Development of a Cross Style Quadrotor

Ali Reza Partovi

NATIONAL UNIVERSITY OF SINGAPORE

2012

Development of a Cross Style Quadrotor

Ali Reza Partovi

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2012

In the name of Allah, Most Gracious, Most Merciful

“Guide us to the straight path.”

““and whosoever keepeth his duty to allah, allah will appoint a way out for him, and provide for him from where he does not expect, Allah is sufficient for whosoever puts his trust in him. Indeed, allah brings about whatever he decrees, Allah has set a measure for all things.”

“Holy Quran”

I dedicate this thesis to my parents, brothers, sisters and all teachers, relatives and friends who have helped and accompanied me to explore, learn and contribute.

Acknowledgements

Many thanks to my god, Allah, who has continuously showed me the right way, and facilitated and motivated me to pave the path. First and foremost, I would like to gratefully thank my supervisor Professor Hai Lin for his guidance, encouragement and kindness. Without his supervision, this thesis would not have been accomplished. I highly appreciate Prof. K.Y. Lum for his great guidance and valuable suggestions on my project; Prof. Ben Chen and his UAV group for their consistent support and help and all the NUS professors and lectures who have helped me to develop my academic skills. A very special thanks to Mr. Kevin Ang for his truly great help in this work; And a great appreciation to all my friend in campus life, specifically my labmates Mr. Ali Karimoddini, Mr. Mohsen Zamani, Mr. Mohammad Karimadini, Ms. Sun Yajuan, Dr. Wang Xinhua, Dr.Yang Yang, Ms. Li Xiaoyang, Mr. Liu Xiaomeng, Ms. Xue Zhengui and Mr. Yao Jin. I also thank my housemates Mr. Sadegh Tavalali, Mr. Behroz Chamanbaz, Mr. Ahmad Shahabinia, Mr. Amin Torabi, Mr. Bahador Saket, Mr. Javad Rezaie, Mr. Pravien, and my friends Mr. Hossein Nejati, Ms. Shima Bayat, Mr. Sajad Maghare, Ms. Narjes Alahrabi, Ms. Safoura Sameni, Mr. Mojataba Binazade, Mr. Ghasem Nadaf for all memorable moments we have had together. Last but not least, I present the thesis to my beloved family for their endless love and unwavering support throughout my life.

Contents

Acknowledgements **ii**

Summary **vi**

List of Tables **viii**

List of Figures **ix**

List of Symbols **xiv**

Abbreviations **xx**

1 Introduction **1**

1.1 Background and Motivation 1

1.2 Contribution 3

1.3 Related Works on Quadrotors 5

1.4 Flight Mechanism 8

1.5 Organization of the Thesis 10

2 Hardware and Software Development **12**

2.1 Introduction 12

2.2 Platform Body 12

2.3 Hardware Development 15

2.3.1 Overview of the On-Board System 15

2.3.2 Embedded Computer Module 18

2.3.3 Inertial navigation system Module 20

2.3.4 Servo Control 24

2.3.5 Fail Safe Switch 25

2.3.6 Airspeed Sensor Board 25

2.4 Software Development 28

2.4.1 Firmware 28

2.4.2 Ground Control Station 30

2.5 Conclusion 31

3 Quadrotor Dynamic Modeling **33**

3.1 Introduction 33

3.2 Mathematical model 34

3.3 Model Parameter Identification 38

3.3.1 Static Experiments 39

3.3.2 Dynamic Experiments 49

3.4 Quadrotor Model with Gyro in the Control Loop 66

3.4.1 Gyro : Hover Mode 68

3.4.2 Gyro : Cruise Mode 71

3.5 Model Validation 74

3.6 Conclusion 75

4 Control Design and Implementation **80**

4.1 Simplified Model 81

4.2 Simplified Model with Gyro in the Control Loop 82

4.3 Ground Analysis 84

4.4 Attitude Control 85

4.4.1 Gyro in Hover Mode 86

4.4.2 Gyro in Cruise Mode 89

4.5 Position Hold 94

4.6 Conclusion 98

5 Conclusions **99**

5.1 Conclusions 99

Bibliography **103**

Summary

This thesis presents development of an autonomous quadrotor in a new configuration. In this work, a small scale cross style quadrotor is introduced and compared to a standard plus style type. The new style quadrotor's flight capability and reliability are analyzed through manual flight tests. Through the development of the quadrotor unmanned aerial vehicle (UAV), achievement of autonomous flight is discussed in the following three main steps:

- Hardware and software development: The avionic system is built up based on the NUS (National University of Singapore) UAV group's designed hardware system. However, both its hardware and software are significantly modified to achieve an outdoor quadrotor flight. In a compact and cheap design, the avionic system can provide and record all the required navigation data, communicate with the ground station, drive the motors, and perform real-time control tasks.
- Dynamic model derivation and identification: Through this part, the mathematical model of the quadrotor in cross style is derived and unknown parameters are determined. Static and dynamic identification experiments are conducted to numerically estimate the unknown model parameters. In this thesis, the static approach is addressed as the parameter measurement experiments which are conducted on the ground. Special dynamic flight experiments

are also conducted to collect necessary model identification data. These data are then used in off-line processes to identify the quadrotor dynamic model. Furthermore, to observe the fidelity of identified model, confirmation flight tests are performed to ascertain the results obtained.

- Control design and implementation: This part mainly covers the inner-loop and outer-loop control of the quadrotor. To design a simple realizable controller, the nonlinear dynamic model is first simplified. Proportional integral derivative (PID) controllers are then designed and tuned based on the simplified model. The goal of the controllers is to stabilize the quadrotor's attitude, track the desired attitude, and hold the quadrotor position in a hovering condition. The hardware-in-the-loop simulations, ground tests and real autonomous flight results are given to illustrate performance of the whole system.

List of Tables

2.1 The INS components and their descriptions. 20

2.2 The software framework threads description. 29

3.1 Unknowns model parameters and method of identification in static experiments. 40

3.2 Numerical value of identified parameters from the ground experiments. 44

3.3 Dynamic states' and inputs' description, and measurability status. . . 50

3.4 States and inputs trim values at hover. 53

3.5 Unknowns model parameter and their description for dynamic model identification. 60

3.6 Numerical value of identified parameters from flight data. 64

3.7 The identified parameters of attitude dynamic with gyro in the control loop. 74

4.1 The PID coefficients for attitude tracking and position hold. 96

List of Figures

1.1 Quadrotor in plus and cross styles. 4

1.2 The developed quadrotor in hovering. 6

1.3 Free body diagram of a cross style quadrotor. 8

1.4 Vertical and rolling motion in cross style quadrotor. 9

1.5 Pitching and heading motion in cross style quadrotor. 10

2.1 First developed quadrotor body, mounted on the regulator stand. . . 13

2.2 Left: x650 quadcopter assembled frame. Right: Mounted avionic board on the frame. 14

2.3 Avionic system block diagram. 16

2.4 Left: Assembled avionic board (Top side). Right: Assembled avionic board (Bottom side). 18

2.5 Left: Summit Expansion board. Right: Gumstix Overo Fire. 19

2.6 INS components. 21

2.7 INS communication diagram. 22

2.8 RC receiver hardware modification on Futaba R607FF/R617FF. . . . 23

2.9 Left: micro serial servo controller board. Right: servo controller board input-output connection. 24

2.10 Left: fail safe board. Right: fail safe board input-output connection. . 26

2.11 Airspeed sensor fusion setup and components. 27

2.12 The measured airspeed in compared to wind tunnel air flow reference. 27

2.13 The GUO of ground control station. 31

3.1 Inertia and body coordinate system. 35

3.2 The avionic system hardware configuration and the logging signal points. 39

3.3 Left: rotor thrust measurement stand. Right: measuring the whole platform weight. 41

3.4 Left: J_{zz} measurement experiment setup. Right: J_{xx},J_{yy} measurement experiment setup. 43

3.5 Finding the CG location. Left: Captured picture. Right: Printed intersectional lines. 45

3.6 Rotor-Propeller thrust experiment's collected data. (a) servo input to rotor thrust, (b) square of propeller angular velocity to rotor thrust, (c) servo input to propeller angular velocity, (d) consumption current to rotor thrust. 46

3.7 Rotor rotational velocity coefficient (K_v). 47

3.8 Rotor thrust to square of angular velocity coefficient (K_Ω). 48

3.9 Rotor thrust lift coefficient (K_f). 48

3.10 A typical chirp signal. 51

3.11 Control input data in trimmed flight. 54

3.12 Attitude angle data in trimmed flight. 54

3.13 Attitude angular ratio data in trimmed flight. 55

3.14 Body velocity data in trimmed flight. 55

3.15 Body acceleration data in trimmed flight. 56

3.16 Input signal in the throttle channel perturbation. 58

3.17 Input signal in the throttle channel perturbation. 61

3.18 Euler angles in the throttle channel perturbation. 61

3.19 Inertial acceleration in the throttle channel perturbation. 62

3.20 Input-output data set for rotor thrust dynamic identification. 63

3.21 Rotor thrust dynamic model comparison. 63

3.22 Input signal in the rudder channel perturbation. 65

3.23 Euler angles in the rudder channel perturbation. 65

3.24 Yaw ratio dynamic estimated and measured outputs. 66

3.25 Gyro in the loop control structure. 67

3.26 Gyro in the hover mode, rolling identified model. 70

3.27 Gyro in th hover mode, pitching identified model. 70

3.28 Gyro in the hover mode, yaw angle ratio identified model. 71

3.29 Gyro in the cruise mode, roll angle ratio identified model. 72

3.30 Gyro in the cruise mode, pitch angle ratio identified model. 73

3.31 Gyro in the cruise mode, yaw angle ratio identified model. 73

3.32 Model verification using aileron perturbation. 76

3.33 Model verification using elevator perturbation. 77

3.34 Model verification using rudder perturbation. 78

3.35 Model verification using throttle perturbation. 79

4.1 Control block diagram of the gyro (hover mode) in the loop configuration. 87

4.2 Attitude regulation in outdoor flight (gyro in hover mode). 87

4.3 Attitude tracking in outdoor flight (gyro in hover mode). 88

4.4 Control block diagram of the gyro (cruise mode) in the loop configuration. 89

4.5 Attitude tracking in hardware-in-the-loop simulation (gyro in cruise mode). 91

4.6 Attitude tracking in ground flight test (gyro in cruise mode). 91

4.7 Attitude regulation robustness in ground flight test (gyro in cruise mode). 92

4.8 Attitude regulation in outdoor flight (gyro in cruise mode). 93

4.9 Attitude tracking in outdoor flight (gyro in cruise mode). 93

4.10 Control block diagram of the hierarchical control for position hold. . . 96

4.11 Autonomous position hold. 97

List of Symbols

Alt	altitude in cm
B	barometric pressures in mm of mercury
acx_{b0}	trim value of body frame x axis acceleration
acy_{b0}	trim value of body frame y axis acceleration
acz_{b0}	trim value of body frame z axis acceleration
C_d	body drag coefficient
d	rotor induced torque coefficient
D	body drag
e^x	NED frame x axis
e^y	NED frame y axis
e^z	NED frame z axis
e^{xb}	body frame x axis
e^{yb}	body frame y axis
e^{zb}	body frame z axis
f_i	rotor$_i$ thrust
f_g^I	gravity force in NED frame
f_g^b	gravity force in body frame
F^b	total acting force vector in body frame
GPS_{GAlt}	ground GPS altitude in meter

g	the acceleration of gravity
I	identity matrix
J	moment of inertia matrix
J_{xx}	rolling moment of inertia
J_{yy}	pitching moment of inertia
J_{zz}	yawing moment of inertia
k_{dx}	x position PID controller, differentiator coefficient
k_{dy}	y position PID controller, differentiator coefficient
$k_{d\phi}$	rolling PID controller, differentiator coefficient
$k_{d\theta}$	pitching PID controller, differentiator coefficient
$k_{d\psi}$	heading PID controller, differentiator coefficient
k_{ix}	x position PID controller, integrator coefficient
k_{iy}	y position PID controller, integrator coefficient
$k_{i\phi}$	rolling PID controller, integrator coefficient
$k_{i\theta}$	pitching PID controller, integrator coefficient
$k_{i\psi}$	heading PID controller, integrator coefficient
K_f	rotor thrust lift coefficient
K_v	rotor rotational velocity coefficient
K_Ω	rotor thrust to rotational velocity coefficient
k_{px}	x position PID controller, proportional coefficient
k_{py}	y position PID controller, proportional coefficient
$k_{p\phi}$	rolling PID controller, proportional coefficient

$k_{p\theta}$	pitching PID controller, proportional coefficient
$k_{p\psi}$	heading PID controller, proportional coefficient
K_p	gyro (cruise mode) rolling angle model coefficient
K_q	gyro (cruise mode) pitching angle ratio model coefficient
K_r	gyro (hover cruise mode) heading angle ratio model coefficient
K_ϕ	gyro (hover mode) rolling angle model coefficient
K_θ	gyro (hover mode) pitching angle model coefficient
l	distance from CG to rotors
m	mass of quadrotor
p	body frame pitching angular velocity
p_0	trim value of body frame pitching angular velocity
q	body frame rolling angular velocity
Q_i	rotor$_i$ reactive torque
q_0	trim value of body frame pitching angular velocity
r	body frame yawing angular velocity
r_0	trim value of body frame pitching angular velocity
t	time
T_ϕ	gyro (hover mode) rolling angle model time constant
T_θ	gyro (hover mode) rolling angle model time constant
$TEMP$	centigrade temperatures
u	body frame x axis velocity
u_0	trim value of body frame x axis velocity in hovering

u_a	aileron input
u_{a0}	trim offset for aileron input
u_e	elevator input
u_{e0}	trim offset for elevator input
u_{Ft}	rotors total thrust driven by throttle input
u_r	rudder input
u_{rc}	gyro input vector
u_{r0}	trim offset for rudder input
u_{th}	throttle input
u_{th0}	trim offset for throttle input
u_x	virtual input of position dynamic in x direction
u_y	virtual input of position dynamic in y direction
v	body frame y axis velocity
v_0	trim value of body frame y axis velocity in hovering
v^b	velocity vector in body frame
v^i	rotor$_i$ input voltage
w	body frame z axis velocity
w_0	trim value of body frame z axis velocity in hovering
x	NED frame x-axis position
x_{ref}	position reference in x direction
y_{ref}	position reference in y direction
y	NED frame y-axis position
z	NED frame z-axis position

ϕ	rolling angle in NED frame
ϕ_{ref}	rolling reference angle
ϕ_0	trim value of rolling angle velocity in hovering
θ	pitching angle in NED frame
θ_{ref}	pitching reference angle
θ_0	trim value of pitching angle velocity in hovering
ψ	yawing angle in NED frame
ψ_{ref}	heading reference angle
ψ_0	trim value of yawing angle velocity in hovering
η	euler angle vector
ω^b	angular velocity vector in body frame
ξ	position vector in NED frame
Ω_i	rotor$_i$ rotational velocity
τ_f	rotor thrust model time constant
τ^b	total acting force vector in body frame
τ_p	gyro (cruise mode) rolling angle ratio model time constant
τ_q	gyro (cruise mode) pitching angle ratio model time constant
τ_r	gyro (cruise mode) heading angle ratio model time constant
$\tau_{\phi 1}$	gyro (hover mode) rolling angle model time constant 1
$\tau_{\phi 2}$	gyro (hover mode) rolling angle model time constant 2
$\tau_{\theta 1}$	gyro (hover mode) pitching angle model time constant 1

$\tau_{\theta 2}$	gyro (hover mode) pitching angle model time constant 2
ρ	air density
$\mu_1(.)$	mean of given vector
$\mu_2(.)$	standard derivation of given vector

Abbreviations

CG	center of gravity
DCM	direction cosine matrix
DOF	degree of freedom
DGPS	differential global positioning system
DSP	digital signal processing
ESC	electronic speed controller
GPS	global positioning system
GUI	graphic user interface
IDENT	time-domain identification MATLAB toolkit
IMU	inertia measurement unit
INS	inertial navigation system
LQR	linear quadratic regulator
NUS	National University of Singapore
NED	North-East-Down
MEMS	micro electro-mechanical system
PEM	predication error method
PID	proportionalintegralderivative (controller)

PPM	pulse position modulation
PWM	pulse width modulation
RC	radio control
RF	radio frquency
RTOS	real time operating system
STARMAC	Stanford testbed of autonomous rotorcraft for multi-agent control
TTL	transistor-transistor logic
UAV	unmanned aerial vehicle
USART	universal synchronous/asynchronous receiver/transmitter
VTOL	vertical take-off and landing
Wi-Fi	wireless fidelity

Chapter 1

Introduction

1.1 Background and Motivation

In the last few decades, the role of autonomous robotics in human life is getting more pivotal. In many situations, humans can be replaced by autonomous robots to deal with tasks in a more efficient and safer way. Among them, aerial robots with their ability to move easily in 3-dimensional space are potentially more viable in many applications where the robot's maneuverability is crucial. Autonomous aerial vehicles exhibit great potential to play in roles such as : data and image acquisition [1], localization of targets [2], map building [3], search and rescue [4], multi vehicle cooperation systems [5] and others.

Nowadays, with the significant advancement of micro electro-mechanical system (MEMS) sensors and microprocessors, higher energy density Lithium Polymer batteries, and more efficient and compact actuators, thus resulting in the rapid development of unmanned aerial vehicles. The vertical take off and landing (VTOL) crafts due their capability for flying in many different flight missions have obtained more attention in this trend.

The helicopter as a VTOL aircraft, is able to take off in a confined area, hover on the spot, perform horizontal flight movements, and land vertically. However, besides these features, traditional helicopters have a complex architecture. The conventional helicopters require a tail rotor to cancel the main rotor's reactive torque. They also typically need a large propeller and main rotor. Moreover, their flight control mechanism is relatively complicated. Other than the mechanical complexity of the main rotor cycle pitch mechanism, the helicopters up and downwards motion control requires the main rotor to maintain rotational speed while changing the pitch angle of the rotor blades which needs a special mechanical configuration setup [6].

The quadrotors as compared to conventional rotorcrafts have a simpler mechanical structure and flight mechanism. Using four rotors in a symmetrical configuration removes the need of a tail rotor. In fact, since each pair of rotors rotate in opposite directions, the generated reactive torque is inherently canceled. The quadrotor's fixed pitch blades reduces the platform's mechanical complexity, and the relatively small propeller could decrease mechanical vibration. From the maneuverability point of view, quadrotors are generally able to offer better performance over traditional helicopters. Quadrotors have a symmetrical rigid structure which can be considered as an omnidirectional vehicle. They can be configured such that right, left, front and back would have a relative direction. In particular, this means it can potentially fly in any direction without maintaining its head towards the desired direction [7]. Furthermore, quadrotors have a relatively simple flight control mechanism which is only based on individual propellers' rotational speed.

However, quadrotors as with most VTOL aircraft have low performance in aspects of forward flight speed, range, and endurance. Although new platforms have been proposed to increase the performance of VTOL aircraft and in specific quadrotors in horizontal flight [8], [9], through a simple modification on its orientation control, the improvement to the quadrotor's maneuverability in horizontal displacement can be expected. In fact, the standard quadrotor is constructed from four propeller-rotor sets in a plus style and flies in horizontal plane by changing its attitude. The desire attitude is obtained by differing the rotors speed of different pairs of rotors. Hence, only two rotors are involved in horizontal movement which basically because the defined body frame axes lied on the quadrotor arms. By simply considering the quadrotor in a cross style comparing to the body frame (see Figure 1.1), it is able to take advantage of using all four rotors in achieving horizontal displacement. In such a configuration, all rotors participate together to rotate the platform around the desired axis of orientation. Thus for a quadrotor in the cross style when compared to the standard quadrotor, for the same desired motion, provides higher momentum which can increase the maneuverability performance [10].

Moreover, this configuration is more suitable for mounting camera to capture the platform front video.

1.2 Contribution

In the followings, we will present the development of an autonomous cross styled quadrotor. In this work, our contributions in development of this platform are presented in the aspects of hardware and software development, mathematical dynamic

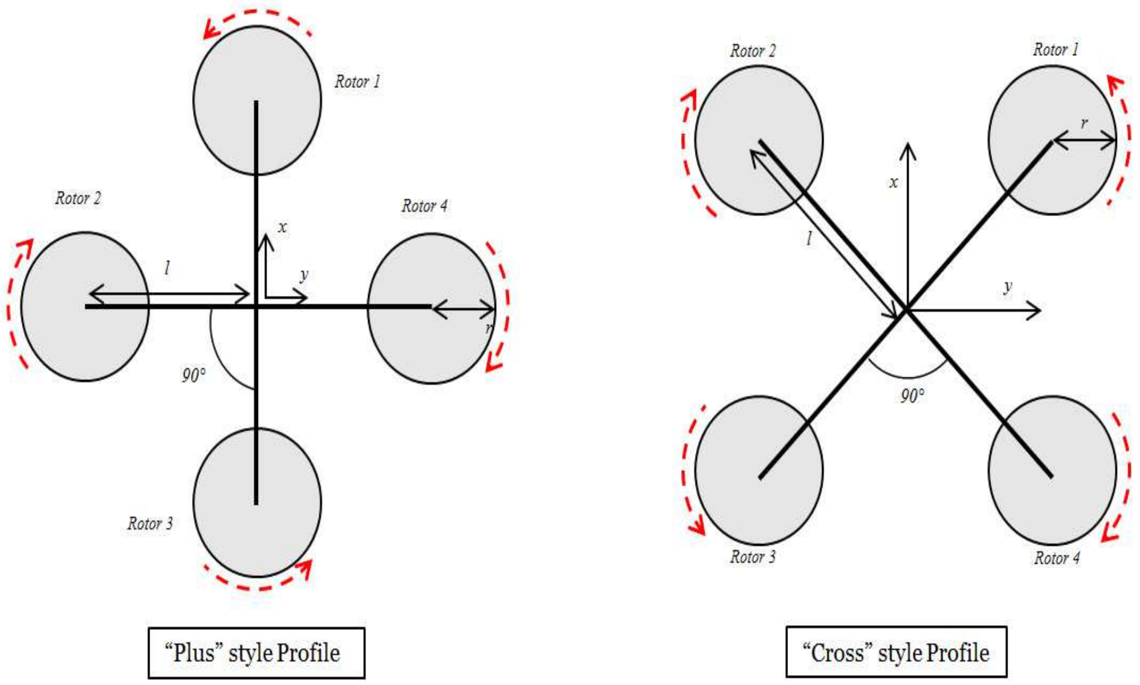


Figure 1.1: Quadrotor in plus and cross styles.

modeling, model identification, and autonomous hovering control. The platform in action is shown in Figure 1.2.

The onboard hardware and software systems are actually built up based on the avionic system of NUS UAV group. For this quadrotor, the hardware design is modified for outdoor flight which is discussed in Chapter 2. A global positioning system (GPS) and a pressure sensor are added to the avionic system; the navigation unit firmware has been revised to support these components and also perform the preliminary data filtering and integration. Furthermore, to have an airplane compatible avionic system, an airspeed measurement unit is designed and integrated to the onboard system.

The software of the avionic board is also significantly modified. The modular structure of the original avionic software provides this option to modify only the necessary parts without changing the whole program. For this project, the sensor fusion and the data logging parts are revised to support the new sensors data. A dead reckoning algorithm also is developed to provide smooth and realtime global position data. Moreover, in the control module, the hardware-in-the-loop simulation and real time control functions are written for the developed quadrotor.

The cross style quadrotor as compared to the standard plus type has slightly different mathematical model which is derived in Chapter 3. We also consider the quadrotor model when the stabilization gyro module is in the control loop. The unknown parameters of these models are identified through the static and dynamic experiments. The identified dynamic models are validated based on the recorded real flight data to be used for further control and simulation tasks.

For the control part, we designed PID controllers for attitude stabilization, attitude tracking and, position hold scenarios. With the identified dynamic model, we proposed a systematic procedure to evaluate the controllers with hardware-in-the-loop simulations and ground tests. Finally, for each scenario, a real autonomous flight test is conducted to illustrate the whole system performance.

1.3 Related Works on Quadrotors

Many groups have worked on standard quadrotors and realized various controllers and scenarios based on these platforms. Among these works in this area, we wish to comment on the following papers. The Stanford Testbed of Autonomous Rotorcraft

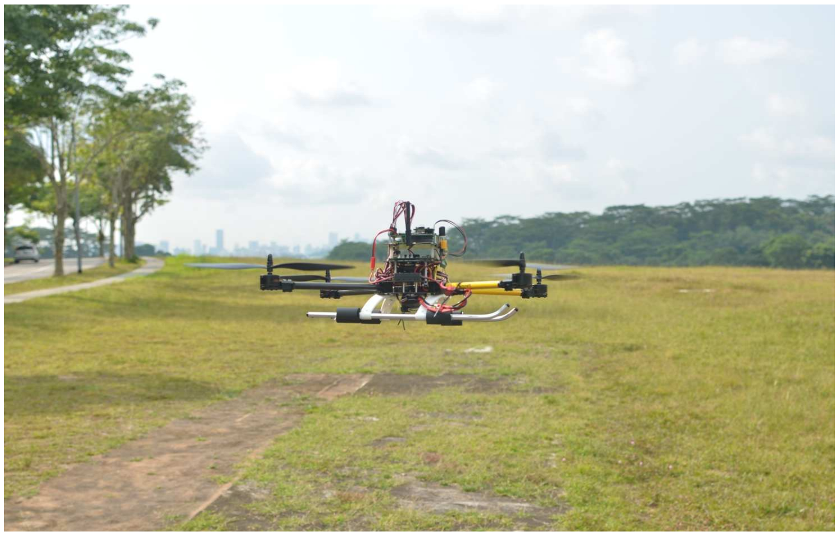


Figure 1.2: The developed quadrotor in hovering.

for Multi-Agent Control (STARMAC) is one of the more successful outdoor platform from Stanford University [11]. They build up their quadrotor based on the Dragonflyer III platform and developed their own avionic system. One of the first progress of this project is presented in [11]. They took the advantages of the sliding mode controller to stabilize the height and linear quadratic regulator (LQR) method to control the attitude. An extensive dynamic modeling including aerodynamic analysis was done on STARMAC [12]. By using differential global positioning system (DGPS), the STARMAC has been succeeded to perform autonomous hovering flight with a duration of up to two minutes and within a 3 meters circle [13]. In addition, an outdoor autonomous trajectory tracking and platform flapping were realized based on the STARMAC quadrotor [13], [14].

A research group in Australian National University developed a large scale quadrotor. In the preliminary study, platform design and fabrication and hardware development of the first design, MARK I, is described in [15]. The issue of insufficient thrust margin and unstable dynamic behavior led them to design the second platform MARK II [1]. The complete dynamic modeling and aerodynamic analysis is presented in [16]. Through this work, a discrete PID controller is used to realize the attitude control.

Another outdoor flying quadrotor was introduced in [17]. This group developed an advanced powerful avionic system based on the gumstix processor and cross bow inertia measurement unit (IMU). A nonlinear hierarchical flight controller and a complete stability analysis is presented in this work. Based on the proposed controller, they achieved autonomous take off and landing, and an outstanding attitude and path tracking performance.

The indoor quadrotor also attracted many groups. A truly successful indoor quadrotor was built in MIT university [18]. The goal of this project is to realize a single or multi-vehicle health management system. The development of advanced equipped quadrotor for detection of target observation under indoor calamity environment was discussed in [19]. More advanced controllers were also implemented on this type of helicopter. A robust adaptive and back stepping control is proposed and validated experimentally on the quadrotor in [20], [21].

However, the addressed quadrotor have mainly focused on standard plus style quadrotor.

1.4 Flight Mechanism

The quadrotor proposed in this work is developed in a cross style which has a different flight mechanism as compared to standard quadrotors (see Figure 1.3). As with standard quadrotors, the basic motion of the quadrotor is realized by adjusting individual propeller's speed. The propellers have a fixed-pitch and their air flows point downwards to produce an upward lift. On each end of the horizontal arms, a rotating motor is placed. The whole quadrotor setup consists of two clockwise rotating motors and two counter-clockwise rotating motors.

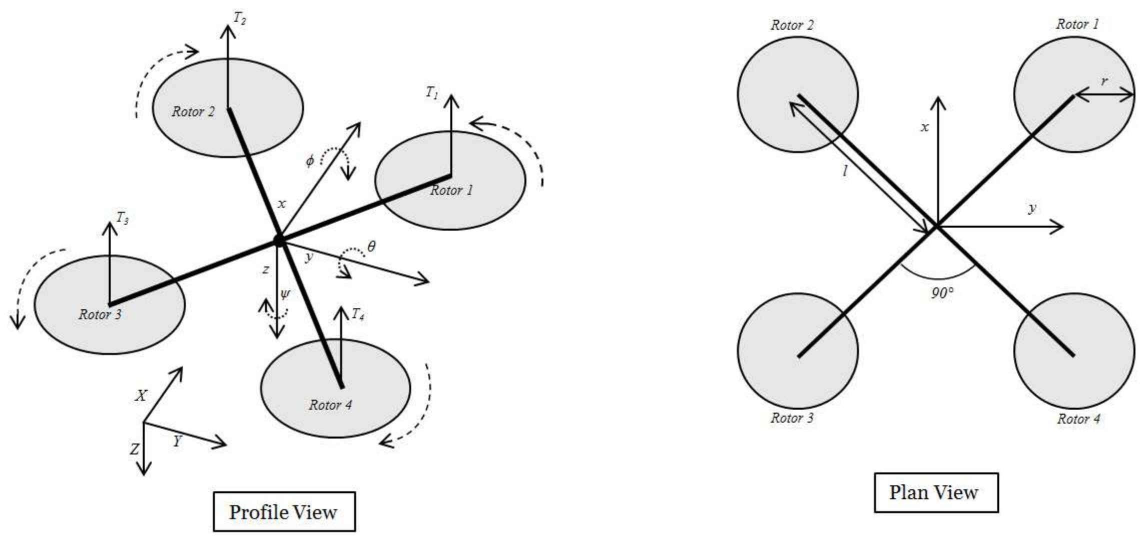


Figure 1.3: Free body diagram of a cross style quadrotor.

The quadrotor's translational motion requires tilting the platform towards the desired axis. Hence, similar to the traditional helicopters, the translational and rotational motion are coupled. Basically, changing the speed of one motor can cause a motion in three degrees of freedom (DOF). This is the reason that allows the quadrotor with six DOF to be controlled by four inputs.

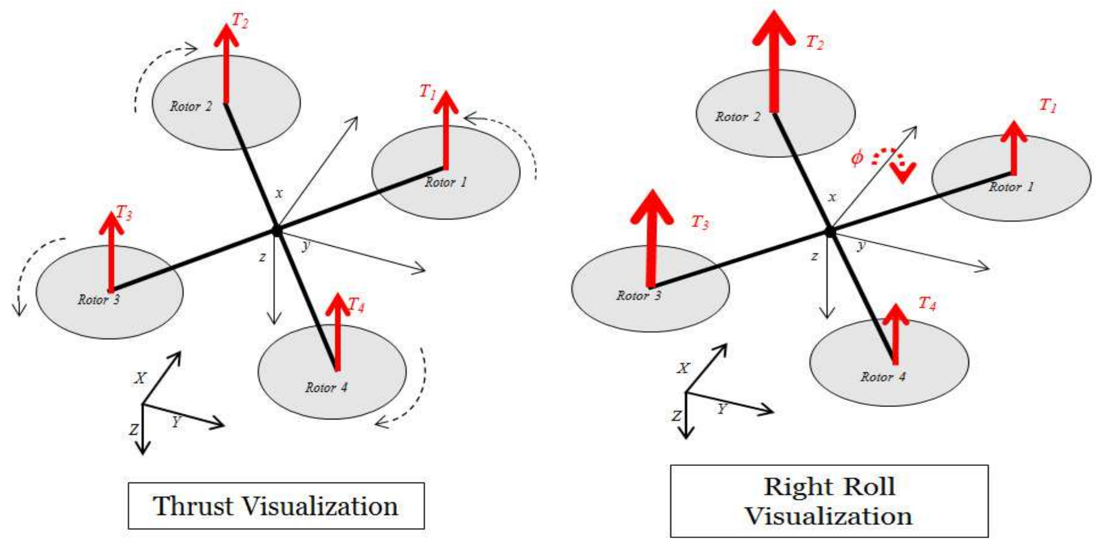


Figure 1.4: Vertical and rolling motion in cross style quadrotor.

The thrust in the vertical direction is produced by the summation of all the rotors' forces. Changing all four rotors' rotational speed by the same amount generates a vertical acceleration which in manual flight, it is controlled by the throttle channel. The pitching motion is produced by applying torque around the y axis. Changing the front rotors' speed against the back rotors will result in the platform tilting around the y axis. The elevator channel controls this motion in the manual flight. The rolling motion exhibits the same mechanism as pitching but around the x axis. The difference of the left and right pair of rotors provides rolling. The pilot uses the aileron channel to induce this motion. Yawing motion is introduced by exerting torque around the z axis. Applying different rotational speed to the pair of counter rotating motors, causes a change of heading. In fact, the yaw motion is generated by the rotors' reactive torque. During manual flight, the rudder channel is used to change the quadrotor's heading. See Figures 1.4 and 1.5 for the flight mechanism illustration.

For hovering flight, all the rotors' thrust must be equivalent. Slight differences may cause the quadrotor to tilt and becomes unstable. Although the quadrotor flight control mechanism may seem simple, in real flight, it is almost humanly impossible to fly it without the help of a controller. For the manual flight, in order to assist the pilot, a commercial gyro will be used to regulate the quadrotor's attitude.

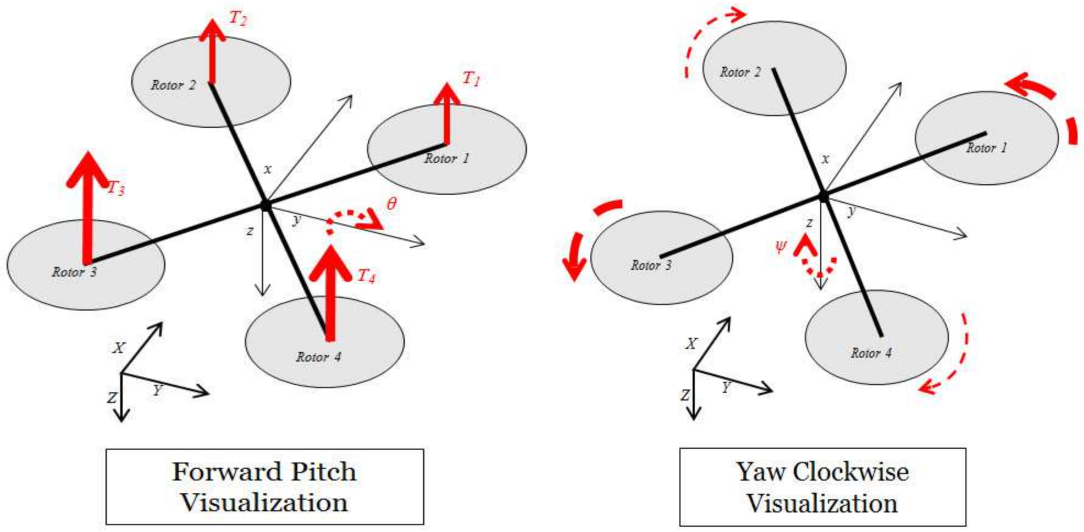


Figure 1.5: Pitching and heading motion in cross style quadrotor.

1.5 Organization of the Thesis

Development of the quadrotor from hardware and software aspects is presented in Chapter 2. In this chapter, the platform body frame, the avionic system including the description of all the components, and sensor data fusion are discussed. Dynamic model of the quadrotor in cross style is mathematically derived and numerically identified in Chapter 3. This chapter also covers the gyro in the loop configuration and model validation parts. Chapter 4 mainly describes control of the quadrotor in hover flight. The nonlinear model simplification, the preliminary ground simulations and analysis, and autonomous flight implementation are presented in this chapter. The thesis is finally concluded in Chapter 5, highlighting the contributions and outlining the future works.

Chapter 2

Hardware and Software Development

2.1 Introduction

This chapter aims to describe the systematic design and development of a cross style quadrotor platform. It will introduce the mechanical structure including the platform body and configuration of the rotor-propeller. Development of the avionics board as an electronic heart of the vehicle, from both hardware and software aspects, will also be extensively described.

The rest of the chapter is organized as follows. Section 2.2 covers the development progress of platform body. A complete description of avionic system including all the components' functionality, features, and interconnections are given in Section 2.3. The computer module firmware and the ground control station are discussed in Section 2.4. The chapter is concluded in Section 2.5.

2.2 Platform Body

The body of the quadrotor should satisfy some important structural features. It should be strong enough to withstand all the forces produced by the rotors, and pay-load of the hardware and the battery, and meanwhile be sufficiently lightweight to allow the mounting of extra sensors or using higher capacity battery. It also should be relatively flexible to absorb the impact of a hard landing. Furthermore, the symmetrical shape of the body is important as it can help to improve the control performance. Considering these features, we first tried to use our custom designed body frame. It was made from light aluminium material in a dimension of 44 cm width and had two layers of mounting area in the center, for mounting the onboard system and the battery. The total body weight including four motor-propeller sets and battery was about 3 kg. Figure 2.1 shows the first platform body which is mounted on the regulator stand. However based on our further tests, we found the manufactured platform is quite heavy and therefore is not aligned with our expectations. Manufacturing it from another material such as carbon fiber would also proven to be cost inefficient. Moreover, since most of the body's part were solid, in case of any damages, the whole body frame would have to be remanufactured from the scratch.

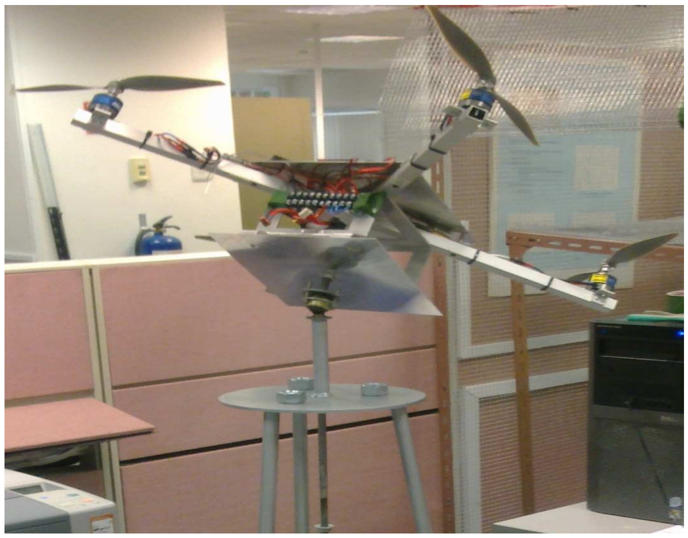


Figure 2.1: First developed quadrotor body, mounted on the regulator stand.

Among the various types of commercial quadrotor frame, we chose the X650 carbon fiber quadrotor from XAircraft [22]. The platform body frame made from carbon fiber which is strong and lightweight. It has four sets of rotors with efficient high strength 12 inch propellers, high frequency ultra pulse width modulation (PWM) supported motor drivers, and a programmable three axes gyro. The total weight of the platform excluding the battery, receiver, and the avionic board is about 800 grams. We added one layer of carbon fiber mounting board on center of the frame where the avionic board, remote control (RC) receiver, GPS module, and battery regulator are placed. To ensure that the center of gravity (CG) is near the central axis, the battery is mounted under the center of the frame to counter the weight caused by adding the onboard system. Figure 2.2 shows the assembled platform.

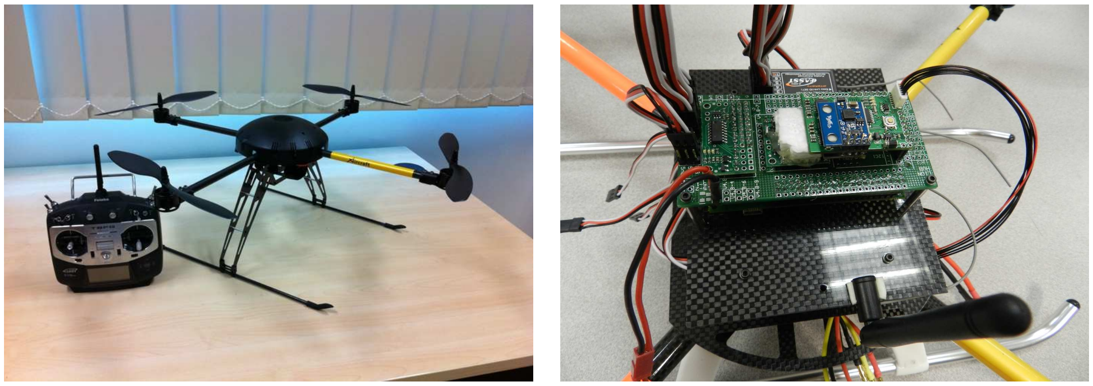


Figure 2.2: Left: x650 quadcopter assembled frame. Right: Mounted avionic board on the frame.

2.3 Hardware Development

2.3.1 Overview of the On-Board System

In this project, we aim to design and development a lightweight, powerful, cheap, and expandable avionics system. Since the goal is to use this setup for different types of mini-UAVs, it is essential to choose standard, low-cost and high quality components for each part. In fact, the current on-board system weight does not exceed 100 g, which would be appropriate for most mini helicopters and airplanes. The block diagram of the on-board system is depicted in Figure 2.3. The system is governed by the main computer board which is chosen to be the Gumstix Overo Fire processor [23], supported by the Summit extension board [24]. This mainboard is sufficiently powerful for normal control processing tasks. This embedded computer has enough communication ports to drive the other attached boards. Also, it has an embedded wireless fidelity (Wi-Fi) module which can be used as a communication link to the ground station or other UAV avionic system. In addition, the Gumstix processor supports micro SD card which is useful for logging the flight data.

The sensing core of the avionics setup is an inertial navigation system (INS). It consists of an inertial measurement unit (IMU), global positioning system (GPS), barometer and ultrasonic sonar. Usually this component is the most expensive item in the avionics systems. However, since we are trying to achieve a low cost design, ArduIMU [25], one the cheapest and lightest IMU in current market is selected. On contrary to its prices, the ArduIMU can provide accurate orientation data. More-

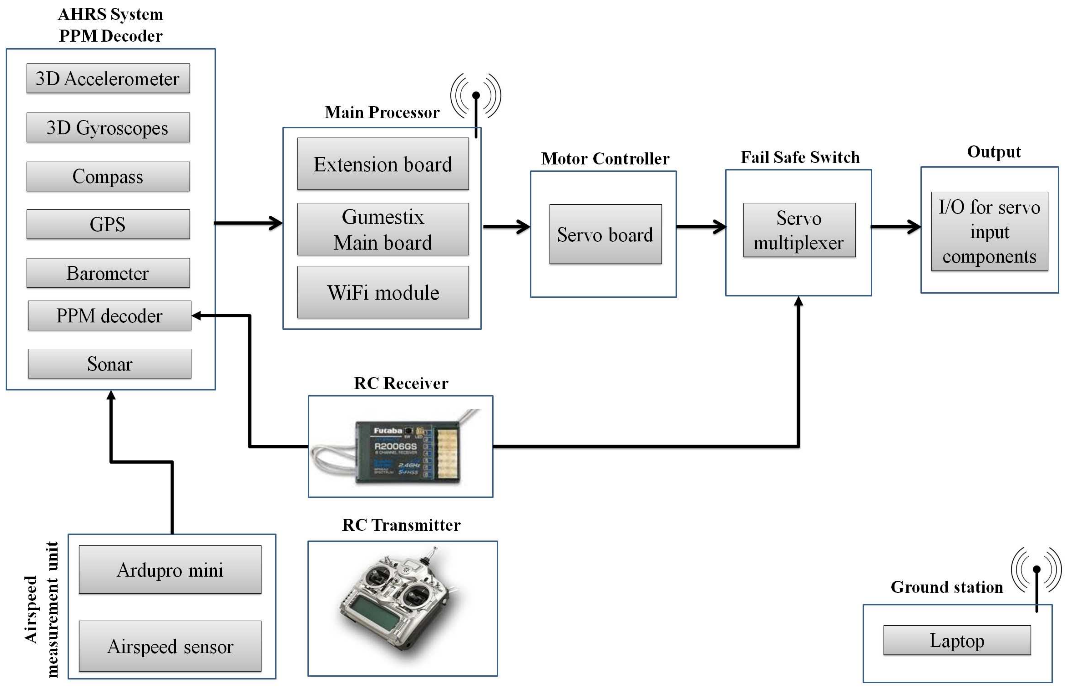


Figure 2.3: Avionic system block diagram.

over, this IMU has a open source code, which gives us the opportunity to modify the code based on our own design demands. In the INS unit, the GPS is responsible for ground position and velocity data. The Ublox GPS [26], as it can be easily matched with ArduIMU is chosen for our design. However, the GPS position, in particular the altitude signal, is quite inaccurate. In order improve it, a barometer and ultrasonic sonar are added to the INS. The barometer unit measures air pressure to further estimate the relative height. Furthermore, for automatic take off and landing scenarios, precise height measurements are necessary. The sonar can be mounted under the platform facing the ground to provide accurate relative altitude data through analysis of ultrasonic signal reflection. We also modified the IMU firmware to read the RC receiver pulse-position-modulation (PPM) signal and send it to the main processor.

This data is necessary for flight data based model identification methods.

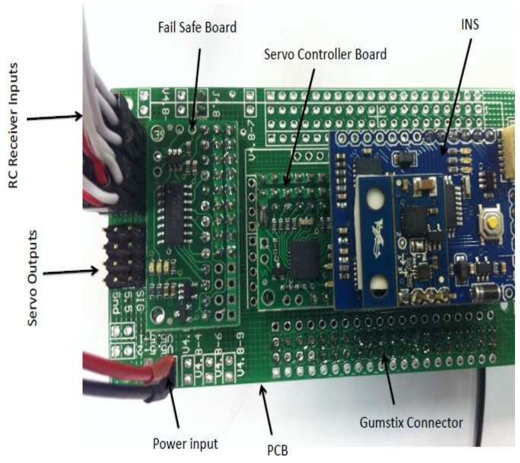
The standard actuators in UAV such as servo motors and rotor-speed controllers are typically driven by a PWM signal. A PWM supported servo controller board is added to the setup to drive the actuators. The Pololu servo controller [27], with having a compact design and light weight, is a very suitable candidate for our design.

Regarding to the safety of the system, fail safe switch is one of the important part of any avionics system. Basically, during any autonomous flight, there are potentially dangerous situations where a reliable option to switch back to manual flight mode is crucial. The fail safe board is a hardware switch that acts as a servo multiplexer. By assigning one channel of the RC receiver to the fail safe channel selection input, the pilot is able to switch back to manual flight mode overriding the autopilot system.

Our system also contains an airspeed sensing board. Even though our platform is a small size VTOL helicopter, and hence the airspeed data fusion may not be necessary, the airplane UAV usually needs this data for control or model identification purpose. Theorem, the airspeed sensing kit is added to the avionic system which includes a stand-alone airspeed sensor, a pitot-tube, and an Arduino-Pro-Mini processor board [28].

In our setup, the data monitoring and sending user commands are handled by a ground control station which is a laptop linked to the on-board computer through a Wi-Fi module. The flight data is continuously sent to the ground station which is displayed to the user. The user during autonomous flight can also send predefined commands to the avionic system. Similar to all hobby aircrafts, we have a radio frequency (RF) transmitter and receiver module for manual flight. The top and bottom side of the assembled avionic board are depicted in Figure 2.4.

It should be noted that the current setup is the result of cooperative team work with my colleagues which is modified and customized for this project. The aforementioned board's hardware and firmware have been developed in process of several years consistent work in NUS UAV group. For more details of past progresses in this regards see [29], [30], [31], [32].



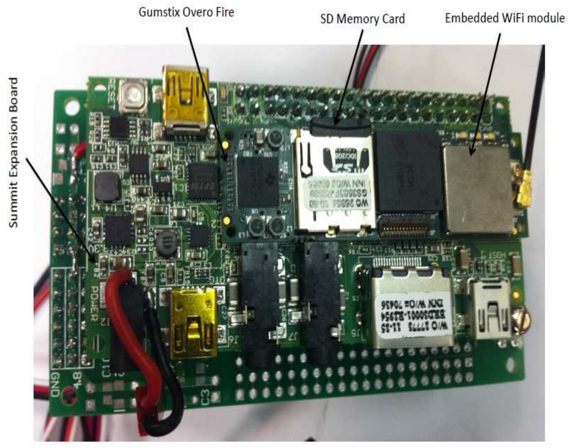


Figure 2.4: Left: Assembled avionic board (Top side). Right: Assembled avionic board (Bottom side).

2.3.2 Embedded Computer Module

The core and brain of avionic system is a computer module. This on-board computer processes all control algorithms, path planning, sensor fusion, and communication with the ground station tasks. Therefore, selecting an appropriate device which is able to handle all theses processes, is very crucial. Obviously, size and weight of this part is also important due to UAV payload and space limitation. Furthermore, since the embedded computer handles main computations, its power consumption could be considerable and should be taken into the account. Considering all aforementioned features, we selected the Gumstix Overo Fire (see Figure 2.5) as a mainboard and Overo Summit as its expansion board. This unit equipped with OMAP3530 processor, a digital signal processing (DSP) chip, Wi-Fi module, two serial universal synchronous/asynchronous receiver/transmitter (USART) ports, and many more attached modules. This embedded computer has a very compact design, while it offers fast computation capability with a 720 MHz processor. It also has a small foot-print with dimensions of 80 mm $\times$ 39 mm and weight of 18 gram. Considering its processing power and extra attached modules, the Gumstix Over Fire currently is one of the low power consumption embedded computer which is available in the market.

Generally speaking, realizing a realtime controller with consistent control execution time is a very important step to achieve a good performance. In order to improve this feature, the QNX Neutrino real time operating system (RTOS) is installed on the Gumstix. We have built up our firmware based on autopilot software developed by NUS UAV research group [33].

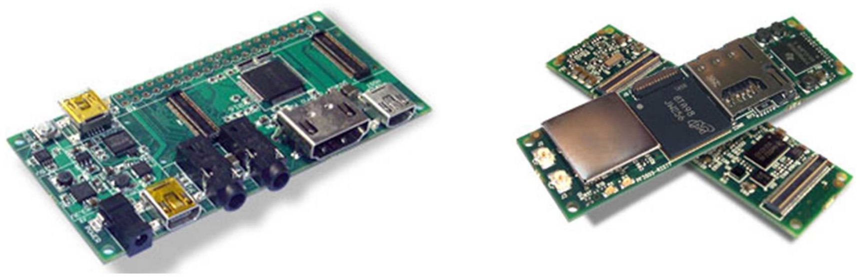


Figure 2.5: Left: Summit Expansion board. Right: Gumstix Overo Fire.

2.3.3 Inertial navigation system Module

An inertial navigation system (INS) is developed to provide all necessary information for the avionic system. Table 2.1 and Figure 2.6 include all the INS components and their brief description.

	Component	Model	Output data
1.	IMU	ArduIMU V_2 Flat	Euler angles, angular rates and body acceleration
2.	Magnetometer	HMC5883L	Heading angle (yaw correction)
3.	GPS	GS407 U-Blox5	Ground position and velocity
4.	Barometer	BMP085	air pressure (can be converted to relative hight)
5.	Sonar	MB1260 XL-MaxSonar-EZL0	Accurate hight (close to ground)

Table 2.1: The INS components and their descriptions.

The core of this module is the ArduIMU. This board contains a gyro, an accelerometer, and an 8-bit microcontroller with Arduino bootloader [25]. The open source firmware combines all sensors data and base on the direction cosine matrix (DCM) algorithm, provides the euler angles and angular rates. In this unit, the three axes body acceleration is directly obtained from accelerometer sensor; and a triple axes magnetometer provides accurate heading.

In order to compensate the GPS altitude inaccuracy, a barometer sensor is added to the INS module. It provides air pressure and temperature data for inertia measure-

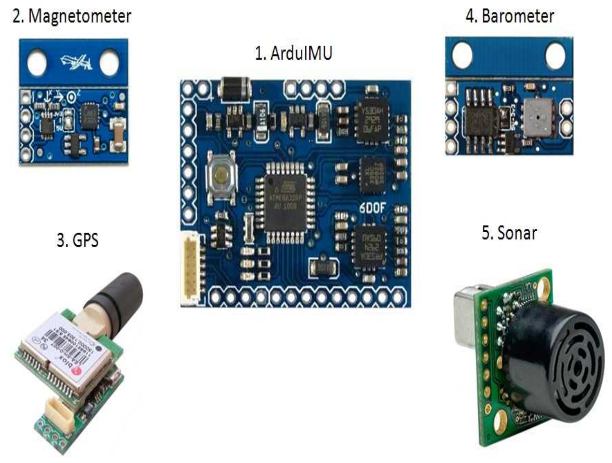


Figure 2.6: INS components.

ment unit (IMU) processor to calculate the relative hight. The barometric estimates the altitude based on (2.1), which is used following the sensor technical manual. However, the coefficients are further tuned in practice to improve the estimation accuracy.

$$Alt = \log(B)\left(\frac{TEMP}{20} + 273.15,\right) 2927.1267 + GPS_{GAlt}, \tag{2.1}$$

where Alt is the true altitude in cm, $TEMP$ is a centigrade temperature, B is a barometric pressures in mm of mercury, and GPS_{GAlt} is the ground GPS altitude. Since global true altitude data is require for navigation, the barometer hight data is first initiated with GPS altitude, and then after passing through a low-pass filter, it will be send to the main processor.

The inertia position information is obtained from the GPS module. The ublox GPS [26] provides the position and velocity data in the inertia frame and sends the data to the IMU's embedded processor via serial port. The update rate of this GPS can be set up to 4 Hz which is normal update rate for outdoor GPS. Furthermore, usually for automatic take off and landing applications, precise height measurement is necessary. The sonar potentially can provide altitude data with centimeter accuracy based on the analysis of emitted ultrasonic signal reflection. This module should be mounted under the platform and towards the ground to observe the distance from the ground surface. The selected sonar has analog, serial, and PWM output types. However, since all the IMU analog and serial inputs have already been occupied with the other devices, the PWM output of the sonar, as depicted in Figure 2.7, is connected to the the IMU processor. The embedded IMU processor measures the pulse width which is proportional to the detected distance range, and converts it to the altitude data. However, this data is only useful in low altitude range and within the sonar beam zone.

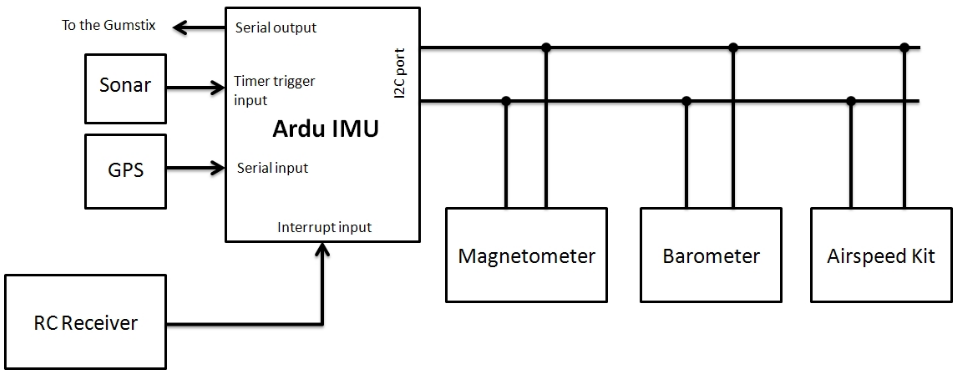


Figure 2.7: INS communication diagram.

Recording the joystick inputs could be necessary for various applications such as UAV model identification. To provide this feature, the IMU firmware is modified to decode the radio control (RC) receiver's PPM signals. Since all the receiver channels data may be required, it is possible to modify the receiver hardware to get all the channels data from one output. Figure 2.8 shows the hardware modification procedure [34]. Following this modification, all the receiver channels are mixed and available on a free channel which is connected to the interrupt input of IMU processor. Decoding algorithm also is realized inside the IMU to fetch all the receiver outputs data and prepare it for autopilot system.

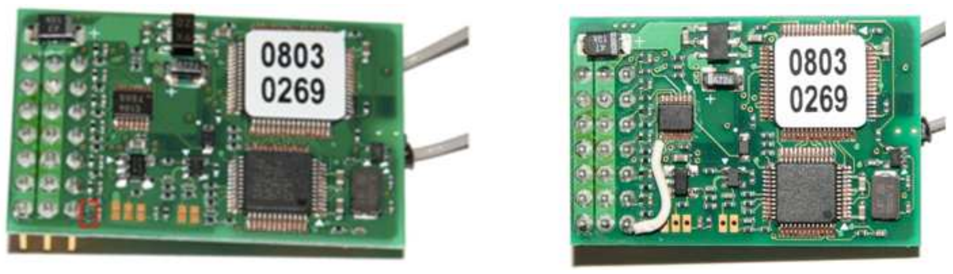


Figure 2.8: RC receiver hardware modification on Futaba R607FF/R617FF.

Finally, all of INS components data are sorted in a specific data packet and is sent to the Gumstix through the serial port. It worth to note that, we have modified the IMU firmware to send selected sensor raw data together with the processed one for further flight test analysis. Specifically, the output data packet contains: Packet header and length, GPS position and velocity, barometer height, all axes accelerations, (roll, pitch, yaw) angles and angular rates, RC receiver outputs and check sum bytes. Additionally if the airspeed kit and sonar are used, their measured data will also be added to the output packet. The Gumstix then extracts the received data, calculates the packet checksum and verifies it. If the information is validated, the sensors data after preprocessing will be passed to the other parts of the program.

2.3.4 Servo Control

The role of this part is to control platform's actuators. Most of the small scale UAV's mechanical parts are controlled by servo motors or brushless motors which usually are driven by electronic speed control (ESC). The standard version of these actuators are controlled by PWM signal which are supported by our servo controller board. Micro serial servo controller from Pololu company is a very light and compact design driver with 6 output channels (see Figure 2.9). It has a transistor-transistor logic (TTL) level serial input which is used to communicate with the main processor. This board can generate PWM signal from 250 us to 2.75 ms, which covers the operating range of most commercial servo input actuators. Supporting high baud rate also allows fast refreshing rate. Each channel's output hold its value till next update command. This characteristic emulate a Zero-Order-Hold block in discrete systems which actually placed in the output of the controller.

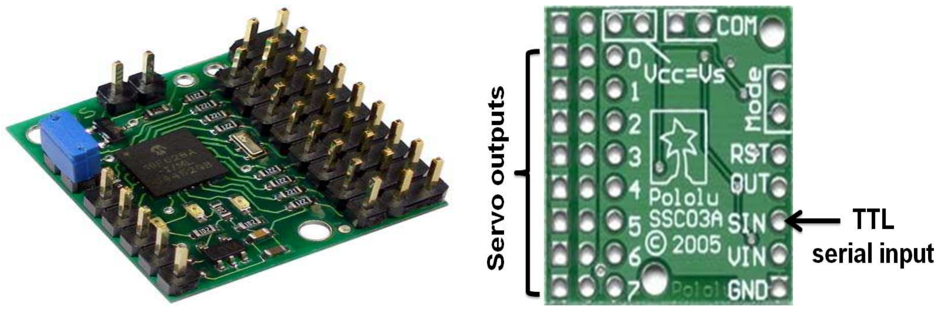


Figure 2.9: Left: micro serial servo controller board. Right: servo controller board input-output connection.

2.3.5 Fail Safe Switch

As it is mentioned earlier, the fail safe board is a crucial component for safety of the system. In fact, there is possibility that the autopilot becomes unstable and it may not be possible to recover it back by the ground station emergency command or any intelligent fault detection algorithm. This behavior could be due to the processor fault, sensor failure, large disturbance, unstable controller algorithm, or even loose hardware connection. Thus, it is necessary to design a remote control switch to remove the autopilot from the control loop and give the control authority to the pilot in face of any unpredictable situations. It is important to have a pure hardware switch that could achieve this safety feature despite any processor fault or wrong firmware algorithm. The fail safe switch is a good solution for this problem. This board serves as a four channels multiplexer of radio control signals. Selection input channel allows easy switching between two independent signal sources which in our setup are autopilot and RC receiver outputs. Figure 2.10 shows the fail safe board wiring configuration. We assign an extra RC receiver channel, an on-off type, to the fail safe switch input during autonomous flights. The pilot thus is able to easily switch back to manual mode, by his on hand RC transmitter.

2.3.6 Airspeed Sensor Board

Our avionic system also includes airspeed sensor fusion. Even though for small size helicopters similar our current platform, airspeed data usually is not necessary, we have added this part to make the avionic system usable for any further application on

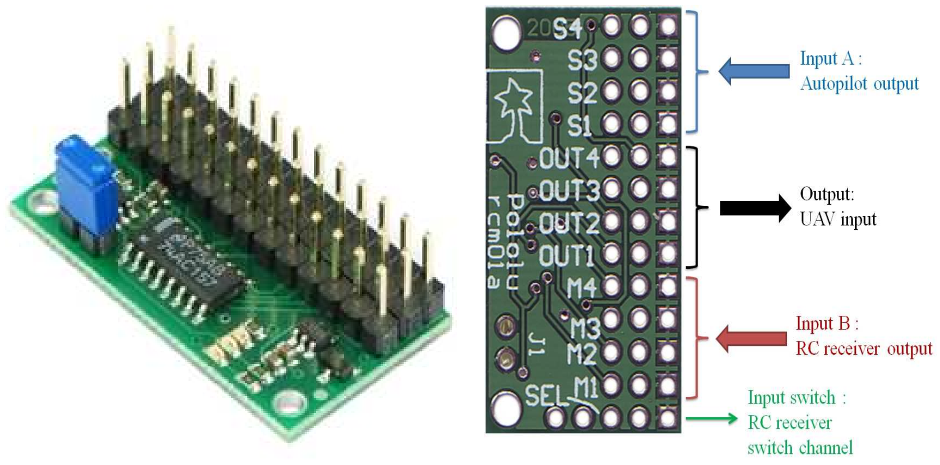

S4
S3
S2
S1
OUT4
OUT3
OUT2
OUT1
M4
M3
M2
M1
SEL
J1
Pololu rcm01a
Input A : Autopilot output
Output: UAV input
Input B : RC receiver output
Input switch : RC receiver switch channel

Figure 2.10: Left: fail safe board. Right: fail safe board input-output connection.

airplane UAVs. In addition, for large size helicopters the airspeed data could be used to find the propeller or body, lift and drag coefficients. Hence, the airspeed data fusion board as an optional part is designed and added to the avionic system. This board includes an air pressure sensor, an Arduino-Pro-Mini board which acts as a sensor driver and communication link. The airspeed sensor [35] measures relative air pressure and provides analog output. The processor board is an 8-bit microcontroller with Arduino bootloader which reads the sensor analog output, calculates air pressure and the corresponding airspeed, and sends the airspeed data to the IMU. Since Arduino-Mini board and the sensor have different operating voltage, a resistor based voltage divider is used between these boards. Figure 2.11 shows the airspeed kit setup.

The sensor is equipped with a pitot tube which has static and dynamic probes. The pitot tube should be mounted in way that the propeller airflow does not affect on it.

In order to have an accurate airspeed data, the estimation formula parameters

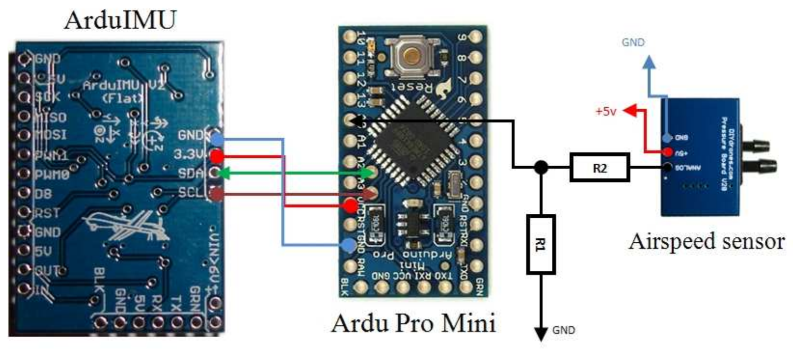


Figure 2.11: Airspeed sensor fusion setup and components.

have to be calibrated. To do so, a wind tunnel calibration experiment is conducted. The pitot tube is placed inside the wind tunnel and a reference air flow with different speeds is applied to the sensor. The measured data from the airspeed sensor unit is recorded and then compared to the wind tunnel reference to calibrate the sensor parameters. The experiment result after calibration is depicted in Figure 2.12. As it is clear, the airspeed kit output is quite accurate.

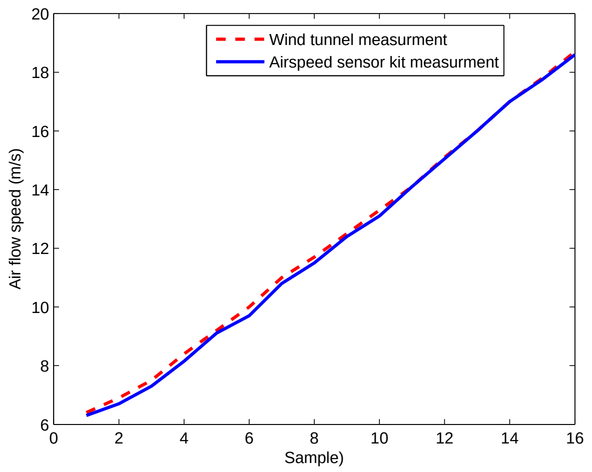


Figure 2.12: The measured airspeed in compared to wind tunnel air flow reference.

2.4 Software Development

2.4.1 Firmware

The onboard firmware is built up based on the NUS UAV research group software [30]. Although this software was originally developed for a standard helicopter UAV, the modular structure provided an option to customize it based on our requirements and the possible hardware differences. The on-board software has been developed based on the QNX Neutrino real time operating system (RTOS). Precise time management and reliable real-time program execution are the main features of this operating system, which provided an appropriate environment to design a comprehensive onboard software. Most autonomous flight scenarios need multi-tasking processing. To offer such a feature, a multi-thread structure firmware has been designed to manage all the autonomous processing such as sensors data fusion, flight data logging, flight scheduling, ground station communication, and control tasks. In this structure each thread is in charged of one specific task. The main thread manages all the tasks and controls the threads' activation timing and messages. A hardware-in-the-loop feature also can be implemented through this the software. In this simulation mode, the real platform is replaced with the identified model while the remaining hardware and parts are actually operating. Therefore, in this mode, the controller behavior, hardware, and firmware of the system can be evaluated. Furthermore, since all the onboard components are in the simulation loop, most of the practical issues such as the sensors fusion and controller execution delay, and output update rate variation can be observed. As a result, if the identified model is accurate, the simulation result should be very close to the real flight test. The frame work thread names and descriptions are given in Table 2.2.

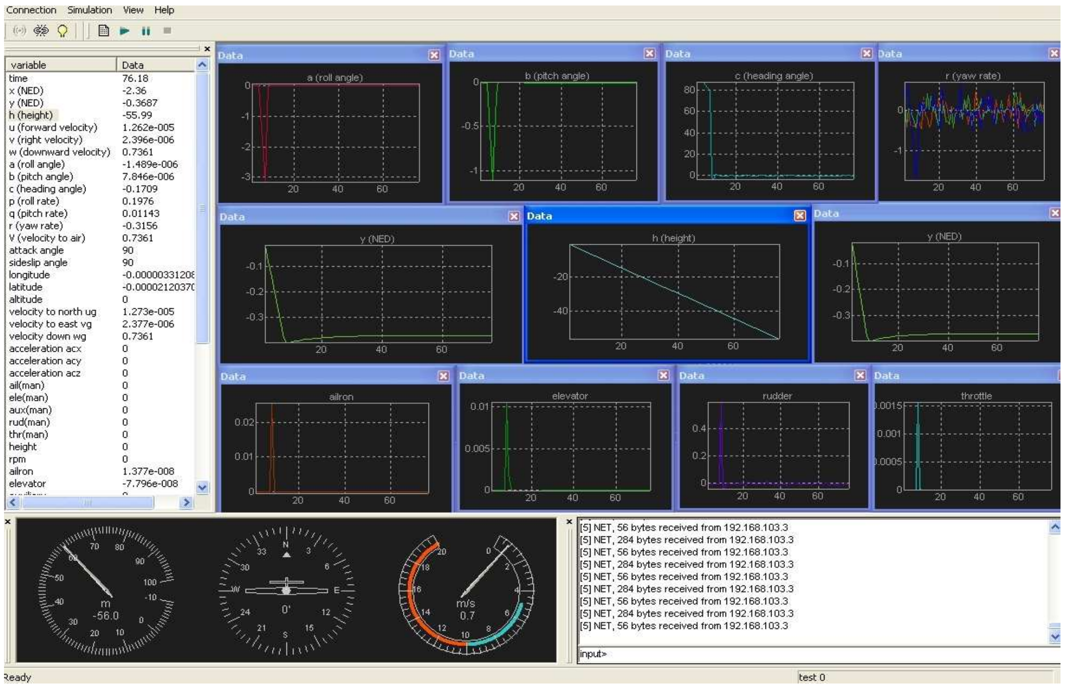
Thread name	Task description
IMU	INS data fusion and servo reading
DLG	Data logging
CTL	Control functions
SVO	Servo driving
CMM	Communication to control ground station

Table 2.2: The software framework threads description.

The INS data fusion is handled in the IMU thread. The software at a consistent 20 ms update rate activates this thread to read the INS output which in our setup also contains the RC receiver PPM data. The data logging thread, DLG, helps by recording the flight data. The time, sensors raw and filtered data, controller outputs and user defined variables are logged in a memory card. The data is written in a text file format which can be easily read in a computer and imported to MATLAB or excel programs for post-flight analysis. The control thread, CTL, realizes the flight controllers where the flight scheduling and hierarchical control algorithms are executed. In a real flight, the controller reads the UAV states from the IMU thread, but when hardware-in-the-loop simulation is applied, the states are updated from the identified dynamic model. The main role of servo driver thread is sending the controller output to the servo board, reading the gyro output, and scaling the received data for logging thread. The communication with ground control station is realized in the communication thread, CMM, where the whole quadrotor states together with RC transmitter data are sent to ground station via Wi-Fi. The preliminary received command processing also is done in this thread.

2.4.2 Ground Control Station

The ground control station has been developed by the NUS UAV team [33]. This software is mainly used for monitoring the aerial vehicle's behavior, sending the flight task command, and hardware-in-the-loop simulation. Figure 2.13 shows the graphic user interface (GUI) of the ground control station. In the left side of the GUI, the received states are listed; the user can choose one or several signals to be plotted in the graph windows. The monitored state variables are the position, body and ground velocity, ground acceleration, attitude angles and angular rate, manual and auto servo outputs, and GPS raw data. At the bottom of the GUI, the command bar input is designed to obtain the input command from user which also are defined in the embedded onboard firmware as task commands. In the predefined user commands however, some of them may have two parts, command name and input argument. For example trajectory tracking command and the desire path number as the command argument. Once the onboard computer command is received, it will be processed in several steps. First it is validated, and if it contains extra input, it will be extracted. If the new command requests a change of flight task, a sequence of procedure will be followed. For instance, sending *hover* will execute the following sequence of process to perform a hover flight scenario: the received command is identified to be validated, the task management layer obtains the current position, sets it as a position reference for the outer-loop controller, and the control layer call the corresponding outer-loop and inner-loop hovering controller functions to hold the quadrotor at the reference position.



variable	Data
time	76.18
x (NED)	-2.36
y (NED)	-0.3687
h (height)	-55.99
u (forward velocity)	1.262e-005
v (right velocity)	2.396e-006
w (downward velocity)	0.7361
a (roll angle)	-1.489e-006
b (pitch angle)	7.846e-006
c (heading angle)	-0.1709
p (roll rate)	0.1976
q (pitch rate)	0.01143
r (yaw rate)	-0.3156
V (velocity to air)	0.7361
attack angle	90
sideslip angle	90
longitude	-0.0000033120€
latitude	-0.0000212037C
altitude	0
velocity to north ug	1.273e-005
velocity to east vg	2.377e-006
velocity down wg	0.7361
acceleration acx	0
acceleration acy	0
acceleration acz	0
ail(man)	0
ele(man)	0
aux(man)	0
rud(man)	0
thr(man)	0
height	0
rpm	0
ailron	1.377e-008
elevator	-7.796e-008

```
[5] NET, 56 bytes received from 192.168.103.3
[5] NET, 284 bytes received from 192.168.103.3
[5] NET, 56 bytes received from 192.168.103.3
[5] NET, 284 bytes received from 192.168.103.3
[5] NET, 56 bytes received from 192.168.103.3
[5] NET, 284 bytes received from 192.168.103.3
[5] NET, 56 bytes received from 192.168.103.3
[5] NET, 284 bytes received from 192.168.103.3
[5] NET, 56 bytes received from 192.168.103.3
input>
```

Figure 2.13: The GUO of ground control station.

2.5 Conclusion

In this chapter, the hardware and software development of the cross style quadrotor platform were discussed. The platform body was built up based on a commercial quadrotor frame and equipped with an autopilot system. The developed autopilot system consist of a Gumstix processer, Arduino based INS unit, customized real-time QNX operating system, and a remote control safety switch. The detailed description of all these components, together with the on-board software structure, and the ground control station features were covered in this chapter.

Chapter 3

Quadrotor Dynamic Modeling

3.1 Introduction

In this chapter, we aim to mathematically and numerically derive the nonlinear dynamic model of a cross style quadrotor. The first step is to derive the platform mathematical structure based on the platform kinematic features. The second step is to observe all the force and momentum acting on the platform. However, depending on the model accuracy requirement, it is possible to neglect minor acting forces and torques. Following these steps, the independent model parameters which are needed to be numerically identified are highlighted. Based on the physical parameters characteristics, either a ground experimental setup or manual flight test is conducted. The experiments collected data then will be analyzed to identify the unknown parameters. Finally a model verification analysis is designed to illustrate the fidelity of identified dynamic model.

The remaining contents of this chapter are organized as follows. The quadrotor mathematical dynamic model is presented in Section 3.2. Section 3.3 and 3.4 report the whole model identification procedure. The ground and flight experiments, data preprocessing and analysis methods are given in these sections. This is followed by model validation results which are demonstrated in Section 3.5. This chapter is concluded with Section 3.6.

3.2 Mathematical model

This section derives the mathematical model of the quadrotor in cross style. As a first step, the quadrotor dynamic and kinematic equations should be defined. The dynamic model describes how the applied forces and torques result in translational and rotational accelerations. The kinematic equations also addresses the relation between the vehicle's position and velocity [36].

The inertial and body frames are introduced as shown in Figure 3.1. The origin of the body frame is assumed to be at the center of gravity (CG) with definition of the following notations. The North-East-Down (NED) inertial frame and body frame is defined as $F^I = (e^x, e^y, e^z)$ and $F^B = (e^{xb}, e^{yb}, e^{zb})$, respectively. Body orientation with respect to (NED) frame is defined as $\eta \triangleq [\phi, \theta, \psi]^T$ which respectively are roll, pitch, and yaw. The body frame angular rates are introduced as $\omega^b \triangleq [p, q, r]^T$ as rolling , pitching, and yawing rates. Position in the inertia frame is defined as $\xi \triangleq [x, y, z]^T$ and body frame velocities as $v^b \triangleq [u, v, w]^T$.

The rigid body dynamic equation subjected to acting forces and torques in the body-fixed frame and in Newton-Euler scheme is:

$$\begin{bmatrix} mI_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & J \end{bmatrix} \begin{pmatrix} \dot{v}^b \\ \dot{\omega}^b \end{pmatrix} + \begin{bmatrix} \omega^b \times (mv^b) \\ \omega^b \times (J\omega^b) \end{bmatrix} = \begin{pmatrix} F^b \\ \tau^b \end{pmatrix}, \tag{3.1}$$

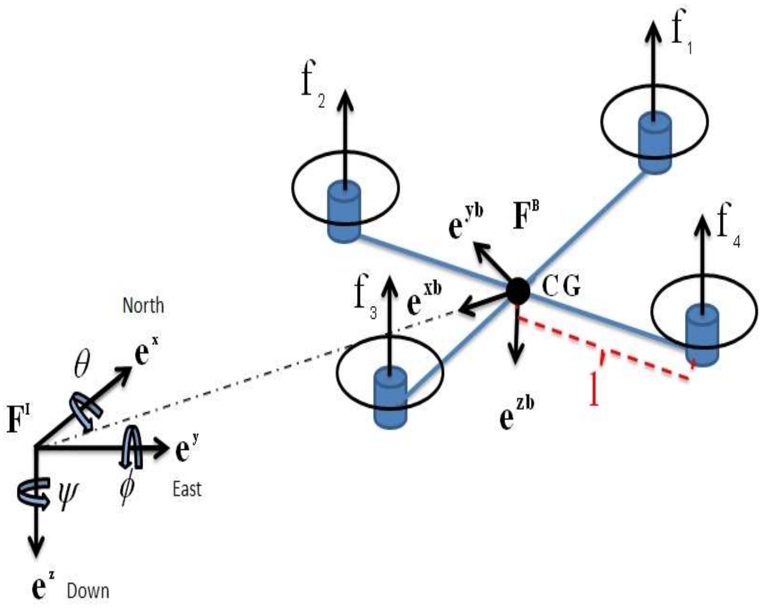


Figure 3.1: Inertia and body coordinate system.

where m [kg] is the mass of platform, I is an identity matrix, and J is the moment of inertia matrix for the quadrotor platform. The vehicle body has a axis-symmetric structure and hence it can be assumed the constant inertia matrix is essentially symmetric about all the three axes which implies $J = diag(J_{xx}, J_{yy}, J_{zz})$ [19]. F^b and τ^b are total force and torque applied to the aircraft body, respectively. The body velocity is projected to its inertial reference frame through the rotational matrix, $\dot{\xi} = Rv^b$, given as

$$R = \begin{pmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta c_\phi + s_\psi s_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{pmatrix}, \tag{3.2}$$

where $c_k = \cos(k)$ and $s_k = \sin(k)$. The transformation matrix from $\dot{\eta}$ to ω^b also is given as $\dot{\eta} = \Gamma \omega^b$, where

$$\Gamma = \begin{pmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{pmatrix}. \tag{3.3}$$

The main acting forces on the quadrotor are the weight, rotor thrusts, and body drag [37]. The motor thrust f_i is proportional to square of the propeller's angular velocity Ω_i, and acts along the e^z as shown in Figure 3.1, and defined as

$$f_i = K_\Omega \Omega_i^2 \tag{3.4}$$

The rotor rotational speed is controlled through the input voltage v_i [13], given as

$$\Omega_i = \frac{K_v}{1 + \tau_f s} v_i, \tag{3.5}$$

where K_Ω and K_v are the constant coefficients, and τ_f is the time constant of the rotor thrust model.

As the platform moves through the air, it produces drag which in the body frame can be written as

$$D = -\frac{1}{2}\rho C_d \left| v^b \right| v^b, \tag{3.6}$$

where ρ is the density of air and C_d is the drag coefficient.

Gravity also exerts force on the quadrotor. This force acts on the CG and in the inertial frame is given by

$$f_g^I = \begin{pmatrix} 0 & 0 & mg \end{pmatrix}^T. \tag{3.7}$$

However, since we are defining all the acting forces in the body frame, the gravity force must be transferred to the body frame which is then obtained as the following

[38].

$$f_g^b = R^T f_g^I = \begin{pmatrix} -mg\sin(\theta) \\ mg\cos(\theta)\sin(\phi) \\ mg\cos(\theta)\cos(\phi) \end{pmatrix} \tag{3.8}$$

As a result, the total acting force in body frame can be given as

$$F^b = -\frac{1}{2}\rho C_d \left|v^b\right| v^b + f_g^b - \sum_{i=1}^{4} f_i e^z. \tag{3.9}$$

The roll, pitch and yaw $(\tau_\phi, \tau_\theta, \tau_\psi)$ torques acting on the body frame are mainly produced by the rotors' deferential thrust. Specifically, changing the motors' thrust resultantly provides roll and pitch torques. However, yaw is mainly generated by differential thrust between the two pairs of counter-rotating motors. This momentum, in fact is produced by the rotors' reactive torque Q, which is given as

$$Q_i = d\Omega_i, \tag{3.10}$$

where d is a rotor induced torque drag coefficient which mainly depends on the propeller's specification. Thus, the acting torques $\tau^b = [\tau_\phi, \tau_\theta, \tau_\psi]$ on body fixed frame are as shown in (3.11) to (3.13).

$$\tau_\phi = \frac{\sqrt{2}}{2} l[(f_2 + f_3) - (f_1 + f_4)], \tag{3.11}$$

$$\tau_\theta = \frac{\sqrt{2}}{2} l[(f_1 + f_2) - (f_3 + f_4)], \tag{3.12}$$

$$\tau_\psi = Q_1 - Q_2 + Q_3 - Q_4. \tag{3.13}$$

The quadrotor complete dynamic model can be derived by substituting (3.9) and (3.11) into (3.1) and projecting the body states to the inertial frame with the transformation matrices (3.2) and (3.3). The complete mathematical nonlinear quadrotor model is presented as follows.

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - s_\psi c_\phi & c_\psi s_\theta s_\phi + s_\psi c_\phi \\ s_\psi c_\theta & s_\psi s_\theta s_\phi + c_\psi c_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi \\ -s_\psi & c_\theta s_\phi & c_\theta c_\phi \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \tag{3.14}$$

$$\begin{pmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} rv - qw \\ pw - ru \\ qu - pv \end{pmatrix} + \begin{pmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi \end{pmatrix} + \frac{1}{m} \begin{pmatrix} 0 \\ 0 \\ -F^b \end{pmatrix}, \tag{3.15}$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{pmatrix} \begin{pmatrix} p \\ q \\ r \end{pmatrix}, \tag{3.16}$$

$$\begin{pmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{pmatrix} = \begin{pmatrix} \frac{J_{yy}-J_{zz}}{J_{xx}} qr \\ \frac{J_{zz}-J_{xx}}{J_{yy}} pr \\ \frac{J_{xx}-J_{yy}}{J_{zz}} pq \end{pmatrix} + \begin{pmatrix} \frac{1}{J_{xx}} \tau_\phi \\ \frac{1}{J_{yy}} \tau_\theta \\ \frac{1}{J_{zz}} \tau_\psi \end{pmatrix} \tag{3.17}$$

The next step is to determine the unknown model parameters and conduct identification experiments to obtain the numerical representation of this model.

3.3 Model Parameter Identification

In this section, we present the model parameter identification procedure. Based on the model structure which was derived in the previous section, the unknown parameters are determined. However, not all of them are necessary to be identified. For instance, the body drag force due to the low velocity of the platform, does not significantly influence the quadrotor's dynamic and thus it can be neglected. The unknown parameters are identified through the static and dynamic experiments. The ground experiments are mainly suitable for identifying the parameters that can be measured in a static condition such as platform weight, center of gravity position, and moment of inertia. For those parameters that should be identified based on the flight data analysis, data logging hardware is designed. The developed avionic system is able to record the RC receiver inputs, IMU and GPS outputs which respectively contain joystick signal, quadrotor attitude, and position data. Additionally, a PWM measurement board is added to system to record the gyro outputs, i.e., motors inputs. Figure 3.2 shows the hardware configuration and the logging signal points.

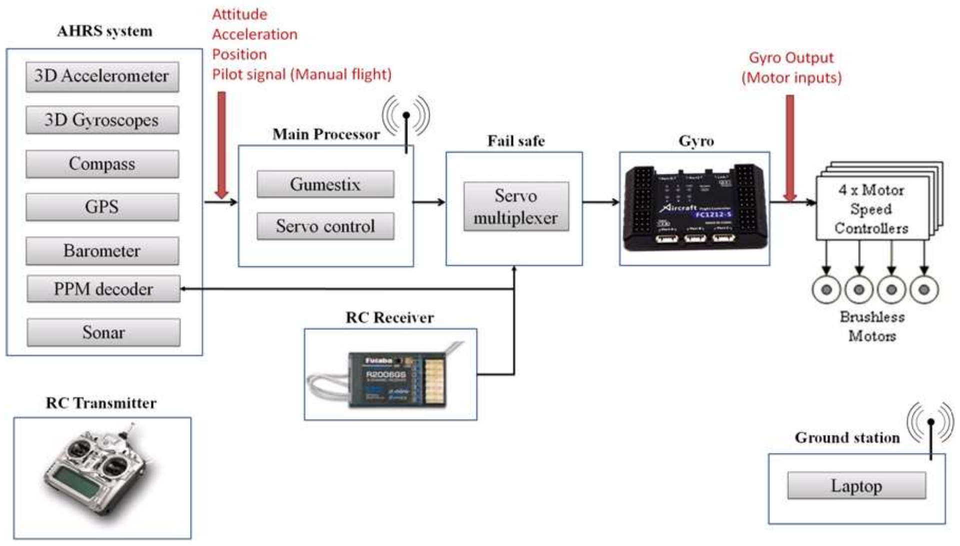


Figure 3.2: The avionic system hardware configuration and the logging signal points.

3.3.1 Static Experiments

Some of the model unknown parameters can be estimated by ground experiments. The ground test equipments for modeling the rotor thrust are a thrust stand as a force meter, tachometer for measuring the propeller rotational speed, servo control board which drives the motors and, current and voltage monitoring units. Figure 3.3 shows the assembled setup. Before each experiment, the force measurement unit and tachometer are carefully adjusted and calibrated. This experiment is mainly used for modeling the rotors thrust dynamics, however, the current and voltage data also will be used to estimate total power consumption. The platform's moment of inertia also can be measured through the static experiments. We follow the same method used in [39] which is theoretically based on the Trifilar Pendulum theory. Through the conducted ground experiments, the following parameters are numerically identified (see Table 3.1).

Parameters	Method
J_{xx}, J_{yy}, J_{zz}	Trifilar pendulum theory
CG location	Trifilar pendulum theory
K_Ω, K_v	Rotor-propeller thrust measurement
τ_f	Thrust measurement with periodic input
l	Meter
m	Weighing scale

Table 3.1: Unknowns model parameters and method of identification in static experiments.

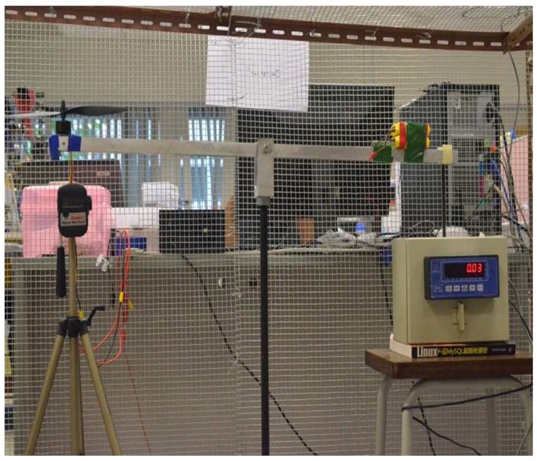
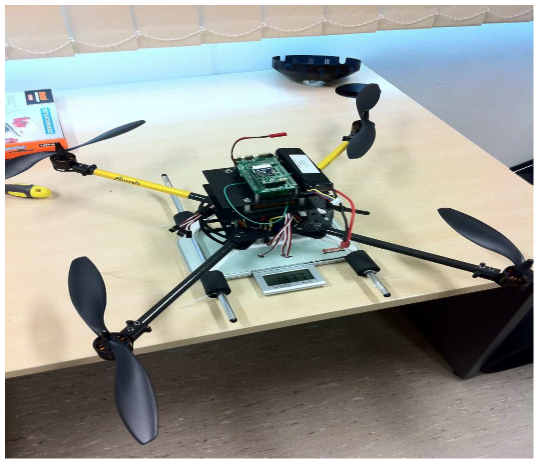



Figure 3.3: Left: rotor thrust measurement stand. Right: measuring the whole platform weight.

3.3.1.1 Inertial Measurement

Generally, every object could have different moment of inertia which varies for different axis. To measure the inertial properties of rigid structures, various methods have been designed which can be generally categorized as time domain methods and frequency domain methods [40]. Among them, trifilar pendulum as a time domain approach, has been known as a reliable, accurate, and simple method [41]. For the quadrotor, we follow the approach used in [39] and described in [42] which is called trifilar (three wires) pendulum method.

The mass moment of inertia can be consider as a point on an ellipsoid, if the center of gravity lies on the corresponding axis of rotation. For the quadrotor, since the center of gravity and principal axes are known, the mass moment of inertia about each principal axis can be measured by creating a torsional pendulum such that the rotation of the pendulum passes through the center of gravity and aligned with the axis of rotation [42].

Using the trifilar pendulum theory [43], by the setup of a simple experiment, the moment of inertia of a compound object such as the quadrotor with onboard systems can be measured. Since quadrotor structurally is symmetric, only the moment about each axis is required to be identified. Furthermpre, the moment of inertial in the x and y directions, are expected to be almost similar due to the symmetrical setup. The platform in this case is suspended by three lines with equal length to create a torsional pendulum (see Figure 3.4). The suspended quadrotor is then excited minutely in the axis of interest. The natural frequency of the trifilar pendulum (3.18) then can be measured by recording the period of platform oscillation. For this experiment, since (3.18) is derived using the small angle assumption [42], the initial displacement should be sufficiently small, i.e., no more than 10 degrees. The rolling, pitching and yawing moment of inertia then can all be calculated using the following equation

$$J_{xx,yy,zz} = \frac{mgl_1l_2l_3t^2}{4\pi^2 L}\frac{l_1\sin\alpha_1 + l_2\sin\alpha_2 + l_3\sin\alpha_3}{l_2l_3\sin\alpha_1 + l_1l_2\sin\alpha_2 + l_1l_2\sin\alpha_3}, \tag{3.18}$$

where α_i is the angle between the three strings which for our setup all are 120°. As Figure 3.4 shows, l_i is the distance between the strings and L is length of them. t is the platform oscillation period around the axis of measured moment of inertia. In order to obtain the period, first the platform should be suspended from three points around the desired axis and then excited gently. The platform oscillation is recorded by a camera for several periods. Each perturbation experiment is done three times to improved accuracy and reliability of the results. The captured video is analyzed in computer and the average of periods for all experiments is calculated. This procedure is repeated to obtain the moment of inertia about each axis which are presented in Table 3.2.

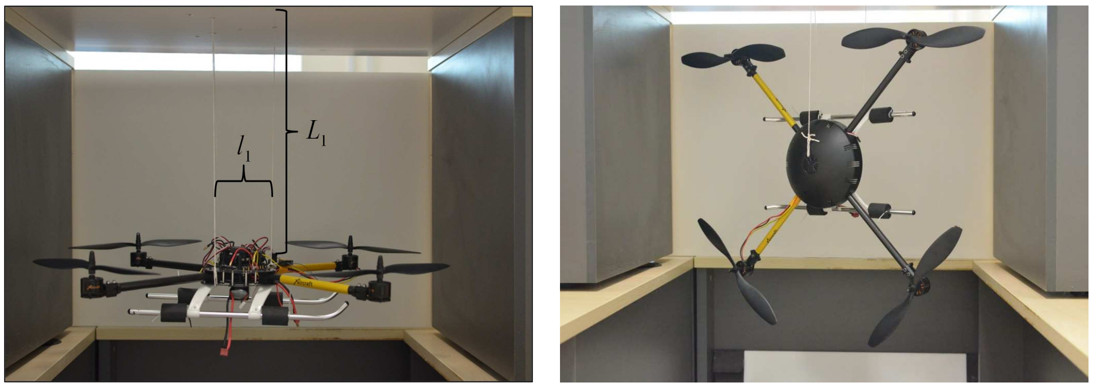


Figure 3.4: Left: J_{zz} measurement experiment setup. Right: J_{xx},J_{yy} measurement experiment setup.

3.3.1.2 Central Gravity Location

Finding and adjusting the center of gravity is very important step to have a stable platform. In order to find the center of gravity location, in at least three experiments, the platform should be suspended from one arbitrary point. For simplicity, we attached the string to the tip of quadrotor rotor arm. For each plane (x-y, x-z, y-z), the platform is suspended from an appropriate point such that the desired plane faces the camera in which a high resolution picture is captured. In the computer, a straight line connects the attached points. The intersection of these lines gives the location of center of gravity. Figures 3.5 illustrates this procedure. For our quadrotor, CG is

Parameter	Value	Description
J_{xx}	0.03356 kg.m^2	Rolling moment of inertia
J_{yy}	0.03122 kg.m^2	Pitching moment of inertia
J_{zz}	0.05423 kg.m^2	Yawing of inertia
K_Ω	0.059	Rotor thrust to rotational velocity coefficient
K_v	9.2994	Rotor rotational velocity coefficient
K_f	4.4861	Rotor thrust lift coefficient
τ_f	0.06 s	Rotor thrust model time constant
l	0.325 m	Rotor to CG distance
m	1.37 kg	Total platform mass

Table 3.2: Numerical value of identified parameters from the ground experiments.

obtained quite close to center of the platform which verifies the correct placement of the avionics system.

3.3.1.3 Rotor Thrust Measurement

Since the quadrotor mainly operates in hovering conditions and at low velocities, it would be reasonable to observe the rotor dynamic through a ground experiment. In fact, it is assumed that the UAV will fly in low wind conditions and, therefore the rotor-propeller's indoor behavior will replicate a real outdoor flight. In order to identify the rotor dynamic, a thrust force experimental setup was designed. For a full range of input from minimum to full throttle, rotor thrust, propeller angular velocity,

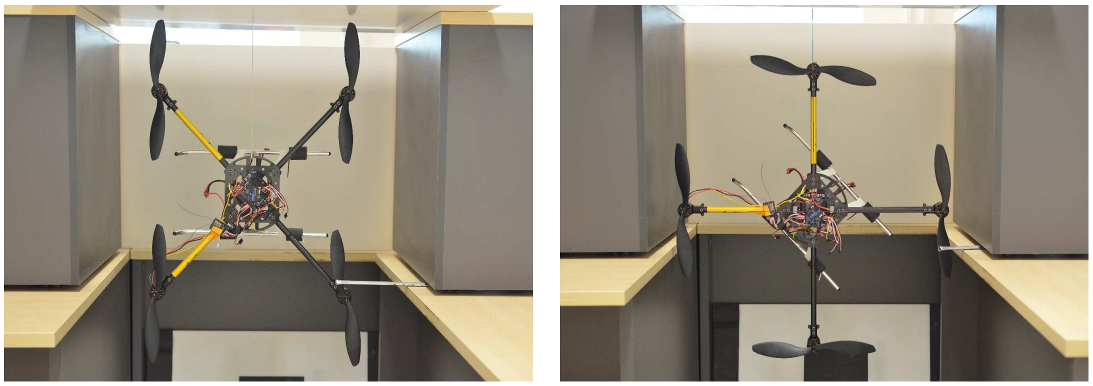


Figure 3.5: Finding the CG location. Left: Captured picture. Right: Printed intersectional lines.

rotor current, and applied voltage are measured. In order to improve the accuracy of the results, this experiment was repeated at least three times. Figure 3.6 shows the experiment's collected data.

Figure 3.6(b) shows that the square of angular rotational velocity (Ω^2) is proportional to the thrust which validates (3.4). However, from the Figure 3.6(c) it can be observed that, on contrary to equation (3.5), the propeller angular velocity is not linearly proportional to the input voltage. This phenomena seems to be due to the motor speed controller characteristic. In fact, the electronic speed controller (ESC) regulates the rotor rotational speed to regulate the output power. Hence, as Figure 3.6(a) illustrates, the rotor thrust is approximately linearly proportional to the servo input. From the control aspect, only this relationship is necessary. As a result the below equation is valid for our setup.

$$f_i = \frac{K_f}{1+\tau_f s} v_i, \tag{3.19}$$

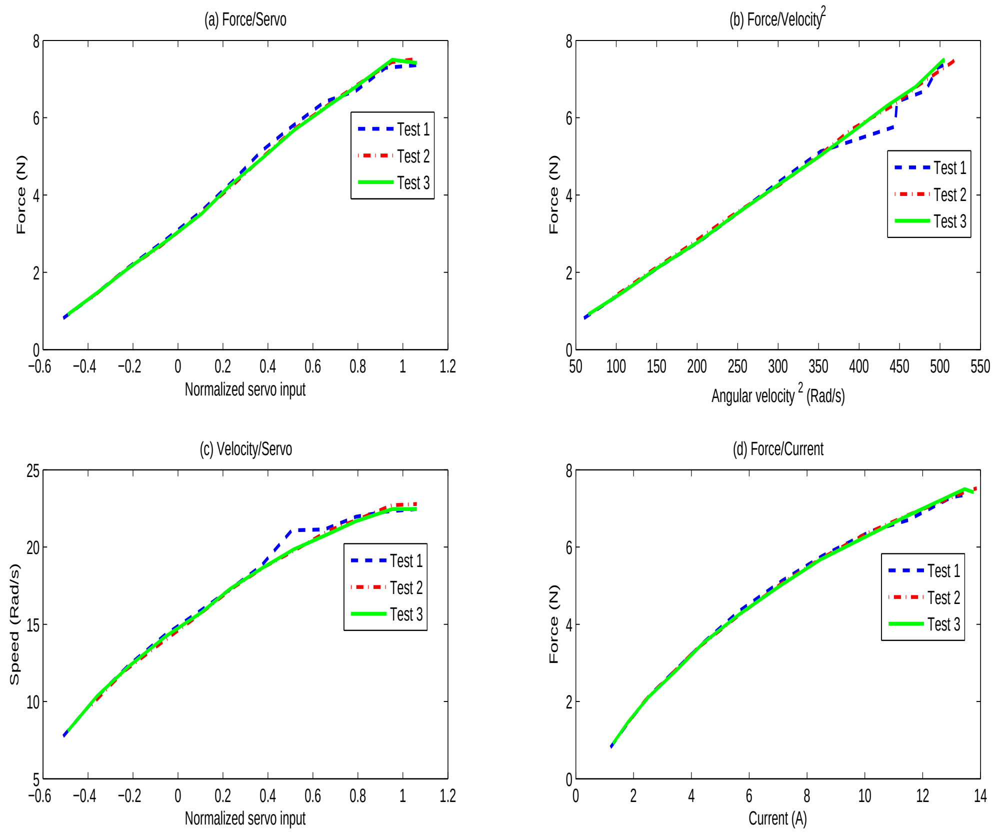


Figure 3.6: Rotor-Propeller thrust experiment's collected data. (a) servo input to rotor thrust, (b) square of propeller angular velocity to rotor thrust, (c) servo input to propeller angular velocity, (d) consumption current to rotor thrust.

where K_f is the constant thrust lift coefficient. To obtain the time constant τ_f, a square wave is applied to the motor input and the corresponding thrust response is observed by force measurement unit. Meanwhile, the voltage of the battery as the power source, is monitored by a oscilloscope. Since, the force measurement unit only shows the steady thrust, the time constant is estimated from battery voltage drop, which depends on the rotor current and correspondingly output thrust. The rotor dynamic coefficients are obtained based on the least square polynomial curve fitting. We used Matlab software to perform the signals curve fitting which for the given polynomial $p(a) = p_1 a^n + p_2 a^{n-1} + \cdots + p_n a + p_{n+1}$, uses the following formula (3.20) for estimation [44].

$$\hat{a} = \frac{a - \mu_1(a)}{\mu_2(a)}, \tag{3.20}$$

where, μ_1 is the average of vector a, and μ_2 standard derivation of vector a, and $\hat{a}$ is the estimated polynomial coefficients. Figure 3.7- 3.9 show the curve fitting results.

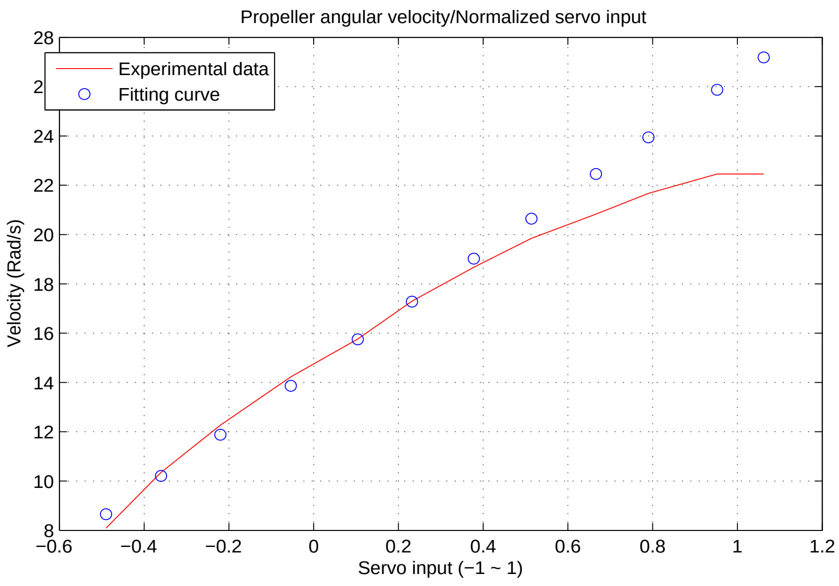


Figure 3.7: Rotor rotational velocity coefficient (K_v).

All the parameters identified by static experiments are listed in Table 3.2.

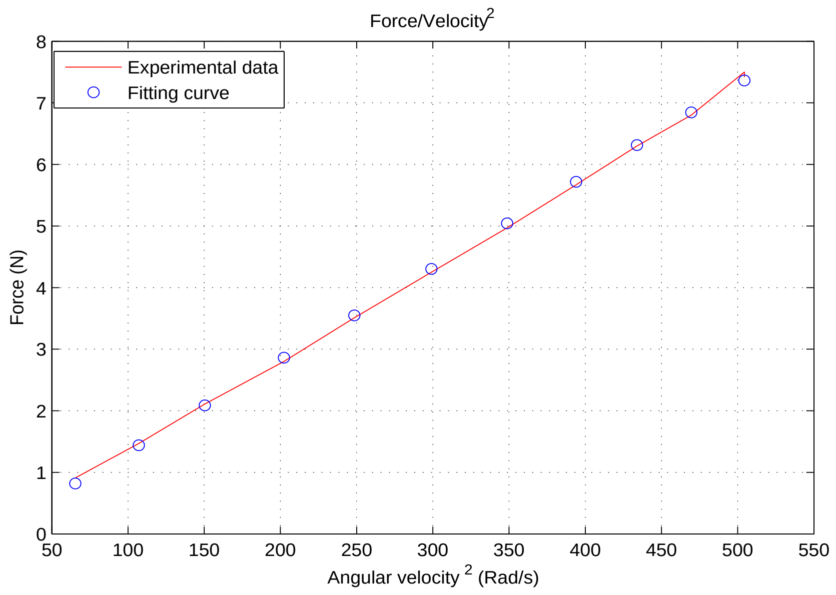


Figure 3.8: Rotor thrust to square of angular velocity coefficient (K_Ω).

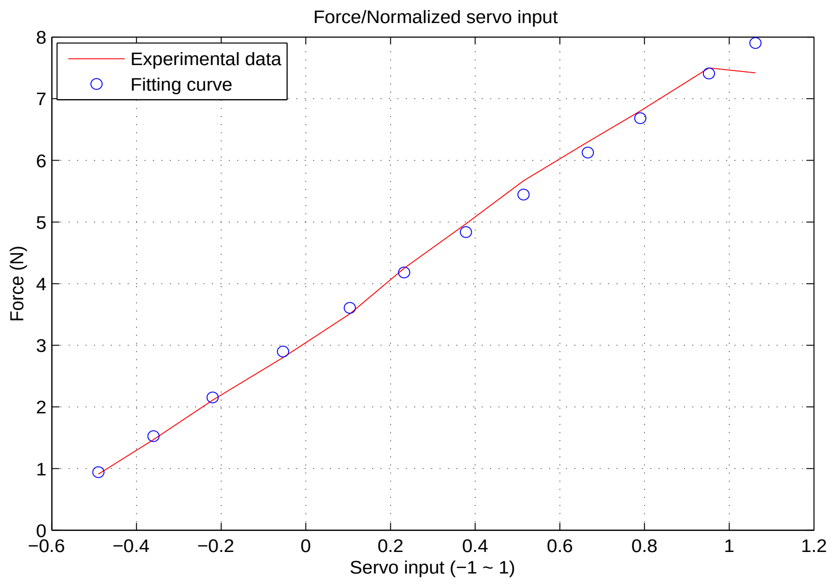


Figure 3.9: Rotor thrust lift coefficient (K_f).

3.3.2 Dynamic Experiments

In this part, we aim to use the flight experiment data to identify the remaining unknown model parameters. The obtained parameters from the static experiments can also be tuned or validated through this method. In this approach the pilot will be asked to excite the requested dynamic of platform in a certain way. Meanwhile, the avionic board through the INS system observes the response of UAV and records the sensor data on the memory card. We will then use MATLAB time-domain identification toolbox (IDENT) to analyze the collected data. The implementation procedure of this approach is described in the following steps.

3.3.2.1 Data collection

It is clear that the dynamic input-output data is necessary for model identification. However, not all the signals are measurable. For our platform, the model states and their measurability status are given in Table 3.3, and the recorded signals places are marked in Figure 3.2. Besides the model states, recording RC receiver outputs $U_{rc} = (U_a, U_e, U_{th}, U_r)$ could also be useful either when the gyro is in the control loop or as a input source for gyro model identification. For more details see Section 3.4. For non-measurable variables highlighted in Table 3.3, it is possible to observe them based on the mathematical relationships expressed in (3.1) and (3.2). Ultimately, all the dynamic states and inputs data for model identification are available.

Variable	Physical expression	Unit	Direct measurablity
x	Position in inertial frame x-axis	m	Yes
y	Position in inertial frame y-axis	m	Yes
z	Position in inertial frame z-axis	m	Yes
u	Velocity in body frame x-axis	m/s	No
v	Velocity in body frame y-axis	m/s	No
w	Velocity in body frame z-axis	m/s	No
ϕ	Roll angle	rad	Yes
θ	Pitch angle	rad	Yes
ψ	Yaw angle	rad	Yes
p	Roll angle rate	rad/s	Yes
q	Pitch angle rate	rad/s	Yes
r	Yaw angle rate	rad/s	Yes
acx_b	Body acceleration in in body frame x-axis	m / s^2	Yes
acy_b	Body acceleration in in body frame y-axis	m / s^2	Yes
acz_b	Body acceleration in in body frame z-axis	m / s^2	Yes
M$_i$	Normalized rotor$_i$ input $(-1 \sim 1)$	NA	Yes
u$_a$	Normalized aileron servo input $(-1 \sim 1)$	NA	Yes
u$_e$	Normalized elevator servo input $(-1 \sim 1)$	NA	Yes
u$_{th}$	Normalized throttle servo input $(-1 \sim 1)$	NA	Yes
u$_r$	Normalized rudder servo input $(-1 \sim 1)$	NA	Yes

Table 3.3: Dynamic states' and inputs' description, and measurability status.

3.3.2.2 Input signal

Based on the recommendation on the perturbation input signal in [39], a chirp (sweep frequency) signal is chosen. In fact, since chirp signal contains a variety of frequency components, it is an appropriate input signal for model identification or validation purposes. A typical linear chirp signal is demonstrated in Figure 3.10 and mathematically expressed as (3.21) [45].

$$u_{chirp}(t) = A_{chirp}\sin\left[2\pi(f_0 + \frac{k_{chirp}}{2}t)t\right], \tag{3.21}$$

where A_{chirp} is chirp signal amplitude, f_0 is its initial frequency, and k_{chirp} is the rate of frequency increase.

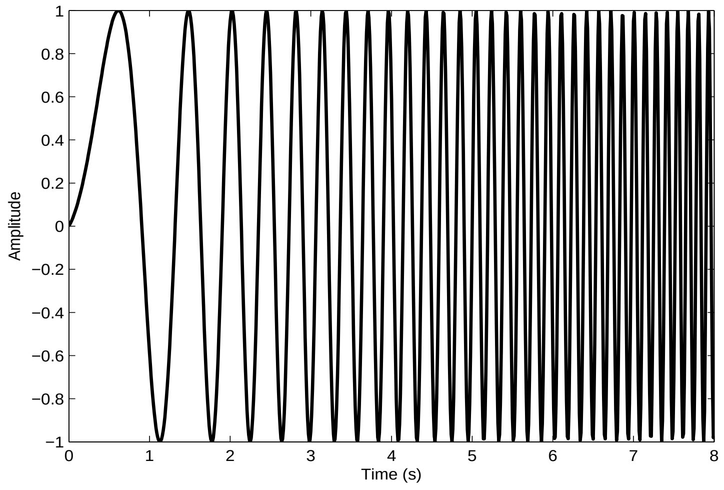


Figure 3.10: A typical chirp signal.

In order to appropriately excite the desired dynamic, selecting the chirp frequency range is a crucial step. Furthermore, since this signal is actually applied manually by the pilot, preparing some simulation based flight tests for training purpose is highly recommended. The initial frequency is preferred to be as low as possible however, it depends on pilot's sight and maneuverability. The maximum frequency should be chosen to avoid exciting undesired dynamic nonlinearities, mechanical vibration, and actuators rate constrains. It is recommended that in each flight experiment, the pilot initially apply two sinusoid long period inputs to the desired channel. Basically, this is to ensure that the low frequency is completely perturbed [39]. Thereafter, the input frequency should be increased smoothly to the desired maximum frequency. It is worth mentioning that the input amplitude does not necessary have to be constant, typically keeping the variation in the range of $\pm 10 - 20\%$ is acceptable [46].

3.3.2.3 Trim value

For any further analysis on the flight data, trim values of the states and inputs are necessary. In fact, the quadrotor is modeled around the trim values which physically is the quadrotor states in a hovering condition and the corresponding applied input level. In order to practicably find the trim values, we asked the pilot to keep the quadrotor in a hovering mode and adjust the joystick trim inputs to achieve a good hovering flight with minimum input correction. Meanwhile, the onboard system records all the measurable signals. After this experiment, the data is analyzed and a period of best hovering flight is extracted; then averaging the data in this period gives the trim values. The flight data are shown in Figures 3.11 - 3.15. As it is depicted in the figures, the attitude angle and angular rate are quite close to zero. The small fluctuation in the body velocity and acceleration is due to external disturbance and existence of wind. Please note that, even though the inputs' trim $(u_{a0}, u_{e0}, u_{th0}, u_{r0})$, is mathematically expected to have zero inputs for hovering, due to the unbalance rotors' thrust and payload, and also non-symmetric mechanical structure, a small inputs bias is necessary. Hence, in this period, although the pilot almost did not correct the quadrotor, it is still in a good hovering condition which provides a reliable set of flight data to find the trim values. The obtained trims are presented in Table 3.4. From the control point of view, these trim values are correspond to zero controller outputs in autonomous flight. Furthermore, in the model identification processes, the required parameters will be identified around the obtained trim levels.

Variable	Trim values in hover condition	Unit
u_{a0}, u_{e0}, u_{th0}, u_{r0}	0.0660, 0.0660, 0.1800, -0.0130	NA
u_0, v_0, w_0	-0.0739,-0.1759,-0.5610	m/s
ϕ_0, θ_0, ψ_0	-0.0381,0.0035, -0.0288	rad
p_0, q_0, r_0	0.00092,0.00095,-0.0023	rad/s
$acx_{b0}, acy_{b0}, acz_{b0}$	-0.0023,-0.0024,-0.1139	m/s^2

Table 3.4: States and inputs trim values at hover.

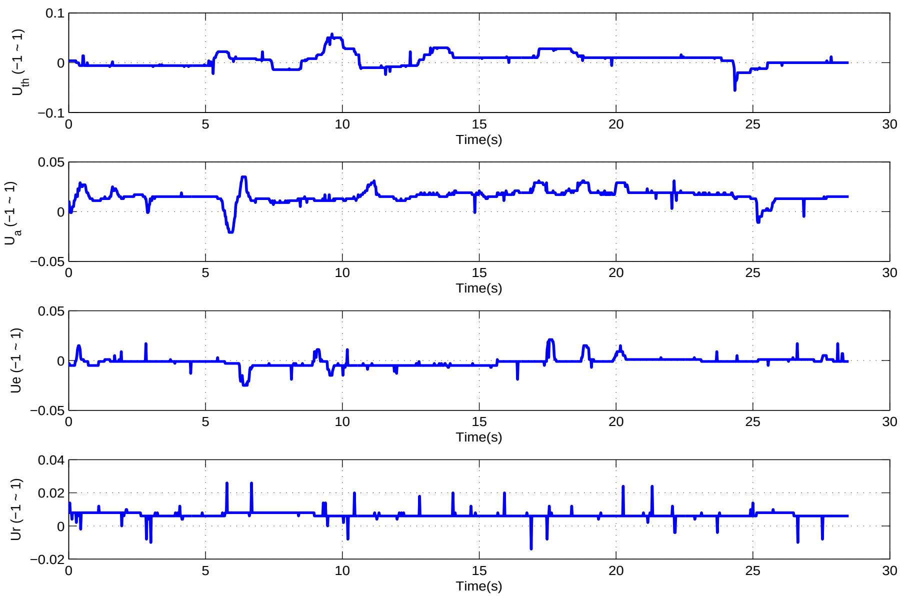


Figure 3.11: Control input data in trimmed flight.

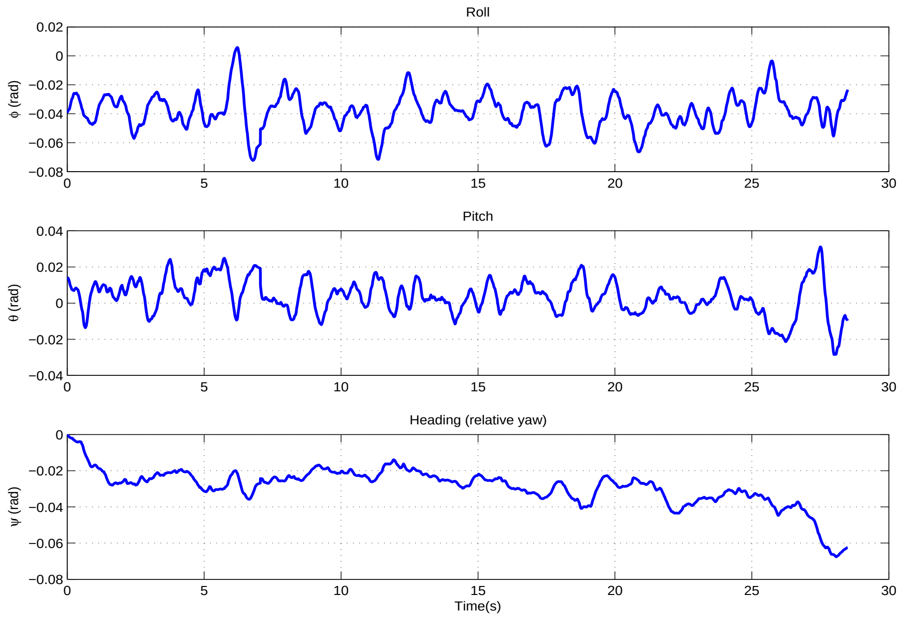


Figure 3.12: Attitude angle data in trimmed flight.

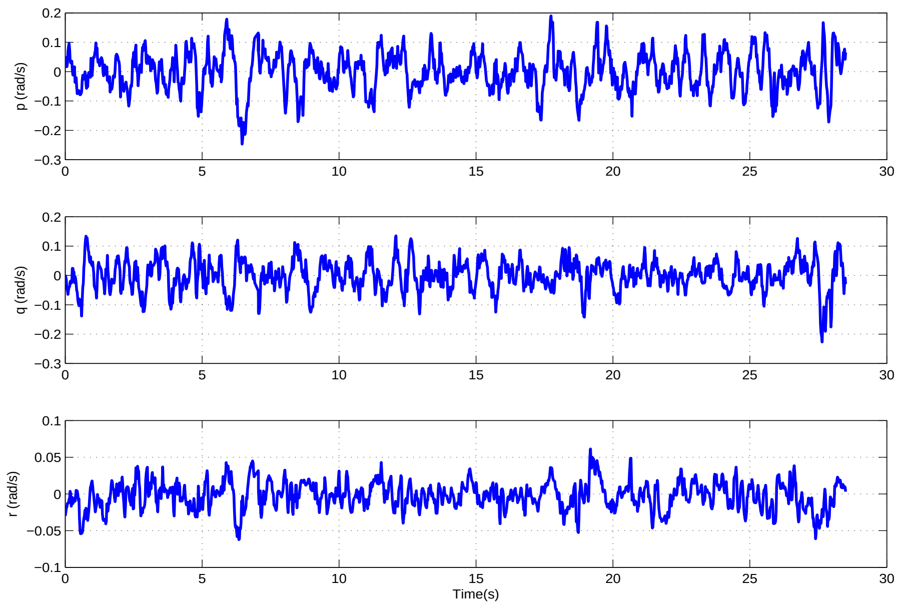


Figure 3.13: Attitude angular ratio data in trimmed flight.

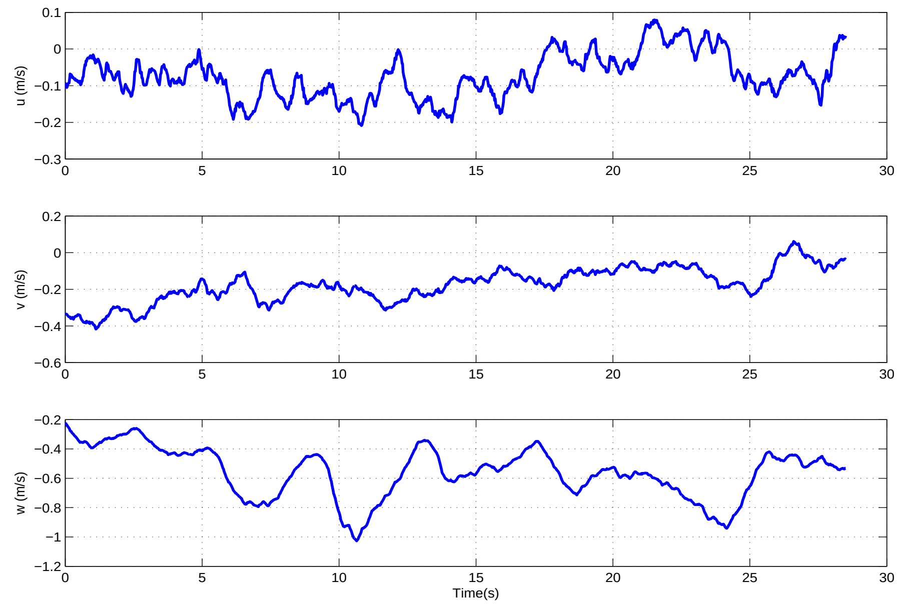


Figure 3.14: Body velocity data in trimmed flight.

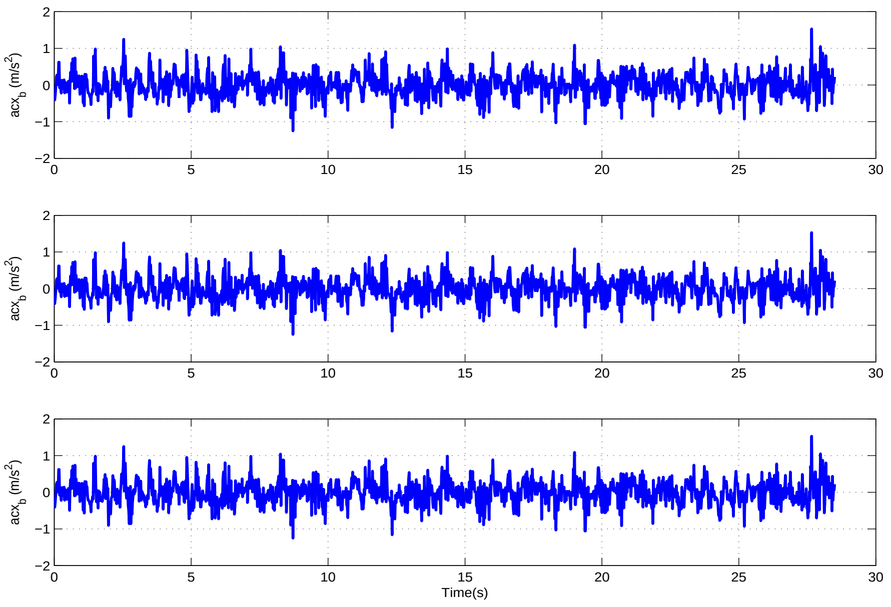


Figure 3.15: Body acceleration data in trimmed flight.

3.3.2.4 Flight test

During the flight test, the quadrotor is expected to exhibit abnormal behavior due to the chirp input and its nonlinear dynamics. Therefore, it would be better to do this experiment in a wide and open area. Moreover, since the quadrotor in the hovering will be considered as the model equilibrium point, it should be perturbed around hovering condition. Hence, firstly the quadrotor hovers at an appropriate point within eyesight, then the pilot excites only one desired channel as a chirp signal style. Meanwhile the onboard system is recording all the desired data. For each channel, the perturbation is repeated several times to ensure enough qualified data is recorded. A throttle perturbation experiment as an example, is presented in Figures 3.16. In this flight experiment the pilot manually has applied a chirp input by the joystick throttle channel. However, as it can be observed from the figure, the rate at which the frequency increases is not perfectly linear which is unavoidable. Between each perturbation, the pilot is requested to keep the quadrotor in a hovering flight for several seconds. This part later would help to easily identify each set of data, and also verify the trim values. Furthermore, because of dynamic coupling and disturbances, when one channel is perturbed, the platform may drift in different directions or orientations. Yet, the pilot is asked to do not correct the quadrotor back, unless in emergency situations. This guarantees in each flight experiment, the recorded outputs are the result of one perturbed input. This method is quite important, specifically when it is desired to decoupled the dynamic sub-models and identified them separately. However, in the gap of each perturbation experiment, the pilot has to bring back the quadrotor in an appropriated position and make it ready for the next experiment part, which is unavoidable. The raw recorded data typically has to be filtered and cut off before identification process. Figure 3.16 shows how the pilot changed the inputs.

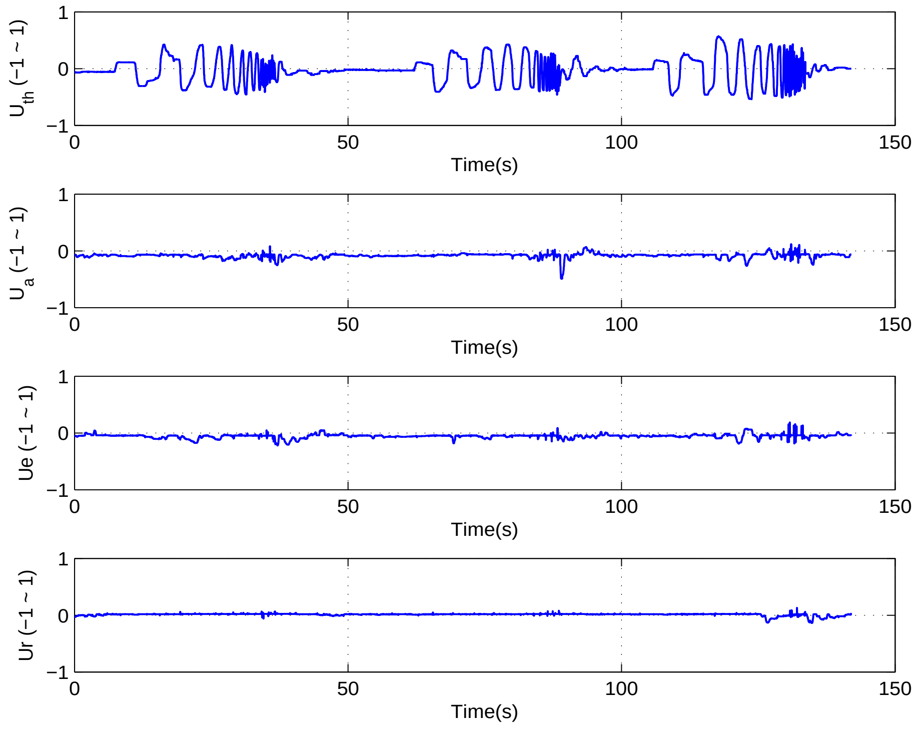


Figure 3.16: Input signal in the throttle channel perturbation.

3.3.2.5 Preprocess raw data

In order to make the logged data useable for model identification, data preprocessing is required which is addressed as the followings.

- **Data range selection**: In order to reliably use model identification methods, it is important to select an appropriate range of data set. The chosen data set should be rich in aspects of frequency of perturbation and contains a full range of input.

- **Low pass filtering**: Mainly because of sensor measurement noise and vibration cause by the rotors, the data set should be passed through a low pass filter. In this case using same filter for the input and output data is necessary to avoid inserting extra time delay.
- **Data detrending**: Basically, pilot may start the experiment from non-trimmed condition or due to hardware changes such as using different battery models, collected data may drift from trim values. If so, the data should be corrected such that the affect of this phenomena on the fidelity of the parameter identification is minimized [46].

3.3.2.6 Parameter identification

After preprocessing the raw data, the time domain data set is ready for model identification. We use the MATLAB IDENT toolbox [47] software and prediction error method (PEM) as an identification algorithm. A typical accepted error for the helicopter model identification can be consider somewhere around the corresponding output sensor accuracy [48]. For instant, a few degrees error for attitude dynamics model is practically acceptable. The selected parameters to be identified through the time domain approach are given in Table 3.5.

The rotor thrust dynamic can be identified based on the vertical body accel-

Parameters	Description
d	Rotor induced torque coefficient
K_f	Rotor thrust lift coefficients
τ_f	Rotor thrust model time constant

Table 3.5: Unknowns model parameter and their description for dynamic model identification.

eration. However, through the dynamic experiment, individual rotor thrust is not directly measurable. Yet, if the body vertical axis (z) acceleration (3.22) is perturbed in hovering condition, it can be assumed all the rotors thrust are approximately equal and throttle input is equivalently distributed on the rotors' input. Hence, an individual rotor thrust can be observed explicitly by the help of (3.15) and ignoring the drag effect which is therefore can be written as the following

$$\sum_{i}^{4} f_i = m\left(-\dot{w} + qu - pv + g\cos\theta\cos\phi\right). \qquad (3.22)$$

Figure 3.17 demonstrates one set of applied signal to the throttle channel. As it can be seen, the other channels almost did not perturb and only throttle is excited to avoid the dynamic coupling effect. The attitude data is demonstrated in Figure 3.18. As the figure shows, although the attitude angle is not perfectly zero, this small fluctuation is avoidable and so the quadrotor can be acceptably assumed to be in a hovering condition. The perturbation of body vertical acceleration in response of the throttle channel also is illustrated in Figure 3.19. Individual rotor thrust data is then can be obtained from (3.22).

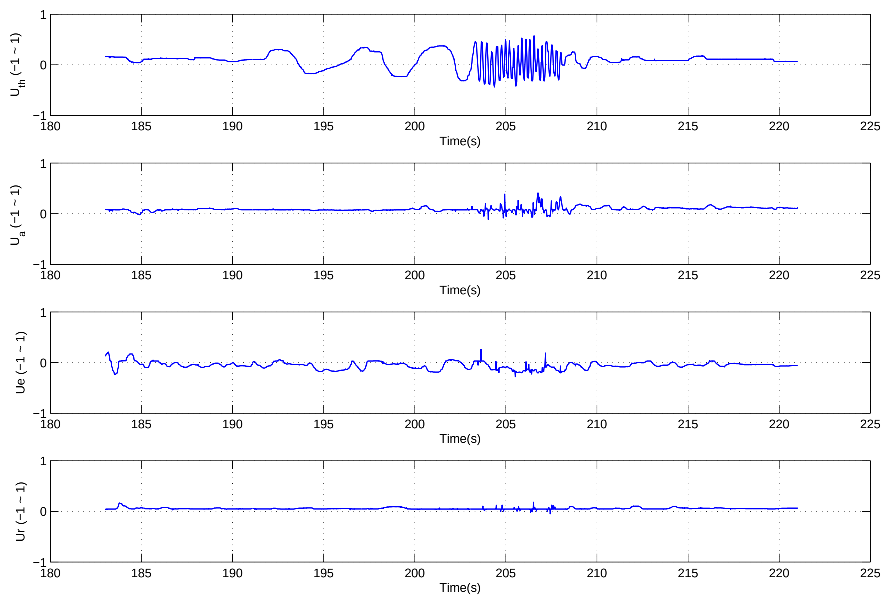


Figure 3.17: Input signal in the throttle channel perturbation.

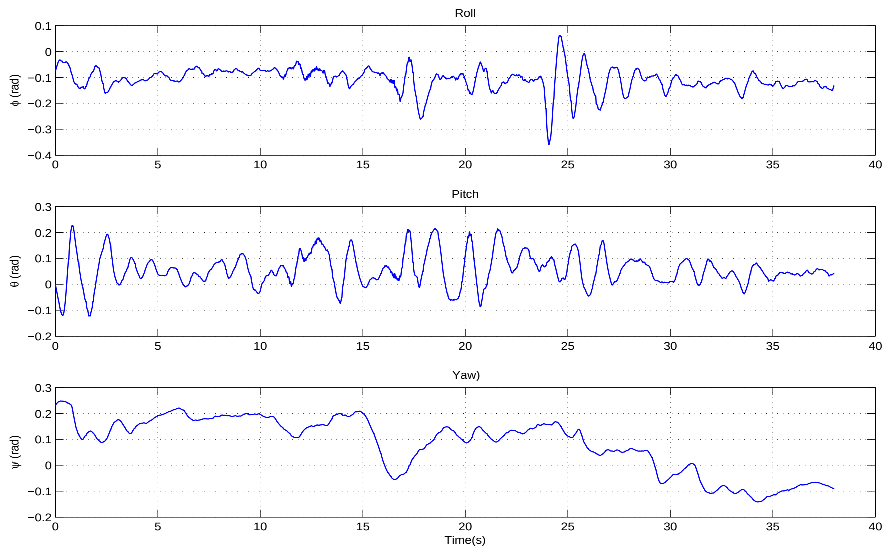


Figure 3.18: Euler angles in the throttle channel perturbation.

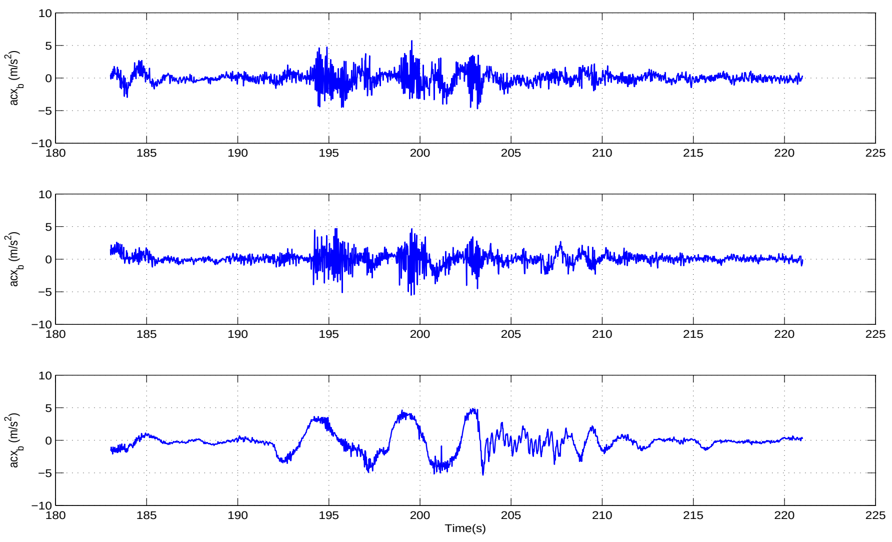


Figure 3.19: Inertial acceleration in the throttle channel perturbation.

After data detrending based on the signal trim values, the input-output data source is ready to identify the rotor thrust model (see Figure 3.20).

The data is then fed into the MATLAB IDENT toolbox with 20 ms sampling rate. The model structure is assigned as (3.19) and initial guess values are chosen from ground static experiment results. Another set of throttle perturbation is also used for model validation. Figure 3.21 shows a comparison between the identified rotor thrust model from the ground static tests and dynamic experiment based on flight data. As this figure illustrates, two estimated models are quite identical and can closely track the observed output force which validates the estimated rotor thrust model. Please note that both approaches are using the same model structure and order.

The induced torque coefficient d, can be estimated based on yaw dynamic. From

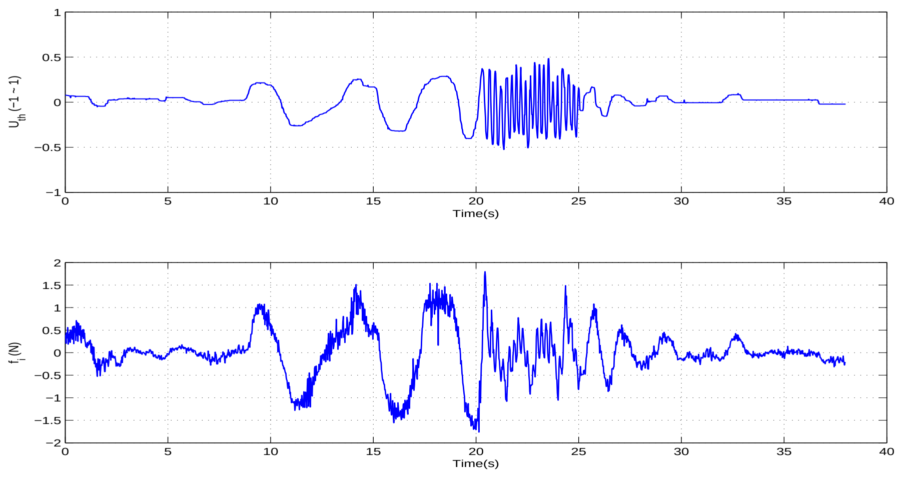


Figure 3.20: Input-output data set for rotor thrust dynamic identification.

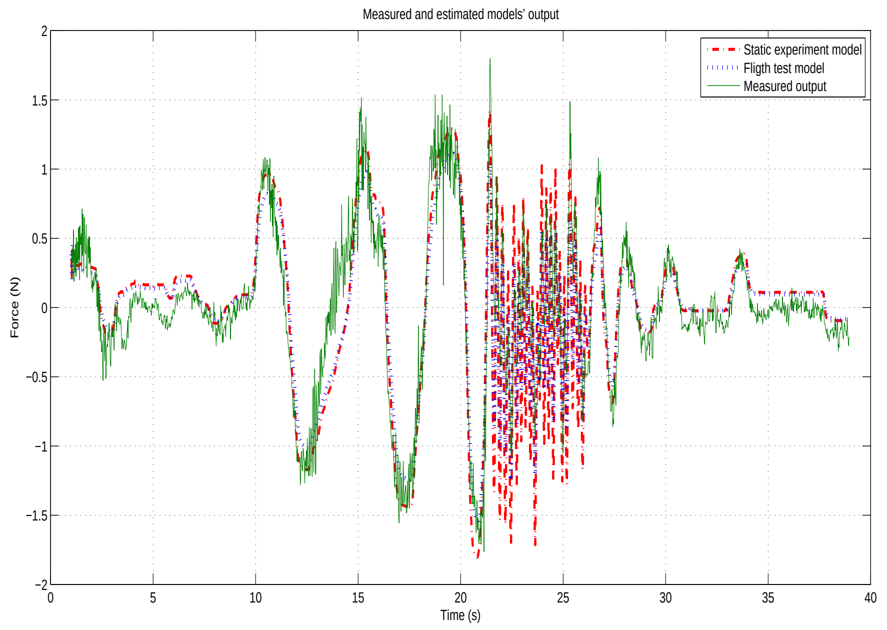


Figure 3.21: Rotor thrust dynamic model comparison.

(3.10) and (3.11), the yaw ratio dynamics with respect to motors voltage input can be written as the following

$$\begin{cases} \dot{r} = \frac{J_{xx}-J_{yy}}{J_{zz}}pq + \frac{d}{J_{zz}}\Omega \\ \Omega(s) = \frac{K_v}{1+\tau_f s}\sum\limits_{i=1}^{4}(-1)^{i+1}v_i \end{cases} . \tag{3.23}$$

In a similar manner, a flight experiment is conducted to collect the necessary data to identify the yaw dynamic parameters. The rudder channel is excited (see Figure 3.10 for illustration), and the sensors data are recorded correspondingly. Figures 3.22 and 3.23 show the applied inputs and attitude angles ratio in this flight experiment. Due to the small attitude dynamics coupling, and unavoidable pilot correction applied on the other channels, we observed small changes on the pitch and roll outputs which are negligible. Based on the model structure (3.23) and collected data, MATLAB identified the induced torque coefficient d. The estimated model output is compared with real measurement and the result is demonstrated in Figure 3.24.

As a resul,t all the required model parameters are numerically identified which are presented in Table 3.6.

Parameter	Value	Description
K_f	3.8738	Rotor thrust lift coefficient
τ_f	0.095583	Rotor thrust model time constant
d	0.013	Induced torque coeffcient

Table 3.6: Numerical value of identified parameters from flight data.

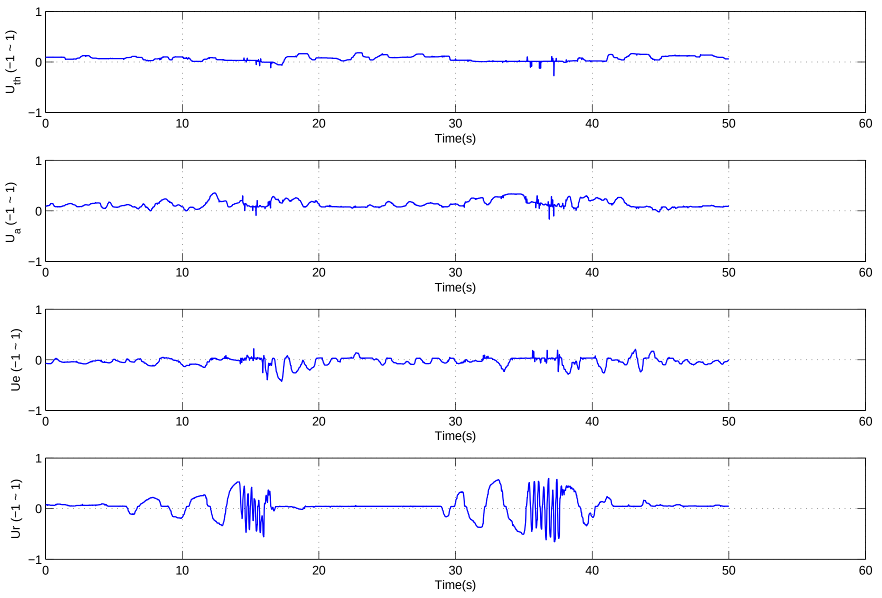


Figure 3.22: Input signal in the rudder channel perturbation.

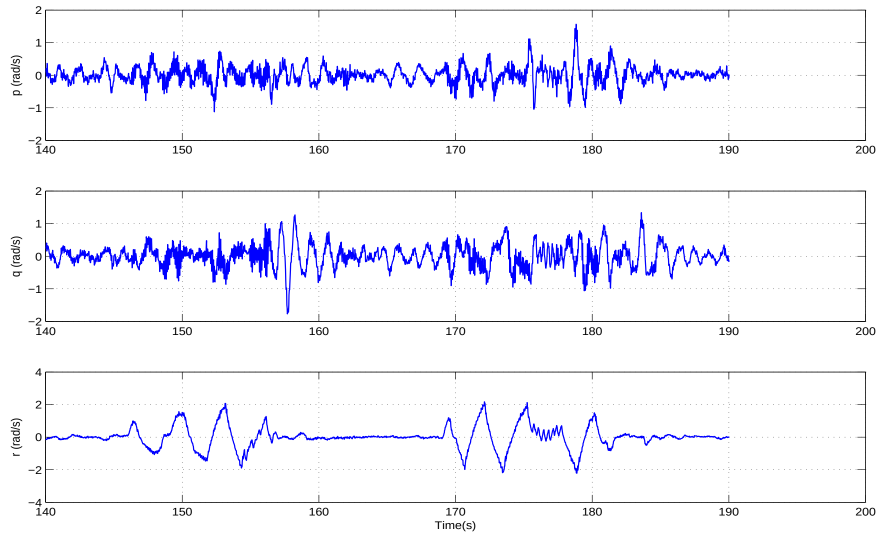


Figure 3.23: Euler angles in the rudder channel perturbation.

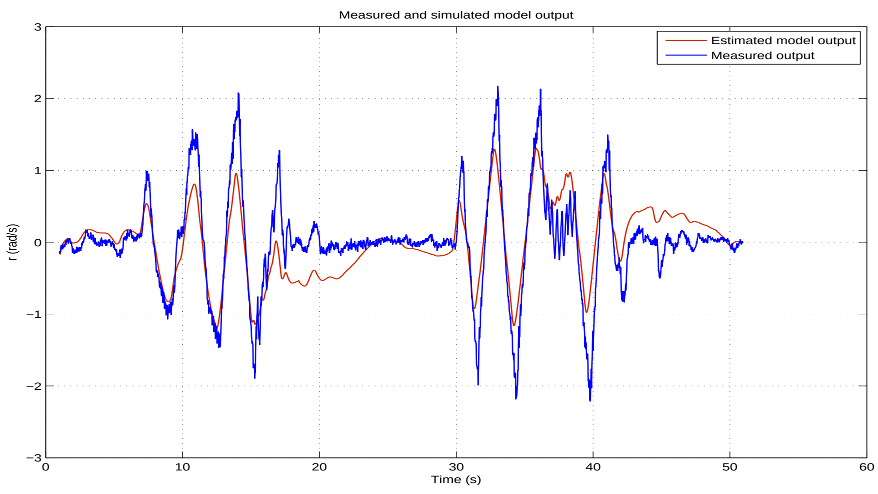


Figure 3.24: Yaw ratio dynamic estimated and measured outputs.

3.4 Quadrotor Model with Gyro in the Control Loop

One way to control the quadrotor is to use the gyro in both manual and autopilot flight modes. In this approach, the autopilot system acts as a pilot and apply the control outputs to the gyro instead of directly controlling the motors. As Figure 3.25 illustrates, the gyro gets the control signals from the autopilot and accordingly distributes them to the motors. The gyro itself is also able to control the quadrotor attitude angles or angular ratios. In fact, adding the gyro in the automatic flight control loop makes the overall controlling simpler, since the gyro can help to stabilize the attitude. However, it arises extra dynamical identification which is modeling the closed loop attitude dynamics with having the gyro as a attitude stabilizer in the control loop.

Furthermore, from the safety aspect, this structure can be helpful specifically during preliminary autopilot tests. As it is mentioned previously, in this configuration the gyro is in the control loop in the both manual and auto flight modes. Therefore, when the autopilot system goes wrong and the pilot has to switch back to manual mode, the rotors inputs do not observe this hard switch, since the gyro has already been driving the motors. Now consider the configuration when the gyro is not in the autonomous control loop. In the same scenario when the pilot changes the flight mode to manual, the gyro outputs are sharply connected to the motors' driver. Noted that, the gyro, in spite of the flight mode, is always getting feedback from the platform orientation and trying to control it, so its output is always changing according to the quadrotor status. This sharp switch potentially can cause dangerous behavior because the gyro outputs are unknown at the moment of bringing back the gyro in the control loop.

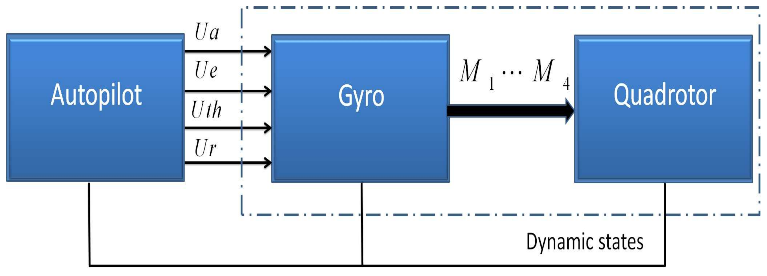


Figure 3.25: Gyro in the loop control structure.

The gyro used in our system has two working configurations: hover mode and cruise mode. In the hover mode, the gyro, based on the control inputs, stabilizes quadrotor attitude. The vehicle with gyro in hover mode, is inherently stable and is more suitable for hover flight. As hover mode mainly is for hovering flight, the gyro in cruise mode is more suitable for cruise flight. Based on our flight tests, in the cruise mode, the gyro is controlling attitude angular ratio and the quadrotor is very sensitive to the input signals. We have conducted several flight tests to observe the gyro characteristics and also to model the closed loop attitude dynamics. In the following sections, we will identify the model of quadrotor attitude dynamics when the gyro is in the control loop. The gyro to some extent controls the attitude states, hence, we will assume the attitude dynamic are decoupled. Moreover, it has been verified that the motor inputs are linearly proportional to the throttle channel, which ensures that the gyro does not affect the rotor thrust dynamic.

3.4.1 Gyro : Hover Mode

We set the gyro in hover mode and ask the pilot to perturbed all the channels around hovering condition, similar to Section 3.3.2.2. The joystick signals together with attitude angles and ratio have been analyzed in this part. It is observed that the gyro in this mode, stabilized angles of pitch and roll, and angular ratio of heading. In order to have an initial model order guess, it could be assumed that, based on the linearized mathematical model, each attitude dynamic can be approximated as a double integrator. Furthermore, the gyro as a feedback on the attitude, acts as a PID controller which is known from its setting. Hence, by the closed loop this system should behave as a third order system with two stable zeros. However, during model identification analysis, it is found that a second-order system can adequately match the model structure. The attitude dynamics model structure are given in

(3.24) - (3.26), and Figures 3.26 to 3.28 illustrate the accuracy of identified models. In this mode, since the gyro directly controls the pitch and roll angles, these attitude dynamics has a considerable delay which has to be taken into the account. It is worth mentioning that, it maybe possible to increase the model order to obtain a perfect output matching, but it will increase the complexity of the controller and correspondingly increase the computational costs. Furthermore, from the control point of view, since the closed loop roll and pitch dynamics are stable, for roll and pitch dynamics, the gyro can take the responsibility of inner-loop tracking control; and as for the heading, the yaw angular ratio dynamic can be considered as a simple linear first-order system with input u_r, given as (3.26)

$$\phi(s) = \frac{K_\phi}{(1+\tau_{\phi 1}s)(1+\tau_{\phi 1}s)} e^{-T_\phi} u_a, \tag{3.24}$$

$$\theta(s) = \frac{K_\theta}{(1+\tau_{\theta 1}s)(1+\tau_{\theta 1}s)} e^{-T_\theta} u_e, \tag{3.25}$$

$$r(s) = \frac{K_r}{(1+\tau_r s)} u_r. \tag{3.26}$$

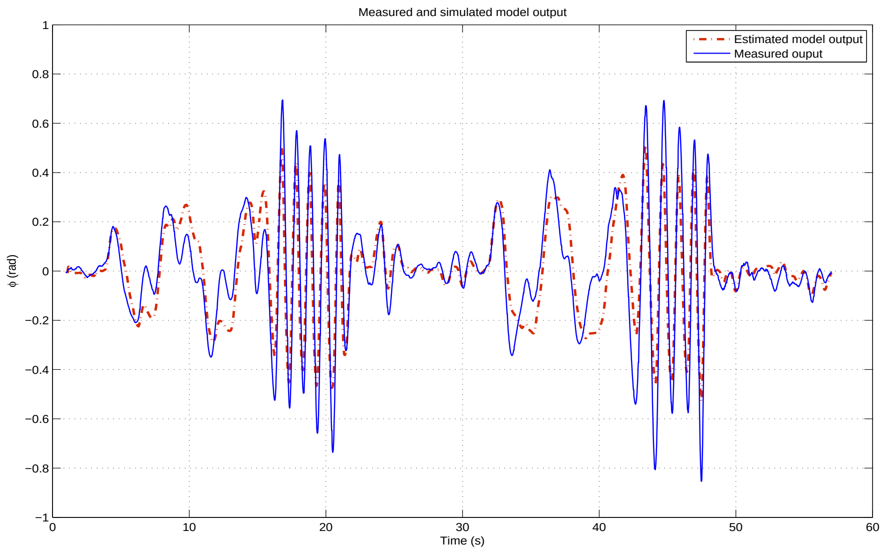


Figure 3.26: Gyro in the hover mode, rolling identified model.

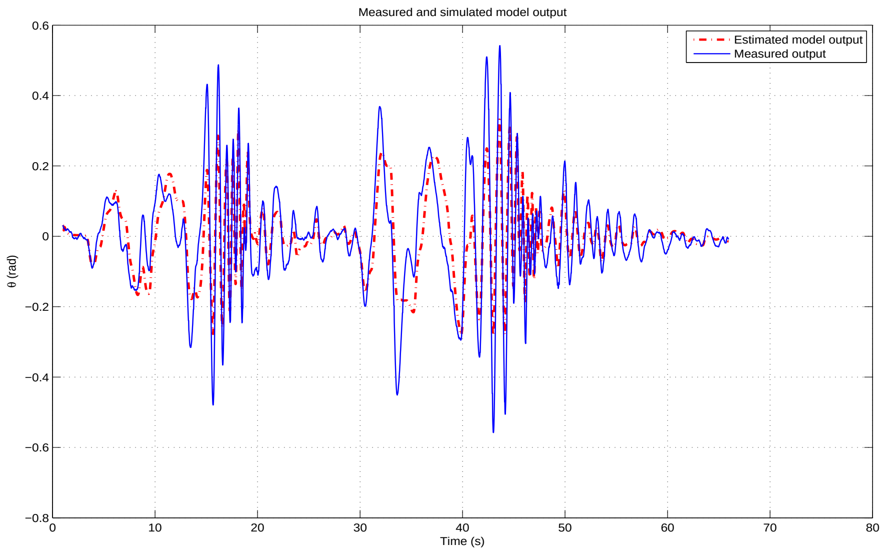


Figure 3.27: Gyro in th hover mode, pitching identified model.

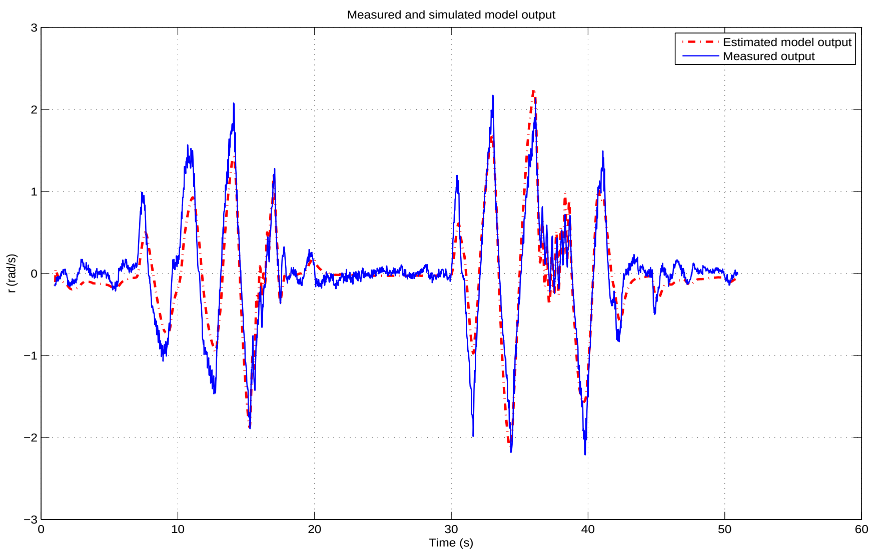


Figure 3.28: Gyro in the hover mode, yaw angle ratio identified model.

3.4.2 Gyro : Cruise Mode

In this mode, the gyro just stabilizes the attitude angular ratio and hence the platform behavior is not as smooth as in hover mode. In a similar flight experiment manner, we have done several flight tests for each individual channel identification. However, for this mode, not only in the hovering flight but also in cruise flight the inputs are perturbed. From the collected flight data, it is observed that, a first-order system can quite perfectly emulate each attitude dynamic. Hence, the attitude angular ratio structures are estimated as presented in (3.27) to (3.29). The heading dynamic basically is the same as gyro in hover mode. With such model structures the unknown parameters are identified. Figures 3.29, 3.30, and 3.31 show the identified model response.

$$p(s) = \frac{K_p}{(1 + \tau_p s)} u_a, \tag{3.27}$$

$$q(s) = \frac{K_q}{(1 + \tau_q s)} u_e, \tag{3.28}$$

$$r(s) = \frac{K_r}{(1 + \tau_r s)} u_r \tag{3.29}$$

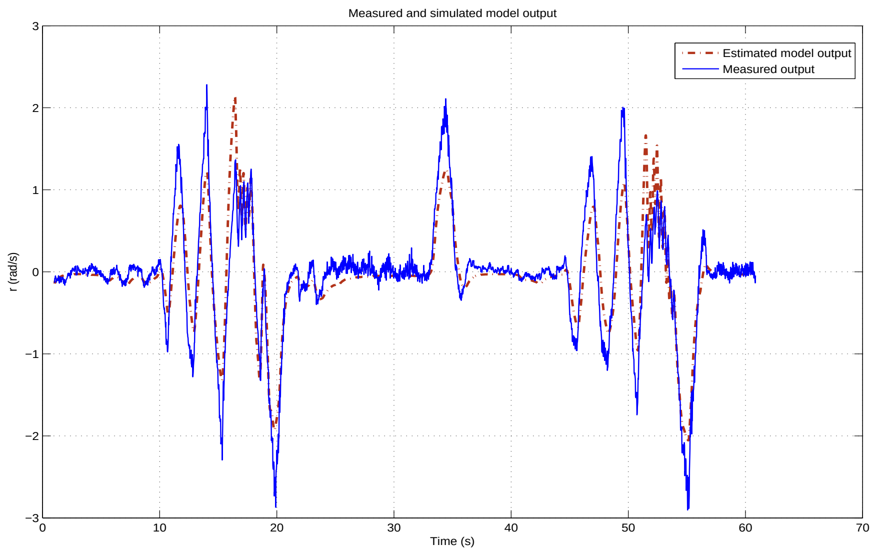


Figure 3.29: Gyro in the cruise mode, roll angle ratio identified model.

The numerical value of identified model parameters for the both operation gyro modes are listed in Table 3.7.

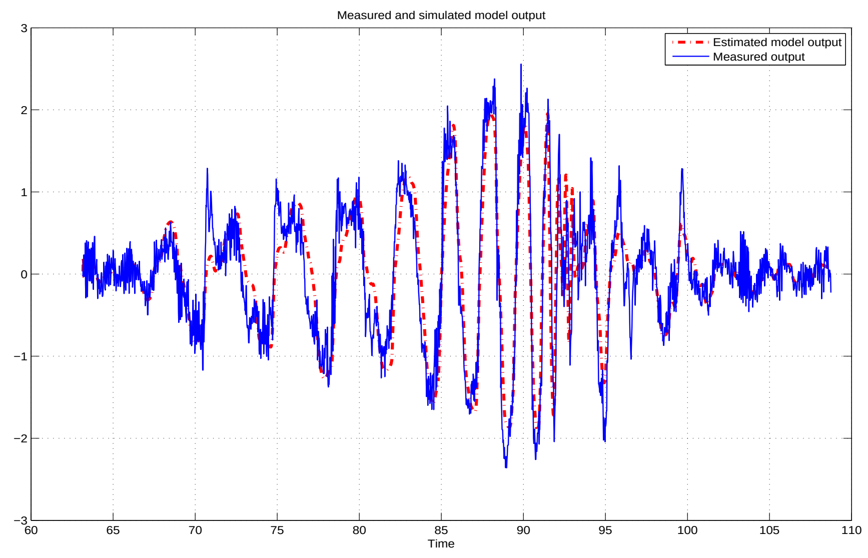


Figure 3.30: Gyro in the cruise mode, pitch angle ratio identified model.

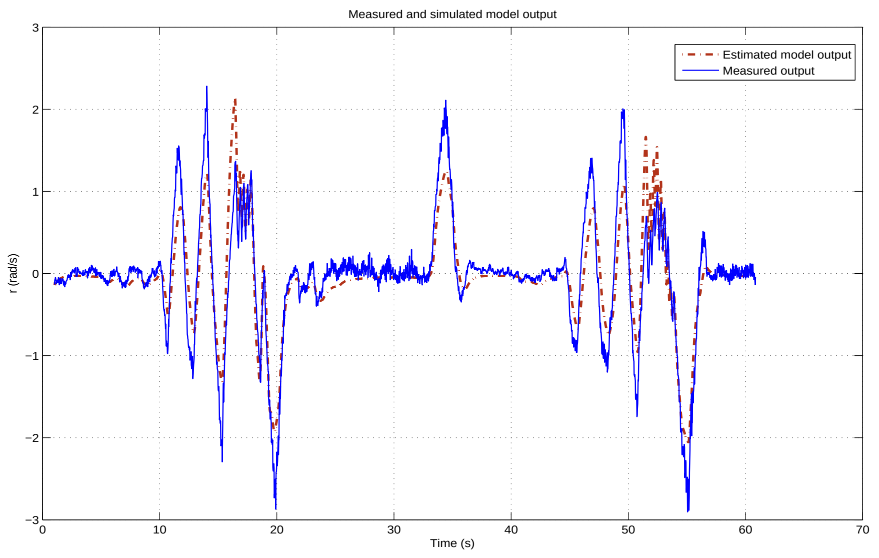


Figure 3.31: Gyro in the cruise mode, yaw angle ratio identified model.

Parameter	Value
K_ϕ,K_θ	-0.8628, -0.59151
K_r,K_p,K_q	-4.8121,-2.3167,-2.4967
$\tau_{\phi1}$,$\tau_{\phi2}$	0.076615,0.076615
$\tau_{\theta1}$,$\tau_{\theta2}$	0.079119,0.079119
τ_r,τ_p,τ_q	0.37029,0.033855,0.046672
T_ϕ,T_θ	0.1,0.1

Table 3.7: The identified parameters of attitude dynamic with gyro in the control loop.

3.5 Model Validation

In this section, it is aimed to verify the fidelity of identified models. We have used the collected data of previous flight tests to verify the obtained numerical model. For perturbation of each input channel, the measured body acceleration and angular rate ratios are compared with the simulated model outputs. In fact, these are the states which directly are obtained from the raw sensor data.

Since for further control tasks, we plan to mainly use the gyro in the loop configuration, the model validation result of this configuration is only presented here. Figures 3.32 - 3.35 illustrate this comparison. As it can be observed from the figures, the identified quadrotor model is able to track the measured states with an acceptable accuracy. Practically and also by considering the size of the UAV, we considered angle ratio error less than $\pm.1$ rad/s, and acceleration error less than $\pm.5$ m/s^2 as an acceptable model accuracy. This model hence, is validated and will be used for further control design and simulation analysis.

3.6 Conclusion

Through this chapter, the nonlinear mathematical model of the quadrotor in cross style was derived. In order to identify the numerical model, the unknown parameters are determined. Considering the nature of each parameter, static and dynamic experiments were designed accordingly. Based on the ground experiments, the rotor thrust dynamic, platform weight and arms' length were obtained. Thereafter, plenty of flight tests were conducted to identify the remaining model parameters and also to verify the ground experiment results. We also considered the configuration that the gyro is in the control loop. In this configuration, the attitude closed loop dynamic in the two gyro settings: hover and cruise mode were identified. At the end, the estimated numerical model was validated based on the collected flight data.

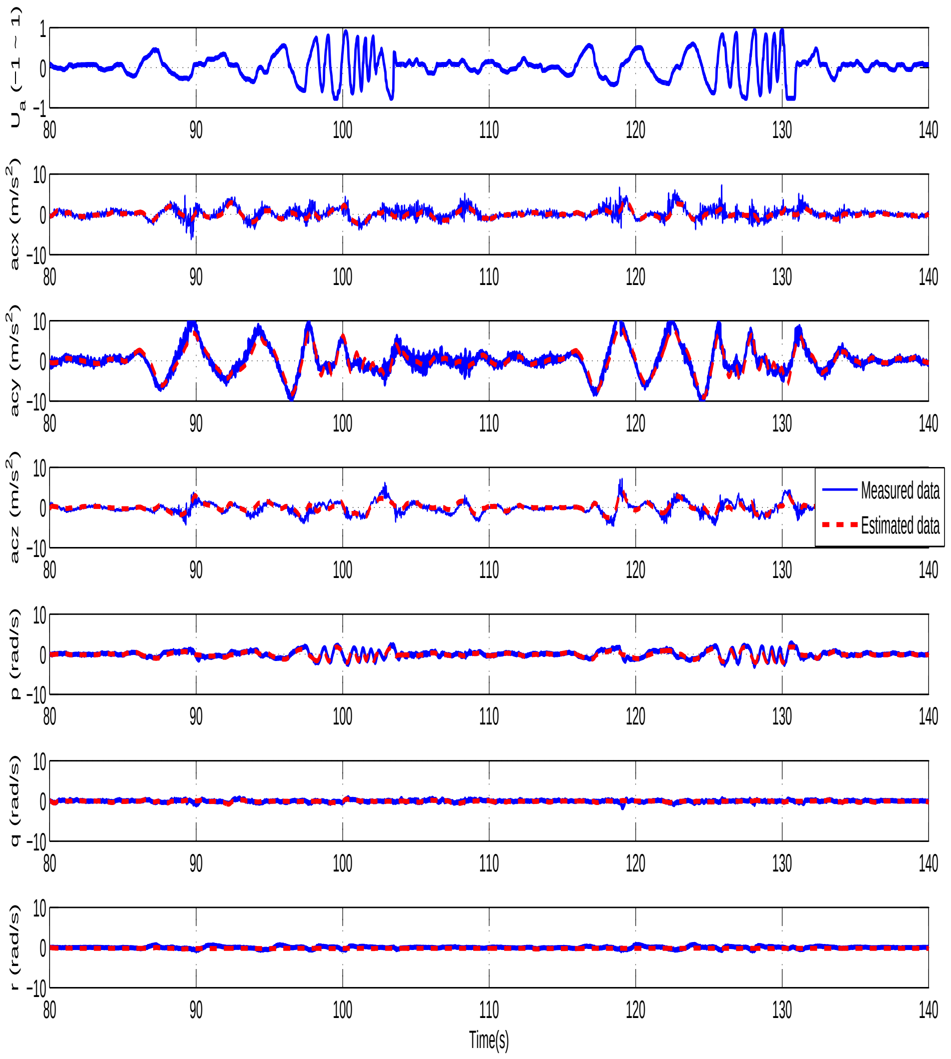


Figure 3.32: Model verification using aileron perturbation.

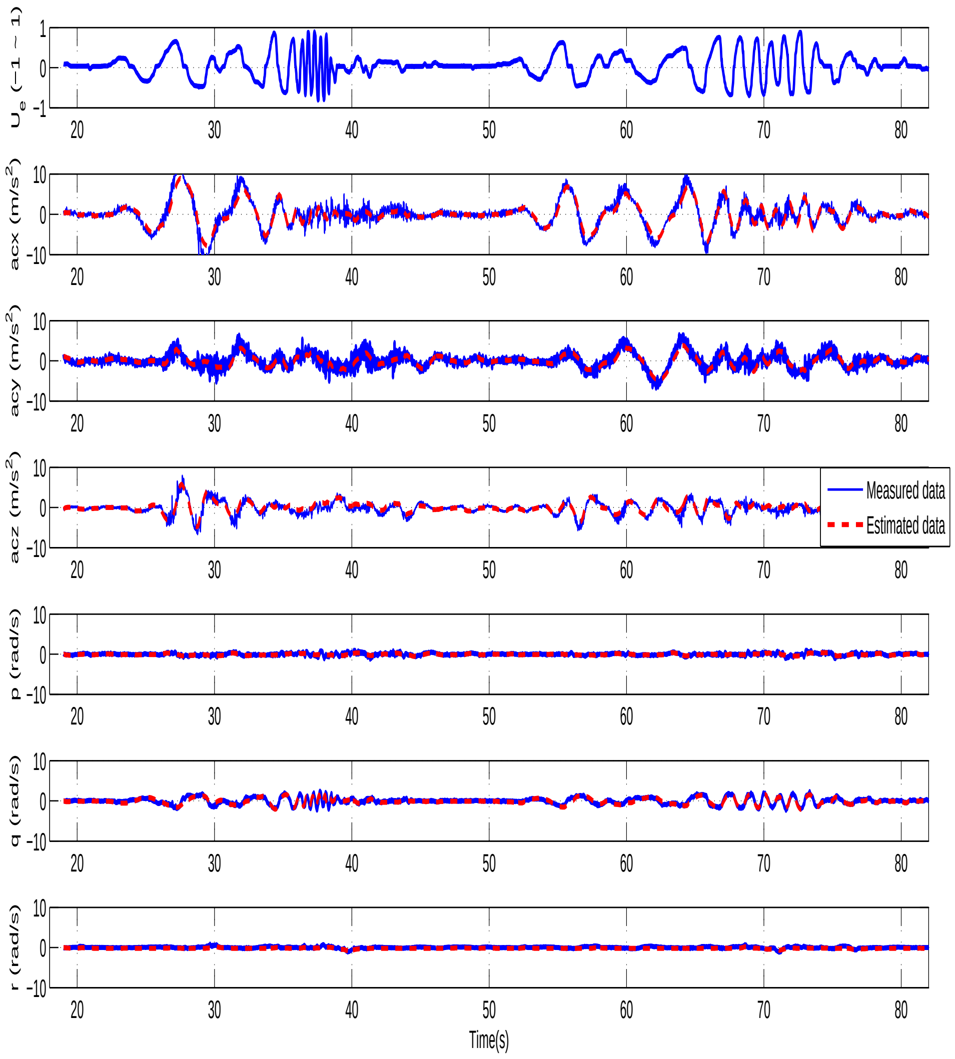


Figure 3.33: Model verification using elevator perturbation.

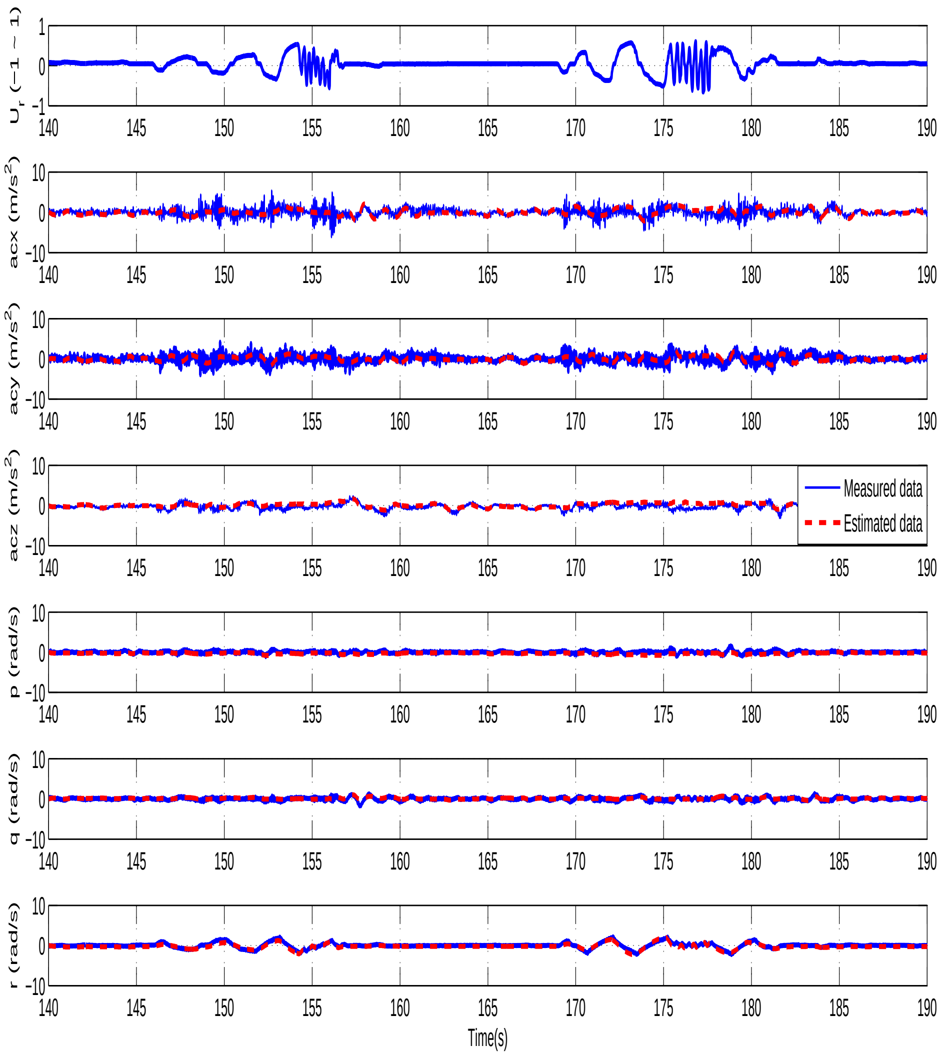


Figure 3.34: Model verification using rudder perturbation.

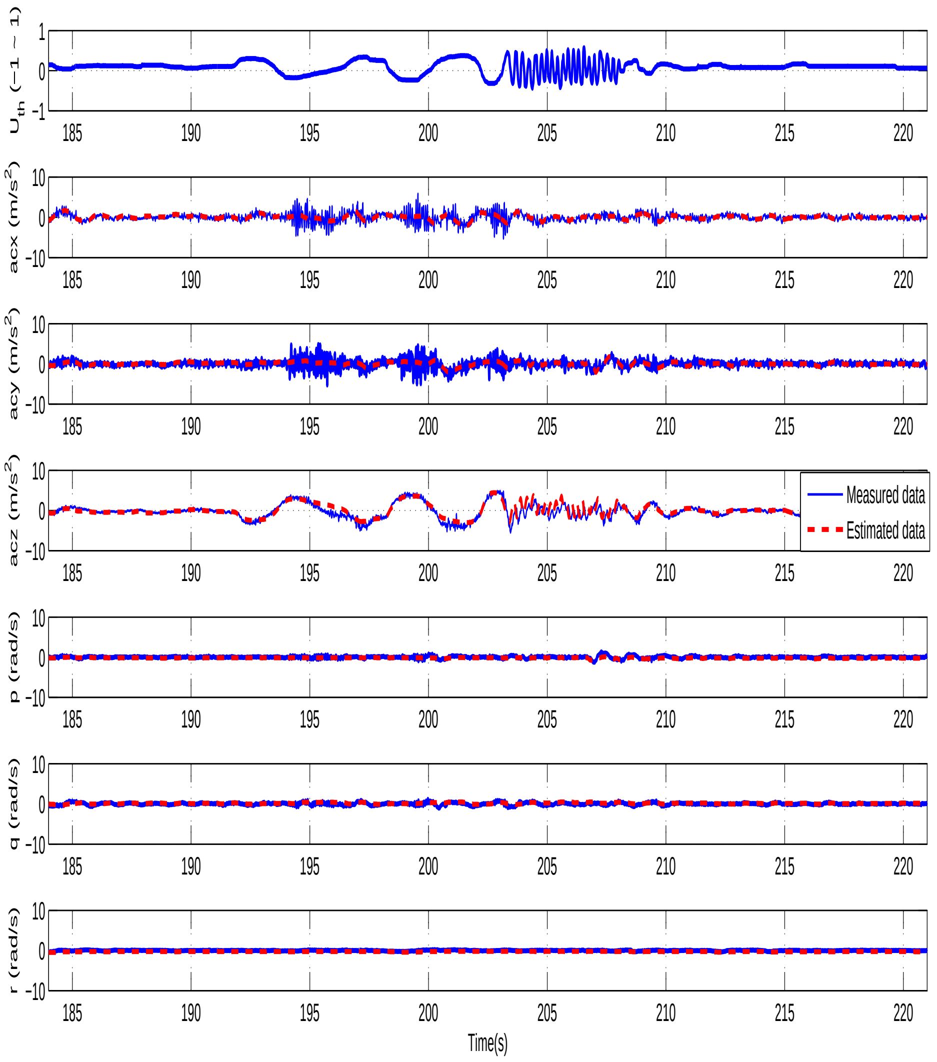


Figure 3.35: Model verification using throttle perturbation.

Chapter 4

Control Design and Implementation

This chapter mainly presents the quadrotor autonomous position hold and attitude tracking implementations. We use the identified quadrotor model described in Chapter 3, to design the attitude and the position controllers. To reduce the complexity of controller design, we will simplify the quadrotor nonlinear dynamic model around hovering flight point and also use the gyro in the control loop configuration. Through this chapter, the performance of both gyro operating modes in different flight scenarios will be evaluated. Furthermore, before real flight tests, the preliminary flight evaluation should be done which will be discuss as hardware-in-the-loop simulations and ground tests. The simulation and experimental results will be presented to demonstrate performance of the proposed control methods.

The remaining contents of this chapter are organized as follows. Section 4.1 shows the model simplification method. We have the same concept in Section 4.2 when the gyro is considered to be in the control loop. The hardware-in-the-loop simulation and preliminary autonomous flight analysis are discussed in Section 4.3. The attitude tracking and position hold results are respectively presented in Sections 4.4 and 4.5. This chapter is finally concluded in Section 4.6.

4.1 Simplified Model

A complete dynamic of the platform contains highly nonlinear terms and coupling factors. However, since we are dealing with a small scale VTOL platform, the operating flight envelope is in the low flight speed and small angle of attack. Therefore, in order to obtain a better insight into the aircraft motion dynamics, it could be reasonable to simplify the derived nonlinear dynamics (3.14) around hovering condition. Hence, if the perturbations from hovering are small, the euler angles (3.16) can be approximated as $\begin{pmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{pmatrix}^T \simeq \begin{pmatrix} p & q & r \end{pmatrix}^T$ and the inertia velocity as $\begin{pmatrix} \ddot{x} & \ddot{y} & \ddot{z} \end{pmatrix}^T \simeq \begin{pmatrix} \dot{u} & \dot{v} & \dot{w} \end{pmatrix}^T$.

In addition, with the aforementioned assumption, the drag force, coriolis terms, and the gravity effect in the horizontal plane are negligible [38]. As a result, the simplified dynamic can be derived as follows.

$$\ddot{x} = \frac{-F}{m}\sin(\theta)\cos(\phi), \tag{4.1}$$

$$\ddot{y} = \frac{F}{m}\sin(\phi), \tag{4.2}$$

$$\ddot{z} = g + \frac{-F}{m}\cos(\theta)\cos(\phi), \tag{4.3}$$

$$\ddot{\phi} = C_1(\dot{\eta}) + \frac{1}{J_{xx}}\tau_{\phi}, \tag{4.4}$$

$$\ddot{\theta} = C_2(\dot{\eta}) + \frac{1}{J_{yy}}\tau_{\theta}, \tag{4.5}$$

$$\ddot{\psi} = C_3(\dot{\eta}) + \frac{1}{J_{zz}}\tau_{\psi}, \tag{4.6}$$

where $F = \sum_{i=4}^{4} f_i$ is the total rotors thrust and $C(\dot{\eta})$ terms are defined as

$$C_1(\dot{\eta}) = {}^1\!/_{J_{xx}}(J_{yy} - J_{zz})\dot{\theta}\dot{\psi}, \tag{4.7}$$

$$C_2(\dot{\eta}) = {}^1\!/_{J_{yy}}(J_{zz} - J_{xx})\dot{\phi}\dot{\psi}, \tag{4.8}$$

$$C_3(\dot{\eta}) = {}^1\!/_{J_{zz}}(J_{xx} - J_{yy})\dot{\theta}\dot{\phi}. \tag{4.9}$$

In the control design, the input system variables can be considered as $\begin{pmatrix} \tau_\varphi & \tau_\theta & \tau_\psi & F \end{pmatrix}$. However, in the control realization, we only access to the physical inputs, i.e., motor's driver input, which can be expressed as rotor forces (f_i) by (4.10) and projected to the real hardware inputs through (3.4).

$$\begin{pmatrix} \tau_\varphi \\ \tau_\theta \\ \tau_\psi \\ F \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2}l & \frac{\sqrt{2}}{2}l & \frac{\sqrt{2}}{2}l & -\frac{\sqrt{2}}{2}l \\ \frac{\sqrt{2}}{2}l & \frac{\sqrt{2}}{2}l & -\frac{\sqrt{2}}{2}l & -\frac{\sqrt{2}}{2}l \\ \frac{d}{K_w} & -\frac{d}{K_w} & \frac{d}{K_w} & -\frac{d}{K_w} \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{pmatrix} \tag{4.10}$$

The derived simplified model is useful for control designed when the autopilot does not use the gyro. In the next section, we will derive the simplified quadrotor model when the gyro is in the control loop.

4.2 Simplified Model with Gyro in the Control Loop

From the control point of view, if the gyro in the loop configuration is used, model of the quadrotor will be different. In fact, since in this configuration the gyro drives the motors and stabilizes the attitude angle or angle ratio, from the autopilot perspective, the attitude dynamics are changed. However, there is no significant difference on the translational dynamics and thus, it can be defined as follows

$$\ddot{x} = \frac{-u_{Ft}}{m}\sin(\theta)\cos(\phi), \tag{4.11}$$

$$\ddot{y} = \frac{u_{Ft}}{m}\sin(\phi), \tag{4.12}$$

$$\ddot{z} = g + \frac{-u_{Ft}}{m}\cos(\theta)\cos(\phi), \tag{4.13}$$

where u_{Ft} can be seen as rotors total thrust which is proportional to the throttle input, given as

$$u_{Ft} = \frac{K_f}{1+\tau_f}u_{th}. \tag{4.14}$$

Equation (4.14) means for the upward thrust variation, i.e., altitude control, the gyro just controls all the motors proportionally to the throttle input u_{th}.

In this configuration, the attitude dynamics are depended on the gyro operating mode. As mentioned in Sections 3.4.1 and 3.4.2, for the gyro in hover mode we have the following closed loop attitude dynamics

$$\phi(s) = \frac{K_\phi}{(1+\tau_{\phi 1}s)(1+\tau_{\phi 1}s)}e^{-T_\phi}u_a, \tag{4.15}$$

$$\theta(s) = \frac{K_\theta}{(1+\tau_{\theta 1}s)(1+\tau_{\theta 1}s)}e^{-T_\theta}u_e, \tag{4.16}$$

$$\psi(s) = \frac{K_r}{s(1+\tau_r s)}u_r, \tag{4.17}$$

and for the cruise mode they are defined as

$$\phi(s) = \frac{K_p}{s\,(1+\tau_p s)} u_a, \tag{4.18}$$

$$\theta(s) = \frac{K_q}{s\,(1+\tau_q s)} u_e, \tag{4.19}$$

$$\psi(s) = \frac{K_r}{s\,(1+\tau_r s)} u_r. \tag{4.20}$$

In this configuration the model inputs are $u_{rc} = (u_a, u_e, u_{th}, u_r)$, which in fact are the same as the joystick outputs.

Due to the advantages indicated in Section 3.4, specifically from the safety aspect, we will use the gyro in the control loop configuration to realize the autonomous flight. Since the gyro has two operating modes, we will first observe the performance of each mode and then design the controller accordingly. However, before any autonomous flight realization, preliminary evaluation tests are necessary. These evaluations are addressed as ground analysis which will be described in the next section.

4.3 Ground Analysis

Before each flight test, we first analyze the whole system in the hardware-in-the-loop simulations and ground flight tests. In the hardware-in-the-loop simulation setup, the identified nonlinear model of the quadrotor is included in the embedded computer software to mimic the real quadrotor. However, all the sensors and hardware units are operating in this simulation. In fact, the only difference to the real flight is the controller outputs are applied to the software model instead of real servo inputs and the quadrotor states are not measured from the sensors but observed from the identified model states. This analysis mainly can help us to find the firmware bugs, algorithm mistakes, and avionic hardware failures. Furthermore, either from the ground station plotted states or off-line data analysis, we can roughly observe behavior of the quadrotor in the real flight situation. However, the analysis reliability highly depends on the accuracy of identified model.

If the hardware-in-the-loop simulation results are satisfactory, we go one step towards the real flight and do the ground flight tests. In this test, the quadrotor is tighten to a stand such that it can not fly but it is able to change its orientation. We can also hold the quadrotor in hand above the head, to observe its behavior in response to the designed controller. For this test, using safety equipments is necessary and important. The ground flight test is very close to the real flight, since the controller actually drives the motors and also sensors read the real platform states. If all these simulations and analysis results are satisfactory, a real flight test would be conducted. During the actual flight also, the pilot always carefully monitors the quadrotor and in case of any dangerous situation, he immediately switches back to the manual flight to recover the platform.

4.4 Attitude Control

In this section, we aim to design a controller to track the desire attitude angles. Since the gyro directly affects the attitude dynamics, its operating mode should be carefully selected. In general, the gyro in hover mode is appropriate for attitude regulation in hovering control, and the gyro in cruise mode is suitable for attitude and trajectory tracking purposes. We will analyze both gyro operating modes in detail.

4.4.1 Gyro in Hover Mode

In the hover mode, the gyro actually controls the quadrotor attitude angles. To observe its performance, we have done plenty of hardware-in-the-loop simulations and ground flight tests. The first test is to regulate the quadrotor attitude. Having the gyro is in the hover mode, it is known that, it inherently stabilizes the pitch and roll angles. However, for the heading, by considering the yaw dynamic model (4.17), it is required a controller to stabilize the heading. We use a PID controller which is given as

$$\begin{cases} C_\psi(s) & = k_{d\psi}s + k_{p\psi} + \frac{k_{i\psi}}{s} \\ u_r & = C_\psi(s)(\psi_{ref} - \psi), \end{cases} \tag{4.21}$$

where ψ_{ref} is the heading reference angle and $k_{d\psi}, k_{p\psi}, k_{i\psi}$ are the heading PID controller coefficients. The PID values are chosen to have a stable and smooth output. The controller coefficients are further tuned in the hardware-in-the-loop simulation and ground flight analysis which are given in Table 4.1. Therefore, the autopilot in this mode, just controls the heading and based on the closed loop pitching and rolling dynamics, applies the desire angles directly to the gyro input. Figure 4.1 illustrates the control block diagram of this setup.

The quadrotor response to attitude regulation controller is depicted in Figure 4.2. As it can be seen, the gyro performance is quite good for this specific task.

With having the gyro in the same mode, the next step is to observe the attitude

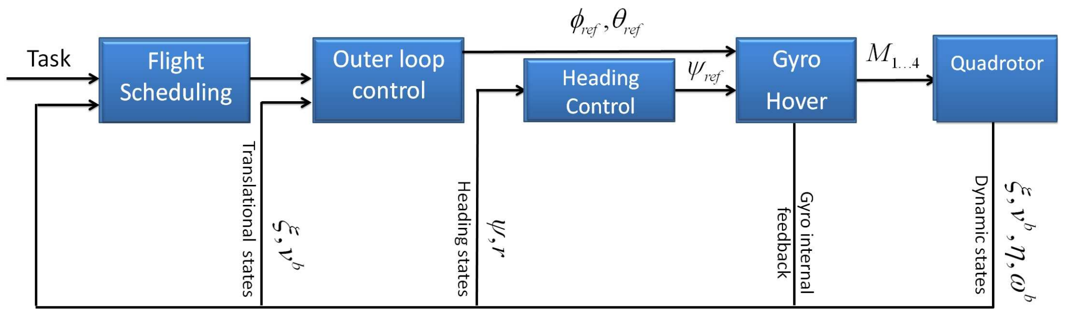


Figure 4.1: Control block diagram of the gyro (hover mode) in the loop configuration.

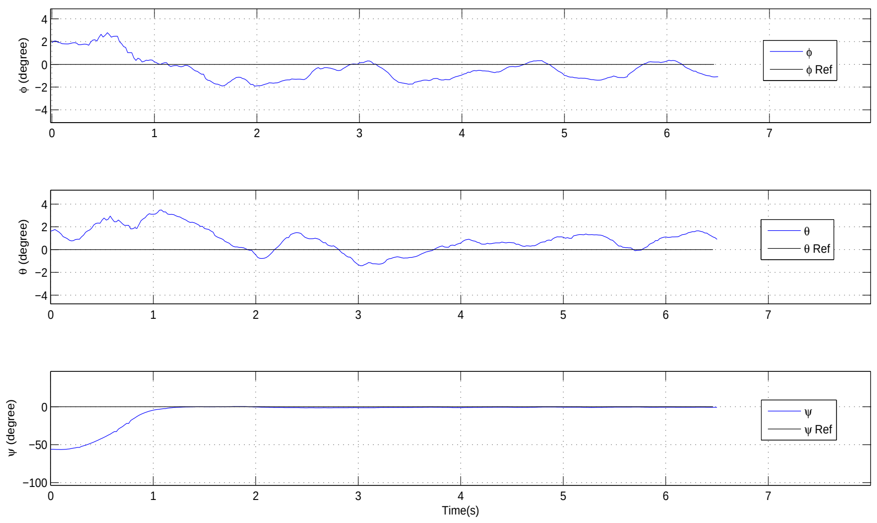


Figure 4.2: Attitude regulation in outdoor flight (gyro in hover mode).

tracking performance. Again, since the gyro is in the hover mode, it is able to control the pitch and roll angles. These dynamics, thus can be considered as closed loop systems which only need reference pitch and roll angles to track. Furthermore, as the gyro in this mode regulates the heading ratio, the yaw dynamic can be seen as a second order model (4.17), which tracks the desire heading by the help of autopilot heading controller (see Figure 4.1). The real flight result is demonstrated in Figure 4.3.

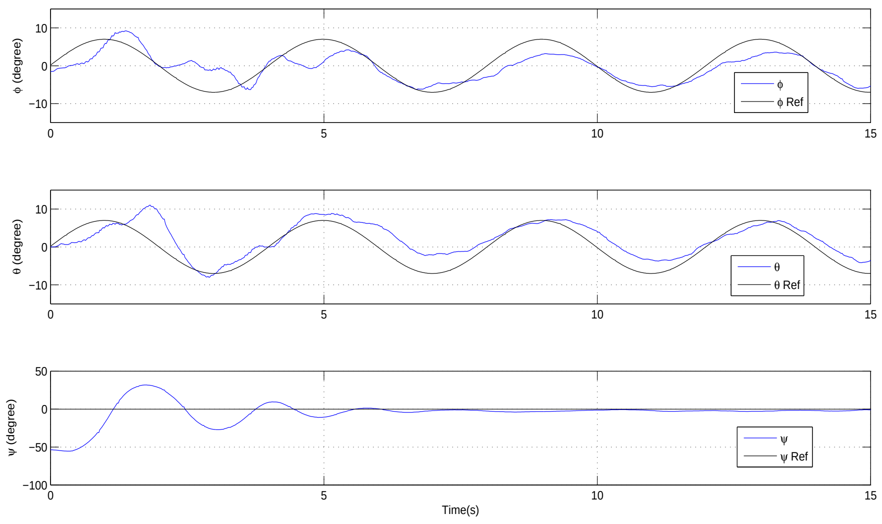


Figure 4.3: Attitude tracking in outdoor flight (gyro in hover mode).

As the figure shows, the performance of the gyro for this task is not very acceptable. This could be mainly due to the existence of the large delay in the gyro hover mode dynamic which is about 100 ms (see Table 3.7). In fact, the gyro in hover mode is designed to work closely around hovering condition, and therefore the embedded controller has a very slow but has smooth response.

4.4.2 Gyro in Cruise Mode

The gyro in the cruise mode has a different behavior. As discussed in Section 3.4, it only stabilizes the attitude ratios and hence, for attitude angle tracking, controllers are required to achieve the desired orientations. The control block diagram of this setup is shown in Figure 4.4. In this configuration, the gyro together with the autopilot attitude controller can be consider as a inner-loop system which tracks the outer-loop attitude references.

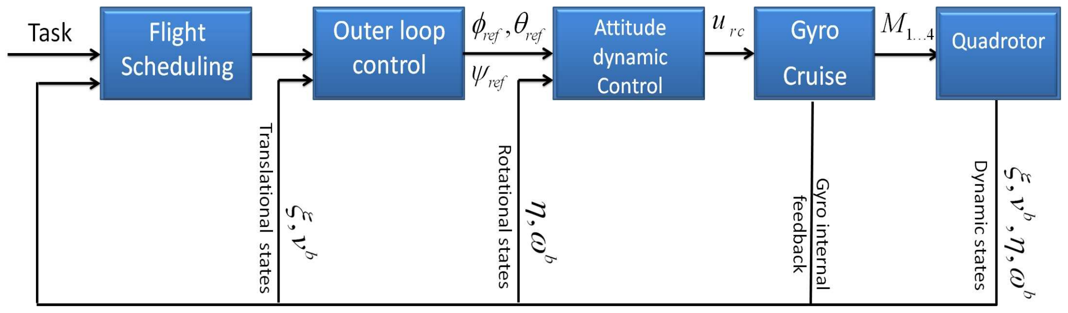


Figure 4.4: Control block diagram of the gyro (cruise mode) in the loop configuration.

In this part, we will include all the hardware-in-the-loop simulation results and ground flight tests data in this section. Based on the attitude dynamics (4.18)- (4.20), in this mode, the pitch and roll PID controllers are designed as (4.22),(4.23) and the heading controller is used the same as in the hover mode (4.21).

$$\begin{cases} C_\phi(s) = k_{d\phi}s + k_{p\phi} + \frac{k_{i\phi}}{s} \\ u_a = C_\phi(s)(\phi_{ref} - \phi) \end{cases}, \tag{4.22}$$

$$\begin{cases} C_\theta(s) = k_{d\theta}s + k_{p\theta} + \frac{k_{i\theta}}{s} \\ u_e = C_\theta(s)(\theta_{ref} - \theta) \end{cases}, \tag{4.23}$$

Where ϕ_{ref} and θ_{ref} respectively are the rolling and heading reference angles, $k_{d\phi}, k_{p\phi}, k_{i\phi}$ are the rolling PID controller coefficients, and $k_{d\theta}, k_{p\theta}, k_{i\theta}$ are the pitching PID controller coefficients. The PID coefficients then tuned in the ground analysis tests which are given in Table 4.1.

The designed controller firstly realized in the hardware-in-the-loop simulation. To illustrate the performance of the controller, a sinusoid reference signal with variable amplitude is applied to the controller and the quadrotor model states are recorded accordingly. The result is depicted in Figure 4.5 which demonstrates a perfect reference tracking. Next is to verify this result in the ground flight test which is shown in Figure 4.6. As the figure is demonstrated, the designed controllers are able to smoothly track the applied attitude references. Furthermore, to verify the controller robustness, it is set to regulate the quadrotor attitude and meanwhile several large disturbances are applied to the platform. As the Figure 4.7 shows, the controller is able to tolerate the external disturbance as large as 30 degrees tilt angle.

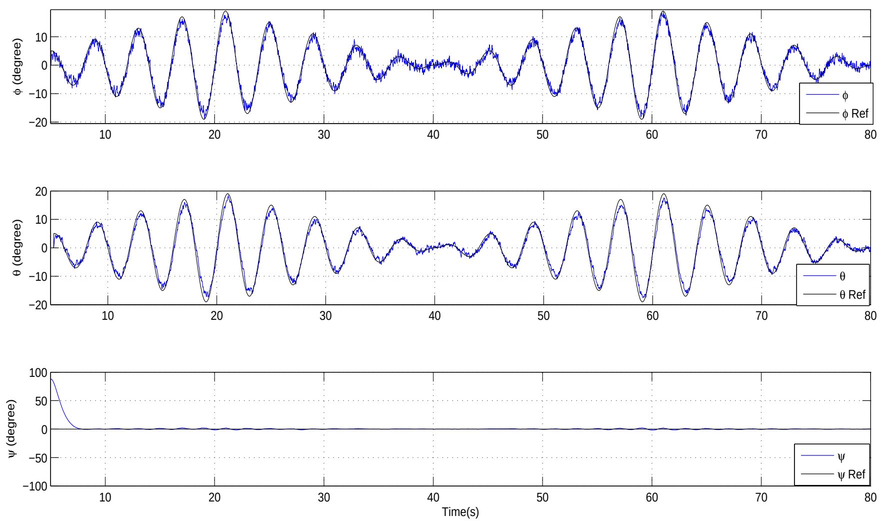


Figure 4.5: Attitude tracking in hardware-in-the-loop simulation (gyro in cruise mode).

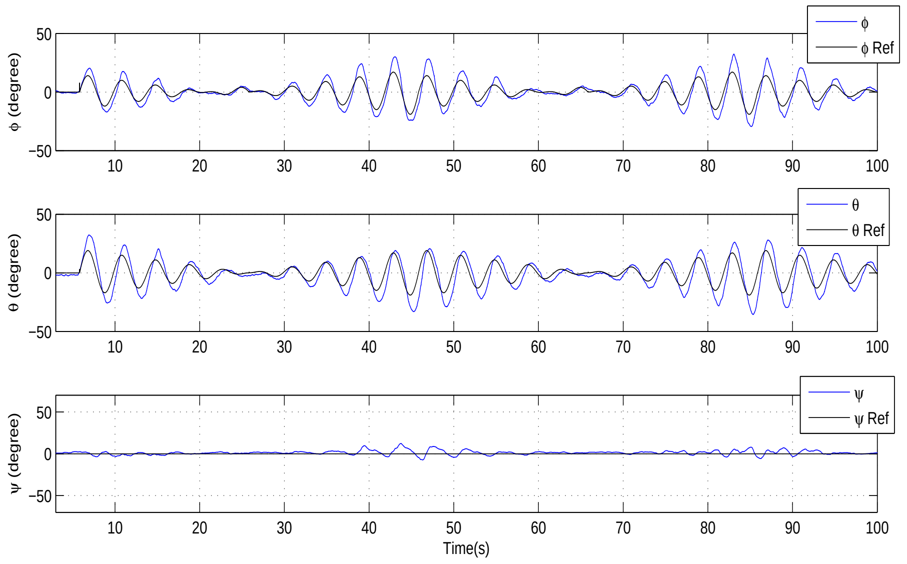


Figure 4.6: Attitude tracking in ground flight test (gyro in cruise mode).

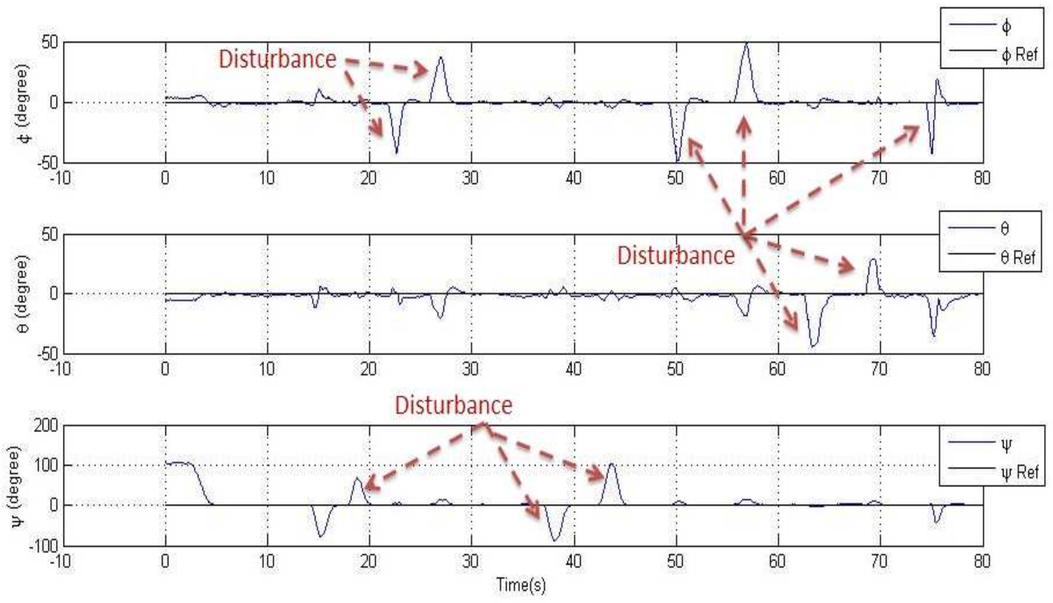


Figure 4.7: Attitude regulation robustness in ground flight test (gyro in cruise mode).

After the aforementioned simulations and experiments, we are ready for the real flight tests. The autonomous attitude regulation result is depicted in Figure 4.8. As it can be observed, the controller is able to stabilize the quadrotor attitude within the ± 4 degrees which is an acceptable hover mode attitude angle error. In addition, for the attitude tracking scenario, a sinusoid signal reference is applied to the flight control system. However, due to the drifting of the quadrotor position and also our flight field limitation, we only applied two complete sinusoid periods for this test. The Figure 4.9 demonstrates the flight test result which shows the quadrotor could smoothly track the given reference.

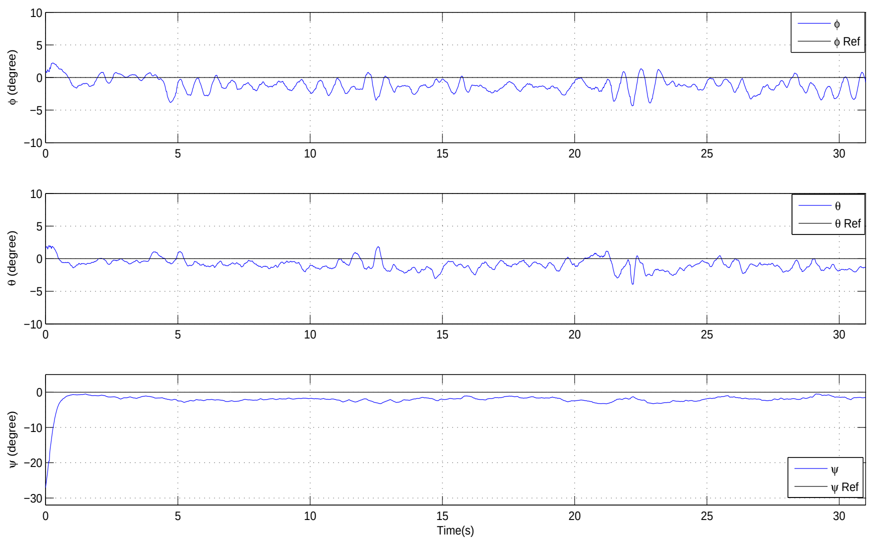


Figure 4.8: Attitude regulation in outdoor flight (gyro in cruise mode).

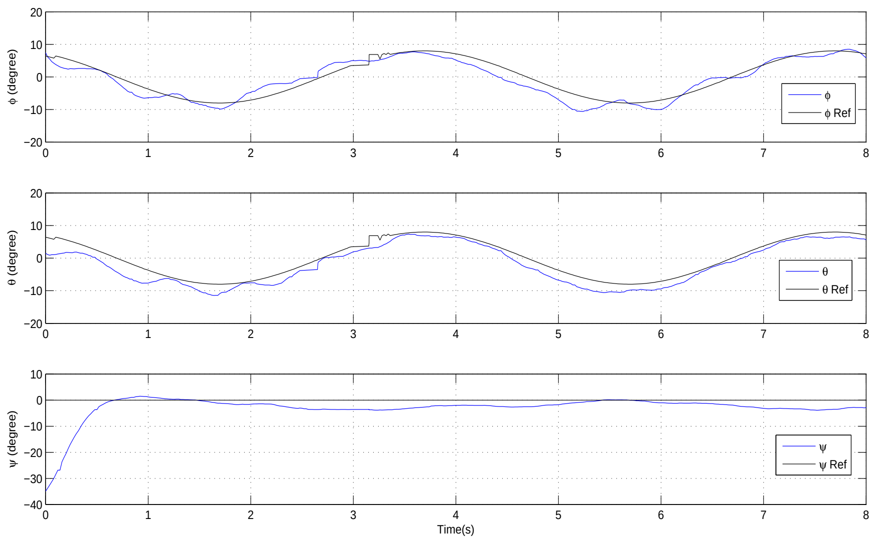


Figure 4.9: Attitude tracking in outdoor flight (gyro in cruise mode).

4.5 Position Hold

In this section, the aim is to design hierarchical controller to hold position of the quadrotor. In this part we are only interested to hold the quadrotor in x-y plane. The hardware setup is thus changed such that the pilot is able to control the altitude during the autonomous flight tests. For the inner-loop control, although the quadrotor is expected to operate close to the hovering condition, the gyro in the cruise mode is used, mainly because it has faster attitude tracking response which is required for hierarchical control architecture. Moreover, in order to instantly predict the quadrotor motion direction for possible safety reaction, the platform heading has been kept towards the true north, i.e., $\psi_{ref} = 0$.

As it is mentioned earlier, the quadrotor is an under-actuated system, hence there is no explicit inputs for horizontal positioning control [19]. To overcome this limitation, we use a hierarchical control structure, which as the Figure 4.10 shows, it has two layers of inner-loop and outer-loop controllers. The outer-loop layer controls the translational dynamic and generates require orientation for desire position displacement. The inner layer controls the rotational dynamic and tracks the requested desire orientation. For positioning control hence, the following variables, as the outer-loop inputs are required to be defined

$$u_x = \frac{-u_{Ft}}{m} \sin(\theta) \cos(\phi), \tag{4.24}$$

$$u_y = \frac{u_{Ft}}{m} \sin(\phi). \tag{4.25}$$

In these equations, u_x and u_y are the virtual inputs of the position dynamics in x and y directions respectively. Using (4.24) and (4.25), the (x,y) translational dynamics can be consider as double integrator, given as

$$\ddot{x} = u_x, \tag{4.26}$$

$$\ddot{y} = u_y. \tag{4.27}$$

Based on the introduced inputs (4.24) and(4.25), and the approximated position dynamic model (4.26) and (4.27), the following PID controllers for the outer-loop layer are designed to control the quadrotor position.

$$\begin{cases} C_x(s) = k_{dx}s + k_{px} + \frac{k_{ix}}{s}, \\ u_x = C_x(s)(x_{ref} - x), \end{cases} \tag{4.28}$$

$$\begin{cases} C_y(s) = k_{dy}s + k_{py} + \frac{k_{iy}}{s}, \\ u_y = C_y(s)(y_{ref} - y). \end{cases} \tag{4.29}$$

In (4.28) and (4.29), similarly (k_{dx}, k_{px}, k_{ix}) and (k_{dy}, k_{py}, k_{iy}) respectively are the (x,y) PID position controller coefficients which are given in Table 4.1.

As a hierarchical controller, the outer-loop layer provides the reference signals for the inner-loop controller based on (4.30) and (4.31). These attitude reference variables are obtained by inverting the translational dynamics (4.24) [49]. In Figure 4.10 this part is addressed as inverse dynamic block.

$$\theta_{ref} = \tan^{-1}\left(\frac{u_x}{-g}\right), \tag{4.30}$$

$$\phi_{ref} = \tan^{-1}\left(\frac{u_y \cos\theta}{g}\right). \tag{4.31}$$

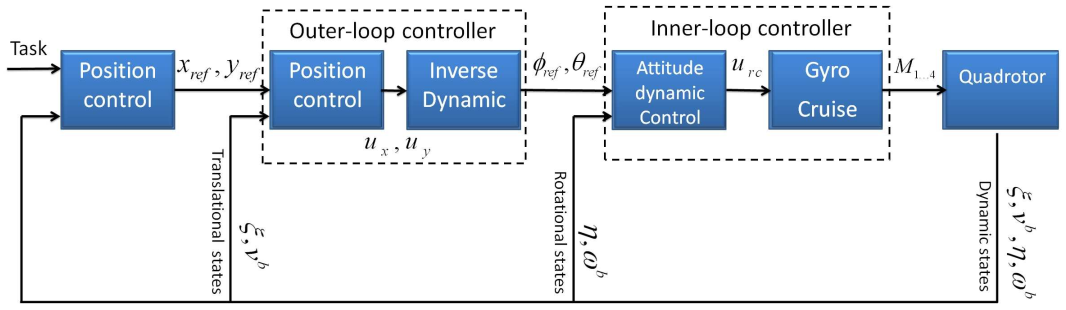


Figure 4.10: Control block diagram of the hierarchical control for position hold.

The result of autonomous position hold is shown in Figure 4.11. This figure contains the ground velocity, position drift in x-y plane, and the outer-loop input signals. As the figure demonstrates, the quadrotor horizontal position is kept bounded within a ± 1 m which is quite acceptable range for outdoor autonomous hovering. However, this situation due to the external disturbances such as wind and GPS drift can not be maintained for a long time. Improving this result will be considered as the future steps.

Parameter	Value
$k_{p\phi}, k_{d\phi}, k_{i\phi}$	$50, 10, 0.02$
$k_{p\theta}, k_{d\theta}, k_{i\theta}$	$60, 10, 0.02$
$k_{p\psi}, k_{d\psi}, k_{i\psi}$	$8, 3, 0.0001$
k_{px}, k_{dx}, k_{ix}	$0.01, 0.001, 0.05$
k_{py}, k_{dy}, k_{iy}	$0.2, 0.002, 0.0001$

Table 4.1: The PID coefficients for attitude tracking and position hold.

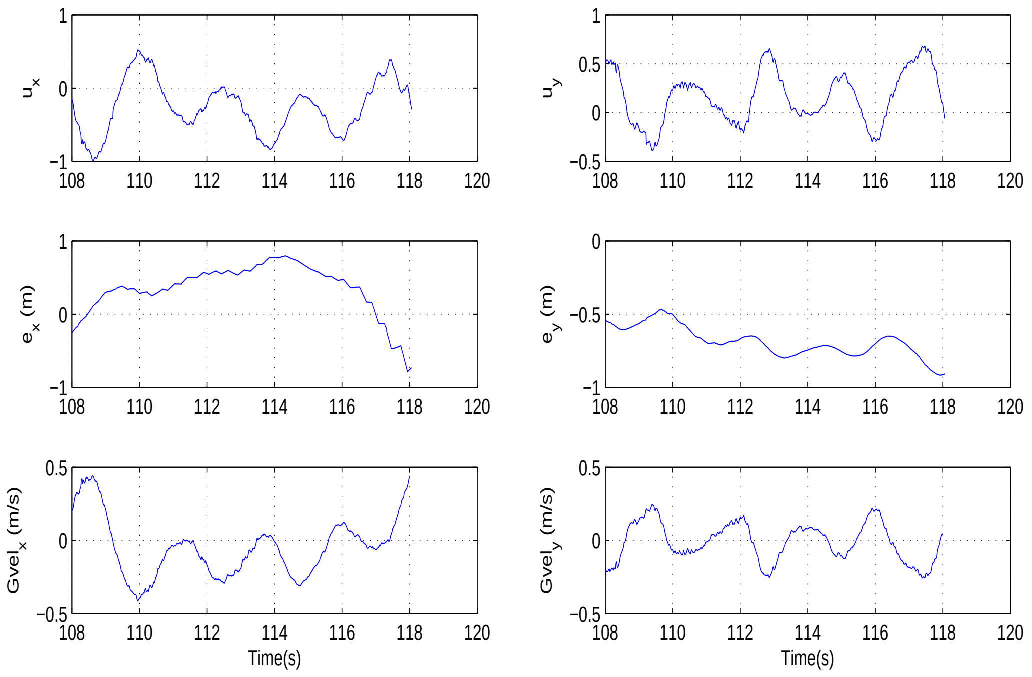


Figure 4.11: Autonomous position hold.

4.6 Conclusion

This chapter mainly focused on the realization of autonomous flight control. Firstly, to simplify the control design, the complete quadrotor nonlinear model, with/without considering the gyro in the control loop, was approximated and given as simplified models. However, due the gyro safety advantages, it is used in the control loop for autonomous flight realization. Testing both gyro operating modes, it is found that, the gyro in the hover mode provides smooth control signal but very slow response which results poor attitude tracking performance. However in the cruise mode, the faster controller provides good attitude tracking, but quadrotor has sharp displacement. This chapter also covers the preliminary flight analysis, including hardware-in-the-loop simulation and ground test which help to debug the autopilot system and predict the designed controller performance in the real autonomous flight. At the end, hierarchical controller is used to hold the quadrotor position. For each control task, the simulation and experimental results are included to illustrate the performance of proposed control method.

Chapter 5

Conclusions

5.1 Conclusions

This thesis mainly was focused on the development of a cross style quadrotor. Through this design, by taking advantages of using all the rotors in horizontal displacement, the cross style quadrotor could provide higher maneuverability as compared to standard quadrotors. Moreover, since in cross style quadrotor the front of platform is not blocked by the rotors, this style is more suitable for front camera mounting which usually is a desire position for most vision base navigation systems.

We also addressed the flight mechanisms difference in cross style quadrotor comparing to the standard plus type. In this new configuration, for all degrees of motion, all the rotors are participating to perform the desire displacement which is the main advantages of cross style quadrotor. The introduced quadrotor UAV was designed, developed, and experimentally archived an autonomous flight. The whole procedure of developing the quadrotor was discussed in three main areas: platform development, dynamic modeling, and autonomous attitude and position control.

- Platform development: In this aspect, mainly the hardware and software design procedure was discussed. The hardware development addressed the platform body features, the avionic system structure, and the manual flight components. Each part of the avionic system including the sensors, the main computer module, and the motor drivers was explicitly explained and their interconnection was illustrated. The on-board software also was developed to collect the sensors data, drive the rotors, and execute the control algorithms.

- Dynamic modeling: Through this aspect, the whole process of quadrotor modeling was studied. Firstly, the nonlinear mathematical model of the quadrotor in cross style was derived and it's unknown parameters were determined. Considering the nature of each parameter, static and dynamic experiments were designed accordingly. Through the static experiments, the platform weight, arms' length, rotor thrust dynamic were identified. Thereafter, the procedure of designing the dynamic experiments was presented. Several flight tests were conducted to identify the remaining model's unknown parameters and also to verify the ground experiments results. In addition, the configuration when the gyro is in the control loop was addressed and it's advantages and drawbacks were highlighted. In this configuration the attitude closed loop dynamic in the two gyro settings: hover mode and cruise mode, were identified. Finally, the estimated numerical quadrotor model was validated based on collected flight data.

- Control: This part mainly addressed the autonomous control of the quadrotor. Due to the safely features of the gyro in the control loop mode, the auto pilot was configured to used the gyro for inner-loop control. It was found that, the gyro in the hover mode provides smooth control signal but very slow response which results low performance attitude tracking. In the cruise mode, however, the faster controller provides a good attitude tracking, but the quadrotor showed sharp displacement motions. In the realization of autonomous flight, specifically for the outdoor tests, reliability of the whole system is crucial. Therefore, preliminary flight evaluations including the hardware-in-the-loop simulations and ground flight analysis were conducted before each flight tests. At the end, the PID based outer-loop and inner-loop controllers were design to hold the quadrotor position and perform the attitude regulation and tracking scenarios. The simulation and experimental results were included to illustrate the performance of proposed control structure.

There are still a lot of room for advanced development of this type of quadrotor. Realizing the trajectory tracking scenario can be the next future step. For such a kind of task, the simplified model may not be helpful, and hence considering more general model and using more advanced controllers may be required. The gyro also can be removed from the control loop, if the autopilot system can provide a reliable attitude stabilization in both manual and autonomous flight. This requires that the autopilot inner-loop controller be tuned reliably on the ground and also the avionic system should be able to collect the reference signal both from outer-loop layer and the RC receiver. If accurate localization is required, adding camera and laser scanner to the quadrotor can be very useful. They also can be used to realize more advanced scenarios such as map building and target detection. Ultimately, the cross style quadrotor, with having higher maneuverability in a simple structure platform has a potential to be a reliable testbed for arial robotic research.

Bibliography

[1] A. Ollero and L. Merino, “Control and perception techniques for aerial robotics,” *Annual Reviews in Control*, vol. 28, no. 2, pp. 167–178, 2004.

[2] J. Redding, T. McLain, R. Beard, and C. Taylor, “Vision-based target localization from a fixed-wing miniature air vehicle,” in *American Control Conference, 2006*, Jun. 2006, p. 6 pp.

[3] J. Artieda, J. Sebastian, P. Campoy, J. Correa, I. Mondragn, C. Martnez, and M. Olivares, “Visual 3-D SLAM from UAVs,” *Journal of Intelligent & Robotic Systems*, vol. 55, no. 4, pp. 299–321, 2009.

[4] P. Doherty and P. Rudol, “A UAV search and rescue scenario with human body detection and geolocalization,” in *AI 2007: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science, M. Orgun and J. Thornton, Eds. Springer Berlin / Heidelberg, 2007, vol. 4830, pp. 1–13.

[5] R. M. Murray, “Recent research in cooperative control of multivehicle systems,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 571–583, 2007.

[6] A. Tayebi and S. McGilvray, “Attitude stabilization of a VTOL quadrotor aircraft,” *Control Systems Technology, IEEE Transactions on*, vol. 14, no. 3, pp.

562 – 571, May 2006.

[7] S. McGilvray, *Attitude Stabilization of a Quadrotor Aircraft*, 2005.

[8] A. Partovi, W. Xinhua, K. Lum, and L. Hai, “Modeling and control for a small-scale hybrid aircraft,” vol. 18, 2011, pp. 10 385–10 390.

[9] A. Parotvi, X. Wang, and H. Lin, “Robust trajectory tracking for a small-scale hybrid aircraft,” *23rd Canadian Congress of Applied Mechanics*, 2011.

[10] A. R. Partovi, H. Lin, G. Cai, B. Chen, and A. Z. Y. Kevin, “Development of a cross style quadrotor,” in *AIAA Guidance, Navigation, and Control Conference.* American Institute of Aeronautics and Astronautics, Aug. 2012.

[11] G. Hoffmann, D. Rajnarayan, S. Waslander, D. Dostal, J. Jang, and C. Tomlin, “The stanford testbed of autonomous rotorcraft for multi agent control (STAR-MAC),” in *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, vol. 2, 2004, pp. 12.E.4–121–10 Vol.2.

[12] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, “Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering,” in *IEEE International Conference on Robotics and Automation, 2009. ICRA ’09.* IEEE, May 2009, pp. 3277–3282.

[13] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, “Quadrotor helicopter flight dynamics and control: Theory and experiment,” in *Proc. of the AIAA Guidance, Navigation, and Control Conference*, 2007.

[14] G. M. Hoffmann, H. Huang, S. L. Waslander, and Tomlin, “Precision flight

control for a multi-vehicle quadrotor helicopter testbed," *Control Engineering Practice*, vol. 19, no. 9, pp. 1023–1036, Sep. 2011.

[15] P. Pounds, R. Mahony, P. Hynes, and J. Roberts, "Design of a four-rotor aerial robot," in *Proc. 2002 Australasian Conference on Robotics and Automation*, vol. 27, 2002, p. 29.

[16] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a large quadrotor robot," *Control Engineering Practice*, vol. 18, no. 7, pp. 691–699, Jul. 2010.

[17] K. Nonami, F. Kendoul, S. Suzuki, W. Wang, D. Nakazawa, K. Nonami, F. Kendoul, S. Suzuki, W. Wang, and D. Nakazawa, "Mathematical modeling and nonlinear control of VTOL aerial vehicles," in *Autonomous Flying Robots.* Springer Japan, 2010, pp. 161–193.

[18] M. Valenti, B. Bethke, D. Dale, A. Frank, J. McGrew, S. Ahrens, J. P. How, and J. Vian, "The MIT indoor Multi-Vehicle flight testbed," in *2007 IEEE International Conference on Robotics and Automation.* IEEE, Apr. 2007, pp. 2758–2759.

[19] J. Kim, M. Kang, and S. Park, "Accurate modeling and robust hovering control for a quadrotor VTOL aircraft," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 9–26, Sep. 2009.

[20] C. Nicol, C. Macnab, and A. Ramirez-Serrano, "Robust adaptive control of a quadrotor helicopter," *Mechatronics*, vol. 21, no. 6, pp. 927–938, Sep. 2011.

[21] T. Madani and A. Benallegue, "Control of a quadrotor Mini-Helicopter via full state backstepping technique," in *2006 45th IEEE Conference on Decision and*

Control. IEEE, Dec. 2006, pp. 1515–1520.

[22] *XAIRCRAFT X650 Quadrocopter*, 2011. [Online]. Available: http://www.fpvhobby.com/118-xaircraft-x650-cf-edition.html

[23] *Gumstix Overo Fire*, 2011. [Online]. Available: https://www.gumstix.com/store/product_info.php?products_id=227

[24] *Summit expansion board*, 2011. [Online]. Available: https://www.gumstix.com/store/product_info.php?products_id=215

[25] *Ardu-IMU*, 2011. [Online]. Available: http://code.google.com/p/ardu-imu/

[26] *GS407 u-blox GPS*, 2011. [Online]. Available: http://www.sparkfun.com/datasheets/GPS/GS407-090812.pdf

[27] *Pololu Micro Serial Servo Controller*, 2011. [Online]. Available: http://www.pololu.com/catalog/product/207

[28] *Arduino Pro Mini*, 2011. [Online]. Available: http://arduino.cc/it/Main/ArduinoBoardProMini

[29] F. Wang, T. Wang, B. Chen, and T. Lee, "An indoor unmanned coaxial rotorcraft system with vision positioning," pp. 291–296, 2010.

[30] X. Dong, M. Dong, B. Wang, B. M. Chen, and T. H. Lee, "A comprehensive software system architecture for unmanned aerial vehicles," in *2011 IEEE International Conference on Service Operations, Logistics, and Informatics (SOLI)*. IEEE, Jul. 2011, pp. 595–600.

[31] X. Dong, B. M. Chen, G. Cai, H. Lin, and T. H. Lee, "Development of a com-

prehensive software system for implementing cooperative control of multiple unmanned aerial vehicles," in *IEEE International Conference on Control and Automation, 2009. ICCA 2009.* IEEE, Dec. 2009, pp. 1629–1634.

[32] A. Karimoddini, G. Cai, B. Chen, H. Lin, and T. Lee, "Hierarchical control design of a uav helicopter," *Advances in Flight Control Systems. INTECH, Vienna, Austria, in press. ISBN*, pp. 978–953, 2011.

[33] M. Dong, B. Chen, G. Cai, and K. Peng, "Development of a real-time onboard and ground station software system for a UAV helicopter," *Development*, vol. 4, 2007.

[34] *Receiver PPM Sum Signal*, 2011. [Online]. Available: http://pixhawk.ethz.ch/wiki/tutorials/receiver_sum_signal

[35] *Barometer sensor*, 2011. [Online]. Available: http://www.freescale.com/files/sensors/doc/data_sheet/MPXV7002.pdf

[36] S. Raza, "Design and control of a quadrotor unmanned aerial vehicle." University of Ottawa, 2010.

[37] L. Besnard, "Control of a quadrotor vehicle using sliding mode disturbance observer," Ph.D. dissertation, The University of Alabama in Huntsville, United States – Alabama, 2007, M.S.E.

[38] R. Beard, "Quadrotor dynamics and control," *Brigham Young University*, 2008.

[39] G. Cai, B. Chen, and T. Lee, *Unmanned Rotorcraft Systems.* Springer, 2010.

[40] G. Genta and C. Delprete, "Some considerations on the experimental determi-

nation of moments of inertia," *Meccanica*, vol. 29, no. 2, pp. 125–141, 1994.

[41] Z.-C. Hou, Y.-n. Lu, Y.-x. Lao, and D. Liu, "A new trifilar pendulum approach to identify all inertia parameters of a rigid body or assembly," *Mechanism and Machine Theory*, vol. 44, no. 6, pp. 1270–1280, Jun. 2009.

[42] *Lab Experiment for Measurment Moment of Inertia*, 2011. [Online]. Available: http://www.sdrl.uc.edu/academic-course-info/docs/ucme571/moment_inertia.pdf

[43] C. Harris, A. Piersol, and I. ebrary, *Harris' shock and vibration handbook.* McGraw-Hill, 2002, vol. 5.

[44] *Matlab polyfit function*, 2011. [Online]. Available: http://www.mathworks.com/help/techdoc/ref/polyfit.html

[45] M. Tischler and R. Remple, *Aircraft and rotorcraft system identification.* American Institute of Aeronautics and Astronautics, 1801 Alexander Bell Drive, Suite 500, Reston, VA, 20191-4344, USA,, 2006.

[46] G. Cai, "Development of small-scale unmanned-aerial-vehicle helicopter systems," 2009.

[47] L. Ljung, *System identification.* Wiley Online Library, 1999.

[48] S. K. Kim and D. M. Tilbury, "Mathematical modeling and experimental identification of an unmanned helicopter robot with flybar dynamics," *Journal of Robotic Systems*, vol. 21, no. 3, p. 95116, 2004.

[49] F. Kendoul, I. Fantoni, and R. Lozano, "Asymptotic stability of hierarchical

inner-outer loop-based flight controllers," in *Proceedings of the 17th IFAC World Congress*, pp. 6–11.