2015-10-21

Vision-based control and autonomous landing of a VTOL-UAV

Schram, Kurtis W

Vision-based control and autonomous landing of a VTOL-UAV

by

Kurtis W. Schram

Under the Supervision of Dr. Abdelhamid Tayebi

A Thesis
Presented to Lakehead University
in Partial Fulfillment of the Requirement for the Degree of
Master of Science
in
Electrical and Computer Engineering

Thunder Bay, Ontario, Canada

September 2014

Abstract

In recent years the popularity of quadrotor unmanned aerial vehicles (UAVs) has increased. Today, UAVs are widely used by military and police forces for surveillance. They are used by industry for such tasks as traffic monitoring, infrastructure inspection or even delivery of goods. They are used by individuals for hobby flying and aerial photography. It is currently of great interest in the research community to improve the level of autonomy of the UAV for these and future uses. One particular problem is the ability to stabilize over and land on a moving platform. This situation can easily arise for a quadrotor returning to a ship at sea or even a landing pad affixed to a vehicle. Many current techniques rely on knowledge of the platform and its motion, or a predictive model. This information is not always available or accurate. A solution that does not require knowledge of the target is desirable.

This thesis deals with practical implementation of optical flow based position stabilization and autonomous landing algorithms for a quadrotor UAV. The quadrotor used is a common low cost platform with a large open source community.

Firstly, non-linear estimation and control techniques are implemented for the attitude stabilization using low-cost sensors and limited computational power. Some methods for the system parameters estimation are presented and some challenges related to the implementation are discussed. Despite the ability of the attitude controller to stabilize the orientation of the quadrotor, hovering and landing precisely over a specific area is not possible without a position stabilization scheme. In applications where GPS signals are not available and the hovering target is a priori unknown, it is common to rely on visual information. In this context, this thesis aims for the development of an efficient optical-flow-based position stabilization and autonomous landing scheme for the quadrotor UAV.

Acknowledgments

There are many people to thank for their invaluable contributions to this thesis. Firstly, I would like to thank my supervisor Dr. Abdelhamid Tayebi for his patience, support and guidance. Dr. Tayebi was always more than happy to share his considerable experience and knowledge.

I would also like to thank technologists Kailash Bhatia and Bruce Misner for helping to design and provide various components used in the project.

Finally I would like to thanks fellow graduate students Siddhant Nayak, Munir Enbaia and Joshua Abela for their friendship and insights.

Kurtis W. Schram

kschram@lakeheadu.ca

Contents

List of Figures

List of Tables

List of Abbreviations

UAV – Unmanned Aerial Vehicle.
DCM – Direction Cosine Matrix.
ESC – Electronic Speed Controller.
MEMS – Micro-Electro-Mechanical System.
SDK – Software Development Kit.
APM – ArduPilot Mega.
SPI – Serial Peripheral Interface.
IGRF – International Geomagnetic Reference Field.
WMM – World Magnetic Model.
PWM – Pulse Width Modulation.
PPM – Pulse Position Modulation.
ADC – Analog-to-Digital Converter.
EMF – Electro-Motive Force.
IMU – Inertial Measurement Unit.
GPS – Global Positioning System.
SONAR – Sound Navigation and Ranging.
GES – Globally Exponentially Stable.
CPU – Central Processing Unit.
DC – Direct Current.
AC – Alternating Current.
FPU – Floating Point Unit.
CSV – Comma Separated Values.
LIPO – Lithium Polymer.
PCB – Printed Circuit Board.
EKF – Extended Kalman Filter.
MEKF – Multiplicative Extended Kalman Filter
PID – Proportional Integral Derivative.
PD – Proportional Derivative.
PI – Proportional Integral.
OF – Optical Flow.

Chapter 1

Introduction

The quadrotor unmanned aerial vehicle (UAV) is a flying robotic vehicle which does not have a human operator on board. The quadrotor design consists of a crossed frame with a rotor on each end for a total of four rotors. Quadrotor UAVs have some interesting capabilities. They are capable of vertical take-off and landing (VTOL), are more manoeuvrable and are of simpler mechanical design than conventional helicopters. Unlike helicopters which rely on the tail rotor and a variable pitch main rotor for control, quadrotors consist of four main propellers. These propellers are all of the same size with a fixed pitch and control is achieved by varying the angular velocity of the rotors with respect to each other to introduce a net torque on the airframe.

These abilities make the quadrotor ideally suited to many applications. Quadrotors are currently used in military and police surveillance applications, aerial photography, inspection of infrastructure, monitoring of hazardous environments and recently even delivery of packages. These applications are all hazardous or too dull for humans to undertake, making them ideally suited for an autonomous UAV.

Similar designs exist having 6 and 8 rotors on a frame consisting of 6 or 8 arms. These configurations offer a higher lift capacity. It is also possible to have a 3 or 4 arm configuration with rotors on the top and bottom of the end of each arm. These configurations

offer redundancy in the case of one motor failure. Larger platforms with higher lift capacity are becoming increasingly popular as new applications are developed requiring more advanced (and usually heavier) payloads. The control problem is very similar in all of these cases, only the quadrotor configuration is considered in this thesis.

1.1 History of the Quadrotor

Quadrotors have been around for many years and were among the first successful vertical take-off and landing vehicles. In the past many attempts were made to design and build quadrotor aircraft. None of the designs really caught on for commercial or military applications as they were unrefined and unable to achieve suitable flight performance. Recent advances in low cost, lightweight sensors as well as lightweight materials have revived interest in the quadrotor. Over the past decade quadrotors have been steadily gaining popularity as it is now possible to build operational aircraft cheaply and with relative simplicity. Today, quadrotor aircraft can be found in many applications and are becoming more familiar to many people. While once only built as secretive military prototypes, quadrotors are now commonly found as everything from research platforms to military surveillance craft and even toys.

The first quadrotor capable of carrying a human in flight was constructed in 1907 by the Bréguet brothers [1]. Louis and Jacques Bréguet became interested in helicopters in 1906 and began conducting experiments. They had considerable engineering knowledge and funding as the Bréguet family were well known clock makers. They used a more scientific approach in their design than others at the time. The Bréguet-Richet Gyroplane No. 1 was constructed of steel girders in a cross configuration. At the end of each arm was a rotor made of 4 cloth covered surfaces. The pilot sat at the center of the craft below the eight cylinder engine used to turn the rotors via belts and pulleys. The craft weighed 1124 lbs and with a 150 lb pilot on board the total flying weight was 1274 lbs. Reports indicate that the machine was able to rise to about 5 meters in flight. There was no way to control the craft other than the speed of the motors resulting in very poor stability. Many flights incorporated a tether or men to help hold the craft stable during tests.

In 1908 the brothers attempted a new design called the Gyroplane No. 2, however the design could no longer be considered a quadrotor [1].

The next significant attempt at building a quadrotor aircraft was made by Etienne Oemichen in 1922. Oemichen was an engineer with Peugeot motor company and enjoyed experimenting with rotary wing aircraft. Oemichen built several different vertical take-off machines some of which included a hydrogen filled balloon to increase lift and stability. The most successful craft was the Oemichen No. 2. This machine had 4 rotors and 8 propellers and was powered by a single engine. The Oemichen No. 2 was the first machine of this type to offer some stability and controllability. Over the next few years the aircraft made thousands of successful flights. On May 4, 1924 the craft was able to fly for over a mile while being airborne for more than 14 minutes. Oemichen was still unhappy with this performance and turned to hydrogen balloon assisted aircraft for his future work [2].

In 1922 the US Army made its first serious attempt to develop a quadrotor aircraft. The contract was awarded to Dr. George de Bothezat. The resulting design was an X shaped frame with six bladed rotors on each end. There were also additional variable pitch propellers used for yaw control. Over one hundred test flights were completed, some of which with as many as three men on board. The aircraft was only ever able to a achieve a maximum altitude of 5 meters, far from the goal of 100 meters. While de Bothezat had demonstrated that an aircraft of this type was theoretically possible, it was too complex, under-powered and not very manoeuvrable. Additionally it proved very difficult to fly. The Army abandoned the project when the performance did not meet their expectations [3].

One of the greatest contributions to the quadrotor aircraft came in the 1950's by D.H. Kaplan. Kaplan's quadrotor was further refined than previous attempts and was the first to demonstrate the use of differential thrust between the pairs of rotors for control, as used by current quadrotors. However, this project was terminated later as there was not a sufficient amount of orders made for commercial or military versions.

Following Kaplan's design, interest in a quadrotor aircraft capable of carrying a human in flight had diminished. Interest in the quadrotor platform shifted to an unmanned vehicle approach resulting in much smaller and lighter aircraft. Advances in technology have also resulted in autopilot systems allowing the aircraft to self-stabilize, eliminating many of the difficulties involved in flying quadrotor aircraft. One of the first of these small modern quadrotors was the Draganflyer, built in Canada by RCToys [4]. While it was only intended as a toy, it showed that quadrotor technology could be realized on a low-cost mass produced platform. Since then the quadrotor platform has gained popularity as a research platform at many universities. In 2001 Stanford University sponsored the Mesicopter project [5], which focused on control design and manufacturing. The Star-Mac project created a highly manoeuvrable quadrotor platform and helped to show the concept of multi-agent flight where multiple UAVs can fly together and cooperate to achieve their task. [6].

Very recently, highly capable quadrotor aircraft platforms have begun to move out of university laboratories and into widespread commercial use. These models offer a ready-to-fly solution aimed at law enforcement and professional aerial photography and videography. The SkyRanger from Aeryon Labs Inc. [7] is a recently introduced model. It offers touch screen control, fifty minutes of flight time, self-stabilized camera platform and a convenient folding frame. The SkyRanger camera can be streamed live to any device on the network and is capable of autonomous navigation using commercially available maps. Draganflyer, the same company that started with toy quadrotors, now produces a line of highly featured quadrotors. Like the SkyRanger, the Draganflyer line offers long flight time, easy to fly controls and an integrated camera. The Draganfly quadrotors are a popular choice for police and military applications [8].

1.2 Motivation

A great deal of work has been done in the field of quadrotor UAVs. Stabilization and attitude estimation of quadrotors are fairly well understood topics. There are many successful works in literature in these areas. Much of the work on quadrotors is theoretical

in nature. An increasing number of researchers are using vision systems for position estimation and control. While these systems are highly accurate and work well in a controlled setting where cameras can be mounted, the estimation is not available outdoors or outside of the controlled environment. This thesis focuses on a full estimation and control solution that can be run on-board the quadrotor's embedded hardware. This way, the quadrotor does not require a complex and expensive camera system to operate. The methods used are selected for simplicity and efficiency such that the algorithms can operate using the low-cost sensors and microcontrollers commonly available on many quadrotor platforms.

Currently, there is a strong push to increase autonomy of UAVs. One major problem is the ability to land without operator assistance on a moving platform. This is an important ability especially for maritime operations. Consider a ship at sea bobbing up and down in the waves. As the UAV approaches the deck during the landing manoeuvre, the upward motion of the ship due to the waves may cause a collision. It is necessary to match the motion of the UAV with the deck to prevent such a collision. In many cases, the landing target would not be known and the motion of the platform may also not be well known. Even if the landing platform at the home base was known, a situation may arise where an emergency landing is required in an unknown environment. For this reason it is necessary to use a technique for landing that does not require any knowledge of the landing pad or its motion. Also, it is important for the process to not only have a strong theoretical background but to also have the ability to be implemented in real-time using off-the-shelf hardware.

1.3 Research Challenges

The quadrotor platform is naturally unstable, and consists of non-linear dynamics. Estimation and control can become challenging for real-world systems where noise and sensor biases affect the system. The vision system requires time consuming calculations which can lead to low sample rates and delay of data. Since one goal of this thesis is to implement a working practical system, these problems need to be addressed. Some of the

main challenges are summarized below.

- The system used consists of low-cost sensors which will require careful consideration of noise and sensor bias in the estimation process.
- All testing will be done in an indoor environment which will make obtaining accurate measure of the magnetic field from the magnetometer difficult.
- The estimation and control algorithms are implemented on a low-cost 16 MHz microcontroller with no floating point unit (FPU).
- The vision system sample rate will be limited due to time consuming calculations of optical flow. This process will also need to be done off-board and will require two-way wireless serial communication.
- The quadrotor platform is inherently dangerous to be around due to the high speed spinning propellers. Safety precautions will need to be observed during testing.

1.4 Literature Review

1.4.1 Attitude Estimation

To control the attitude of a quadrotor, it is first necessary to estimate its attitude. There is no single sensor that can measure the attitude. Many quadrotor platforms have three sensors available: accelerometers, gyroscopes and magnetometers that allow for reconstruction of the attitude. Many methods have been proposed over the years to solve the attitude estimation problem. A few popular methods are briefly discussed in this section.

One of the first solutions to the problem of attitude estimation was the TRIAD algorithm. This method was proposed by Harold D. Black in 1964 [9]. It is also sometimes called the Algebraic Method. In the TRIAD method, the information from two non-collinear vectors known in the inertial frame and body frame is used to construct the attitude matrix. The algorithm can only accommodate a maximum of two vector measurements,

but is simple to implement. Also, the contributions of each vector measurement cannot be weighted to adjust for the reliability of each measurement. This method was used in spacecraft attitude estimation for a number of years.

Following the introduction of the TRIAD method, some methods were introduced to optimize the attitude matrix. Optimal methods are computationally more complex but allow for more than two vector measurements to be used, and use a cost function to calculate the best attitude estimate. The first and most well known optimal approach was Wahba's problem [10]. Proposed solutions to Wahba's problem required extensive calculations and no solutions were able to outperform TRIAD in practice until the q-method, and later QUEST, were introduced.

The q-method was proposed by Paul Davenport in 1968 [11]. He used the unit-quaternion representation to exploit the computational efficiency resulting from the four parameter representation. Shuster expanded on this work with the introduction of QUEST [12]. The algorithm was designed to solve for the optimal quaternion resulting from the q-method. QUEST proved to be one of the most popular and widely used solutions to Wahba's problem. QUEST was implemented in the Magsat satellite in 1979. Numerous other solutions to Wahba's problem and variants of the QUEST were proposed over the years, all requiring a trade-off between precision and computational efficiency. One of the main drawbacks of these methods is that information about measurements of previous attitude is not carried forward.

Since measurement from the sensors are contaminated with noise, a method which uses filtering is preferred. A number of filtering methods for attitude estimation have been proposed. These methods use vector measurements in combination with the kinematic model of rotation and the angular rate measurements [13]. One of the most common and successful filtering based estimation techniques is the Kalman filter.

The Kalman filter was first introduced in [14] by Rudolf E. Kalman, since then it has become the workhorse of attitude estimation. The Kalman filter has been successfully implemented in applications ranging from guidance and navigation systems of NASA's

Apollo program and Space Shuttle program to U.S. Navy submarines and cruise missiles [15]. The filter operates in a recursive fashion on a set of noisy measurements to produce an optimal estimate of the system state. The Kalman filter algorithm is broken down into two steps. The first step of the Kalman filter predicts the current state variables based on the system dynamic model and the previous estimate. The next step involves updating the predicted states with weighted noisy measurements from the sensors to produce the current estimate of attitude. Due to the recursive nature of the Kalman filter, it is ideally suited for real-time applications involving noisy or otherwise corrupted sensor measurements. The Kalman filter was originally developed for a system defined by a set of linear differential equations and in its original form is not well suited for the non-linear dynamics of the quadrotor platform. The Kalman filter was "extended" to produce the extended Kalman filter (EKF) which can be applied to non-linear systems. It linearizes the non-linear system about the most recent state estimate to obtain the Jacobian matrices in the Kalman filter [16]. Multiplicative EKF (MEKF) proposed in [17] and the Additive EKF (AEKF) proposed in [18] are both popular variations of the EKF. The MEKF has been successfully implemented in the Space Precision Attitude Reference System (SPARS) in 1969 [19] and NASA's Multimission Modular Spacecraft [20] and since then has been widely used in many practical applications [21], [22]. The MEKF algorithm has also been modified to incorporate GPS measurements to include the rigid body position and velocity in the state vector (see [23] and [24]). While the various versions of the Kalman filter have proven themselves effective in practical applications, the main drawback is its computational complexity and its local nature as it relies on the system linearization.

Another non-linear estimation techniques that uses filtering is the explicit complementary filter proposed in [25]. This approach relies on the non-linear attitude dynamics without any linearization and uses the inertial vector measurement and the angular velocity in a non-linear dynamic model to estimate the attitude and gyro-bias. The authors prove that with a minimum of two non-collinear vector measurements the estimated attitude converges to the real attitude for a wide variety of initial conditions. A complimentary filter technique is chosen for implementation in this thesis, for more details see section

4.1.

1.4.2 Attitude Control

Once attitude estimation is achieved it is necessary to implement a controller that can ensure the actual (estimated) attitude converges to the desired attitude in order to control the flight of the quadrotor. Many classical control techniques have been applied to the quadrotor. Both linear and non-linear methods have been used.

The Arducopter project utilizes a PID based control to achieve stable flight. While the controller works adequately in practice, there is no proof of stability for the controller. The controller has been implemented based more on intuition and trial and error than rigorous theory. Additionally, since the controller is linear while the quadrotor model consists of non-linear dynamics the performance will likely degrade during aggressive manoeuvres or large attitude angle errors.

Work in [26] and [27] uses a simple PID controller. In both cases it is shown that the PID controller used is sufficient for stable flight. Many other researchers move away from PID control to more complex methods in an effort to improve performance. Research in [28], [29] shows that sliding mode control and backstepping control are also effective in achieving stable flight, but do add some complexity to the control scheme.

Due to the fact that quadrotors come in a variety of shapes and sizes there are often model uncertainties. Also, changing payloads can alter the model parameters. To overcome these problems some research has been done in applying adaptive control to the quadrotor stability problem. In [30] and [31] model predictive and adaptive control are used to effectively control the quadrotor and overcome model uncertainties. Experiments performed in [32] show that both model reference adaptive control and PID control perform well in practice.

Since the controller must be implemented on low power embedded hardware, a controller with limited complexity is desirable. Additionally the payload and quadrotor model will be constant throughout the experiment. In [33] and [34] a PD like controller in quaternion

form is developed and shown to perform well. Since the attitude estimation in this case is to be done in quaternion form, applying this type of control does not require any conversion of the attitude and the controller's simplicity and lack of dependence on the system model make it well suited to practical implementation.

1.4.3 Autonomous Landing

There have been numerous attempts over the years to develop a system for autonomous landing on a moving platform. Methods have been developed with and without the aid of vision systems. These systems rely on tether guides, predictive models or simply known motion of the platform. All of these methods require some sort of knowledge of the motion beforehand. Systems incorporating vision have been attempted using visual servoing and more recently optical flow.

In [35] the authors use a model of ship motion to develop a landing controller. The motion is modelled as a fixed number of superimposed sinusoids having unknown frequencies, amplitudes and phases. The authors designed an internal-model based dynamic regulator that secures global convergence to the zero error manifold. The controller was tested in simulation and found to be robust against typical uncertainties.

In [36] the authors concentrate on building predictive models of the ship motion. Simulation results show accurate prediction for the tests completed. The paper does not propose a controller for the actual landing manoeuvre.

In [37] a tether is used to connect the landing craft to the moving deck. Controllers are developed for translational motion, attitude and altitude. Position control is used first to bring the helicopter to the desired place and then the landing phase in initiated. The work considers a conventional helicopter, not a quadcopter, however the goal is still autonomous landing. Successful operation is achieved in simulation, however practical implementation was not complete at the time the paper was published.

In [38] a vision system is introduced to aid in the landing manoeuvre. A known target image is used to bring the UAV to the required position and a sonar is used in the final

landing stage. The technique was tested on a helicopter system with positive results.

The work in [39] again uses a vision system to land on a known target. The system is comprised of off-the-shelf hardware using custom software. The proposed control is tested on a conventional model helicopter in real-time and good results are obtained.

1.5 Thesis Contribution

In this thesis a framework for estimation and control of a quadrotor is presented, implemented and tested. Estimation and control methods are reviewed and the most suitable methods for implementation on a common low-cost platform are chosen and shown to work well even in the presence of measurement noise. The estimation and attitude control of the quadrotor is all done on the embedded system, making the platform completely self-contained and not reliant on external vision systems or computing. The completed system is used as a test bed for position and landing control and can also be used for future work in the lab.

Secondly a method of position control and autonomous landing on a moving platform is implemented and tested. The method chosen uses an airframe mounted camera to capture images and optical flow is calculated to be used as feedback in the controllers. Work in the thesis shows that this method can be effectively implemented in practice but requires off-board computations due to the required image processing.

1.6 Thesis Outline

This thesis is divided into 6 chapters. Chapter 2 covers some background information, introduces the system model and provides an overview of optical flow. Chapter 3 describes the experimental setup and some calibration procedures. Chapter 4 discusses attitude estimation and the various controllers as implemented on the practical system. Chapter 5 presents the simulation and experimental results obtained. Chapter 6 summarizes the main conclusions of the thesis and suggests future work.

Chapter 2

Theory

2.1 Attitude Representations

There have been numerous attitude representations introduced over the years. Each representation has its own advantages and disadvantages. In this thesis, we make use of the Euler angles representation, the rotation matrix representation and the unit quaternion representation, hence only these three are discussed in detail. For more information regarding the other representations, a more detailed survey of many more representations is provided in [40]. The goal of each attitude representation is to allow for a relationship between the defined inertial, or reference, frame and the body-fixed frame using the fewest number of parameters and avoiding singularities. Also, the representation should allow for mathematical manipulation in a convenient way.

In Fig. 2.1.1, the inertial frame $\{I\}$ is described by the orthogonal unit vectors, $\hat{x}_I$, $\hat{y}_I$, $\hat{z}_I$. Although the definition of the inertial reference frame is arbitrary, it is common to establish the frame with the $\hat{z}_I$ vector orthogonal to the ground, the $\hat{x}_I$ vector pointing in the North direction and $\hat{y}_I$ vector in the West direction. The body fixed frame $\{B\}$ is described by the orthogonal unit vectors $\hat{x}_B$, $\hat{y}_B$, $\hat{z}_B$.

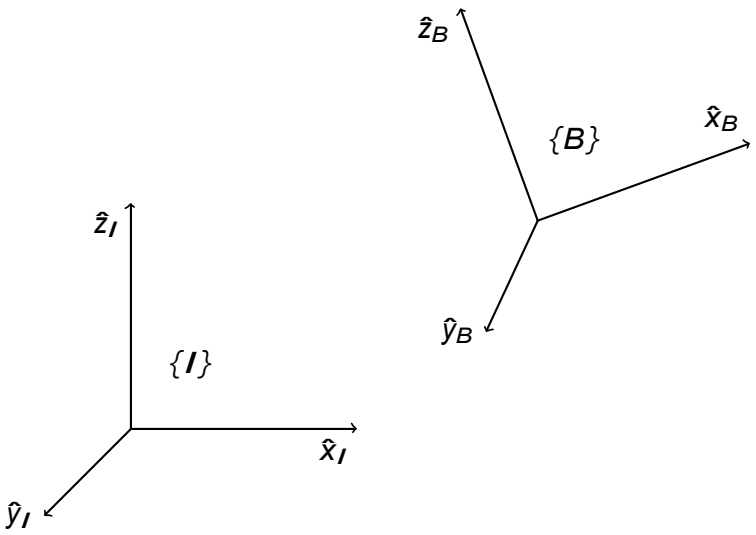

Figure 2.1: Inertial frame I, body-fixed frame B

2.1.1 Euler Angles

Introduced by Leonard Euler, the Euler angles are designed to describe the rotation of a rigid body in space. The main advantage of the Euler angles is the clear physical meaning of each angle allowing for easy visualization of the orientation given a set of angles. In order to fully describe the orientation of a body in three dimensional Euclidean space, three parameters are necessary. Numerous three-dimensional attitude parametrizations have been presented over the years, see [41] and [40], but Euler angles have remained very popular. However, similar to the other parametrizations, it can be shown that it cannot be both non-singular and unique. In common terminology, the Euler angles (φ, θ, ψ) are known as roll, pitch and yaw of the rigid body, where φ, θ and ψ define a positive rotation about x, y and z axes respectively. The rotation matrix can be defined in terms of three consecutive rotations about the given axes in the specific order of rotation. The possible sets of consecutive rotations are commonly divided in symmetric and asymmetric sets [40] as follows.

Symmetric sets:

$$z-x-z, \quad x-z-x$$
$$x-y-x, \quad y-x-y$$
$$y-z-y, \quad z-y-z$$

Asymmetric sets:

$$x-y-z, \quad z-y-x$$
$$y-z-x, \quad x-z-y$$
$$z-x-y, \quad y-x-z$$

The symmetric sets of rotations are not well suited to describing the orientation of an aircraft since a singularity occurs when the second rotation is 0° or 180°. This happens to occur at the level reference orientation of the aircraft. The singularity cannot be removed with Euler angles but can be moved to a location which is more convenient. Choosing a set of rotations from the Asymmetric sets allows for the singularity to be moved. The order of rotations in this case is $z \rightarrow y \rightarrow x$. This provides the orientation about the body fixed frame axes. If the $x \rightarrow y \rightarrow z$ order is chosen the orientation is about the inertial frame axes. Choosing this roll - pitch - yaw sequence, one can obtain the rotation matrix as

$$R = R_z(\psi)R_y(\theta)R_x(\varphi)$$

$$= \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\varphi & -s\varphi \\ 0 & s\varphi & c\varphi \end{bmatrix} \tag{2.1}$$

$$= \begin{bmatrix} c\theta c\psi & s\theta s\varphi c\psi - s\psi c\varphi & s\theta c\varphi c\psi + s\psi s\varphi \\ c\theta s\psi & s\theta s\varphi s\psi + c\psi c\varphi & s\theta c\varphi s\psi - c\psi s\psi \\ -s\theta & c\theta s\varphi & c\theta c\varphi \end{bmatrix}$$

where s and c denote the sine and cosine of the respective angles.

The extraction of the Euler angles from the rotation matrix, results in a singularity at $\theta = \pm\pi/2$. There is no unique solution for yaw and roll at this singular configuration. Therefore, Euler angles representation is not a global parametrization of the attitude. However, it is easier to imagine the orientation of a rigid body when the values of roll, pitch and yaw are provided. The rotation matrix and quaternion representation fail to provide such insight into the physical meaning of the orientation.

2.1.2 Rotation Matrix

The rotation matrix, also known at the direction cosine matrix (DCM) is the most popular representation of the attitude of a rigid body. The rotation matrix belongs to the special orthogonal group of matrices SO(3). The rotation matrix allows for easy computation of any number of rotations through simple matrix multiplication. The major drawback of the rotation matrix is the 9 parameter representation. This can result in a high number of computations when dealing with many rotations.

Consider the orientation of $\{B\}$ with respect to $\{I\}$. It can be described by three vectors:

$$ {}^I\hat{X}_B = \begin{bmatrix} r_{11} \\ r_{12} \\ r_{13} \end{bmatrix} \quad {}^I\hat{Y}_B = \begin{bmatrix} r_{21} \\ r_{22} \\ r_{23} \end{bmatrix} \quad {}^I\hat{Z}_B = \begin{bmatrix} r_{31} \\ r_{32} \\ r_{33} \end{bmatrix} \tag{2.2} $$

The components of each of these vectors are the projection of each vector on the unit direction of the frame $\{I\}$. The rotation matrix of frame $\{B\}$ with respect to frame $\{I\}$ is defined as

$$ {}^I_B R = \begin{bmatrix} {}^I\hat{X}_B & {}^I\hat{Y}_B & {}^I\hat{Z}_B \end{bmatrix}^{-1} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.3} $$

where each component is defined as

$$ {}^{I}_{B}R = \begin{bmatrix} \hat{x}_B \cdot \hat{x}_I & \hat{y}_B \cdot \hat{x}_I & \hat{z}_B \cdot \hat{x}_I \\ \hat{x}_B \cdot \hat{y}_I & \hat{y}_B \cdot \hat{y}_I & \hat{z}_B \cdot \hat{y}_I \\ \hat{x}_B \cdot \hat{z}_I & \hat{y}_B \cdot \hat{z}_I & \hat{z}_B \cdot \hat{z}_I \end{bmatrix} \tag{2.4} $$

where $u \cdot v$ represents the dot product $u \cdot v = ||u||\ ||v||\ cos(\Theta)$ and Θ is the angle between the vectors u and v.

For a rotation about the x axis by an angle φ the rotation matrix is given as

$$ R_x(\varphi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\varphi & -sin\varphi \\ 0 & sin\varphi & cos\varphi \end{bmatrix} \tag{2.5} $$

Similarly for rotation about the y and z axes the rotation matrices are

$$ R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix} \tag{2.6} $$

$$ R_z(\psi) = \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.7} $$

Obtaining a rotation matrix for an Euler angles sequence requires only the multiplication of 3 rotation matrices. The resulting rotation matrix from the $x - y - z$ Euler angles sequence and the one referred to in the remainder of this thesis is

$$ R = R_z(\psi)R_y(\theta)R_x(\varphi) = \begin{bmatrix} c\theta c\psi & s\theta s\varphi c\psi - s\psi c\varphi & s\theta c\varphi c\psi + s\psi s\varphi \\ c\theta s\psi & s\theta s\varphi s\psi + c\psi c\varphi & s\theta c\varphi s\psi - c\psi s\psi \\ -s\theta & c\theta s\varphi & c\theta c\varphi \end{bmatrix} \tag{2.8} $$

2.1.3 Unit Quaternion

A third representation of the attitude of a rigid body is the unit quaternion. Compared to the rotation matrix the unit quaternion representation uses fewer parameter, 4 instead of 9, to represent the orientation. The unit quaternion is globally non-singular but non-unique. It is the preferred method among many researchers.

The unit quaternion is defined as

$$Q = \begin{bmatrix} q_0 \\ q \end{bmatrix} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\frac{\gamma}{2} \\ \hat{k}_x \sin\frac{\gamma}{2} \\ \hat{k}_y \sin\frac{\gamma}{2} \\ \hat{k}_z \sin\frac{\gamma}{2} \end{bmatrix} \tag{2.9}$$

where γ is the angle of rotation around the unit vector $\hat{k} = (\hat{k}_x\ \hat{k}_y\ \hat{k}_z)$ as shown in Fig. 2.2 where reference point p_1 is rotated to p_2.

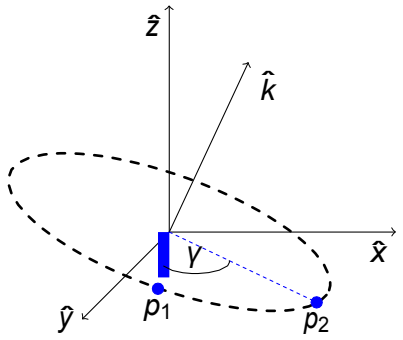

Figure 2.2: Definition of quaternion

From this definition it is obvious that the quaternion is composed of a scalar part, q_0 and a vector part q. The unit quaternion is subject to the following constraint

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = q_0^2 + q^T q = 1 \tag{2.10}$$

Both the Euler angles representation and rotation matrix can be converted to and from the quaternion representation. The unit quaternion can be converted to a rotation matrix through the use of the Rodrigues formula

$$
\begin{aligned}
R(Q) &= \boldsymbol{I}_3 + 2S(q)^2 - 2q_0 S(q) \\
&= \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & -2q_0q_3 + 2q_1q_2 & 2q_0q_2 + 2q_1q_3 \\ 2q_0q_3 + 2q_1q_2 & q_0^2 - q_1^2 + q_2^2 - q_3^2 & -2q_0q_1 + 2q_2q_3 \\ -2q_0q_2 + 2q_1q_3 & 2q_0q_1 + 2q_2q_3 & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}
\end{aligned} \tag{2.11}
$$

where $S(x)$ is a skew-symmetric matrix associated with $x \in \mathrm{R}^3$. The skew-symmetric matrix is defined as

$$
S(x) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{2.12}
$$

with $x = (x_1, x_2, x_3)^T \in \mathrm{R}^3$. Given a rotation matrix R and two vectors $x, y \in \mathrm{R}^3$, we have the following useful properties: $S(x)y = -S(y)x = x \times y$, $S(x)x = 0$, $S(x)S(y) = yx^T - (x^Ty)\boldsymbol{I}_3$ and $S(Rx) = RS(x)R^T$, where $\times$ denotes the vector cross product [42].

To extract the unit quaternion from a rotation matrix R, a method is provided by [43]

$$
\begin{aligned}
q_0 &= \pm\tfrac{1}{2}(1 + R_{11} + R_{22} + R_{33})^{\frac{1}{2}} \\
q &= \frac{1}{4q_0} \begin{bmatrix} R_{23} - R_{32} \\ R_{31} - R_{13} \\ R_{12} - R_{21} \end{bmatrix}
\end{aligned} \tag{2.13}
$$

However, if $q_0 = 0$, the vector component should be calculated as shown in [43] to avoid numerical inaccuracies when q_0 is close to 0.

$$
q = \frac{1}{4q_0} \begin{bmatrix} \pm(\frac{1+R_{11}}{2})^{\frac{1}{2}} \\ \pm(\frac{1+R_{22}}{2})^{\frac{1}{2}} \\ \pm(\frac{1+R_{33}}{2})^{\frac{1}{2}} \end{bmatrix} \tag{2.14}
$$

It is also useful to have an equation to obtain the unit quaternion from the Euler angles since that is still a popular representation. To obtain the quaternion from the Euler

angles the following equation can be used

$$\begin{bmatrix} q_0 \\ q \end{bmatrix} = \begin{bmatrix} \cos(\frac{\varphi}{2})\cos(\frac{\theta}{2})\cos(\frac{\psi}{2}) + \sin(\frac{\varphi}{2})\sin(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \sin(\frac{\varphi}{2})\cos(\frac{\theta}{2})\cos(\frac{\psi}{2}) - \cos(\frac{\varphi}{2})\sin(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \cos(\frac{\varphi}{2})\sin(\frac{\theta}{2})\cos(\frac{\psi}{2}) + \sin(\frac{\varphi}{2})\cos(\frac{\theta}{2})\sin(\frac{\psi}{2}) \\ \cos(\frac{\varphi}{2})\cos(\frac{\theta}{2})\sin(\frac{\psi}{2}) - \sin(\frac{\varphi}{2})\sin(\frac{\theta}{2})\cos(\frac{\psi}{2}) \end{bmatrix} \tag{2.15}$$

Eq. (2.15) is obtained through the multiplication of three quaternions each representing a rotation about one of the axes. This is similar to how rotation matrices can be multiplied to obtain the three axis orientation. In fact, the order of multiplication follows the same rules as described in the rotation matrix section.

$$\begin{bmatrix} \tilde{q}_0 \\ q \end{bmatrix} = Q_\psi \otimes Q_\theta \otimes Q_\varphi = \begin{bmatrix} \cos(\frac{\psi}{2}) \\ 0 \\ 0 \\ \sin(\frac{\psi}{2}) \end{bmatrix} \otimes \begin{bmatrix} \cos(\frac{\theta}{2}) \\ 0 \\ \sin(\frac{\theta}{2}) \\ 0 \end{bmatrix} \otimes \begin{bmatrix} \cos(\frac{\varphi}{2}) \\ \sin(\frac{\varphi}{2}) \\ 0 \\ 0 \end{bmatrix} \tag{2.16}$$

Similar to rotation matrices, unit quaternion representation can be used to combine two or more unit quaternions to describe the overall attitude of a moving body. It can also be used to transform a vector from one frame to another. Let $Q_1 = (q_{0,1}, q_1)$ and $Q_2 = (q_{0,2}, q_2)$, be two unit quaternions. Then the quaternion product $Q_3 = (q_{0,3}, q_3)$ is given by

$$Q_3 = Q_1 \otimes Q_2 = \begin{bmatrix} q_{0,1}q_{0,2} - q_1^T q_2 \\ q_{0,1}q_2 + q_{0,2}q_1 + q_1 \times q_2 \end{bmatrix} \tag{2.17}$$

where $\otimes$ denotes the quaternion multiplication and $\times$ denotes the vector cross product. The unit quaternion multiplication is non-commutative.

The inverse of a unit quaternion $Q = (q_0, q)$ is given by $Q^{-1} = (q_0, -q)$, where

$$Q \otimes Q^{-1} = Q^{-1} \otimes Q = (1, \mathbf{0}) \tag{2.18}$$

The quaternion representation $Q = (\pm 1, \mathbf{0})$ is equivalent to $R = \boldsymbol{I}$.

The unit quaternion multiplication can also be used to transform a vector from one reference frame to another. Let a_I be a vector expressed in the inertial frame $\boldsymbol{I}$ and a_B

be the vector projection of a, in the body frame B. Then,

$$\bar{a}_B = Q \otimes \bar{a}, \otimes Q^{-1} \tag{2.19}$$

where $\bar{x} = (0, x)$, $x \in \mathrm{R}^3$

The quaternion representation has some advantages over the other attitude representations. Its reduced parameter representation makes it more suitable for implementation on practical systems. As opposed to the rotation matrix, which has 9 elements, the quaternion uses only 4 elements resulting in a reduced computational load. Even though the Euler angles representation is more computationally efficient than the unit quaternion representation, the Euler angle representation is somewhat ineffective as it is a non-global representation.

The unit quaternion provides a non-singular representation of the attitude. Despite its advantages there are certain drawbacks. The quaternion representation is an over-parametrization of the rotation space $SO(3)$, meaning that both unit quaternions Q and $-Q$ represent the same rotation matrix (i.e. $R(Q) = R(-Q)$). Hence, the unit quaternion is not a unique representation of attitude.

2.2 Quadrotor Model

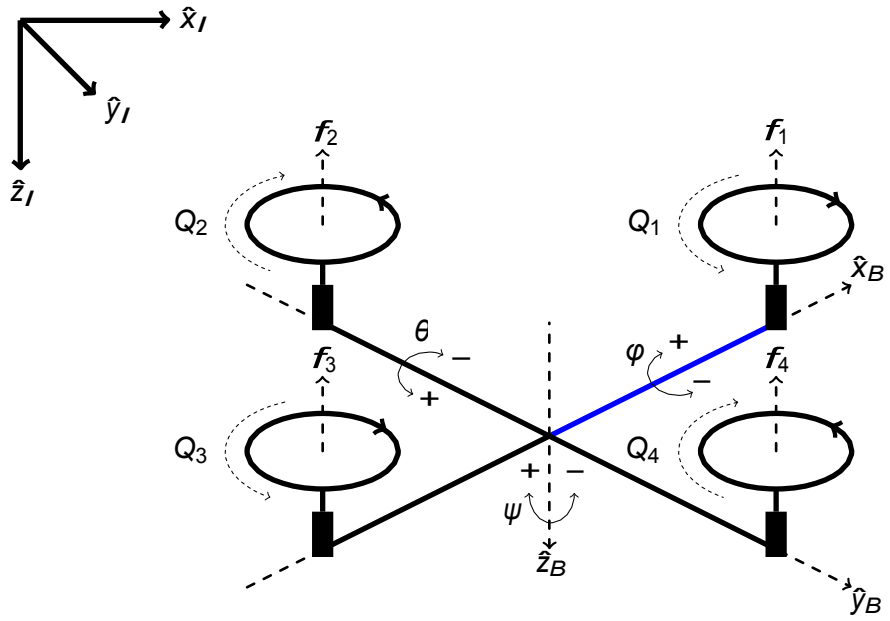

Figure 2.3: Quadrotor model

The quadrotor UAV consists of a rigid frame of four arms joined at the centre. At the end of each arm is a rotor attached to a motor as shown in Fig. 2.3. The front of the quadrotor is commonly marked using a different colour arm. Any arm can be taken as the front, the front rotor is simply defined in the software. Traditional helicopters change the axis of rotation of the main rotor as well as the pitch of the blades to control roll, pitch and thrust. The quadrotor platform does not have these capabilities, the axis of rotation and pitch of the rotor blades are fixed. Also, the rotors spin in a fixed direction, as illustrated in Fig. 2.3, that cannot be changed. As a result, manoeuvres requiring negative thrust cannot be achieved with this configuration.

The motion of the quadrotor is achieved through the control of the speed of the individual motors. Each rotor generates an upward thrust and a torque about its centre of rotation. Each propeller produces a drag force opposite to the vehicle's direction of flight. If individual rotor angular velocities are the same, with left and right rotors rotating clockwise and front and rear rotors counter-clockwise, the net torque about the

yaw axis is exactly zero. This implies that the yaw stabilizing rotor of conventional helicopters is not needed. In a hovering condition all four rotors have the same angular velocity such that the net torques around all three axes are zero. To move vertically, the angular velocity of rotors are increased or decreased to increase or decrease thrust. To obtain a roll motion the angular velocity of the left (or right) rotor is increased while the opposite one is decreased. To obtain a pitch motion the angular velocity of the front (or back) rotor is increased while the opposite one is decreased. In either roll or pitch motion the increase and decrease of thrust in the opposing rotors is the same in order to maintain the same overall thrust. To obtain the yaw motion the reactive torque of the rotors is used instead of the thrust. Since the front and back rotors spin in the same direction, their reactive torques combine. Similarly the reactive torques of the left and right rotors combine, but act in an opposite direction to the front and back pair. When the reactive torque from each of the pairs is equal, there is no yaw motion. To achieve the yaw motion, the angular velocity of one pair is increased and the other is decreased. Again, the increase and decrease are equal to maintain the same overall thrust. Fig. 2.4 graphically summarizes how motion is obtained. The only way to achieve translational motion is to control the attitude to direct the thrust.

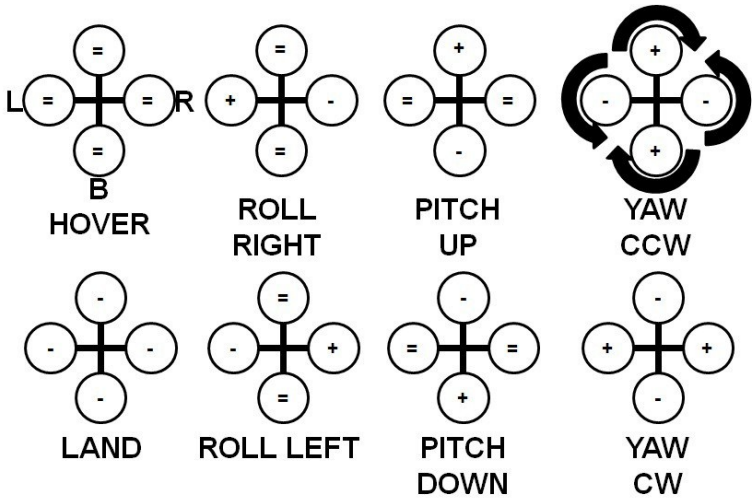

Figure 2.4: Illustrated principle of control for quadrotor

Let $I = \{\hat{x}_I, \hat{y}_I, \hat{z}_I\}$ denote an inertial frame, and $B = \{\hat{x}_B, \hat{y}_B, \hat{z}_B\}$ denote a frame rigidly attached to the aircraft as shown in Fig. 2.3. Then the dynamical model of a quadrotor as described in [44] and [34] using Newton's equations of motion is given by

$$\dot{p} = v \tag{2.20}$$

$$\dot{v} = g\hat{z}_I - \frac{1}{m} T R \hat{z}_I \tag{2.21}$$

$$\dot{R}^{I_F} = RS(\Omega) \tag{2.22}$$

$$\dot{\Omega} = -\Omega \times I_F \Omega - G_a + \tau_a \tag{2.23}$$

$$I_r \dot{\omega} = \tau_i - Q_i, \quad i \in 1, 2, 3, 4 \tag{2.24}$$

$$T = \sum_{i=1}^{4} |f_i| = b \sum_{i=1}^{4} \omega_i^2 \tag{2.25}$$

$$G_a = \sum_{i=1}^{4} I_r (\Omega \times e_z) (-1)^{i+1} \omega_i \tag{2.26}$$

$$Q_i = \kappa \omega_i^2 \tag{2.27}$$

The notations used in Eq. (2.20) to Eq. (2.27) are defined in Table 2.1. Eq. (2.22) can also be expressed by the unit-quaternion representation [42] as,

$$\dot{Q} = \frac{1}{2} Q \begin{bmatrix} 0 \\ \Omega \end{bmatrix} \tag{2.28}$$

and by Euler angles representation [45] as,

$$\begin{aligned} \dot{\varphi} &= \Omega_1 + \Omega_2 \sin\varphi \tan\theta + \Omega_3 \cos\varphi \tan\theta \\ \dot{\theta} &= \Omega_2 \cos\varphi - \Omega_3 \sin\varphi \\ \dot{\psi} &= \Omega_2 \sin\varphi \sec\theta + \Omega_3 \cos\varphi \sec\theta \end{aligned} \tag{2.29}$$

Symbol	Definition
m	mass of airframe
g	acceleration due to gravity $= 9.81 m/s^2$
$\hat{z}_I$	$(0, 0, 1)^T$ unit vector in I
$p = (x, y, z)^T$	position of the origin of the body fixed frame B with respect to I
$v = (\dot{x}, \dot{y}, \dot{z})^T$	linear velocity vector of the origin of B expressed in I
T	total thrust
R	rotation matrix describing orientation of the airframe
$S(x)$	skew symmetric operator as given by Eq. (2.12)
$\Omega = (\Omega_1, \Omega_2, \Omega_3)^T$	angular velocity of quadrotor in B
I_f	symmetric positive-definite constant inertia matrix of the airframe with respect to the frame B whose origin is at the centre of mass
$\times$	vector cross product
$G_a = (G_{a,1}, G_{a,2}, G_{a,3})^T$	gyroscopic torques
$\tau_a = (\tau_{a,1}, \tau_{a,2}, \tau_{a,3})^T$	airframe torques
ω_i	angular velocity of motor i
τ_i	torque produced by motor i
Q_i	reactive torque generated in free air by rotor i due to rotor drag
κ	positive proportionality constant that relates reactive torque its respective angular velocity
b	positive proportionality constant that relates total thrust to the sum of angular velocity
f_i	lift generated by rotor i in free air

Table 2.1: Symbol definitions for quadrotor model

2.3 Optical Flow

The vision system used in this thesis is based upon the calculation of optical flow. In nature, it has been suggested [46] that honeybees and other flying insects rely on visual flow for navigation. Optical flow measurement provides relative velocity and proximity information with respect to the local environment [47]. For this reason it is a popular technique used in landing-control, terrain following, and object avoidance. Given two subsequent images, one can determine the optical flow vector at any point in the image. The optical flow vector represents a scaled measure of the velocity of that point from the first frame to the second. It can become computationally intensive to compute optical flow at many points on an image and even more so for a continuous sequence of images. For these reasons optical flow is rarely calculated using the UAVs embedded hardware. A better approach to overcome the limited hardware is to calculate the optical flow on a remote computer and relay the results to the UAV. There are many methods proposed in literature to calculate optical flow. A brief overview of the Lucas-Kanade method chosen for implementation is given. Some unique properties can be exploited if the optical flow is calculated in spherical coordinates, this process is also discussed. Finally, the inertial average optical flow measurement as proposed in [48] is outlined and implementation of optical flow calculation in real-time is discussed.

2.3.1 Calculating Optical Flow

Over the years many methods have been developed to calculate optical flow. Work in [49] provides an intensive comparison of many of the methods. The Lucas-Kanade [50] method has proven to be a popular, accurate and efficient choice for many researchers such as [48] and [44]. Since this is a proven method, the Lucas-Kanade method is chosen for implementation in this thesis. An overview of the method is provided here.

In simple terms, the Lucas-Kanade method is a technique to provide an estimate of the movement of features in two successive image frames. There are two important assumptions made by the algorithm:

- The two successive images should be separated by a small time, Δt, so that there is not a large displacement.
- The image should contain a textured object with varying shades of grey having different intensity levels. The algorithm is not sensitive to colour.

The algorithm essentially works by guessing which way the pixels have moved in order to explain the change in intensity. Consider the movement vector (u, v) corresponding to a given pixel (x, y). Define the increase in brightness per pixel at pixel (x, y) as $I_x(x, y)$ in the x-direction and as $I_y(x, y)$ in the y-direction. Then the total increase in brightness after a movement by u pixels in the x-direction and v pixels in the y-direction is

$$I_x(x, y)u + I_y(x, y)v \tag{2.30}$$

This is equal to the local difference in intensity defined as $I_t(x, y)$ resulting in

$$I_x(x, y)u + I_y(x, y)v = -I_t(x, y) \tag{2.31}$$

One pixel does not contain enough information to achieve a useful matching result. An $n \times n$ neighbourhood of pixels is used around the given pixel. For this example consider a 3×3 neighbourhood. The result is a series of 9 linear equations

$$I_x(x + \Delta x, y + \Delta_y)u + I_y(x + \Delta x, y + \Delta_y)v = -I_t(x + \Delta x, y + \Delta_y) \tag{2.32}$$

for $\Delta x = -1, 0, 1$ and $\Delta y = -1, 0, 1$. In matrix form Eq. (2.32) can be written as

$$A \begin{bmatrix} u \\ v \end{bmatrix} = b \tag{2.33}$$

where

$$A = \begin{bmatrix} I_x(x+(-1),y+(-1)) & I_y(x+(-1),y+(-1)) \\ I_x(x+(-1),y+(0)) & I_y(x+(-1),y+(0)) \\ I_x(x+(-1),y+(1)) & I_y(x+(-1),y+(1)) \\ I_x(x+(0),y+(-1)) & I_y(x+(0),y+(-1)) \\ I_x(x+(0),y+(0)) & I_y(x+(0),y+(0)) \\ I_x(x+(0),y+(1)) & I_y(x+(0),y+(1)) \\ I_x(x+(1),y+(-1)) & I_y(x+(1),y+(-1)) \\ I_x(x+(1),y+(0)) & I_y(x+(1),y+(0)) \\ I_x(x+(1),y+(1)) & I_y(x+(1),y+(1)) \end{bmatrix} \tag{2.34}$$

and

$$b = \begin{bmatrix} -I_t(x+(-1),y+(-1)) \\ -I_t(x+(-1),y+(0)) \\ -I_t(x+(-1),y+(1)) \\ -I_t(x+(0),y+(-1)) \\ -I_t(x+(0),y+(0)) \\ -I_t(x+(0),y+(1)) \\ -I_t(x+(1),y+(-1)) \\ -I_t(x+(1),y+(0)) \\ -I_t(x+(1),y+(1)) \end{bmatrix} \tag{2.35}$$

To solve for the pixel motion a least squares method is used resulting in

$$\begin{bmatrix} u \\ v \end{bmatrix} = (A^T A)^{-1} A^T b \tag{2.36}$$

This process is done for each pixel of interest in the image. For a large image with many pixels, the process can become computationally intensive.

2.3.2 Optical Flow in Spherical Coordinates

There are some unique advantages to computing optical flow in spherical coordinates. In spherical coordinates, optical flow exhibits a passivity like property as described in [44]. It is also shown that it is in fact not necessary to implement a spherical camera,

the spherical projection can be implemented numerically using a conventional pin-hole camera. Additionally, the focus of expansion and the focus of contraction are guaranteed to exist when using spherical coordinates, this is not always true for a planar retina [51]. The work in [51] shows that it is possible to numerically calculate the spherical coordinates from the image plane. Once this is completed on the planar flow field data the spherical form of optical flow can be considered. The optical flow equation in spherical coordinates is described below. Fig. 2.5 illustrates the various parameters used.

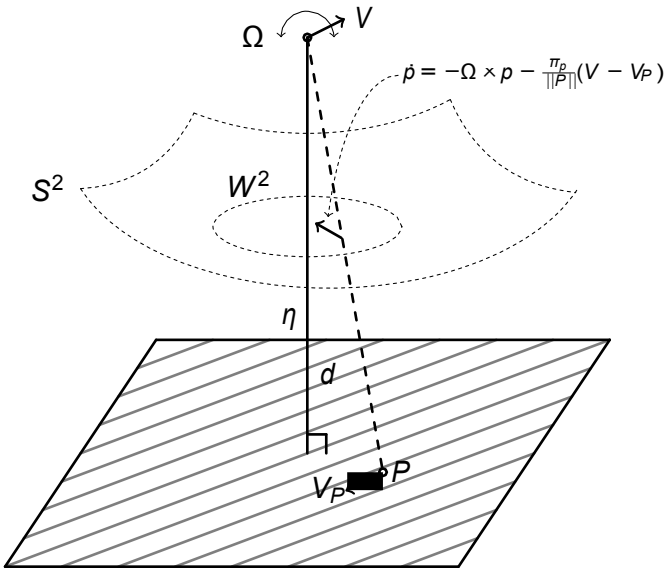

Figure 2.5: Illustration of spherical optical flow

Let $P = (X, Y, Z) \in \mathrm{R}^3$ be a visible target point expressed in the camera frame. The point p observed on the image is given by

$$p = \frac{P}{||P||} \tag{2.37}$$

describing the projection of point P onto the surface S^2 of the camera.

Taking the time derivative of p, one can obtain the kinematics of the image point which is called the optical-flow equation for a spherical lens surface. The derivative is given as

$$\dot{p} = -\Omega \times p - \frac{\pi_p}{||P||}(V - V_P) \tag{2.38}$$

where $\pi_p = (\boldsymbol{I}_3 - pp^T)$, the vectors $V = R^T v$ and V_p are expressed in the body-fixed frame and represent, respectively, the translational velocity of the vehicle and the target point. If $d(t)$ is defined as the orthogonal distance from the origin of the body-fixed frame to the surface, and θ is the angle between the inertial direction and the observed target point p then $||P|| = \frac{d(t)}{\cos(\theta)}$ and Eq. (2.38) becomes

$$\dot{p} = -\Omega \times p - \frac{\cos\theta}{d(t)}\pi_p(V - V_P) \tag{2.39}$$

2.3.3 Inertial Average Optical Flow

In order to obtain a useful measure of the optical flow, one approach is to integrate the optical flow around the spherical lens. The surface W^2 is the solid angle representing the spherical surface of the lens. Integrating Eq. (2.39) we obtain.

$$\varphi = \int_{W^2} \dot{p}dp = -\pi(\sin\theta_0)^2\Omega \times R^T\eta - \frac{Q(V - V_t)}{d} \tag{2.40}$$

where $Q = R^T(R_t\Lambda R_t^T)R$ and R_t is the orientation of the target platform. Parameter θ_0 and matrix Λ are constant values that depend on the solid angle W^2.

The value of φ is obtained using image processing methods. The angular velocity Ω and orientation R are obtained using the IMU and attitude estimation techniques. The value of interest is the relative velocity. Solving for $\frac{V-V_t}{d} := w$ one obtains the inertial average optical flow as follows

$$w = -(R_t\Lambda^{-1}R_t^T)R(\varphi + \pi(\sin\theta_0)^2\Omega \times R^T\eta) \tag{2.41}$$

It is clear that when expressed with respect to rigid-body motion, the calculated value w is equal to the relative velocity divided by the distance to the ground plus some noise that is unavoidable in the measurement process.

$$w = \frac{v - v_t}{d} + noise \tag{2.42}$$

where $v_t = RV_t$ is the translational velocity of the target platform and $v = RV$ is the translational velocity of the rigid body, each expressed in frame $\boldsymbol{I}$.

2.3.4 Real-Time Implementation

In practice, it is necessary to measure the optical flow and calculate the inertial average optical flow. Many methods to calculate the optical flow exist. The Lucas-Kanade method is very popular in the literature and provides excellent accuracy and efficiency. Additionally, the pyramidal approach can be used to enhance the results. The number of points on which the optical flow is calculated is directly related to the quality of the measurement and inversely related to the time it takes to calculate the optical flow. A compromise must be made in order to achieve good results while still maintaining a high enough sample rate. To calculate the optical flow and to perform all of the image based processing OpenCV 2.4.4 was used. OpenCV is an image processing library for C++. It provides many optimized functions for dealing with images and many functions are available. In fact, the OpenCV library provides an implementation for both the Lucas-Kanade method and Horn and Shunck method. It also allows for simple manipulation of the image data making it ideal for this application. In practice the optical flow can be enhanced by slightly defocussing the image to act as a low pass filter. Additionally, filtering of outliers helps to improve the results and reduce errors in the measurement. To limit the effects of shadows and lighting on the images, histogram equalization is used to improve the contrast of the image. Results of the optical flow processing are shown in Fig. 2.7 and Fig. 2.8. The images show calculated optical flow fields for a translational motion and a vertical motion. Once the flow fields are obtained, the inertial average optical flow is calculated and a single numerical value is obtained for each axis. The optical flow field was calculated for 208 points and the entire program loop for optical flow calculation is run at 25Hz. The overall function of the optical flow program is represented by the flowchart in Fig. 2.6.

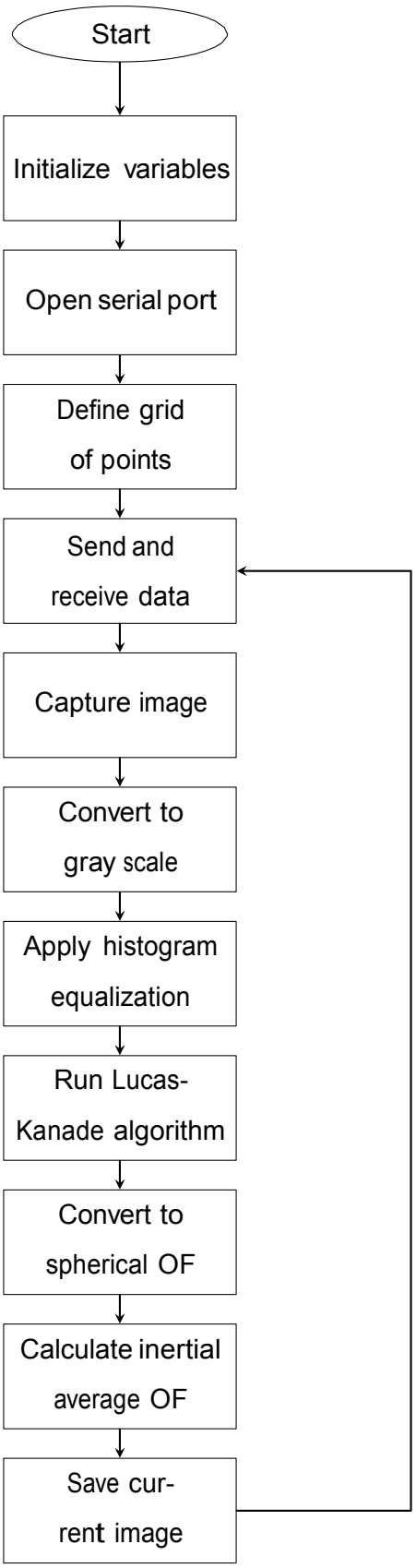

Figure 2.6: Program flow chart for calculating optical flow

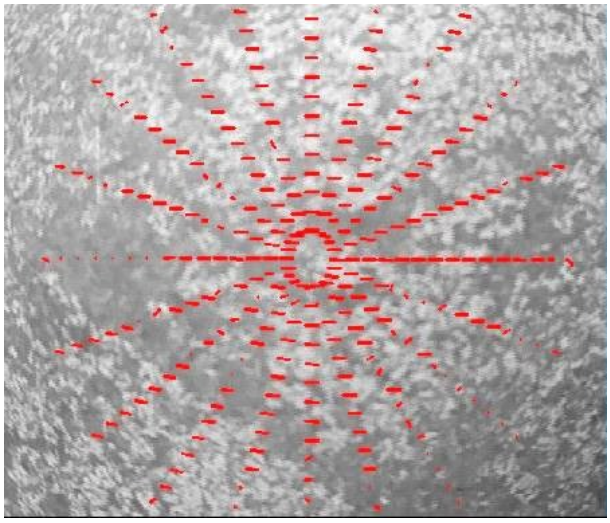

Figure 2.7: Calculated optical flow field for translational motion

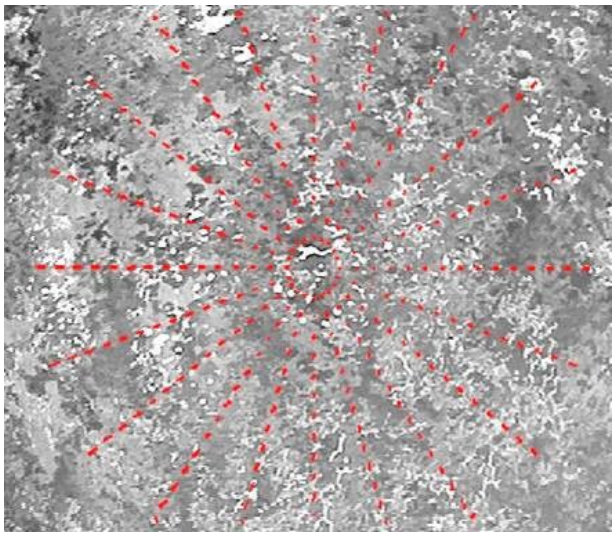

Figure 2.8: Calculated optical flow field for vertical motion

Chapter 3

Experimental Setup

Figure 3.1: Quadrotor aircraft experimental setup

3.1 Airframe and Electronics

The experimental setup is based on the popular Arducopter platform and is shown in Fig. 3.1. The Arducopter project is an open source project dealing with both the hardware and software of the quadrotor. The setup used for experimentation uses the standard Arducopter hardware with custom firmware written in Arduino and C++. Specific details about the individual components are provided in the remainder of this section.

3.1.1 Frame

The quadcopter frame used is the 3DR Arducopter frame. It consists of 4 aluminum arms, fibreglass mounting platforms and fibreglass landing legs. The frame is available as a ready to assemble kit having all holes for mounting motors and other electronics pre-drilled. The kit is designed to be modular such that repairs and additions are easily completed. Parts are widely available and the simple design allows for easy customization.

The 3DR frame is widely used amongst hobbyists and researchers. It has logged many hours of successful flights. Since the focus of this thesis is not the design of the mechanical aspect of the quadrotor, the use of this ready-to-fly frame allows for more effort to be placed on the control aspects.

3.1.2 Processing and Inertial Measurement Unit

The processor and inertial measurement unit (IMU) used are both contained in the Arducopter APM 2.5 module available from 3DR robotics. The APM 2.5 is the heart of the quadrotor. It performs all of the computations and contains the inertial sensors. The APM 2.5 is a low-cost system and is a proven platform. The APM 2.5 is available as a ready-to-fly solution, however in this case custom firmware is used with the existing electronics.

The APM 2.5 contains several integrated circuits onboard. The main controller is the Atmel Atmega 2560 microcontroller. This is a low cost, low power, 8-bit microcontroller. The Atmega 2560 includes 256KB flash memory, 8KB SRAM and 4KB EEPROM. It has 86 general purpose I/O lines, 4 USARTs, serial peripheral interface (SPI) and inter-integrated circuit (I^2C) interface to communicate with the sensors and peripherals. Also available are real-time counters, six flexible timer/counters with compare modes and hardware and software generated PWM for generating motor outputs and demodulating input radio signals. The device operates at 16 MHz between 4.5-5.5 volts which is sufficient for the implementation of estimation and control algorithms as well as communication functions. For more details refer to [52].

The APM also includes a secondary microcontroller to offset some low level processes. The Atmel Atmega 32U2 is used to offload the radio input and motor output generation from the Atmega 2560. Up to eight radio input channels can be fed to the general purpose pins of 32U2 and are converted to PPM signal to be decoded by Atmega 2560. It also acts as the in-line programmer for the Atmega 2560. The 32U2 is connected to a USB header and can serve as a programmer via the UART0 pins of the 2560. It is the main source of communication between the base station computer and the APM 2.5. The firmware for the 32U2 is used as is from the Arducopter project. Further details are provided in [53]. Using this hardware, the main program loop including estimation and control can run at 100Hz.

The main inertial sensor used is the MPU6000 from Invensense. This integrated circuit contains 3-axis gyroscopes and accelerometers all on one chip. The advantage of this is less noise caused by multiple connections on the PCB. The MPU6000 contains other useful features such as a temperature sensor, built in motion processor and digital filtering. Communication with the microcontroller is available using either I^2C or SPI. All specifications are available in the datasheet [54].

To supplement the accelerometer and gyroscope, a 3-axis magnetometer is also used. The Honeywell HMC5883L magnetometer is a magneto-resistive sensor circuit with a set of three orthogonal axes to measure the surrounding magnetic field [55]. In the presence of a magnetic field, the bridge elements produce a change in the voltage across the bridge corresponding to a change in the bridge resistive elements. Therefore, the sensor produces a differential voltage output based on the measured magnetic field in the sensitive axis directions. This voltage is sampled on the MPU6000 by a 12-bit ADC. The reading is placed in a storage register and accessed by the main microcontroller through the I^2C protocol. The HMC5883L has a full scale reading configurable up to ±8.1 Gauss that is scaled through a 3-bit gain control ranging the output in steps from ±0.88 Gauss to ±8.1 Gauss. For this application the full scale reading is configured to ±1.3 Gauss. The output rate can be selected from 0.75 Hz to 75 Hz by writing to the appropriate register. The magnetometer is configured to sample at 50Hz, or one half of the main loop sample

rate.

Fig. 3.2 shows the circuit board of the APM 2.5 and Fig. 3.3 shows the enclosure for the APM 2.5 as mounted on the quadrotor with the input and output wires connected.

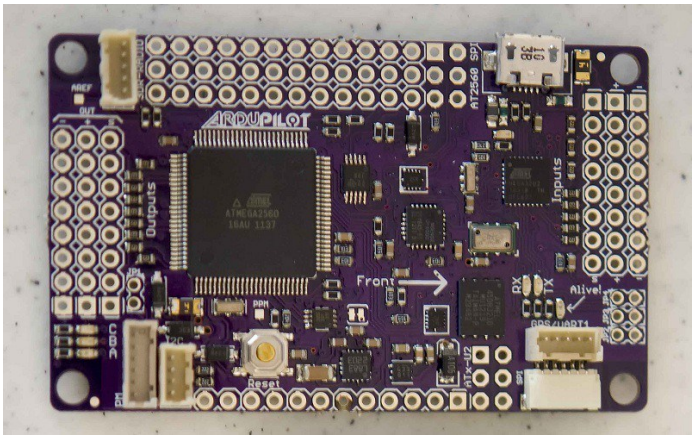

Figure 3.2: APM 2.5 PCB

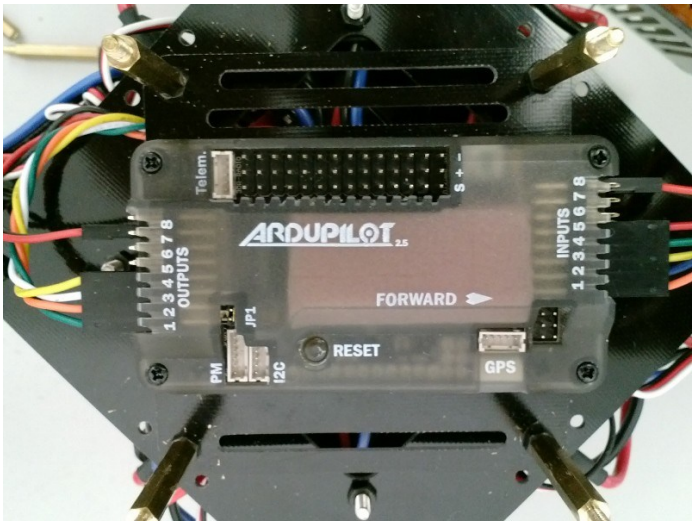

Figure 3.3: APM 2.5 enclosure

3.1.3 Motors and ESCs

The entire system is powered by a 2700 mAh 3-cell lithium polymer battery. The battery is capable of allowing up to 243 A of current in short bursts with a maximum continuous current rating of 121.5 A. The battery voltage when fully charged is 12.3V, and 9.9V

when discharged. Lithium polymer batteries can be dangerous if used incorrectly, it is important to follow manufacturers directions for storing, charging and discharging. Battery life of 10-12 minutes has been observed during testing with this configuration.

The brushless motor and electronic speed controller (ESC) pairs are the actuators for the quadrotor. The ESCs are contolled by the PWM signal calculated and sent from the APM and the ESC converts the input DC voltage from the battery to 3-phase AC current to set the motors at the desired speed. The ESCs rely on measurement of the back-EMF generated in the coil as feedback for the speed control. Many earlier and less expensive speed controllers rely on hall effect sensors which are often noisy and less accurate. The motor is an outrunner brushless motor produced by 3DR Robotics. Outrunner motors operate by spinning the outer shell around the windings. The specifications as provided by the manufacturer are given in Table 3.1.

Voltage	KV(rpm/V)	Max Pull	Weight	Max power	ESC
7.4-15 V	850	880g	52g	200watt	20A

Table 3.1: Motor specifications

3.1.4 Other Components

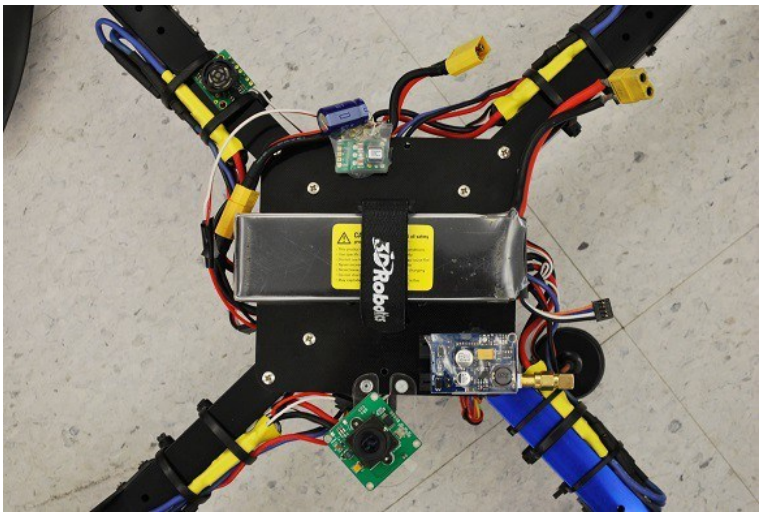

Figure 3.4: Quadrotor aircraft underside showing battery, sonar and camera

Some additional sensors are added to perform the outer-loop position and altitude control. The sensors used are a sonar and of course the video camera system. A wireless serial transceiver is also onboard the quadrotor and another is connected to the ground station PC. These components are mounted to the underside of the quadrotor and are visible in Fig. 3.4.

The sonar used is the Maxbotix EZ0 sonar. This low-cost sonar provides excellent accuracy and a wide beamwidth. The sonar is connected to the APM through the ADC. It has an input voltage range of 2.5-5.5 V and a measuring range of 6 to 254 inches. It is small in size, weighs only 4.3 grams and is mounted on the rear arm of the UAV as close to the airframe centre as possible. Care should be taken to keep the sonar away from other electronics as noise can affect the analog readings.

The camera used to capture images is a generic low-cost CMOS camera. The resolution of the camera is 728×488 and the input supply range is 6-20 V. The power is supplied to the camera and transmitter by the LIPO battery. The camera weighs approximately 15 grams. The camera is combined with a wireless transmitter and receiver operating at 5.8 GHz to transmit the video to the ground station computer so the images can be captured and processed. The transmit power is 200 mW and the transmitter weighs 60 grams. Cloverleaf style antennas are used on both the video transmitter and receiver. This type of antenna is useful for UAV video applications since the circular polarization pattern is more robust to variations in angles during flight manoeuvres than a standard linear polarized monopole antenna. At the ground station, the video receiver is connected to the computer via a USB video capture device.

The ground station computer is responsible for capturing images, processing the images and two-way serial transmission with the UAV. The computer used is an Intel Core i7 running at 3.4 GHz. The program is written in Visual Studio C++ 2010 Express and uses OpenCV 2.4.4 libraries.

Wireless serial connectivity is provided between the ground station computer and UAV using an Xbee Series 1 module on the UAV and the XStick USB module connected to

the computer. The Xbee system operates at 2.4GHz and is capable of a maximum rate of 250kbps.

3.2 Calibration Techniques

3.2.1 RC Transmitter

Figure 3.5: Controller for quadrotor

The RC transmitter, shown in Fig. 3.5, requires some simple calibration to ensure proper operation. Changing the control sticks provides a PPM signal to the receiver having a value between 1100-1900 (for this transmitter). The throttle stick does not return to the middle position when the operator removes their finger while the roll and pitch control sticks do return to centre.

The maximum and minimum throttle values vary slightly from controller to controller as they are derived from a mechanical potentiometer. The minimum and maximum values must be known by the program in order to properly provide throttle inputs in the available range. These values can be determined by reading the values from the receiver through the microcontroller and displaying them to the serial port. The values can then be defined in the source code.

Since the pitch and roll control sticks return to centre, this position will be considered

the "0" position for the desired angle. For this reason the value, and the mid position must be known in order to properly calculate the desired roll and pitch and have the correct sign. For example: consider that the mid point of the roll control stick has a value of 1500. If a value of 1600 is read, the desired roll is now 1600 – 1500 giving a value of positive 100. A scale factor can be applied to correlate this to degrees having the desired sensitivity. If a value of 1400 is read the desired roll is now 1400 – 1500 resulting in a value of -100.

To calculate a scale factor, consider the maximum value of 1900 – 1500 resulting in 400. Obviously using this directly as the desired angle will result in a very hard to fly quadrotor with a 400° full scale desired angle. One can choose a realistic value for maximum angle obtainable through personal preference and test flying. For example a realistic maximum value of 20 degrees works well in a indoor environment and any values higher than that results in extremely fast movement. One simply has to map the previous value of 400 to the desired value of 20, thus the choice of scaling factor would be 1⁄20.

3.2.2 Magnetometer

As previously mentioned, obtaining accurate magnetometer readings in a indoor environment can be quite challenging. Measurements of the magnetic field can be corrupted by any metal in the room or walls. Additionally, any electronics operating nearby generate magnetic fields. The quadrotor itself can produce magnetic fields from both the electronics and the motors. Since the measured earth magnetic field is very weak (in the range of nano Gauss), any corruptions can have a large impact.

For these reasons, it is essential to implement some sort of calibration for the magnetometer. One simple method used to calibrate the magnetometer is to collect a set of readings while rotating the quadrotor around all three axes. Once data is collected, the offset is calculated and removed from all future readings. This method is called Hard Iron Calibration. It is simple to implement but does not account for any warping of the field caused by field distortions. Since the quadrotor in this thesis is to be flown primarily indoors, it is worthwhile to implement more advanced calibration technique. A method

is successfully implemented and good results have been achieved in [56]. The method is described in detail below.

First, the model of the magnetometer is defined as

$$\tilde{m} = \begin{bmatrix} E_1 & 0 & 0 \\ E_2 \sin\rho_1 & E_2 \cos\rho_1 & 0 \\ E_3 \sin\rho_2 \cos\rho_3 & E_3 \sin\rho_3 & E_3 \cos\rho_2 \cos\rho_3 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} + \begin{bmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \end{bmatrix} = K_m m + b_m \quad (3.1)$$

where $\tilde{m} = (\tilde{m}_1, \tilde{m}_2, \tilde{m}_3)^T$ and $m = (m_1, m_2, m_3)^T$ represent the three axis measurements and actual values respectively. E_k, ρ_k, and ξ_k represent the sensor scaling errors, sensor misalignment, and sensor offsets of the individual sensor axes, $1 \leq k \leq 3$, respectively. These 9 parameters need to be solved in order to perform the calibration. It is interesting to note that if $E_k = 1$ and $\rho_k = 0$ such that $K_m = I$ then Eq. (3.1) reduces to the Hard-Iron only calibration model.

Eq. (3.1) can be inverted as

$$m = K_m^{-1}(\tilde{m} - b_m) \quad (3.2)$$

Substituting Eq. (3.2) into the equation for magnitude of the magnetic field

$$m_1^2 + m_2^2 + m_3^2 = |m|^2 \quad (3.3)$$

the following equation is obtained

$$C_1\tilde{m}_1^2 + C_2\tilde{m}_1\tilde{m}_2 + C_3\tilde{m}_1\tilde{m}_3 + C_4\tilde{m}_2^2 + C_5\tilde{m}_2\tilde{m}_3 + C_6\tilde{m}_3^2 + C_7\tilde{m}_1 + C_8\tilde{m}_2 + C_9\tilde{m}_3 = C_{10} \quad (3.4)$$

where the coefficients C_I ($1 \leq I \leq 10$) are functions of the 10 parameters E_k, ρ_k, ξ_k and $|m|$. The value $|m|$ represents the magnitude of the earth magnetic field at the current location. This value is known and constant and can be obtained from the International Geomagnetic Reference Field (IGRF) [57] or World Magnetic Model (WMM) [58]. For example, the value obtained for the inertial reference using the longitude, latitude and elevation of Thunder Bay, Ontario is $m_I = (0.15550, -0.01036, 0.54601)^T$ Gauss.

Next a magnetometer data set $\{\tilde{m}\}$ must be collected consisting of N data points. The method of collecting the data can vary from system to system. The method used here

is to log the three components of the magnetometer reading in a CSV file while rotating the quadrotor along all axes. At least 1000 samples were collected to ensure accurate results.

The system can be rewritten as

$$\begin{bmatrix} \tilde{m}_{1,1}^2 & \tilde{m}_{1,1}\tilde{m}_{2,1} & \cdots & \tilde{m}_{3,1} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{m}_{1,N}^2 & \tilde{m}_{1,N}\tilde{m}_{2,N} & \cdots & \tilde{m}_{3,N} \end{bmatrix} \begin{bmatrix} C_1/C_{10} \\ \vdots \\ C_9/C_{10} \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \tag{3.5}$$

Now the numerical values of $C_1/C_{10} \ldots C_9/C_{10}$ can be found using a least-squares method. This results in a system of nine non-linear equations having nine unknowns, E_k, ρ_k, ξ_k, which is solved numerically. In this case MATLAB is used to solve for the parameters using the measured data set as the input. Since ideal values are, $E_k = 1$, $\rho_k = 0$, and $\xi_k = 0$, these values are a good choice for the initial guess.

Once the nine parameters are found they can be used to correct future magnetometer readings as given by Eq. (3.2). The parameters do not need to be recalculated for every flight, only if there is a change of location or some other change made to the area resulting in a modified electromagnetic field. It is also important to note that the difference in parameters using data sets with the motors off compared to on is significant. One can conclude that running the engines has a great impact on the measured field. The values calculated are provided below in Table 3.2 and Table 3.3 for both cases. Fig. 3.6 and Fig. 3.7 show both the uncalibrated data and the calibrated data for the case with the motors off and on respectively. Properly calibrated three axis magnetometer measurements should trace out a sphere since the magnitude of the magnetic reference vector is constant, and should centered at (0, 0, 0). Distorted magnetometer measured can appear elliptical instead of spherical. Each individual plot shows one cross section of the sphere. It is clear that the calibrated data is now centered, hence the offset is removed. In this case elliptical distortion is not obvious in the uncalibrated measurement, however the fact that the calibration parameters are not 0 implies some distortion is present in the uncalibrated data.

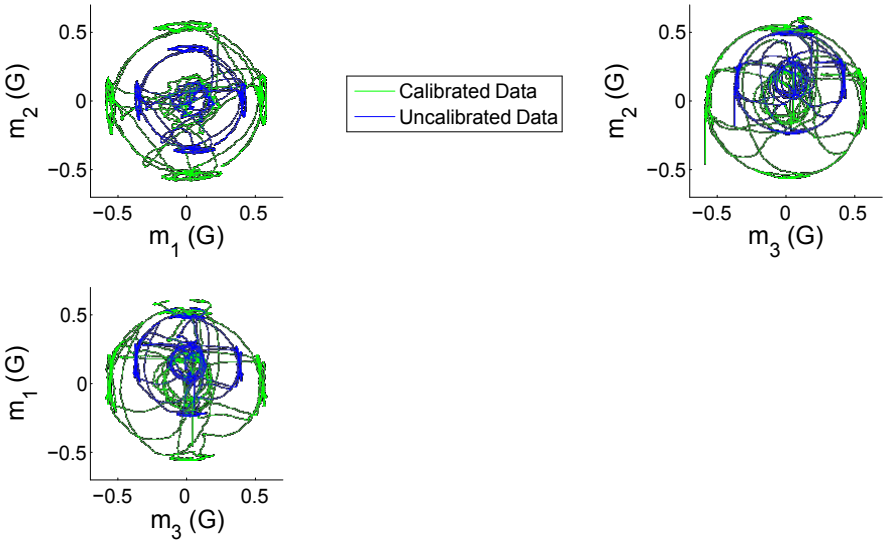

Figure 3.6: Magnetometer calibration results - motors off

	E	ρ	ξ
1	0.6891	-0.0357	0.0307
2	0.6811	0.0377	0.0136
3	0.6705	-0.0239	0.1366

Table 3.2: Magnetometer calibration parameters - motors off

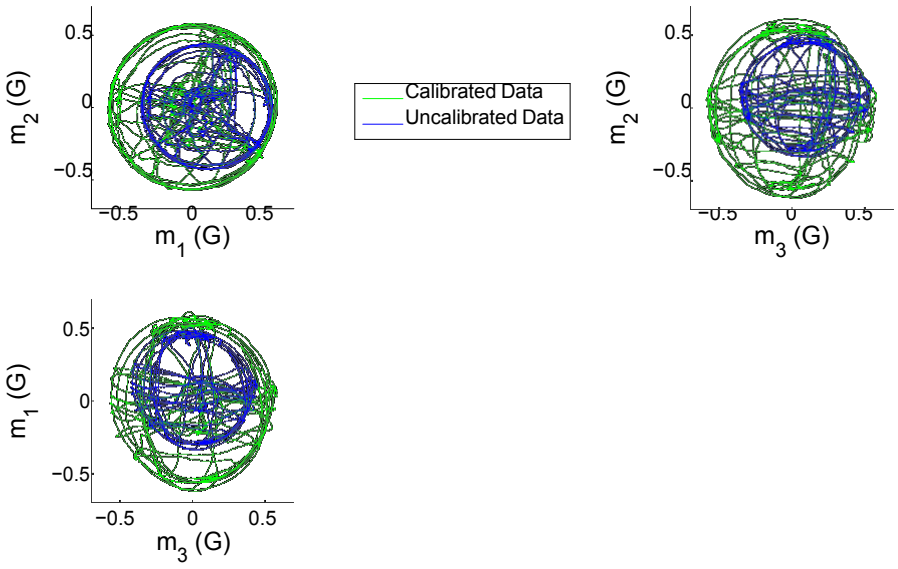

Figure 3.7: Magnetometer calibration results - motors on

	E	ρ	ξ
1	0.7755	-0.0421	0.0952
2	0.7583	0.0078	0.0045
3	0.6833	-0.0464	0.0854

Table 3.3: Magnetometer calibration parameters - motors on

3.2.3 Camera

Usually, a low-cost camera is subject to a significant distortion in the image. These distortions however are constant and can be corrected with proper calibration. Additionally the, camera calibration process can help to determine the relation between the camera's units of pixels and real world units (ie. mm). Also, the focal length of the lens is needed in the calculation of the inertial average optical flow. The camera calibration used here is based on the documentation provided for OpenCV using the included calibration libraries. The process is outlined below.

Consider an individual image point (*x, y*) on the uncalibrated image. To correct for radial distortion the following transformation is applied

$$x_{corr} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{corr} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{3.6}$$

To correct for tangential distortion the following transformation is applied

$$x_{corr} = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_{corr} = y + [p_1(r^2 + 2y^2) + 2p_2 xy] \tag{3.7}$$

where k_1, k_2 and k_3 are the radial distortion parameters, p_1 and p_2 are the tangential distortion parameters, $r^2 = x^2 + y^2$, and (x_{corr}, y_{corr}) is the new position of (x, y) on the calibrated image. In OpenCV the parameters $[k_1, k_2, p_1, p_2, k_3]$ are called the distortion parameters and presented as a row vector. In addition to the distortion parameters the calibration procedure also provides the camera matrix.

$$Camera\_Matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

where f_x, f_y are the camera focal lengths and c_x, c_y are the optical centres expressed in pixel coordinates.

These parameters are obtained through a calibration process that involves capturing multiple images of a known pattern from various perspectives and using OpenCV functions to detect the pattern and calculate the parameters. Once the parameters are obtained they can be applied to images before further processing. Example code for the calibration procedure can be found on docs.opencv.org. For reference, the calibration values

obtained are

$$Distortion\_Parameters = [-0.39385, -0.36887, 0, 0, 1.16185]$$

$$Camera\_Matrix = \begin{bmatrix} 724.616 & 0 & 319.5 \\ 0 & 724.616 & 239.5 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.9}$$

Chapter 4

Estimation and Control

This section will discuss the theory and implementation of the attitude estimation and various controllers used in the thesis.

4.1 Attitude Estimation

In order to control the attitude of a quadrotor UAV, one must first obtain a measure of the attitude. There is no sensor that measures attitude. Obtaining a good attitude estimate can be a challenging problem especially when low-cost sensors are used. Measurements from these sensors are subject to low-resolution, noise and time varying bias terms [25]. Additionally, since the algorithms are to run on embedded hardware, the low memory and limited processing capability must be considered.

In theory, the angular velocity measured from the gyroscope can be integrated over time to obtain the orientation angles. However, in practice the measurement noise and biases of the sensor corrupts the estimate after a few seconds. The roll and pitch angles can be determined from the accelerometer measurements using the following relationship.

$$a_B \approx -gR^T\hat{z}_I$$

$$\frac{a_B}{-g} \approx \begin{bmatrix} -\sin\theta \\ \sin\varphi\cos\theta \\ \cos\varphi\cos\theta \end{bmatrix}$$

where $a_B = (a_{B,x}, a_{B,y}, a_{B,z})^T$ are the body-frame accelerometer measurements and $g = 9.81$ m/s^2

Calculating roll and pitch from the accelerometer results in very noisy values and in addition it is not possible to obtain a value for yaw. The solution to these problems is to fuse sensor readings together to achieve the best result in terms of accuracy and noise. In this thesis the non-linear complementary filter was chosen for implementation. This estimator offers excellent performance in literature and is efficient to implement on embedded hardware in quaternion form.

To discuss the theory of the complementary filter, the standard non-linear complementary filter is given as follows [59]

$$\begin{aligned} \dot{\hat{R}} &= \hat{R}S(\Omega_y - \hat{b} + \sigma_R) \\ \dot{\hat{b}} &= \sigma_b, \ \hat{b}(0) \in \mathrm{R}^3 \\ \sigma_R &= \sum_{i=1}^{n} k_i v_i^B \times \hat{R}^T v_i' \\ \sigma_b &= -k_I \sigma_R \end{aligned} \tag{4.1}$$

where $\Omega_y = \Omega + b$ is the measured value from gyroscope, which consists of the actual angular velocity Ω plus an unknown constant or slowly time varying bias b. $\hat{R}$ and $\hat{b}$ represent the estimates of R and b respectively. v_i^B represents a set of measured data in the body frame and v_i' represents a set of known vectors in the inertial frame where $n \geq 2$. The parameters k_I and k_i represent positive gains.

To implement the above complementary filter on an IMU, three sensors are used. These

are three-axis gyroscope, three-axis accelerometer and three-axis magnetometer. The vector Ω_y represents the measured value from the gyro. If n is taken as 2, v_1 and v_2 represent the accelerometer and magnetometer vectors respectively. $v_1{}^B$ represents the body frame measurement value from the accelerometer sensor and $v_1{}' = (0, 0, g)^T$ is the inertial frame gravitational acceleration. $v_2{}^B$ represents the body frame measurement value from the magnetometer sensor and $v_2{}'$ is the geomagnetic field value in the inertial frame which can be obtained for a given location from the IGRF-11 model.

In practice, there are some issues with the above standard implementation of the complementary filter [59]. Estimation of the roll and pitch angles is influenced by magnetic disturbances and bias. This is especially problematic for the quadrotor since magnetic disturbances can be caused by the electronics and the motors. Additionally the indoor environment can have magnetic fields from electronics or metal in the room. These disturbances are unavoidable and can lead to large errors in yaw estimation and significant errors in the estimation of roll and pitch angles. Also there is a coupling between the dynamics of the estimates of roll, pitch and yaw. In addition to the coupling problems, the integral correction term can suffer from the wind up effect. The term can grow very large leading to slow convergence of the estimation. This is a common problem associated with the integral action. The wind up effect can be caused by large initial errors, poor gain tuning, error in the knowledge of inertial reference vectors, measurement noise and sensor vibration. Lowering the integral gain k_I can help to reduce the problem but can lead to reduced performance.

In [59] a new form of the complementary filter called the "conditioned observer" is introduced to deal with these issues of coupling and wind up on the bias estimation.

$$
\begin{aligned}
\dot{\hat{R}} &= \hat{R}S(\Omega_y - \hat{b} + \sigma_R) \\
\dot{\hat{b}} &= -k_b\hat{b} + k_b sat_{\triangle}(\hat{b}) + \sigma_b, \quad \hat{b}(0) \in \mathrm{R}^3 \\
\tau_R &= k_1 u_B \times \hat{u}_B + k_2 \hat{u}_B \hat{u}_B^T (v_B \times \hat{v}_B) \\
\sigma_b &= -k_3 u_B \times \hat{u}_B - k_4 v_B \times \hat{v}_B
\end{aligned}
\tag{4.2}
$$

$$\begin{aligned} u_B &= -\tfrac{a_B}{g},\ v_B = \tfrac{\pi_{u_B} m_B}{|\pi_{u_I} m_I|},\ u_I = e_3 \\ v_I &= \tfrac{\pi_{u_I} m_I}{|\pi_{u_I} m_I|},\ \hat{u}_B = \hat{R}^T u_I,\ \hat{v}_B = \hat{R}^T v_I \end{aligned} \tag{4.3}$$

where $\pi_x = |x|^2 I_3 - xx^T, \forall x \in \mathrm{R}^3$ is the orthogonal projection on the plane orthogonal to x. k_1, k_2, k_3, k_4, k_b and Δ are positive numbers and $sat_\Delta(x) = x \min(1, \frac{\Delta}{x})$. The author in [59] provides proofs for the following properties:

1. The dynamics of the estimate errors $(\tilde{R}, \tilde{b})$, with $\tilde{R} = R\hat{R}^T$ and $\tilde{b} = b - \hat{b}$, have only four isolated equilibria $(\tilde{R}, \tilde{b}) = (\tilde{R}_i^*, 0)$, $i = 0, ..., 3$, with $\tilde{R}_0^* = I_3$.
2. The equilibrium $(\tilde{R}, \tilde{b}) = (I_3, 0)$ is locally exponentially stable.
3. The equilibria $(\tilde{R}_1^*, 0), (\tilde{R}_2^*, 0), (\tilde{R}_3^*, 0)$ are unstable. For almost all initial conditions $(\hat{R}(0), \hat{b}(0)) \not= (\tilde{R}_i^{*T} R(0), b)$, $i = 1, 2, 3$, the trajectory $(\hat{R}(t), \hat{b}(t))$ converges to the trajectory $(R(t), b(t))$.
4. The dynamics of $\hat{u}_B$ does not depend on m_B when considering $\hat{b}$ as an input.
5. The estimated gyro-bias $\hat{b}$ is bounded in norm by $\bar{\Delta} = \Delta + (k_3 + k_4)/k_b$

To exploit the increased computational efficiency achieved when using the quaternion representation the conditioned observer can be easily expressed in quaternion form.

$$\begin{aligned} \dot{\hat{Q}} &= \tfrac{1}{2} A(\hat{\Omega}) \hat{Q},\ \ Q(0) \in Q \\ \dot{\hat{b}} &= -k_b \hat{b} + k_b sat_\Delta(\hat{b}) + \sigma_b \\ \hat{\Omega} &= \Omega - \hat{b} + \sigma_R \\ A(\hat{\Omega}) &= \begin{bmatrix} 0 & -\hat{\Omega}^T \\ \hat{\Omega} & -S(\hat{\Omega}) \end{bmatrix} \end{aligned} \tag{4.4}$$

In order to implement the conditioned observer on the embedded microcontroller, the equations need to be expressed in the discrete form using the method from [59]. By exact integration of the first equation in Eq. (4.4) the following is obtained

$$\hat{Q}_{k+1} = exp\left(\frac{T}{2}A(\hat{\Omega}_k)\right)\hat{Q}_k \tag{4.5}$$

where T is the sample time of the digital system.

Using Taylor expansion

$$exp(\frac{T}{2}A(\hat{\Omega}_k)) = [\cos\left(\frac{T|\hat{\Omega}_k|}{2}\right)\boldsymbol{I}_4 + \frac{T}{2}sinc\left(\frac{T|\hat{\Omega}_k|}{2}\right)A(\hat{\Omega}_k)] \tag{4.6}$$

where $sinc(x) = sin(x)/x$. The discrete version is then written as

$$\begin{aligned} \hat{Q}_{k+1} &= [\cos\left(\tfrac{T|\hat{\Omega}_k|}{2}\right)\boldsymbol{I}_4 + \tfrac{T}{2}sinc\left(\tfrac{T|\hat{\Omega}_k|}{2}\right)A(\hat{\Omega}_k)]\hat{Q}_k \\ \hat{b}_{k+1} &= T(-k_b\hat{b}_k + k_b sat_{\triangle}(\hat{b}_k) + \sigma_{b,k}) + \hat{b}_k \end{aligned} \tag{4.7}$$

A flowchart of the code implementation of the conditioned observer is shown below.

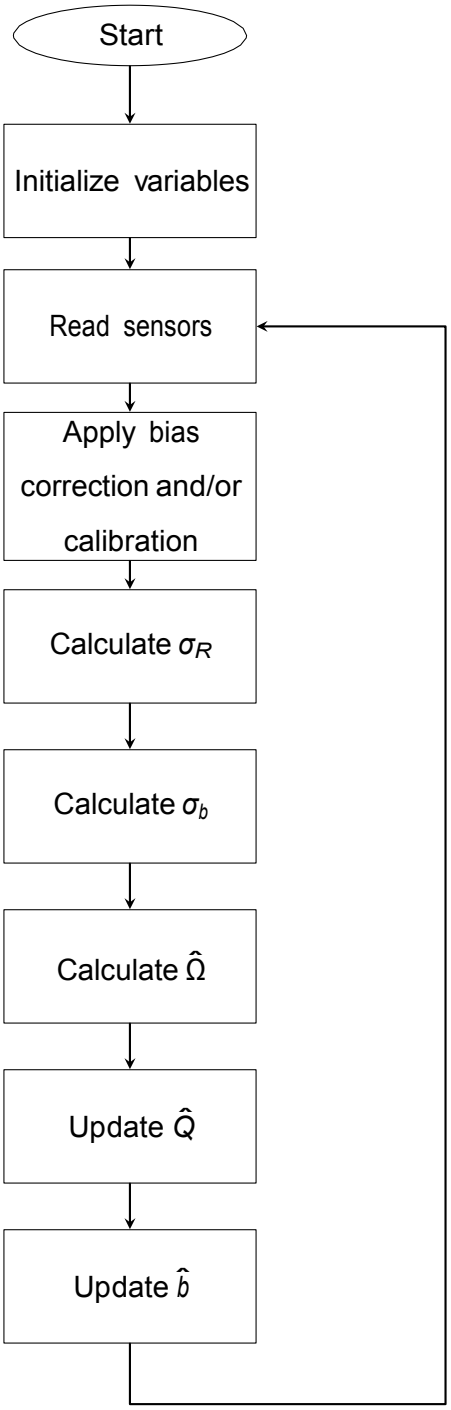

Figure 4.1: Program flow chart for estimating attitude

4.2 Attitude Control

Once the attitude estimation is complete the controller needs to be implemented to ensure the convergence of the estimated attitude to the desired attitude. Many controllers have been proposed in literature using many different approaches. Since the quaternion is available from the estimation it is desirable to also use a controller expressed in quaternion form. This allows for the continued benefits of the four parameter representation with respect to processing time.

The controller chosen for implementation is obtained from [33] and [34]. The controller is given as

$$\tau = -\Gamma\Omega - \alpha q \tag{4.8}$$

where Γ is a 3×3 symmetric positive definite matrix, Ω is the angular velocity, α is a positive scalar gain and q is the vector part of a quaternion.

This controller is designed to stabilize q to zero (or equivalently $R = I$), however it is required that the UAV be stabilized to any desired attitude. Therefore a modification must be made. Define $\tilde{Q} = Q^{d-1} \otimes \hat{Q}$ where $\tilde{Q}$ represents the error quaternion, Q^d represents the desired quaternion and $\hat{Q}$ is the estimated quaternion. The controller is now defined as

$$\tau = -\Gamma\Omega - \alpha\tilde{q} \tag{4.9}$$

where $\tilde{q}$ is the vector part of $\tilde{Q}$.

The proof in [33] and [34] is still applicable, and guarantees asymptotic stability for the equilibrium point ($\tilde{q} = 0, \tilde{q}_0 = \pm 1$).

This controller is a simple design that requires no knowledge of the model and is easy to compute. For these reasons it is an excellent candidate for practical implementation. In discrete form the controller can be expressed as

$$\tau_k = -\Gamma(\Omega_k - \hat{b}_k) - \alpha\tilde{q}_k \tag{4.10}$$

Best flight results were obtained using

$$\Gamma = \begin{bmatrix} 0.8 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 0.1 \end{bmatrix} \text{ and } \alpha = 4.2 \tag{4.11}$$

4.3 Rotor Torques Design

In order to properly control the UAV, it is necessary to convert the rotor torques derived from the attitude control to a rotor angular velocity value that can be sent to the motors. The method used here is obtained from [33] and [34]. From the quadrotor model we have,

$$\begin{aligned} \tau_1 &= db(\omega_2^2 - \omega_4^2) \\ \tau_2 &= db(\omega_1^2 - \omega_3^2) \\ \tau_3 &= \kappa(\omega_1^2 + \omega_3^2 - \omega_2^2 - \omega_4^2) \\ T &= b\sum_{i=1}^{4} \omega_i \end{aligned} \tag{4.12}$$

where d is the distance from the centre of mass to the rotor, b is the motor torque constant and κ is the rotor reactive torque constant. Determination of these parameters is discussed in Section 4.4. Define $\mu = (\tau_1, \tau_2, \tau_3, T)^T$ and $\bar{\omega}_d = (\omega_{d,1}^2, \omega_{d,2}^2, \omega_{d,3}^2, \omega_{d,4}^2)$. Then $\bar{\omega}_d = M^{-1}\mu$ where

$$M = \begin{bmatrix} 0 & db & 0 & -db \\ db & 0 & -db & 0 \\ \kappa & -\kappa & \kappa & -\kappa \\ b & b & b & b \end{bmatrix} \tag{4.13}$$

The relation can be expressed explicitly for each motor in terms of the torques and the total thrust.

$$\begin{aligned} \omega_{d,1}^2 &= \tfrac{1}{9}[\tfrac{2}{b}T + \tfrac{4}{db}\tau_2 + \tfrac{2}{\kappa}\tau_3] \\ \omega_{d,2}^2 &= \tfrac{1}{9}[\tfrac{2}{b}T + \tfrac{4}{db}\tau_1 - \tfrac{2}{\kappa}\tau_3] \\ \omega_{d,3}^2 &= \tfrac{1}{9}[\tfrac{2}{b}T - \tfrac{4}{db}\tau_2 + \tfrac{2}{\kappa}\tau_3] \\ \omega_{d,4}^2 &= \tfrac{1}{9}[\tfrac{2}{b}T - \tfrac{4}{db}\tau_1 - \tfrac{2}{\kappa}\tau_3] \end{aligned} \tag{4.14}$$

Since the ESCs are responsible for the speed control of the motors, a controller does not need to be developed to ensure the actual angular velocity of the motors converges to the

desired angular velocity. In this case we only need to determine the PPM signal to send the ESC based on the required angular velocity. To determine this signal, it is necessary to know the maximum speed of the motors and the range of PPM values given by the transmitter. The ESC, when properly calibrated, sets the throttle curve to begin and end at these values. Therefore at the maximum PPM, the maximum angular velocity is achieved. From this, the conversion from PPM value to angular velocity is obtained,

$$\frac{PPM_{sent} - PPM_{min}}{PPM_{max} - PPM_{min}} AngularVelocity_{max} = AngularVelocity_{sent} \tag{4.15}$$

4.4 Determination of Model Parameters

As discussed in Section 4.3, knowledge of certain parameters is required for the procedure to calculate the desired output to each motor. These values are also useful in simulation to obtain results consistent with the actual model used and allow for accurate gain tuning. In some cases some of the model parameters appear in the controller. The controller used here (Eq. (4.8)) does not require knowledge of model parameters which makes it an ideal choice for real-time implementation as some parameters, such are inertia, can often be difficult to determine. The controller should be robust enough to still ensure stability for small errors in the model parameters. In practice a reasonable approximation of the parameters, combined with careful gain tuning, is sufficient to obtain good results. In this case mass, distance and the two proportionality constants, b and κ need to be determined. This section will outline the experimental methods used to determine the parameters of the experimental quadrotor.

4.4.1 Mass

The mass of the UAV is used in the altitude controller and is an important parameter. The mass is one of the easiest parameters to determine. In this case the quadrotor was outfitted with all flight electronics to be used, propellers and a battery. The mass was measured using a digital hanging scale. The measured mass is 1.23kg. The mass does not change throughout the flight, unlike some gas powered aircraft and spacecraft where the mass can change considerably as fuel is spent.

4.4.2 Distance

The distance is a measure of the length from the centre of mass to the centre of one rotor. This distance value is used in the determination of the motor outputs. The distance was measured using a ruler and the value obtained is 0.27m. Assuming a rigid structure, this value does not change.

4.4.3 Motor thrust constant b

The value for b depends on size, shape and pitch of the propellers as well as the air density. The parameter is seen in the relationship

$$T = b \sum_{i=1}^{4} \omega_i \tag{4.16}$$

rearranging for b and assuming all four rotors have the same angular velocity

$$b = \frac{T}{4\omega} \tag{4.17}$$

To determine this value experimentally the following method was used. Consider that at a hover condition the total thrust is equal to the gravitational force such that $T = mg$. Additionally, for a stabilized hover condition, the angular velocity of each rotor is essentially equal. Therefore to calculate b, the UAV was flown to a hover condition and the PPM signal was recorded for each motor. From this, the angular velocity was obtained. Knowing both the thrust and angular velocity, b is calculated using Eq. (4.17). The experiment was performed 3 times, and the average was taken as the value for b. In this case the value obtained is $b = 9.408 \times 10^{-6} N \cdot s/rad$.

4.4.4 Motor reactive torque constant κ

The constant κ appears in the equation for reactive torque. It depends on air density, rotor radius, shape and pitch angle. The parameter is seen in the relationship

$$Q_i = k\omega_i^2 \tag{4.18}$$

To determine this value experimentally the following method was used. All four rotors were spun in the same direction at the same speed with the same pitch propellers attached. Reactive torque around the yaw axis was measured and the angular velocity

of the rotors was recorded. All four rotors were used in order to get a larger value to measure for more accuracy. The result is then divided by four. Once Q is measured and ω is known, κ is calculated as

$$\kappa = \frac{Q}{4\omega} \tag{4.19}$$

The experiment was performed 3 times, and the average was taken as the value for κ. In this case the value obtained is $\kappa = 3.76 \times 10^{-7} Nm \cdot s^2/rad^2$.

4.5 Altitude Control

While the main focus of the thesis is altitude control based on optical flow, it is also possible to obtain altitude hold via feedback from a sonar. Obtaining the altitude from a sonar is computationally much easier to achieve than from optical flow. One simply has to read the analogue height value from the sonar and use that for feedback. The main disadvantages of the sonar are that the measurements are noisy and it offers no possibility for translational control. Sonar based altitude hold was achieved on the quadrotor model in real-time with positive results. Consider the vertical dynamics of the UAV,

$$\begin{aligned} \dot{z} &= v_z \\ m\dot{v}_z &= mg - TR\hat{z}_I \end{aligned} \tag{4.20}$$

In near-hover conditions, it can be approximated that $R = I$ resulting in

$$\begin{aligned} \dot{z} &= v_z \\ m\dot{v}_z &= mg - T \end{aligned} \tag{4.21}$$

substituting and solving for $\ddot{z}$,

$$\ddot{z} = \frac{T}{m} + g \tag{4.22}$$

Let $u = g - \frac{T}{m}$ such that $\ddot{z} = u$ Then the desired thrust is given as

$$T_d = (g - u)m \tag{4.23}$$

The control input u is designed as follows. Let the altitude error $\tilde{z} = z^d - z$, differentiating twice results in

$$\begin{aligned} \dot{\tilde{z}} &= z^d - \dot{z} = -\dot{z} \\ \ddot{\tilde{z}} &= -\ddot{z} \\ \dddot{\tilde{z}} &= -u \end{aligned} \tag{4.24}$$

The control input u is selected as

$$u = k_1 \tilde{z} + k_2 \dot{\tilde{z}} \tag{4.25}$$

The dynamical equation of $\tilde{z}$ is then equal to

$$\ddot{\tilde{z}} + k_1 \dot{\tilde{z}} + k_2 \tilde{z} = 0 \tag{4.26}$$

It is clear that stability and convergence of $\tilde{z} \to 0$ is guaranteed if $k_1, k_2 > 0$.

4.6 Optical Flow Control

To obtain position hold and landing control two separate optical flow based controllers are used. First recall the translational part of the quadrotor model, Eq. (2.21). Multiplying by mass, the equation is expressed as

$$m\dot{v} = mg\hat{z}_I - TR\hat{z}_I \tag{4.27}$$

In order to control the translational dynamics the full vector term $TR\hat{z}_I$ is taken as the control input. The desired value is assigned as $u = (u_x, u_y, u_z)^T = (TR\hat{z}_I)^d$. The controller defined in Eq. (4.9) ensures the orientation R converges to the desired orientation R^d while the optical flow based controllers introduced below determine u. Eq. (4.27) can now be expressed as

$$m\dot{v} = mg\hat{z}_I - u \tag{4.28}$$

For hovering flight, the controller proposed in [48] is

$$u = k_P w + k_I \int_0^t w d\tau + mgz_I, \quad k_P, k_I > 0 \tag{4.29}$$

This controller is a PI-type controller and only depends on the measured value of optical flow, w.

During the landing phase the controller Eq. (4.29) is used to control the x and y axes. A new controller is applied to the z axis only to achieve landing control.

$$u_z = mk(w_z - w^*) + mg \tag{4.30}$$

where $w^* > 0$ is the desired value of optical flow. The rate of descent can be controlled by adjusting w^*.

In order to implement the controller it is necessary to extract the desired quaternion so that it can be used in the attitude controller to achieve the position control. The position control scheme is illustrated in Fig. 4.2. The method used is obtained from [60]. Recall that $(TR\hat{z}_I)^d = u$, where u is the control input obtained from the optical flow based controllers. Using the following method it is always possible to extract both the desired thrust and desired quaternion as

$$\begin{aligned} T^d &= m||g\hat{z}_I - u|| \\ q_{0,d} &= \sqrt{\frac{1}{2} + \frac{m(g-u_z)}{2T^d}} \\ q_d &= \frac{m}{2T^d q_{0,d}} \begin{bmatrix} u_y \\ -u_x \\ 0 \end{bmatrix} \end{aligned} \tag{4.31}$$

for the condition that $u \neq (0, 0, \chi)$, for $\chi \geq g$.

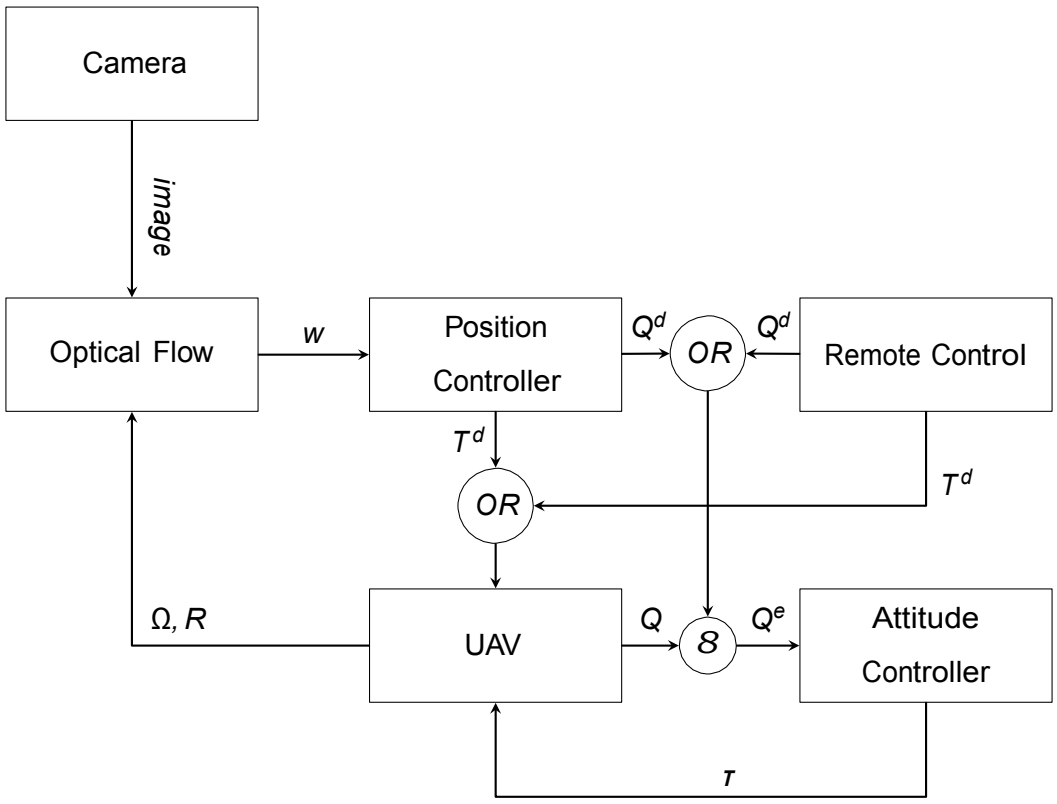

Figure 4.2: Position control scheme

Chapter 5

Results

5.1 Simulation Results

5.1.1 Simulation 1 - Attitude Controller

In this simulation, the attitude controller, given by Eq. (4.8), is examined. The initial conditions of the attitude are set as 60°, -40° and 80° for roll, pitch and yaw respectively. This corresponds to the quaternion $Q(0) = (0.5135, 0.5503, 0.0751, 0.6541)^T$. The desired orientation is the level position $Q^d = (1, 0, 0, 0)^T$. The mass in the simulation is taken as 1.23kg. The gain are obtained by trial and error and the values used in the simulation are $\Gamma = 0.4\boldsymbol{I}_{3\times 3}$ and $\alpha = 2.3$. Fig. 5.1 shows the attitude error quaternion versus time. Fig. 5.2 shows the aircraft attitude versus time. Fig. 5.3 shows the angular velocity of the aircraft versus time. Fig. 5.4 shows the control input versus time.

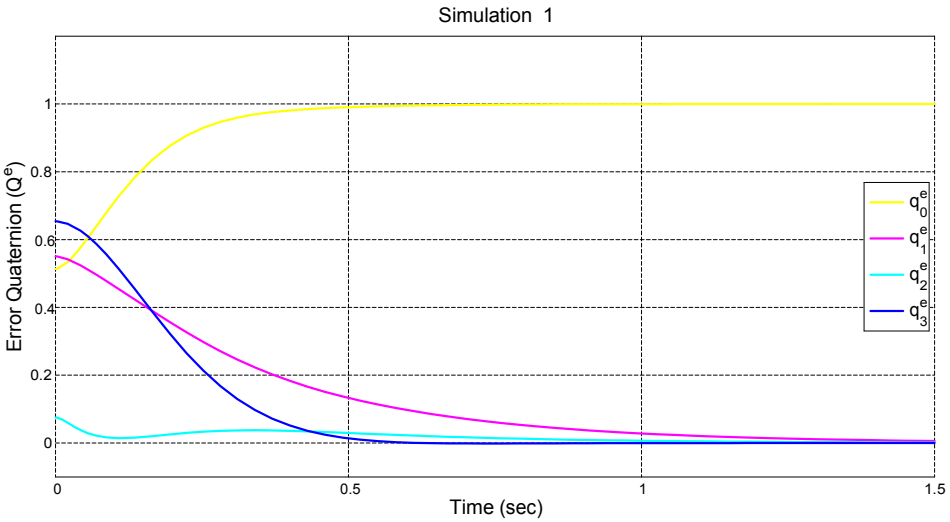

Figure 5.1: Aircraft error quaternion

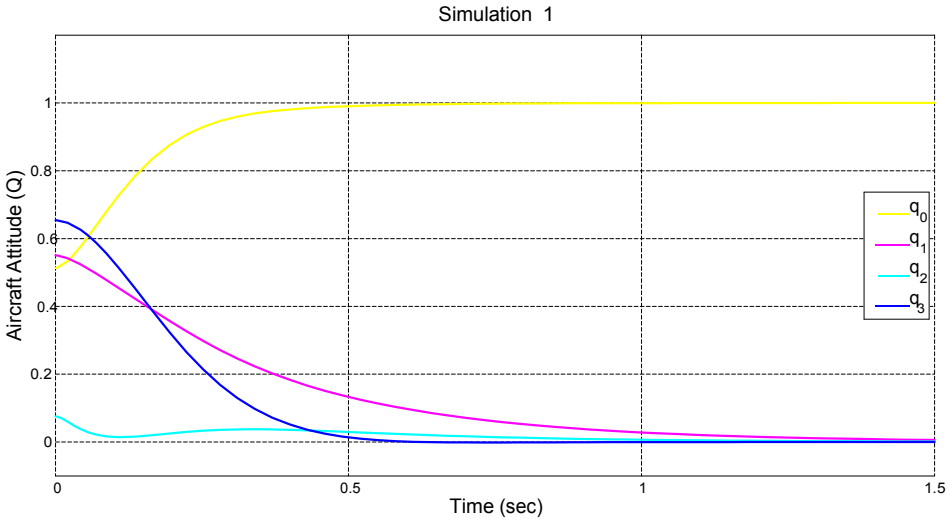

Figure 5.2: Aircraft quaternion

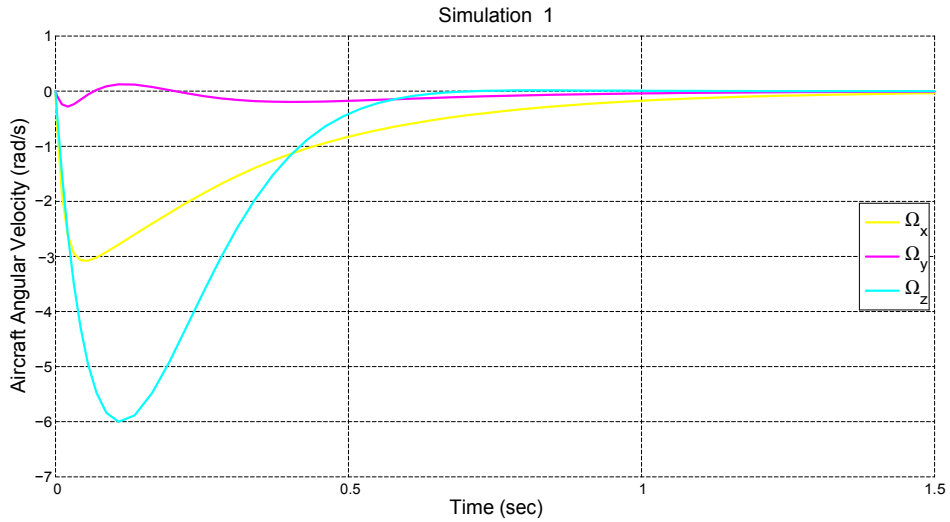

Figure 5.3: Aircraft angular velocity

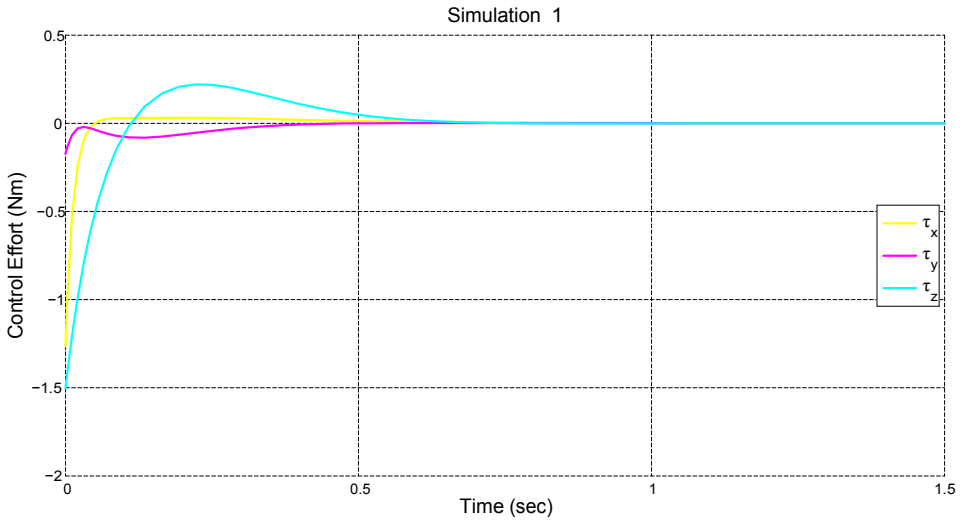

Figure 5.4: Control effort

5.1.2 Simulation 2 - Landing Controller

In this simulation, the landing controller is tested using various platform motions. The controller simulated is Eq. (4.30) applied to the vertical component of the translational dynamics given by Eq. (4.28). The optical flow value is simulated by calculating $w = \dot{h}/h$ where the necessary velocity and position values are obtained from the known target motion selected and from the aircraft model, resulting in an ideal optical flow value. In the simulations $h = -z + z_G$. The mass of the quadrotor used in the simulations is 1.23kg and the gain, k, is equal to 15. Fig. 5.5 shows the vertical landing manoeuvre of the aircraft on a non-oscillating platform (i.e. $z_G = 0$). Fig. 5.6 shows the landing manoeuvre on an oscillating platform where the platform motion is described by $z_G = 0.1\sin(0.6\pi t)$. Fig. 5.7 shows the landing manoeuvre on a stochastically moving platform where the motion is described by the sum of several sine functions with randomly chosen values for frequency and amplitude.

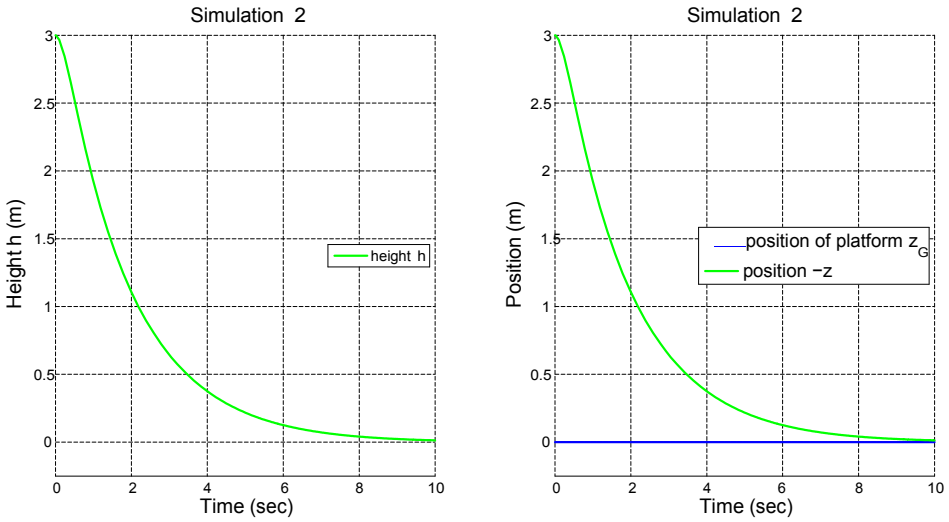

Figure 5.5: Vertical landing on non-oscillating platform

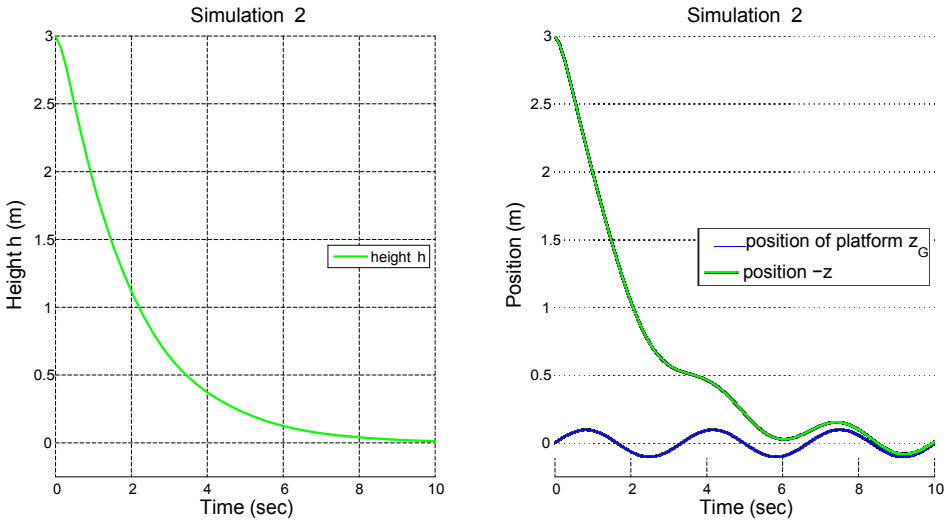

Figure 5.6: Vertical landing on an oscillating platform

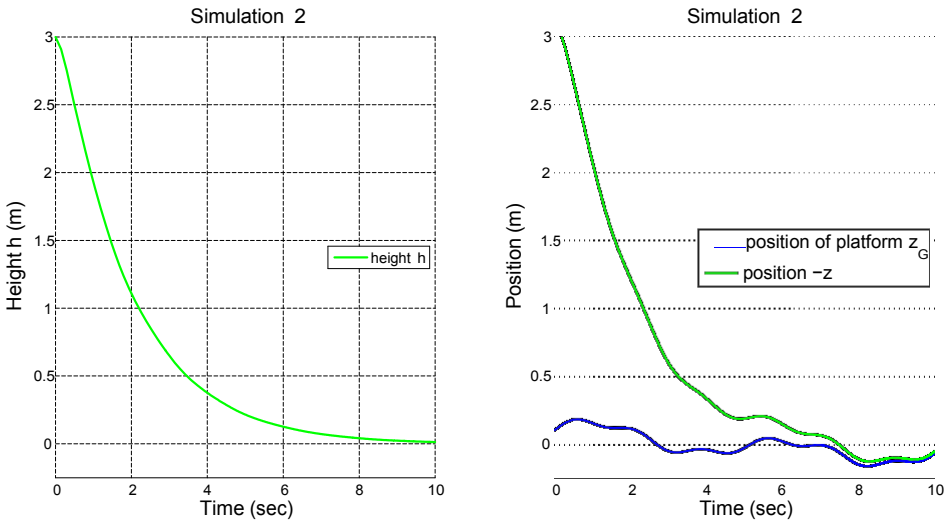

Figure 5.7: Vertical landing on a stochastically oscillating platform

5.2 Experimental Results

For all experiments and flight testing, the main program loop is run at 100Hz. Sensors are configured for a maximum range of 1000 degrees/s for the gyro, ±4*g* for the accelerometer and ±1.3 Gauss for the magnetometer. The MPU6000 samples the sensors at 200Hz before applying the digital low-pass filter with a cut-off frequency of 42Hz. The optical

flow based position control loop is run at 25Hz.

5.2.1 Experiment 1 - Accelerometer and Gyro Sensor Noise

In the first experiment, noise in the accelerometer and gyro measurements is examined. For all of the tests, the readings are taken at approximately half throttle which is where take-off normally occurs. The vibrations from the motors introduce a significant amount of noise into the measurements, especially in the case of the accelerometer. It is important to minimize the noise to obtain accurate attitude estimation. Fig. 5.8 and Fig. 5.9 show the results of the accelerometer and gyro measurement with the APM mounted to the quadrotor using double sided foam tape and no low-pass filtering. The double sided tape offers little in vibration reduction. Applying the digital low-pass filter in the MPU6000 at 42Hz improves the result, as shown in Fig. 5.10 and Fig. 5.11, but noise in the accelerometer reading is still quite significant. The next set of tests were performed with the APM mounted on vibration dampening gel placed only in the 4 corners as shown in Fig. 5.16. Results in Fig. 5.12 and Fig. 5.13, show that even without the application of the digital low pass filter, the noise is greatly reduced. Adding the low pass filter at 42Hz as in Fig. 5.14 and Fig. 5.15 improves the results further. This is the configuration that is chosen for implementation.

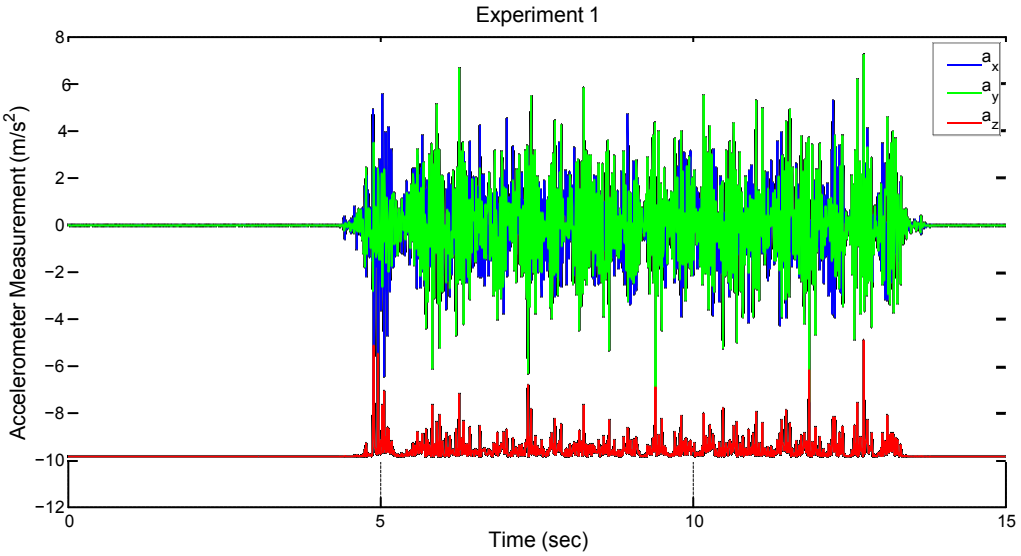

Figure 5.8: Unfiltered accelerometer readings, APM mounted on foam tape

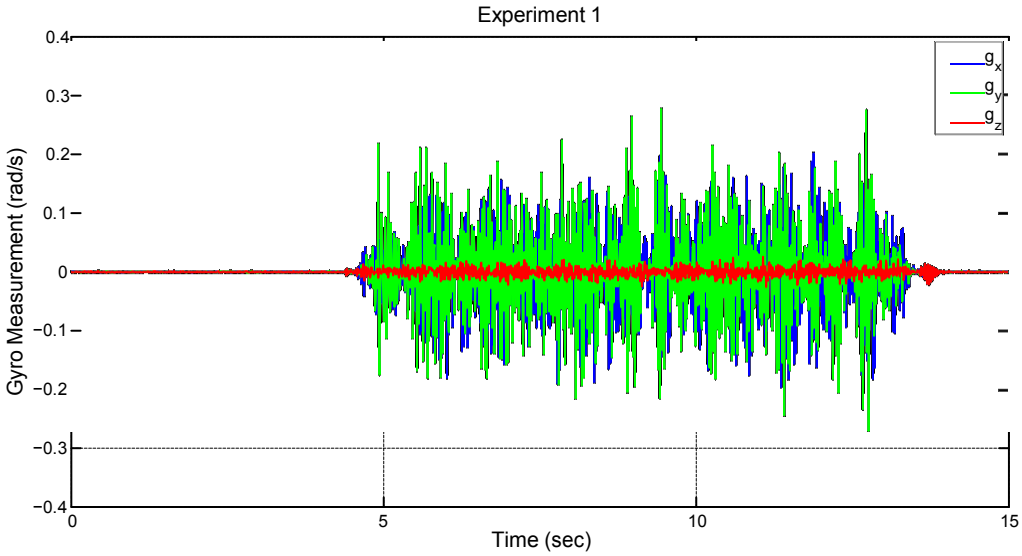

Figure 5.9: Unfiltered gyro readings, APM mounted on foam tape

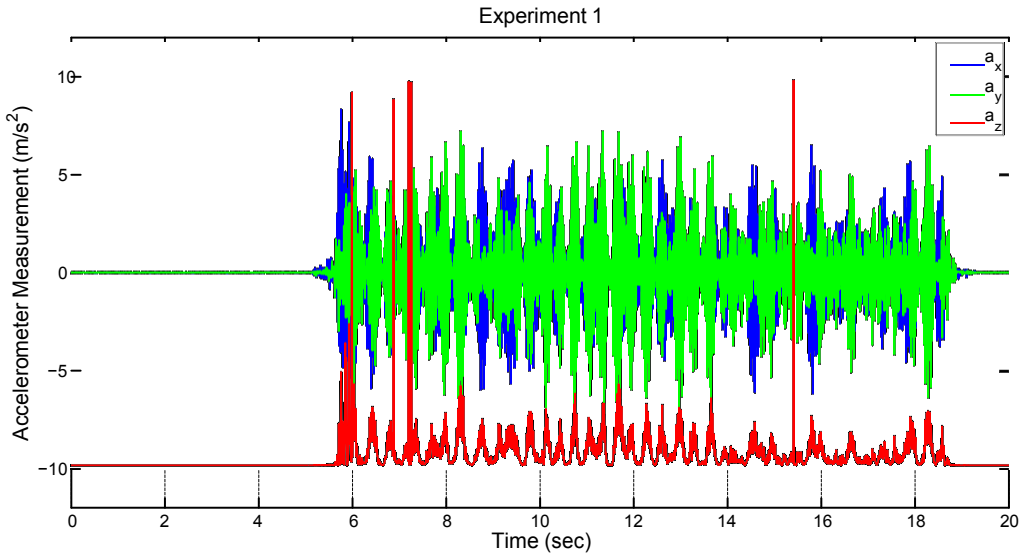

Figure 5.10: Filtered accelerometer readings, APM mounted on foam tape

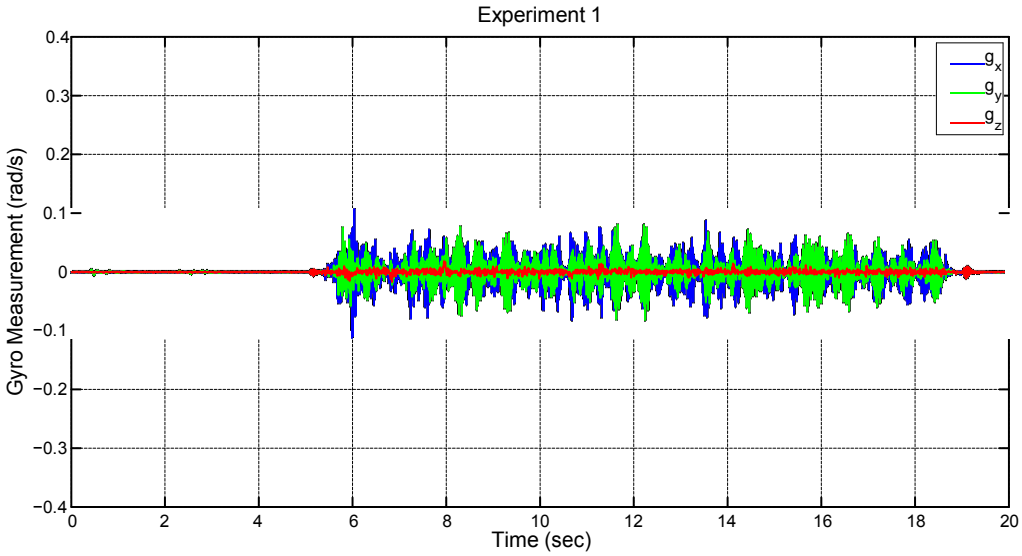

Figure 5.11: Filtered gyro readings, APM mounted on foam tape

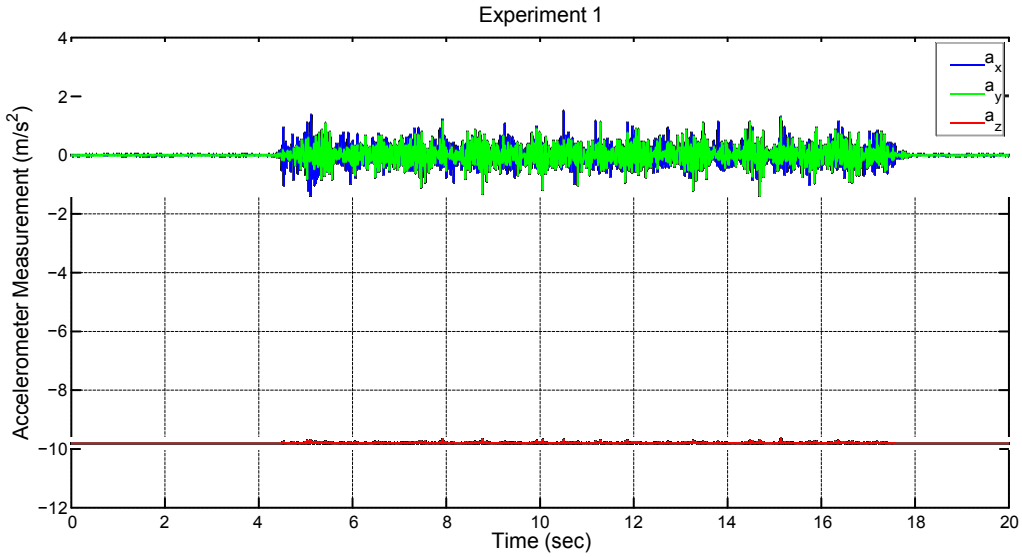

Figure 5.12: Unfiltered accelerometer readings, APM mounted on gel

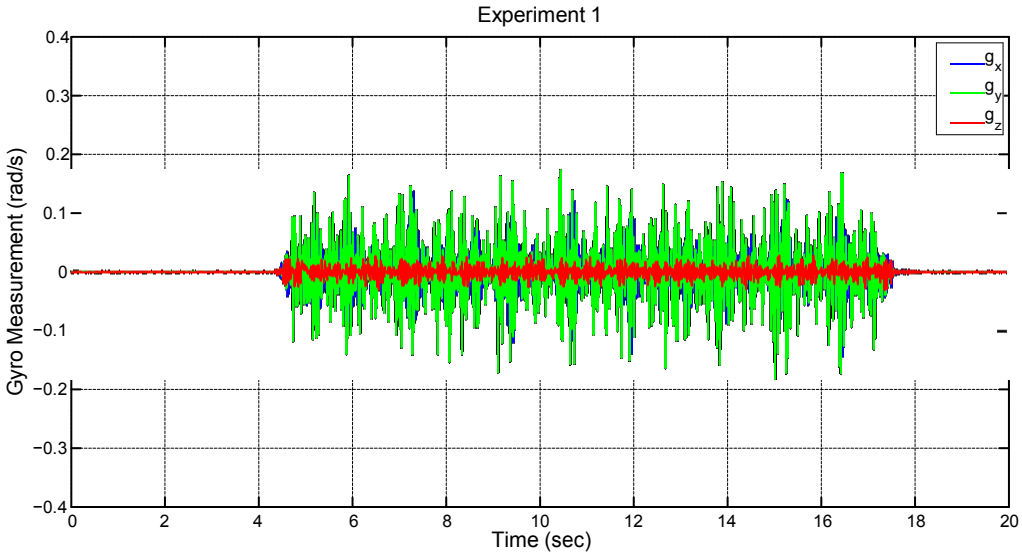

Figure 5.13: Unfiltered gyro readings, APM mounted on gel

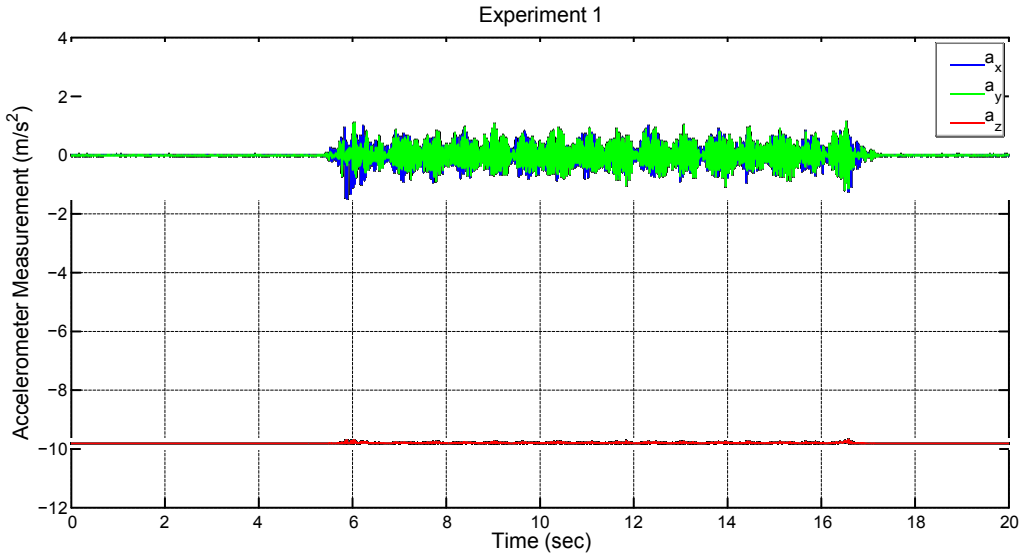

Figure 5.14: Filtered accelerometer readings, APM mounted on gel

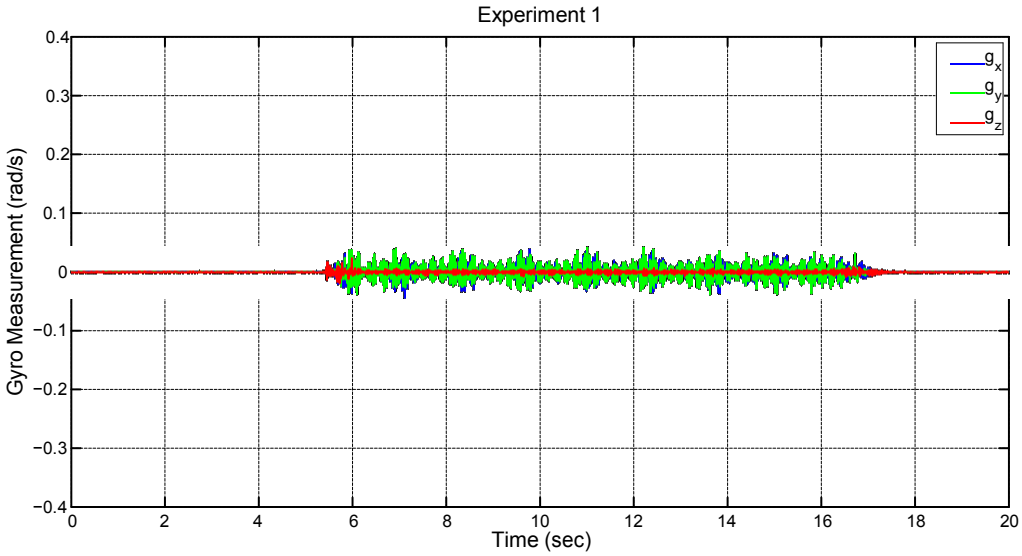

Figure 5.15: Filtered gyro readings, APM mounted on gel

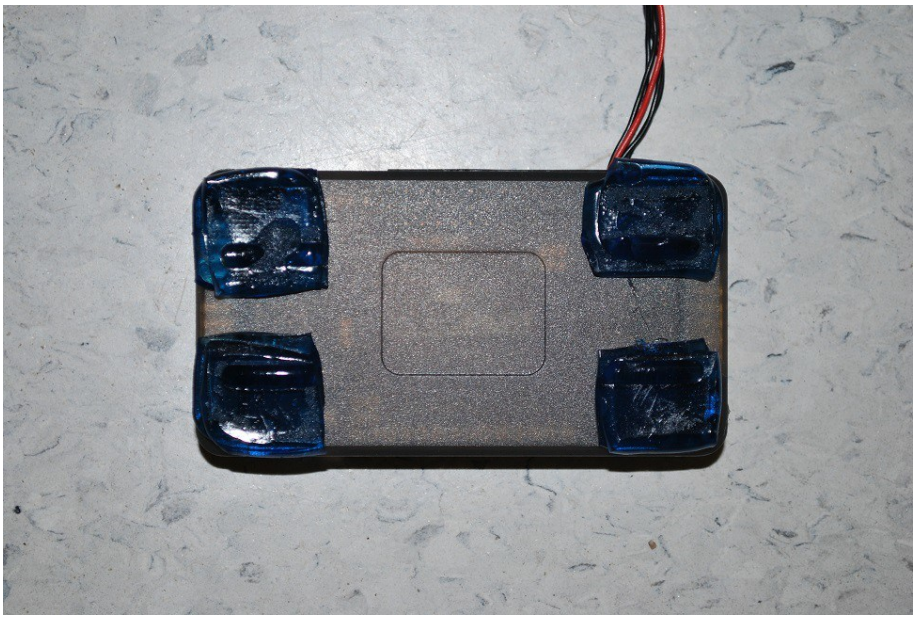

Figure 5.16: Vibration dampening gel on APM

5.2.2 Experiment 2 - Comparison of Attitude Estimation Techniques

In the first part of this experiment, the roll and pitch estimation from the complimentary filter is compared to the roll and pitch obtained from the accelerometer readings alone. Results in Fig. 5.17 and Fig. 5.18 show the comparison with the APM mounted on the double sided foam tape. It is clear that the noise creates large errors in the estimation of roll and pitch when only the accelerometer is used. The result from the complimentary filter is greatly improved with respect to noise in the estimated values. The results in Fig. 5.19 and Fig. 5.20 are obtained with the APM mounted on the vibration dampening gel. Even with the reduced noise, obtaining roll and pitch angles from the accelerometer is still very noisy relative to the complimentary filter results.

In the second part of this experiment, estimation results of roll, pitch and yaw angles using the standard observers and the conditioned observer are compared. For the experiment the gains are tuned to obtain similar performance in terms of speed of convergence. Fig. 5.21 shows the results of the experiment. It is clear that the result of increasing the gains to improve speed of convergence of the standard observer results in increased noise in the estimated results. For the standard observer gains are chosen as $k_1 = 6$, $k_2 = 1.5$, and $k_I = 0.03125$. For the conditioned observer gains are chosen as $k_1 = 1$, $k_2 = 0.5$, $k_3 = k_1/32$, $k_4 = k_2/32$, $k_b = 32$ and $\Delta = 0.02$.

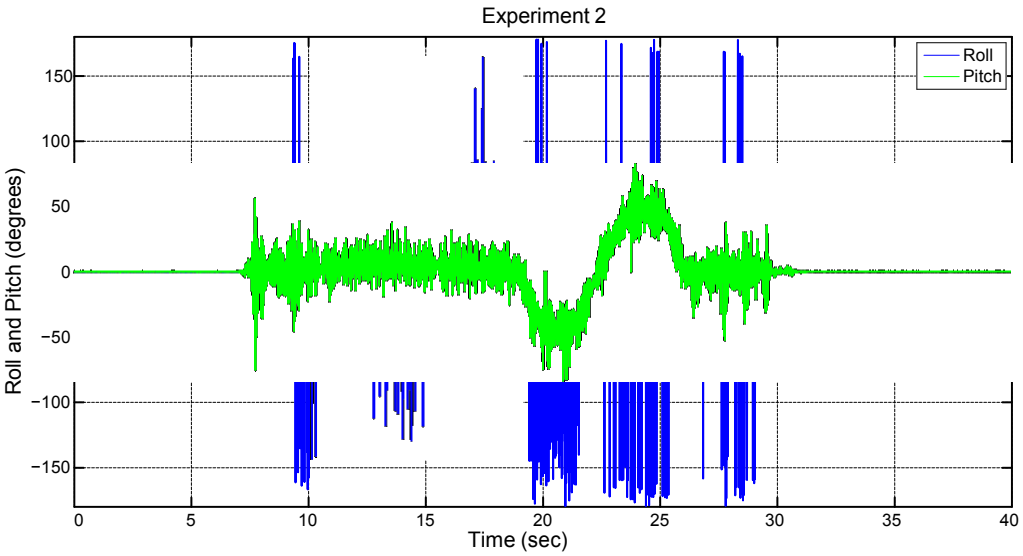

Figure 5.17: Roll and pitch angles from accelerometer data only, APM mounted on foam tape

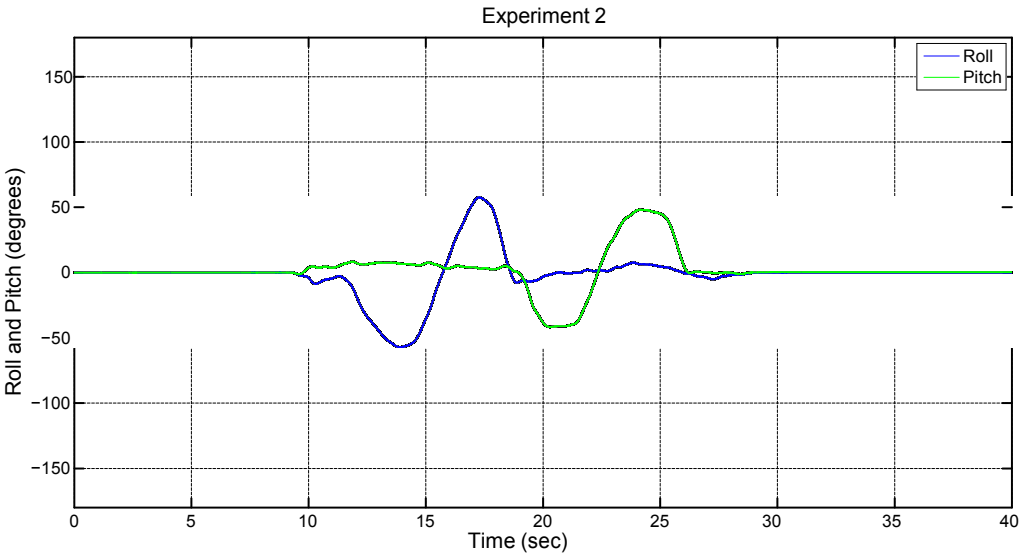

Figure 5.18: Roll and pitch angles from complementary filter, APM mounted on foam tape

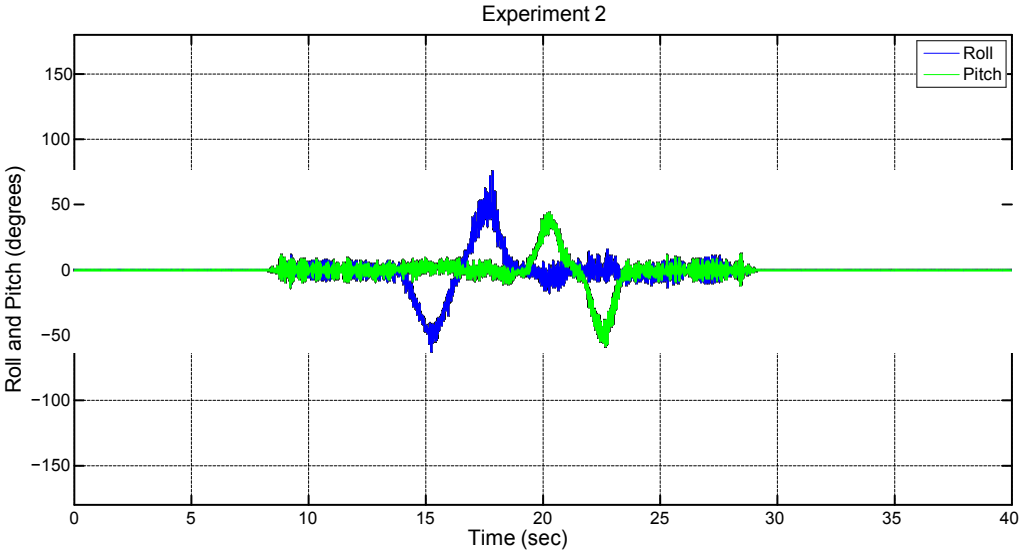

Figure 5.19: Roll and pitch angles from accelerometer data only, APM mounted on gel

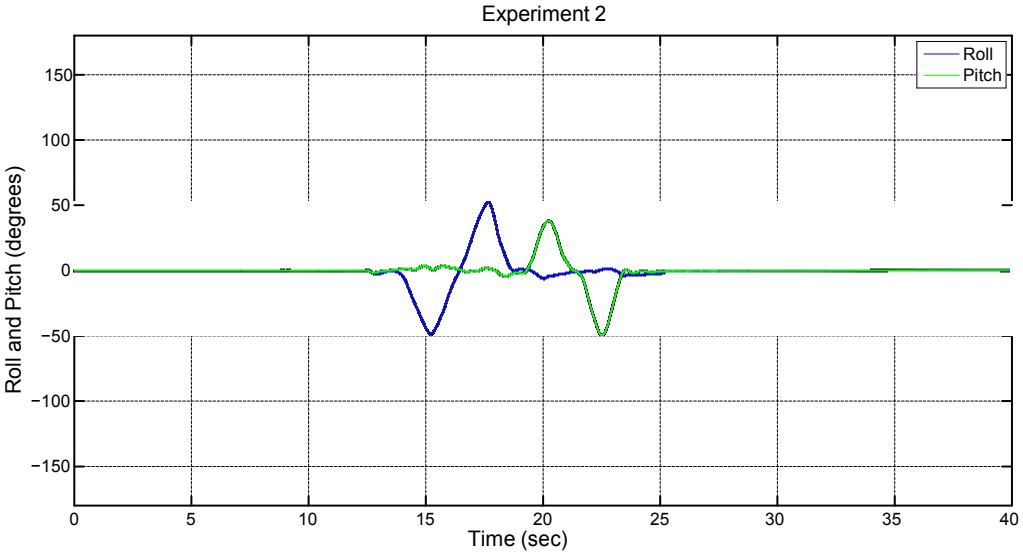

Figure 5.20: Roll and pitch angles from complementary filter, APM mounted on gel

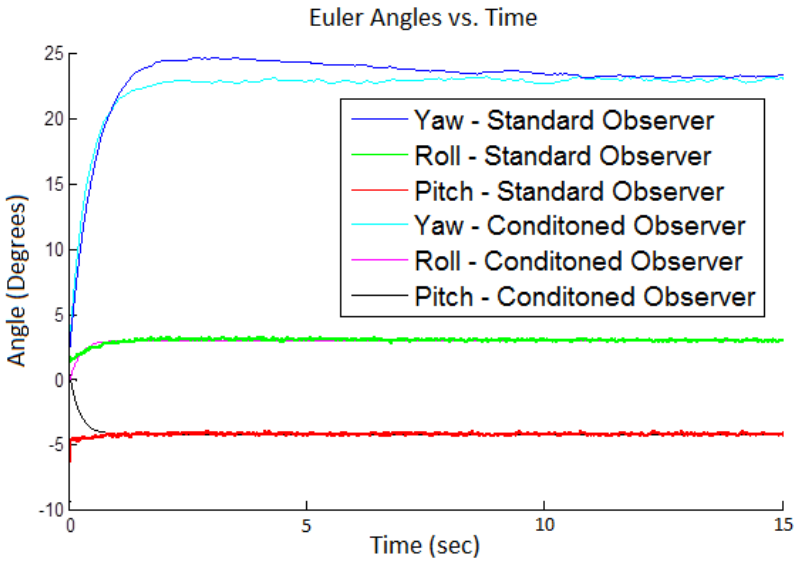

Figure 5.21: Comparison of conditioned and standard observer

5.2.3 Experiment 3 - Optical Flow Based Hovering and Landing

The optical flow experiments are performed using a painted surface with various shades of white, gray and black paint. The optical flow algorithm is run at 25Hz during the experiments. Fig. 5.22 illustrates the results of the hovering controller test using Eq. (4.29) on all three axes with gains set as $k_p = (3.4, 3.4, 27)^T$ and $k_I = (35, 35, 100)^T$. The controller is activated from 16 to 55 seconds and the take-off and landing phases are not shown. The platform is stationary during this test. Fig. 5.23 show the results of the test of the landing controller. During phase 1 the UAV is manually controlled for take-off, during phase 2 position hold is activated and platform motion is started. In Phase 3 the landing controller is activated and the UAV completes the landing. In this case the planar motion (i.e. x and y) is controlled by Eq. (4.29), while the z axis is controlled by Eq. (4.30) with gain $k = 32$ and $w^* = 0.1$. Since no measurements of relative position of the UAV with respect to the target platform are available, the relative position is estimated from the measured optical flow [48]. Define the relative position as $\bar{\xi} = \xi - \xi_G$, then

$$\begin{aligned} \frac{\bar{\xi} - \bar{\xi}_0}{h_0} &= \int_0^t w \exp\left(-\int_0^\tau w_z d\tau\right) dt \\ \frac{h(t)}{h_0} &= \exp\left(-\int_0^t w_z dt\right) \end{aligned} \tag{5.1}$$

where $\xi = (x, y, z)^T$ is the position of the UAV, $\xi_G = (x_G, y_G, z_G)^T$ is the position of the target, h_0 is the initial height and $h(t)$ is the time-varying height. $w = (w_x, w_y, w_z)^T$ is the calculated inertial average optical flow.

In both experiments there is some initial deviation from the zero position accumulated during take-off before the control is activated. However once the controller is activated the position remains fairly constant. In the landing experiment the quadrotor lands at a height greater than zero due to the landing gear. The motion of the platform is created by a person holding and moving the platform vertically up and down. From the simulations it is expected that during the landing manoeuvre the height above the platform decreases monotonically, however during the experiment oscillations are observed.

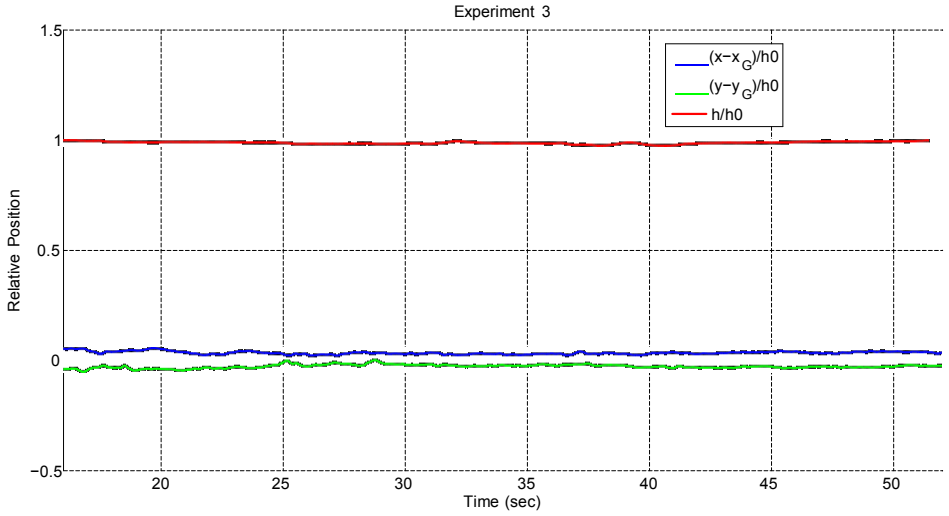

Figure 5.22: Hovering position hold

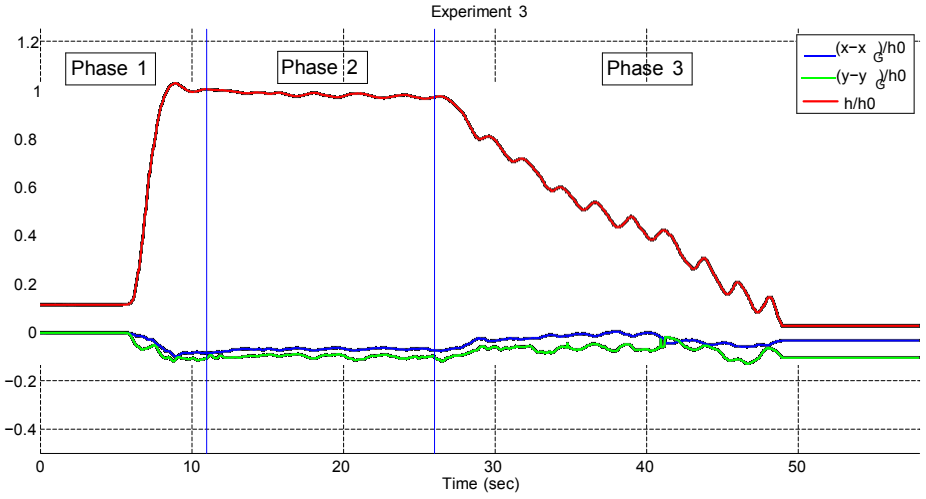

Figure 5.23: Landing on a moving platform

Chapter 6

Conclusion

In this thesis a fully operational quadrotor based on low-cost sensors and embedded processing has been developed. Estimation and control techniques were reviewed and methods that are well suited to practical implementation were chosen. Two non-linear attitude estimators (namely standard and conditioned complementary filters) were compared in real-time running on the embedded hardware. In the presence of sensor noise and magnetic field distortion the conditioned observer was found to perform better than the standard observer. A method for the calibration of the magnetometer was presented and was shown to work well. Some consideration was also given to sensor noise and results are provided for both software filtering and physical vibration dampening. Further work in the area of vibration dampening should be completed to improve estimation results. A non-linear attitude controller in quaternion form was implemented and found to perform acceptably in practice in an indoor environment. Methods to experimentally determine some model parameters of a quadrotor are also presented.

An optical flow based position control scheme was implemented to achieve position hold and the ability to land on a moving platform. Optical flow was calculated from images sent via wireless link from a UAV mounted camera and results were relayed back to the on-board processor for calculation of the position control inputs. The system performed well in the case of a slowly moving landing platform, but eliminating noise from the

optical flow and tuning control gains proved challenging.

Extension of this work could include using optimal gain tuning methods in an effort to improve performance as the gains used in this thesis were obtained through trial and error. Additionally even though all of the estimation and control techniques used are proven stable individually, some robustness and stability analysis should be completed on the full interconnected system. Implementation of more vision based position control techniques using landmarks. Optical flow has the advantage of functioning in the case of an unknown target but in practice requires intensive calculations and is sensitive to the texture of the target as well as shadows and lighting differences. Recently higher performance quadrotor controllers have become available based on the ARM processor. The availability of higher performance hardware may allow for calculation of optical flow onboard the aircraft which would reduce the communication time delay. At the very least overall flight performance could be enhanced by running the program at a higher sampling rate.

Bibliography

[1] J. Leishman, *Principles of Helicopter Aerodynamics*. Cambridge aerospace series, Cambridge University Press, 2002.

[2] K. Munson, *Helicopters and Other Rotorcraft Since 1907*. The Pocket Encyclopedia of World Aircraft in Color, Macmillan, 1969.

[3] R. Brown, "Planes that go straight up open new fields for aviation," *Popular Science Monthly*, vol. 126, no. 3, pp. 13–15, 1935.

[4] G. Hoffmann, D. Rajnarayan, S. Waslander, D. Dostal, J. S. Jang, and C. Tomlin, "The stanford testbed of autonomous rotorcraft for multi agent control (starmac)," *Digital Avionics Systems Conference*, vol. 2, pp. 12.E.4–1–10, 2004.

[5] [Online]. http://aero.stanford.edu/Reports/AHSPaper.pdf.

[6] [Online]. http://hybrid.eecs.berkeley.edu/starmac/.

[7] [Online]. www.aeryon.com.

[8] [Online]. http://www.draganfly.com/.

[9] H. Black, "A passive system for determining the attitude of a satellite," *AIAA Journal*, vol. 2, no. 7, pp. 1350–1351, 1964.

[10] G. Wahba, "A least squares estimate of spacecraft attitude," *SIAM Review*, vol. 7, no. 3, p. 409, 1965.

[11] P. B. Davenport, "A vector approach to the algebra of rotations with applications," *Technical Report TN D-4696, NASA*, 1968.

[12] M. D. Shuster and S. D. Oh, "Three-axis attitude determination from vector observations," *Journal of Guidance, Control, and Dynamics*, vol. 4, no. 1, pp. 70–77, 1981.

[13] M. D. Hua, *Contributions to the automatic control of aerial vehicles*. PhD thesis, Univeriste de Nice-Sophia Antipolis, 2010.

[14] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[15] M. Grewal and A. Andrews, "Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]," *IEEE Trans. on Control Syst. Technol.*, vol. 30, no. 3, pp. 69–78, 2010.

[16] A. Jazwinski, *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970.

[17] E. Lefferts, F. L. Markley, and M. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, 1982.

[18] I. Y. Bar-Itzhack and Y. Oshman, "Attitude determination from vector observations: quaternion estimation," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. AES-21, no. 1, pp. 128–136, 1985.

[19] N. F. Toda, J. L. Heiss, and F. H. Schlee, "Hubble space telescope angular velocity estimation during the robotic servicing mission," *Report TR0066 (5306)-12, Proceedings of the Symposium on Spacecraft Attitude Determination*, vol. 1, pp. 361–370, 2007.

[20] J. Murrell, "Precision attitude determination for multimission spacecraft," *AIAA Guidance and Control Conference*, vol. 78, p. 1248, 1978.

[21] K. Ernandes, B. Joseph, and P. Cefola, "Implementation of a multiplicative extended kalman filter (mekf) for spinning sacecraft attitude determination in the astrodynamics environment (ade)," *Advances in the Astronautical Sciences*, vol. 129, no. 2, 2007.

[22] J. Bijker and W. Steyn, "Kalman filter configurations for a low-cost loosely integrated inertial navigation system on an airship," *Control Engineering Practice*, vol. 16, no. 12, pp. 1509–1518, 2008.

[23] S. Bonnabel, P. Martin, and E. Salaun, "Invariant extended kalman filter: theory and application to a velocity-aided attitude estimation problem," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pp. 1297–1304, 2009.

[24] P. Martin and E. Salaün, "Generalized multiplicative extended kalman filter for aided attitude and heading reference system," in *Proc. AIAA Guidance, Navigation and Control Conference*, pp. 1–13, 2010.

[25] R. Mahony, T. Hamel, and J. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Trans. on Autom. Control*, vol. 53, no. 5, pp. 1203–1218, 2008.

[26] A. Zul Azfar and D. Hazry, "Simple approach on implementing sensor fusion in pid controller for stabilizing quadrotor flight control," *IEEE 7th International Colloquium on Signal Processing and its Applications*, pp. 28–32, 2011.

[27] J. Li and Y. Li, "Dynamic analysis and pid control for a quadrotor," *Proc. of IEEE International Conference on Mechatronics and Automation*, pp. 573–578, 2011.

[28] F. Sharifi, M. Mirzaei, B. Gordon, and Y. Zhang, "Tolerant control of a quadrotor uav using sliding mode control," *Conference on Control and Fault Tolerant Systems*, pp. 239–244, 2010.

[29] M. Huang, B. Xian, C. Diao, K. Yang, and Y. Feng, "Adaptive tracking control of underactuated quadrotor unmanned aerial vehicles via backstepping," *American Control Conference, Baltimore, USA*, pp. 2076–2081, 2010.

[30] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Experimental model predictive attitude tracking control of a quadrotor helicopter subject to wind gusts," *18th Mediterranean Conference on Control and Automation, Marrakech, Morocco*, pp. 1461–1466, 2010.

[31] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Model predictive quadrotor indoor position control," *19th Mediterranean Conference on Control and Automation, Corfu, Greece*, pp. 1247–1252, 2011.

[32] I. Sadeghzadeh, A. Mehta, Y. Zhang, and C. Rabbath, "Fault-tolerant trajectory tracking control of a quadrotor helicopter using gain scheduled pid and model reference adaptive control," *Report for Defence Research and Development Canada*, 2011.

[33] S. J. McGilvray, "Attitude stabilization of a quadrotor aircraft.," Master's thesis, Lakehead University, 2004.

[34] A. Tayebi and S. McGilvray, "Attitude stabilization of a vtol quadrotor aircraft," *IEEE Trans. on Control Syst. Technol.*, vol. 14, no. 3, pp. 562–571, 2006.

[35] L. Marconi, A. Isidori, and A. Serrani, "Autonomous vertical landing on an oscillating platform: An internal-model based approach," *Automatica*, vol. 38, 2002.

[36] X. Yang, H. Pota, M. Garratt, and V. Ugrinovski, "Prediction of vertical motions for landing operations of uavs," pp. 5048–5053, 2008.

[37] S.-R. Oh, K. Pathak, S. Agrawal, H. Pota, and M. Garratt, "Approaches for a tether-guided landing of an autonomous helicopter," *IEEE Trans. Robot.*, vol. 22, no. 3, pp. 536–544, 2006.

[38] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme, "Visually-guided landing of an unmanned aerial vehicle," *IEEE Trans. Robot. Autom.*, vol. 19, no. 3, pp. 371–380, 2003.

[39] C. S. Sharp, O. Shakernia, and S. S. Sastry, "A vision system for landing an unmanned aerial vehicle," *IEEE Int. Conf. Robot. Autom*, pp. 1720–1727, 2001.

[40] M. Shuster, "A survey of attitude representations," *Journal of Astronautical Sciences*, vol. 41, pp. 439–517, 1993.

[41] J. Stuelpnagel, "On the parametrization of the three-dimensional rotation group," *SIAM review*, vol. 6, no. 4, pp. 422–430, 1964.

[42] A. Tayebi, A. Roberts, and A. Benallegue, "Inertial vector measurements based velocity-free attitude stabilization," *IEEE Trans. on Autom. Control*, vol. 58, no. 11, pp. 2893–2898, 2013.

[43] P. Hughes, *Spacecraft attitude dynamics*. Dover Publications, 2012.

[44] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski, "Dynamic modeling and configuration stabilization for an x4-flyer," *Proceedings of the 15th IFAC World Congress*, vol. 15, no. 1, pp. 846–846, 2002.

[45] P. Crouch, "Spacecraft attitude control and stabilization: Applications of geometric control theory to rigid body models," *IEEE Trans. on Autom. Control*, vol. AC-29, no. 4, pp. 321–331, 1984.

[46] M. V. Srinivasan, S. W. Zhang, J. S. Chahl, E. Barth, and S. Venkatesh, "How honeybees make grazing landings on flat surfaces," *Biol. Cybern*, vol. 83, pp. 171–183, 2000.

[47] J. J. Koenderink and A. J. van Doorn, "Facts on optic flow," *Biol. Cybern.*, vol. 56, pp. 247–254, 1987.

[48] B. Herisse, T. Hamel, R. Mahony, and F.-X. Russotto, "Landing a vtol unmanned aerial vehicle on a moving platform using optical flow," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 77–89, 2012.

[49] J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques," *International Journal of Computer Vision*, 1994.

[50] B. Lucas and T. Kanade, “An image regisration technique with an application to stereo vision,” *in Proceedings of Image Understanding Workshop*, pp. 121–130, 1981.

[51] R. F. Vassallo, J. Santos-Victor, and H. J. Schneebeli, “A general approach for ego-motion estimation with omnidirectional images,” *Proc. Omnidirectional Vis.*, pp. 97–103, 2002.

[52] Datasheet: Atmel ATmega2560.

[53] Datasheet: Atmel ATmega32U2.

[54] Datasheet: Invensense MPU-6000.

[55] Datasheet: Honeywell HMC5883l.

[56] M. Barczyk, *Nonlinear state estimation and modeling of a helicopter UAV*. PhD thesis, University of Alberta, 2012.

[57] C. Finlay, S. Maus, C. Beggan, T. Bondar, A. Chambodut, T. Chernova, A. Chulliat, V. Golovkov, B. Hamilton, M. Hamoudi, *et al.*, “International geomagnetic reference field: the eleventh generation,” *Geophysical Journal International*, vol. 183, no. 3, pp. 1216–1230, 2010.

[58] S. Maus, S. Macmillan, S. McLean, B. Hamilton, A. Thomson, M. Nair, and C. Rollins, “The us/uk world magnetic model for 2010-2015,” *NOAA Technical Report NESDIS/NGDC*, 2010.

[59] M. Hua, G. Ducard, T. Hamel, R. Mahony, and K. Rudin, “Implementation of a nonlinear attitude estimator for aerial robotic vehicles,” *IEEE Trans. on Control Syst. Technol.*, pp. 201–213, 2013.

[60] A. Abdessameud and A. Tayebi, “Global trajectory tracking control of vtol-uavs without linear velocity measurements,” *Automatica*, vol. 46, pp. 1053–1059, 2010.