranks in vehicle formations based on rigidity. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2959–2964, Las Vegas, Nevada, December 2002.

[38] Y. Fang, W.E. Dixon, D.M. Dawson, and P. Chawda. Homography-based visual servoing of wheeled mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 35(5):1041–1050, October 2005.

[39] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Artificial Intelligence. The MIT Press, November 1993.

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

[41] J.T. Feddema, R.D. Robinett, and R.H. Byrne. An optimization approach to distributed controls of multiple robot vehicles. In *Workshop on Control and Cooperation of Intelligent Miniature Robots, IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, Nevada, October 31 2003.

[42] M. Fiala. Vision guided control of multiple robots. In *Proceedings of the First Canadian Conference on Computer and Robot Vision*, pages 241–246, Washington DC, USA, May 2004. IEEE Computer Society.

[43] R. Fierro, L. Chaimowicz, and V. Kumar. Multi-robot cooperation. In S.S. Ge and F.L. Lewis, editors, *Autonomous Mobile Robots: Sensing, Control, Decision Making and Applications*, chapter 11, pages 417–459. CRC Press - Taylor & Francis Group, May 2006.

[44] R. Fierro, A. Das, J. Spletzer, J. Esposito, V. Kumar, J.P. Ostrowski, G. Pappas, C.J. Taylor, Y. Hur, R. Alur, I. Lee, G. Grudic, and J. Southall. A framework and architecture for multi-robot coordination. *The International Journal of Robotics Research*, 21(10-11):977–995, October-November 2002.

[45] R. Fierro and F.L. Lewis. Control of a nonholonomic mobile robot: backstepping kinematics into dynamics. In *Proceedings of the IEEE Conference on Decision and Control*, volume 4, pages 3805–3810 vol.4, 1995.

[46] R. Fierro and K. Wesselowski. Optimization-based control of multi-vehicle systems. In V. Kumar, N.E. Leonard, and A.S. Morse, editors, *A Post-Workshop Volume 2003 Block Island Workshop on Cooperative Control Series*, volume 309 of *LNCIS*, pages 63–78. Springer, 2005.

[47] D.A. Forsyth and J. Ponce. *Computer vision: A modern approach.* Prentice Hall, 2003.

[48] R.A. Freeman and P.V. Kototovic. *Robust nonlinear control design: State-space and Lyapunov techniques.* Birkhäuser Boston Inc, 1996.

[49] L. Fridman, A. Levant, and J. Davila. High-order sliding-mode observer for linear systems with unknown inputs. In *Proceedings of the International Workshop on Variable Structure Systems*, pages 202–207, 2006.

[50] B.P. Gerkey and M.J. Matarić. Pusher-watcher: An approach to fault-tolerant tightly-coupled robot coordination. In *Proceedings of the IEEE International*

Conference on Robotics and Automation, number 1, pages 464–469, Washington, DC, May 2002.

[51] D. Grünbaum. Align in the sand. *Science*, 312:1320–1322, June 2006.

[52] Y h. Tu and J. Toner. Long range order in a two dimensional xy model: how birds fly together. *Physical Review Letters*, 75:4326–4329, December 1995.

[53] Y. Han and H. Hahn. Visual tracking of a moving target using active contour based SSD algorithm. *Robotics and Autonomous Systems*, 53(3–4):265–281, 2005.

[54] R. Hermann and A. Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22(5):728–740, 1977.

[55] S. Ho and H.K. Khalil. Nonlinear output-feedback tracking using high-gain observer and variable structure control. *Automatica*, 33(10):1845–1856, 1997.

[56] Q. Huang, W. Gao, and W. Cai. Thresholding technique with adaptive window selection for uneven lighting image. *Pattern Recognition Letters*, 26(8):801–808, May 2005.

[57] T.S. Huang and A.N. Netravali. Motion and structure from feature correspondences: A review. *Proceedings of the IEEE*, 82(2):252–268, 1994.

[58] A. Isidori. *Nonlinear Control Systems*. Communications and Control Engineering. Springer, 3rd edition, 1995.

[59] A. Isidori. *Nonlinear Control Systems II*. Communications and Control Engineering. Springer, 1999.

[60] A. Isidori and L. Marconi. An internal model-based approach to certain pursuit-evasion problems. In *Proceedings of NOLCOS 2004*, pages 113–118, Stuttgart, Germany, 2004.

[61] A. Isidori, L. Marconi, and A. Serrani. *Robust Autonomous Guidance: An Internal Model Approach*. Advances in Industrial Control. Springer, 2003.

[62] A. Isidori, A.R. Teel, and L. Praly. A note on the problem of semiglobal practical stabilization of uncertain nonlinear systems via dynamic output feedback. *Systems and Control Letters, Volume 39, Number 3, 15 March 2000, pp. 165-171(7)*, 39(3):165–171, March 2000.

[63] A. Jadbabaie, J. Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.

[64] A. Jadbabaie, N. Motee, and M. Barahona. On the stability of the kuramoto model of coupled nonlinear oscillators. In *Proceedings of the American Control Conference*, volume 5, pages 4296–4301, 2004.

[65] S. Julier and J. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proceedings of the International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, Orlando, FL, 1997.

[66] S. Julier and J. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004.

[67] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top AR environment. In *Proceedings of the International Symposium on Augmented Reality (ISAR 2000)*, pages 111–119, Munich, Germany, 2000.

[68] T. Keviczky, F. Borrelli, and G.J. Balas. A study on decentralized receding horizon control for decoupled systems. In *Proceedings of the American Control Conference*, volume 6, pages 4921–4926, Boston, MA, June 2004.

[69] H.K. Khalil. *Nonlinear Systems*. Prentice Hall, 3rd edition, 2002.

[70] H.K. Khalil. A note on the robustness of high-gain-observer-based controllers to unmodeled actuator and sensor dynamics. *Automatica*, 41(10):1821–1824, October 2005.

[71] P. Kokotovic and M. Arcak. Constructive nonlinear control: a historical perspective. *Automatica*, 37:637–662, 2001.

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

[73] J. Lawton, B. Young, and R. Beard. A decentralizad approach to formation maneuvers. *IEEE Transactions on Automatic Control*, 19(6):933–941, December 2003.

[74] V. Lepetit and P. Fua. Monocular model-based 3D tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, October 2005.

[75] A. Levant. Higher order sliding: differentiation and black-box control. In *Proceedings of the IEEE Conference on Decision and Control*, volume 2, pages 1703–1708, 2000.

[76] A. Levant. Higher-order sliding modes, differentiation and output-feedback control. *International Journal of Control*, 76(9 - 10):924–941, 2003.

[77] A. Levant. Homogeneity approach to high-order sliding mode design. *Automatica*, 41:823–830, 2005.

[78] A. Levant. Quasi-continuous high-order sliding-mode controllers. *IEEE Transactions on Automatic Control*, 50(11):1812–1816, 2005.

[79] F.L. Lewis, C.T. Abdallah, and D.M. Dawson. *Control of Robot Manipulators*. MacMillan Pub., 1993.

[80] D. Liberzon, A.S. Morse, and E.D. Sontag. A new definition of the minimum-phase property for nonlinear systems, with an application to adaptive control. In *Proceedings of the IEEE Conference on Decision and Control*, volume 3, pages 2106–2111, 2000.

[81] D. Liberzon, A.S. Morse, and E.D. Sontag. Output-input stability and minimum-phase nonlinear systems. *IEEE Transactions on Automatic Control*, 47(3):422–436, 2002.

[82] D.G. Lowe. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.

[83] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[84] Magdi S. Mahmoud and Hassan K. Khalil. Robustness of high-gain observer-based nonlinear controllers to unmodeled actuators and sensors. *Automatica*, 38(2):361–369, February 2002.

[85] G. Mariottini, G.J. Pappas, D. Prattichizzo, and K. Daniilidis. Vision-based localization of leader-follower formations. In *Proceedings of the IEEE Conference on Decision and Control and European Control Conference (CDC-ECC '05)*, pages 635–640, Seville, Spain, December 2005.

[86] J.M. Mendel. *Lessons in Estimation Theory for Signal Processing, Communications, and Control*. Signal Processing. Prentice Hall, 2nd edition, 1995.

[87] P. Moreels and P. Perona. Evaluation of features detectors and descriptors based on 3D objects. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 800–807, 2006.

[88] N. Moshtagh, A. Jadbabaie, and K. Daniilidis. Vision-based distributed coordination and flocking of multi-agent systems. In *Proceedings of Robotics: Science and Systems*, pages 41–48, Cambridge, USA, June 2005.

[89] N. Moshtagh, A. Jadbabaie, and K. Daniilidis. Vision-based control laws for distributed flocking of nonholonomic agents. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2769–2774, 2006.

[90] R.M. Murray. Recent research in cooperative control of multi-vehicle systems. *submitted to ASME Journal of Dynamic Systems, Measurement, and Control Special issue on the "Analysis and Control of Multi Agent Dynamic Systems"*, 2006.

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

[92] E.M. Nebot, M Bozorg, and H.F. Durrant-Whyte. Decentralized architecture for asynchronous sensors. *Autonomous Robots*, 6(2):147–164, April 1999.

[93] H. Nijmeijer and A.J. van der Schaft. *Nonlinear Dynamical Control Systems*. Springer-Verlag, 1991.

[94] E.V.L. Nunes, Liu Hsu, and F. Lizarralde. Output-feedback sliding mode control for global asymptotic tracking of uncertain systems using locally exact differentiators. In *Proceedings of the American Control Conference*, pages 5407–5412, 2006.

[95] D. Oberkampf, D.F. DeMenthon, and L.S. Davis. Iterative pose estimation using coplanar feature points. *Computer Vision and Image Understanding*, 63(3):495–511, May 1996.

[96] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. Technical Report CIT-CDS 2004-005, Control and Dynamical Systems, Pasadena, CA, June 2004.

[97] R. Olfati-Saber and R.M. Murray. Distributed structural stabilization and tracking for formations of dynamic multi-agents. In *Proceedings of the IEEE Conference on Decision and Control*, volume 1, pages 209–215, 2002.

[98] S. H. Or, W. S. Luk, K. H. Wong, and I. King. An efficient iterative pose estimation algorithm. *Image and Vision Computing*, 16(5):353–362, April 1998.

[99] O.A.A. Orqueda. Software and algorithms for motion planning in virtual environments. In V. H. Cortínez M. B. Rosales and D. V. Bambill, editors, *Computational Mechanics*, number XXII, Argentina, November 2003.

[100] O.A.A. Orqueda. *Motion Planning and Control of Autonomous Robots*. Doctoral thesis, Universidad Nacional del Sur, Department of Electrical and Computer Engineering, Bahía Blanca, Argentina, August 2006. In Spanish.

[101] O.A.A. Orqueda and R. Fierro. Robust vision-based nonlinear formation control. In *Proceedings of the American Control Conference*, pages 1422–1427, Minneapolis, MN, June 2006.

[102] O.A.A. Orqueda and R. Fierro. A vision-based nonlinear decentralized controller for unmanned vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1–6, Orlando, FL, May 2006.

[103] O.A.A. Orqueda and R. Fierro. Visual tracking of mobile robots in formation. In *Proceedings of the American Control Conference*, September 2006. Submitted.

[104] O.A.A. Orqueda and R. Fierro. A visual tracking system for controlling mobile robots in formation. *Robotics and Autonomous Systems*, December 2006. Submitted.

[105] O.A.A. Orqueda and R. Fierro. Model predictive path-space iteration for multi-robot coordination. In D. Grundel, R. Murphey, P. Pardalos, and O. Prokopyev, editors, *Cooperative Systems: Control and Optimization*, volume 588 of *Lecture Notes in Economics and Mathematical Systems*, pages 1–25. Springer, February 2007.

[106] O.A.A. Orqueda, T. Zhang, and R. Fierro. An output feedback nonlinear decentralized controller design for multiple unmanned vehicle coordination. *International Journal on Robust Control*, November 2006. Accepted for publication.

[107] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–69, 1979.

[108] J. Ousingsawat and M.E. Campbell. Establishing optimal trajectories for multi-vehicle reconnaissance. In *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Providence, RI, August 2004. (CD-ROM).

[109] D. Pagac, E.M. Nebot, and H. Durrant-Whyte. An evidential approach to map-building for autonomous vehicles. *IEEE Transactions on Robotics and Automation*, 14(4):623–629, 1998.

[110] A. Pant, P. Seiler, and K. Hedrick. Mesh stability of look-ahead interconnected systems. *IEEE Transactions on Automatic Control*, 47(2):403–407, February 2002.

[111] A. Papachristodoulou and A. Jadbabaie. Synchronization in oscillator networks: Switching topologies and non-homogeneous delays. In *44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference (CDC-ECC '05)*, pages 5692–5697, 2005.

[112] L.E. Parker. Current state of the art in distributed autonomous mobile robotics. In L.E. Parker, G. Bekey, and J. Barhen, editors, *Distributed Autonomous Robotic Systems 4*, volume 4, pages 3–12, Tokyo, October 2000. Springer-Verlag.

[113] N.S.V. Rao. Projective method for generic sensor fusion problem. In *Proceedings of the IEEE/SICE/RSJ International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Taipei, Taiwan, August 1999.

[114] P. Renaud, E. Cervera, and P. Martinet. Towards a reliable vision-based mobile robot formation control. In *Proceeding of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3176–3181, 2004.

[115] C.W. Reynolds. Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, July 1987.

[116] S. Scheding, G. Dissanayake, E.M. Nebot, and H. Durrant-Whyte. An experiment in autonomous navigation of an underground mining vehicle. *IEEE Robotics and Automation Magazine*, 15(1):85–95, February 1999.

[117] S. Se, D.G. Lowe, and J.J. Little. Vision-based global localization and mapping for mobile robots. *IEEE Transactions on Robotics*, 21(3):364–375, June 2005.

[118] R. Sepulchre, M. Janković, and P. Kokotović. *Constructive Nonlinear Control*. Communications and Control Engineering (CCES). Springer-Verlag, London, 1997.

[119] D. Shreiner, M. Woo, J. Neider, and T. Davis. *OpenGL programming guide: The official guide to learning OpenGL, Version 2*. Addison-Wesley Professional, 5th edition, August 2005.

[120] S.J. Simpson, A.G. Sword, P.D. Lorch, and I.D. Couzin. Cannibal crickets on a forced march for protein and salt. usa 103,. In *Proceedings of the National Academy of Sciences*, number 103, pages 4152–4156, 2006.

[121] J. Slotine and W. Li. *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice-Hall, 1991.

[122] E.D. Sontag. *Mathematical Control Control Theory: Deterministic Finite Dimensional Systems*. Number 6 in Texts in Applied Mathematics. Springer, New Yory, 2nd. edition, 1998.

[123] S. Suzuki and K. Abe. Topological structural analysis of digital binary images by border following. *CVGIP*, 30(1):32–46, 1985.

[124] D. Swaroop and J.K. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, 41(3):349–357, 1996.

[125] P. Tabuada, G. Pappas, and P. Lima. Feasible formations of multi-agent systems. In *Submitted to the American Control Conference*, 2001.

[126] C-P. Tang, R. Bhatt, and V. Krovi. Decentralized kinematic control of payload by a system of mobile manipulators. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, number 3, pages 2462–2467, New Orleans, LA USA, April 2004.

[127] H.G. Tanner. ISS properties of nonholonomic vehicles. *Systems & Control Letters*, 53(3-4):229–235, November 2004.

[128] H.G. Tanner, A. Jabdabaie, and G.J. Pappas. Stable flocking of mobile agents, part I: Fixed topology. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2010–2015, Maui, Hawaii, USA, December 2003.

[129] H.G. Tanner, A. Jabdabaie, and G.J. Pappas. Stable flocking of mobile agents, part II: Dynamic topology. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2016–2021, Maui, Hawaii, USA, December 2003.

[130] H.G. Tanner and K.J. Kyriakopoulos. Discontinuous backstepping for stabilization of nonholonomic mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3948–3953, 2002.

[131] H.G. Tanner and K.J. Kyriakopoulos. Backstepping for nonsmooth systems. *Automatica*, 39(7):1259–1265, July 2003.

[132] H.G. Tanner, G.J. Pappas, and V. Kumar. Input-to-state stability on formation graphs. In *Proceedings of the 41st IEEE Conference on Decision and Control*, number 3, pages 2439–2444, Las Vegas, Nevada USA, December 2002.

[133] H.G. Tanner, G.J. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, June 2004.

[134] A. Teel and L. Praly. Global stabilizability and observability imply semi-global stabilizability by output feedback. *Systems & Control Letters*, 22:313–325, 2001.

[135] V.I. Utkin. *Sliding modes in control and optimization.* Communication and control engineering. Springer-Verlag, 1992.

[136] V.I. Utkin, J. Guldner, and J. Shi. *Sliding mode control in electromechanical systems*, volume 9 of *Taylor & Francis systems and control book series.* Taylor & Francis, 1999.

[137] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet. Novel type of phase transition in a system of self-driven particles. *Physical Review Letters*, 75(6):1226–1229, August 1995.

[138] R. Vidal, O. Shakernia, and S. Sastry. Formation control of nonholonomic mobile robots with omnidirectional visual servoing and motion segmentation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, number 1, pages 584–589, Taipei, Taiwan, September 2003.

[139] R. Vidal, O. Shakernia, and S. Sastry. Following the flock. *IEEE Robotics and Automation Magazine*, 11(4):14–20, 2004.

[140] E.A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC 2000)*, pages 153–158, 2000.

[141] E.A. Wan and R. Van Der Merwe. *The Unscented Kalman Filter*, chapter 7, pages 221–280. Wiley Publishing, 2001.

[142] B.J. Young, R.W. Beard, and J.M. Kelsey. A control scheme for improving multi-vehicle formation maneuvers. In *Proceedings of the American Control Conference*, number 2, pages 704–709, Arlington VA, USA, June 2001.

[143] S. Zelinski, T.J. Koo, and S. Sastry. Optimization-based formation reconfiguration planning for autonomous vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3758–3763, Taipei, Taiwan, September 2003.

Part IV

Appendices

Appendix A

Projective geometry

Euclidean geometry preserves distances and angles after translation and rotation transformations. However, it is not applicable to represent the imaging process of a camera because lengths and angles are no longer preserved after perspective transformation. It is necessary another type of geometry to describe the camera transformation process. This type of geometry is known as *projective geometry*. Projective geometry allows a larger class of transformation than rotations and translations. However, fewer measurements, or *invariants*, are preserved [39, 47].

Points of the real n-dimensional projective space $\mathbb{P}^n$ are represented by $n+1$-component real column vectors $[x_1, \ldots, x_{n+1}]^T \in \mathbb{R}^{n+1}$, with the provision that at least one coordinate must be non-zero, and that $[x_1, \ldots, x_{n+1}]^T$ and $[\lambda x_1, \ldots, \lambda x_{n+1}]^T$ represent the same point of $\mathbb{P}^n$ for all $\lambda \neq 0$. The x_i are called *homogeneous coordinates* for the projective point.

The affine space $\mathbb{R}^n$ can be embedded isomorphically in $\mathbb{P}^n$ by the standard injection $[x_1, \ldots, x_n] \mapsto [x_1, \ldots, x_n, 1]$. Affine points can be recovered from projective ones with $x_{n+1} \neq 0$ by the mapping

$$[x_1, \ldots, x_n] \sim \left[\frac{x_1}{x_{n+1}}, \ldots, \frac{x_n}{x_{n+1}}, 1\right] \mapsto \left[\frac{x_1}{x_{n+1}}, \ldots, \frac{x_n}{x_{n+1}}\right].$$

A projective point with $x_{n+1} = 0$ corresponds to an ideal *point at infinity* in the $[x_1, \ldots, x_n]$ direction in the affine space. The invertible transformation $T : \mathbb{P}^n \to \mathbb{P}^n$ is called a *projective transformation*, *collineation*, *homography* or *perspectivity* and is completely determined by $n+2$ point correspondences.

The *invariants* upon projective transformations are:

- *Type:* points remain points and lines remain lines.

- *Incidence:* when a point lies on a line, it will lie also on a line in the transformed space.

- *Cross ratio:* this ratio is defined as

$$C_r\left(p_1, p_2; p_3, p_4\right) = \frac{d_{13}d_{24}}{d_{14}d_{23}},$$

where d_{ij} is the Euclidean distance between two points $p_i = [x_1^i, x_2^i, x_3^i]^T$ and $p_j = \left[x_1^j, x_2^j, x_3^j\right]^T$, given by

$$d_{ij} = \sqrt{\left(\frac{x_1^i}{x_3^i} - \frac{x_1^j}{x_3^j}\right)^2 + \left(\frac{x_2^i}{x_3^i} - \frac{x_2^j}{x_3^j}\right)^2},$$

as can be seen in Figure A.1. Therefore, if q_i, $i = 1, \ldots, 4$, denote the projective transformed points p_i, $i = 1, \ldots, 4$, then

$$C_r\left(p_1, p_2; p_3, p_4\right) = C_r\left(q_1, q_2; q_3, q_4\right).$$

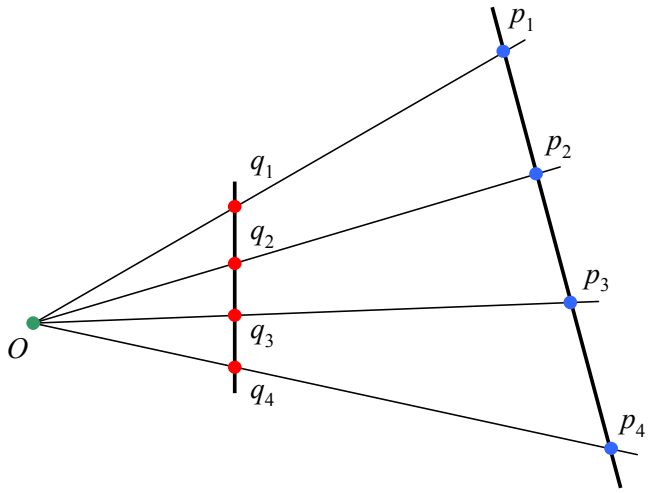


Figure A.1: Cross-ratio invariance.

The points v_1 and v_2 in Figure A.2 are called *vanishing points*, they could be at infinity and can be computed with the equations

$$v_1 = \left(p_{SE} \times p_{SW}\right) \times \left(p_{NE} \times p_{NW}\right),$$
$$v_2 = \left(p_{NE} \times p_{SE}\right) \times \left(p_{NW} \times p_{SW}\right).$$

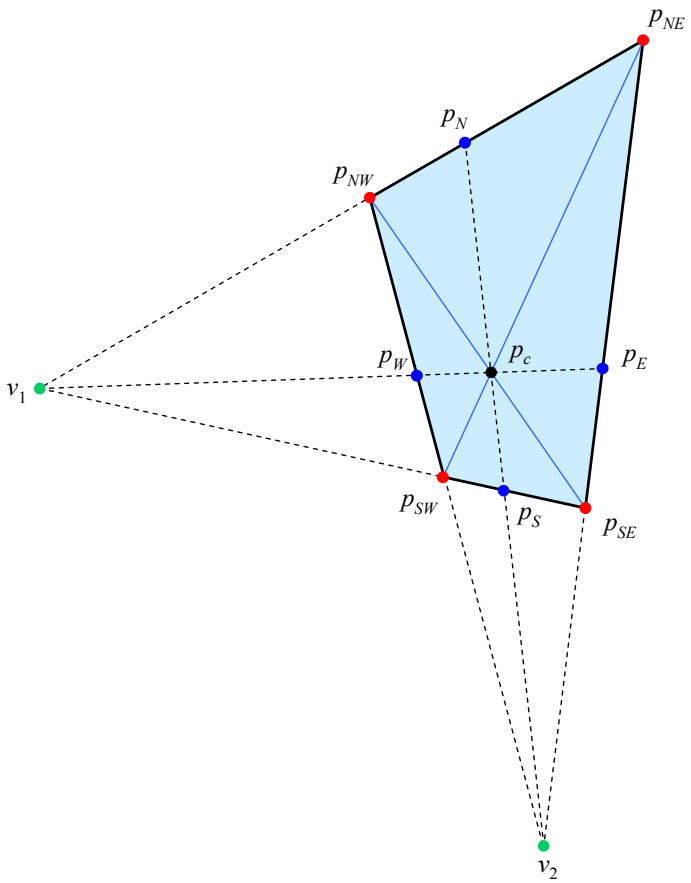


Figure A.2: Vanishing points.

Appendix B

Proofs of lemmas and theorems

B.1 Proof of Theorem 5.3.1

Proof. Let the filtered error signal $r(t) = [r_1(t), r_2(t)]^T \in \mathbb{R}^2$ be defined as

$$r(t) := \dot{e}_k(t) + Ke_k(t), \tag{B.1}$$

where $K = \text{diag}(k_1, k_2)$, k_1, $k_2 \in \mathbb{R}_{>0}$ are positive gain constants. Differentiating (B.1) with respect to time and using (5.32) or (5.37) yields

$$\dot{r}(t) = -\bar{u}_k(t) + \ddot{s}_k^d(t) - \bar{g}_0(x, u_k) - \bar{g}_v(x) V_k(t) + Ke_2(t). \tag{B.2}$$

Substituting (5.40) into (B.2), we have

$$\dot{r}(t) = -Kr(t) - \bar{g}_v(x) V_k(t) - \bar{u}_k^c(t). \tag{B.3}$$

Let the auxiliary function $L(t)$ be defined as

$$L(t) := -r^T(t) [\bar{g}_v(x) V_k(t) + \bar{u}_k^c(t)]. \tag{B.4}$$

It can be proved that $\int_{t_0}^{t} L(\tau)\, d\tau < \zeta_b$, see Lemma B.2.1 in this Appendix.

Let the function $P(t) \in \mathbb{R}_{\geq 0}$ be defined as

$$P(t) := \zeta_b - \int_{t_0}^{t} L(\tau)\, d\tau, \tag{B.5}$$

where ζ_b and $L(t)$ are defined according to Lemma B.2.1. Based on the non-negativity

of $P(t)$, let the Lyapunov function candidate $V(t)$ be defined as

$$V(t) := \frac{1}{2} r^T(t)\, r(t) + P(t). \tag{B.6}$$

Taking time derivative of (B.6), using (B.3), (B.4), and (B.5), it follows

$$\dot{V}(t) = -\, r^T(t)\, K r(t) \leq -\underline{k}_{12} \left\| r(t) \right\|^2, \tag{B.7}$$

with $\underline{k}_{12} := \min(k_1, k_2)$. Therefore, $V(t) \in \mathcal{L}_\infty \cap \mathcal{L}_2$, then $r(t)$ and $P(t) \in \mathcal{L}_\infty \cap \mathcal{L}_2$. From (B.1), it is clear that $e_k(t)$, $\dot{e}_k(t) \in \mathcal{L}_\infty$, and from (B.3), it can be seen that $\dot{r}(t) \in \mathcal{L}_\infty$; then $\ddot{V}(t) \in \mathcal{L}_\infty$. Considering that $V(t)$ is lower bounded by $\left\| r(t) \right\|^2$, by the Barbalat's lemma $\lim_{t\to\infty} \dot{V}(t) = 0$. This means that $\lim_{t\to\infty} \underline{k}_{12} \left\| r(t) \right\|^2 = 0$, or $r(t) \to 0$ as $t \to \infty$ by Rayleigh-Ritz theorem [79]. Since (B.1) is a stable first order differential equations driven by $r(t)$, it can be ensured that $\lim_{t\to\infty} e_k(t) = 0$ and $\lim_{t\to\infty} \dot{e}_k(t) = 0$. □

B.2 Lemma B.2.1

Lemma B.2.1. *Let the auxiliary function $L(t) \in \mathbb{R}$ be defined as*

$$L(t) := -r^T(t) \left[\bar{g}_v(t)\, V_k(t) + \bar{u}_k^c(t) \right]. \tag{B.8}$$

If the control gain $\beta = \mathrm{diag}(\beta_1,\, \beta_2)$ is selected to satisfy the sufficient condition

$$\beta_1 > \left| \bar{g}_v(t)\, V_k(t) \right|_1 + \frac{1}{k_2} \left| \frac{d}{d\tau} \left[\bar{g}_v(t)\, V_k(t) \right] \right|_1,$$
$$\beta_2 > \left| \bar{g}_v(t)\, V_k(t) \right|_2 + \frac{1}{k_1} \left| \frac{d}{d\tau} \left[\bar{g}_v(t)\, V_k(t) \right] \right|_2,$$

where K is given in (B.1) and $\underline{k}_{12} := \min(k_1, k_2)$. Then

$$\int_{t_0}^{t} L(\tau)\, d\tau < \zeta_b,$$

where the positive constant $\zeta_b \in \mathbb{R}_{>0}$ is defined as

$$\zeta_b := \left| e_k^T(t_0)\, \bar{g}_v(t_0)\, V_k(t_0) \right| + \beta_1 \left| e_{k,1}(t_0) \right| + \beta_2 \left| e_{k,2}(t_0) \right|. \tag{B.9}$$

Proof. Let $P_L(t)$ be defined as

$$P_L(t) := \int_{t_0}^{t} L(\tau)\, d\tau, \tag{B.10}$$

where $L(t)$ is defined in (B.8). Substituting (B.1) into (B.10), we have

$$\begin{aligned} P_L(t) = &- \int_{t_0}^{t} \frac{de_k^T(\tau)}{dt} g_v(\tau) V_k(\tau)\, d\tau - \int_{t_0}^{t} \frac{de_k^T(\tau)}{dt} \beta \operatorname{sign}(e_k(\tau))\, d\tau \\ &- \int_{t_0}^{t} e_k^T(\tau) K \bar{g}_v(\tau) V_k(\tau)\, d\tau - \int_{t_0}^{t} e_k^T(\tau) K \beta \operatorname{sign}(e_k(\tau))\, d\tau. \end{aligned} \tag{B.11}$$

Integrating by parts the first term on the right-hand side of (B.11), we obtain

$$\begin{aligned} P_L(t) = &\, e_k^T(t_0) \bar{g}_v(t_0) V_k(t_0) - e_k^T(t) \bar{g}_v(t) V_k(t) - \int_{t_0}^{t} \frac{de_k^T(\tau)}{d\tau} \beta \operatorname{sign}(e_k(\tau))\, d\tau \\ &- \int_{t_0}^{t} e_k^T(\tau) K \left\{ \bar{g}_v(\tau) V_k(\tau) - K^{-1} \frac{d}{d\tau} \left[\bar{g}_v(\tau) V_k(\tau) \right] \right\} d\tau \\ &- \int_{t_0}^{t} e_k^T(\tau) K \beta \operatorname{sign}(e_k(\tau))\, d\tau. \end{aligned}$$

Since, $\int_{t_0}^{t} \frac{dx(\tau)}{d\tau} \operatorname{sign}(x(\tau))\, d\tau = |x(t)| - |x(t_0)|$, we have

$$\begin{aligned} P_L(t) = &\, e_k^T(t_0) \bar{g}_v(t_0) V_k(t_0) + \beta_1 |e_{k,1}(t_0)| + \beta_2 |e_{k,2}(t_0)| \\ &- e_k^T(t) \bar{g}_v(t) V_k(t) - \beta_1 |e_{k,1}(t)| - \beta_2 |e_{k,2}(t)| \\ &- \int_{t_0}^{t} e_k^T(\tau) K \left\{ \bar{g}_v(\tau) V_k(\tau) - K^{-1} \frac{d}{d\tau} \left[\bar{g}_v(\tau) V_k(\tau) \right] + \beta \operatorname{sign}(e_k(\tau)) \right\} d\tau \\ = &\, \zeta_b - e_k^T(t) \bar{g}_v(t) V_k(t) - \beta_1 |e_{k,1}(t)| - \beta_2 |e_{k,2}(t)| \\ &- \int_{t_0}^{t} e^T(\tau) K \left\{ \bar{g}_v(\tau) V_k(\tau) - K^{-1} \frac{d}{d\tau} \left[\bar{g}_v(\tau) V_k(\tau) \right] + \beta \operatorname{sign}(e(\tau)) \right\} d\tau, \end{aligned}$$

with $\zeta_b := e_k^T(t_0) \bar{g}_v(t_0) V_k(t_0) + \beta_1 |e_{k,1}(t_0)| + \beta_2 |e_{k,2}(t_0)|$. As it can be easily seen, if β is chosen such that

$$\begin{aligned} \beta_1 &> \left| \bar{g}_v(t) V_k(t) \right|_1 + \frac{1}{k_2} \left| \frac{d}{d\tau} \left[\bar{g}_v(t) V_k(t) \right] \right|_1, \\ \beta_2 &> \left| \bar{g}_v(t) V_k(t) \right|_2 + \frac{1}{k_1} \left| \frac{d}{d\tau} \left[\bar{g}_v(t) V_k(t) \right] \right|_2, \end{aligned}$$

then the second and third right-hand terms are less than zero. Hence,

$$P_L(t) = \int_{t_0}^{t} L(\tau)\, d\tau \quad < \quad \zeta_b.$$

□

B.3 Proof of Theorem 5.3.3

Proof. The closed-loop formation tracking error dynamics under the OFB control law (5.52) is

$$\dot{r}(t) = -Kr(t) - 2K\eta_2(t) - \bar{g}_1(x, u_k)\,\eta_2 - g_v(x)\,V_k(t) - \bar{u}_k^c(t). \tag{B.12}$$

We prove that the combined closed loop system (5.47) and (B.12) is asymptotically stable. The proof is done in three steps: In the first step, it is shown that there exists an invariant set for the closed loop output feedback system based on a composite Lyapunov function. In the second step, we show that any trajectory will be trapped into this invariant set in finite time if the HGO constant ε is chosen small enough. In the third step, it is demonstrated that this invariant set is globally uniformly ultimately bounded (GUUB).

Let $\mathcal{R} = \mathbb{R}^2$ be the region of attraction of system (5.29) with control law (5.52). Let $\mathcal{D}_r$ be a compact set in the interior of $\mathcal{R}$. Let the compact set $\mathcal{D}_c$, which contains the origin of (B.12), be defined by $\mathcal{D}_c := \{r(t) \in \mathcal{R} | V(t) \leq c\}$, where $V(t)$ is defined in (B.6) and $c > \max_{r\in\mathcal{D}_r} V(t)$ is a small positive constant. The set $\mathcal{D}_c$ is a compact subset of $\mathcal{R}$ and $\mathcal{D}_r$ is in the interior of $\mathcal{D}_c$. Let the compact set $\mathcal{D}_\varepsilon$ be defined by $\mathcal{D}_\varepsilon := \{\eta(t) \in \mathbb{R}^4 | W(t) \leq \rho\varepsilon^2\}$, where $W(t)$ is given in (5.49), ρ is a positive constant to be selected later, and ε is the HGO constant. Finally, let the set Σ_c be defined as $\Sigma_c := \mathcal{D}_c \times \mathcal{D}_\varepsilon \in \mathbb{R}^6$.

The derivative of (B.6) for $(e(t), \dot{e}(t), \eta(t)) \in \{V(t) = c\} \times \mathcal{D}_\varepsilon$ verifies

$$\begin{aligned} \dot{V}(t) &\leq -\underline{k}_{12} \|r(t)\| [\|r(t)\| - \sigma_1 \|\eta(t)\|] \\ &\leq -\underline{k}_{12}\mu(\mu - L_1\varepsilon), \end{aligned} \tag{B.13}$$

with $\sigma_1 := \frac{\sigma_s + 2\bar{k}_{12}}{\underline{k}_{12}}$, $\sigma_s := \max_x \|g_1(z_k)\|$, $\bar{k}_{12} := \lambda_{\max}(K)$, $L_1 := \sigma_1\sqrt{\frac{\rho}{\lambda_{\min}(P_0)}}$, and $\mu = \min_{r\in\partial\mathcal{D}_c}\{\|r(t)\|\}$. Note that we dropped the subindex $'k'$ in $e_k(t)$ for simplicity.

Analyzing the derivative of (5.49) for $(e(t), \dot{e}(t), \eta(t)) \in \mathcal{D}_c \times \{W(t) = \rho\varepsilon^2\}$ we

find

$$\begin{aligned}\dot{W}(t) &\leq -\frac{\|\eta(t)\|^2}{2\varepsilon}\left(1-4\|P_0\|\sigma_s\varepsilon\right)-\frac{\|\eta(t)\|}{2\varepsilon}\left[\|\eta(t)\|-4\|P_0\|\sigma_v\varepsilon\right] \\ &\leq -\frac{\|\eta(t)\|^2}{2\varepsilon}\left(1-4\|P_0\|\sigma_s\varepsilon\right)-\frac{\|\eta(t)\|}{2}\left(\sqrt{\frac{\rho}{\|P_0\|}}-4\|P_0\|\sigma_v\right), \qquad \text{(B.14)}\end{aligned}$$

with $\sigma_v := \max_{x,V_k}\|\bar{g}_v(x)V_k(t)\|$.

Taking $\rho = 16\sigma_v^2\|P_0\|^3$ and $\varepsilon_1 = \min\left(\frac{1}{4\sigma_s\|P_0\|}, \frac{\mu}{L_1}\right)$, for every $0 < \varepsilon \leq \varepsilon_1$, we have

$$\dot{V}(t) \leq 0,$$

for $(e(t), \dot{e}(t), \eta(t)) \in \{V(t) = c\} \times \mathcal{D}_\varepsilon$, and

$$\dot{W}(t) \leq 0,$$

for $(e(t), \dot{e}(t), \eta(t)) \in \mathcal{D}_c \times \{W(t) = \rho\varepsilon^2\}$. Therefore, Σ_c is a positively invariant set.

Let us show that the trajectory of the system is trapped in this set. The initial state satisfy $(e(0), \dot{e}(0), \eta(0)) \in \mathcal{D}_r \times \mathcal{Q}$, where $\mathcal{Q}$ is a compact set such that $\mathcal{Q} \subseteq \mathbb{R}^4$. Using (5.45) it can be seen that

$$\|\eta(0)\| \leq \frac{c_3}{\varepsilon},$$

where $c_3 \in \mathbb{R}_{>0}$ is an appropriate constant. Because $(e(0), \dot{e}(0)) \in \mathcal{D}_r$, we have

$$\begin{aligned}\|e(t) - e(0)\| &\leq c_2^a t, \\ \|\dot{e}(t) - \dot{e}(0)\| &\leq c_2^b t, \\ \|r(t) - r(0)\| &\leq c_2 t,\end{aligned} \qquad \text{(B.15)}$$

where c_2^b, c_2^a, $c_2 \in \mathbb{R}_{>0}$ are some positive constants. Therefore, there exists a finite time T_0, independent of ε, such that $(e(t), \dot{e}(t)) \in \mathcal{D}_r$ for all $t \in [0, T_0]$. In consequence, when $t \in [0, T_0]$ and $W(\eta(t)) \geq \rho\varepsilon^2$, from (B.14), $\dot{W}(t) \leq -\frac{1}{2\varepsilon}\|\eta(t)\|^2$. Then

$$\dot{W}(t) \leq -\frac{\mu_1}{\varepsilon}W(t), \qquad \text{(B.16)}$$

with $\mu_1 := \frac{1}{2\|P_0\|}$. The solution for (B.16) is

$$W(t) \leq \frac{\mu_2}{\varepsilon^2}\exp\left(-\frac{\mu_1}{\varepsilon}t\right), \qquad \text{(B.17)}$$

with $\mu_2 := c_3^2 \|P_0\|$. As it can be seen from (B.17), $\lim_{t\to\infty} W(t) = 0$. Let T_ε be the time for which $W(t)$ falls below to $\rho\varepsilon^2$, it must satisfy

$$W(\eta(T_\varepsilon)) \leq \frac{\mu_2}{\varepsilon^2} \exp\left(-\frac{\mu_1}{\varepsilon} T_\varepsilon\right) \leq \rho\varepsilon^2, \tag{B.18}$$

In consequence, $T_\varepsilon \geq \frac{\varepsilon}{\mu_1} \ln\left(\frac{\mu_2}{\rho\varepsilon^4}\right)$ and $\lim_{\varepsilon\to 0} T_\varepsilon = 0$. Then, it is possible to choose ε_2 small enough such that $T_\varepsilon = \frac{1}{2}T_0$, for all $\varepsilon \in (0, \varepsilon_2]$. It follows that $W(\eta(T_\varepsilon)) < \rho\varepsilon^2$ for all $\varepsilon \in (0, \varepsilon_2]$. Choosing $\varepsilon_1^* = \min(\varepsilon_1, \varepsilon_2)$, the trajectory $(e(t), \dot{e}(t), \eta(t))$ enters into the invariant set Σ_c in $t \in [0, T_\varepsilon]$ and remains in Σ_c, for all $t \geq T_\varepsilon$ and every $0 < \varepsilon \leq \varepsilon_1^*$. Moreover, the trajectory $(e(t), \dot{e}(t), \eta(t))$ is bounded by (B.15) and (B.17), for $t \in [0, T_\varepsilon]$ and $\varepsilon \in (0, \varepsilon_1^*]$.

If the initial state satisfies $(e(0), \dot{e}(0), \eta(0)) \in \mathcal{D}_r \times \mathcal{Q}$, then the trajectory of the system will be inside Σ_c, for all $t \geq T_\varepsilon$ and $0 < \varepsilon \leq \varepsilon_1^*$. Because, from (B.17), $\lim_{\varepsilon\to 0} W(\eta(t)) = 0$, it is possible to find $\varepsilon_3 = \varepsilon_3(\delta_0) \leq \varepsilon_1^*$, for any given small value δ_0, such that

$$\|\eta(t)\| \leq \frac{\delta_0}{2}, \tag{B.19}$$

for $t \geq T_{\varepsilon_3} = T_{\varepsilon_3}(\delta_0)$.

Let the compact sets $\mathcal{D}_1$ and $\mathcal{D}_2$ be given by

$$\mathcal{D}_1 := \left\{(e(t), \dot{e}(t)) \in \mathcal{R} : \|r(t)\|^2 \leq 2L_1\varepsilon\right\}, \tag{B.20}$$

and

$$\mathcal{D}_2 := \{(e(t), \dot{e}(t)) \in \mathcal{R} : V(t) \leq \upsilon(\varepsilon)\}, \tag{B.21}$$

with $\upsilon(\varepsilon) := \max_{\|r\|^2 < 2L_1\varepsilon}\{V(t)\}$. If $(e(t), \dot{e}(t)) \notin \mathcal{D}_1$,

$$\dot{V}(t) \leq -\frac{1}{2}\underline{k}_{12} \|r(t)\|^2.$$

Let $\varepsilon_4 = \varepsilon_4(\delta_0)$ be chosen such that $\mathcal{D}_2$ is in the interior of $\mathcal{D}_c$ and

$$\mathcal{D}_2 \subset \left\{(e(t), \dot{e}(t)) \in \mathcal{R} : \|e(t)\| \leq \frac{1}{4}\delta_0, \|\dot{e}(t)\| \leq \frac{1}{4}\delta_0\right\}.$$

Then for all $(e(t), \dot{e}(t)) \in \mathcal{D}_c$, $(e(t), \dot{e}(t)) \notin \mathcal{D}_2$,

$$\dot{V}(t) \leq -\frac{1}{2}\underline{k}_{12} \|r(t)\|^2.$$

Therefore, the set $\Sigma_1 := \mathcal{D}_2 \times \mathcal{D}_\varepsilon$ is positively invariant and every trajectory in

$\mathcal{D}_c \times \mathcal{D}_\varepsilon$ enters Σ_1 in finite time $T_{\varepsilon_4} = T_{\varepsilon_4}(\delta_0)$, for $\varepsilon \in (0, \varepsilon_4]$. Let $\varepsilon_2^* = \min\{\varepsilon_3, \varepsilon_4\}$ and $T_1 = \max\{T_{\varepsilon_3}, T_{\varepsilon_4}\}$, therefore

$$\|e(t)\| + \|\dot{e}(t)\| + \|\eta(t)\| \leq \delta_0, \tag{B.22}$$

with $\delta_0 > 0$, $\varepsilon \in (0, \varepsilon_2^*]$, and $t \geq T_1$. Then $(e(t), \dot{e}(t), \eta(t))$ is GUUB. □

Appendix C

Definitions and mathematical background

This Appendix reviews key definitions and theorems for analysis stability of nonlinear systems. The interested reader is referred to references [48, 54, 58, 59, 69, 71, 80, 81, 91, 93, 118, 122, 134] for more detailed treatment.

C.1 Preliminaries

We consider smooth nonlinear control systems of the general form

$$\Sigma : \begin{cases} \dot{x} = f(x, u), \\ y = h(x), \end{cases} \tag{C.1}$$

where $x = [x_1, \ldots, x_n]^T$ are local coordinates for a smooth manifold $\mathcal{M}$, the functions f and h are smooth, $y = [y_1, \ldots, y_p]^T \in \mathbb{R}^p$ is the output vector, C^∞, and $u = [u_1, \ldots, u_m]^T \in \mathcal{U} \subseteq \mathbb{R}^m$ is an input vector signal, where $\mathcal{U}$ is convex and contains the origin. Admissible input signals are locally essentially bounded, Lebesgue measurable functions $u : [0, \infty) \mapsto \mathcal{U}$.

Alternative, we consider smooth affine nonlinear control systems of the general form

$$\Sigma : \begin{cases} \dot{x} = f_0(x) + \sum_{j=1}^{m} f_j(x)\, u_j, \\ y = h(x), \end{cases} \tag{C.2}$$

where $f_0, \ldots, f_m$ are smooth vector fields on $\mathcal{M}$. If $f_0(x)$ for all $x \in \mathcal{M}$, system (C.2) is called *driftless*.

Definition C.1.1 (Lipschitz condition). *A function $f : \mathcal{D} \subseteq \mathbb{R} \to \mathbb{R}$ is called* Lipschitz continuous *or is said to satisfy a* Lipschitz condition *if there exists a constant $K > 0$ such that for all x, $y \in \mathcal{D}$*

$$|f(x) - f(y)| \leq K |x - y| .$$

The smallest such K is called the *Lipschitz constant* of the function f. The function is called *locally Lipschitz continuous* if for every $x \in \mathcal{D}$ there exists an neighborhood $\mathcal{U}(x)$ so that f restricted to $\mathcal{U}$ is Lipschitz continuous.

Definition C.1.2 (Analytic function). *A function $f : \mathcal{D} \mapsto \mathbb{R}$ is said to be* analytic*, denoted C^ω, if it is $\mathcal{C}^\infty$ and for each point $x_0 \in \mathcal{D}$ there exists a neighborhood $\mathcal{B}(x_0)$, such that the Taylor series expansion of f at x_0 converges to $f(x)$ for all $x \in \mathcal{B}(x_0)$.*

Definition C.1.3. *A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to be of class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$. It is said to be of class $\mathcal{K}_\infty$ if $a = \infty$ and $\alpha(r) \to \infty$ as $r \to \infty$.*

Definition C.1.4. *A continuous function $\beta : [0, a) \times [0, \infty) \to [0, \infty)$ is said to be of class $\mathcal{KL}$ if $\beta(\cdot, t)$ is of class $\mathcal{K}$ for each fixed $t \geq 0$ and $\beta(s, t)$ decreases to 0 as $t \to \infty$ for each fixed $s \geq 0$.*

Definition C.1.5 (Coordinate chart). *A* coordinate chart *on a manifold $\mathcal{M}$ of dimension n is a pair $(\mathcal{U}, \phi)$, where ϕ is a homeomorphism of $\mathcal{U} \subset \mathcal{M}$ onto and open set of $\mathbb{R}^n$.*

Definition C.1.6 (Vector field). *Let $\mathcal{M}$ be a smooth manifold of dimension n and denote $T_x(\mathcal{M})$ the tangential space of $\mathcal{M}$. A* vector field *f on $\mathcal{M}$ is a mapping assigning to each point $p \in \mathcal{M}$ a tangent vector $f(p) \in T_x(\mathcal{M})$. A vector field f is* smooth *if for each $p \in \mathcal{M}$ there exists a coordinate chart $(\mathcal{U}, \phi)$ about p and n real-valued functions $f_1, \ldots, f_n$ defined on $\mathcal{U}$ such that,*

$$f(p) = \sum_{i=1}^{n} f_i(p) \left(\frac{\partial}{\partial \phi_i} \right)_q ,$$

for all $q \in \mathcal{U}$.

Definition C.1.7 (Lie derivative). *Let f be a smooth vector field on a manifold $\mathcal{M}$ of dimension n and λ a smooth real-valued function on $\mathcal{M}$. The Lie derivative of*

λ along f is a function $f\lambda : \mathcal{M} \to \mathbb{R}$, written $L_f\lambda$ and defined as

$$L_f\lambda\,(p) = f\,(p)\,\lambda.$$

In local coordinates, $L_f\lambda$ is represented by

$$L_f h\,(x_1, \ldots, x_n) = \left(\frac{\partial\lambda}{\partial x_1}, \ldots, \frac{\partial\lambda}{\partial x_n}\right)\begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}.$$

Definition C.1.8 (Lie bracket). *Let f and g be two smooth vector fields on a manifold $\mathcal{M}$ of dimension n and λ a smooth real-valued function on $\mathcal{M}$. The* Lie bracket *of f and g, denoted $[f, g]$, is a new vector field defined by*

$$[f, g]\,(h)\,(\lambda) = (L_f L_g\lambda)\,(p) - (L_g L_f\lambda)\,(p)\,.$$

The expression of $[f, g]$ in local coordinates is given by the n-vector

$$[f, g] = \frac{\partial g}{\partial x}f - \frac{\partial f}{\partial x}g.$$

It is convenient to write

$$ad_f g = [f, g]\,,$$

and to think ad_f as a linear operator.

Definition C.1.9 (Distribution). *Let $\mathcal{N}$ be an open subset of $\mathbb{R}^n$. A* distribution *Δ is a map which assigns a subspace $\Delta\,(x)$ of $\mathbb{R}^n$ to each $x \in \mathcal{N}$. The distribution generated by a set of d smooth vector fields $f_1\,(x)$, ..., $f_d\,(x)$ defined on $\mathcal{N}$, is defined as*

$$\Delta\,(x) = \text{span}\,\{f_1\,(x)\,, \ldots,\, f_d\,(x)\}\,.$$

Definition C.1.10 (Involutive distribution). *A distribution Δ is called* involutive *if $[f_1, f_2] \in \Delta$, whenever $f_1,\ f_2 \in \Delta\,(x)$.*

Definition C.1.11. *A nonsingular d-dimensional distribution Δ defined on an open set $\mathcal{N}$ of $\mathbb{R}^n$ is completely* integrable *if, or each point $x \in \mathcal{N}$ there exists a neighborhood $\mathcal{B}\,(x)$, and $n-d$ real-valued smooth functions λ_1, ..., λ_{n-d}, defined on $\mathcal{B}\,(x)$ such that*

$$\text{span}\,\{d\lambda_1, \ldots, d\lambda_{n-d}\} = \Delta^{\perp},$$

on $\mathcal{B}(x)$*, where* $\Delta^{\perp}$ *is the annihilator of* Δ *and* $d\lambda_i$*,* $i = 1, \ldots, n-d$*, is the covector field, called* exact differential*, given by*

$$d\lambda_i(x) = \frac{\partial \lambda_i(x)}{\partial x} = \left(\frac{\partial \lambda_i}{\partial x_1}, \ldots, \frac{\partial \lambda_i}{\partial x_n} \right).$$

Theorem C.1.12 (Frobenius). *A nonsingular distribution is completely integrable if and only if it is involutive.*

C.2 Controllability and observability

Definition C.2.1 (Reachability). *Considering system (C.1). A state* $x_1 \in \mathcal{M}$ *is called* reachable*, or* accessible*,* from $x_0 \in \mathcal{M}$*, if there exist a time* T *and a control* $u(t) \in \mathcal{U}$*, for all* $t \in [0, T]$*, which steers the system from* x_0*to* x_1*, with* $x(t) \in \mathcal{M}$ *for all* $t \in [0, T]$*. The set of points accessible from* x_0 *is denoted by* $\mathcal{R}(x_0)$

Definition C.2.2. *System (C.1) is said to be* controllable *at* x_0 *if* $\mathcal{R}(x_0) = \mathcal{M}$ *and controllable if* $\mathcal{R}(x) = \mathcal{M}$ *for every* $x \in \mathcal{M}$*.*

Definition C.2.3. *System (C.1) is said to be* locally controllable *at* x_0 *if for every neighborhood* $\mathcal{N}(x_0)$*,* $\mathcal{R}_{\mathcal{N}}(x_0)$ *is also a neighborhood of* x_0*. It is said that the system is* locally controllable *if it is locally controllable at every* $x \in \mathcal{M}$*.*

C.2.1 Observability

The observability problem answers the question of when and how it is possible to reconstruct the internal states from output measurements of the system.

Note that for nonlinear system, observer errors tending exponentially to zero do not guarantee that the system will be stable. The problem is known as *finite escape time*, which allows solutions to grow unbounded before the estimated states have converged.

Definition C.2.4 (Indistinguishability). *Two states* x_1*,* $x_2 \in \mathcal{M}$ *are said indistinguishable, denoted* $x_1 I x_2$ *for (C.2) if for every admissible input function* u *the output function* $t \mapsto y(t, 0, x_1, 0)$*,* $t \geq 0$*, of the system for initial state* $x(0) = x_1$*, and the output function* $t \mapsto y(t, 0, x_2, 0)$*,* $t \geq 0$*, of the system for initial state* $x(0) = x_2$*, are identical on their common domain of definition.*

Definition C.2.5 (Observability). *A system is called* observable *if* $x_1 I x_2$ *implies* $x_1 = x_2$*, with* x_1*,* $x_2 \in \mathcal{M}$*.*

Definition C.2.6 (Observation space). *The observation space $\mathcal{O}$ of system (C.2) is the linear space over $\mathbb{R}$ of the functions $L_{f_k}h_j(x)$, $j \in [1,p]$, $k = 0,1,2,\ldots$.*

Let $d\mathcal{O}(q)$ be the involutive *observability codistribution* defined by the observation space $\mathcal{O}$, with

$$d\mathcal{O}(q) = \operatorname{span}\{dH(q) |\, H \in \mathcal{O}\},\ q \in \mathcal{M}.$$

Now, it is possible to state the following theorem:

Theorem C.2.7 (Observability rank condition). *Let us consider system (C.2) with $\dim \mathcal{M} = n$. Assume that*

$$\dim(d\mathcal{O}(x)) = n,$$

then the system is locally observable at x.

C.3 Input-to-state stability

Definition C.3.1 (Input-to-state stability). *The system (C.1) is said to be input-to-state stable (ISS) if there exist a class $\mathcal{KL}$ function β and a class $\mathcal{K}$ function γ such that for any initial state $x(t_0)$ and any bounded input $u(t)$, the solution $x(t)$ exists for all $t \geq t_0$ and satisfies*

$$\|x(t)\| \leq \beta(\|x(t_0)\|, t - t_0) + \gamma\left(\sup_{t_0 \leq \tau \leq t} \|u(\tau)\|\right).$$

Theorem C.3.2. *Let $V : [0,\infty) \times \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function such that*

$$\alpha_1(\|x\|) \leq V(t,x) \leq \alpha_2(\|x\|),$$
$$\frac{\partial V}{\partial t} + \frac{\partial V}{\partial x} f(t,x,u) \leq -W_3(x), \quad \forall \|x\| \geq \rho(\|x\|) > 0,$$

$\forall (t,x,u) \in [0,\infty) \times \mathbb{R}^n \times \mathbb{R}^m$, where α_1, α_2 are class $\mathcal{K}_\infty$ functions, ρ is a class $\mathcal{K}$ function, and $W_3(x)$ is a continuous positive definite function on $\mathbb{R}^n$. Then, the system (C.1) is input-to-state stable with $\gamma = \alpha_1^{-1} \circ \alpha_2 \circ \rho$.

Definition C.3.3 (Relative degree). *A multivariable nonlinear system of the form (C.2) has a vector relative degree $\{r_1,\ldots,r_m\}$ at a point x_0 if*

1.

$$L_{f_j} L_{f_0}^k h_i(x) = 0,$$

for all $1 \leq j \leq m$*, for all* $k < r_i - 1$*, for all* $1 \leq i \leq m$*, and for all* x *in a neighborhood of* x_0*,*

2. the $m \times m$ *matrix*

$$A(x) = \begin{bmatrix} L_{f_1} L_{f_0}^{r_1-1} h_1(x) & \cdots & L_{f_2} L_{f_0}^{r_1-1} h_1(x) \\ L_{f_2} L_{f_0}^{r_2-1} h_2(x) & \cdots & L_{f_2} L_{f_0}^{r_2-1} h_2(x) \\ \vdots & \cdots & \vdots \\ L_{f_m} L_{f_0}^{r_m-1} h_m(x) & \cdots & L_{f_m} L_{f_0}^{r_m-1} h_m(x) \end{bmatrix},$$

is nonsingular at $x = x_0$*.*

Appendix D

Perturbations, relative dynamics and degrees

D.1 Leader-follower model with uncertain transformation

It should be noticed that the input transformation (5.7) depends on measured variables $\hat{\ell}_{ik}$, $\hat{\alpha}_{ik}$, and $\hat{\theta}_{ik}$. To observe how the error in these variables affect the real controller, let the input transformation be given by

$$\begin{bmatrix} v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} -\cos\hat{\beta}_{ik} & \hat{\ell}_{ik}\sin\hat{\beta}_{ik} \\ -\frac{1}{d}\sin\hat{\beta}_{ik} & -\frac{\hat{\ell}_{ik}}{d}\cos\hat{\beta}_{ik} \end{bmatrix} \begin{bmatrix} \bar{v}_k \\ \bar{\omega}_k \end{bmatrix}, \tag{D.1}$$

with $\hat{\beta}_{ik} := \hat{\alpha}_{ik} + \hat{\theta}_{ik}$ for brevity. Then, the dynamics is given by

$$\begin{aligned} \begin{bmatrix} \dot{\ell}_{ik} \\ \dot{\alpha}_{ik} \\ \dot{\theta}_{ik} \end{bmatrix} = & \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ \frac{1}{d}\sin\beta_{ik} & \frac{\ell_{ik}}{d}\cos\beta_{ik} \end{bmatrix} \begin{bmatrix} \bar{v}_k \\ \bar{\omega}_k \end{bmatrix} + \begin{bmatrix} \cos\alpha_{ik} & 0 \\ -\frac{\sin\alpha_{ik}}{\ell_{ik}} & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix} \\ & + \begin{bmatrix} \cos\hat{e}_\beta - 1 & \hat{\ell}_{ik}\sin\hat{e}_\beta \\ -\frac{\sin\hat{e}_\beta}{\ell_{ik}} & \left(1 - \frac{\hat{e}_\ell}{\ell_{ik}}\right)\cos\hat{e}_\beta - 1 \\ \frac{\left(\cos\hat{e}_\beta - 1\right)\sin\beta_{ik} - \cos\beta_{ik}\sin\hat{e}_\beta}{d} & \left(\cos\hat{e}_\beta - 1\right)\cos\beta_{ik} + \sin\beta_{ik}\sin\hat{e}_\beta \end{bmatrix} \begin{bmatrix} \bar{v}_k \\ \bar{\omega}_k \end{bmatrix}, \end{aligned} \tag{D.2}$$

with $\hat{e}_\beta := \beta_{ik} - \hat{\beta}_{ik}$, $\hat{e}_\ell := \ell_{ik} - \hat{\ell}_{ik}$. Therefore, system (5.9) is perturbed by a term of the form

$$\delta_\omega \leq \kappa_p \left(\left| \hat{e}_\ell \right|, \left| \hat{e}_\beta \right| \right) \left(\left| \bar{v}_k \right| + \left| \bar{\omega}_k \right| \right)$$

The rest of this work will take into account the generic perturbation form, without looking for an exact expression.

D.2 Internal dynamics

The internal dynamics in equation (5.9) is given by

$$\dot{\theta}_{ik} = \frac{1}{d} \sin\left(\alpha_{ik} + \theta_{ik}\right) \bar{v}_k + \frac{\ell_{ik}}{d} \cos\left(\alpha_{ik} + \theta_{ik}\right) \bar{\omega}_k + \omega_i. \tag{D.3}$$

From equation (5.9), it can be seen that

$$\begin{bmatrix} \bar{v}_k \\ \bar{\omega}_k \end{bmatrix} = \begin{bmatrix} -\cos\alpha_{ik} v_i + \dot{\ell}_{ik} \\ \frac{\sin\alpha_{ik}}{\ell_{ik}} v_i + \omega_i + \dot{\alpha}_{ik} \end{bmatrix}.$$

Then (D.3) is

$$\begin{aligned} \dot{\theta}_{ik}(t) = &-\frac{1}{d} \sin\theta_{ik} v_i(t) + \left[1 + \frac{\ell_{ik}}{d} \cos\left(\alpha_{ik} + \theta_{ik}\right)\right] \omega_i(t) \\ &+ \frac{1}{d} \sin\left(\alpha_{ik} + \theta_{ik}\right) \dot{\ell}_{ik}(t) + \frac{\ell_{ik}}{d} \cos\left(\alpha_{ik} + \theta_{ik}\right) \dot{\alpha}_{ik}(t). \end{aligned} \tag{D.4}$$

The system is said to be of minimum phase if $\dot{\theta}_{ik}(t)$ is stable, with $\dot{\ell}^d_{ik}(t) = \dot{\alpha}^d_{ik}(t) \equiv 0$, then

$$\dot{\theta}_{ik}(t) = -\frac{1}{d} \sin\theta_{ik} v_i(t) + \left[1 + \frac{\ell_{ik}}{d} \cos\left(\alpha_{ik} + \theta_{ik}\right)\right] \omega_i(t).$$

As can be seen, if $\omega_i(t) = 0$ and the speed of the leader $v_i(t)$ is positive, the system reduces to

$$\dot{\theta}_{ik}(t) = -\frac{1}{d} \sin\theta_{ik} v_i(t),$$

this system is clearly stable except for $\theta^d_{ik}(0) = \pi$. To analyze the effect of having a

negative leader velocity, let $\vartheta_{ik}(t)$ be defined as $\vartheta_{ik}(t) := \theta_{ik}(t) - \pi$, then

$$\dot{\vartheta}_{ik}(t) = \frac{1}{d}\sin\vartheta_{ik}v_i(t) + \left[1 - \frac{\ell_{ik}}{d}\cos(\alpha_{ik} + \vartheta_{ik})\right]\omega_i(t).$$

If $\omega_i(t) = 0$ and the speed of the leader $v_i(t)$ is negative, the system reduces to

$$\dot{\vartheta}_{ik}(t) = -\frac{1}{d}\sin\vartheta_{ik}\left|v_i(t)\right|,$$

then the system is stable except for $\vartheta_{ik}(t) = \pi$, that is, $\theta_{ik}(t) = 0$.

When $\omega_i(t) \neq 0$, Chen and Serrani present a relationship between $v_i(t)$ and $\omega_i(t)$ to guarantee internal dynamic stability [19]. Not such a result has been found for the two-leader-follower problem.

VITA

Omar Armando Adrián Orqueda

Candidate for the Degree of

Doctor of Philosophy

Thesis: VISION-BASED CONTROL OF MULTI-AGENT SYSTEMS

Major Field: Electrical Engineering

Biographical: Omar A.A. Orqueda received his B.Sc. and Ph.D. degrees from the Universidad Nacional del Sur, Bahía Blanca, Argentina, in 1996 and 2006, respectively. He has completed the Requirements for the Ph.D. degree at Oklahoma State University in December, 2006. He is currently a Research Assistant at School of Electrical and Computer Engineering, Oklahoma State University. His research interests include distributed coordination of multi-agent systems, motion planning, artificial vision, nonlinear control theory, nonlinear observers, and virtual environments.

Professional Memberships: Member of the Institute of Electrical and Electronics Engineers, Inc. (IEEE).

Name: Omar Armando Adrián Orqueda Date of Degree: December, 2006

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: VISION-BASED CONTROL OF MULTI-AGENT SYSTEMS

Pages in Study: 185 Candidate for the Degree of Doctor of Philosophy

Major Field: Electrical Engineering

Scope and Methodology of Study: Creating systems with multiple autonomous vehicles places severe demands on the design of decision-making supervisors, cooperative control schemes, and communication strategies. In last years, several approaches have been developed in the literature. Most of them solve the vehicle coordination problem assuming some kind of communications between team members. However, communications make the group sensitive to failure and restrict the applicability of the controllers to teams of friendly robots. This dissertation deals with the problem of designing decentralized controllers that use just local sensor information to achieve some group goals.

Findings and Conclusions: This dissertation presents a decentralized architecture for vision-based stabilization of unmanned vehicles moving in formation. The architecture consists of two main components: (*i*) a *vision system*, and (*ii*) *vision-based control algorithms*. The vision system is capable of recognizing and localizing robots. It is a model-based scheme composed of three main components: image acquisition and processing, robot identification, and pose estimation.

Using vision information, we address the problem of stabilizing a groups of mobile robots in leader- or two leader-follower formations. The strategies use relative pose between a robot and its designated leader or leaders to achieve formation objectives. Several leader-follower formation control algorithms, which ensure asymptotic coordinated motion, are described and compared. Lyapunov's stability theory-based analysis and numerical simulations in a realistic tridimensional environment show the stability properties of the control approaches.

Advisor's Approval: Dr. Rafael Fierro