University of New Mexico
UNM Digital Repository

Electrical and Computer Engineering ETDs

Engineering ETDs

1-30-2013

Real-time control architecture for a multi UAV test bed

Patricio J. Cruz Davalos

Follow this and additional works at: https://digitalrepository.unm.edu/ece_etds

Recommended Citation

Cruz Davalos, Patricio J.. "Real-time control architecture for a multi UAV test bed." (2013). https://digitalrepository.unm.edu/ece_etds/59

This Thesis is brought to you for free and open access by the Engineering ETDs at UNM Digital Repository. It has been accepted for inclusion in Electrical and Computer Engineering ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.

Patricio J. Cruz Davalos

Candidate

Electrical and Computer Engineering

Department

This thesis is approved, and it is acceptable in quality and form for publication:

Approved by the Thesis Committee:

Dr. Rafael Fierro, Chairperson

Dr. Meeko Oishi

Dr. Asal Naseri

Real-time Control Architecture for a Multi UAV Test Bed

by

Patricio J. Cruz Dávalos

B.S., Electronic and Control Engineering,
Escuela Politécnica Nacional, Quito-Ecuador, 2005

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Electrical Engineering

The University of New Mexico

Albuquerque, New Mexico

December, 2012

©2012, Patricio J. Cruz Dávalos

Dedication

To my wife, Magaly, for her encouragement and patience. I would never have achieved this goal without her support. Mi Magaly, este es un triunfo de ambos. Gracias por estar junto a mi.

To our children, Sebastián and Eimmy, our two little brave angels.
You two give us the best reason to keep on going.

To our second mom, Janet, for her help and kindness.
You will always be part of our family.

Acknowledgments

I am grateful to the Fulbright Program, especially with the Fulbright Commission in Ecuador, for being selected as a grantee in the Fulbright Faculty Development Program. I am also indebted to the National Polytechnic School (Escuela Politécnica Nacional) in Ecuador for their sponsorship and for providing partially the financial support for my family. Besides, I want to thank my family and my family-in-law for their encouragement despite the distance, I truly appreciate you.

I would like to thank my academic advisor, Professor Rafael Fierro, for giving me the opportunity to work in this research, and for his support and advise throughout my studies. I would also like to thank Professor Meeko Oishi and Professor Asal Naseri for their important suggestions and for being part of my committee. Lastly, I want to thank all the Marhes members for their invaluable help during the realization of this thesis.

This work was partially supported by NSF grant ECCS # 1027775 and by DOE URPR (University Research Program in Robotics) grant DE-FG52-04NA25590.

Real-time Control Architecture for a Multi UAV Test Bed

by

Patricio J. Cruz Dávalos

B.S., Electronic and Control Engineering,
Escuela Politécnica Nacional, Quito-Ecuador, 2005

M.S., Electrical Engineering, University of New Mexico, 2012

Abstract

The purpose of this thesis is to develop a control architecture running at real-time for a multi unmanned aerial vehicle test bed formed by three AscTec Hummingbird mini quadrotors. The reliable and reconfigurable architecture presented here has a FPGA-based embedded system as main controller. Under the implemented control system, different practical applications have been performed in the MARHES Lab at the University of New Mexico as part of its research in cooperative control of mobile aerial agents.

This thesis also covers the quadrotor modeling, the design of a position controller, the real-time architecture implementation and the experimental flight tests. A hybrid approach combining first-principles with system identification techniques is used for modeling the quadrotor due to the lack of information around the structure of the onboard controller designed by AscTec. The complete quadrotor model structure is formed by a black-box subsystem and a point-mass submodel. Experimental

data have been gathered for system identification and black-box submodel validation purposes; while the point-mass submodel is found applying rigid-body dynamics.

Using the dynamical model, a position control block based in lead-lag and PI compensators is developed and simulated. Improvements in trajectory tracking performance are achieved estimating the linear velocity of the aerial robot and incorporating velocity lead-lag compensators to the control approach. The velocity of the aerial robot is computed by numerical differentiation of position data. Simulation results to a variety of input signals of the control block in cascade with the complete dynamic model of the quadrotor are included.

The control block together with the velocity estimation is fully programmed in the embedded controller. A graphical user interface, GUI, as part of the architecture is designed to display real-time data of position and orientation streamed from the motion tracking system as well as to contain useful user controllers. This GUI facilitates that a single operator conducts and oversees all aspects of the different applications where one ore multiple quadrotors are used.

Experimental tests have helped to tune the control parameters determined by simulation. The performance of the whole architecture has been validated through a variety of practical applications. Autonomous take off, hovering and landing, target surveillance, trajectory tracking and suspended payload transportation are just some of the applications carried out employing the real-time control architecture proposed in this thesis.

Contents

List of Figures xiii

List of Tables xvii

Glossary xviii

1 Introduction **1**

1.1 Motivation 1

1.2 Problem Statement 4

1.3 Related Work 4

1.4 Organization of the Thesis 5

2 System Overview **7**

2.1 The MARHES Laboratory 7

2.2 The AscTec Hummingbird Autopilot 8

2.2.1 Main Components 9

2.2.2 On-board Serial Interface 13

2.2.3 Control Input Ranges 15

2.3 The VICON MX System 16

2.4 The NI CompactRIO System 17

3 Quadrotor Modeling 20

3.1 Basic Concepts and Assumptions 20

3.1.1 Coordinate Systems 21

3.1.2 Orientation and Rotations 22

3.1.3 Quadrotor Basic Maneuvers 23

3.2 Model Structure 26

3.2.1 Point-Mass Model 27

3.2.2 Black-Box Model 29

3.2.3 Complete Dynamical Model 30

3.3 Modeling for Control Purposes 31

3.3.1 Linear Model 33

4 System Identification and Parameter Estimation 36

4.1 Thrust Force Relation 36

4.1.1 Experimental Test 37

4.1.2 Polynomial Relation 38

4.2 Black-Box Model Identification 41

4.2.1 Acquiring Data 42

4.2.2 Estimating the Model 43

4.2.3 Model Validation 46

5 Controller Design **48**

5.1 Position and Heading Controller 49

5.1.1 Simulations 50

5.2 Modified X and Y Controllers 51

5.2.1 Velocity Estimation 51

5.2.2 Modifications 53

5.2.3 Simulations 53

6 Control System Implementation **56**

6.1 Hardware Architecture 56

6.2 Software Architecture 57

6.2.1 VICON Interface VI 59

6.2.2 Quadrotor Interface VI 60

6.3 System Adaptations 61

6.3.1 Trajectory Tracking 62

6.3.2 Adding a Mobile Target 62

6.3.3 Managing Multiple Quadrotors 63

7 Experimental Results and Applications 66

7.1 Tuning the Controllers 67

7.2 Experimental Tests 68

7.2.1 Autonomous Take off and Landing 68

7.2.2 Hovering Performance 72

7.2.3 Aerial Tracking of a Mobile Target 72

7.2.4 Trajectory Tracking 75

7.3 Practical Applications 76

7.3.1 Mobile Wireless Heterogeneous System 76

7.3.2 Surveilling using Mobile Sensor Agents 78

7.3.3 Agile Load Transportation 80

7.3.4 Cooperative Lifting of a Suspended Payload 80

8 Conclusions, Contributions and Improvements 83

8.1 Conclusions 83

8.2 Contributions 84

8.3 Limitations and Improvements 85

A Thrust Measurement Data 87

B Data for System Identification 93

B.1 Stimulus Signals and Acquired Data 93

B.2 Model Validation Data 94

C Simulation Results **98**

Bibliography **106**

List of Figures

1.1 A real quadrotor during hovering and its virtual representation. . . . 2

1.2 Quadrotor fire fighters in an virtual emergency mission. 3

2.1 The Marhes test bed. 8

2.2 The X-CSM mechanical structure. 10

2.3 The AscTec Hummingbird power board and the AutoPilot board. . 11

2.4 The AscTec 3D-MAG and the GPS unit. 12

2.5 The X-BL-52S motor and the X-BLDC brushless motor controller. . 13

2.6 An XBee module and its adapter to RS232. 14

2.7 Command frame over the serial link. 14

2.8 Scheme of the VICON MX system installed in the Marhes Lab. . . 16

2.9 Screen shots from the VICON Tracker software. 17

2.10 The NI CompactRIO system. 18

3.1 Fixed-body and world coordinate systems. 21

3.2 Quadrotor structure at hovering position. 24

3.3 Quadrotor movement in the XY_W plane. 25

3.4 Structure of the quadrotor model. 26

3.5 Forces and momentums acting on the quadrotor. 28

3.6 Complete quadrotor dynamic model. 32

4.1 U and $\mathrm{S_{Th}}$ relation based in the third order polynomial approximation. 39

4.2 U and $\mathrm{S_{Th}}$ relation based in the fourth order polynomial approximation. 40

4.3 Scheme of the system identification procedure. 41

4.4 Stimulus signal and the acquired data for the yaw rate $\dot{\psi}_B$. 44

4.5 Quadrotor attached to the metal support. 44

4.6 Input Signal, Acquired Data and Model Validation for the roll angle ϕ_B. 47

5.1 Main cascade control diagram. 48

5.2 Block diagram of the position and heading controller. 49

5.3 Simulations of the step response for the position controllers. 52

5.4 Block diagram showing the controller modifications. 54

5.5 Simulations of the step response for the X and Y modified controllers. 55

6.1 Hardware architecture. 57

6.2 Software architecture - basic structure. 58

6.3 Front panel of the VICON interface VI. 59

6.4 Front panel of the quadrotor interface VI. 61

6.5 Software architecture - trajectory tracking. 62

6.6 Software architecture - adding a mobile target. 63

6.7 Hardware architecture - multiple UAVs. 64

6.8 Software architecture - multiple UAVs. 65

7.1 Experimental results for the take off maneuver. 69

7.2 Experimental results for the landing maneuver. 70

7.3 Experimental results during hovering. 71

7.4 Aerial tracking of a mobile ground robot. 72

7.5 Results from tracking a ground robot that is describing a circular trajectory. 73

7.6 Results from tracking a ground robot that is describing an arbitrary trajectory. 74

7.7 Results from tracking a square path. 75

7.8 A cooperative heterogeneous mobile wireless mechatronic system. . . 77

7.9 Mobile agents and experimental results for surveilling using MSA. . 79

7.10 Set-up and experimental results for swing-free trajectory tracking. . 81

7.11 Picture and block diagram of the application about lifting an overweighted payload. 82

A.1 Thrust measurement data plot for the ten experimental tests. 87

B.1 Stimulus signal and acquired data for the roll angle ϕ_B. 93

B.2 Stimulus signal and acquired data for the pitch angle θ_B. 94

B.3 Stimulus signal and acquired data for the yaw rate $\dot{\psi}_B$. 94

B.4 Input signal, acquired data and model validation for the roll angle ϕ_B. 95

B.5 Input signal, acquired data and model validation for the pitch angle θ_B. 96

B.6 Input signal, acquired data and model validation for the yaw rate $\dot{\psi}_B$. 97

C.1 Simulation responses for different inputs-X controller. 99

C.2 Simulation responses for different inputs-Y controller. 100

C.3 Simulation responses for different inputs-Z controller. 101

C.4 Simulation responses for step and square input-Modified X controller. 102

C.5 Simulation responses for sawtooth and sinusoidal input-Modified X controller. 103

C.6 Simulation responses for step and square input-Modified Y controller. 104

C.7 Simulation responses for sawtooth and sinusoidal input-Modified Y controller. 105

List of Tables

2.1 General specifications and components of the AscTec Hummingbird Autopilot. 9

2.2 Connection settings for the serial interface. 14

4.1 Mean of the Thrust Force, U for each value of Thrust Input, $\mathrm{S_{Th}}$. . . 38

5.1 Parameters of the position and heading controllers - simulation results. 51

5.2 Parameters of modified controllers - simulation results. 53

7.1 Parameters of the position and heading controllers - experimental tuning. 67

7.2 Parameters of the modified controllers - experimental tuning. 67

A.1 Measurement data experiments 1 and 2. 88

A.2 Measurement data experiments 3 and 4. 89

A.3 Measurement data experiments 5 and 6. 90

A.4 Measurement data experiments 7 and 8. 91

A.5 Measurement data experiments 9 and 10. 92

Glossary

${}^{W}\mathbf{a}$	Vector $\mathbf{a}$ in the coordinate system $\{W\}$.
${}^{W}\mathbf{r}_{cg} =^{W} \mathbf{r}$	Quadrotor's center of mass position vector respect to the coordinate system $\{W\}$, ${}^{W}\mathbf{r}_{cg} =^{W} \mathbf{r} = [x \quad y \quad z]^{T}$.
${}^{W}\mathbf{\Theta}$	Quadrotor's orientation vector respect to the coordinate system $\{W\}$.
ϕ, θ, and ψ	Roll, pitch, and yaw rotational angles, respectively.
${}^{W}\mathbf{R}_{B}$	Rotational matrix from the fixed-body coordinate system, $\{B\}$, to the world coordinate system, $\{W\}$.
$c\alpha, s\alpha$, and $t\alpha$	Abbreviations of cosine α, sine α, and tangent α, respectively.
m	Total mass of the quadrotor.
$\mathbf{M}$	Momentum acting on the quadrotor.
$\mathbf{\Omega}$	Angular velocity of a quadrotor's motor.
$\mathbf{\Omega}^{des}$	Desired angular velocity for a quadrotor's motor.
U	Total vertical force or total thrust respect to the fixed-body coordinate system, $\{B\}$.

$\mathrm{U_h}$	Total vertical for at hovering.
$\mathrm{S}_\phi, \mathrm{S}_\theta, \mathrm{S}_\psi$, and $\mathrm{S_{Th}}$	Roll, pitch, yaw and thrust input values, respectively.
$\mathbf{r}^{des}$	Desired quadrotor's position vector.
$\dot{\mathbf{r}}^{des}$	Desired quadrotor's velocity vector.
${}^{B}\boldsymbol{\omega}$	Quadrotor's angular velocity respect to the coordinate system $\{B\}$, ${}^{B}\boldsymbol{\omega} = \begin{bmatrix} \dot{\phi}_B & \dot{\theta}_B & \dot{\psi}_B \end{bmatrix}^T = [p \quad q \quad r]^T$.
$V(s)$	Control Signal, output from the controller.
$E(s)$	Error Signal, difference between the reference and the response values.
WP	Working point value of $\mathrm{S_{Th}}$ to get $\mathrm{U_h}$.

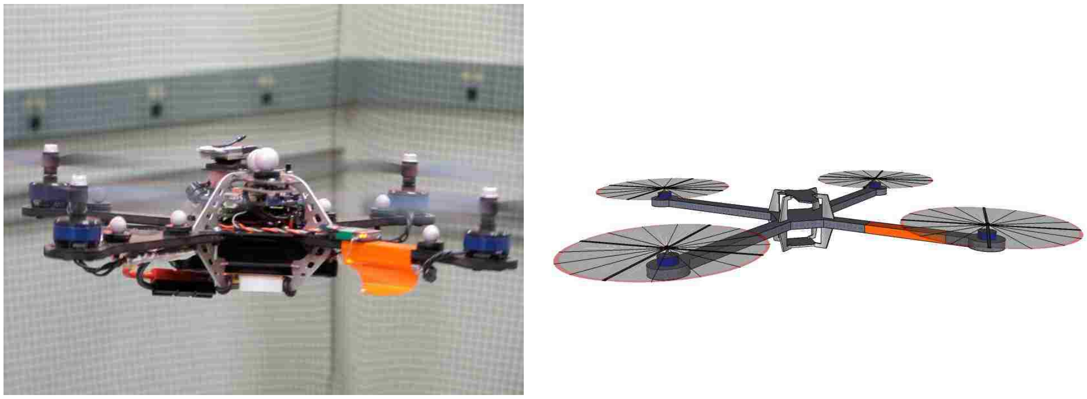
Chapter 1

Introduction

1.1 Motivation

In the last decade, Unmanned Aerial Vehicles, UAVs, have become promising mobile platforms which are capable of navigating semi-autonomously or full-autonomously within uncertain environments. In addition, the level of autonomy and the flexible technology of these flying robots have rapidly evolved making it possible to accomplish successfully a wide spectrum of indoor and outdoor tasks. These include environmental monitoring and surveillance applications, disaster-response operations, and search and rescue missions. For example, radio controlled UAVs carrying radiation sensors and video cameras were employed recently to monitor, diagnose and evaluate the situation at Japan's Fukushima Daiichi nuclear plant facility [1], an unmanned aircraft autonomous system has helped to monitor forest fires [2]; and a UAV cooperating with a USV (unmanned surface vehicle) has carried out sea robot-assisted inspection [3]. A detailed description of the current state of the art in autonomous rotorcraft UAVs is presented in the comprehensive survey [4] where also a classification of these aerial vehicles is included.

One special type of aerial vehicle, the quadrotor (see Figure 1.1(a)), has not only the capability of taking off and landing in a very limited area, but also of carrying more weight than other aerial platforms thanks to its four propellers. Furthermore, these unmanned rotorcraft vehicles have become one of the most popular platforms for UAV research. In fact, diverse research groups [5, 6, 7, 8] have included quadrotors as active part of their robotic test beds. Existing experimental results range from basic hovering [9, 10] and trajectory tracking [11] to formation [12] and surveillance [13], and including even aggressive maneuvers [14] and aerobatic flips [15]. Moreover, there are some novel applications in the communications area where a quadrotor is considered as an autonomous antenna relay [16] or a team of quadrotors is configured as an array of antennas [17].



(a) The AscTec Hummingbird quadrotor. (b) Virtual Hummingbird quadrotor.

Figure 1.1: A real quadrotor during hovering and its virtual representation.

Most of the experiments mentioned above employ the UAV for passive observation or simple motion without any physical interaction between the quadrotor and its surrounding objects. Nevertheless, aerial manipulation has been extremely important for military, and industrial purposes as wall as in emergency rescue missions such as fire fighting (see Figure 1.2). For instance, the safe aerial transport of a victim from a dangerous area is vital in an emergency response. Therefore, in the recent years, the functionality of UAVs has advanced beyond simple environmental sensing to manipulation of their external environment introducing a wide novel set of practical applications. In fact, individual or cooperative transport of a suspend load [18, 19, 20], aerial grasping and manipulation [21, 22], applying force to a wall [23], and building structures [24, 25] are some examples where aerial robotic platforms interact with their surroundings and/or with external objects.

All these practical examples show how active the UAV research within the control community is. For that reason, this thesis presents a real-time control architecture for a multi UAV test bed. Specifically, this architecture is implemented in the Marhes Laboratory at the Electrical and Computer Department of the University of New Mexico.

Figure 1.2: Quadrotor fire fighters in an virtual emergency mission.

1.2 Problem Statement

The main goal of this thesis is the development and implementation of a real-time control architecture for a mini UAV quadrotor (according to the categories presented in [4]) in an indoor environment. Furthermore, this architecture must be distributed and modular such that more similar aerial vehicles can be easily added. In addition, a graphical user interface, GUI, must be created for single-operator monitoring and supervision purposes. This GUI displays real-time position and orientation data of the aerial vehicle. Besides, it must include all the necessary controls to enter reference values and desired waypoints as well as to load pre-generated 3D trajectories (position and velocity data).

Diverse subgoals are involved around the previous main purpose. First, a viable and adequate model of the quadrotor is obtained and simulated. After, a linear position controller is designed and implemented. Indeed, the controller has to enable the quadrotor to perform stable hovering, accurate waypoint tracking and reliable trajectory tracking. Furthermore, diverse practical experiments should be carried out to test the controller performance. In particular, a basic cooperative task using two quadrotors should be outlined and implemented to prove the expansion capability of the control scheme.

1.3 Related Work

The growing interest in mini and micro quadrotors has brought a lot of challenges around the hardware and software within this aerial platform. Specifically, an increasing demand is placed on developing its guidance, navigation and control (GN&C) systems. In addition, the design of reliable and robust controllers for this aerial vehicle is challenging due to its nonlinear, complex and underactuated dynamical model. As a result, the quadrotor control problem has attracted the attention of many researches from both the control and robotic communities because it presents an excellent opportunity for developing and testing new control design methodologies [4].

Most published papers on modeling and control of a mini or micro quadrotor use a model obtained considering a rigid-body dynamics evolving in a 3D space that is described by the Newton-Euler equations of motion [7, 6, 9, 11, 26]. On the other hand, different approaches use a combination of the rigid-body dynamic equations with a system-identification approach as a modeling technique [10, 27, 28, 29, 30]. A system-identification method is applied to estimate a linear model of some uncertainties that could appear in the quadrotor dynamic model such as unknown aerodynamic coefficients.

The literature includes a variety of linear and nonlinear control techniques for controlling mini and micro quadrotors, . The most applied linear controllers are PID (proportional-integral-derivative) or just PD (proportional-derivative) controllers [7, 11, 26, 28, 31], linear quadratic regulator (LQR) controllers [10, 28, 29, 30, 31] and robust H_∞ controllers [29]. Conversely, also nonlinear control systems have been developed and tested. For instance, a backstepping technique is applied in [32, 33] and a sliding mode control is employed in [32, 34, 35]. For this thesis, cascade lead-lag compensators are principally used to implement a reliable position controller.

1.4 Organization of the Thesis

The remainder of this thesis is organized as follows. Chapter 2 provides an overview of the different systems, quadrotors, sensors and controllers, that are part of the Marhes Laboratory and of the real-time control architecture. Subsequently, Chapter 3 presents the quadrotor model that is used in this thesis. Since part of the quadrotor model was determined using system identification and parameter estimation, this is discussed in Chapter 4. The controller design and its corresponding simulations are described in Chapter 5. Chapter 6 outlines the control system implementation. The different experiments and applications to evaluate the performance of the control architecture and their results are given in Chapter 7. Finally, Chapter 8 discusses the conclusions and future possible improvements to the real-time architecture proposed herein.

Chapter 2

System Overview

2.1 The Marhes Laboratory

The Multi-Agent Robotic Hybrid and Embedded Systems, Marhes, Laboratory is a clear example of a heterogeneous mechatronic system since its test bed consists of unmanned aerial vehicles and unmanned ground vehicles, UGV. Indeed, five Pioneer P3-AT [36] and 10 TXT (under development) [37] are part of the ground robotic platform; while, three AscTec Hummingbird Autopilot quadrotors [38] and one AR.Drone Parrot [39] are part of the aerial robotic platform. Because this thesis is focus on the aerial robotic platform, specifically the AscTec Hummingbird quadrotor, no more explanation about the other platforms will be presented. In addition, the laboratory is equipped with a VICON MX system [40] that is a high-precision motion capture system. This system is installed within a capture room surrounded by a protective net. Figure 2.1 shows one of the Hummingbird quadrotors hovering over a Pioneer robot while they are overseen by a motion tracking system within the capture room.

The laboratory also counts with five NI CompactRio (cRIO) systems, that could be employed as reconfigurable control and monitoring systems [41]. In fact, the embedded real-time controller of this system will be used as part of the architecture implemented in this work.

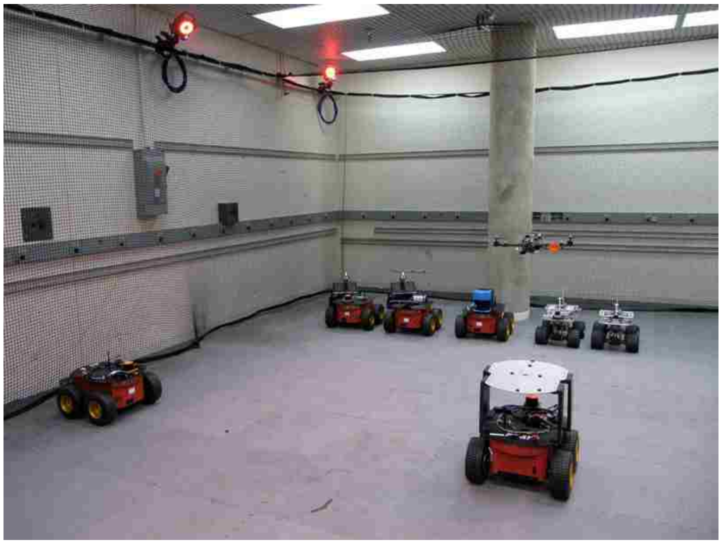


Figure 2.1: The MARHES test bed.

2.2 The AscTec Hummingbird Autopilot

Figure 1.1(a) shows this aerial system flying in the capture room. This system is a fusion between the robust Hummingbird frame with the AscTec AutoPilot sensor board [38]. The aerial vehicle consists of a carbon fiber-balse wood sandwich material with two pairs of counter-rotating fixed-pitch blades. Indeed, a general overview of the most relevant specifications and components are summarized in the Table 2.1. Based on these specifications, the AscTec Hummingbird is a Mini RUAS (Rotorcraft Unmanned Aerial System) according to the categories proposed in [4].

Size (Diameter)	36.5 cm
Propellers Size	19.8 cm
Weight (without battery)	353 g
Payload	up to 350 g
Maximum Thrust	1320 g at 12.6 V
Flight Time	up to 30 min. (without Payload, 1.8 Ah battery)
Components	• X-CSM mechanical frame • AscTec Hummingbird PowerBoard • AscTec Autopilot (sensing and flight control unit) • AscTec 3D-MAG (triple axial compass module) • X-BL-52 motors • X-BLDC brushless motor controller • XBee 2.4 GHz Module

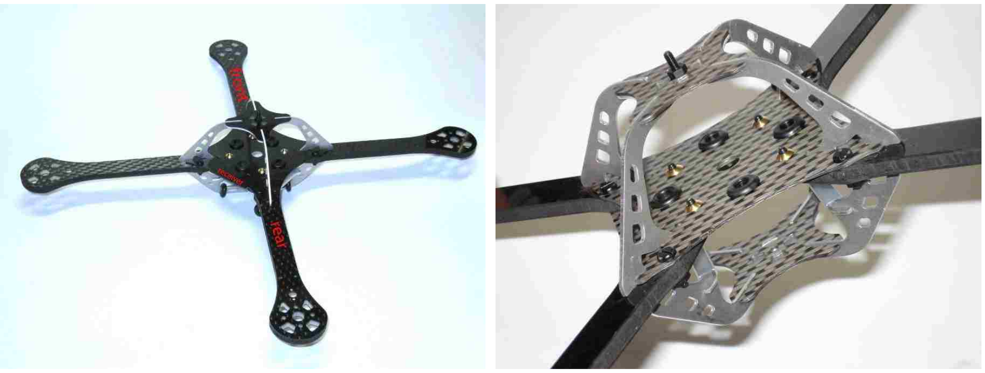
Table 2.1: General specifications and components of the AscTec Hummingbird Autopilot.

2.2.1 Main Components

In this section, some information about the main parts of the AscTec Hummingbird Autopilot quadrotor are provided. This information as well as of the different figures are obtained from the Hummingbird user's manual [42] and they are presented here just as a reference to understand how this aerial platform can be controlled. Additional details about the quadrotor parts and its on-board controller can be also obtained in [43].

X-CSM

X-CSM is the mechanical frame of the quadrotor. It consists of four light weight carbon fiber booms and a magnesium core structure. The magnesium core improves the structural integrity and houses the electronics and battery. In fact, the use of light weight but sturdy material renders the platform durable and rigid. For that reason, the quadrotor is capable of enduring small crashes without permanent damage. Figure 2.2 illustrates one the complete frame with a close up of the magnesium core.



(a) Complete frame.

(b) Magnesium core.

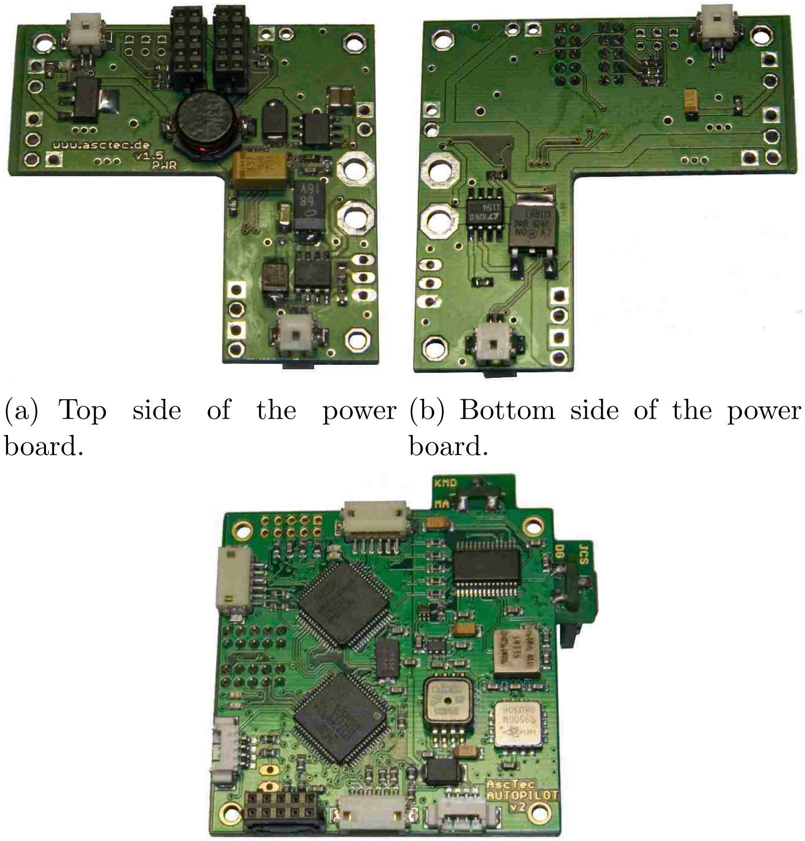
Figure 2.2: The X-CSM mechanical structure.

Power Board

It is used to distribute power and communication lines to all motor controllers. Figures 2.3(a) and 2.3(b) depicts the power board module. It comprises a switching power regulator to generate a stable 6 V supply for the AscTec AutoPilot board, a high-power MOSFET to switch the motor current and a 5 V regulator to supply any custom payload. The ON-OFF switch is designed active low, so if for some reason it breaks or loses connection the vehicle is switched ON by default.

AscTec AutoPilot

The AscTec Autopilot is the flight control unit (see Figure 2.3(c)) that reads sensor data, compute angular velocities and angles in all axes (roll, pitch and yaw) and



(a) Top side of the power board.

(b) Bottom side of the power board.

(c) The AscTec AutoPilot.

Figure 2.3: The AscTec Hummingbird power board and the AutoPilot board.

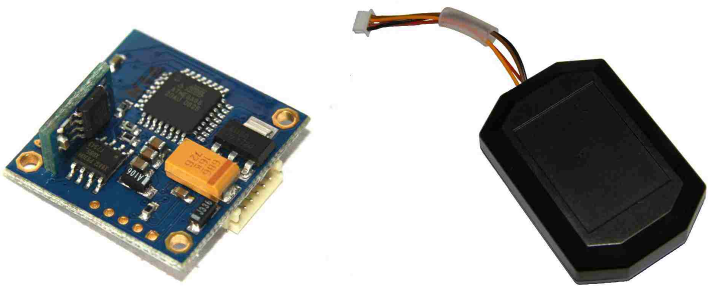
runs three independent PD loops for each axis. Subsequently, the control outputs are combined to compute a desired speed for each motor which is transmitted to the respective motor controller. All these processes are done with a control loop frequency of 1 kHz.

3D-MAG

It is the triple-axial compass module used to determine the vehicle heading by measuring the Earth's magnetic field. The AscTec AutoPilot mathematically transforms the measured vector into the horizontal plane and can thus determine the correct heading even if the vehicle is tilted in pitch or roll. The 3D-MAG is shown in Figure 2.4(a).

GPS unit

The GPS unit (Figure 2.4(b)) is mounted on top of the X-CSM core and used for autonomous outdoor flight. However, this unit will be disconnected for the current implementation.

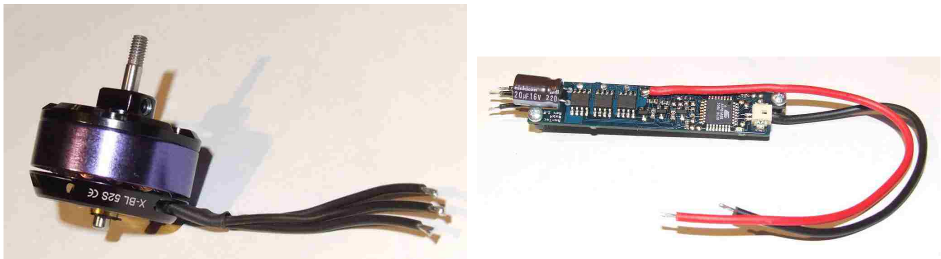


(a) The 3D-MAG (b) The GPS unit

Figure 2.4: The AscTec 3D-MAG and the GPS unit.

Motors and Drivers

The X-BL-52S motors are custom-built for the AscTec Hummingbird. Every motor (in total four motors on the platform) is controlled by an independent brushless motor controller. One of the four drivers along with one of the motors are shown in Figure 2.5.



(a) The X-BL-52S motor.

(b) The X-BLDC motor driver.

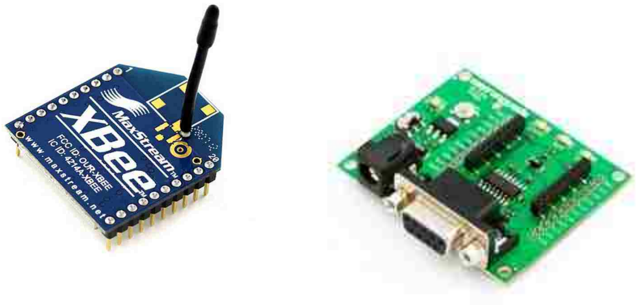
Figure 2.5: The X-BL-52S motor and the X-BLDC brushless motor controller.

2.2.2 On-board Serial Interface

The system is designed to be manually operated through a six or more channels R/C transmitter or to be autonomously controlled through a wireless serial link. Furthermore, it can be used indoors by employing an external optical positioning system. For the purpose of this thesis, the Hummingbird quadrotor will be controlled through the wireless serial link and its on-board attitude controller will be enabled. Therefore, a summary about the operation and data protocol of the on-board serial interface is explained next. A complete discussion of this interface could be found in [42].

The wireless serial link is established by using a pair of cost-effective XBee ZB ZigBee modules [44] (see Figure 2.6) which manage a serial data interface. One module is connected directly to the serial port 0 of the low-level controller in the AscTec Autopilot Board and the other should be connected to the base station that sends the control commands. For a complete description, please refer to Section 4 of the User's Manual [42]. Moreover, the connection settings of the serial interface with the module connected to the ground station are reproduced here in the Table 2.2.

As indicated in [42], the serial communication must be enabled by its corres-



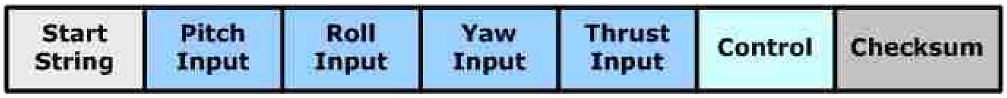
(a) XBee module. (b) RS232 adapter.

Figure 2.6: An XBee module and its adapter to RS232.

Baudrate	57600
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	Hardware

Table 2.2: Connection settings for the serial interface.

ponding switch in the R/C transmitter. Furthermore, during operation through the serial command interface the R/C transmiter has to stay ON all the time to enable manual operation if the autonomous control is not working as desired. In addition, the command data should be sent to the vehicle with a minimum rate of 10 Hz over the wireless link. If the data frequency is less than 10 Hz or the transmission stops completely, the system will automatically switch back to manual operation over the R/C transmitter.



Start String	Pitch Input	Roll Input	Yaw Input	Thrust Input	Control	Checksum

Figure 2.7: Command frame over the serial link.

The command frame that should be sent over the serial link is shown in Figure

2.7. Besides, each field of this frame has the following format:

Start String $> * >$di

Pitch Input is a signed integer between -2047 and 2047 with the value of 0 as neutral.

Roll Input is a signed integer between -2047 and 2047 with the value of 0 as neutral.

Yaw Input is a signed integer between -2047 and 2047 with the value of 0 as neutral.

Thrust Input is an unsigned integer between 0 and 4095.

Control is a byte whose bits can be used to enable only one axis at a time and thus to control manually the other axes. This is:

bit 0: pitch control enabled.

bit 1: roll control enabled.

bit 2: yaw control enabled.

bit 3: thrust control enabled.

Checksum is calculated by Checksum = Pitch Input + Roll Input + Yaw Input + Thrust Input + Control + 0x$AAAA$.

2.2.3 Control Input Ranges

Because of the reduced free space available in the capture room, the values for each one of the previous inputs have been constrained except for the yaw input. The pitch and roll inputs are constrained to signed integers between -1200 and 1200; while the thrust input to an unsigned integer between 0 and 2700. The range of the yaw input is maintained. All these values have been determined by experimental tests within the capture room. Furthermore, outside these limits, there is no guarantee that the quadrotor's maneuver has enough room to avoid crashes against the walls or ceiling.

2.3 The VICON MX System

The VICON MX system is a precise optical motion capture system that consists of special cameras, a controlling hardware module (known as the MX Giganet) and a host computer where a software is installed to analyze and present the data [40]. With the combination of these components, the system can track the position and orientation of any object with high precision at a data rate of 225 Hz. In fact, the VICON MX system works by tracking special reflective markers that should be mounted on the desired object. Furthermore, the system is capable of tracking multiple objects simultaneously.

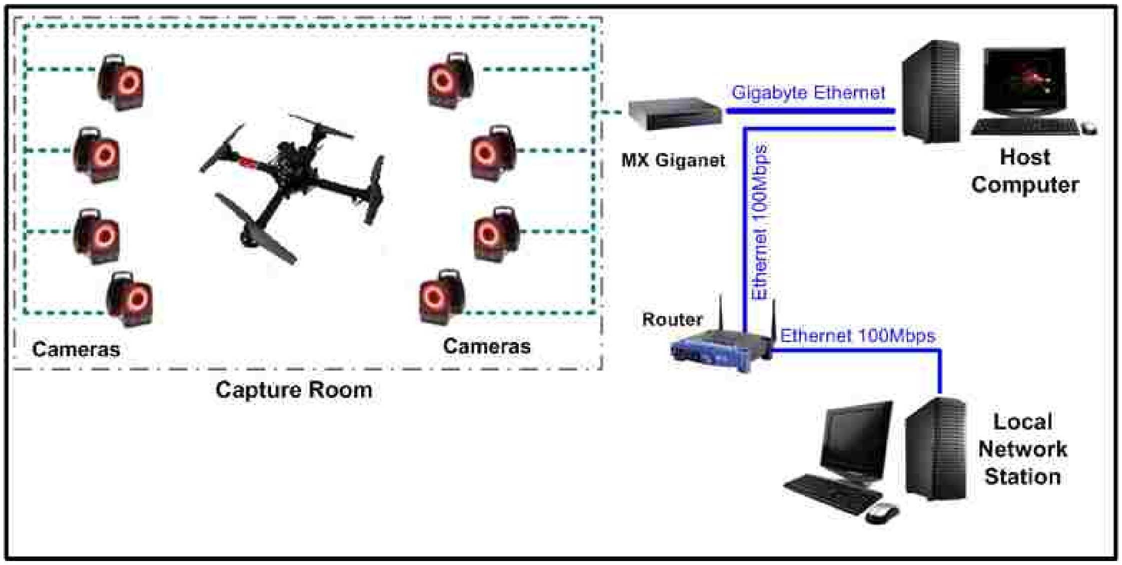
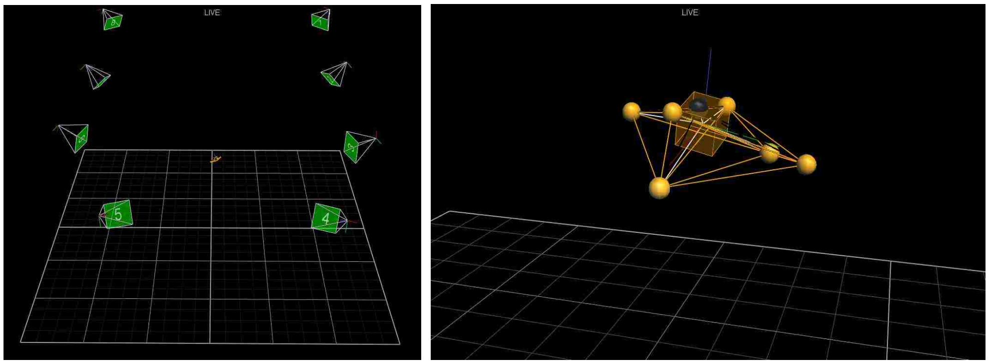


Figure 2.8: Scheme of the VICON MX system installed in the Marhes Lab.

In the capture room at the Marhes laboratory, a total of eight cameras have been installed. These cameras are connected to the MX Giganet controlling module. At the same time, this module is connected to the host computer through a Gigabyte Ethernet interface. With the software installed on the host computer, the VICON Tracker, it is possible to manage, record and display the acquired data in real-time. Moreover, using the VICON SDK [45], it is possible to access to the VICON DataStream created by the VICON Tracker and stream these data in real time to third-party computer graphics software such as MATLAB or LabVIEW via TCP/IP. A scheme of the architecture of the system installed in the laboratory is presented in Figure 2.8 and two screen shots showing the capture room and a close up of the 3D object from the VICON Tracker software are shown in Figure 2.9.



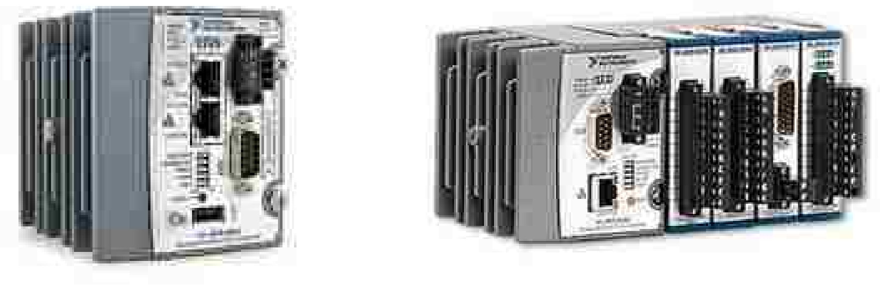
(a) The 3D Perspective of the capture room.

(b) The 3D Perspective of the quadrotor.

Figure 2.9: Screen shots from the VICON Tracker software.

2.4 The NI CompactRIO System

The NI CompactRIO system, NIcRIO, consists of an embedded controller for communications and processing, a reconfigurable chassis housing the user-programmable FPGA, hot swappable I/O modules, and NI LabVIEW graphically programming tools [41]. In addition, this system offers powerful stand-alone embedded execution for LabVIEW Real-Time (RT) applications. Indeed, the LabVIEW Real-Time Module is an add-on installable component for the LabVIEW Development System that compiles and optimizes the LabVIEW graphical code for the selected real-time target.



(a) The NI cRIO-9024 RT controller. (b) Controller, chassis and modules.

Figure 2.10: The NI CompactRIO system.

In the case of the Marhes Lab, each one of the NI cRIO systems is composed by the NI cRIO-9024 real-time controller (see Figure 2.10). In addition, the lab has available chassis, analog input modules, analog output modules, bidirectional digital I/O modules and brushed DC servo driver modules for each system. Specifically for the purpose of this thesis, just the real-time controller is required. Therefore, its most relevant characteristics of this controlled are enumerated next:

- Embedded controller that runs LabVIEW Real-Time for deterministic control, data logging, and analysis,
- 800 MHz processor, 4 GB nonvolatile storage, 512 MB DDR2 memory,
- Dual Ethernet ports with embedded Web and file servers for remote user interfacing,

- Hi-Speed USB host port for connection to USB flash and memory devices,
- RS232 serial port for connection to peripherals; dual 9 to 35 VDC supply inputs.

In this Chapter, an overview of the elements that are going to be part of the control architecture has been presented. Moreover, the most relevant details around the AscTec Hummingbird Autopilot quadrotor, have been included. The following Chapter will address the quadrotor modeling and supply a valid nonlinear system model for this aerial platform.

Chapter 3

Quadrotor Modeling

A model for the quadrotor is necessary not only to develop a model based controller but also to understand the physical principles around how this aerial vehicle can be commanded. Thus, some general concepts and assumptions need to be introduced before modeling.

3.1 Basic Concepts and Assumptions

A quadrotor is considered as a single rigid body with 6 degrees of freedom (DOF): 3 DOF for translation and 3 DOF for orientation. In fact, this assumption is common in the literature as it is stated in different theses [10, 27, 28, 29, 30] and papers [5, 6, 7, 9, 26, 46]. Furthermore, the basic model of a quadrotor is composed just by a thin cross structure with four propellers on its ends (see Figure 3.1). Moreover, all the propellers have fixed-pitch blades and the air-flow points downwards obtaining an upward lift. Therefore, the only parameter that can vary is the speed of each propeller.

3.1.1 Coordinate Systems

To describe the position and orientation of an object in a 3D space, it is desired to define coordinate systems and to develop conventions for representation.

The two right-handed coordinate systems used in this thesis are illustrated in Figure 3.1. The world or inertial coordinate system, $\{W\}$, is defined by the axes X_W, Y_W, and Z_W and it is related to the capture room in the Marhes Lab. This system is oriented so the plane XY_W is in the room floor and the Z_W axis is pointing upwards. On the other hand, the body-fixed coordinate system, $\{B\}$, coincides with the center of mass of the quadrotor and is defined by the axes X_B, Y_B, and Z_B. This helps to follow the attitude of the quadrotor at any instant. The x-body axis, X_B, is always aligned with the front of the aerial robot. On the Asctec Hummingbird quadrotor, the front is marked with an orange tape to identify the attitude of the vehicle when it is flown by a human operator.

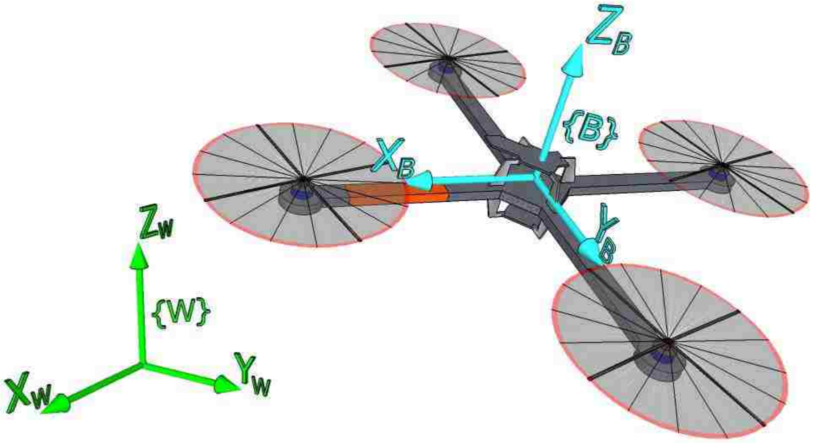


Figure 3.1: Fixed-body and world coordinate systems.

The vectors in the coordinate system $\{W\}$ will be denoted with a ^{W} in front of the vector variable and those in $\{B\}$ with a ^{B}. Hence, the position vector of the origin of the body-fixed coordinate system i.e., the position of the quadrotor's center of mass respect to $\{W\}$ is given by ${}^{W}\mathbf{r}_{cg} = [r_x \quad r_y \quad r_z]^T$ or just ${}^{W}\mathbf{r} = [x \quad y \quad z]^T$.

3.1.2 Orientation and Rotations

The quadrotor is able to rotate around all three fixed-body axes and the sign of the rotations are according to the right-hand convention. In fact, the orientation of one Cartesian coordinate system with respect to another can always be described by three successive rotations. The Tait-Bryan or Cardan angle rotation description [47] is mainly used in the different quadrotor papers, theses and reports, but there is no agreement about the sequence of the three successive rotations. In this thesis, the rotation sequence $Z-Y-X$ or $3-2-1$ is employed because this is the most typical sequence used in the different quadrotor references such as in [9, 10, 26, 27, 28]. Thus, the rotation sequence is given as follows: first a rotation about the X_W axis by the roll angle, ϕ, a then rotation about the intermediate Y axis by the pitch angle, θ, and, finally, a rotation about the Z_B axis by the yaw angle, ψ [47]. Consequently, the rotational matrix from the fixed-body coordinate system to the world coordinate system, ${}^{W}\mathbf{R}_B$, is defined as

$$\begin{aligned}
{}^{W}\mathbf{R}_B &= \mathbf{R}_Z(\psi)\,\mathbf{R}_Y(\theta)\,\mathbf{R}_X(\phi), \\
{}^{W}\mathbf{R}_B &= \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix}, \\
{}^{W}\mathbf{R}_B &= \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix},
\end{aligned} \tag{3.1}$$

where c and s are abbreviations for cosine and sine, respectively.

The angular velocity of the fixed-body coordinate system is given by ${}^{B}\boldsymbol{\omega} =$

$[p \quad q \quad r]^T$ and it is related to the time derivatives of the roll, pitch and yaw angles according to [29]

$$
\begin{aligned}
{}^{B}\boldsymbol{\omega} &= \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \mathbf{R}_X(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \mathbf{R}_X(\phi)\,\mathbf{R}_Y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}, \\
{}^{B}\boldsymbol{\omega} &= \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & s\phi c\theta \\ 0 & -s\phi & c\phi c\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}.
\end{aligned}
\tag{3.2}
$$

Inverting the relation in (3.2) provides the relation between the time derivatives of the roll, pitch, and yaw angles with the angular velocity of the fixed-body coordinate system.

$$
\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{3.3}
$$

where t is an abbreviation of tangent. This representation has a singularity at $\theta = \pm\frac{\pi}{2}$, but this limitation does not affect the quadrotor performance in normal flight [9].

3.1.3 Quadrotor Basic Maneuvers

Figure 3.2 shows the quadrotor structure at hovering position where all the propellers have the same speed to counterbalance the acceleration due to gravity and the aerial vehicle is aligned with the XY_W plane. In this figure, two arrows are drawn, one in blue and one in red, and they are related with the angular speed of the propellers. The red curved arrow represents the direction of rotation, while the blue straight arrow represents the thrust. This arrow always points upwards because it would be confusing to have two vectors pointing upwards and the other pointing downwards. In addition, the front and rear propellers rotate counter-clockwise while the left and the right ones turn clockwise. This assumption is made since the rotors of the AscTec Hummingbird quadrotor present this configuration. Indeed, this format of opposite directions in pairs removes the need for a tail rotor that is needed in a standard helicopter.

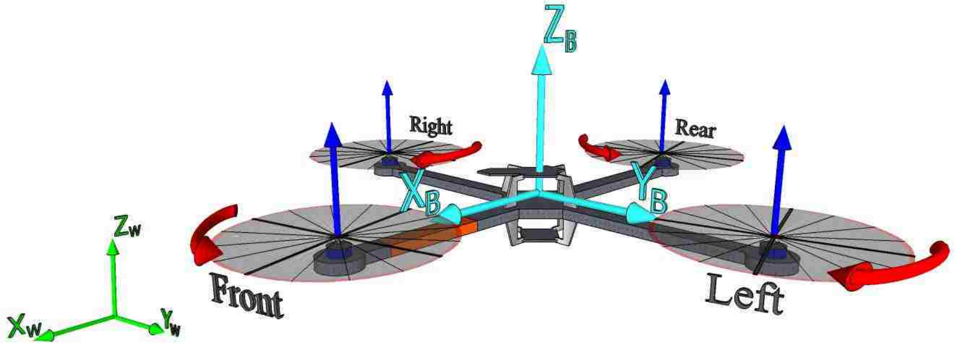


Figure 3.2: Quadrotor structure at hovering position.

Since the UAV is equipped with four propellers, it is not possible to reach a desired set-point for all the 6 degrees of freedom. Consequently, the quadrotor is an under-actuated system and there are just four basic maneuvers. These maneuvers allow the aerial vehicle to reach a certain height and attitude. These four basic maneuvers are [27]:

Thrust or Collective This command is provided by increasing (or decreasing) all the propellers speeds by the same amount. It leads to a vertical force, U, with respect to the body-fixed frame which raises or lowers the quadrotor.

Roll This command is provided by increasing (or decreasing) the left propeller speed and by decreasing (or increasing) the right one. It leads to a torque with respect to the X_B axis which makes the quadrotor turn.

Pitch This command is very similar to the roll and is provided by increasing (or decreasing) the rear propeller speed and by decreasing (or increasing) the front one. It leads to a torque with respect to the Y_B axis which makes the quadrotor turn.

Yaw This command is provided by increasing (or decreasing) the front-rear propellers speed and by decreasing (or increasing) the left-right couple. It leads to a torque with respect to the Z_B axis which makes the quadrotor turn. The yaw moment is generated thanks to the fact that the left-right propellers rotate clockwise while the front-rear ones rotate counter-clockwise. Indeed, the rotorcraft vehicle turns on itself around Z_B when the overall torque is unbalanced.

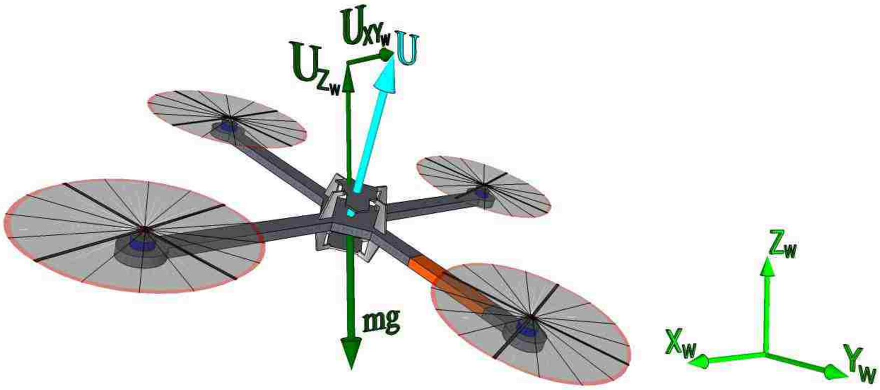


Figure 3.3: Quadrotor movement in the XY_W plane.

Based on this basic maneuvers, the quadrotor can reach a desired position in a 3D space with a desired heading (yaw angle) i.e., a desired set-point of 4 DOF $[x \quad y \quad z \quad \psi]^T$. The movement in the XY_W plane is done by tilting the quadrotor in the desired direction. When the quadrotor is tilted, the direction of the total thrust, U, is no longer aligned with the Z_W axis as it is during hovering. Furthermore, U can be decomposed in a Z_W axis component and in a XY_W plane component (see Figure 3.3). This last component results in an acceleration in the XY_W plane.

3.2 Model Structure

The most applied technique to model a quadrotor is the first principle modeling technique. This method involves deriving the mathemathical equations of motion using the fundamental laws of mechanics and aerodynamics. The resulting model is nonlinear and coupled, and describes the aerial vehicle dynamics in a large portion of its flight envelope. However, it contains a number of unknown physical parameters such as geometrical data and aerodynamics coefficients [4]. Consequently, a combination between this technique and system identification is an option presented in the literature about quadrotors [10, 27, 28, 29, 30]. Hence, this approach is used in this thesis.

The structure of the dynamical model is presented in Figure 3.4. The inputs are the roll, S_ϕ, pitch, S_θ, yaw, S_ψ, and thrust, S_{Th}, values that are sent via the wireless serial link as it is presented in Section 2.2.2. Meanwhile, the outputs are the position vector, ${}^W\mathbf{r}$, and the orientation vector, ${}^W\boldsymbol{\Theta}$, both with respect to the world coordinate system.

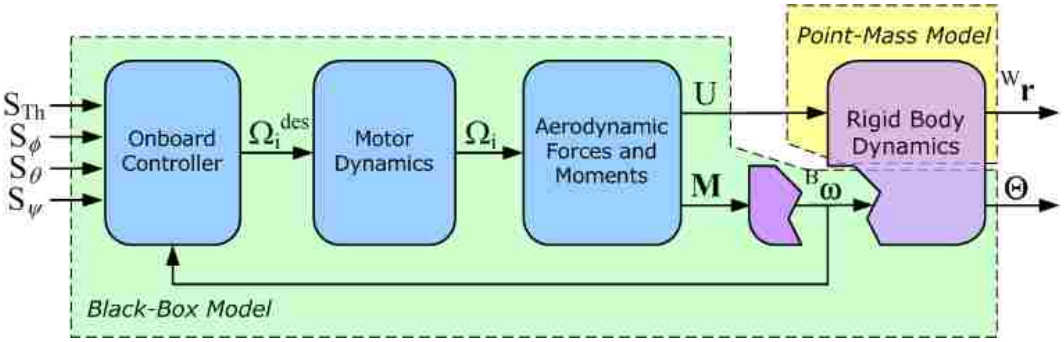


Figure 3.4: Structure of the quadrotor model.

As it is also shown in Figure 3.4, the model is split into two main components: the Point-Mass model and the Black-Box model. The first one involves the translational part of the rigid body dynamics; while the second represents the onboard controller, the motor dynamics, the aerodynamics forces and momentums, and the rotational part of the rigid body dynamics.

3.2.1 Point-Mass Model

Since the quadrotor has small size and it is symmetric, it is possible to model it as a point-mass [30]. Thus, applying Newton's Second Law at the world coordinate system, $\{W\}$, one can obtain

$$m^W\ddot{\mathbf{r}} = \sum {}^W\mathbf{F}_i,$$

where m is the quadrotor mass, ${}^W\ddot{\mathbf{r}}$ denotes the acceleration of the center of gravity respect to the world frame and ${}^W\mathbf{F}_i$ represents any outer force acting on the point-mass.

Figure 3.5 shows the forces, the momentums acting on the quadrotor and the position vector of the center of gravity of the aerial vehicle respect to the world frame, ${}^W\mathbf{r}$. The forces are the gravitational force and the upward forces generated by each rotor ($\mathbf{F}_1$ to $\mathbf{F}_4$). Thus,

$$m^W\ddot{\mathbf{r}} = -m\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \sum_{i=1}^{4} {}^W\mathbf{F}_i,$$

where ${}^W\mathbf{F}_i =^W \mathbf{R}_B\, {}^B\mathbf{F}_i$ then

$$m^W\ddot{\mathbf{r}} = -m\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + {}^W\mathbf{R}_B\sum_{i=1}^{4} {}^B\mathbf{F}_i.$$

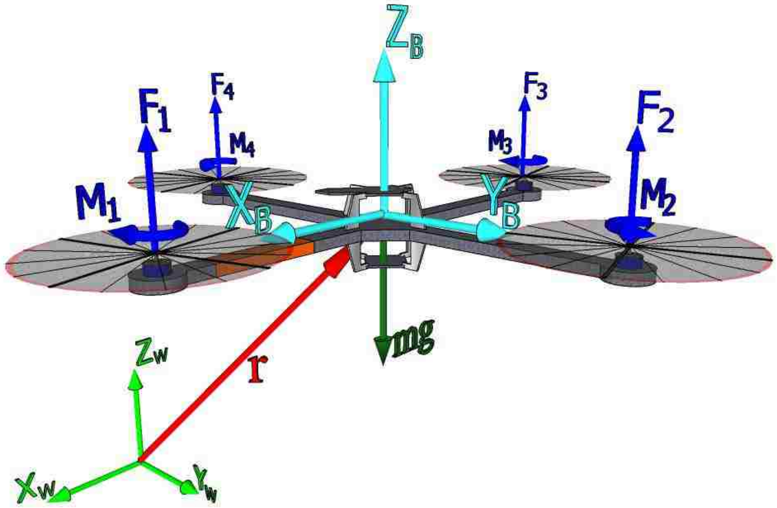


Figure 3.5: Forces and momentums acting on the quadrotor.

Considering the body-fixed coordinate system, it is clear that the forces $\mathbf{F}_1$ to $\mathbf{F}_4$ are always directed along the Z_B axis, so it is possible to define

$$\sum_{i=1}^{4} {}^{B}\mathbf{F}_i = \begin{bmatrix} 0 \\ 0 \\ {}^{B}\mathbf{F}_1 +^{B} \mathbf{F}_2 +^{B} \mathbf{F}_3 +^{B} \mathbf{F}_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mathrm{U} \end{bmatrix},$$

where U, the total thrust, is depicted in Figure 3.3. Consequently,

$$m^{W}\ddot{\mathbf{r}} = -m \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + {}^{W}\mathbf{R}_B \begin{bmatrix} 0 \\ 0 \\ \mathrm{U} \end{bmatrix},$$

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = -\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \frac{1}{m}{}^{W}\mathbf{R}_B \begin{bmatrix} 0 \\ 0 \\ \mathrm{U} \end{bmatrix}. \tag{3.4}$$

3.2.2 Black-Box Model

In Equation (3.4), there is the requirement of calculating the Euler angles, (ϕ, θ, ψ), and the total thrust force, U. These values depend on the onboard controller designed by Ascending Technologies [42, 43], so an exact description of the internal controller is necessary. Nevertheless, there is not a complete explanation of the controller configuration and how the control-outputs are combined to compute a desired speed of each motor. For that reason, a system identification approach is chosen to derive decoupled closed-loop transfer functions for the Euler angles and a polynomial for the total thrust force.

Total Thrust, U

The force generated for each rotor depends on its angular velocity. Since the angular velocity of the rotors is controlled by the onboard controller, the resultant force, U, is modeled as a function of the thrust input $\mathrm{S_{Th}}$ [28]. The procedure to estimate the linear function related to the total thrust is explained in Section 4.1 and just its equation is reproduced next,

$$\begin{aligned}\mathrm{U} = &- 0.117 \times 10^{-12} \left(\mathrm{S_{Th}}\right)^4 + 0.32809 \times 10^{-9} \left(\mathrm{S_{Th}}\right)^3 \\ &+ 1.306 \times 10^{-6} \left(\mathrm{S_{Th}}\right)^2 + 0.4929 \times 10^{-3} \left(\mathrm{S_{Th}}\right) + 0.61485.\end{aligned} \tag{3.5}$$

Estimated Transfer Functions

The main idea of black-box modeling is to approximate the dynamics of the system with an educated guess [30]. The system parameters are identified and validated with an adequate method using flight data sets. Furthermore, the three decoupled estimated subsystems are:

$$\begin{aligned} H_{\phi_B}(s) &= \frac{28.8832}{s^2 + 347.8345\,s + 991.0803}, \\ H_{\theta_B}(s) &= \frac{34.5702}{s^2 + 470.7944\,s + 834.5643}, \\ H_r(s) &= \frac{-0.25095}{s + 4.180619}. \end{aligned} \tag{3.6}$$

A detailed explanation about the procedure to identify and to validate these transfer functions is described in Section 4.2.

3.2.3 Complete Dynamical Model

A summary of the quadrotor dynamical equations is presented next. Moreover, the final model structure is illustrated in Figure 3.6. From Section 3.1.2 and applying (3.1) on (3.4), one can get

$$\begin{aligned} \ddot{x} &= \frac{1}{m}\left(c\phi s\theta c\psi + s\phi s\psi\right)\mathrm{U}, \\ \ddot{y} &= \frac{1}{m}\left(c\phi s\theta s\psi - s\phi c\psi\right)\mathrm{U}, \\ \ddot{z} &= -g + \frac{1}{m}\,c\phi\,c\theta\,\mathrm{U} \end{aligned} \tag{3.7}$$

The system of equations (3.7) requires the values of the Euler angles ϕ, θ, and ψ. Thus, these can be calculated using (3.2)

$$\begin{aligned} \dot{\phi} &= p + \left(s\phi t\theta\right)q + \left(c\phi t\theta\right)r, \\ \dot{\theta} &= c\phi\,q - s\phi\,r, \\ \dot{\psi} &= \left(\frac{s\phi}{c\theta}\right)q + \left(\frac{c\phi}{c\theta}\right)r, \end{aligned} \tag{3.8}$$

where, the angular velocities respect to frame $\{B\}$ (p, q and r) could be found by transforming the transfer functions in (3.6) into differential equations.

$$\begin{aligned}
\dot{p} &= -347.8345\, p - 991.0803\, \phi_B + 28.8832\, \mathrm{S}_\phi \\
\dot{q} &= -470.7944\, q - 834.5643\, \theta_B + 34.572\, \mathrm{S}_\theta \\
\dot{r} &= -4.180619\, r - 0.25095\, \mathrm{S}_\psi
\end{aligned} \tag{3.9}$$

Subsequently, Equations (3.7), (3.8) and (3.9) define the dynamical model of the quadrotor. In fact, the model equations are summarized in (3.10).

$$\begin{cases}
\ddot{x} = \frac{1}{m}\left(c\phi s\theta c\psi + s\phi s\psi\right)\mathrm{U} \\
\ddot{y} = \frac{1}{m}\left(c\phi s\theta s\psi - s\phi c\psi\right)\mathrm{U} \\
\ddot{z} = -g + \frac{1}{m}\left(c\phi c\theta\right)\mathrm{U} \\
\dot{\phi} = p + \left(s\phi t\theta\right) q + \left(c\phi t\theta\right) r \\
\dot{\theta} = c\phi\, q - s\phi\, r \\
\dot{\psi} = \left(\frac{s\phi}{c\theta}\right) q + \left(\frac{c\phi}{c\theta}\right) r \\
\dot{p} = -347.8345\, p - 991.0803\, \phi_B + 28.8832\, \mathrm{S}_\phi \\
\dot{q} = -470.7944\, q - 834.5643\, \theta_B + 34.572\, \mathrm{S}_\theta \\
\dot{r} = -4.180619\, r - 0.25095\, \mathrm{S}_\psi
\end{cases} \tag{3.10}$$

where U is given by Equation (3.5).

3.3 Modeling for Control Purposes

The model given in (3.10) can be rewritten in state-space form with the state vector, **X**, chosen as in (3.11). The extra states ϕ_B and θ_B are required to express (3.9) on state-space form.

$$\begin{aligned}
\mathbf{X} &= \left[x \;\; y \;\; z \;\; \dot{x} \;\; \dot{y} \;\; \dot{z} \;\; \phi \;\; \theta \;\; \psi \;\; \phi_B \;\; \theta_B \;\; p \;\; q \;\; r\right]^T \\
&= \left[x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5 \;\; x_6 \;\; x_7 \;\; x_8 \;\; x_9 \;\; x_{10} \;\; x_{11} \;\; x_{12} \;\; x_{13} \;\; x_{14}\right]^T,
\end{aligned} \tag{3.11}$$

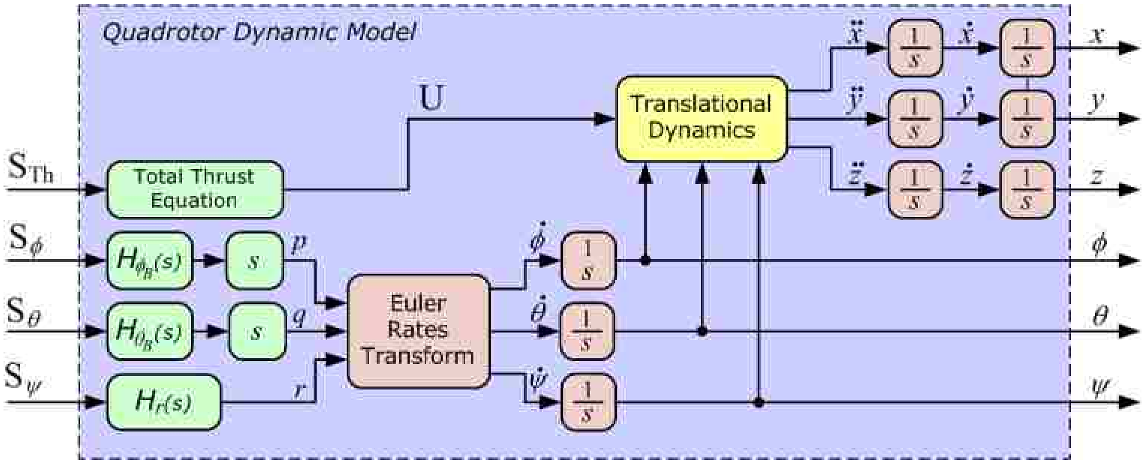


Figure 3.6: Complete quadrotor dynamic model.

Under these definitions, the state-space description of the quadrotor is given by

$$\begin{cases} \dot{x}_1 = x_4 \\ \dot{x}_2 = x_5 \\ \dot{x}_3 = x_6 \\ \dot{x}_4 = \frac{1}{m}\left[c\left(x_7\right)s\left(x_8\right)c\left(x_9\right) + s\left(x_7\right)s\left(x_9\right)\right]\mathrm{U} \\ \dot{x}_5 = \frac{1}{m}\left[c\left(x_7\right)s\left(x_8\right)s\left(x_9\right) - s\left(x_7\right)c\left(x_9\right)\right]\mathrm{U} \\ \dot{x}_6 = -g + \frac{1}{m}\left[c\left(x_7\right)c\left(x_8\right)\right]\mathrm{U} \\ \dot{x}_7 = x_{12} + \left[s\left(x_7\right)t\left(x_8\right)\right]x_{13} + \left[c\left(x_7\right)t\left(x_8\right)\right]x_{14} \\ \dot{x}_8 = c\left(x_7\right)\, x_{13} - s\left(x_7\right)\, x_{14} \\ \dot{x}_9 = \left[\frac{s(x_7)}{c(x_8)}\right]x_{13} + \left[\frac{c(x_7)}{c(x_8)}\right]x_{14} \\ \dot{x}_{10} = x_{12} \\ \dot{x}_{11} = x_{13} \\ \dot{x}_{12} = -347.8345\, x_{12} - 991.0803\, x_{10} + 28.8832\, \mathrm{S}_\phi \\ \dot{x}_{13} = -470.7944\, x_{13} - 834.5643\, x_{11} + 34.572\, \mathrm{S}_\theta \\ \dot{x}_{14} = -4.180619\, x_{14} - 0.25095\, \mathrm{S}_\psi \end{cases} \tag{3.12}$$

In addition, the system input, $\mathbf{U_i}$, is given by

$$\mathbf{U_i} = [\mathrm{U} \quad \mathrm{S}_\phi \quad \mathrm{S}_\theta \quad \mathrm{S}_\psi]^T, \tag{3.13}$$

Meanwhile, the system output, $\mathbf{Y}$, is formed by the states that can be measured using the VICON system (Section 2.3). These states are the position and orientation of the flying robot in the capture room respect to the $\{W\}$ frame. Thus, the output of the system is

$$\mathbf{Y} = [x_1 \quad x_2 \quad x_3 \quad x_7 \quad x_8 \quad x_9]^T. \tag{3.14}$$

3.3.1 Linear Model

In order to use the state-space representation (3.12) in designing a based linear controllers, the system needs to be linearized. Consequently, a first order Taylor approximation is used and the hovering condition is the operating point selected for the linearization. The angles, the angular velocities and the translational velocities are supposed equal to zero. This is

$$\begin{array}{l|l|l|l}
x_4 = \dot{x} = 0, & x_7 = \phi = 0, & x_{10} = \phi_B = 0, & x_{12} = p = 0, \\
x_5 = \dot{y} = 0, & x_8 = \theta = 0, & x_{11} = \theta_B = 0, & x_{13} = q = 0, \\
x_6 = \dot{z} = 0, & x_9 = \psi = 0, & & x_{14} = r = 0,
\end{array}$$

Linearizing (3.12) around the operating point and considering the system input (3.13) and the system output (3.14), the linear state-space model (3.15) can be found.

$$\begin{aligned}
\dot{\mathbf{X}} &= \mathbf{A}\,\mathbf{X} + \mathbf{B}\,\mathbf{U_i}, \\
\mathbf{Y} &= \mathbf{C}\,\mathbf{X}.
\end{aligned} \tag{3.15}$$

where the matrices **A**, **B** and **C** are given by Equations (3.16), (3.17) and (3.18), respectively.

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_2 & 0 & a_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_4 & 0 & a_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_6 \end{bmatrix} \tag{3.16}$$

with $a_0 = \frac{\mathrm{U_h}}{m}$, $a_1 = -a_0$, $a_2 = -991.0803$, $a_3 = -347.8345$, $a_4 = -834.5643$, $a_5 = -470.7944$ and $a_6 = -4.180619$. $\mathrm{U_h}$ is the total thrust for hovering and m is the total mass of the quadrotor. The value of $\mathrm{U_h}$ is supposed to be equal to the total weight of the quadrotor. The mass of the AscTec Hummingbird without battery is 353 g (see Table 2.1) and the mass of its battery is 152 g. As a result, the total mass of the quadrotor, m, is 505 g and the value of the total thrust during hovering, $\mathrm{U_h}$, is 4.95 N.

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b_0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & b_1 & 0 & 0 \\ 0 & 0 & b_2 & 0 \\ 0 & 0 & 0 & b_3 \end{bmatrix} \tag{3.17}$$

with $b_0 = \frac{1}{m}$, $b_1 = 28.8832$, $b_2 = 34.572$, and $b_3 = -0.25095$.

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.18}$$

A quadrotor nonlinear dynamical model has been formulated in this Chapter combining classical modeling techniques with system identification. Indeed, the procedure and results around the system identification of the black-box component of the complete dynamical model is the main focus of the next Chapter.

Chapter 4

System Identification and Parameter Estimation

System identification and parameter estimation involve building mathematical models of a dynamic system based on a set of measured stimulus and response data samples [48]. Consequently, a mathematical representation that describes the black-box part of the model structure (see Figure 3.4) is explained and obtained in this chapter.

4.1 Thrust Force Relation

The thrust force acting on the quadrotor in near hover flight, U, can be found empirically from measuring how it changes as the *Thrust* input, $\mathrm{S_{Th}}$, changes. This polynomial relation of U with respect to $\mathrm{S_{Th}}$ can be estimated by measured data. This approach avoids the application of an aerodynamic model of the rotor. In fact, this method is described and employed in [10, 28, 29]. Some details about its procedure and results are described next.

4.1.1 Experimental Test

The quadrotor is attached to a load heavy enough to be lifted. Thus, measurements of changes in the total weight are recorded while the Thrust input is increased. Since the maximum thrust depends on factors such as the type and condition of the propeller and battery level, the experiments should be carried out with a full charged battery and with a new set of propellers.

The steps of the experiment are as follows:

1. The quadrotor attached to the load mass is placed on a scale that is over an elevated platform (0.75 m). This platform helps to minimize any ground effect during the test.

2. The initial weight is registered before starting the motors.

3. After starting the quadrotor, thrust input values are sent using the wireless serial link described in Section 2.2.2. Values from 0 to 4095 in steps of 250 are transmitted. Meanwhile, the changes on the weight are recorded at each sent value.

4. Once the experiment is concluded, the quadrotor is returned to idle mode.

The test was carried out ten times and the measured raw data are provided in Appendix A. These data should be processed to determine the Thrust force, U. Indeed, the force is evaluated by first calculating the weight loss in kg and then inferring the generated force. Thus, for the weight loss, one has $\Delta m = m_{ini} - m_{meas}$, where m_{ini} is the initial value of weight and m_{meas} is the measured value for each transmitted thrust input. Meanwhile the generated force is determined by applying Newton's second law U $= \Delta m\, g$, where U is in N and g is the acceleration due to gravity, 9.81 m/s^2. Moreover, extra columns in the tables in Appendix A present

Thrust Input $\mathrm{S_{Th}}$	Thrust Force, U [N]
0	0.529
250	0.817
500	1.273
750	1.881
1000	2.756
1250	3.645
1500	4.677
1750	5.911
2000	7.434
2250	8.887
2500	10.531
2750	12.212
3000	13.724
3250	14.281
3500	14.433
3750	14.372
4000	14.374
4095	14.369

Table 4.1: Mean of the Thrust Force, U for each value of Thrust Input, $\mathrm{S_{Th}}$.

the results after employing these two equations. Furthermore, from the thrust force valued calculated for each experiment, the mean value of U for each thrust input is obtained. These results are shown in Table 4.1.

4.1.2 Polynomial Relation

Based on the data from Table 4.1, it is possible to estimate a polynomial relationship between U and $\mathrm{S_{Th}}$. In order to find the polynomial coefficients, the General Polynomial VI (Virtual Instrument) of LabVIEW [49] is used with the Least Square method as fitting method. This VI returns the polynomial coefficients and the weighted mean absolute error of the fitted model, generally known as *residue.* A third order polynomial is selected as first choice and its resulting equation is

$$\begin{aligned}\mathrm{U} = &-0.6398 \times 10^{-9}\,(\mathrm{S_{Th}})^3 + 3.8218 \times 10^{-6}\,(\mathrm{S_{Th}})^2 \\ &+ 1.6588 \times 10^{-3}\,(\mathrm{S_{Th}}) + 0.9291,\end{aligned}$$

and Figure 4.1 presents the resulting plot. Also, the residue for this case is 0.0924.

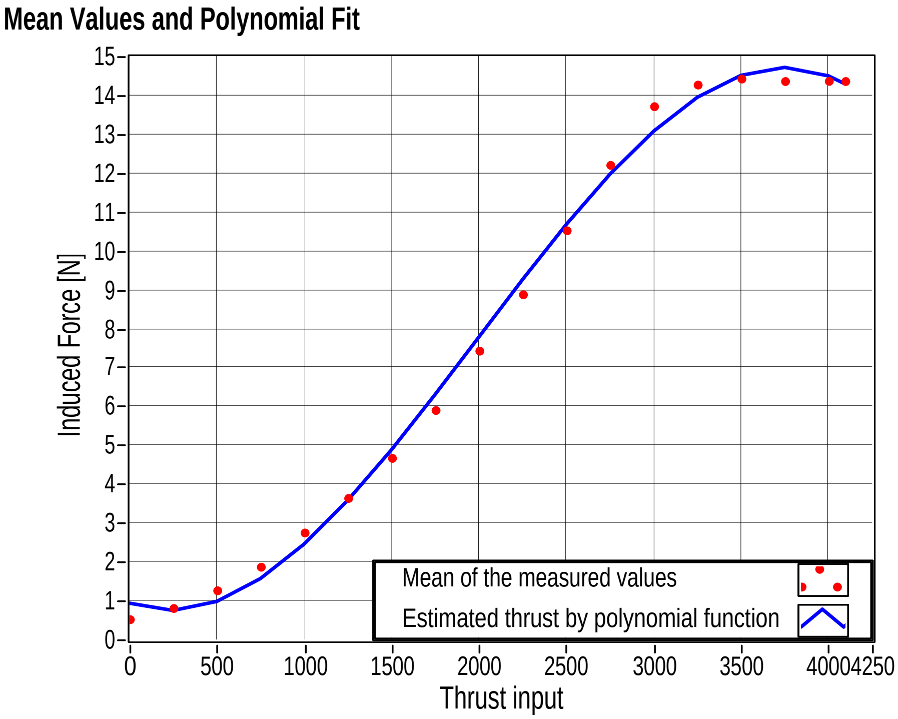


Figure 4.1: U and $\mathrm{S_{Th}}$ relation based in the third order polynomial approximation.

It is clear from Figure 4.1 that the third order polynomial estimation does not fit correctly with the values at the beginning and at the end of the curve. Since it is important to have a better approximation specially for the starting values (normally, the quadrotor will not be commanded using the upper section of the thrust input range), a fourth order polynomial is employed whose resulting equation is

$$\begin{aligned} \mathrm{U} = & - 0.117 \times 10^{-12} \left(\mathrm{S_{Th}}\right)^4 + 0.32809 \times 10^{-9} \left(\mathrm{S_{Th}}\right)^3 \\ & + 1.306 \times 10^{-6} \left(\mathrm{S_{Th}}\right)^2 + 0.4929 \times 10^{-3} \left(\mathrm{S_{Th}}\right) + 0.61485 \end{aligned} \tag{4.1}$$

Figure 4.2 shows the corresponding plot for the fourth order polynomial estimation and its residue is 0.056966. Therefore, this last polynomial equation is the thrust force relation using for modeling purposes, see Section 3.2.2.

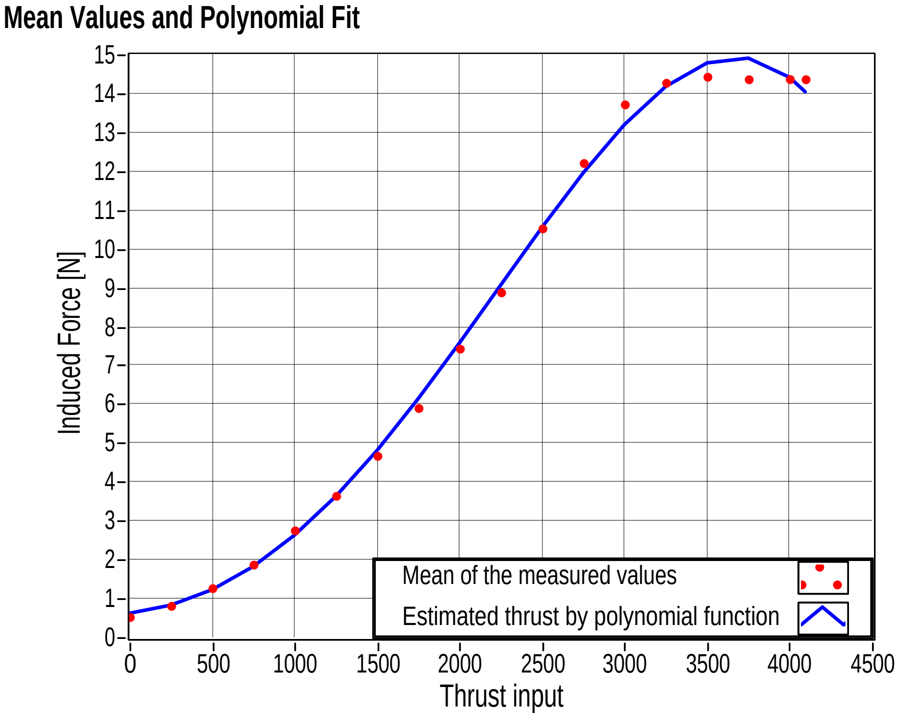


Figure 4.2: U and $\mathrm{S_{Th}}$ relation based in the fourth order polynomial approximation.

4.2 Black-Box Model Identification

In this section, the models for roll, pitch and yaw input-output relation are identified and estimated. As it was stated in Section 3.2, a black-box system identification approach is employed in order to accomplish this goal. A black-box model from experimental data proposes a mathematical representation that allows sufficient description of any observed input and output measurements [50]. Indeed, system identification is a process that includes acquiring, formatting, processing and of course identifying mathemathical models based on raw data from real-world systems. In addition, the resulting model should be validated by checking that it fits the observed system behavior. If the results are unsatisfactory, the parameters should be changed and iterated through the process [48].

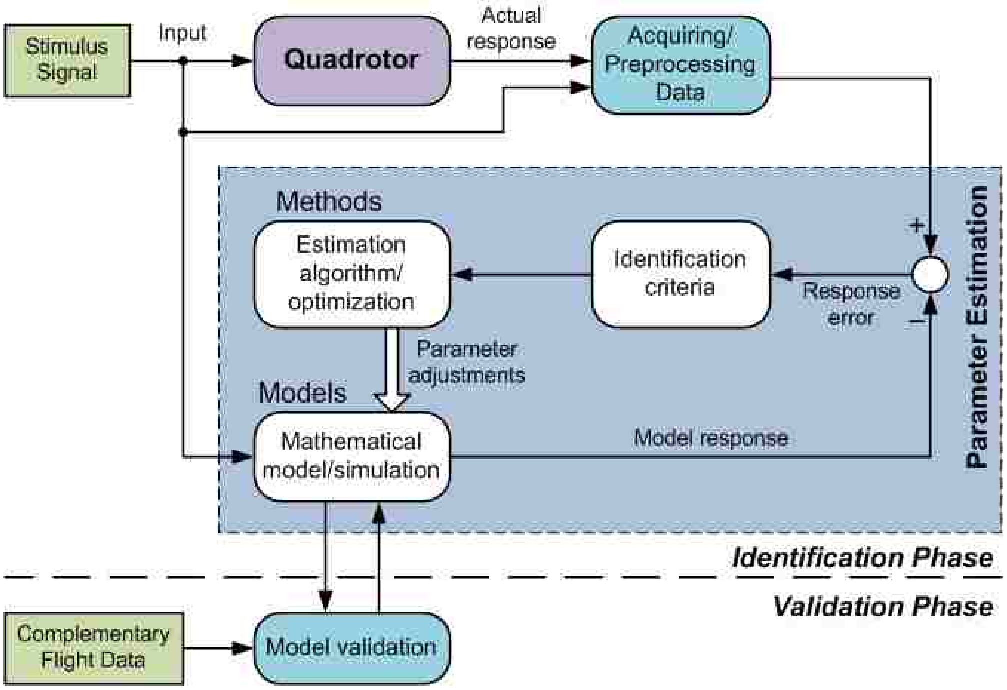


Figure 4.3: Scheme of the system identification procedure.

A diagram that summarizes the system identification procedure is presented in Figure 4.3. This diagram has been adapted from [51]. Furthermore, the procedure steps: *acquiring and preprocessing data, estimating the model and model validation*, have been implemented employing the LabVIEW System identification Toolkit [48].

In this thesis, the main idea is to estimate models that describe the dynamics from the command input values, sent through Zig-Bee, to the changes in the roll, pitch and yaw angles. From initial experimental tests, it is clear that changes in the roll and pitch input values are related to changes in the roll and pitch angle values, respectively. However, the yaw input value is related to change in the yaw *rate* value. Besides, it is assumed that the models are decoupled due to the internal controllers as in [28, 30]. Consequently, the models that should be estimated are:

- From roll input value, S_{ϕ}, to roll angle respect to the body fixed coordinated system, ϕ_B,
- From pitch input value, S_{θ}, to pitch angle respect to the body fixed coordinated system, θ_B, and
- From yaw input value, S_{ψ}, to yaw rate respect to the body fixed coordinated system, $\dot{\psi}_B = r$.

4.2.1 Acquiring Data

The first step in identifying an unknown system is data acquisition. In fact, identifying a system involves a number of choices with regard to the system output signals to measure and the input signals to manipulate. Moreover, the choices are about how to manipulate system inputs, the type of stimulus signal and signal ranges [48].

As it was explained, the inputs of the system are the command values sent through the serial wireless link explained in Section 2.2.2. On the other hand, the outputs of the system are the roll angle, the pitch angle and the yaw rate, all these respect to the body fixed coordinated system. Thus, it is required to get these values from the onboard gyroscopes. As it is explained in [42], the data structure IMUCalcData that includes calibrated sensor outputs and data fusion results can be polled using the serial wireless interface. Indeed, this structure contains the roll angle, the pitch angle and the yaw rate derived by integration of the gyro readings. Consequently, this data structure is requested after each command input and stored together with the corresponding input command value.

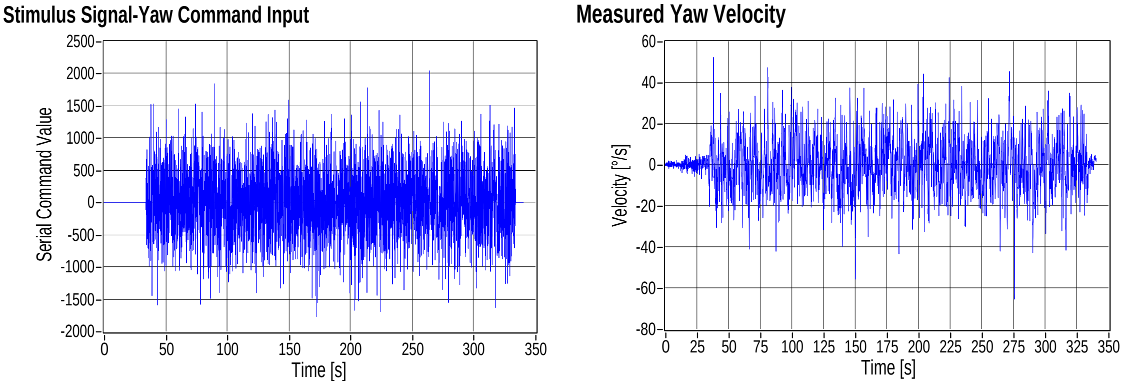
The stimulus signal needs to cover the operation range of the system inputs, see Section 2.2.2, and deliver as much input power to the system as possible [48]. Therefore, a filtered Gaussian white noise is used as stimulus signal for each one of the commands i.e., roll, pitch and yaw. An example of the stimulus signal and of the acquired response for yaw are shown in Figure 4.4. Furthermore, plots of each command input signal and its corresponding response are shown in Appendix B.

In order to prevent some possible crashes since the stimulus signal is random, the quadrotor is attached to a metal support during the experiments as it is shown in Figure 4.5. Attached to this support, the quadrotor can rotate freely but it cannot gain height. However, the system identification experiments are carried out with the quadrotor attached to the metal support, these leads to suitable results, as can be seen in Chapter 7.

4.2.2 Estimating the Model

Similarly to what is assumed in [28, 30], the structure of the models is chosen to be a linear transfer function and the dynamics can be described as a low-pass filter since the quadrotor reacts with delay to fast changes of the input.

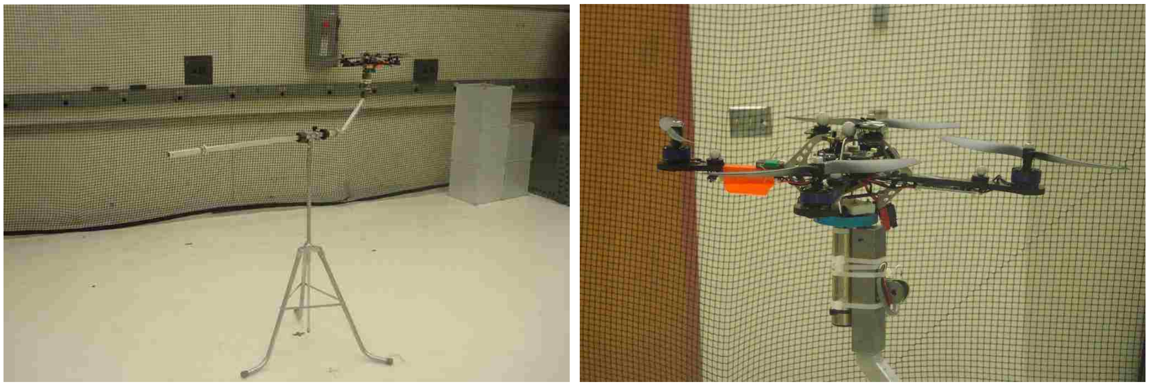
For the yaw dynamics, a first order transfer function is considered since it des-



(a) Stimulus Signal

(b) Acquired Data

Figure 4.4: Stimulus signal and the acquired data for the yaw rate $\dot{\psi}_B$.



(a) Full View

(b) Close up View

Figure 4.5: Quadrotor attached to the metal support.

cribes adequately the system. However, second order transfer functions are employed for the roll and pitch dynamics. In fact, a first order approach for the roll and pitch transfer functions does not accurately characterize accurate the system behavior.

The SI Estimate Transfer Function Model VI

The System Identification Toolkit of LabVIEW is used. A component of this toolkit, the *SI Estimate Transfer Function Model VI* estimates the parameters of a continuous time or discrete time transfer function model for an unknown system. For continuous models, this VI internally performs three consecutive steps to estimate a model:

1. Calculates a discrete model with the prediction error method.
2. Applies the Zero-Order-Hold method to convert the discrete model to a continuous model.
3. Uses the Gauss-Newton method to optimize the continuous model found in step 2.

The *prediction error method* minimizes a cost function that consists of a weighted sum of squares of the errors using an iterative nonlinear least-squares algorithm. For the second step, the converted model might have a numerator of higher order than the one selected, so the VI truncates unnecessarily high order coefficients. Moreover, these coefficients are the initial estimates for minimization in step 3. While in the third step, the initial estimates are tuned by the Gauss-Newton optimization method to minimize the following cost function [52]:

$$V_N = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}e^2(k), \tag{4.2}$$

then the estimate parameters, $\hat{\mathbf{a}}$, are given by

$$\hat{\mathbf{a}} = \arg\min_{(a)} V_N, \tag{4.3}$$

where $e(k)$ denotes the error between the measured output and the output calculated with the estimated parameters, and N is the number of samples in the measurement used for estimation.

The three estimated transfer functions are shown in Equations (4.4) - (4.6). The relation that each transfer function represents is also indicate in this equations.

$$H_{\phi_B}\left(s\right) = \frac{\phi_B}{\mathrm{S}_\phi} = \frac{28.8832}{s^2 \,+\, 347.8345\, s \,+\, 991.0803}, \tag{4.4}$$

$$H_{\theta_B}\left(s\right) = \frac{\theta_B}{\mathrm{S}_\theta} = \frac{34.5702}{s^2 \,+\, 470.7944\, s \,+\, 834.5643}, \tag{4.5}$$

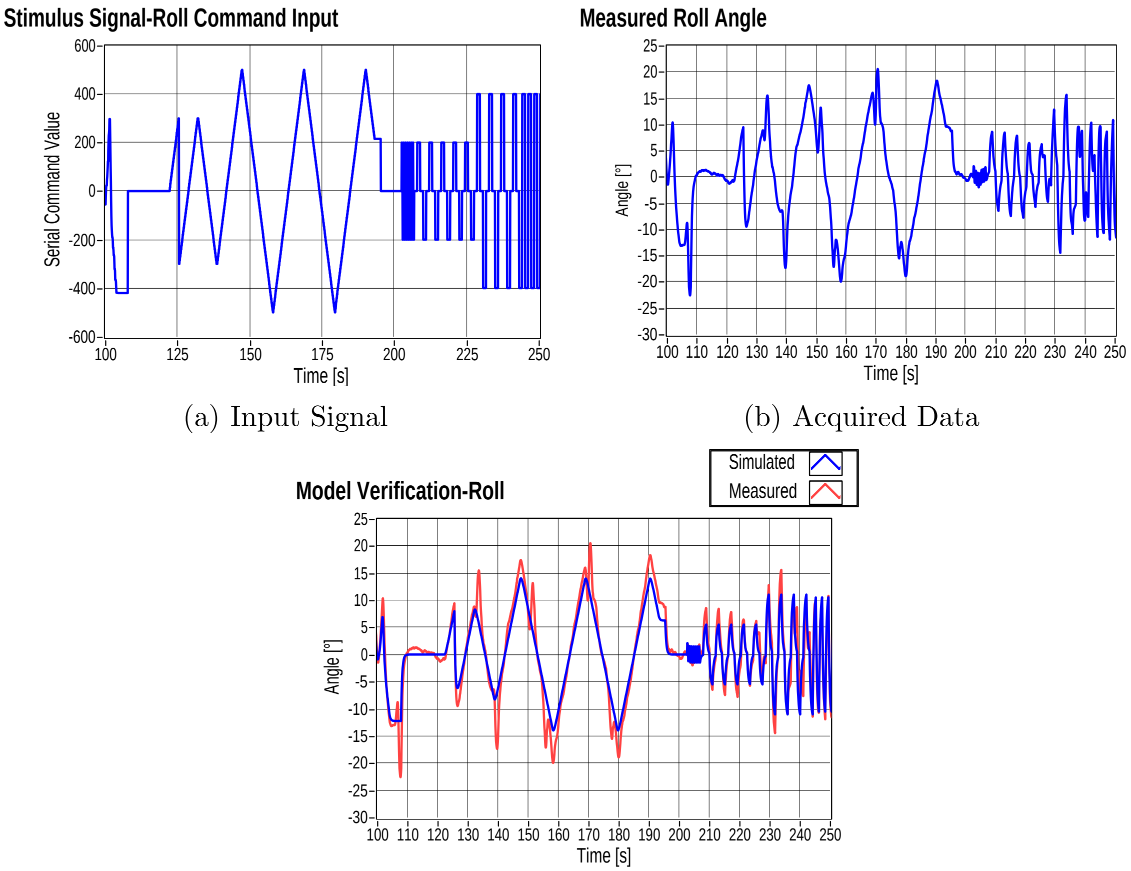
$$H_{r}\left(s\right) = \frac{r}{\mathrm{S}_\psi} = \frac{-0.25095}{s \,+\, 4.180619}. \tag{4.6}$$

For a more detailed explanation about the prediction error method and the System Identification LabVIEW Toolkit, the reader is referred to [48, 51, 52, 53].

4.2.3 Model Validation

Simulating the model response for a given output and comparing it with the actual response is the most useful and intuitively approach for validating an estimated model. Looking at the simulation results, one can inspect visually what features have been reproduced and what features have not been captured [53]. Generally, the acquired data for estimation purposes are splited in *estimation* data and in *validation* data. Furthermore, this technique was applied during the system identification procedure of the transfer functions (4.4) - (4.6). Nevertheless, the real test is checking if the model will be capable of also describing new data sets obtained from the system. To this end, a new experimental command signals formed by step and ramp inputs were sent to the quadrotor while it was attached to the support. One more time, measured data from the internal gyroscopes was acquired for comparison purposes. The stimulus signal, the measured output and the comparison with the simulated response for the roll angle are depicted in Figure 4.6. The plots for additional data included the roll angle are provided in Appendix B.

As can be seen in Figure 4.6 and in Appendix B that the models resemble the



(a) Input Signal

(b) Acquired Data

(c) Model Verification

Figure 4.6: Input Signal, Acquired Data and Model Validation for the roll angle ϕ_B.

dynamics of the measured signal. Nonetheless, there are some deviations comparing the acquired response and the simulated response. These differences are due to the effects of noise in the measurement from the gyroscopes and/or possible model errors. The parameter estimation procedure picks out the *best* model within the chosen model structure, but the crucial question is whether this *best* model is *good* enough for control purposes [53]. As it will be explained in the next chapters, the estimated models help to find satisfactory controllers. Thus, the models are possible *valid* ones for the problem that motivates the identification exercise.

Chapter 5

Controller Design

As it was mentioned in Section 2.2, an onboard controller that maintains the desired thrust and runs an attitude controller is already implemented by AscTec [38]. This controller consists of one independent PD loop for each axis. Moreover, system identification was used to estimate this part of the model, see Figure 3.4. An outer position controller should be designed and implemented for a complete flight control. The complete cascade control structure taking in count the model described in Section 3.2 is illustrated in Figure 5.1.

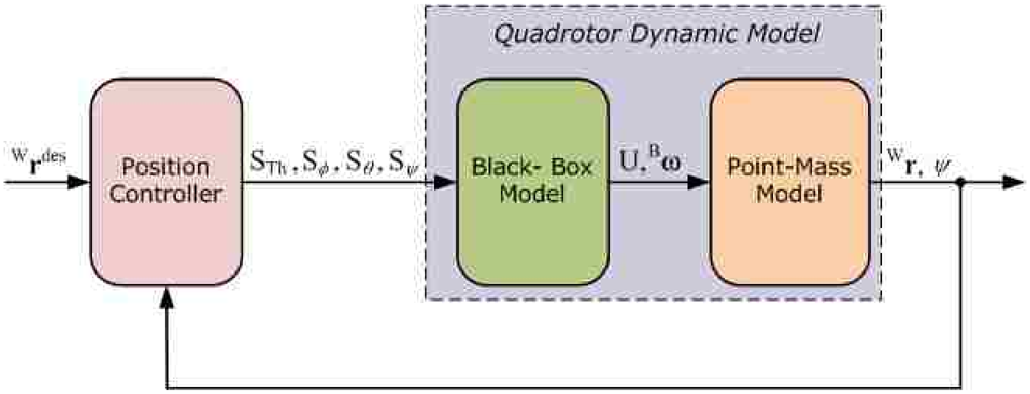


Figure 5.1: Main cascade control diagram.

5.1 Position and Heading Controller

The linear model of the system obtained in Section 3.3 is used in this section to design the required position controller. Even though the main objective is to control the $X-Y-Z$ movement of a flying object, the heading (yaw angle) is maintained at $0\,^\circ$ constantly. For that reason, the linear controller is formed by a position controller and a heading controller.

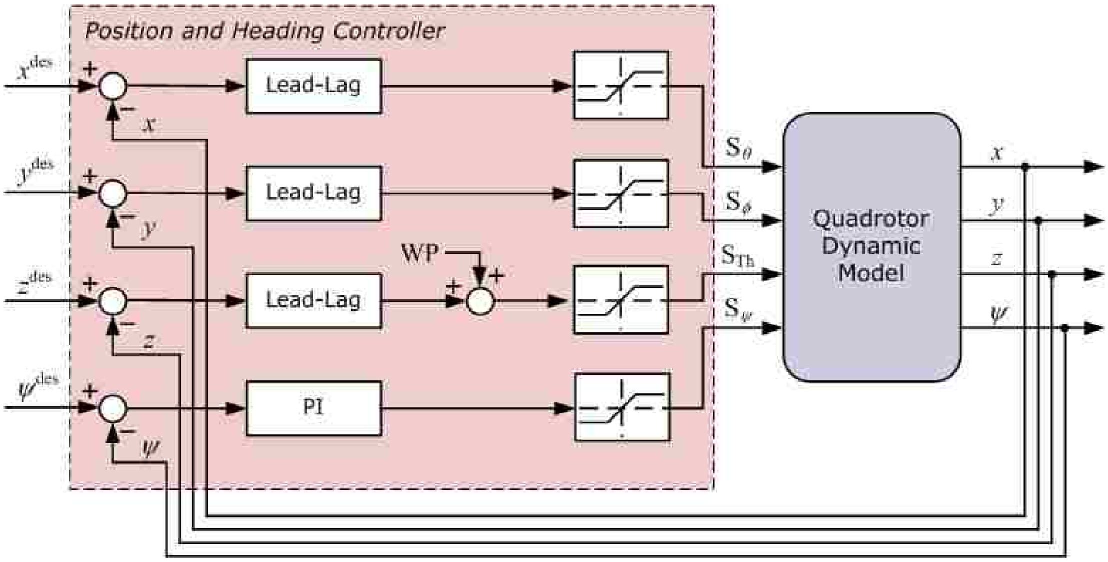


Figure 5.2: Block diagram of the position and heading controller.

Figure 5.2 illustrates the configuration details of the controllers. Furthermore, A lead-lag controller is introduced for each position axis and a PI controller for the heading. In fact, Equation (5.1) corresponds to the lead-lag controller while Equation (5.2) to the PI controller. For both equations, $V(s)$ is the control signal and $E(s)$ is the error signal.

$$V(s) = K\left(\frac{1+\tau_{ld}\,s}{1+\tau_{lg}\,s}\right)E(s), \tag{5.1}$$

where K is the gain, τ_{ld} is the lead time and τ_{lg} is the lag time,

$$V(s) = K_p \left(1 + \frac{1}{T_i\, s}\right) E(s), \tag{5.2}$$

where K_p is the gain, T_i is the integral time.

Specifically, the X and Y position controllers are formed by the lead-lag compensator 5.1 in cascade with a saturation block. The same configuration is used for the yaw angle with a PI function (5.2) instead of lead-lag. The limits of the saturation blocks are set up according to the explanation in Section 2.2.3. Meanwhile, the value of $\mathrm{S_{Th}}$ for hovering (denoted as working point WP) should be added to the output of the Z position controller since the hovering condition is considered as the operating point of the system. Since the total weight of the quadrotor is 4.954 N and from (3.5), the WP is equal to 1604. Moreover, a saturation block is used with the maximum and minimum values indicated in Section 2.2.3.

5.1.1 Simulations

Several simulations using the LabVIEW Simulation Module [54] have been performed in order to tune the controller parameters. The linear model, Section 3.3.1, and the complete model, Equation (3.10), have been used during these simulations. Indeed, the linear model is selected as initial point to determine the parameters of the controllers and then these were tuned considering the complete model. For that reason, only the results obtained with the complete model are presented.

Table 5.1 contains the parameters of the different controllers obtained by simullation. Furthermore, Figure 5.3 shows the corresponding step responses for the X, Y and Z positions. Other simulation results are also shown in Appendix C. The response for the yaw angle is not included since this compensator just maintained the angle at $0\,^{\circ}$. These parameters and responses have been obtained by simulation.

Parameter	X Controller	Y Controller	Z Controller
K	0.303	0.532	4.929
τ_{ld} [s]	1.342	1.023	0.625
τ_{lg} [s]	0.0166	0.0153	0.0943

Parameter	Yaw Controller
K_p	35
T_i [s]	1.4993

Table 5.1: Parameters of the position and heading controllers - simulation results.

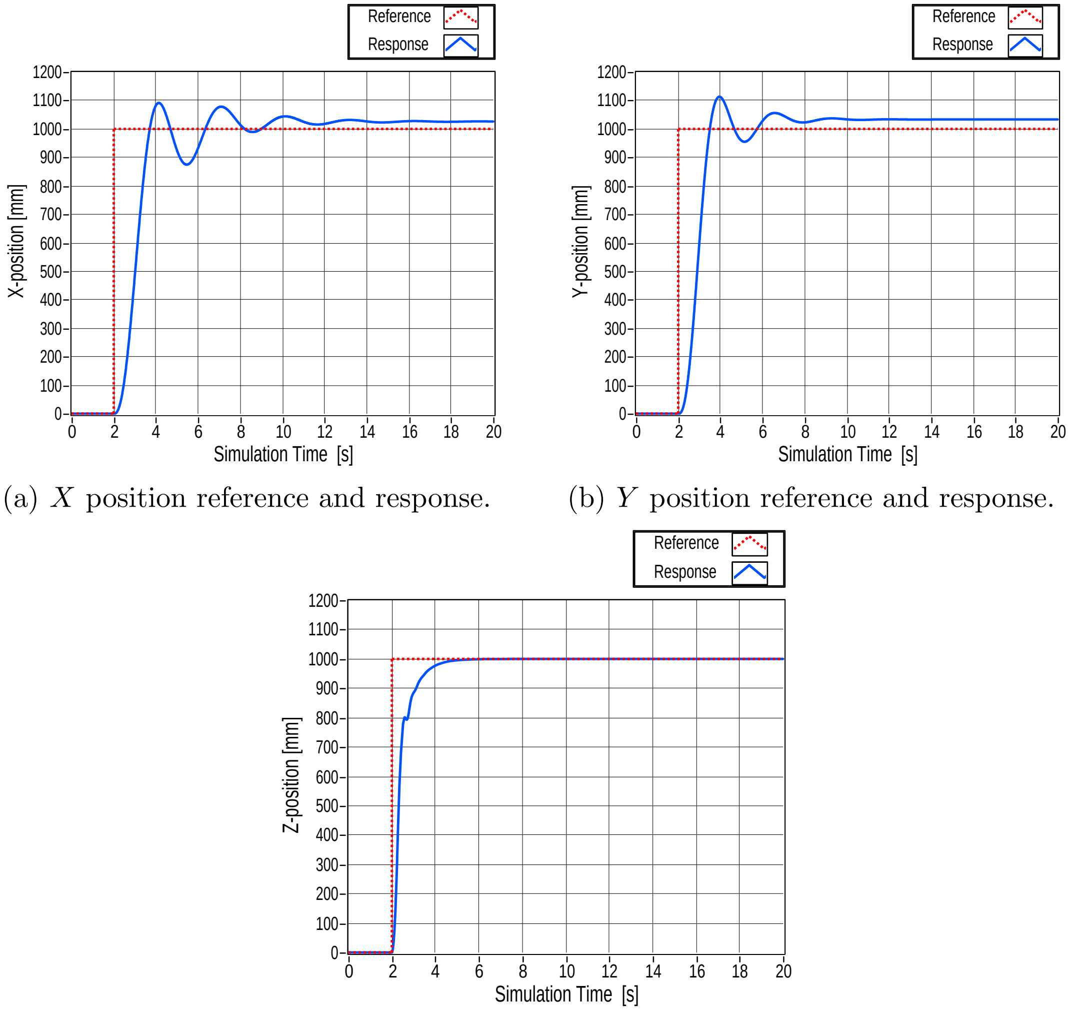
The simulated performance is satisfactory regarding the simple control synthesis approach, specially for the Z position.

5.2 Modified X and Y Controllers

From the results obtained in the simulations, see Figure 5.3 and Appendix C, and from the experimental results in Chapter 7, the performance of the X and Y controllers needs to be improved specially for trajectory tracking purposes. Besides, an accurate trajectory tracking in the XY plane requires to determine the velocity around these axes. However, the VICON sensor does not provide directly velocity measurements. Consequently, the structure of these two controllers should be modified and the X and Y velocities should be estimated.

5.2.1 Velocity Estimation

In order to simplify the estimation of the velocity, the Al-Alaoui differentiator [55, 56] is applied in the context of this work. In [55, 56], a novel first order digital differentiator is presented. This is eminently suitable for real-time applications. The resulting differentiator is obtained by taking the inverse on the transfer function of a integrator derived interpolating the rectangular and the trapezoidal digital integra-



(a) X position reference and response.

(b) Y position reference and response.

(c) Z position reference and response.

Figure 5.3: Simulations of the step response for the position controllers.

tion techniques. In fact, the resulting transfer function of the digital differentiator is given by

$$G(z) = \frac{V_x(z)}{X(z)} = \frac{8\,(z-1)}{7\,T_s\,\left(z+\frac{1}{7}\right)} \tag{5.3}$$

Parameter	X Controller		Y Controller	
	Position	Velocity	Position	Velocity
K	0.591	0.258	1.173	0.378
τ_{ld} [s]	1.002	0	0.656	0
τ_{lg} [s]	0.0087	0.0057	0.0107	0.0083

Table 5.2: Parameters of modified controllers - simulation results.

where $X(z)$ is the input, $V_x(z)$ is the output, and T_s is the sampling time. From Equation (5.3), the following difference equation is obtained

$$v_x[k] = -\frac{1}{7}v_x[k-1] + \frac{8}{7\,T_s}\left(x[k] - x[k-1]\right), \tag{5.4}$$

where x and v_x denote the position and velocity at the corresponding time instant, respectively. As a result, (5.4) is used to estimate the velocity in X and Y axes of the quadrotor at time k.

5.2.2 Modifications

Some modifications were made to the control scheme illustrated in Figure 5.2 to introduce the velocity estimator and to use this value as part of the controller. These additions can be seen in Figure 5.4. The velocity in X and Y are calculated using the Al-Alaoui differentiator in (5.4). Note that, the X and Y controllers are split in two parts: one lead-lag compensator for the position and one lead-lag for the velocity whose outputs are added and limited by a saturation block to obtain the control signal. The configuration for the Z and Yaw Controllers is kept unchanged.

5.2.3 Simulations

Extensive simulations have been performed using the modified controller in cascade with the complete dynamical model in (3.10) in order to tune the parameters. Table

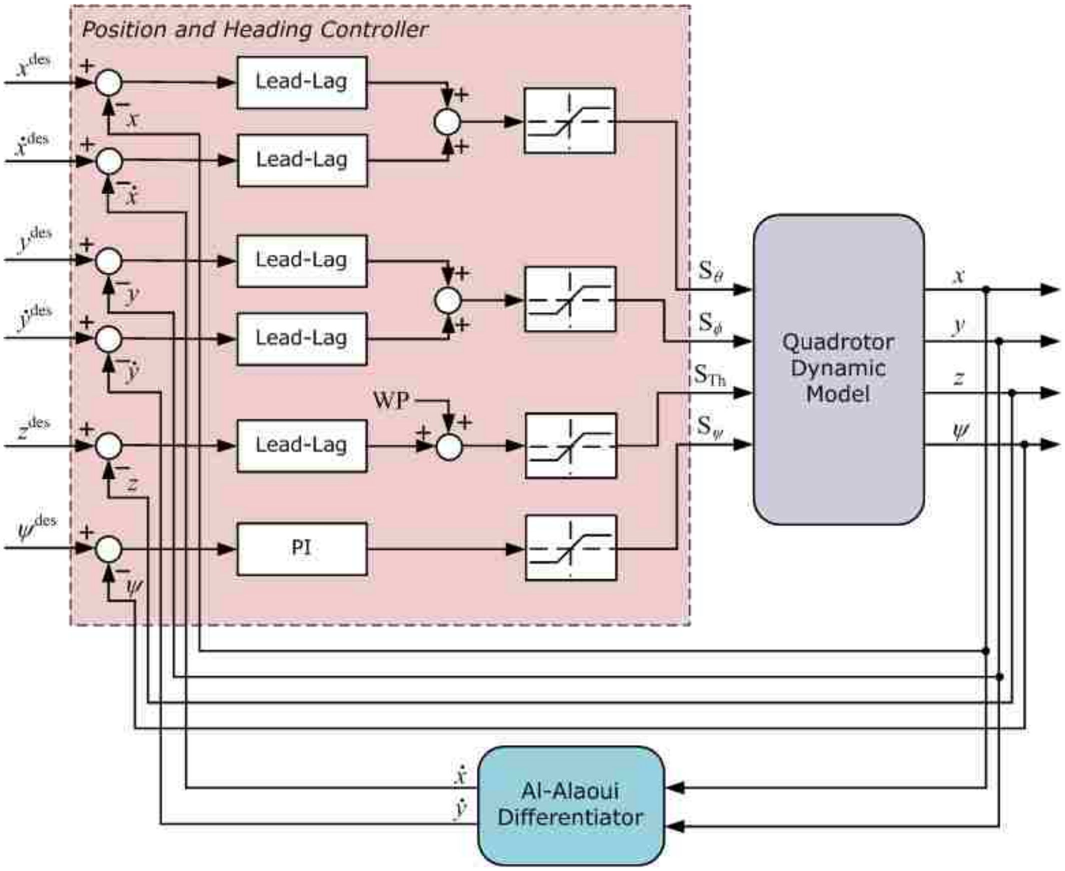
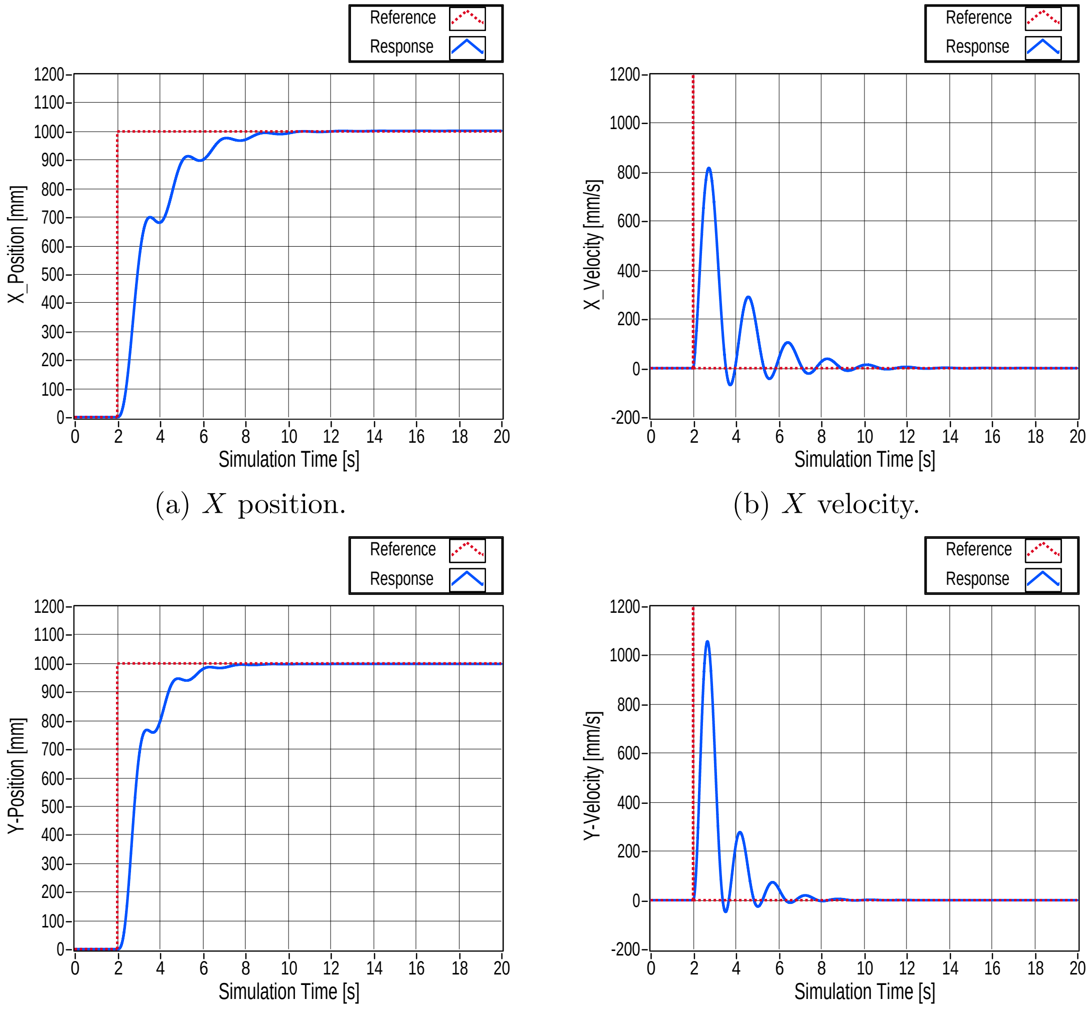


Figure 5.4: Block diagram showing the controller modifications.

5.2 presents the parameter values tuned by simulation. In addition, Figure 5.5 shows the step response using the modifications where the plots for the velocity response are also included. Appendix C contains a compilation of the simulation responses to different types of input.

The control strategy of designing a position controller for the quadrotor has been developed in this Chapter. Simulation results of the proposed controllers and their modifications for suitable trajectory tracking have been also provided. The control architecture where these controllers are going to be implemented and tested will be presented in the next chapter.



(a) X position.

(b) X velocity.

(c) Y position.

(d) Y velocity.

Figure 5.5: Simulations of the step response for the X and Y modified controllers.

Chapter 6

Control System Implementation

The main components of the control system, the VICON MX and the NI CompactRIO, were introduced in Chapter 2 together with a description of the aerial platform used in this thesis, the AscTec Hummingbird quadrotor. All these devices are part of the Marhes Laboratory at University of New Mexico where the control system has been implemented and tested.

6.1 Hardware Architecture

The hardware architecture is based in the VICON MX System architecture described in Section 2.3 and shown in Figure 2.8. This system tracks the orientation and position of the quadrotor within the capture room. Therefore, a real-time engine should be added to this scheme to acquire the data from VICON, implement the position controllers and transmit the control signals to the aerial vehicle creating a real-time control loop. For this reason, the NI cRIO system, Section 2.4, and a base station are included as part of the hardware architecture. Moreover, the wireless serial communication between the quadrotor and the NI cRIO is established via XBee embedded RF modules providing cost-effective wireless connectivity in a ZigBee network. The complete hardware architecture is shown in Figure 6.1.

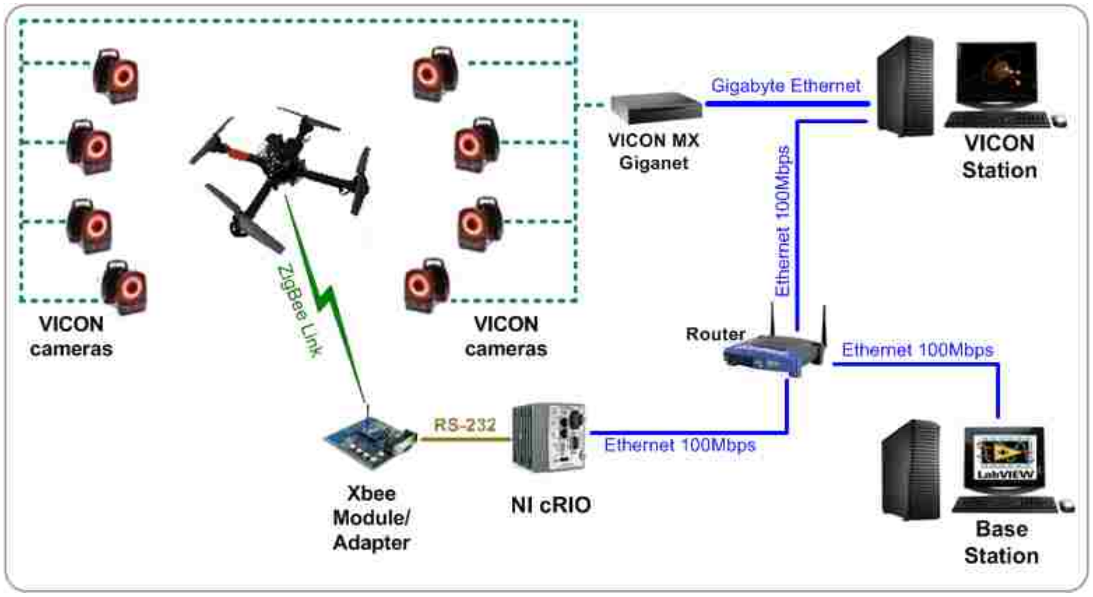


Figure 6.1: Hardware architecture.

6.2 Software Architecture

Based on the hardware configuration, the controllers designed and simulated in Chapter 5 can be implemented. The entire control application is created in LabVIEW building as Virtual Instruments (front panel plus block diagram), VIs, to acquire data from VICON, to calculate the control commands and to send these commands to the quadrotor via an XBee module. The basic software architecture shown in Figure 6.2 includes two VIs that are interacting within a LabVIEW Project.

Through this architecture, delays similar to the ones presented in [14] are observed. The pose of the flying robot is acquired and stored by the VICON Interface

VI that runs on the Base Station at 225 Hz, see Section 2.3. The pose stored data is read by the Quadrotor Interface VI running in the NI cRIO. Also, this VI computes the control commands and sends them to the quadrotor at a fixed rate of 100 Hz. Even though the scan engine period of the cRIO is configurable and set up at 1 ms (1 kHz), this fixed rate is due to the limitation in the serial interface data rate with the XBee modules (57.6 kbps). Thus, the command data is sent with a frequency over the critical limit of 10 Hz, see Section 2.2.2.

Under this system architecture, it is possible to manage more than one quadrotor simultaneously adding extra NI cRIOs to the network. For multiple vehicles, the required modifications are outlined in Section 6.3.3 while an experimental application is described in Section 7.3.4.

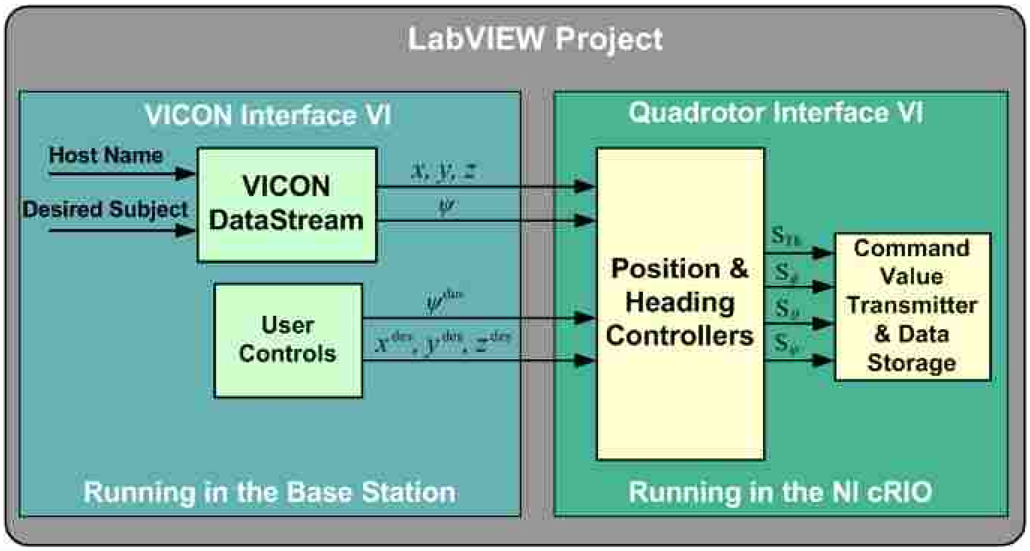
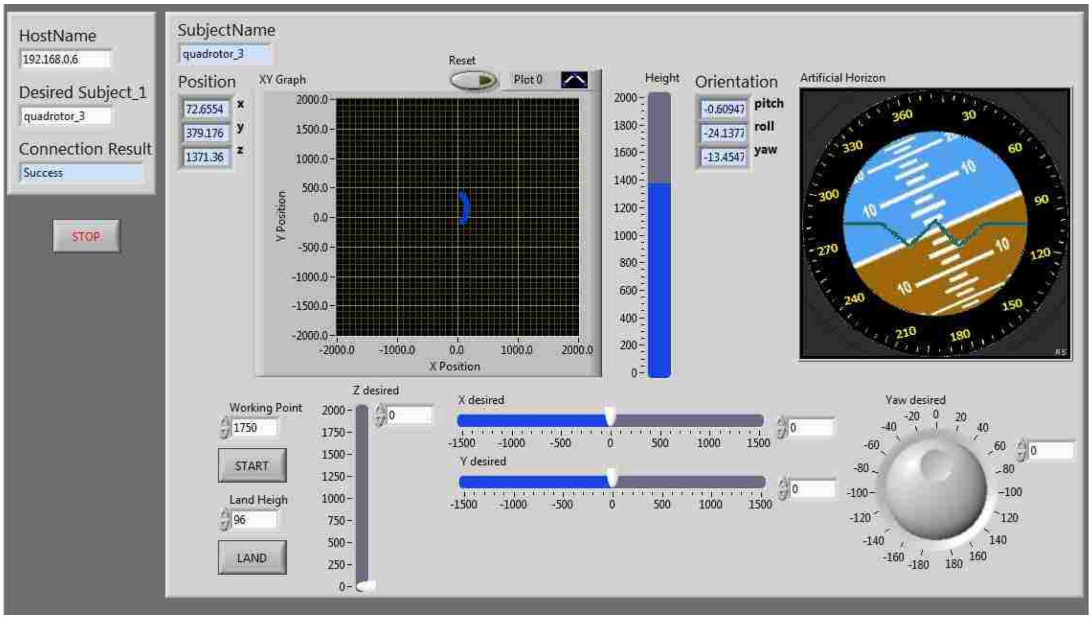


Figure 6.2: Software architecture - basic structure.



HostName: 192.168.0.6
Desired Subject_1: quadrotor_3
Connection Result: Success
SubjectName: quadrotor_3
Position: x 72.6554, y 379.176, z 1371.36
Orientation: pitch -0.60947, roll -24.1377, yaw -13.4547
Working Point: 1750
Land Heigh: 96
Z desired: 0
X desired: 0
Y desired: 0
Yaw desired: 0

Figure 6.3: Front panel of the VICON interface VI.

6.2.1 VICON Interface VI

LabVIEW offers the required tools to create a friendly GUI with adequate indicators to display real-time data during flight tests. Also, LabVIEW has the necessary user controls to allow that a single operator can execute a desired application. In fact, this and the next section describe the functionality of the two VIs that are part of the software architecture.

A screen shot of the front panel of this VI is presented in Figure 6.3 where an artificial horizon, a XY graph and a height indicator are used to display the orientation, XY position and Z position of the quadrotor, respectively.

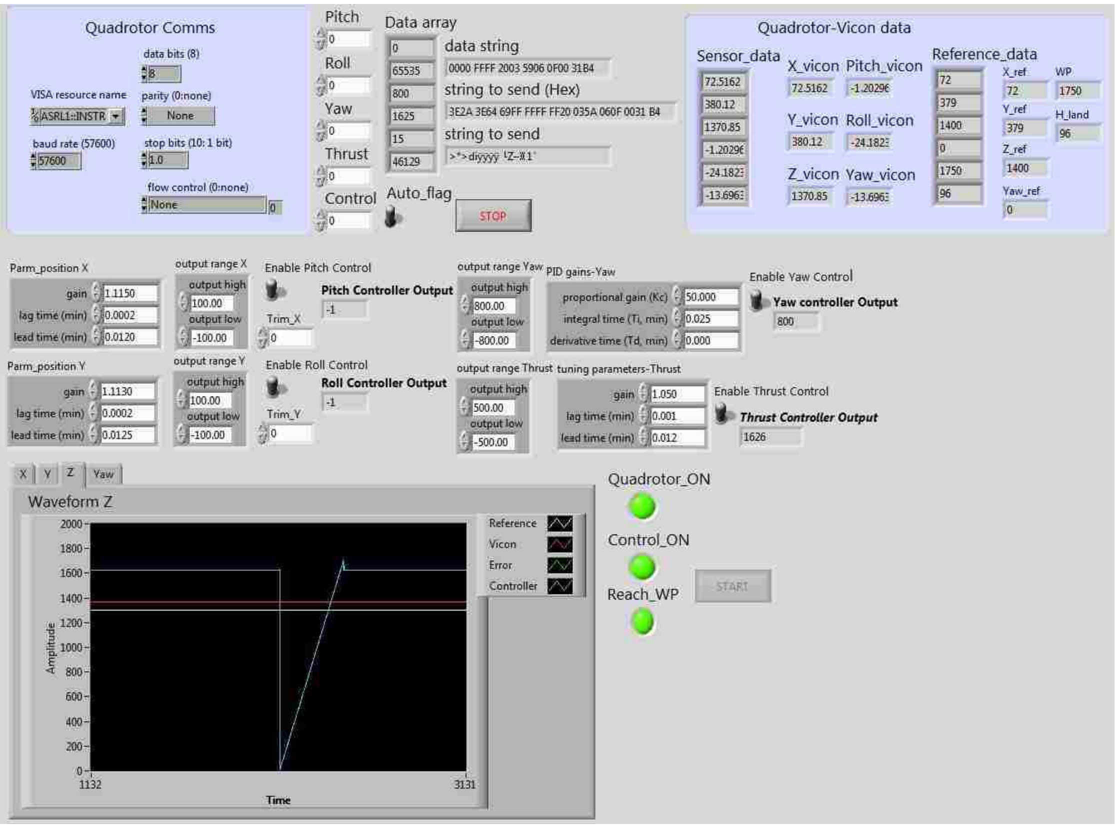
This VI acquires the pose data from the VICON System, reads the data from the user controllers located at the front panel and stores all this information in global variables. To set up the VICON Data SDK in LabVIEW, the steps described in

[57] should be followed. The DS-SDK dll file, (*ViconDataStreamSDK_DotNET.dll*) should be included in the LabVIEW Project. Consequently, the function calls within the SDK allow to connect to and request data from the VICON DataStream. The manual for developers [45] includes the list and parameters of all the SDK functions. The VICON DataStream block in Figure 6.2 represents the different functions to establish connection with the Host VICON Station, select the desired subject created in the VICON Tracker and acquire its position and orientation data. Subsequently, the position and heading angle (yaw angle) are saved in global variables together with the desired values (X desired, Y desired, Z desired, Yaw desired, Working Point and Land Height). Once the VI is running, it constantly acquires and stores pose data and desired values.

6.2.2 Quadrotor Interface VI

Figure 6.4 is a screen shot of the front panel of this VI. All the indicators of the global variable readings, the controllers for the lead-lag and PI compensators, the configuration controllers and frame-to-send indicator of the communication with the quadrotor, and helpful charts to tune the controllers are included in the front panel.

This VI accesses the global variables, implements the controllers and sends the calculated control commands to the quadrotor. The Lead-Lag controllers and the PI controller designed in Chapter 5 are implemented using the PID Lead-Lag VI and the PID VI, respectively. These VIs are available in the LabVIEW PID Control Toolkit [58]. The inputs to these controllers are the values read from the global variables and their outputs are used to create the frame depicted in Figure 2.7 that is transmitted to the quadrotor according to the protocol explained in Section 2.2.2. Indeed, this protocol has been implemented in a SubVI to facilitate the control commands transmission. For future plotting purposes, the reference, response, error and control values for X, Y and Z positions and Yaw angle are stored in a text file in an external USB drive that can be connected to the NIcRIO.



Quadrotor Comms

data bits (8): 8
VISA resource name: ASRL1::INSTR
parity (0:none): None
baud rate (57600): 57600
stop bits (10: 1 bit): 1.0
flow control (0:none): None 0

Pitch: 0
Roll: 0
Yaw: 0
Thrust: 0
Control: 0

Data array: 0, 65535, 800, 1625, 15, 46129
data string: 0000 FFFF 2003 5906 0F00 31B4
string to send (Hex): 3E2A 3E64 69FF FFFF FF20 035A 060F 0031 B4
string to send: >*>diÿÿÿÿ ˡZ-※1`

Auto_flag

Quadrotor-Vicon data

Sensor_data: 72.5162, 380.12, 1370.85, -1.20296, -24.1823, -13.6963
X_vicon: 72.5162
Pitch_vicon: -1.20296
Y_vicon: 380.12
Roll_vicon: -24.1823
Z_vicon: 1370.85
Yaw_vicon: -13.6963
Reference_data: 72, 379, 1400, 0, 1750, 96
X_ref: 72
Y_ref: 379
Z_ref: 1400
Yaw_ref: 0
WP: 1750
H_land: 96

Parm_position X: gain 1.1150, lag time (min) 0.0002, lead time (min) 0.0120
output range X: output high 100.00, output low -100.00
Enable Pitch Control
Trim_X: 0
Pitch Controller Output: -1

Parm_position Y: gain 1.1130, lag time (min) 0.0002, lead time (min) 0.0125
output range Y: output high 100.00, output low -100.00
Enable Roll Control
Trim_Y: 0
Roll Controller Output: -1

output range Yaw: output high 800.00, output low -800.00
PID gains-Yaw: proportional gain (Kc) 50.000, integral time (Ti, min) 0.025, derivative time (Td, min) 0.000
Enable Yaw Control
Yaw controller Output: 800

output range Thrust: output high 500.00, output low -500.00
tuning parameters-Thrust: gain 1.050, lag time (min) 0.001, lead time (min) 0.012
Enable Thrust Control
Thrust Controller Output: 1626

Quadrotor_ON
Control_ON
Reach_WP

Figure 6.4: Front panel of the quadrotor interface VI.

6.3 System Adaptations

A variety of modifications, specialy around the software architecture, have been made to accomplish a variety of tasks that will be explained in the next Chapter. These modifications do not affect the whole system and in most of the different cases, only the VICON Interface VI needs some minor modifications.

6.3.1 Trajectory Tracking

As in Section 5.2, the performance of the X and Y controllers needs to be improved specially for trajectory tracking purposes. Furthermore, the modified configuration is shown in Figure 5.4. Hence, the numerical differentiation given by (5.4) to compute the velocities is added to the VICON Interface VI and the controllers in the Quadrotor Interface VI are updated with the ones in Section 5.2. Moreover, the desired trajectory (position & velocity) is read from a text file that contains the data generated previously. Figure 6.5 presents the modifications to the software architecture to implement trajectory tracking.

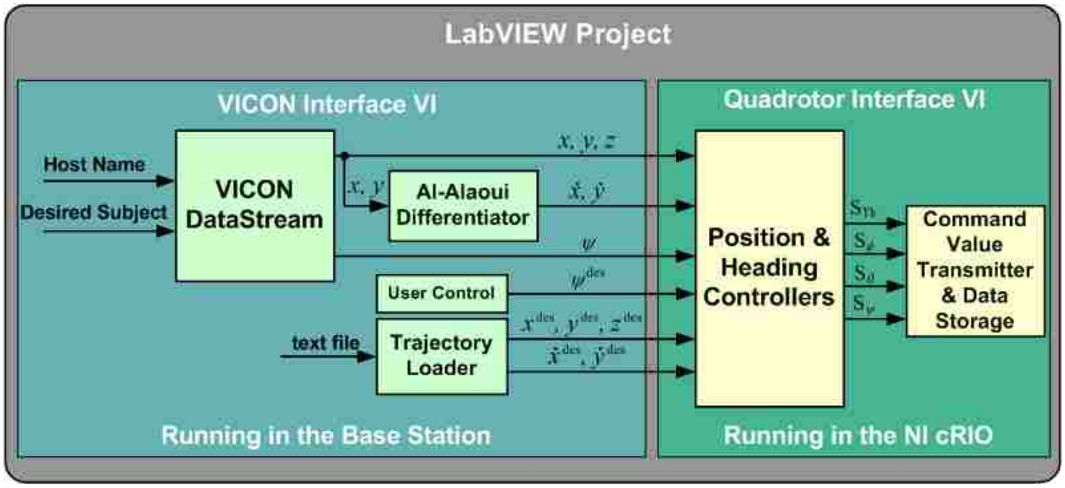


Figure 6.5: Software architecture - trajectory tracking.

6.3.2 Adding a Mobile Target

Since the VICON System has the capability of tracking more than one object within the capture room at the same time, a second moving object could be used as XYZ reference mobile point. In order to do this, the changes in the software architecture shown in Figure 6.6 are introduced. Notice that now the reference values for the position are taken from the mobile object defined as *Intruder*. Additionally, an offset value in each direction could be added to avoid possible collisions with the mobile target.

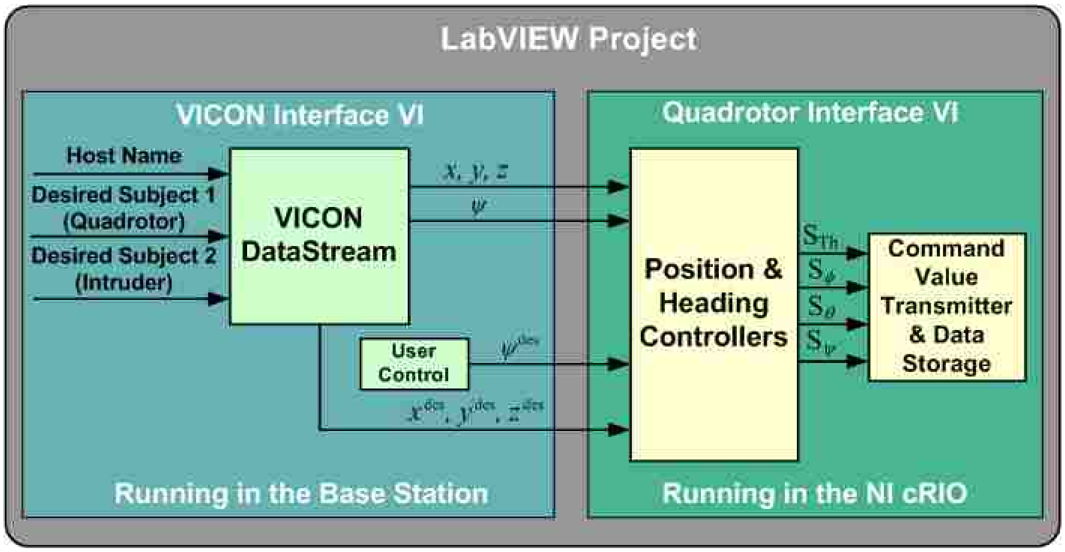

LabVIEW Project
VICON Interface VI
Host Name
Desired Subject 1 (Quadrotor)
Desired Subject 2 (Intruder)
VICON DataStream
x, y, z
ψ
User Control
ψ^{des}
$x^{des}, y^{des}, z^{des}$
Running in the Base Station
Quadrotor Interface VI
Position & Heading Controllers
S_{Th}
S_{ϕ}
S_{θ}
S_{ψ}
Command Value Transmitter & Data Storage
Running in the NI cRIO

Figure 6.6: Software architecture - adding a mobile target.

6.3.3 Managing Multiple Quadrotors

One of the purposes of this thesis is to develop an architecture capable of easily adding similar UAVs to be controlled at the same time. As it was indicated before, the system can manage more than one quadrotor simultaneously adding extra NI cRIOs to the network. Therefore, the required modifications to achieve this goal affect both the hardware and software parts of the architecture. Pertaining to the hardware, see Figure 6.7, a NI cRIO per quadrotor should be connected to the network. For the software part, the VICON Interface VI is updated to define the necessary global variables according to the number of quadrotors. In addition, the Quadrotor Interface

VI should be should be downloaded to each NI cRIOS updating the corresponding variables. A scheme of the modifications of the software is illustrated in Figure 6.8.

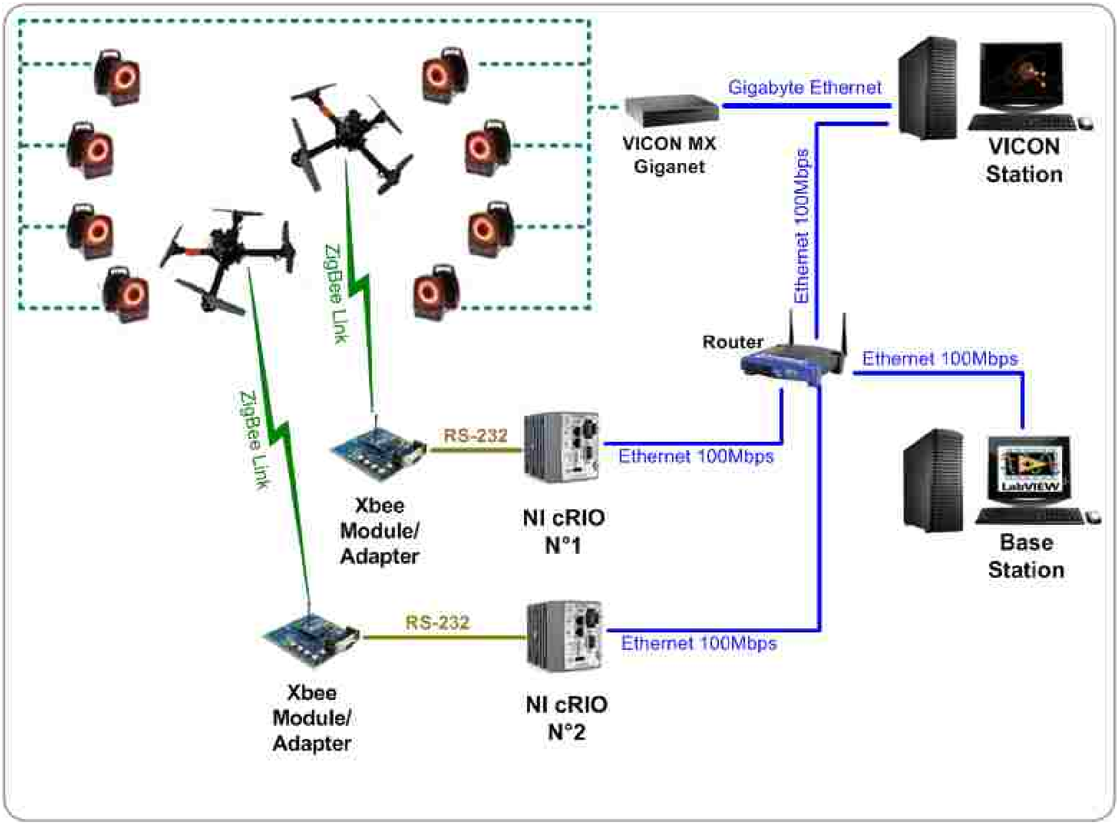


Figure 6.7: Hardware architecture - multiple UAVs.

In this Chapter, the implementation of the real-time control system architecture has been described including the possible variations that it can have. Different experiments and applications have been performed using this system and they will be presented in the next Chapter.

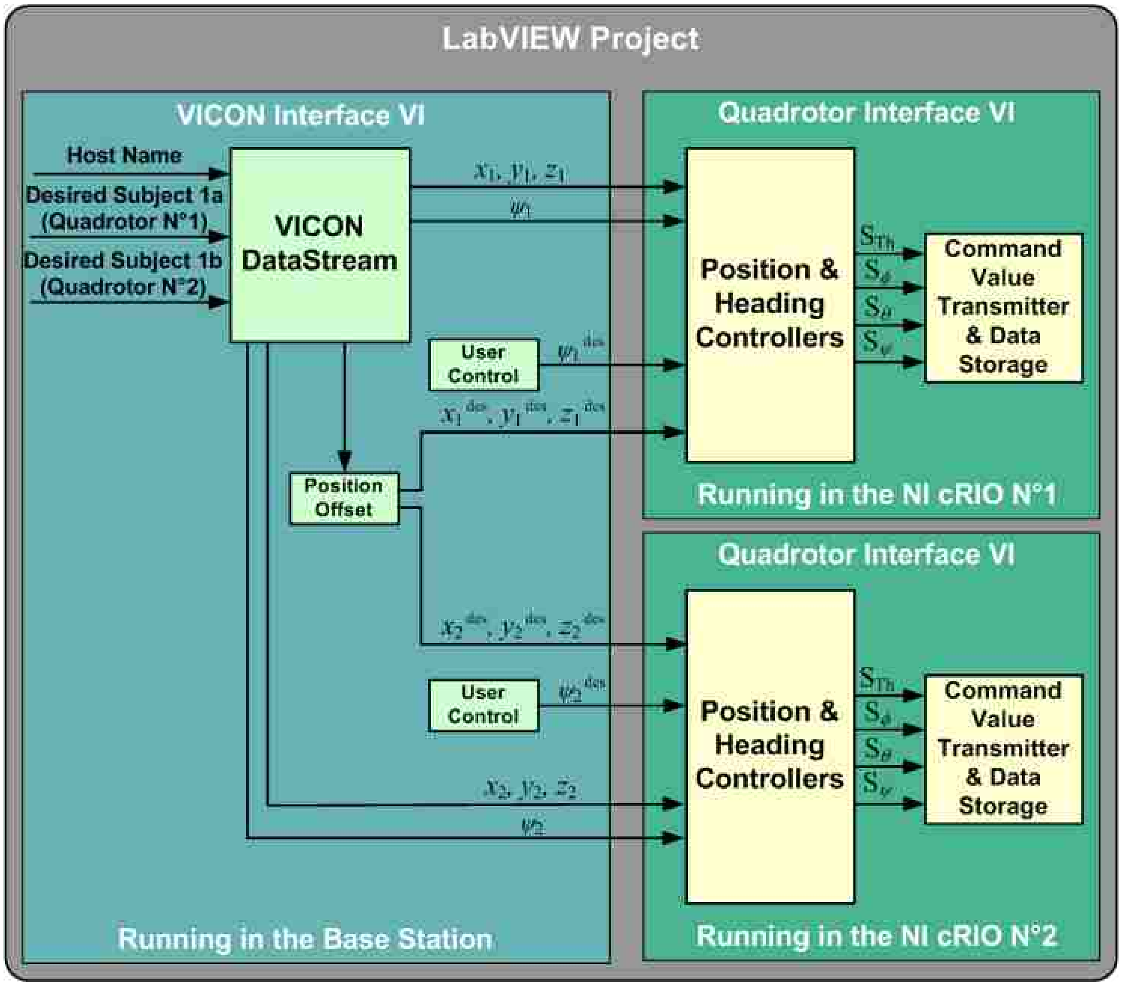

LabVIEW Project
VICON Interface VI
Host Name
Desired Subject 1a (Quadrotor N°1)
Desired Subject 1b (Quadrotor N°2)
VICON DataStream
x_1, y_1, z_1
ψ_1
User Control
ψ_1^{des}
$x_1^{des}, y_1^{des}, z_1^{des}$
Position Offset
$x_2^{des}, y_2^{des}, z_2^{des}$
User Control
ψ_2^{des}
x_2, y_2, z_2
ψ_2
Running in the Base Station
Quadrotor Interface VI
Position & Heading Controllers
S_{Th}
S_ϕ
S_θ
S_ψ
Command Value Transmitter & Data Storage
Running in the NI cRIO N°1
Quadrotor Interface VI
Position & Heading Controllers
S_{Th}
S_ϕ
S_θ
S_ψ
Command Value Transmitter & Data Storage
Running in the NI cRIO N°2

Figure 6.8: Software architecture - multiple UAVs.

Chapter 7

Experimental Results and Applications

To evaluate the performance of the real-time control architecture, extensive flight tests have been performed and their results are described here. Also, the four applications presented in this Chapter demonstrate the architecture flexibility to adapt to different practical tasks. These applications taken together show that the objective of this work, outlined in Chapter 1, has been achieved successfully.

To perform a test or application, the next procedure is followed:

1. Turn the motors on;
2. take off vertically and reach a desired height at the current XY position;
3. hover at the current XYZ position during a fixed time;
4. run the experiment or application;
5. land at the final location determined by the experiment or application;
6. turn off the motors.

Parameter	X Controller	Y Controller	Z Controller
K	1.31	1.32	1.05
τ_{ld} [s]	0.852	0.87	0.72
τ_{lg} [s]	0.048	0.054	0.06

Parameter	Yaw Controller
K_p	50
T_i [s]	1.5

Table 7.1: Parameters of the position and heading controllers - experimental tuning.

Parameter	X Controller		Y Controller	
	Position	Velocity	Position	Velocity
K	1.115	0.545	1.113	0.535
τ_{ld} [s]	0.72	0	0.75	0
τ_{lg} [s]	0.012	0.012	0.012	0.012

Table 7.2: Parameters of the modified controllers - experimental tuning.

Before proceeding with the flight tests and applications, the controller parameters found by simulation, Table 5.1 and Table 5.2 should be tested and tuned.

7.1 Tuning the Controllers

The controller gains can be tuned empirically by trial and error, but this is a time-consuming process. However, this process was improved using as a reference the controllers found by simulation and described in Sections 5.1 and 5.2.

Thanks to the control field in the serial communication frame, see Section 2.2.2, the controllers can be tested and tuned individually with a pilot controlling the remaining three input signals. Once a stable hovering is achieved, a fine tuning is done by giving step inputs individually to each one of the three axes. Table 7.1 and Table 7.2 show the control parameters for the different controllers.

Comparing the values of the controller parameters in Table 7.1 and Table 7.2 with the ones obtained by simulation in Chapter 5, the differences are small. In fact, tuning by simulation before facilitates the hand tuning over the real platform.

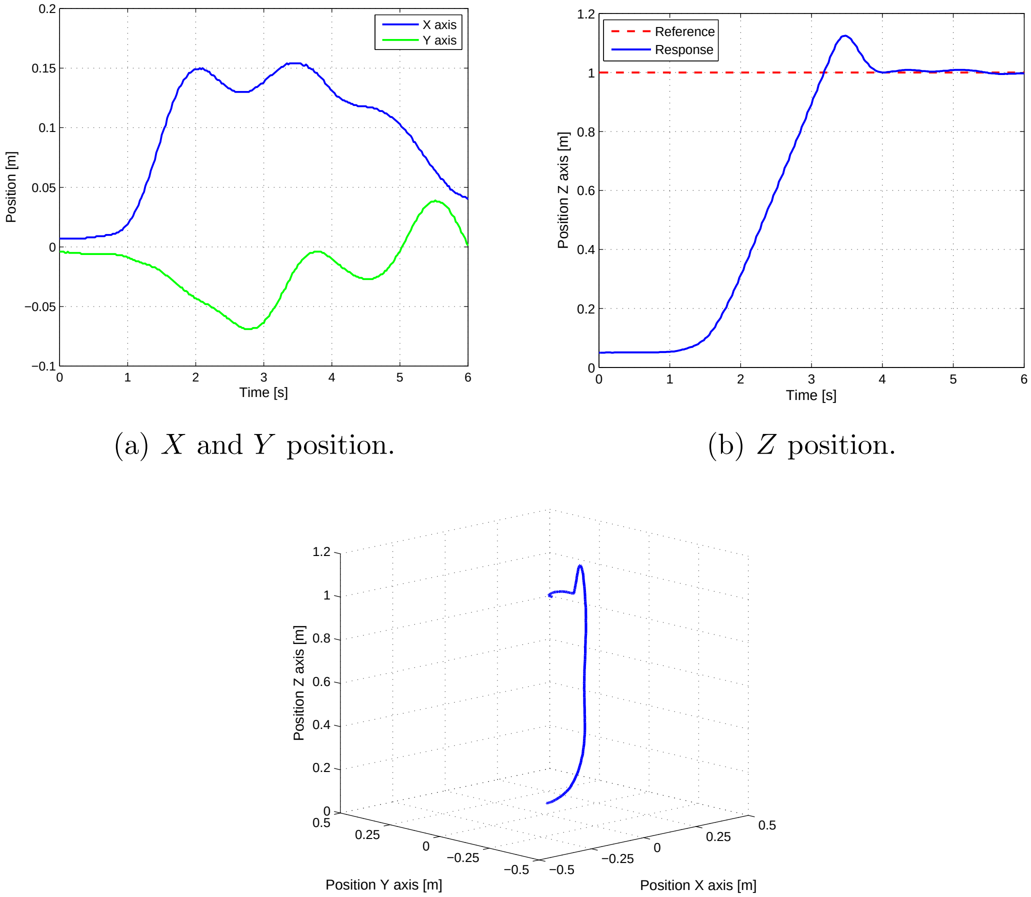
7.2 Experimental Tests

Once the controllers are tuned, experimental flights have tested the performance of the whole architecture. These tests include autonomous take off and landing, hovering, aerial target tracking, and trajectory tracking.

7.2.1 Autonomous Take off and Landing

During the experiments for tuning, an experienced pilot starts the motors and takes off the quadrotor reaching the desired height where the controllers are activated. Also, the pilot lands the quadrotor once the experiment is over and the controllers are deactivated. Therefore, there is the need of implementing an autonomous take off and landing. To accomplish these goals, only the Z controller is modified and implemented as a state machine that employs three different states: take off, normal controller, and landing.

The take off is started once the START button in the GUI in Figure 6.3 is pressed. First, the motors are turned on sending the respective command explained in [42]. Second, an incremental value of S_{Th} in steps of 5 units is sent to the quadrotor while the X, Y and Yaw controller are working. This value is incremented until it reaches the WP value. This value was calculated in Section 5.1 using (3.5) and it is equal to 1604. However, experimentally this value has been modified to 1650. In fact, this parameter should be checked every time that the quadrotor's propellers are changed or a payload is attached to the quadrotor. Once the WP is reached, it is maintained until the Z position error respect to the desired height is less than 10 mm when the Normal Controller is activated.



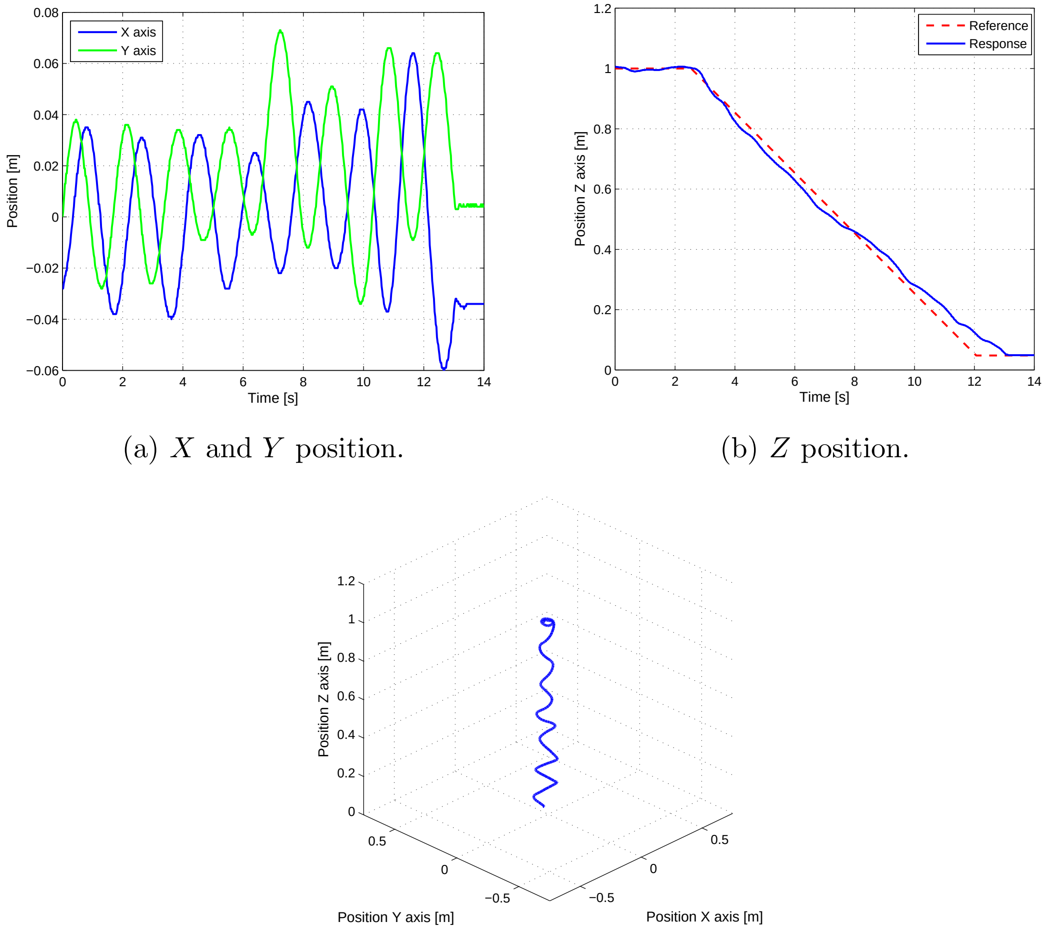
(a) X and Y position.

(b) Z position.

(c) 3D plot.

Figure 7.1: Experimental results for the take off maneuver.

The Normal Controller, the Z position controller explained in Section 5.1, will continue working to the end of the experiment. Once the experiment or application is over, the LAND button should be pressed and the quadrotor will descend autonomously until its altitude reaches a desired land height. Thus, the value of reference of the Z position is decreased by 3 units every cycle. When the Z position error with the desired land height is less than 10 mm, the S_{Th} is set up to 0 and the motors are turned off after 1 second.



(a) X and Y position.

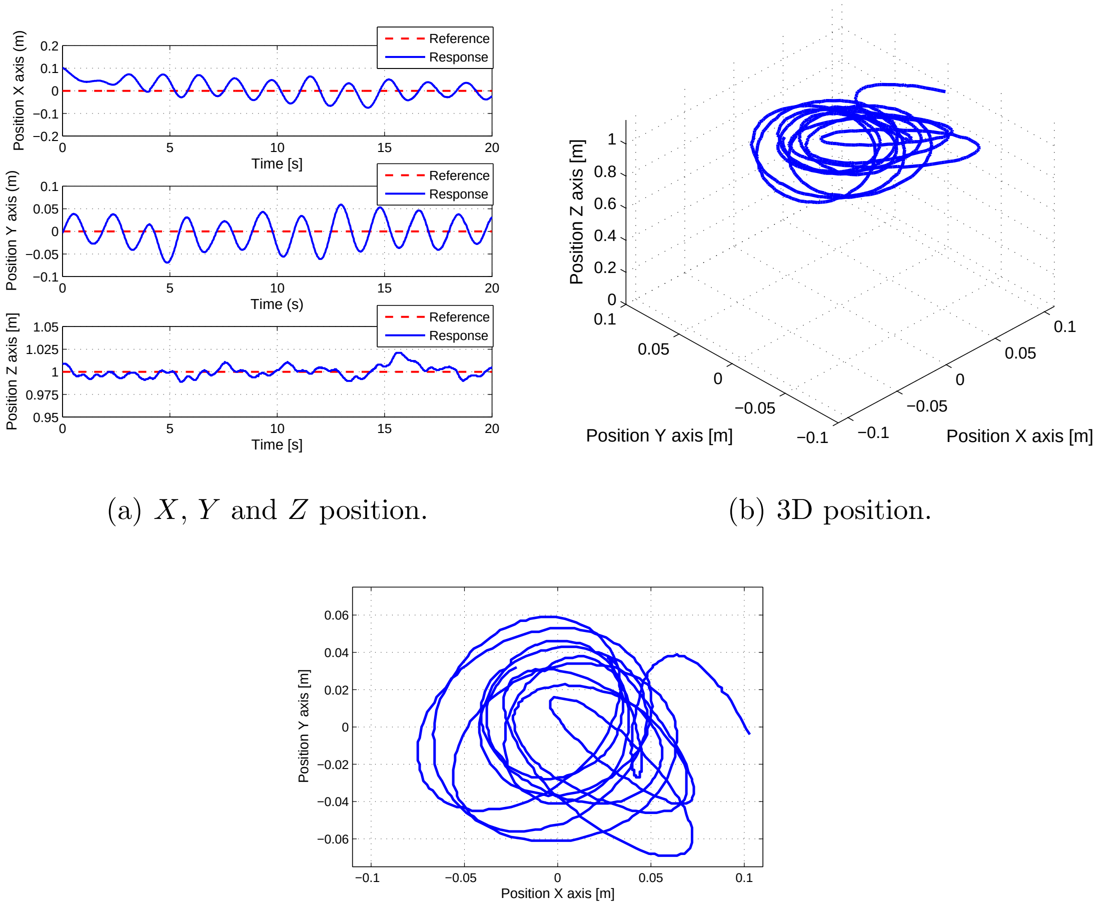
(b) Z position.

(c) 3D plot.

Figure 7.2: Experimental results for the landing maneuver.

Figure 7.1 and Figure 7.2 illustrate the results for the take off and landing, respectively. In the experiment, the quadrotor is located at the XY position $(0, 0)$ and it had to reach a desired height of 1 m. The take off maneuver is achieved in about 3 s once it is initiated and the land maneuver in about 10 s. The motion in the XY

plane is accurately controlled during the take off and landing maneuvers. During the take off, the X position error is less than 0.16 m and the Y position error less than 0.06 m. Meanwhile for the landing, the X position error is less than 0.06 m and the Y position error less than 0.08 m.



(a) X, Y and Z position.

(b) 3D position.

(c) 2D position.

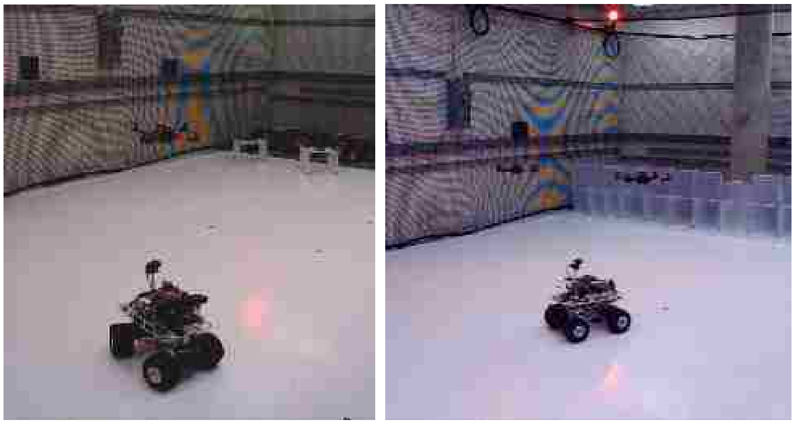
Figure 7.3: Experimental results during hovering.

7.2.2 Hovering Performance

The reference during the test is set to $(0, 0, 1)$ m. The results for 20 s of a total of around 450 s, that corresponds to the flight time with a fully charged battery, are presented in Figure 7.3. From these results, the quadrotor achieved a stable and accurate hovering and was able to stay within of a circle of radius of 0.08 m in the XY plane. At the same time, the error in the Z position is less than 0.025 m.

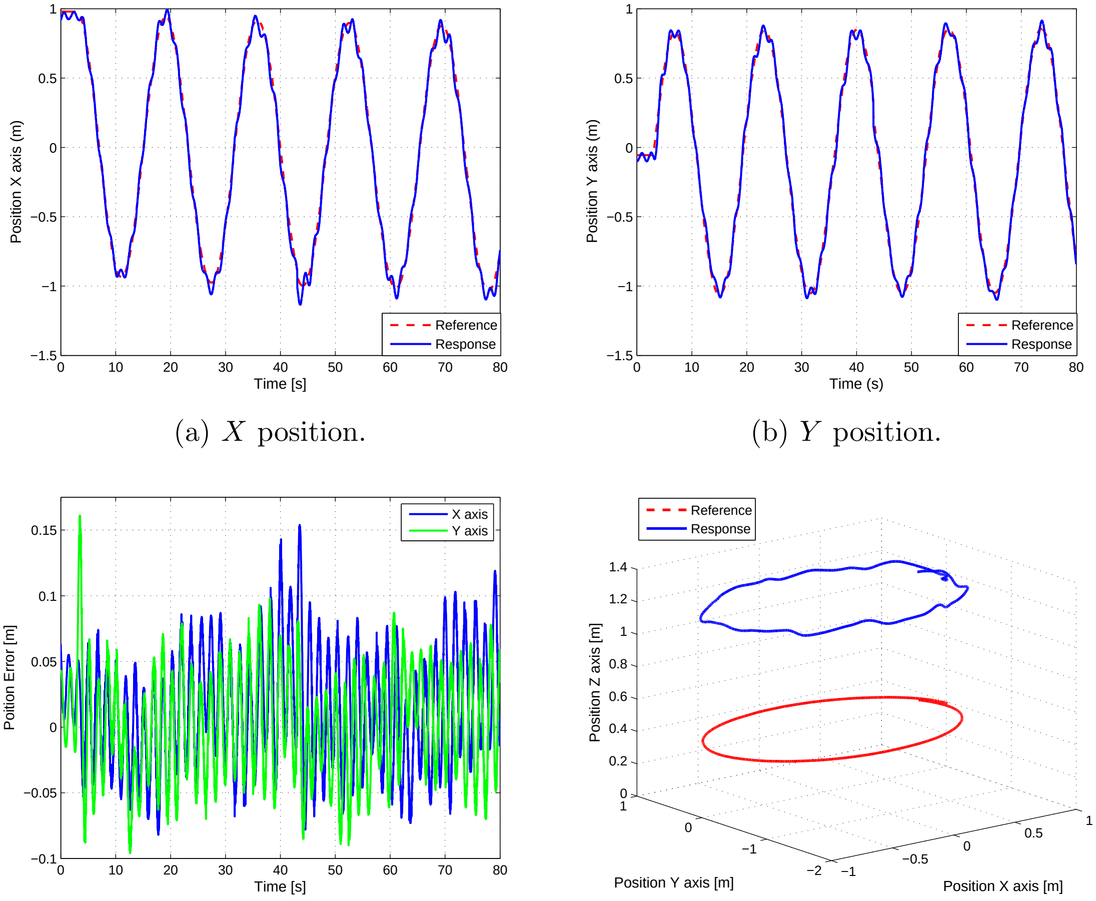
7.2.3 Aerial Tracking of a Mobile Target

Instead of giving a sequence of way-points to follow, a mobile ground robot is used as ground reference as shown in Figure 7.4(a). Using the modifications depicted in Figure 6.6, the position of a ground vehicle is acquired and used as reference input for X and Y. Furthermore, two quadrotors working under the schemes of Figures 6.7 and 6.8 have been used to track a mobile ground target as illustrated in Figure 7.4(b).



(a) By one quadrotor. (b) By two quadrotors.

Figure 7.4: Aerial tracking of a mobile ground robot.



(a) X position.

(b) Y position.

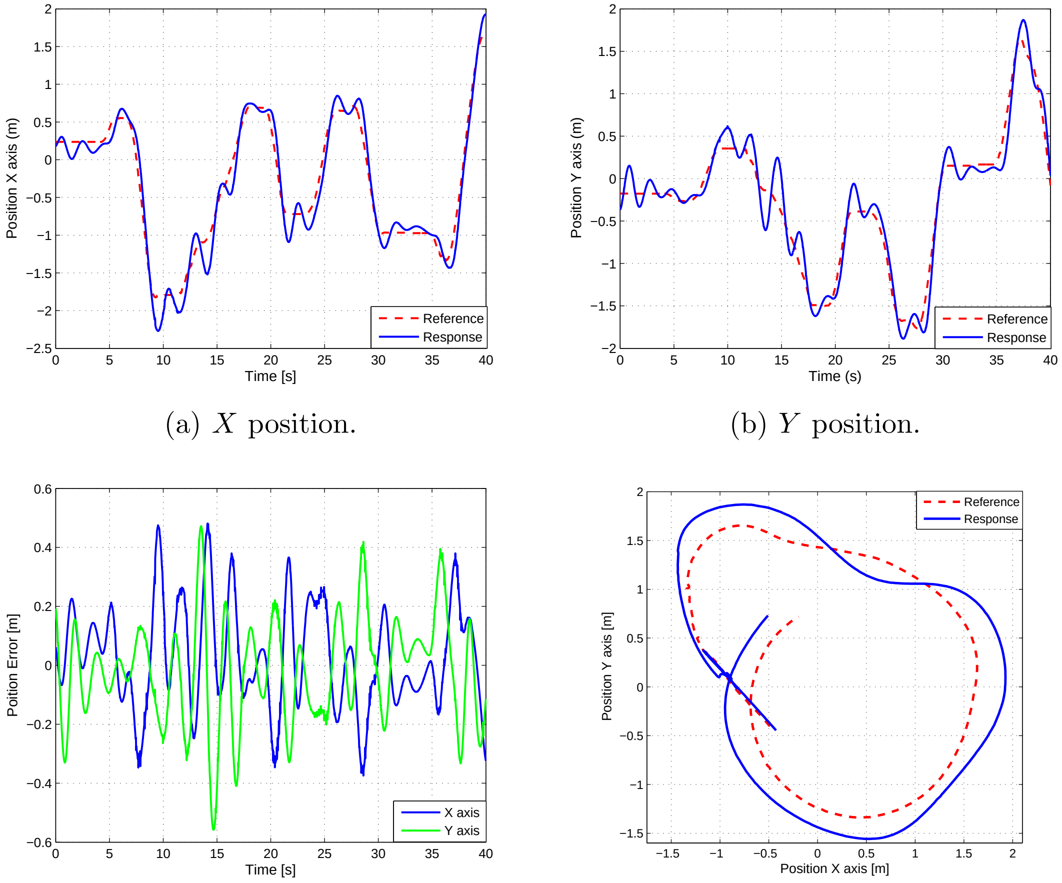
(c) Position error.

(d) 2D plot.

Figure 7.5: Results from tracking a ground robot that is describing a circular trajectory.

For moving the ground vehicle, two options have been employed: (i) moving according to a defined trajectory, and (ii) being driven by a human operator using a joystick. Figure 7.5 includes the results when the ground target follows a circular path with a constant speed of 0.4 m/s. As it can be seen, the position error is less than 0.16 m for the X and Y axes.

Figure 7.6 shows the results when the target is driven manually. In this scenario, the target moves freely around the capture room having the ability of going faster or slower at any moment. In this case, the position error is less than 0.5 m for X and 0.6 m for Y. The increment in the error is due to the ability of the ground vehicle of fast changes in its speed and direction. As it can be seen in the video for this test, see [59], the ground vehicle even can go backwards. Similar results are obtained when two quadrotors track the mobile target.



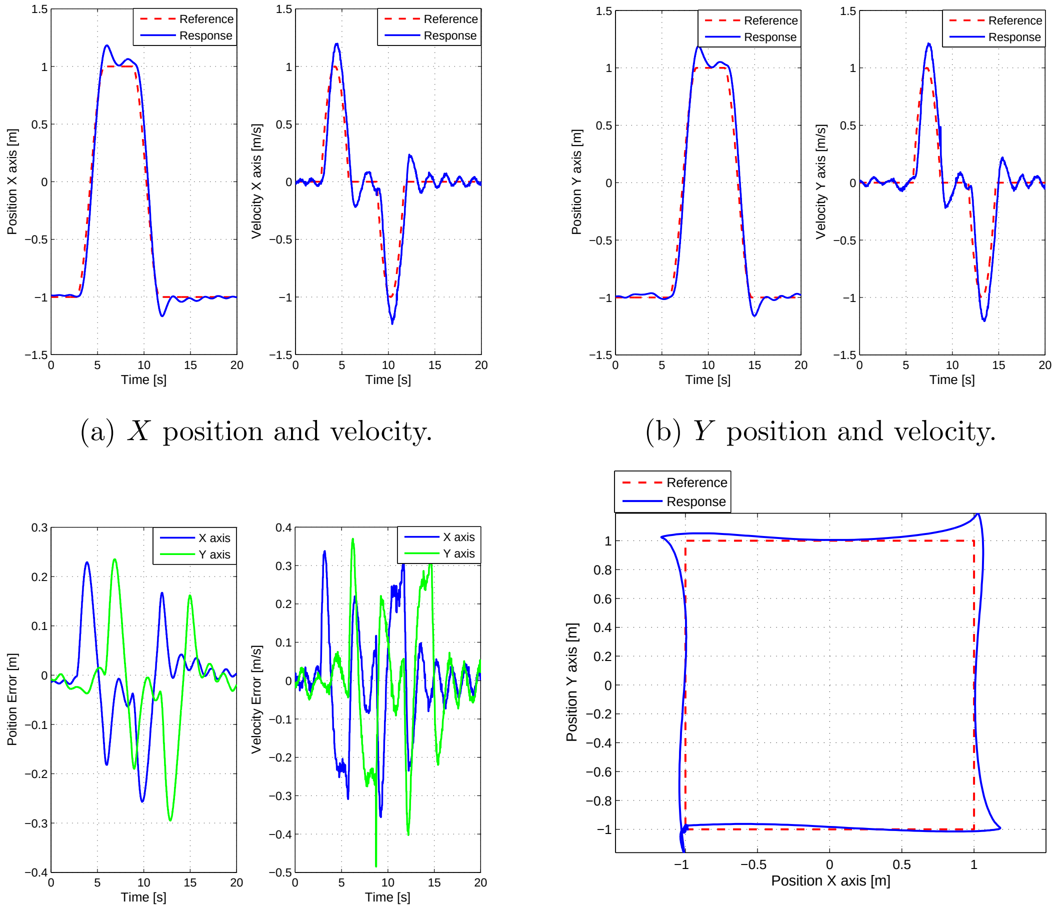
(a) X position.

(b) Y position.

(c) Position error.

(d) 3D plot.

Figure 7.6: Results from tracking a ground robot that is describing an arbitrary trajectory.



(a) X position and velocity.

(b) Y position and velocity.

(c) Position and velocity error.

(d) 2D plot.

Figure 7.7: Results from tracking a square path.

7.2.4 Trajectory Tracking

Based on the modifications presented in Figure 6.5, a trajectory tracking experiment is tested. In this experiment, cubic trajectories have been used to generate a path that is a square of 2 m per side. The whole trajectory lasts 12 s (3 s per side). The results are shown in Figure 7.7 where also the velocity data is included. The results demonstrate an acceptable tracking error since the position error is under 0.3 m and the velocity error is under 0.5 m/s. An overshoot of less than 20% occurs on the corners of the square due to the sudden change in the desired direction of travel.

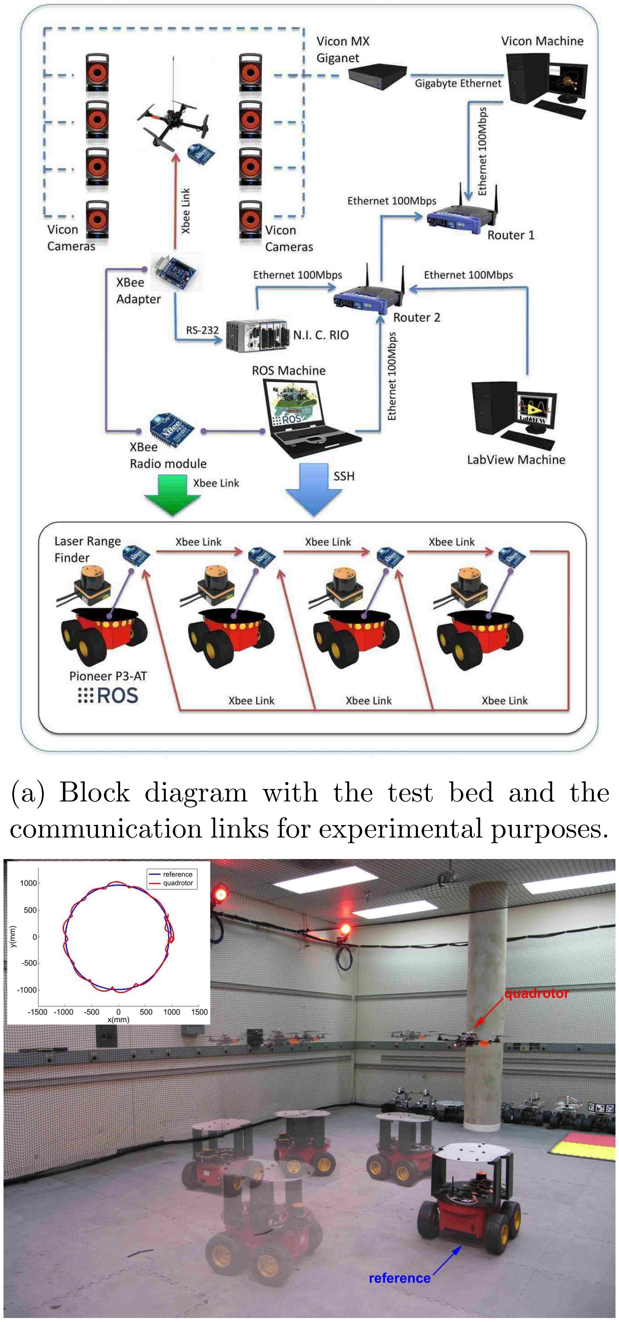
7.3 Practical Applications

The architecture presented in this this work has been used to validate experimentally the results of different research topics. In fact, four applications are briefly described this section. In the first three cases, the experimental verification has been included in their respective publications. In addition, the theory and autonomous implementation of the last application is under development. Videos about the different applications are available at [59].

7.3.1 Mobile Wireless Heterogeneous System

The term *heterogeneous* implies the synergy of multiple robotic platforms characterized by different dynamics and specialized sensing capabilities. In this type of robotic systems, the communication in the network is fundamentally important due to the uncertainties of the wireless channels. Search and rescue missions, disaster relief operations, and surveillance are just a few common examples of scenarios where wireless communications need to be reliable over the robotic network.

In [60], a couple of problems related to communication, centered around two types of platforms: *unmanned ground vehicles* (UGVs), and *unmanned aerial vehicles* (UAVs) are analyzed. Indeed, strategies to enhance the connectivity of a network made of robotic agents, fixed base stations and human interactions are pursued. Furthermore, this article describes an architecture presented in Figure 7.8(a) for controlling a heterogeneous wireless robotic network consisting of aerial and ground vehicles. In addition, the implementation of aerial target tracking by using a Pioneer



(a) Block diagram with the test bed and the communication links for experimental purposes.

(b) A quadrotor tracking a ground vehicle in a circular trajectory.

Figure 7.8: A cooperative heterogeneous mobile wireless mechatronic system.

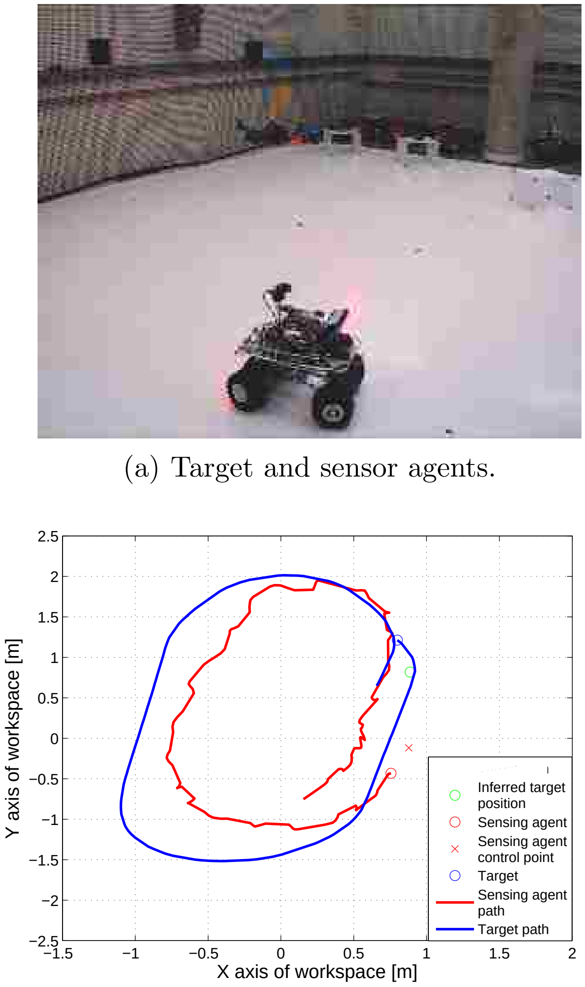
robot as a mobile ground reference is included to validate the embedded hardware controller.

Figure 7.8(b) shows overlapped multiple snapshots of the experiment in which a Pioneer UGV move in a circular pattern and a quadrotor hovers on top of the vehicle following its trajectory. The subfigure within Figure 7.8(b) displays the trajectories of the UGV and the quadrotor. The Pioneer robot was moved with a speed of 0.4 m/s and the experiment shows that the aerial vehicle follows the reference trajectory with a small error (< 0.15 m). For more details please refer to [60].

7.3.2 Surveilling using Mobile Sensor Agents

The paradigm of the moving target surveillance using a network of mobile sensor agents (MSAs) is found in a variety of applications, including the monitoring of urban environments, tracking anomalies in merchandise, manufacturing plants and tracking of endangered species in a wild area. Modern surveillance systems often consist of MSAs deployed to detect and track moving targets in a complex and unstructured environment.

A novel potential function method is developed in [61] for planning the MSA path based on the feedback provided by the target tracking algorithm. A physical experiment demonstrating preliminary results for the simplest scenario and testing various aspects of the theory is conducted at the Marhes Lab. A quadrotor is employed in the experiment as the sensing agent, and a robotic hybrid electric vehicle is used as the target, see Figure 7.9(a). The target vehicle is driven by a human operator. The VICON data is used both in the low level controller for the quadrotor for positioning data for feedback control, and in the artificial simulated sensor for the sensing agent that is used to measure target states. The high-level sensing agent control for conducting the target tracking experiment is written in Matlab and embedded in LabVIEW. The sensing agents tracking behavior of the inferred target position, once the target begins moving, is easily seen in Figure 7.9(b). This preliminary experiments successfully demonstrate the proposed algorithm. For a more detailed explanation, please refer to [61].



(a) Target and sensor agents.

(b) Target and sensor motion.

Figure 7.9: Mobile agents and experimental results for surveilling using MSA.

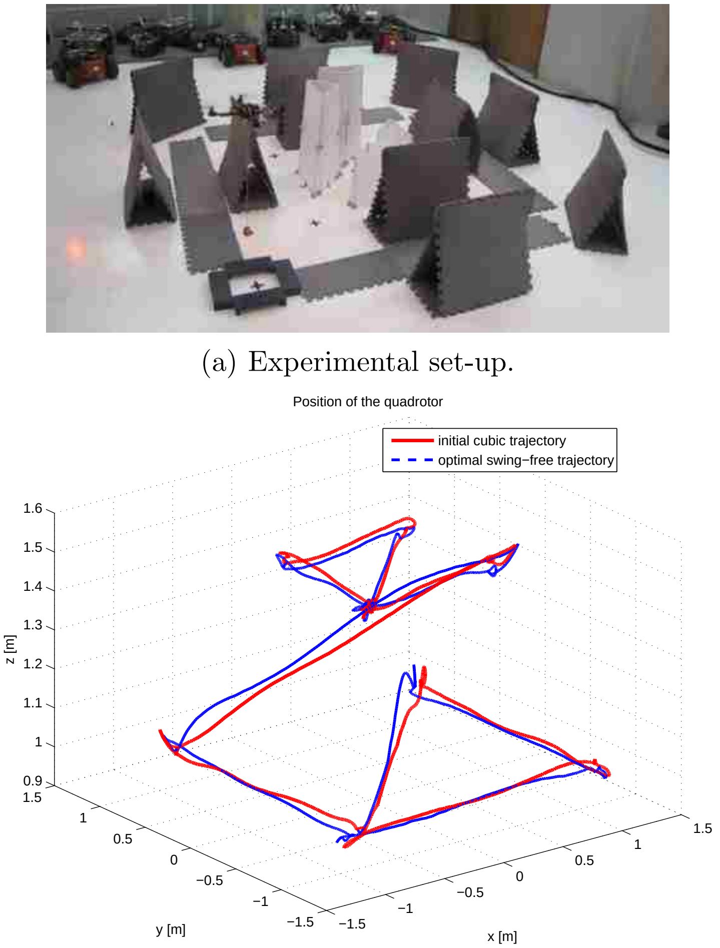
7.3.3 Agile Load Transportation

Aerial transportation and manipulation have a great practical significance. However, flying with a suspended load is a very challenging task since the load significantly changes the flight characteristics of the aerial vehicle. Hence, it is essential that the flight controllers can reduce the effects of the changes in the system dynamics. For that reason, an optimal trajectory generation based on dynamic programing for swing-free maneuvers is introduced in [62] and an extension of its experimental verification is presented in [63].

The experiment emulates a swing-free trajectory tracking in a urban environment. For that reason, a maze of obstacles shown in Figure 7.10(a) is built. A quadrotor is flying above the obstacles while carrying the suspended load through narrow corridors. The mass of the suspended load used is 47 g and the length of the suspension link is 0.62 m. First, the quadrotor tracks an initial 3D trajectory with cubic profile with respect to time. Then, using a dynamic programming based algorithm, a 3D trajectory with an optimal swing-free profile with respect to time is tracked by the aerial vehicle. The quadrotor position is shown in Figure 7.10(b). The spectral analysis of the data shows less energy for the load displacement while tracking the optimal trajectory. Please refer to [62, 63] for details of this approach and its experimental results.

7.3.4 Cooperative Lifting of a Suspended Payload

In this section, the capability of the developed GUI to work with two quadrotors at the same time is demonstrated. Two aerial vehicles lift a suspend load whose weight exceeds the individual payload that each quadrotor can carry. In other words, one quadrotor cannot lift alone the load used in this experiment. The over-weighted payload is a LEGO robot that weights 514 g (the quadrotor's payload is 350 g, see



(a) Experimental set-up.

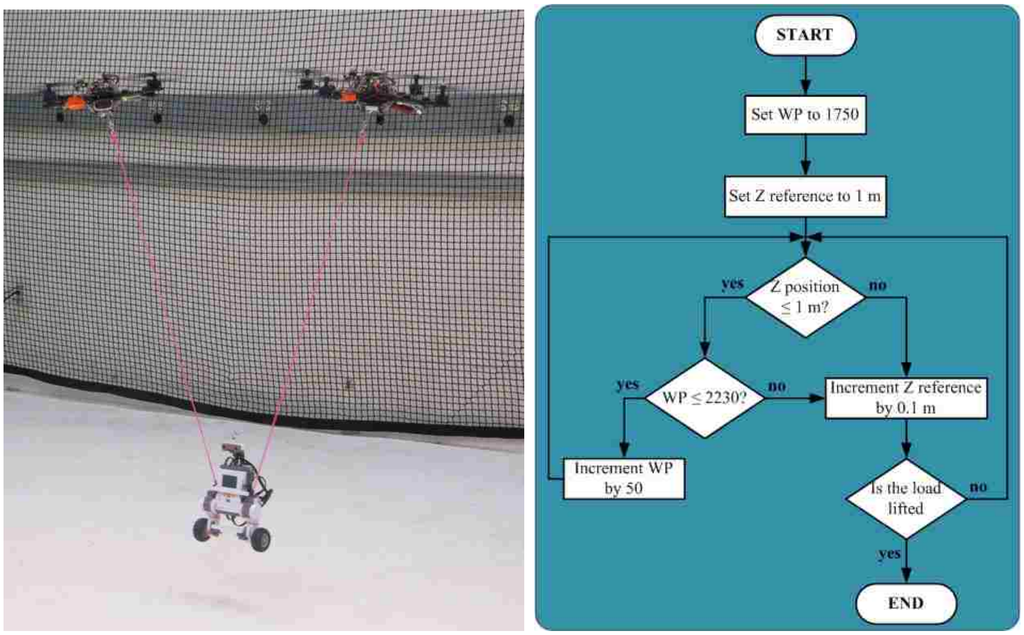
(b) Quadrotor position.

Figure 7.10: Set-up and experimental results for swing-free trajectory tracking.

Section 2.2). Figure 7.11(a) shows a picture of the two quadrotors lifting the LEGO robot and a video can be found in [59].

In this experiment, the maneuver is executed manually while the developed controllers are running. Therefore, following the steps explained at the beginning of this chapter, a single operator can command both quadrotors in way that they can lift the suspended load. Once, both quadrotors are in hovering at a height of 0.7 m using a WP of 1700, the operator follows the steps given in Figure 7.11(b). The value of

2230 for the WP is obtained considering that each quadrotor will lift half of the total mass and then applying Equation (3.5). Using these steps, a single operator needs around 2 min to execute the whole maneuver and finally lift the load. The automation of the maneuver and the theory around this experiment is under development.



(a) Two quadrotors lifting a LEGO robot.

(b) Block diagram of the maneuver.

Figure 7.11: Picture and block diagram of the application about lifting an overweighted payload.

The experiments and applications together with their results have been described and presented in this Chapter. In fact, the real-time control architecture has been part of the experimental section of three journal papers [60, 61, 63] and one conference paper [62]. The conclusions, contributions and improvements of this work are outlined in the next Chapter.

Chapter 8

Conclusions, Contributions and Improvements

8.1 Conclusions

In this thesis, a real-time control architecture for an AscTec Humingbird quadrotor was developed. Moreover, multiple quadrotors can be managed with sligh changes in the proposed architecture. In addition, the control system has demonstrated its flexibility of being easily adaptable to implement different applications. Therefore, this work has provided to the Marhes Lab with a control system to perform experimental tests related with its research in cooperative control in multi-vehicle coordination.

The quadrotor model derived in this work involved a combination between a point-mass submodel derived from rigid-body dynamics and a black-box submodel. The black-box model was obtained by applying system identification and validated employing experimental data. This model configuration has been adopted due to the lack of information about the structure of the internal attitude and thrust controller designed by AscTec.

Using the identified model, a linear position controller was determined by simulation. Moreover, the basic configuration of the linear controller is formed by three lead-lag position compensators, one for each axis, and one PI heading stabilizer. For trajectory tracking purposes, the linear velocity along the X and Y axes is calculated by numerical differentiation and two velocity lead-lag controllers, one for X and one for Y, were added to the basic linear controller configuration.

The linear position controller and its modifications were implemented within the real-time engine that is the hearth of the control architecture. A variety of experimental tests showed that the control system is able to achieve autonomous take off and landing, stable hovering, trajectory tracking and target surveillance. Also, two applications employing simultaneously two quadrotors were performed exhibiting how easily a single operator can execute and supervise the desired maneuver.

8.2 Contributions

The contributions of this thesis are summarized as follows:

Quadrotor Model This thesis presented a hybrid approach on modeling a quadrotor combining a first-principle technique and a system identification approach. Moreover, satisfactory results were obtained utilizing the derived model for control purposes.

Lead-Lag Position Controller Most of the linear position controllers for UAVs described in the literature consider PID compensators. However, lead-lag controllers were used in this project for position and velocity compensation with very good results.

Velocity Estimation For adequate trajectory tracking, the value of the linear velocity needs to be estimated from position data. One available option to accomplish this task is numerical differentiation. Therefore, the Al-Alaoui differentiator was selected and implemented for velocity estimation.

Single Software Platform The whole analysis and implementation were achieved employing a single platform, NI LabVIEW. The NI LabVIEW toolkits for controller design, system identification, simulation and implementation have facilitated the realization of a hierarchical real-time architecture for UAV coordination.

8.3 Limitations and Improvements

The range of the VICON System limits the movement of the UAV's and the capture room is an additional space limitation. Unfortunately, the available space in this room is not enough to safely develop experiments using simultaneously the three Hummingbird quadrotors available at the MARHES Lab.

Another limitation is presented by the internal controller developed by AscTec. This controller behaves as a low pass filter introducing an additional lag to the system and reducing its bandwidth. In applications that require fast trajectories, this lag in the response would deteriorate the performance of the closed-loop system. According to the User's Manual [42], it is possible to disable the internal controller, to read sensor values from the on board instrumentation and to send directly commands to the motor drivers. This means that an inner loop attitude controller can be also designed.

In addition, different linear controllers (e.g., a LQR) or nonlinear controllers based on feedback linearization could be designed and implemented using the current architecture. Besides, the execution in the applications involving two quadrotors were made manually so far. Consequently, autonomous cooperative applications or vehicle coordination algorithms could be investigated and tested under the real-time control architecture presented in this work.

Appendix A

Thrust Measurement Data

The measurement data tables and a summary plot are included next.

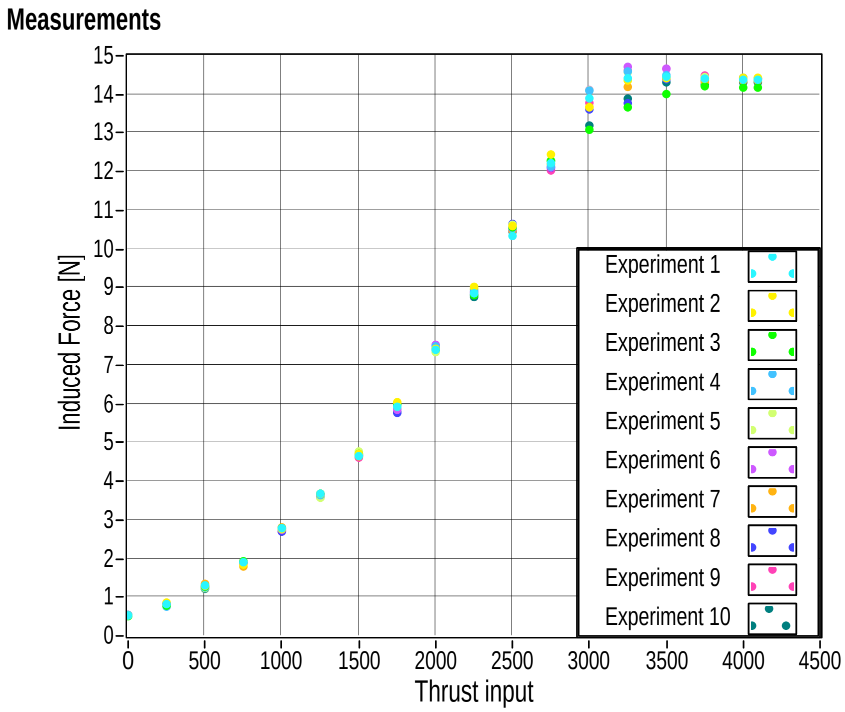


Figure A.1: Thrust measurement data plot for the ten experimental tests.

	Experiment 1			Experiment 2		
Initial Value [kg]:	2.081			2.082		
Thrust Input S_{Th}	m [kg]	Δm [kg]	U [N]	m [kg]	Δm [kg]	U [N]
0	2.027	0.054	0.529	2.028	0.054	0.529
250	1.997	0.084	0.824	1.994	0.088	0.863
500	1.948	0.133	1.305	1.950	0.132	1.295
750	1.887	0.194	1.903	1.894	0.188	1.844
1000	1.797	0.284	2.786	1.801	0.281	2.757
1250	1.706	0.375	3.679	1.708	0.374	3.669
1500	1.607	0.474	4.650	1.602	0.480	4.709
1750	1.477	0.604	5.925	1.466	0.616	6.043
2000	1.327	0.754	7.397	1.325	0.757	7.426
2250	1.178	0.903	8.858	1.162	0.920	9.025
2500	1.027	1.054	10.340	1.002	1.080	10.595
2750	0.835	1.246	12.223	0.813	1.269	12.449
3000	0.664	1.417	13.901	0.689	1.393	13.665
3250	0.611	1.470	14.421	0.618	1.464	14.362
3500	0.607	1.474	14.460	0.612	1.470	14.421
3750	0.612	1.469	14.411	0.614	1.468	14.401
4000	0.615	1.466	14.381	0.611	1.471	14.431
4095	0.615	1.466	14.381	0.611	1.471	14.431

Table A.1: Measurement data experiments 1 and 2.

	Experiment 3			Experiment 4		
Initial Value [kg]:	2.082			2.082		
Thrust Input S_{Th}	m [kg]	Δm [kg]	U [N]	m [kg]	Δm [kg]	U [N]
0	2.030	0.052	0.510	2.027	0.055	0.539
250	2.002	0.080	0.785	2.005	0.077	0.755
500	1.952	0.130	1.275	1.954	0.128	1.256
750	1.885	0.197	1.933	1.889	0.193	1.893
1000	1.798	0.284	2.786	1.798	0.284	2.786
1250	1.707	0.375	3.679	1.711	0.371	3.640
1500	1.602	0.480	4.709	1.604	0.478	4.689
1750	1.471	0.611	5.994	1.475	0.607	5.955
2000	1.327	0.755	7.407	1.319	0.763	7.485
2250	1.184	0.898	8.809	1.173	0.909	8.917
2500	1.005	1.077	10.565	1.007	1.075	10.546
2750	0.830	1.252	12.282	0.843	1.237	12.135
3000	0.748	1.334	13.087	0.645	1.437	14.097
3250	0.689	1.393	13.665	0.594	1.488	14.597
3500	0.654	1.428	14.009	0.604	1.478	14.499
3750	0.633	1.449	14.215	0.614	1.468	14.401
4000	0.637	1.445	14.175	0.611	1.471	14.431
4095	0.637	1.445	14.175	0.614	1.468	14.401

Table A.2: Measurement data experiments 3 and 4.

	Experiment 5			Experiment 6		
Initial Value [kg]:	2.080			2.081		
Thrust Input S_{Th}	m [kg]	Δm [kg]	U [N]	m [kg]	Δm [kg]	U [N]
0	2.026	0.054	0.529	2.027	0.054	0.529
250	2.002	0.078	0.765	1.996	0.085	0.834
500	1.954	0.126	1.236	1.953	0.128	1.256
750	1.888	0.192	1.884	1.887	0.194	1.903
1000	1.798	0.282	2.766	1.801	0.280	2.747
1250	1.716	0.364	3.571	1.707	0.374	3.669
1500	1.594	0.486	4.768	1.604	0.477	4.679
1750	1.476	0.604	5.925	1.485	0.596	5.847
2000	1.332	0.748	7.338	1.314	0.767	7.524
2250	1.166	0.914	8.966	1.178	0.903	8.858
2500	0.996	1.084	10.634	1.007	1.074	10.536
2750	0.827	1.253	12.292	0.846	1.235	12.115
3000	0.642	1.438	14.107	0.642	1.439	14.117
3250	0.594	1.486	14.578	0.581	1.500	14.715
3500	0.606	1.474	14.460	0.586	1.495	14.666
3750	0.607	1.473	14.450	0.611	1.470	14.421
4000	0.611	1.469	14.411	0.614	1.467	14.391
4095	0.611	1.469	14.411	0.614	1.467	14.391

Table A.3: Measurement data experiments 5 and 6.

	Experiment 7			Experiment 8		
Initial Value [kg]:	2.081			2.080		
Thrust Input S_{Th}	m [kg]	Δm [kg]	U [N]	m [kg]	Δm [kg]	U [N]
0	2.027	0.053	0.519	2.027	0.054	0.529
250	1.995	0.085	0.834	1.996	0.085	0.834
500	1.949	0.131	1.285	1.953	0.128	1.256
750	1.887	0.193	1.893	1.887	0.194	1.903
1000	1.805	0.275	2.698	1.801	0.280	2.747
1250	1.706	0.374	3.669	1.707	0.374	3.669
1500	1.605	0.475	4.660	1.604	0.477	4.679
1750	1.493	0.588	5.768	1.485	0.596	5.847
2000	1.322	0.758	7.436	1.314	0.767	7.524
2250	1.181	0.899	8.819	1.178	0.903	8.858
2500	0.994	1.086	10.654	1.007	1.074	10.536
2750	0.846	1.234	12.106	0.846	1.235	12.115
3000	0.692	1.388	13.616	0.642	1.439	14.117
3250	0.676	1.404	13.773	0.581	1.500	14.715
3500	0.613	1.467	14.391	0.586	1.495	14.666
3750	0.616	1.464	14.362	0.611	1.470	14.421
4000	0.609	1.471	14.431	0.614	1.467	14.391
4095	0.611	1.469	14.411	0.614	1.467	14.391

Table A.4: Measurement data experiments 7 and 8.

	Experiment 9			Experiment 10		
Initial Value [kg]:	2.081			2.080		
Thrust Input	m	Δm	U	m	Δm	U
$\mathrm{S_{Th}}$	[kg]	[kg]	[N]	[kg]	[kg]	[N]
0	2.026	0.055	0.539	2.025	0.055	0.539
250	1.996	0.085	0.834	1.995	0.085	0.834
500	1.952	0.129	1.265	1.956	0.124	1.216
750	1.891	0.190	1.864	1.887	0.193	1.893
1000	1.805	0.276	2.708	1.802	0.278	2.727
1250	1.712	0.369	3.620	1.715	0.365	3.581
1500	1.611	0.470	4.611	1.605	0.475	4.660
1750	1.480	0.601	5.896	1.491	0.589	5.778
2000	1.315	0.766	7.514	1.327	0.753	7.387
2250	1.175	0.906	8.888	1.187	0.893	8.760
2500	1.012	1.069	10.487	1.014	1.066	10.457
2750	0.854	1.227	12.037	0.827	1.235	12.292
3000	0.676	1.405	13.783	0.735	1.345	13.194
3250	0.591	1.490	14.617	0.664	1.416	13.891
3500	0.586	1.490	14.666	0.621	1.459	14.313
3750	0.604	1.477	14.489	0.627	1.453	14.254
4000	0.611	1.470	14.421	0.621	1.459	14.313
4095	0.611	1.470	14.421	0.621	1.459	14.313

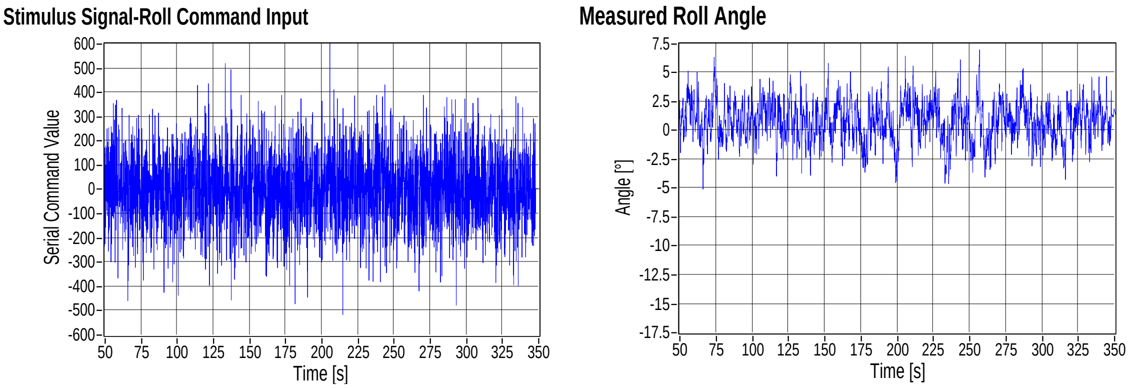
Table A.5: Measurement data experiments 9 and 10.

Appendix B

Data for System Identification

B.1 Stimulus Signals and Acquired Data

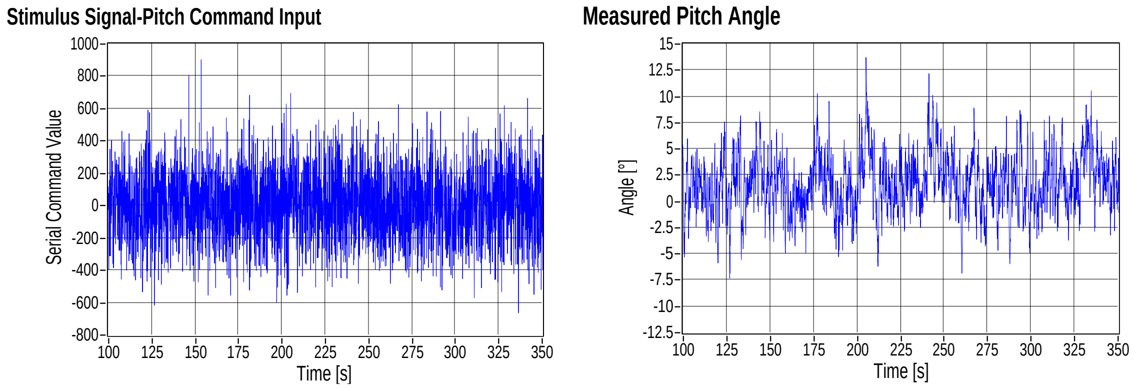
The corresponding plots for each command: roll, pitch and yaw rate are given next.



(a) Stimulus signal

(b) Acquired data

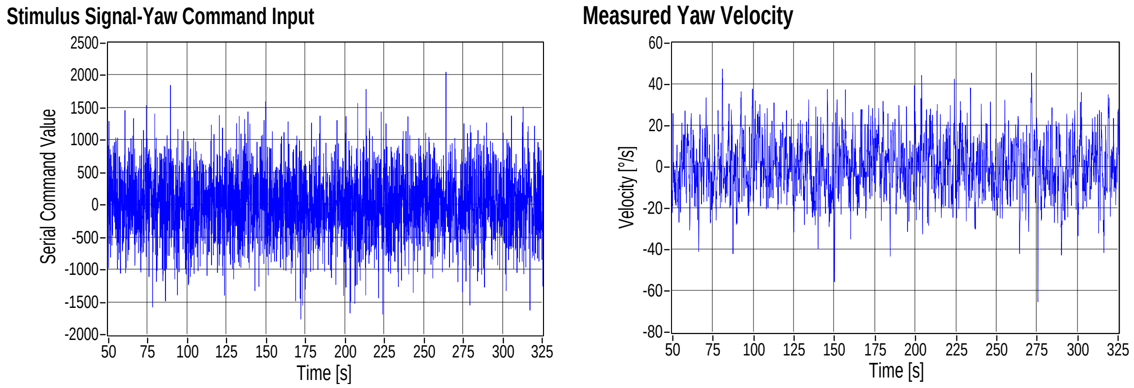
Figure B.1: Stimulus signal and acquired data for the roll angle ϕ_B.



(a) Stimulus signal

(b) Acquired data

Figure B.2: Stimulus signal and acquired data for the pitch angle θ_B.



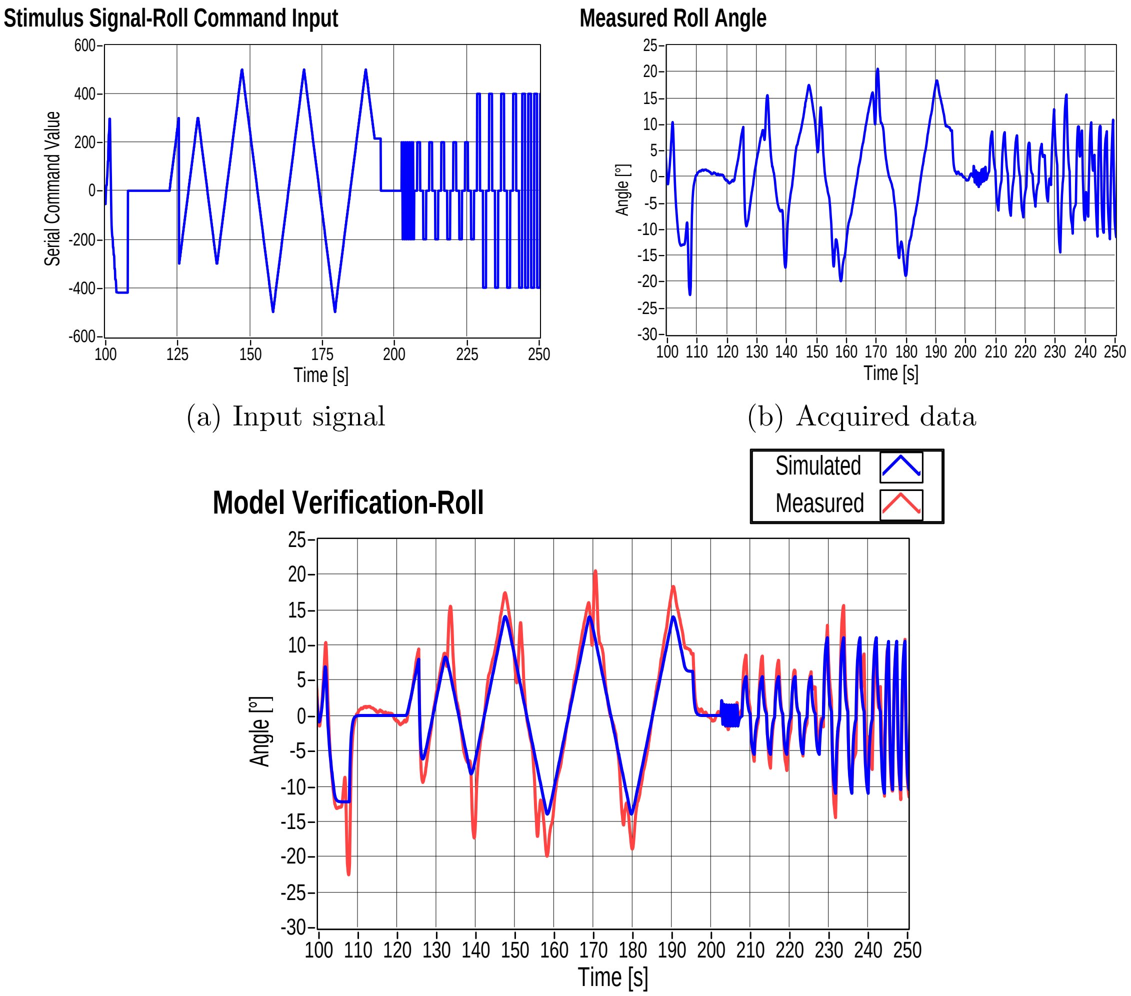
(a) Stimulus signal

(b) Acquired data

Figure B.3: Stimulus signal and acquired data for the yaw rate $\dot{\psi}_B$.

B.2 Model Validation Data

The plots related with the model validation for each case: roll, pitch and yaw rate are presented here.

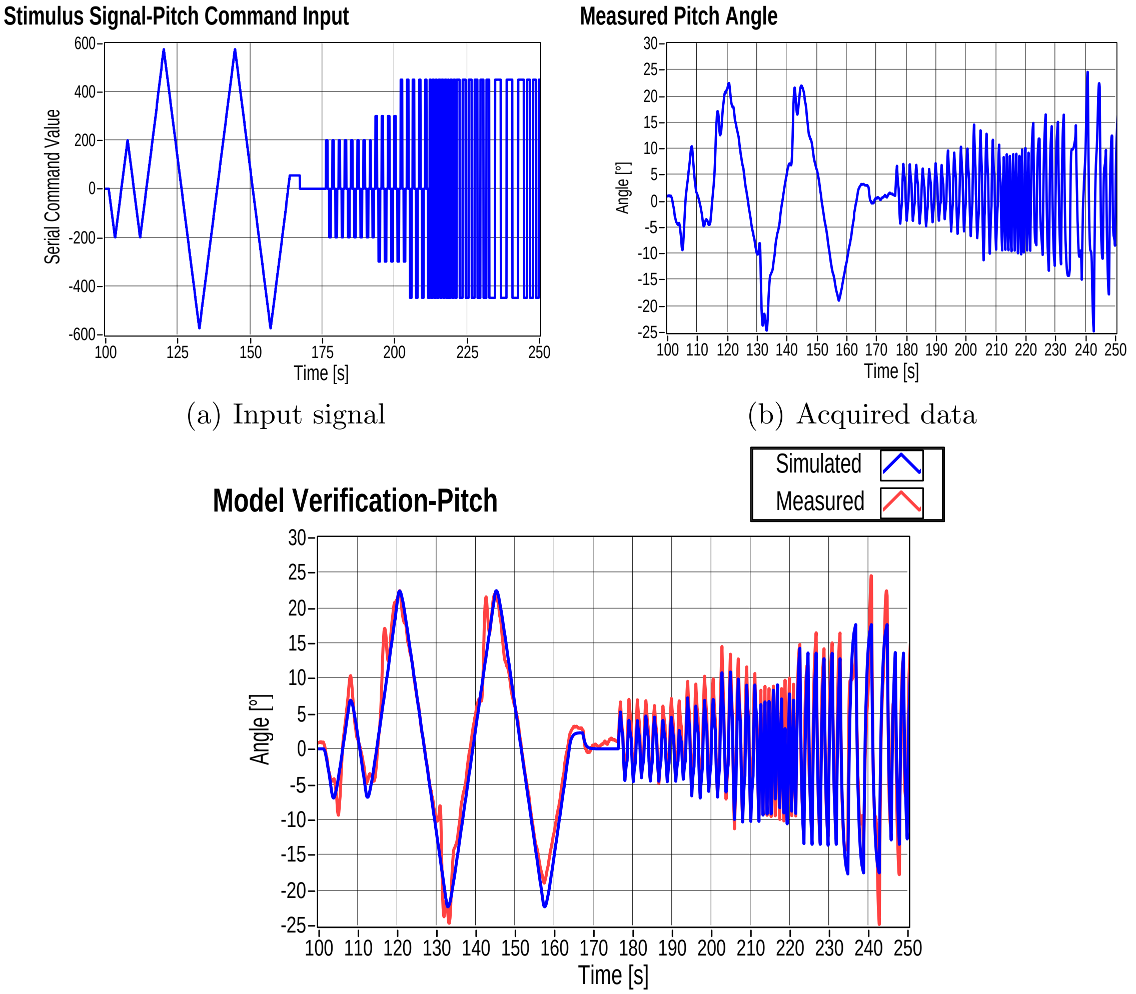


(a) Input signal

(b) Acquired data

(c) Model verification

Figure B.4: Input signal, acquired data and model validation for the roll angle ϕ_B.

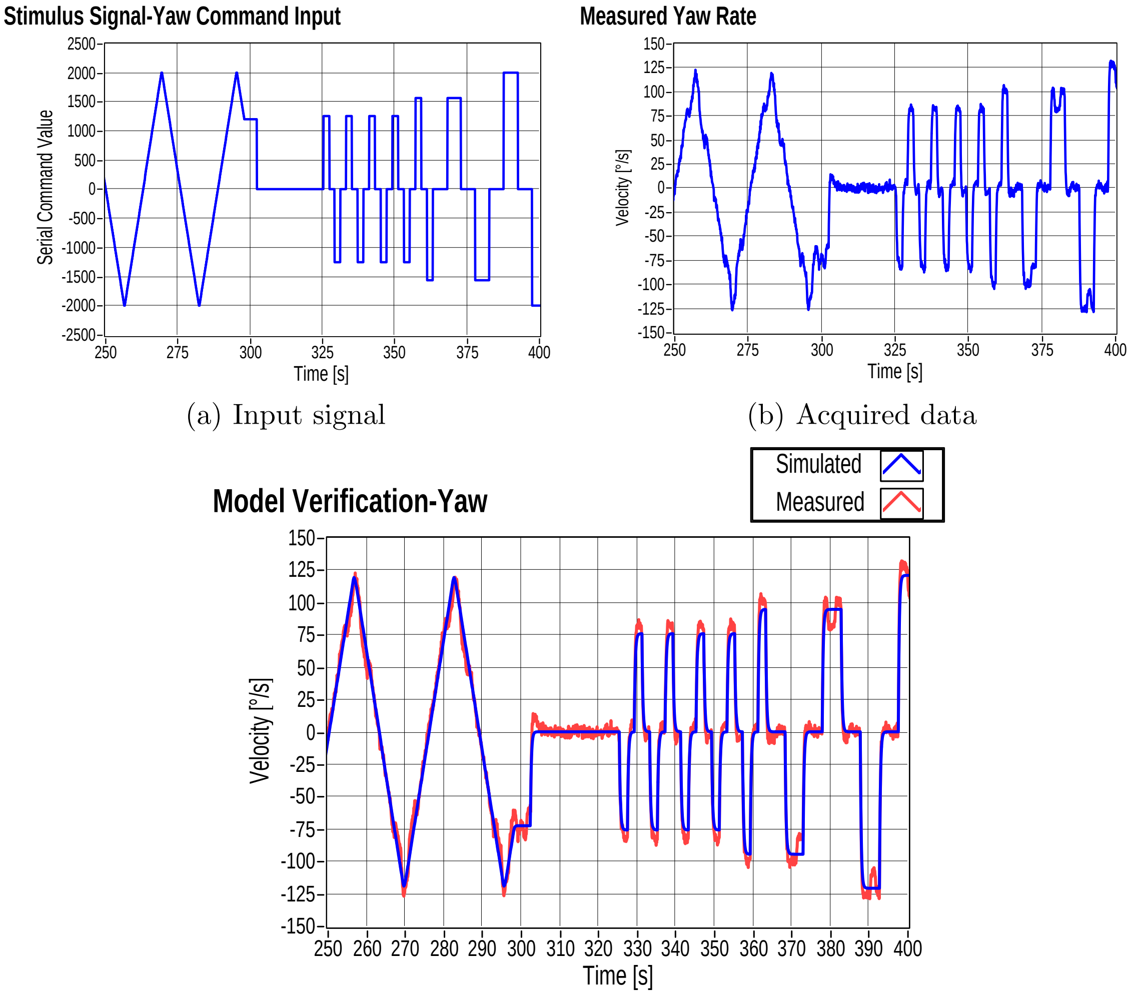


(a) Input signal

(b) Acquired data

(c) Model verification

Figure B.5: Input signal, acquired data and model validation for the pitch angle θ_B.



(a) Input signal

(b) Acquired data

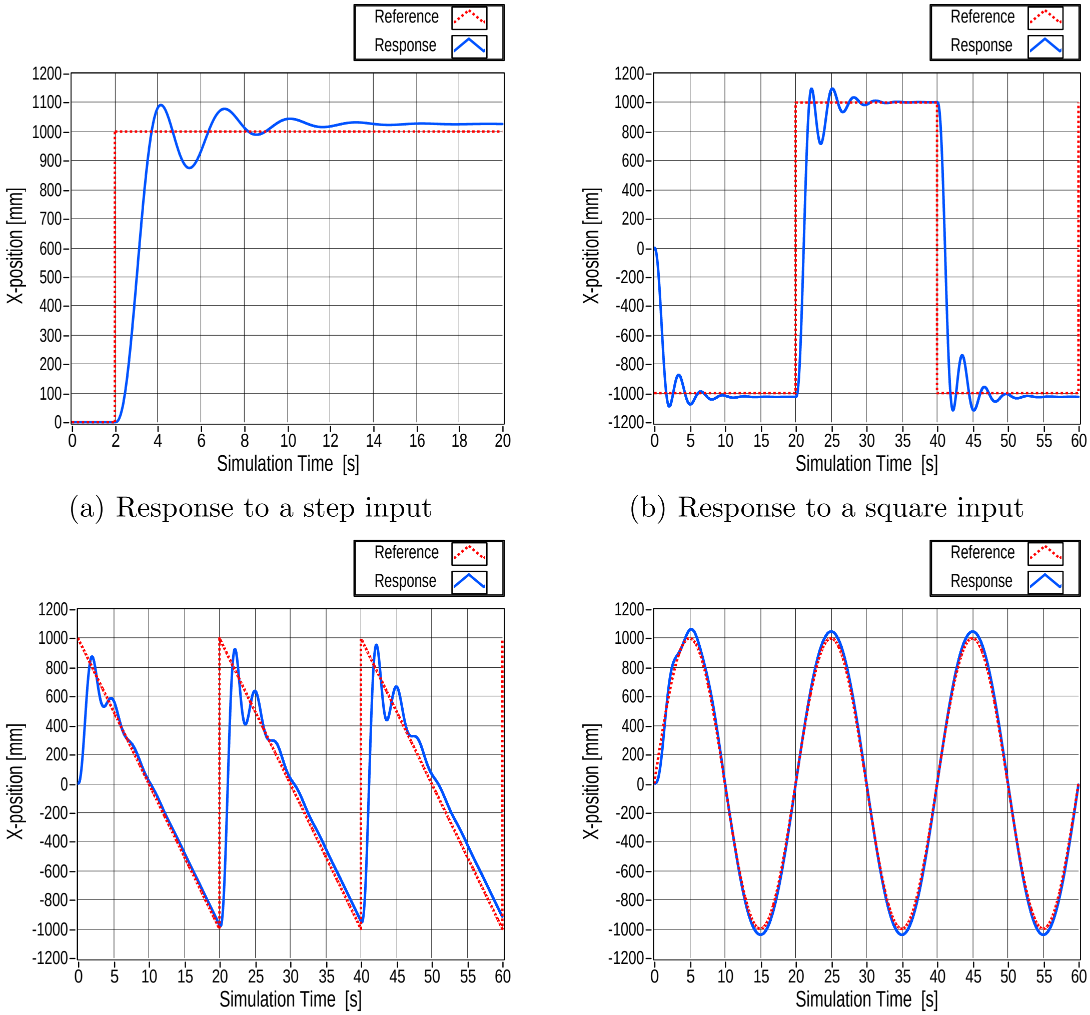
(c) Model verification

Figure B.6: Input signal, acquired data and model validation for the yaw rate $\dot{\psi}_B$.

Appendix C

Simulation Results

The complete dynamical system (3.10) in cascade with the controllers described in Chapter 5 was simulated using the LabVIEW Simulation Module [54]. The results are presented in this appendix.



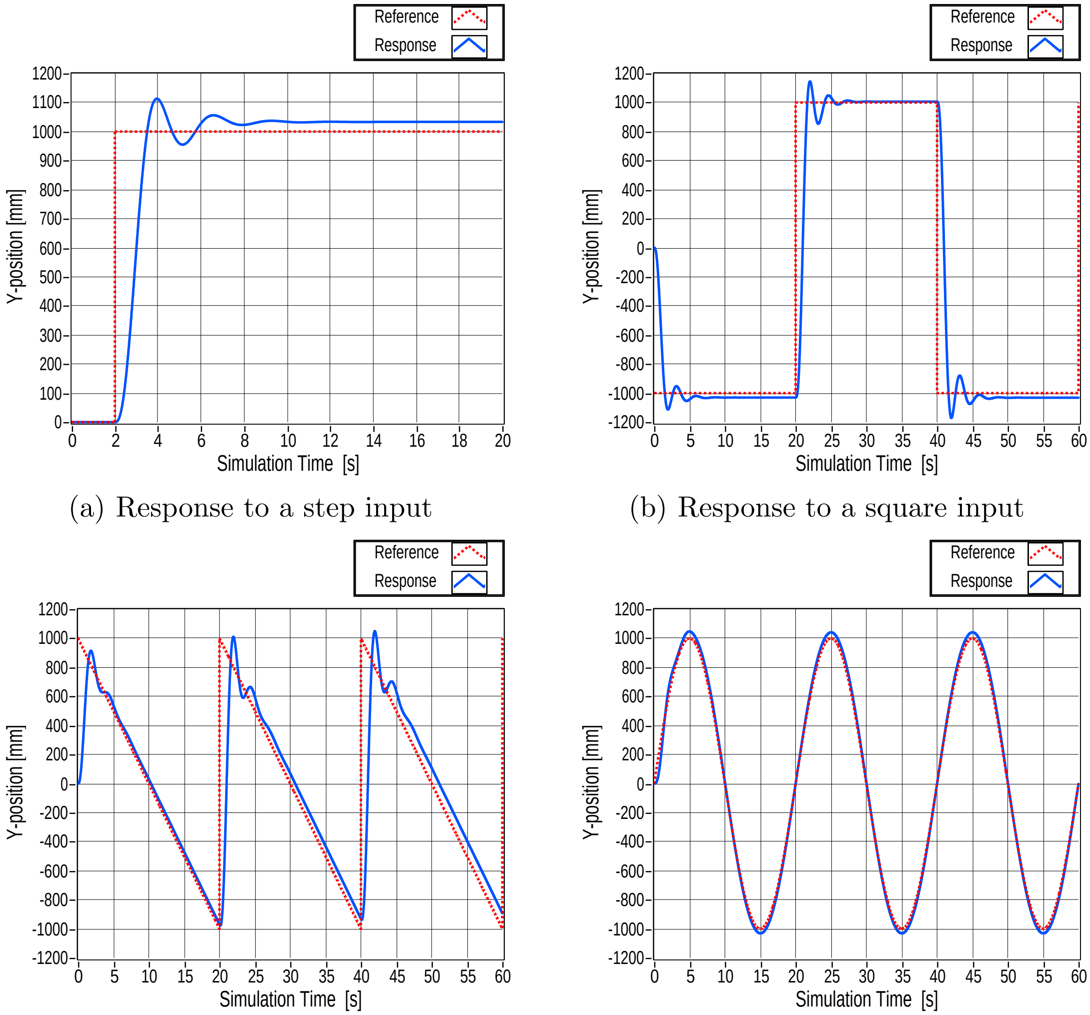
(a) Response to a step input

(b) Response to a square input

(c) Response to a sawtooth input

(d) Response to a sinusoidal input

Figure C.1: Simulation responses for different inputs-X controller.



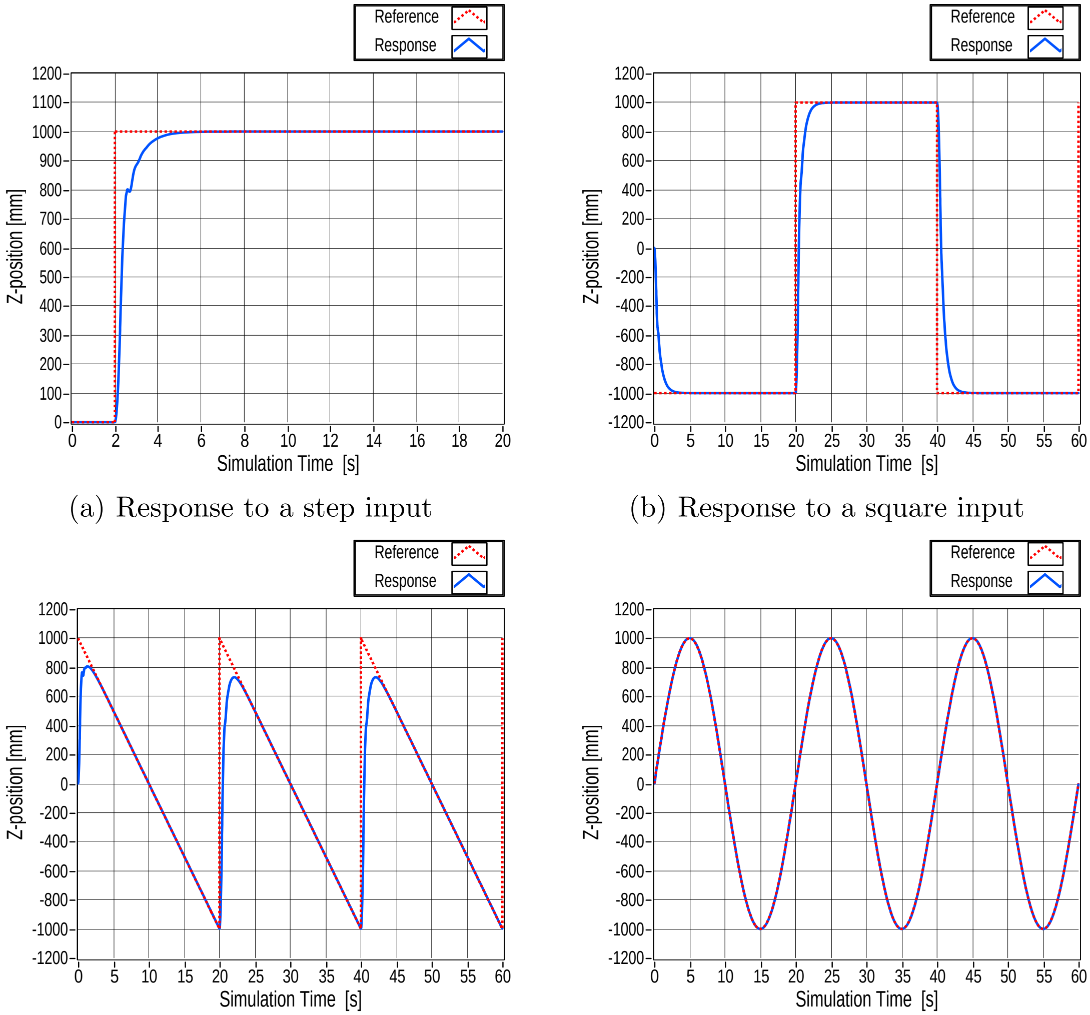
(a) Response to a step input

(b) Response to a square input

(c) Response to a sawtooth input

(d) Response to a sinusoidal input

Figure C.2: Simulation responses for different inputs-Y controller.



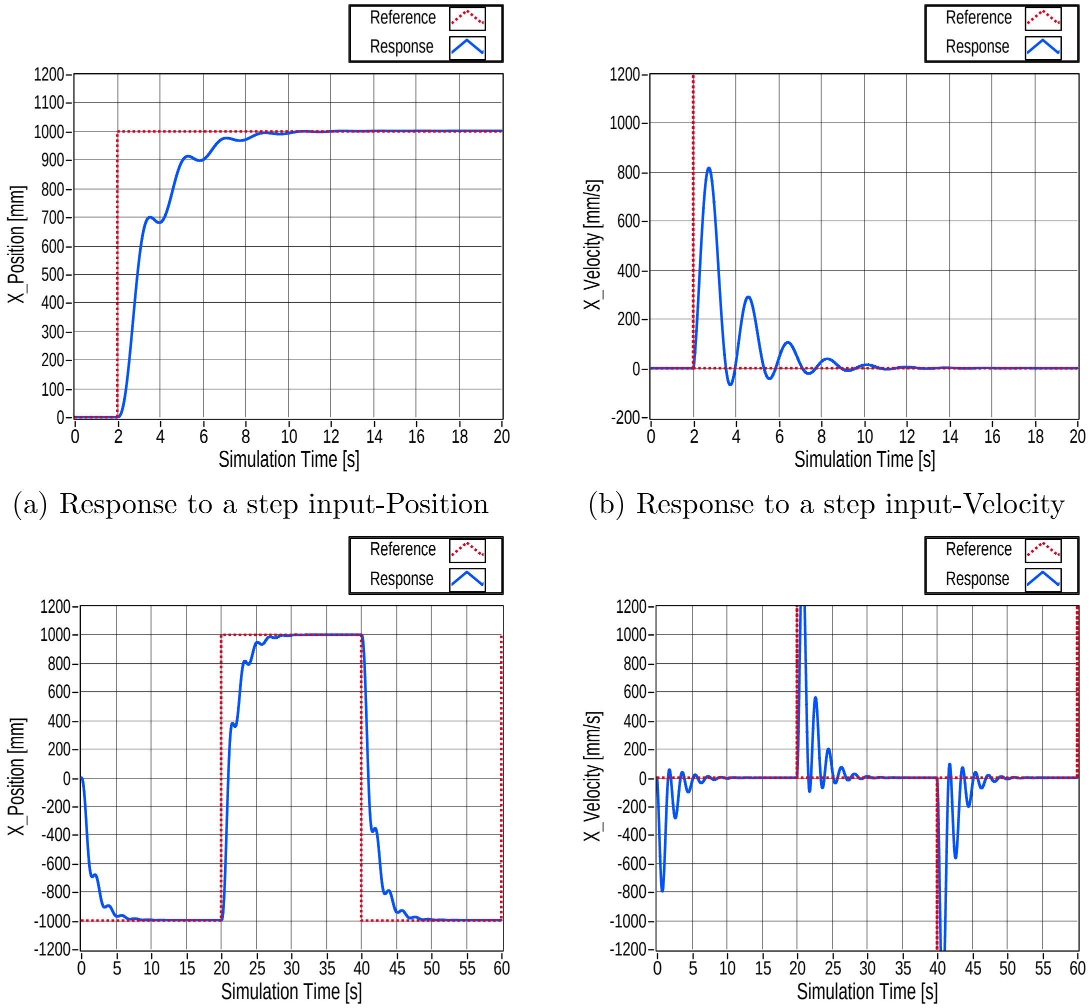
(a) Response to a step input

(b) Response to a square input

(c) Response to a sawtooth input

(d) Response to a sinusoidal input

Figure C.3: Simulation responses for different inputs-Z controller.



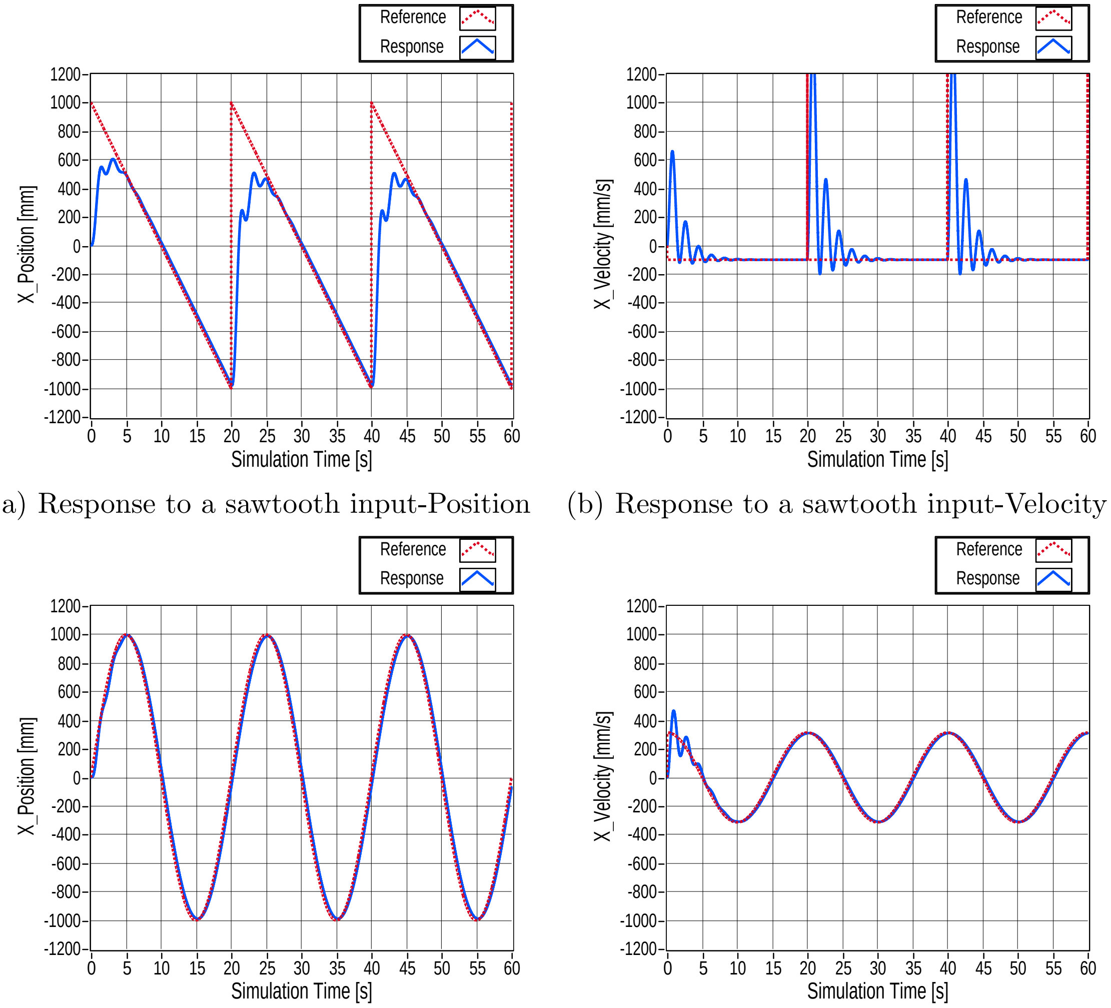
(a) Response to a step input-Position

(b) Response to a step input-Velocity

(c) Response to a square input-Position

(d) Response to a square input-Velocity

Figure C.4: Simulation responses for step and square input-Modified X controller.



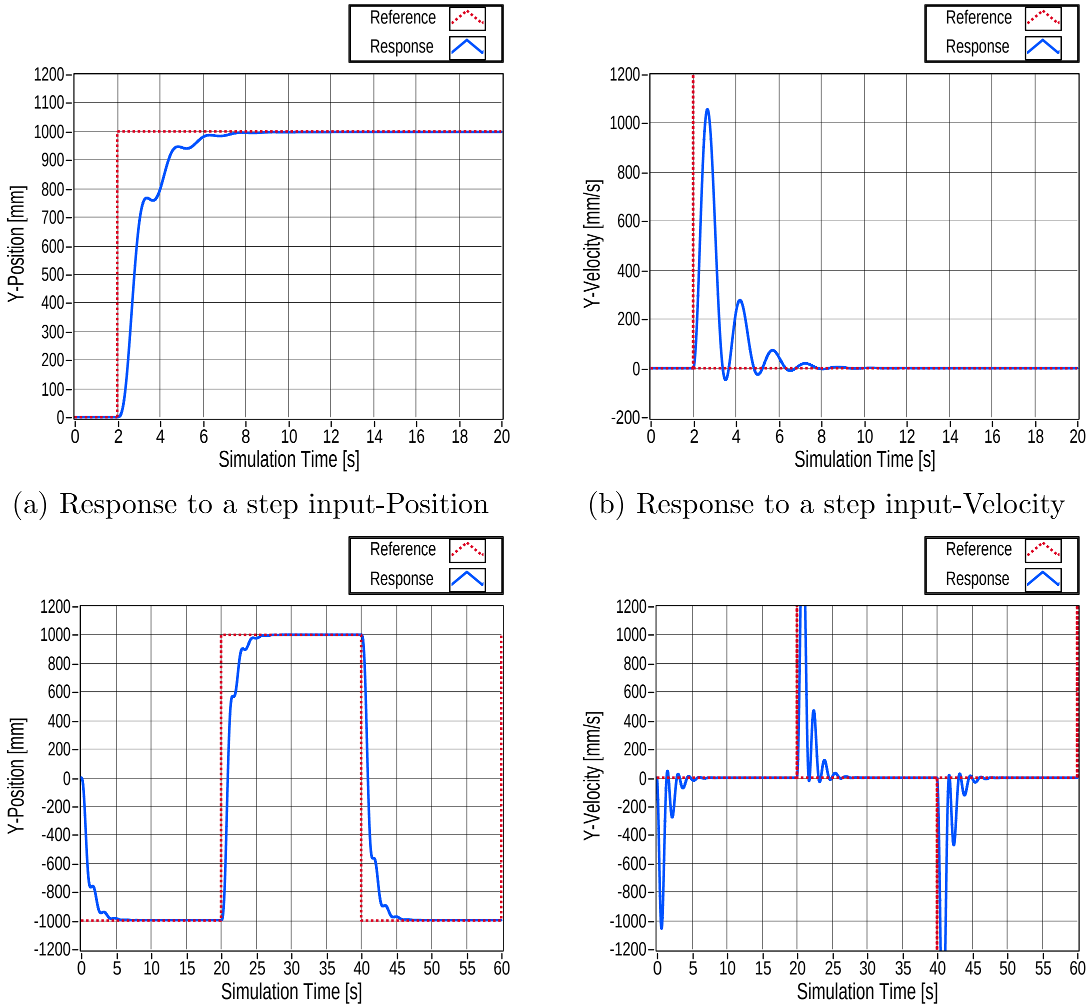
(a) Response to a sawtooth input-Position

(b) Response to a sawtooth input-Velocity

(c) Response to a sinusoidal input-Position

(d) Response to a sinusoidal input-Velocity

Figure C.5: Simulation responses for sawtooth and sinusoidal input-Modified X controller.



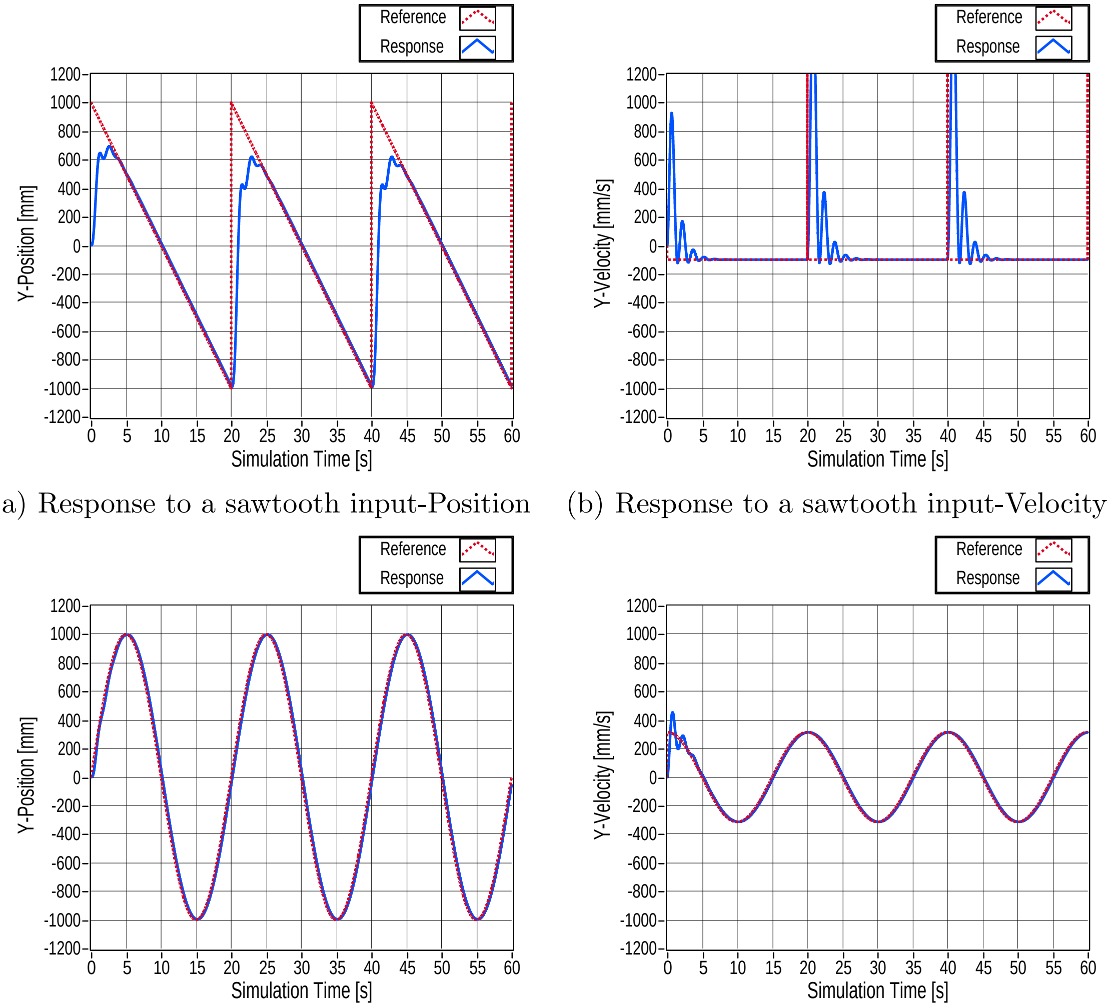
(a) Response to a step input-Position

(b) Response to a step input-Velocity

(c) Response to a square input-Position

(d) Response to a square input-Velocity

Figure C.6: Simulation responses for step and square input-Modified Y controller.



(a) Response to a sawtooth input-Position

(b) Response to a sawtooth input-Velocity

(c) Response to a sinusoidal input-Position

(d) Response to a sinusoidal input-Velocity

Figure C.7: Simulation responses for sawtooth and sinusoidal input-Modified Y controller.

Bibliography

[1] (2011, April) Honeywell T - Hawk aids fukushima daiichi disaster recovery. [Online]. Available: http://honeywell.com/News/Pages/Honeywell-T-Hawk-Aids-Fukushima-Daiichi-Disaster-Recovery.aspx

[2] L. Merino, F. Caballero, J. Martínez de Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent & Robotic Systems*, vol. 65, pp. 533–548, August 2011.

[3] M. Lindermuth, R. Murphy, E. Steimle, W. Armitage, K. Dreger, T. Elliot, M. Hall, D. Kalyadin, J. Kramer, M. Palankar, K. Pratt, and G. C., "Sea robot-assisted inspection," *IEEE Robotics and Automation Magazine*, vol. 18, no. 2, pp. 96–107, June 2011.

[4] F. Kendoul, "Survey of advances in guidance, navigation , and control of unmanned rotorcraft systems," *Journal of Field Robotics*, vol. 29, no. 2, pp. 315–378, March/April 2012.

[5] G. Hoffman, D. Rajnarayan, S. Waslander, D. Dostal, J. Jang, and C. Tomlin, "The stanford testbed of autonomous rotorcraft for multi agent control (STARMAC)," in *The 23rd Digital Avionics Systems Conference*, October 2004, pp. 12.E.4–121–10.

[6] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, April 2008.

[7] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-uav test bed," *IEEE Robotics and Automation Magazine*, vol. 17, no. 3, pp. 56–65, September 2010.

[8] S. Lupashin, A. Schöllig, M. Hehn, and R. D'Andrea, "The flying machine arena as of 2010," in *IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011, pp. 2970–2971.

[9] S. Bouabdallah and R. Siegwart, "Design and control of quadrotors with application to autonomous flying," Thèse sciences, Faculté des sciences et techniques de l'ingénieur STI, Section de microtechnique, Institut d'ingénierie des systèmes I2S (Laboratoire de systèmes autonomes 1 LSA1). Dir.: Roland Siegwart, Ecole polytechnique fédérale de Lausanne, EPFL, Lausanne, 2007.

[10] C. Mary, L. Cristiana, and S. Konge, "Modelling and control of autonomous quad-rotor," 2^{nd} Semester Project of the Intelligent Systems Master Program, Aalborg University, Denmark, June 2010.

[11] G. Hoffman, S. Waslander, and C. Tomlin, "Quadrotor helicopter trajectory tracking control," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Honolulu, Hawaii, April 2008, pp. 1–14.

[12] A. Schöllig, F. Augugliaro, and R. D'Andrea, "A platform for dance performances with multiple quadrocopters," in *IEEE/RSJ International Conference on Intelligent Robots and Systems - Workshop on Robots and Musical Expressions*, Taipei, Taiwan, October 2010, pp. 1–8.

[13] M. Valenti, D. Dale, J. How, and D. Pucci de Farias, "Mission health management for 24/7 persistent surveillance operations," in *AIAA Conference on Guidance, Navigation and Control*, Hilton Head, SC, August 2007, pp. 1–18.

[14] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, April 2012.

[15] S. Lupashin, A. Schöllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, USA, May 2010, pp. 1642–1648.

[16] B. Griffin, R. Fierro, and I. Palunko, "An autonomous communications relay in GPS-denied environments via antenna diversity," *The Journal of Defense Modelling and Simulation: Applications, Methodology, Technology*, vol. 9, no. 1, pp. 33–44, January 2012.

[17] S. Tonetti, M. Hehn, S. Lupashin, and R. D'Andrea, "Distributed control of antenna array with formation of uavs," in *Proceedings of the 18th IFAC World Congress*, vol. 18, no. 1, Milano, Italy, August 2011.

[18] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Autonomous Robots*, vol. 30, no. 1, pp. 73–86, January 2011.

[19] I. Maza, K. Kondak, M. Bernard, and A. Ollero, "Multi-UAV cooperation and control for load transportation and deployment," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1, pp. 417–449, January 2010.

[20] M. Bernard and K. Kondak, "Generic slung load transportation system using small size helicopters," in *IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009, pp. 3258–3264.

[21] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar, "Design, modelling, estimation and control for aerial grasping and manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, USA, September 2011, pp. 2668–2673.

[22] P. Pounds, D. Bersak, and A. Dollar, "Grasping from the air: Hovering capture and load stability," in *IEEE International Conference on Robotics and Automation*, Shangai, China, May 2011, pp. 2491–2498.

[23] A. Albert, S. Trautmann, T. Howard, T. Nguyen, M. Frietsch, and C. Sauter, "Semi-autonomous flying robot for physical interaction with environment," in *IEEE Conference on Robotics Automation and Mechatronics*, Singapore, June 2010, pp. 441–446.

[24] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.

[25] ETH. (2011, December) Flying machine enabled construction. [Online]. Available: http://www.idsc.ethz.ch/Research_DAndrea/fmec

[26] F. Kendoul, Z. Yu, and K. Nonami, "Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles," *Journal of Field Robotics*, vol. 27, no. 3, pp. 311–314, May 2010.

[27] T. Bresciani, "Modelling , identification and control of a quadrotor helicopter," Master Thesis, Department of Automatic Control, Lund University, Sweden, October 2008.

[28] J. Friis, E. Nielsen, R. Foldager, J. Bonding, A. Jochumsen, and A. Friis, "Autonomous landing on a moving platform," Control Engineering, 8th Semester Project, Aalborg University, Denmark, June 2009.

[29] A. Friis, “Autonomous control of miniature quadrotor following fast trajectories,” Control Engineering Master’s Thesis, Aalborg University, Denmark, June 2010.

[30] A. Rüesch, “Dynamic identification & validation, and position control for a quadrotor,” Semester-Thesis, Autonomous Systems Lab, Swiss Federal Institute of Technology, ETH, Zurich, February 2010.

[31] S. Bouabdallah, A. Noth, and R. Siegwart, “Pid vs lq control techniques applied to an indoor micro quadrotor,” in *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, Sendal, Japan, October 2004, pp. 2451 – 2456.

[32] S. Bouabdallah and R. Siegwart, “Backstepping and sliding-mode techniques applied to an indoor micro quadrotor,” in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005, pp. 2247 – 2252.

[33] N. Guenard, T. Hamel, and V. Moreau, “Dynamic modeling and intuitive control strategy for an ”x4-flyer”,” in *International Conference on Control and Automation*, vol. 1, June 2005, pp. 141 – 146.

[34] S. Waslander, G. Hoffmann, J. S. Jang, and C. Tomlin, “Multi-agent quadrotor testbed control design: integral sliding mode vs. reinforcement learning,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, August 2005, pp. 3712 – 3717.

[35] T. Madani and A. Benallegue, “Backstepping sliding mode control applied to a miniature quadrotor flying robot,” in *32nd IEEE Annual Conference on Industrial Electronics*, November 2006, pp. 700 –705.

[36] Mobilerobots. Pioneer 3-AT. ADEPT Technology Inc. [Online]. Available: http://mobilerobots.com/ResearchRobots/P3AT.aspx

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

[38] AscTec Hummingbird Autopilot. Ascending Technologies GmbH. [Online]. Available: http://www.asctec.de/asctec-hummingbird-autopilot-5/

[39] Ar.Drone. Parrot SA. [Online]. Available: http://ardrone.parrot.com/parrot-ar-drone/usa/how-does-it-work

[40] Vicon MX System. VICON. [Online]. Available: http://www.vicon.com/products/viconmx.html

[41] NI CompactRIO. National Instruments Corporation. [Online]. Available: http://www.ni.com/compactrio/

[42] *AscTec Hummingbird with AutoPilot*, User Manual, Ascending Technologies GmbH. [Online]. Available: http://www.asctec.de/assets/Downloads/Manuals/AscTec-Autopilot-Manual-v10.pdf

[43] D. Gurdan, J. Stumpf, M. Achtelik, K. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz," in *IEEE International Conference on Robotics and Automation*, Roma, Italy, April 2007, pp. 361–366.

[44] ZigBee Modules and Mesh Modules. Digi International Inc. [Online]. Available: http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/zigbee-mesh-module/

[45] *VICON DataStream SDK 1.2.0*, Developers Manual, VICON Motion Systems, March 2011.

[46] R. Mahony, V. Kumar, and P. Corke, "Model, estimation and control of quadrotor aerial vehicles," *IEEE Robotics and Automation Magazine*, 2012, *(Accepted)*.

[47] B. Robert, Ed., *The Mechatronics Handbook*, 1st ed. CRC Press, 2002, this reference is published in cooperation with ISA-The Instrumentation, Systems and Automation Society.

[48] *LabVIEW System Identification Toolkit*, User Manual, National Instruments Corporation, 2004. [Online]. Available: ftp://ftp.ni.com/pub/devzone/tut/usermanual.pdf

[49] General polynomial fit VI. National Instruments Corporation. [Online]. Available: http://zone.ni.com/reference/en-XX/help/371361H-01/gmath/general_polynomial_fit/

[50] R. Pintelon and J. Schoukens, *System Identification: A Frequency Domain Approach.* Wiley-IEEE Press, 2001.

[51] R. Jategaonkar, *Flight Vehicle System Identification: a time domain methodology*, ser. Progress in Astronautics and Aeronautics, F. Lu, Ed. American Institute of Aeronautics and Astronautics, Inc., 2006, vol. 216.

[52] *LabVIEW System Identification Toolkit Algorithm References*, User Manual, National Instruments Corporation, June 2008.

[53] L. Ljung, *System Identification, Theory for the User*, 2nd ed. Prentice Hall PTR, 1999.

[54] *LabVIEW Simulation Module*, User Manual, National Instruments Corporation, 2006. [Online]. Available: http://www.ni.com/pdf/manuals/371013c.pdf

[55] M. Al-Alaoui, "Novel digital integrator and differentiator," *Electronics Letters*, vol. 29, no. 4, pp. 376–378, February 1993.

[56] M. Al-Alaoui, "Al-alaoui operator and the new transformation polynomials for discretization of analogue systems," *Electrical Engineering*, vol. 90, pp. 455–467, 2008.

[57] *VICON Data SDK - Getting Started with LabVIEW*, User Manual, VICON Motion Systems, July 2009.

[58] *LabVIEW PID Control Toolkit*, User Manual, National Instruments Corporation, August 2006. [Online]. Available: http://www.ni.com/pdf/manuals/372192a.pdf

[59] Quadrotor's real-time controller using labview. The University of New Mexico - Marhes Laboratory. [Online]. Available: http://marhes.ece.unm.edu/index.php/LabView_Quadrotor

[60] N. Bezzo, B. Griffin, P. Cruz, J. Donahue, R. Fierro, and J. Wood, "A cooperative heterogeneous mobile wireless mechatronic system," in *IEEE/ASME Transactions on Mechatronics Special Issue on Wireless Mechatronics*, 2012 *(Accepted).*

[61] W. Lu, S. Zhang, M. Ferrari, M. Anderson, and R. Fierro, "A particle-filter information method for tracking and monitoring maneuvering targets using a mobile sensor agent," *The Journal of Defense Modelling and Simulation: Applications, Methodology, Technology*, June 2012, published online before print.

[62] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with supended payload: A dynamic programming approach," in *IEEE International Conference on Robotics and Automation*, St. Paul, MN, USA, May 2012, pp. 2691–2697.

[63] I. Palunko, P. Cruz, and R. Fierro, "Towards agile load transportation using aerial robats," *IEEE Robotics and Automation Magazine*, 2012 *(Accepted).*