LAKEHEAD UNIVERSITY

MASTERS THESIS

Position Control of a Quadrotor UAV Using Stereo Computer Vision

Author:
Geordi MCGRATH

Supervisor:
Dr. Abdelhamid TAYEBI

A thesis submitted in partial fulfilment of the requirements for the degree of Master of Science

in

Electrical and Computer Engineering

September 16, 2019

Abstract

Quadrotors are a recent popular consumer product, prompting further developments with regards to combining attitude and position control for such unmanned aerial vehicles (UAVs). The quadrotor's attitude (orientation) can be controlled using measurements from small and inexpensive sensors such as gyroscopes, accelerometers, and magnetometers combined in a complementary filter.

The quadrotor's position can be controlled using a global positioning system (GPS); this has shown to be an effective method but is not reliable in urban or indoor environments where quadrotors may be flown. An alternative method such as visual servoing may provide reliable position control in urban environments. Visual servoing uses one or more on-board cameras to detect visual features in their field-of-view. If more than one camera is used, a more accurate position can be determined.

This work discusses the the application of stereo visual servoing to a quadrotor for position control combined in a Kalman filter with IMU readings to provide a better estimate of its position.

Acknowledgements

I would like to sincerely thank my supervisor, Dr. Abdelhamid Tayebi, for everything he has done for me throughout my education here at Lakehead University. His boundless patience, advice, and mentoring are invaluable.

I would also like to thank my fellow graduate students, Zeke Sedor and Shivek Lekhi, for their insights and input toward mutual problems both practical and theoretical.

Finally, I would like to thank my wife, Kayte Sutherland, whose love, patience, and support were integral in the completion of my work.

Contents

Abstract i

Acknowledgements ii

1 Introduction 1
- 1.1 History . . . 1
- 1.2 Thesis Organisation . . . 2

2 Literature Review and Thesis Contribution 3
- 2.1 Rigid-Body Attitude Control . . . 3
  - 2.1.1 Attitude Estimation . . . 3
  - 2.1.2 Attitude Control . . . 5
- 2.2 Position Control . . . 6
  - 2.2.1 Position Estimation . . . 6
  - 2.2.2 Computer Vision . . . 6
- 2.3 Thesis Contributions . . . 8

3 Attitude Representation 9
- 3.1 Direction Cosine Matrix . . . 10
- 3.2 Euler Angles . . . 10
- 3.3 Unit Quaternion . . . 12
- 3.4 Attitude Representation Comparison . . . 13

4 Attitude Estimation 15
- 4.1 Complementary Filter . . . 15
- 4.2 Conditioned Observer . . . 16
- 4.3 Discrete Implementation . . . 17

5 Mathematical Model of a Quadrotor 19
- 5.1 Model Definition . . . 19

6 Experimental Platform: Hardware 22
- 6.1 Airframe and Flight Hardware . . . 22
  - 6.1.1 Quadrotor Frame and Body . . . 23
  - 6.1.2 Motors, Electronic Speed Controllers (ESCs), and Power Supply . . . 24
  - 6.1.3 Control Unit . . . 25
    - Microcontrollers . . . 25
    - Measurement Devices . . . 26
  - 6.1.4 Remote Control . . . 26

6.1.5 Serial Communication 27
6.2 Cameras 27
6.2.1 Image Capturing 27
6.2.2 Wireless Transmitter/Receiver 27
6.3 Base Station Computer 28

7 Experimental Platform: Sensor Calibration **30**
7.1 Accelerometer 30
7.2 Gyroscope 31
7.3 Magnetometer 31
7.4 Cameras 33
7.4.1 Individual Camera Calibration 33
7.4.2 Stereo Camera Calibration 35

8 Position and Velocity Estimation and Control Using Stereo Vision and Kalman Filtering **39**
8.1 Position Estimation Using Stereo Computer Vision 39
8.2 Discrete Kalman Filter 44
8.3 Position Controller 45
8.4 Experimental Results 49

9 Conclusion and Future Work **53**

Bibliography **54**

List of Figures

3.1 NED inertial and body-fixed coordinate frames. 9

6.1 Block diagram of the experimental platform. 22
6.2 Quadrotor experimental platform. 23
6.3 Quadrotor motor configurations. 24
6.4 A2830-12 outrunner motor with $4.7in/rev$ propeller attached. 24
6.5 $20Amp$ ESC used for this work. 25
6.6 Base station computer with XStick (circled in red) and video receivers (circled in yellow). 28
6.7 Cameras mounted on the bottom of the quadrotor in horizontal stereo. 29

7.1 Magnetometer calibration results. 33
7.2 Pinhole camera projection. 34
7.3 Generalized imaging of a scene using two viewpoints.[49] 36
7.4 Rectified imaging of a scene using two viewpoints.[115] 37

8.1 Block diagram of the quadrotor's control system. 39
8.2 The geometry of the pinhole camera projection where Π is the image capture plane, p is the detected point in the image, O_c is the center of mass of the camera, $\mathbf{P}$ is the detected physical point, f is the focal length of the camera, and Z is the calculated depth of the detected point. 40
8.3 Stereo pinhole camera geometry. 41
8.4 Four-pointed star used for horizontal position reference. 42
8.5 Kalman filter block diagram. 46
8.6 Position of the quadrotor in each Cartesian axis vs. time expressed in $\mathcal{I}$. 50
8.7 3D position from autonomous flight using stereo vision for position estimation in metres. 51
8.8 Velocity of the quadrotor in each Cartesian axis vs. time expressed in $\mathcal{I}$. 52

List of Tables

3.1 Attitude representation comparison 13

5.1 Summary of symbols used in the mathematical model. 20

6.1 Motor characteristics 25
6.2 DLPF configurations for the MPU-6000. 26

7.1 Individual intrinsic camera parameters. 35
7.2 Stereo intrinsic camera parameters. 38

8.1 Control gains. 49

List of Abbreviations

UAV	**U**nmanned **A**erial **V**ehicle
GPS	**G**lobal **P**ositioning **S**ystem
VTOL	**V**ertical **T**ake-**O**ff and **L**anding
3D	**3 D**imensional
TRIAD	Three-Axis (**TRI**-axis) **A**ttitude **D**etermination
QUEST	**QU**aternion **EST**imator
SVD	**S**ingular **V**alue **D**etermination
KF	**K**alman **F**ilter
EKF	**E**xtended **K**alman **F**ilter
MEKF	**M**ultiplicative **E**xtended **K**alman **F**ilter
AEKF	**A**dditive **E**xtended **K**alman **F**ilter
PID	**P**roportional **I**ntegral **D**erivative
LQR	**L**inear **Q**uadratic **R**egulator
PD	**P**roportional **D**erivative
GAS	**G**lobal **A**symptotic **S**tability
TDOA	**T**imed **D**elay **O**f **A**rrival
LoS	**L**ine **o**f **S**ight
UWB	**U**ltra **W**ide **B**and
FoV	**F**ield **o**f **V**iew
OpenCV	**Open C**omputer **V**ision
SIFT	**S**cale **I**nvariant **F**eature **T**ransform
SURF	**S**peeded-**U**p **R**obust **F**eatures
FAST	**F**eatures from **A**ccelerated **S**egment **T**est
BRIEF	**B**inary **R**obust **I**ndependent **E**lementary **F**eatures
ORB	**O**riented FAST and **R**otated **B**RIEF
NED	**N**orth-**E**ast-**D**own
APM	**A**rdu**P**ilot **M**ega
IMU	**I**nertial **M**easurement **U**nit
ESC	**E**lectronic **S**peed **C**ontroller
DC	**D**irect **C**urrent
RPM	**R**otations **P**er **M**inute
DLPF	**D**igital **L**ow **P**ass **F**ilter
CMOS	**C**omplementary **M**etal-**O**xide **S**emiconductor
RISC	**R**educed **I**nstruction **S**et **C**omputer
SPI	**S**erial **P**eripheral **I**nterface
I^2C	**I**nter-**I**ntegrated **C**ircuit
PPM	**P**ulse-**P**osition **M**odulation

RC	**R**emote **C**ontrol
DLPF	**D**igital **L**ow-**P**ass **F**ilter
ADC	**A**nalogue to **D**igital **C**onverter
RF	**R**adio **F**requency
USB	**U**niversal **S**erial **B**us
fps	**f**rames **p**er **s**econd
CPU	**C**entral **P**rocessing **U**nit
RAM	**R**andom **A**ccess **M**emory
GPU	**G**raphics **P**rocessing **U**nit
AWGN	**A**dditive **W**hite **G**aussian **N**oise

List of Symbols

$\mathcal{I}$	Inertial reference frame
$\mathcal{B}$	Body reference frame
$\mathcal{C}$	Camera reference frame
${}^{\mathcal{A}}_{\mathcal{B}}R$	Rotation matrix from frame $\mathcal{A}$ to frame $\mathcal{B}$
$\|\cdot\|$	Euclidean norm
$\mathsf{SO}(3)$	Special orthogonal group in three dimensions
I_n	Square identity matrix with size n
$\mathbb{H}$	Set of quaternions
$sk(\cdot)$	Symmetric skew matrix associated with the cross produce
$\dot{f}$	A dot on a character denotes the time derivative
$\ddot{f}$	Two dots on a character denotes the second time derivative
$\mathbf{\Omega}$	Body-referenced angular velocity
$\mathbf{\Omega}_m$	Measured angular velocity
$\mathbf{m}$	Magnetic field
$\mathbf{a}$	Linear acceleration
g	Gravitational constant
$\pi_{\mathbf{f}}$	Projection operator onto the plane onto the plane orthogonal to $\mathbf{f}$
$\mathbf{p}$	Position of the center of mass of the quadrotor in $\mathcal{I}$
$\mathbf{v}$	Velocity of the center of mass of the quadrotor in $\mathcal{I}$
m	Mass of the quadrotor
$\boldsymbol{\tau}$	Torques acting on the quadrotor in $\mathcal{B}$
$\mathcal{T}$	Total thrust from the motors
$\bar{\omega}$	Angular velocity of a rotor
$\boldsymbol{\eta}$	Additive white Gaussian noise
$\mathbf{P}$	World feature point
$\mathbf{\Pi}$	Image plane
$[u, v]$	Pixel coordinates of a detected feature
$[c_x, c_y]$	Image centre in pixel
f	Camera focal length in pixels
O_c	Camera optical centre

Dedicated to my family.

Chapter 1

Introduction

The multirotor Unmanned Arial Vehicle (UAV) has evolved into a versatile and popular flight platform. Its popularity is primarily due to its mechanical simplicity. This simplicity lends itself well to research in a number of different fields including control theory, navigation, real time systems, and robotics. A multirotor UAV refers to a UAV with more than one rotor (a motor with a vertical rotation axis); however, contemporary multirotors have three to twelve rotors on three to eight arms. The quadrotor platform consists of four or eight rotors mounted on four arms. A quadrotor which used four rotors was the platform used for the purposes of this work.

The simplicity of the quadrotor UAV is due to the fact that there are only four moving mechanical components: the four motors. These four motors are mounted at a sufficient distance from the central body of the quadrotor to create large lever arms to induce larger torques along the Cartesian axes. The long lever arms, along with the large thrust-to-weight ratio make the quadrotor a highly agile platform. The low number of moving parts also allows an ease of repair and maintenance. This allows a quadrotor to find uses in a variety of fields such as spreading pesticides on commercial crops [1] or surveillance [2].

1.1 History

Quadrotors have existed for more than a century and were among the first successful Vertical Take-Off and Landing (VTOL) vehicles. Quadrotors didn't gain popularity during their early iterations and were built mainly as military prototypes. Recent advances in low-cost sensors and lightweight materials have revived interest in the quadrotor UAV since they allow construction of a relatively simple and cheap platform capable of robust performance. The first quadrotor was constructed in 1907 by the Bréguet brothers and was capable of liftoff with a human pilot. The Bréguet-Richet Gyroplane No. 1 was constructed of heavy steel girders in a cross configuration with propellers made from four cloth covered surfaces. There was no way to control this vehicle other than varying the speeds of the rotors which made it very unstable. The next significant attempt at building a quadrotor was made by Etienne Oemichen in 1922. His most successful vehicle was the Oemichen No. 2. In 1922 the US army made its first serious attempt to develop a quadrotor aircraft. Dr. George de Bothezat designed a quadrotor vehicle in a cross configuration with six-bladed propellers on each rotor. This aircraft was only able to achieve a maximum altitude of $5m$ and was abandoned [3]. In the 1950s, D.H. Kaplan designed a more refined quadrotor which utilized differential thrusts of opposing rotors for attitude control. This method of controlling a quadrotor is the method that is widely used today. Smaller, unmanned quadrotors are currently the most popular platform because they do not require large amounts of thrust which

a large manned quadrotor would require. One of the first modern small quadrotor UAVs was the Draganflyer, built in Canada by RCToys [4]. Although intended as a toy, it showed that quadrotor technology could be realised on low-cost, mass-produced platform.

1.2 Thesis Organisation

This thesis is divided into nine chapters with **Chapter 1** providing a brief introduction into the background of the quadrotor.

Chapter 2 provides a review of the literature related to the work performed within this thesis.

Chapter 3 describes methods to transform a vector expressed in one coordinate frame to be expressed in another coordinate frame.

Chapter 4 explains the theory and implementation of the attitude estimation algorithm which was chosen for this work.

Chapter 5 focuses on the mathematical model of a quadrotor.

Chapter 6 details the hardware of the experimental platform which was used for this experiment.

Chapter 7 shows how the sensors of the experimental platform were calibrated.

In **Chapter 8**, the determination and control of the linear position and velocity are presented as well as experimental results.

Finally, a conclusion of the presented work and suggestions for future work are given in **Chapter 9**.

Chapter 2

Literature Review and Thesis Contribution

This chapter discusses the literature that was utilized to complete this work. The final goal of this work was to control the position of a quadrotor unmanned aerial vehicle (UAV) using inexpensive sensors in an environment which does not have access to the Global Positioning System (GPS). A quadrotor is a robot which uses four rotors (vertically mounted propellers) as its actuators to control its movement. It is of particular interest because it is an aerial vehicle which is capable of vertical take-off and landing (VTOL) manoeuvres, can be built small enough for indoor use, and it is much simpler mechanically than a helicopter. The simplicity of a quadrotor over a helicopter is due to a quadrotor being able to use propellers which have a fixed pitch; the pitch of a propeller is the angle of tilt that the propeller has with respect to its plane of rotation. This simplicity allows a quadrotor to be more easily miniaturized which is useful if there is a desire to operate the quadrotor indoors. Operating a quadrotor indoors is useful for recreational or research purposes. Research applications benefit from operating a quadrotor indoors since any equipment which may be required doesn't need to be moved to an alternate location, the lab equipment can remain as permanent fixtures which allows for a faster transition from experimentation to data analysis. An indoor environment can also be advantageous because it is capable of simulating an urban environment which can be deprived of GPS signals due to the tall buildings.

Movement of the quadrotor is accomplished by changing the speeds at which its rotors are spinning which rotates the quadrotor and allows its thrust to be directed in a desired direction. The rotation, or orientation, of an object with respect to a reference is referred to as its attitude; control of a quadrotor's attitude is the first step toward being able to control its position. The first section of this chapter explains the literature which was used to first determine the attitude and then to control it. The subsequent section of this chapter discusses methods of position control of UAVs in a general way. The following section specifically focuses on vision-based position control of UAVs. The final section outlines the accomplishments and contributions of this work.

2.1 Rigid-Body Attitude Control

2.1.1 Attitude Estimation

The rotation, or orientation, of an object with respect to a reference is referred to as its attitude. Rotations around the x-y-z Cartesian axes in aerospace applications are referred to as the roll, pitch, and yaw rotations, respectively. To control the attitude of an aerial rigid-body, the

attitude must first be determined. The three dimensional (3D) attitude of an object cannot be directly measured. Due to this, the attitude must be estimated based upon measurements from sensors – such as a magnetometer, a gyroscope, and an accelerometer – or inferred from other parameters. Each of these sensors provide different measurements with respect to a coordinate frame that is attached to the rigid-body; these measurements are therefore referred to as "body-referenced." The magnetometer provides a body-referenced measurement of the local magnetic field, the gyroscope measures the body-referenced angular velocity, and the accelerometer provides a body-referenced measurement of acceleration forces acting on the rigid body. The topic of coordinate frames and how they are related to one another through a rotation are explained in detail in chapter 3.

Some simple methods exist for attitude determination. The simplest method would be to directly integrate the gyroscope measurements to determine the rotation. Unfortunately, direct integration of the gyroscope measurements may cause the attitude estimation to diverge due to measurement noise or bias. Other simple methods of attitude estimation which rely solely on inertial vector measurements such as the Three Axis Attitude Determination (TRIAD) [5], QUaternion ESTimator (QUEST) [6], or Singular Value Decomposition (SVD) of the rotation matrix [7]. The TRIAD algorithm was one of the earliest attitude estimation algorithms proposed in 1964 by H. D. Black. The TRIAD algorithm uses inertial measurements and requires two non-collinear vectors and independent knowledge of the position of the rigid body to determine its attitude. The TRIAD algorithm was used primarily on man-made Earth satellites and celestial bodies were used for position determination. Attempts were made to improve on the TRIAD algorithm which resulted in the QUEST and SVD algorithms. The QUEST algorithm was a proposed solution to an optimization problem known as Wahba's problem [8]. Wahba's problem was to determine a solution to minimize a cost function of two vectors that are related by a rotation. The QUEST algorithm's innovation was to use a quaternion (a four parameter vector) to relate the two vectors instead of the nine parameter rotation matrix. Another solution to Wahba's problem was to determine the SVD of the rotation matrix. This had the benefit of providing a convenient method for the computation of the covariance matrix of the attitude estimation and providing the eigenvalues and eigenvector of the covariance matrix. Knowing the maximum eigenvalue and the eigenvector provided knowledge about the maximum magnitude and direction of the largest component of the attitude uncertainty. Unfortunately, imperfect measurements or imprecise knowledge of the considered inertial vectors can generate significant errors using these methods; therefore, an alternate method needs to be considered.

One particular method which was proposed in 1960 was the Kalman Filter (KF) [9]. The KF operates using a two step process:

1. Predict the current state variables based upon previous state variable estimates using the dynamical model of the system.

2. Update the state variable estimates using weighted state variable predictions and noisy sensor data.

The KF was not adopted immediately because it was originally designed to work with linear systems and the dynamics of an aerial vehicle are inherently non-linear due to the rotational dynamics. The Extended Kalman Filter (EKF) [10], [11] was introduced in 1970 and solves this problem by linearising the system around the current best estimate of the attitude to then make the next estimate of the attitude. The EKF and its variants have since become the workhorse of satellite attitude estimation. Variants of the EKF can be divided into three general classes:

1. Minimal representation EKF which uses three parameters (the minimum number of parameters required) to represent the attitude. These three parameters are the Euler angles which are explained in detail in section 3.2.
2. Multiplicative EKF (MEKF) which uses the product of the estimated attitude (expressed by a non-singular representation) and a deviation from this estimate to estimate the attitude.
3. Additive EKF (AEKF) which uses the sum of the attitude estimate (expressed by a non-singular representation) and the attitude error to estimate the attitude.

A key drawback of using the minimum number of parameters for the attitude representation is that three-dimensional parametrization cannot exist without singularities for the rotation group [12]. The minimum number of parameters required for a singularity free representation is four. Four-dimensional parametrizations are known as "quaternions" and are discussed in more detail in section 3.3.

Recently, computationally efficient, non-linear attitude estimation methods have been used [13]–[20]. These methods are referred to as "complementary" filters because they rely on the fusion of data from different sensors. The complementary aspect is that one sensor has its higher frequencies filtered out and the other sensor has its lower frequencies filtered out. The measurements from the accelerometer are filtered with a low-pass filter and the gyroscope measurements are filtered with a high-pass filter and are merged to obtain an attitude estimation. The complementary filter has proven popular in recent years because it is a nonlinear filter and can be used to more accurately estimate the attitude. A key assumption that complementary filters employ is that the accelerometer is measuring only the gravitational acceleration force exerted on the aerial vehicle, expressed in the body-fixed frame. This works for low linear acceleration applications such as satellites; however, for agile applications such as quadrotors, if the accelerometer measurements contain relatively high linear accelerations of the body then the previous assumption cannot be relied upon to guarantee an accurate attitude estimate. To compensate for this issue, a velocity-aided observer could be employed [17], [20]. These methods use an additional linear velocity estimate function in the innovation term of the observer to obtain a more accurate estimate of the direction of the gravitational force to obtain a better estimate of the attitude. The translational velocity could be obtained using the GPS; however, the goal of this work was to implement a position control system which can operate without the use of the GPS. The attitude estimation method used within this work was the "conditioned observer" complementary filter and is presented in detail in chapter 4.

2.1.2 Attitude Control

Attitude control of rigid bodies consists of determining an appropriate control torque using an estimated attitude and angular velocity. Early work of attitude control mainly dealt with man-made Earth satellites. Methods of early attitude control included using gas reaction jets, rockets, vapour jets, or momentum exchange devices [21]–[23] which are still used to varying degrees today [24]. Within the context of quadrotors, torque is generated through the speed differentials of the rotors. In practice, these torques have been controlled using linear approaches such as Proportional Integral Derivative (PID) [25] or Linear Quadratic Regulator (LQR) [26] controllers. However, these approaches fail to work in the presence of agile manoeuvres or manoeuvres causing large attitude errors since the linearisation around the operating point is unable to encapsulate

a domain beyond the linearised domain. An approach which has had success in quadrotor applications is a hierarchical method which consists of an inner control loop for the body angular rate and an outer loop for the attitude control [27], [28]. The inner loop operates at a much higher frequency than the outer loop allowing for the inner loop to converge before it receives the command signal from the outer loop. The advantage of this control scheme is that it simplifies the controller design. Nonlinear techniques such as backstepping, feedback linearisation, and sliding mode control have been successfully implemented [29]–[31] with the caveat that they are model dependent. A model-independent, Proportional Derivative (PD) controller is proposed in [32] where the proportional part is in terms of the vector part of the quaternion and the derivative part is in terms of the angular velocity. This model-independent controller has been proven to provide Global Asymptotic Stability (GAS) and was utilized for this work.

2.2 Position Control

2.2.1 Position Estimation

To control the position of a mobile robot or UAV, the position must first be estimated. One of the most common methods of determining the position is the Global Positioning System (GPS). The method the GPS uses for localization is Timed Delay of Arrival (TDOA). The GPS performs this using signals received from Earth-orbiting satellites. GPS measurements can be fused with additional sensors such as rate gyroscopes, magnetometers, odometers, and heading sensors [33], [34] to improve the position estimation. For position estimation methods which utilise the GPS, the inertial sensors are used primarily during periods of GPS unavailability because GPS signals are transmitted at a frequency ($1176.45 MHz$)[35] which is not able to bend around objects so requires continuous line-of-sight (LoS) to determine position. This makes GPS unreliable in indoor or urban environments and therefore necessitates the use of alternate methods. Other methods of position localization are performed using odometry [36], [37], vision [38], or other wireless signals such as ultrasonic range finders, Ultra Wide Band (UWB), or cell phone signals [39]–[41]; a survey of the methods utilised with wireless signals is presented in [41]. Odometry is used to determine a vehicle's translation and rotation by mathematically integrating the change in position over time (linear velocity) and change in rotation over time (angular velocity); however, as mentioned above, direct integration of sensor data can cause divergence over time due to noise or bias. Ultrasonic range finders have been used on aerial vehicles for obstacle avoidance and low-level altitude measurements [42]; however, position estimation would require the environment to be known which will be avoided if possible. UWB position estimation uses a series of transmitters/receivers and TDOA to determine position, similar to the GPS but is able to travel through objects due to the frequencies used [43]; however, UWB requires transmitter/receiver nodes to be set up in the environment where position control will be performed. Visual servoing using computer vision was chosen for this work for position estimation and control. The literature of computer vision will be discussed further in the following section.

2.2.2 Computer Vision

Computer vision that is used to control the position of a robot is called "visual servoing." The concept of visual servoing seems simple to humans because most use it in their daily lives to walk, read, or drive. There are two methods of visual servoing:

1. Stand-alone [44] visual servoing: where the cameras are mounted independently from the robot, observe the robot, and provide feedback for future movements.
2. Eye-in-hand visual servoing: the camera is mounted on the robot itself.

Fixed-camera visual servoing has more limited applications since to properly utilise this type of visual servoing the robot must remain in the Field of View (FoV) of the camera. This method has been used mainly with fixed-position robots [45], [46]. Most mobile robots use the eye-in-hand visual servoing method. This work utilises eye-in-hand cameras to control the position of a quadrotor.

To perform visual servoing, the cameras must first be calibrated to eliminate distortion and be able to transform points captured by the camera into world coordinates [47]–[49]. The camera calibration methods utilised in this work are presented in section 7.4.1. Some literature assumes an ideal camera without calibration, briefly mention the calibration, or perform work with uncalibrated cameras [50]–[52]. However, most experiments using visual servoing utilise calibrated cameras [44], [53], [54], even if it isn't stated [55], and their results suffer due to poor calibration [54], [56], [57]. Camera calibration is performed using a known pattern to estimate the intrinsic parameters of the camera. Estimating these parameters is a tedious process [54], however libraries which contain calibration algorithms, such as Open Computer Vision (OpenCV), are freely available. The calibration method utilised follows the method of [58] for individual camera calibration and a version of the eight-point algorithm [59]–[62] for the stereo vision calibration.

After the cameras have been calibrated, the cameras are then used to determine their position with respect to the observed scene using simple triangulation; however, to perform the triangulation, visual features must be found which are common to both observing cameras. Matching features between stereo vision images, also known as the correspondence problem, is considered to be the hardest and most significant problem of stereo vision [63]. Some of the earliest methods of detecting features in an image are the Moravec Corner Detector and the Canny Edge Detector [64]–[66]. The Moravec Corner Detector relied on directional variance in small square areas to find maxima which could be correlated between multiple images to detect corners but could not distinguish between the true corner of an object or the edge of an object in the background intersecting with the line of an object in the foreground. The Canny Edge Detector uses the first derivative of a Gaussian function to find steps of intensity within an image and providing an output showing linear contours of the most likely places for edges to occur. A problem with edge detection and contour matching is that a simple object like a square could be matched with salt noise (single white pixels) or the chosen object may be mismatched if the matching criteria are not chosen carefully. This work attempted to use a square, encountered these problems, and used a more complex shape as a reference. More recent feature detectors are designed so that scale or orientation do not affect the detected features used for correspondence. A popular method is the Scale Invariant Feature Transform (SIFT) algorithm [67]. The SIFT algorithm builds on the work of Moravec and Canny by detecting points of high contrast and is able to detect features that are invariant to scale due to its method of subsampling the image. Despite its age, the SIFT algorithm is still the best method of feature detection and continues to see use [68]. Other methods of feature detection are the Speeded-Up Robust Features (SURF) and Oriented FAST [69] and Rotated BRIEF (ORB) algorithms [70], [71]. In [71], the SIFT, SURF, and ORB algorithms are compared and the conclusion is that the SIFT algorithm achieves the most correct matches but is slow and the ORB algorithm achieves the second most correct matches but is

faster than the other two algorithms. Other algorithms can find position relative to known objects as pre-defined 3D models [54]. Matched points can either be utilised by detecting the motion of the camera through optical flow [72], [73] or by matching points of images captured by two different cameras [48], [68], [71]. Simplified methods of feature extraction also exist which binary images such as contour detection [74] or linear feature detection and extraction [75]. This work used the SURF algorithm and borrows from the work of [68] to utilise the GPU of the base station computer to allow for real time processing of the captured images and feature matching.

Computer vision for robots has mostly been used with robot manipulators [44], [54], [76] and mobile robots [77]–[79]. Early work which combines vision and VTOL UAVs used custom hardware for the vision task [80]–[82]. Off-the-shelf hardware and eye-in-hand cameras (camera mounted on the robot) have been used mainly for autonomous landing of VTOL UAVs [83]–[86] and obstacle avoidance [87], [88]. Position control of UAVs has mainly been performed using monocular vision [75], [89], [90] while position control using stereo vision has mostly been performed using mobile robots.

2.3 Thesis Contributions

- Real-time implementation of a stereo-vision based position estimation using a Kalman filter.
- Real-time implementation of a stereo-vision based position control scheme on a quadrotor UAV.

Chapter 3

Attitude Representation

The attitude of an object refers to the orientation of that object in space. The orientation of an object describes the rotation of that object with respect to a reference coordinate frame. The reference frame is typically attached to a large object such as the Earth or the Sun. There are many different methods of expressing the attitude of an object, depending on the need of the application. The most common methods of attitude representation are: the direction cosine matrix, Euler angles, and the unit quaternion; a detailed survey of attitude representations is presented by Schuster in [91]. In aerospace applications, coordinate frames are represented by the right-hand, North-East-Down (NED) coordinate system. NED means the unit vectors which form the x-y-z axes and are oriented to point forward, right, and down, respectively. The following discussion refers to two coordinate frames which are differentiated from one another by their subscripts. The static coordinate frame, called the *reference* or *inertial* frame, is denoted by the subscript $\mathcal{I}$. The inertial frame is used as a fixed reference point with respect to any other frames. The coordinate frame affixed to the center of mass of the aircraft, called the *body-fixed* frame, is denoted by the subscript $\mathcal{B}$. The body-fixed is used to represent the aircraft and rotates with the aircraft. The aircraft's orientation can then be described be determining a rotation which aligns the body-fixed frame to the inertial frame. These subscripts are used in the following subsections to describe vectors expressed in its respective coordinate frame. An example of the coordinate frames is shown in fig. 3.1.

This chapter describes methods to transform a vector expressed in one coordinate frame to be expressed in another coordinate frame. First, the direction cosine matrix is discussed in section 3.1. In section 3.2, the Euler angles are discussed and a rotation matrix is constructed using the Euler

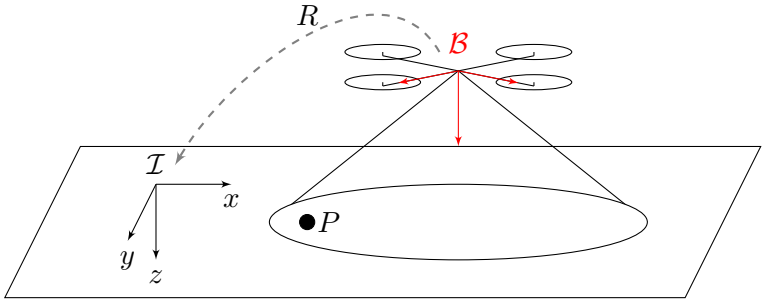


FIGURE 3.1: NED inertial and body-fixed coordinate frames.

angles. Then the unit quaternion is discussed in section 3.3 and a rotation matrix is constructed using the quaternion parameters. Last, the methods are compared in section 3.4.

3.1 Direction Cosine Matrix

The direction cosine matrix, also simply called the rotation matrix, provides a global and unique representation of an object's attitude. Let $\hat{\mathbf{x}}_{\mathcal{I}}$, $\hat{\mathbf{y}}_{\mathcal{I}}$, $\hat{\mathbf{z}}_{\mathcal{I}}$ and $\hat{\mathbf{x}}_{\mathcal{B}}$, $\hat{\mathbf{y}}_{\mathcal{B}}$, $\hat{\mathbf{z}}_{\mathcal{B}}$ be the unit vectors which form the axes of the inertial and body-fixed frames respectively. The direction cosine matrix expresses the rotation of the body-fixed frame, $\mathcal{B}$, with respect to the inertial frame, $\mathcal{I}$, and is given by:

$$
{}^{\mathcal{I}}_{\mathcal{B}}R = \begin{bmatrix} \hat{\mathbf{x}}_{\mathcal{B}} \cdot \hat{\mathbf{x}}_{\mathcal{I}} & \hat{\mathbf{y}}_{\mathcal{B}} \cdot \hat{\mathbf{x}}_{\mathcal{I}} & \hat{\mathbf{z}}_{\mathcal{B}} \cdot \hat{\mathbf{x}}_{\mathcal{I}} \\ \hat{\mathbf{x}}_{\mathcal{B}} \cdot \hat{\mathbf{y}}_{\mathcal{I}} & \hat{\mathbf{y}}_{\mathcal{B}} \cdot \hat{\mathbf{y}}_{\mathcal{I}} & \hat{\mathbf{z}}_{\mathcal{B}} \cdot \hat{\mathbf{y}}_{\mathcal{I}} \\ \hat{\mathbf{x}}_{\mathcal{B}} \cdot \hat{\mathbf{z}}_{\mathcal{I}} & \hat{\mathbf{y}}_{\mathcal{B}} \cdot \hat{\mathbf{z}}_{\mathcal{I}} & \hat{\mathbf{z}}_{\mathcal{B}} \cdot \hat{\mathbf{z}}_{\mathcal{I}} \end{bmatrix}. \tag{3.1}
$$

Here, $\mathbf{u} \cdot \mathbf{v} = \|\mathbf{u}\|\|\mathbf{v}\| \cos(\theta)$ is the dot product between two vectors and θ is the angle between these vectors. The direction cosine matrix, R, is a member of the special orthogonal group in three dimensions denoted by $\mathsf{SO}(3)$ and defined as:

$$
\mathsf{SO}(3) := \left\{ R \in \mathbb{R}^{3\times 3} \mid R^{\top}R = I_3, \ \det(R) = \pm 1 \right\}, \tag{3.2}
$$

where I_3 is the square identity matrix of size three. If both coordinate frames which are related through a rotation matrix are right-handed, then $\det(R) = +1$ [92]. Successive rotations of a coordinate frame can be performed by multiplication of the rotation matrices in the same sequential order that the rotations were performed. The result of combining rotation matrices in this way will result in a single rotation matrix that contains the totality of the rotations, *i.e.*

$$
{}^{\mathcal{B}}_{\mathcal{D}}R\,{}^{\mathcal{A}}_{\mathcal{B}}R \equiv {}^{\mathcal{A}}_{\mathcal{D}}R \tag{3.3}
$$

Let $\mathbf{v}_{\mathcal{I}}, \ \mathbf{v}_{\mathcal{B}} \in \mathbb{R}^3$ be vectors in $\mathcal{I}$ and $\mathcal{B}$ respectively. Assuming there is no displacement between the origins of the two coordinate frames, these vectors are related by

$$
\mathbf{v}_{\mathcal{B}} = R^{\top}\mathbf{v}_{\mathcal{I}}. \tag{3.4}
$$

Though the direction cosine matrix is a global and unique representation of an object's attitude, a key drawback is that this representation requires the estimation of nine unique parameters which makes it computationally expensive. In the following section, the Euler angles will be presented which only use three parameters for expressing an object's attitude.

3.2 Euler Angles

The Euler angles of an object are three parameters which can describe the attitude of an object. The Euler angles of a body are obtained through three successive rotations of a coordinate frame and intermediate frames which are obtained after each rotation. Two successive rotations cannot be about the same axis; however, the first and third rotation can be about the same axis. There are twelve different sets of Euler angles; six symmetric sets whose first and third rotations are about the same axis and six asymmetric sets where each rotation occurs about a different axis. The twelve different sets of rotations are:

x-y-x,	x-z-x,
y-x-y,	y-z-y,
z-x-z,	z-y-z,
x-y-z,	x-z-y,
y-x-z,	y-z-x,
z-x-y,	z-y-x,

In each set of rotations, the rotation axis is chosen from left to right. Within this work, the Euler angle representation used is the z-y-x rotation sequence. Rotations about these axes using this rotation sequence are referred to as the yaw, pitch, and roll angles respectively.

The Euler angles can be used to construct a rotation matrix by expressing each axis rotation as a direction cosine matrix. A rotation about the z-axis by an angle of ψ is expressed as

$$R_z(\psi) = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.5}$$

where $c\psi = \cos(\psi)$ and $s\psi = \sin(\psi)$. Similarly, rotations about the y- and x-axes by angles of θ and ϕ, respectively, are expressed as

$$R_y(\theta) = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}, \tag{3.6}$$

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix}, \tag{3.7}$$

where $c\phi = \cos(\phi)$, $s\phi = \sin(\phi)$, $c\psi = \cos(\psi)$, and $s\psi = \sin(\psi)$. The above axis rotations about individual axes are then combined using matrix multiplication to yield a rotation matrix which expresses an object's rotation using the Euler angles. The rotation matrix using the z-y-x sequence of rotations is expressed as:

$$R = R_z(\psi)\, R_y(\theta)\, R_x(\phi) = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}. \tag{3.8}$$

The Euler angles are useful because they provide an attitude representation which uses only three parameters, the minimum number of parameters required to represent the attitude. The Euler angles are also an intuitive method of visualising an object's rotation and they provide a unique representation for each orientation. The key drawback of the Euler angles is that they do not provide a global representation of the attitude. At a rotation of $\theta = \pi/2$, the dynamics of the Euler angles are singular. The most commonly used non-singular attitude representation is the unit quaternion which is presented in the following section.

3.3 Unit Quaternion

The unit quaternion is an attitude representation which uses only four parameters and is global. A unit quaternion is defined on the four dimensional Hamiltonian space as:

$$\mathbb{H} \coloneqq \left\{ \mathbf{Q} \in \mathbb{R}^4 \mid \|\mathbf{Q}\|^2 = 1 \right\}, \tag{3.9}$$

where $\|\mathbf{x}\|$ denotes the vector norm of $\mathbf{x}$ and

$$\mathbf{Q} = \begin{bmatrix} \cos(\gamma/2) \\ \hat{k}_x \sin(\gamma/2) \\ \hat{k}_y \sin(\gamma/2) \\ \hat{k}_z \sin(\gamma/2) \end{bmatrix} = \begin{bmatrix} \cos(\gamma/2) \\ \hat{\mathbf{k}} \sin(\gamma/2) \end{bmatrix} = \begin{bmatrix} q_0 \\ \mathbf{q} \end{bmatrix}. \tag{3.10}$$

The unit vector $\hat{\mathbf{k}} = [\hat{k}_x \; \hat{k}_y \; \hat{k}_z]^\top$ is the axis of rotation and γ is the rotation about $\hat{\mathbf{k}}$. In eq. (3.10), $\mathbf{q} \in \mathbb{R}^3$ such that $\mathbf{q} = [q_1, q_2, q_3]^\top$. A unit quaternion is bound by the restriction that $q_0^2 + \mathbf{q}^\top \mathbf{q} = 1$. The unit quaternion is capable of representing every orientation and angular velocity; however, it is not unique [93] since there are two quaternions for every orientation [12]. To combine two quaternions, they may be directly multiplied together following the rules of quaternion multiplication. Two quaternions, $\mathbf{P}, \mathbf{Q} \in \mathbb{H}$, can be combined through quaternion multiplication which is given by:

$$\mathbf{P} \odot \mathbf{Q} = \begin{bmatrix} p_0 \\ \mathbf{p} \end{bmatrix} \odot \begin{bmatrix} q_0 \\ \mathbf{q} \end{bmatrix} = \begin{bmatrix} p_0 q_0 - \mathbf{p}^\top \mathbf{q} \\ p_0 \mathbf{q} + q_0 \mathbf{p} + \mathbf{p} \times \mathbf{q} \end{bmatrix}. \tag{3.11}$$

Here, $\mathbf{P}, \mathbf{Q} \in \mathbb{H}$ are two arbitrary quaternions and $\odot$ is the quaternion multiplication symbol; quaternion multiplication is non-commutative.

Given $\mathbf{Q}$ from eq. (3.10), a rotation with the same magnitude but in the reverse direction is given by

$$\mathbf{Q}^{-1} = \begin{bmatrix} q_0 \\ -\mathbf{q} \end{bmatrix}, \tag{3.12}$$

so that

$$\mathbf{Q}^{-1} \odot \mathbf{Q} = \mathbf{Q} \odot \mathbf{Q}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^\top \coloneqq \mathbf{Q}_I, \tag{3.13}$$

where $\mathbf{Q}_I$ is defined as the identity element. The identity element expresses the same rotation as a rotation matrix evaluated as an identity matrix.

For an arbitrary vector, $\mathbf{v} \in \mathbb{R}^3$, where $\mathbf{v}_\mathcal{I} = R\mathbf{v}_\mathcal{B}$, an equivalent expression using quaternions is

$$\bar{\mathbf{v}}_\mathcal{I} = \mathbf{Q} \odot \bar{\mathbf{v}}_\mathcal{B} \odot \mathbf{Q}^{-1}, \tag{3.14}$$

where $\mathbf{Q}$ is the quaternion associated with R, $\bar{\mathbf{v}}_\mathcal{I} = \begin{bmatrix} 0 & \mathbf{v}_\mathcal{I}^\top \end{bmatrix}^\top$, and $\bar{\mathbf{v}}_\mathcal{B} = \begin{bmatrix} 0 & \mathbf{v}_\mathcal{B}^\top \end{bmatrix}^\top$. A quaternion can also be expressed as a rotation matrix through the Rodrigues map

$$\mathcal{R}(\mathbf{Q}) : \mathbb{H} \to \mathsf{SO}(3), \tag{3.15}$$

such that

$$\mathcal{R}(\mathbf{Q}) = \left(q_0^2 - \|\mathbf{q}\|^2\right) I_3 + 2\mathbf{q}\mathbf{q}^\top + 2q_0 \; sk(\mathbf{q}), \tag{3.16}$$

where $sk(\cdot)$ represents the skew symmetric matrix which is defined as

$$sk(\mathbf{x}) := \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & -x_1 & 0 \end{bmatrix}, \tag{3.17}$$

and is associated with the cross product, *i.e.* $\mathbf{x} \times \mathbf{y} = sk(\mathbf{x})\,\mathbf{y}$ for any two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$, where $\times$ is the cross-product operator. Evaluating eq. (3.16) and simplifying yields:

$$R(\mathbf{Q}) = \begin{bmatrix} 1-2\left(q_2^2+q_3^2\right) & 2\left(q_1q_2-q_0q_3\right) & 2\left(q_1q_3+q_0q_2\right) \\ 2\left(q_1q_2+q_0q_3\right) & 1-2\left(q_1^2+q_3^2\right) & 2\left(q_2q_3-q_0q_1\right) \\ 2\left(q_1q_3-q_0q_2\right) & 2\left(q_2q_3+q_0q_1\right) & 1-2\left(q_1^2+q_2^2\right) \end{bmatrix}, \tag{3.18}$$

which is a rotation matrix populated with the elements from a unit quaternion. Each quaternion element in eq. (3.18) is multiplied by another quaternion element. Due to this, the rotational representation given by a unit quaternion is ambiguous, *i.e.* $R(Q) = R(-Q)$. This means the unit quaternion has a 2:1 mapping of quaternions to any particular attitude, *i.e.* it is non-unique for any given attitude. Despite the non-uniqueness of the unit quaternion, it is still incredibly useful because it is a global method of attitude representation using the minimum number of parameters. In the following section, the different attitude representations are compared through the number of parameters required to express the attitude, their uniqueness, and whether or not they are global.

3.4 Attitude Representation Comparison

TABLE 3.1: Attitude representation comparison

Representation	No. of Parameters	Global	Unique
Direction Cosine Matrix	9	Yes	Yes
Euler Angles	3	No	Yes
Unit Quaternion	4	Yes	No

Shown in table 3.1 is a comparison of the above attitude representations. While the direction cosine matrix is a global and unique representation, it is also very computationally expensive and therefore slow due to the requirement of estimating nine unique parameters. The Euler angles provide an intuitive method of visualising attitude using only three parameters; however, it has the major disadvantage of a discontinuity in its pitch dynamics when $\theta = \pi/2$. The unit quaternion provides a continuous global representation but has two different mathematical representations for each physical orientation. Within this work, these representations are used in concert to exploit their strengths while compensating for each representation's respective weakness. Since multiple methods of attitude representation are used, conversion between them may

be necessary. The Euler angles may be converted to a unit quaternion by:

$$\begin{bmatrix} q_0 \\ \mathbf{q} \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) \end{bmatrix}, \tag{3.19}$$

and the quaternion may be converted to Euler angles by:

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \text{arctan2}\big(q_0 q_1 + q_2 q_3, 1 - 2(q_1^2 + q_2^2)\big) \\ \arcsin\big(q_0 q_2 - q_1 q_3\big) \\ \text{arctan2}\big(q_0 q_3 + q_1 q_2, 1 - 2(q_2^2 + q_3^2)\big) \end{bmatrix}, \tag{3.20}$$

where atan2(y, x) is the arctan operator that returns an unambiguous angle based on the Cartesian coordinates of x and y that preserves the quadrant of x and y.

Chapter 4

Attitude Estimation

This chapter describes the theory and implementation of the attitude estimation algorithm which was chosen for this work. The goal of attitude estimation is to find a set of kinematics to make an estimate of the attitude, or rotation matrix, $R \in \mathsf{SO}(3)$. The rotation matrix is described in more detail in chapter 3. The algorithm was written using Arduino C/C++ programming language and was programmed onto the ArduPilot Mega (APM) 2.5 device. The measurement devices installed within the APM 2.5 that were utilised for the attitude estimation are:

- An Invensense MPU-6000 Inertial Measurement Unit (IMU).
- A Honeywell HMC5883L magnetometer.

The APM 2.5, the IMU, and the magnetometer are described in more detail in chapter 6. The topics covered in this chapter are as follows:

1. Complementary Filter.
2. Conditioned Observer.
3. Discrete Implementation.

4.1 Complementary Filter

A complementary filter is a type of observer which fuses multiple independent noisy measurements having complementary characteristics; it is termed an observer because the values which are being estimated cannot be directly measured. The measurements which are being fused are:

1. The linear acceleration exerted on the quadrotor body by gravity and the movement of the quadrotor.
2. The body-referenced magnetic field at the quadrotor.
3. The angular velocity of the quadrotor around the body-referenced axes.

The kinematics of the rotation matrix satisfy

$$\dot{R} = R\, sk(\mathbf{\Omega})\,, \tag{4.1}$$

where $\mathbf{\Omega} \in \mathbb{R}^3$ is the true, body-referenced angular velocity. Defining the measured angular velocity as

$$\mathbf{\Omega}_m := \mathbf{\Omega} + \mathbf{b}, \quad \mathbf{\Omega}_m \in \mathbb{R}^3, \tag{4.2}$$

where $\mathbf{b} \in \mathbb{R}^3$ is an unknown, slowly time-varying bias. The complementary filter requires two non-collinear vectors to operate. Defining the following unit vectors:

$$\mathbf{u}_{\mathcal{I}} \coloneqq \hat{\mathbf{e}}_z, \qquad \mathbf{v}_{\mathcal{I}} \coloneqq \frac{\mathbf{m}_{\mathcal{I}}}{\|\mathbf{m}_{\mathcal{I}}\|}, \tag{4.3a}$$

$$\mathbf{u}_{\mathcal{B}} \coloneqq -\frac{\mathbf{a}_{\mathcal{B}}}{g}, \qquad \mathbf{v}_{\mathcal{B}} \coloneqq \frac{\mathbf{m}_{\mathcal{B}}}{\|\mathbf{m}_{\mathcal{B}}\|}, \tag{4.3b}$$

where $\hat{\mathbf{e}}_z = [0, 0, 1]^\mathsf{T}$ denotes the unit vector of the z-axis of $\mathcal{I}$ which points toward the center of the Earth, $\mathbf{a}_B$ are measurements from the accelerometer on the quadrotor expressed in $\mathcal{B}$, $\mathbf{m}_{\mathcal{I}}$ is the local magnetic field expressed in $\mathcal{I}$, $\mathbf{m}_B$ are measurements from the magnetometer on the quadrotor expressed in $\mathcal{B}$, and g is the gravitational constant. The measurements from the accelerometer, $\mathbf{a}_{\mathcal{B}}$, and magnetometer, $\mathbf{m}_{\mathcal{B}}$ are assumed to be non-collinear. Additionally, the accelerometer measurements are assumed to capture only the acceleration force exerted on the quadrotor by gravity. The observer from [13] is given by:

$$\begin{cases} \dot{\hat{R}} = \hat{R}\, sk\Big(\boldsymbol{\Omega}_m - \hat{\mathbf{b}} + \boldsymbol{\sigma}_{\mathbf{R}}\Big) \\ \dot{\hat{\mathbf{b}}} = -k_I \boldsymbol{\sigma}_{\mathbf{R}} \\ \boldsymbol{\sigma}_{\mathbf{R}} = k_1\, sk(\mathbf{u}_{\mathcal{B}})\, \hat{\mathbf{u}}_{\mathcal{B}} + k_2\, sk(\mathbf{v}_{\mathcal{B}})\, \hat{\mathbf{v}}_{\mathcal{B}}, \end{cases} \tag{4.4}$$

where

$$\hat{\mathbf{u}}_{\mathcal{B}} \coloneqq \hat{R}\mathbf{u}_{\mathcal{I}}, \quad \hat{\mathbf{v}}_{\mathcal{B}} \coloneqq \hat{R}\mathbf{v}_{\mathcal{I}}, \tag{4.5}$$

$\hat{R}$ is the estimate of the rotation matrix R, and $\hat{\mathbf{b}}$ is the estimate of the bias. The innovation term is denoted by $\boldsymbol{\sigma}_{\mathbf{R}}$ and k_I, k_1, and k_2 are positive real gains. The filter given by eq. (4.4) was proven in [13] to ensure almost global asymptotic stability and local exponential stability of the equilibrium $(\tilde{R}, \tilde{\mathbf{b}}) = (I_3, 0)$ where

$$\tilde{R} \coloneqq \hat{R}^\top R, \quad \tilde{\mathbf{b}} \coloneqq \mathbf{b} - \hat{\mathbf{b}}. \tag{4.6}$$

It can be easily seen that $\boldsymbol{\sigma}_{\mathbf{R}}$ will vanish when the estimated attitude matches the true attitude. However, usage of the integral action in the bias estimation may lead to a drift over time in the presence of measurement noise [18], [20]. This problem was solved through the introduction of a saturation function in [18] and [20]. Another improvement proposed in [20] was to modify the unit vectors defined in eq. (4.3) to locally decouple the roll and pitch estimations from the yaw estimation to isolate the magnetometer perturbations from affecting the roll and pitch estimates. These developments will be presented in the following section.

4.2 Conditioned Observer

The complementary filter given in [20] develops upon the work of [13], [18] and is termed "conditioned observer." The observer is "conditioned" by altering the innovation term and the unit vectors defined in eq. (4.3). The purpose of these alterations is to ensure local decoupling of the roll and pitch estimations from the magnetometer measurements and from the yaw estimation.

The unit vectors defined in eq. (4.3) are redefined as:

$$\mathbf{u}_{\mathcal{I}} \coloneqq \mathbf{e}_3, \qquad \mathbf{v}_{\mathcal{I}} \coloneqq \frac{\pi_{\mathbf{u}_{\mathcal{I}}}\mathbf{m}_{\mathcal{I}}}{\|\pi_{\mathbf{u}_{\mathcal{I}}}\mathbf{m}_{\mathcal{I}}\|}, \tag{4.7}$$

$$\mathbf{u}_{\mathcal{B}} \coloneqq -\frac{\mathbf{a}_{\mathcal{B}}}{g}, \qquad \mathbf{v}_{\mathcal{B}} \coloneqq \frac{\pi_{\mathbf{u}_{\mathcal{B}}}\mathbf{m}_{\mathcal{B}}}{\|\pi_{\mathbf{u}_{\mathcal{I}}}\mathbf{m}_{\mathcal{I}}\|}, \tag{4.8}$$

where $\pi_{\mathbf{x}} \coloneqq \|\mathbf{x}\| I_3 - \mathbf{x}\mathbf{x}^\top, \forall \mathbf{x} \in \mathbb{R}^3$ denotes the projection onto the plane orthogonal to $\mathbf{x}$. The conditioned observer is given by:

$$\begin{cases} \dot{\hat{R}} = \hat{R}\, sk\Big(\boldsymbol{\Omega}_m - \hat{\mathbf{b}} + \boldsymbol{\sigma}_{\mathbf{R}}\Big) \\ \dot{\hat{\mathbf{b}}} = -k_b\hat{\mathbf{b}} + k_b \text{sat}_{\Delta}(\hat{\mathbf{b}}) + \boldsymbol{\sigma}_{\mathbf{b}}, \quad \|\hat{\mathbf{b}}(0)\| < \Delta \\ \boldsymbol{\sigma}_{\mathbf{R}} = k_1\, sk(\mathbf{u}_{\mathcal{B}})\, \hat{\mathbf{u}}_{\mathcal{B}} + k_2 \hat{\mathbf{u}}_{\mathcal{B}}\hat{\mathbf{u}}_{\mathcal{B}}^\top\, sk(\mathbf{v}_{\mathcal{B}})\, \hat{\mathbf{v}}_{\mathcal{B}} \\ \boldsymbol{\sigma}_{\mathbf{b}} = -k_3\, sk(\mathbf{u}_{\mathcal{B}})\, \hat{\mathbf{u}}_{\mathcal{B}} - k_4\, sk(\mathbf{v}_{\mathcal{B}})\, \hat{\mathbf{v}}_{\mathcal{B}}, \end{cases} \tag{4.9}$$

where k_b, k_1, k_2, k_3, k_4, and Δ are positive real numbers and $\text{sat}_{\Delta}(\mathbf{x}) \coloneqq \mathbf{x}\min(1, \Delta/\|\mathbf{x}\|)$. In this observer, if the measured accelerometer and magnetometer projection align with the estimated accelerometer and magnetometer projection then the innovation terms $\sigma_{\mathbf{R}}$ and $\sigma_{\mathbf{b}}$ will vanish; however, in practice this will never happen due to measurement noise.

It is computationally expensive to compute the proposed observer given by eq. (4.9). Chapter 3 provides more details on attitude representation and how the different representations relate to one another. To simplify the calculation of the attitude, the unit quaternion can be utilised to express the attitude using four unique variables rather than use the rotation matrix with requires nine unique variables to be estimated. The conditioned observer expressed using the quaternion becomes:

$$\begin{cases} \dot{\hat{\mathbf{q}}} = \frac{1}{2} A\Big(\hat{\boldsymbol{\Omega}}\Big)\, \hat{\mathbf{q}} \\ \dot{\hat{\mathbf{b}}} = -k_b\hat{\mathbf{b}} + k_b \text{sat}_{\Delta}\left(\hat{\mathbf{b}}\right) + \boldsymbol{\sigma}_{\mathbf{b}}, \quad \|\hat{\mathbf{b}}\,(0)\| < \Delta \\ \hat{\boldsymbol{\Omega}} \coloneqq \boldsymbol{\Omega}_m - \hat{\mathbf{b}} + \boldsymbol{\sigma}_{\mathbf{R}}, \quad A\Big(\hat{\boldsymbol{\Omega}}\Big) \coloneqq \begin{bmatrix} 0 & -\hat{\boldsymbol{\Omega}}^{\mathsf{T}} \\ \hat{\boldsymbol{\Omega}} & -sk\Big(\hat{\boldsymbol{\Omega}}\Big) \end{bmatrix}. \end{cases} \tag{4.10}$$

4.3 Discrete Implementation

Implementation of the conditioned observer on the digital system of the APM 2.5, is performed through discretisation. To accomplish this, $\hat{\boldsymbol{\Omega}}$ and $\boldsymbol{\sigma}_{\mathbf{b}}$ are assumed to remain constant over every time period $S_k \coloneqq [(k-1)\,T, kT], \forall k \in \mathbb{N}$ given a small enough sample time T. Under this assumption, the continuous values of $\hat{\boldsymbol{\Omega}}$ and $\boldsymbol{\sigma}_{\mathbf{b}}$ over the time period S_k are denoted as $\hat{\boldsymbol{\Omega}}_k$ and $\boldsymbol{\sigma}_{\mathbf{b},k}$, respectively. The discretised attitude dynamics equation from eq. (4.10) can be expressed as:

$$\hat{\mathbf{q}}_{k+1} = exp\left(\frac{T}{2} A\Big(\hat{\boldsymbol{\Omega}}_k\Big)\right) \hat{\mathbf{q}}_k. \tag{4.11}$$

Expanding the exponential from eq. (4.11) using the Taylor expansion yields:

$$\begin{aligned} exp\left(\frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right)\right) =& \sum_{i=0}^{\infty}\frac{1}{i!}\left(\frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right)\right)^i \\ =& I_4 + \frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right) + \frac{1}{2}\left(\frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right)\right)^2 + \frac{1}{6}\left(\frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right)\right)^3 + \cdots \end{aligned} \tag{4.12}$$

Equation (4.12) can then be simplified using the fact that

$$A\left(\hat{\boldsymbol{\Omega}}_k\right)^2 = -\|\boldsymbol{\Omega}_k\|^2 I_4, \tag{4.13}$$

where I_4 is the square identity matrix of size four. This allows eq. (4.12) to be simplified as

$$exp\left(\frac{T}{2}A\left(\hat{\boldsymbol{\Omega}}_k\right)\right) = \cos\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)I_4 + \frac{T}{2}\text{sinc}\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)A\left(\hat{\boldsymbol{\Omega}}_k\right), \tag{4.14}$$

where $\text{sinc}(x) = \sin(x)/x$. Substituting eq. (4.14) into eq. (4.11) then yields

$$\hat{\mathbf{q}}_{k+1} = \left(\cos\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)I_4 + \frac{T}{2}sinc\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)A\left(\hat{\boldsymbol{\Omega}}_k\right)\right)\hat{\mathbf{q}}_k. \tag{4.15}$$

From eq. (4.9), the gyro-bias is assumed to be bounded in norm by Δ, *i.e.*, $\|\mathbf{b}\| \leq \Delta$. As stated above, the bias innovation term, $\boldsymbol{\sigma}_\mathbf{b}$, was assumed to be constant over every time period S_k. The bias estimation discretisation was obtained using the Euler approach as follows:

$$\frac{\hat{\mathbf{b}}_{k+1} - \hat{\mathbf{b}}_k}{(k+1)T - kT} = -k_b\hat{\mathbf{b}}_k + k_b\text{sat}_\Delta\left(\hat{\mathbf{b}}_k\right) + \boldsymbol{\sigma}_{\mathbf{b},k}, \tag{4.16}$$

which was rearranged to yield:

$$\hat{\mathbf{b}}_{k+1} = T\left(-k_b\hat{\mathbf{b}}_k + k_b\,\text{sat}_\Delta\left(\hat{\mathbf{b}}_k\right) + \boldsymbol{\sigma}_{\mathbf{b},k}\right) + \hat{\mathbf{b}}_k. \tag{4.17}$$

Expressing eq. (4.15) and eq. (4.17) together yields the following discrete observer:

$$\begin{cases} \hat{\mathbf{q}}_{k+1} = \left(\cos\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)I_4 + \frac{T}{2}sinc\left(\frac{T\|\hat{\boldsymbol{\Omega}}_k\|}{2}\right)\mathbf{A}\left(\hat{\boldsymbol{\Omega}}_k\right)\right)\hat{\mathbf{q}}_k \\ \hat{\mathbf{b}}_{k+1} = T\left(-k_b\hat{\mathbf{b}}_k + k_b\,\text{sat}_\Delta\left(\hat{\mathbf{b}}_k\right) + \sigma_{\mathbf{b},k}\right) + \hat{\mathbf{b}}_k. \end{cases} \tag{4.18}$$

The observer given by eq. (4.18) is a discretized version of the observer given by eq. (4.10). In practice, the trigonometric functions of eq. (4.18) are taken as a second order approximation for computational efficiency. This was the observer that was implemented for this work on the APM 2.5.

Chapter 5

Mathematical Model of a Quadrotor

This chapter focuses on the mathematical model of a quadrotor. In the first section, the mathematical equations used to represent the model of a quadrotor are presented. The more widely used model which uses the rotation matrix is presented first and is followed with an equivalent model which uses the unit quaternion. Once that's done, the control inputs will be discussed and will show how the actuators on the quadrotor generate the control inputs.

5.1 Model Definition

The mathematical model which will be established considers a quadrotor UAV. The mathematical model describes the dynamic equations and is also referred to as the dynamical model. The model includes the rotation matrix and refers to coordinate frames which are related by the rotation matrix. The rotation matrix and coordinate frames are described in more detail in section 3.1 The dynamical model described in [94] is given as

$$\dot{\mathbf{p}} = \mathbf{v} \tag{5.1a}$$

$$m\dot{\mathbf{v}} = mg\hat{\mathbf{e}}_z + \mathbf{F} \tag{5.1b}$$

$$\dot{R} = R\, sk(\boldsymbol{\Omega}) \tag{5.1c}$$

$$I_q\dot{\boldsymbol{\Omega}} = sk(\boldsymbol{\Omega})\, I_q\boldsymbol{\Omega} + \boldsymbol{\tau}_q \tag{5.1d}$$

where $\mathbf{p} = [x, y, z]^\top$ denotes the position of the center of mass of the quadrotor expressed in $\mathcal{I}$, m denotes the mass of the quadrotor in kilograms, $\mathbf{v} = [v_x, v_y, v_z]^\top$ denotes the linear velocity of the center of mass of the quadrotor expressed in $\mathcal{I}$, $\hat{\mathbf{e}}_z = [0, 0, 1]^\top$ denotes the unit vector of the z-axis of $\mathcal{I}$, $\mathbf{F}$ denotes the force applied to the quadrotor expressed in $\mathcal{I}$, $\boldsymbol{\Omega} = [\Omega_1, \Omega_2, \Omega_3]^\top$ denotes the angular velocity of the quadrotor expressed in $\mathcal{B}$ whose origin is the center of mass of the quadrotor, $I_q \in \{\mathbb{R}^{3x3} \,|\, I_q = I_q^\mathsf{T} > 0\}$ denotes the symmetric, positive-definite constant inertia matrix of the quadrotor expressed in $\mathcal{B}$, and $\boldsymbol{\tau}_q = [\tau_1, \tau_2, \tau_3]^\top$ denotes the control torque vector expressed in $\mathcal{B}$. These symbols are summarized in table 5.1.

The dynamical model described in eq. (5.1) uses the rotation matrix which contains nine parameters. For computational efficiency, the quadrotor's mathematical model can be expressed using the unit quaternion which uses only four parameters. The unit quaternion is discussed in more detail in section 3.3. Due to the orientation of the rotors mounted on the quadrotor, the force which the rotors exert is in the direction of $\hat{\mathbf{e}}_3$, the unit vector which corresponds to the downward axis of $\mathcal{B}$. The force, $\mathbf{F}$, that the quadrotor exerts is equal to the thrust output by the

TABLE 5.1: Summary of symbols used in the mathematical model.

Symbol	Definition
$\mathcal{I} = \{\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z\}$	Inertial referenced coordinate frame oriented in NED.
$\mathcal{B} = \{\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3\}$	Body referenced coordinate frame oriented in NED.
$\mathbf{p} = [x, y, z]^\top$	Position of the origin of $\mathcal{B}$ expressed in $\mathcal{I}$.
$\mathbf{v} = [v_x, v_y, v_z]^\top$	Velocity of the origin of $\mathcal{B}$ expressed in $\mathcal{I}$.
m	Mass of the quadrotor.
g	Gravity constant.
$\mathbf{F}$	Force exerted by the quadrotor.
$R^\top$	Rotation of $\mathcal{B}$ with respect to $\mathcal{I}$.
$\boldsymbol{\Omega} = [\Omega_1, \Omega_2, \Omega_3]^\top$	Angular velocity of the quadrotor around $\mathcal{B}$.
I_q	3x3 inertial matrix of the quadrotor expressed in $\top$.
$\boldsymbol{\tau}_q = [\tau_1, \tau_2, \tau_3]^\top$	Control torques of the quadrotor.

propellers and can be expressed as

$$\mathbf{F} \coloneqq -\mathcal{T} R^\top \hat{\mathbf{e}}_z, \tag{5.2}$$

where $\mathcal{T}$ is the total thrust exerted by the rotors and $\hat{\mathbf{e}}_z = [0, 0, 1]^\top$ denotes the unit vector of the z-axis of $\mathcal{I}$. The thrust is negative because the coordinate frames are NED and the rotors are accelerating the quadrotor in an upwards direction. The thrust is defined as

$$\mathcal{T} \coloneqq \sum_{i=1}^{4} f_i = b \sum_{i=1}^{4} \bar{\omega}_i^2, \tag{5.3}$$

where $f_i = b\,\bar{\omega}_i^2$ is the thrust exerted by a single rotor and $i = \{f, l, b, r\}$ corresponds respectively to the front, left, back, and right rotors of the quadrotor. The constant b is a positive proportionality constant depending on the density of air and $\bar{\omega}_i$ is the angular velocity of a single rotor. The control torques, $\boldsymbol{\tau}_q$, can be expressed as functions of the angular velocities of the four rotors:

$$\tau_1 = db\left(\bar{\omega}_l^2 - \bar{\omega}_r^2\right) \tag{5.4a}$$

$$\tau_2 = db\left(\bar{\omega}_f^2 - \bar{\omega}_b^2\right) \tag{5.4b}$$

$$\tau_3 = \kappa\left(\bar{\omega}_f^2 + \bar{\omega}_b^2 - \bar{\omega}_l^2 - \bar{\omega}_r^2\right), \tag{5.4c}$$

where $d > 0$ is the distance from $\hat{\mathbf{e}}_3$ to a rotor's axis of rotation that is orthogonal to $\hat{\mathbf{e}}_3$ and $\kappa > 0$ is a constant which depends on the shape of the propellers. It was assumed that d is the same for each rotor. The thrust and control torques were combined into a control vector which is defined as

$$\mathbb{T} \coloneqq \begin{bmatrix} \mathcal{T} \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = M\bar{\boldsymbol{\omega}} \tag{5.5}$$

where

$$M \coloneqq \begin{bmatrix} b & b & b & b \\ 0 & db & 0 & -db \\ db & 0 & -db & 0 \\ \kappa & -\kappa & \kappa & -\kappa \end{bmatrix}, \tag{5.6}$$

and

$$\bar{\boldsymbol{\omega}} \coloneqq \left[\bar{\omega}_f^2, \bar{\omega}_l^2, \bar{\omega}_b^2, \bar{\omega}_r^2\right]^{\mathsf{T}}. \tag{5.7}$$

The matrix M is always invertible for $d, b, \kappa > 0$. The control vector, $\mathbb{T}$, given in eq. (5.5) can then be used to specify the required angular velocities of the rotors as

$$\bar{\boldsymbol{\omega}} = M^{-1}\mathbb{T} = \begin{bmatrix} (4b)^{-1} & 0 & (2db)^{-1} & (4\kappa)^{-1} \\ (4b)^{-1} & (2db)^{-1} & 0 & -(4\kappa)^{-1} \\ (4b)^{-1} & 0 & -(2db)^{-1} & (4\kappa)^{-1} \\ (4b)^{-1} & -(2db)^{-1} & 0 & -(4\kappa)^{-1} \end{bmatrix} \mathbb{T}. \tag{5.8}$$

In certain cases, $M^{-1}\mathbb{T}$ may result in some elements of $\bar{\boldsymbol{\omega}}$ being negative suggesting a reversal of rotation of the rotor which is not practical. This is unlikely to occur if the total thrust, $\mathcal{T}$, is sufficiently large with respect to the control torques [32].

The mathematical model used within this work for designing the control system can now be expressed. Replacing $\mathbf{F}$ with the expression given by eq. (5.2), the rotation matrix in eq. (5.1c) with the unit quaternion, and including eq. (5.8) as part of the mathematical model yields:

$$\dot{\mathbf{p}} = \mathbf{v} \tag{5.9a}$$

$$m\dot{\mathbf{v}} = mg\hat{\mathbf{e}}_z - \mathcal{T}R^{\mathsf{T}}\hat{\mathbf{e}}_3 \tag{5.9b}$$

$$\dot{Q} = \frac{1}{2}Q \odot \begin{bmatrix} 0 \\ \Omega \end{bmatrix} \tag{5.9c}$$

$$I_q\dot{\boldsymbol{\Omega}} = sk(\boldsymbol{\Omega})\, I_q\boldsymbol{\Omega} + \boldsymbol{\tau}_q \tag{5.9d}$$

$$\bar{\boldsymbol{\omega}} = M^{-1}\mathbb{T}, \tag{5.9e}$$

where $\odot$ is the quaternion multiplier described in section 3.3.

Chapter 6

Experimental Platform: Hardware

This chapter details the hardware of the experimental platform which was used for this experiment. Figure 6.1 shows a block diagram of the experimental platform. The experimental platform consists of a quadrotor UAV with two downward facing cameras to enable stereo vision and a desktop computer to perform the required image processing. The cameras and IMU are used to estimate the position and linear velocity of the quadrotor. Section 6.1 details the airframe of the quadrotor UAV and hardware which were needed for flight; this includes the body of the quadrotor, the motors and ESCs, the control unit which was mounted on the quadrotor, and the hardware which was used for control communication. Next, section 6.2 details the cameras which were used and the hardware associated with transmitting the camera image to the base station. Section 6.3 shows the desktop computer which was used for image processing. The final section of this chapter details the methods which were used to calibrate the sensors used for this experiment. The sensors which were used are the IMU, which contains an accelerometer, a gyroscope, and a magnetometer, and the cameras. Chapter 7 also presents the results of the calibration process.

6.1 Airframe and Flight Hardware

A large number of open source platforms exist for quadrotor UAVs; a survey of the most popular platforms is given in [42]. A block diagram of the experimental setup is shown in fig. 6.1. Note that the cameras are on the quadrotor but do not have a connection to the other hardware. This is because the cameras act independently to the rest of the hardware on the quadrotor, capturing

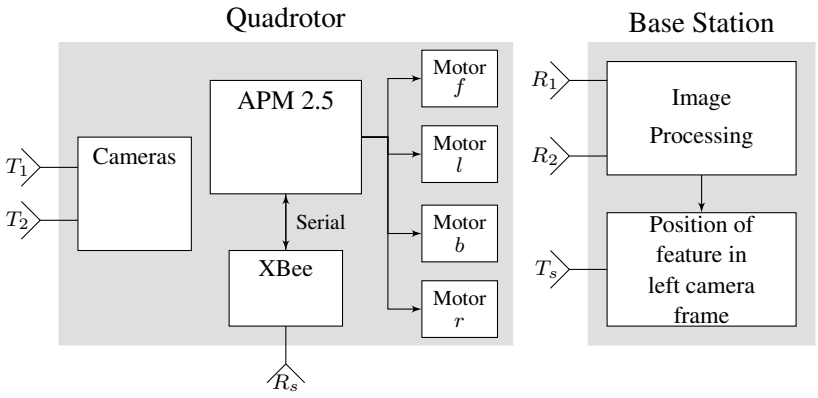


FIGURE 6.1: Block diagram of the experimental platform.

images and transmitting them to the base station computer when ready. The position extraction performed by the base station computer calculates the position of the detected feature in the camera frame of the left camera and a rotation must still be applied to this position vector to bring it into the inertial frame.

The airframe for this experiment was the 3DRobotics Quad-C [95] quadrotor platform, previously available from 3DRobotics, shown in fig. 6.2. The setup used the legacy APM Copter [96] hardware, the APM 2.5. The APM 2.5 was programmed using the Arduino C/C++ language and is mounted on the quadrotor body.

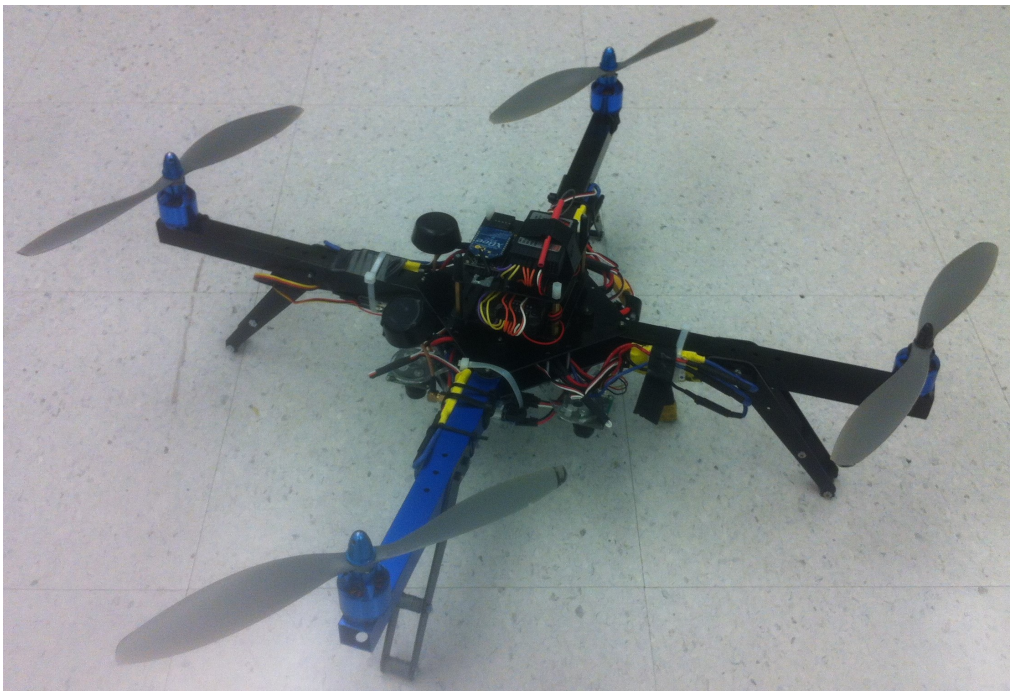
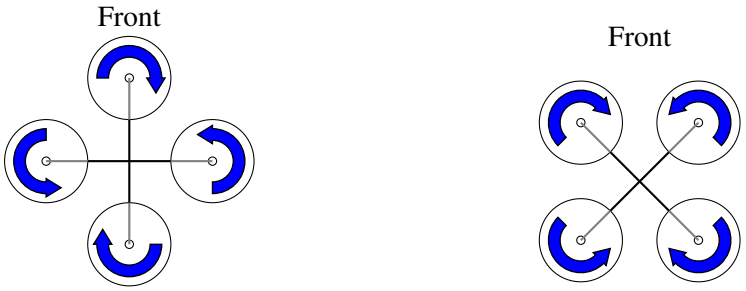


FIGURE 6.2: Quadrotor experimental platform.

6.1.1 Quadrotor Frame and Body

The quadrotor consisted of a cross frame of square aluminium channels for the arms, bolted to each arm are two composite plates which act as the landing legs. The central body consisted of two composite plates bolted to the top and bottom of the aluminium arms, as shown in fig. 6.2. Mounted above the central body were smaller composite plates to which the electronics are mounted. Shown in fig. 6.3 are the two configurations for a quadrotor which are named by the shape of the arms when viewed from above. This work uses the plus (+) configuration for the four motors, where a single motor was mounted on the front, another to the left, one behind, and the last to the right, all respective to the central body. The motors and ESCs are discussed further in the following subsection.



(A) Plus or "+" configuration. (B) Cross or "X" configuration.

FIGURE 6.3: Quadrotor motor configurations.

6.1.2 Motors, Electronic Speed Controllers (ESCs), and Power Supply

The four motors, where one was mounted at the end of each arm, are A2830-12, 850Kv motors from 3DRobotics [97]. These motors are brushless Direct Current (DC) outrunner motors; outrunner motors produce motion by spinning their outer shell around the stationary central windings. Specifications for the motors are shown in table 6.1, where RPM denotes Rotations Per Minute. Each motor spins a slow flyer propeller which is ten inches in diameter and has a pitch

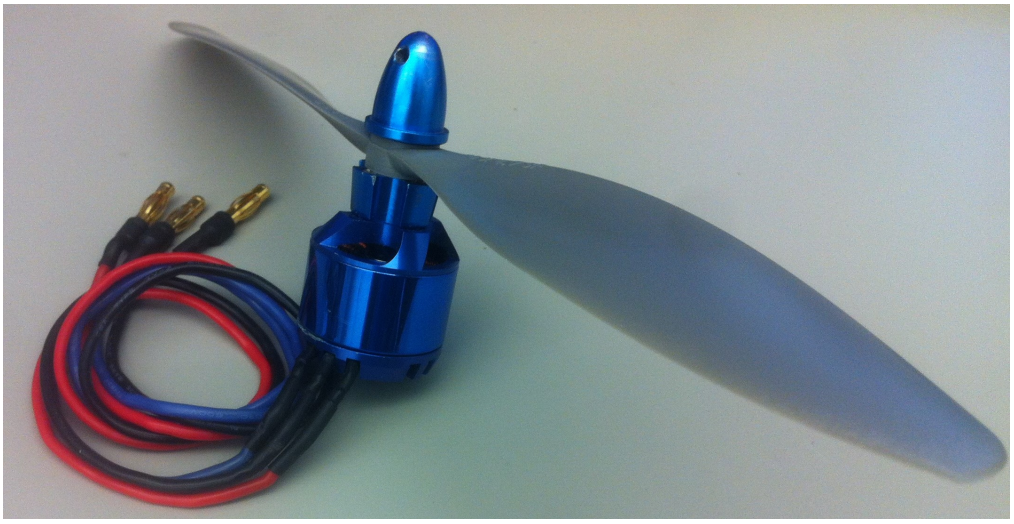


FIGURE 6.4: A2830-12 outrunner motor with $4.7in/rev$ propeller attached.

of $4.7in/rev$. Shown in fig. 6.4 is one of the motors used with the propeller mounted on the motor. The propeller pitch is defined as follows:

$$pitch := \frac{v_{prop}}{n}, \tag{6.1}$$

where v_{prop} is the forward speed of the propeller in in/s and n is the propeller angular velocity in rev/s. The pitch ratio is the forward distance the propeller would move in one revolution if it were unimpeded by negative forces [98], [99]. The quadrotor had four ESCs installed, one for each of its motors, and were mounted to the sides of the its arms. An example of one of the ESCs is shown in fig. 6.5. These ESCs were purchased from 3DRobotics as part of the Quad-C platform [95]. The ESCs are capable of driving motors at a continuous current of $20A$ up to a

TABLE 6.1: Motor characteristics

Model	Input Voltage	Angular Speed	Thrust	Mass	Max Power
A2830-12	7.4-15V	850Kv (RPM/V)	880g	52g	200W

FIGURE 6.5: 20Amp ESC used for this work.

speed of $35000RPM$ [100]; however, the propellers are limited by their manufacturing design to revolution speeds up to $6500RPM$ [101]. The quadrotor was powered by a 3-cell, $4000mAh$ lithium-polymer (LiPo) battery which had a full charge at $12.6V$ and was fully discharged at $9.9V$. A power distribution board was used to provide power to the ESCs. The voltage of the battery was regulated to $5V$ in order to power the electronics on-board the quadrotor, including the microcontroller and cameras. The microcontroller then provided power to the remote control (RC) receiver and XBee serial communication device. The control unit and microcontrollers are discussed in the following subsection.

6.1.3 Control Unit

The control unit used was the APM 2.5, a custom Arduino board from 3DRobotics. The control system consisted of two microcontrollers and two measurement devices which are discussed as follows.

Microcontrollers

The APM 2.5 uses an ATmega2560 microcontroller from Atmel as its main microcontroller. The ATmega2560 is a low-power complementary metal-oxide semiconductor (CMOS) eight-bit microcontroller based on the enhanced reduced instruction set computer (RISC) architecture designed by AVR. The ATmega2560 has a clock speed of $16MHz$ and communicated with peripheral devices through two methods:

1. A hardware serial peripheral interface (SPI).
2. Inter-integrated circuit (I^2C) ports.

The APM 2.5 also includes a secondary microcontroller: an ATmega32U2 from Amtel. The ATmega32U2 is responsible for the programming of the ATmega2560 and some low level operations, such as Pulse-Position Modulation (PPM) control of the motor speeds and processing received Remote Control (RC) signals. The main control loop operates on the ATmega2560 was programmed to operate update every $1/100s$ to ensure that new measurements would be available at the beginning of the control loop.

Measurement Devices

There are two measurement units installed on the APM 2.5: an Invensense MPU-6000 [102] inertial measurement unit (IMU) and a Honeywell HMC5883L [103] magnetometer. The MPU-6000 contains a three-axis accelerometer and a three-axis gyroscope, which measure linear acceleration and angular velocity respectively. The MPU-6000 includes its own configurable Digital Low-Pass Filter (DLPF). Possible configurations for the DLPF are shown in table 6.2, for more information see [104]. For this work the DLPF was configured to a cut-off frequency of $20Hz$

TABLE 6.2: DLPF configurations for the MPU-6000.

Accelerometer ($F_s = 1kHz$)		Gyroscope		
Bandwidth (Hz)	Delay (ms)	Bandwidth (Hz)	Delay (ms)	F_s ($1kHz$)
260	0.0	256	0.98	8
184	2.0	188	1.9	1
94	3.0	98	2.8	1
44	4.9	42	4.8	1
21	8.5	20	8.3	1
10	13.8	10	13.4	1
5	19.0	5	18.6	1

for the gyroscope to ensure acceptable noise filtering; this configuration ensured that new data measurements would be available every time the control loop ran.

The HMC5883L magnetometer is designed for low-field magnetic sensing. It contains a three-axis magnetoresistive sensors with twelve-bit analogue to digital conversion (ADC) which enables $1 - 2^\circ$ of accuracy. The HMC5883L is configurable to provide a continuous output rate of $0.75 - 75Hz$ averaged over one, two, four, or eight samples. This experiment was performed with an output rate of $75Hz$ due to the agile performance of the quadrotor, averaged over the maximum quantity of eight samples. These measurement devices were used in conjunction with a hand-held remote controller or the cameras to control the position and orientation of the quadrotor. The remote controller is discussed below.

6.1.4 Remote Control

The remote control (RC) radio transmitter and receiver pair used in this work were the FlySky FS-TH9X, a hand-held, eight-channel transmitter, and a Flysky FS-R8B receiver [105]. These devices operate on a base frequency of $2.4GHz$ and data is sent between the transmitter and receiver using PPM signals. For this work, six channels are actively used. These channels are used to control the attitude (roll, pitch, and yaw), adjust the throttle, enable or disable the motors, and switch between manual and automatic position control. The PPM values for the attitude and thrust vary from approximately 1000 to 2000 with a midpoint at approximately 1500. The PPM values for the motor enable/disable and operation mode were set to toggle switches which set them to the minimum value of 1000 or the maximum value of 2000. The controls for the attitude are automatically returned to the midpoint when not in use by springs internal to the transmitter. This is done to allow positive or negative adjustment of the quadrotor's attitude where the midpoint is treated as 0°. The thrust control does not have a spring because while adjusting the thrust it was often desirable to set the thrust to a constant value to hover the quadrotor.

The transmitted attitude controls are converted to a desired quaternion through [32]

$$Q_d = \begin{bmatrix} \cos(\phi/2)\cos(\theta/2)\cos(\psi/2) + \sin(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \sin(\phi/2)\cos(\theta/2)\cos(\psi/2) - \cos(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\sin(\theta/2)\cos(\psi/2) + \sin(\phi/2)\cos(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\cos(\theta/2)\sin(\psi/2) - \sin(\phi/2)\sin(\theta/2)\cos(\psi/2) \end{bmatrix}, \tag{6.2}$$

where ϕ, θ, and ψ are the Euler angles for the roll, pitch, and yaw, respectively, of the quadrotor. This Q_d is used by the control law to generate the required control torques to determine the rotor speeds to control how the quadrotor moves. The yaw control differs from the roll and pitch by considering the yaw control as a desired angular velocity rather than a desired angle. This treatment of yaw by the control system ensures that releasing the manual yaw control stick will not be interpreted as a command to return to a yaw angle of $0°$. This was done because it was sometimes desirable to rotate the quadrotor in a complete circle rather than the minor adjustments which were needed for the roll and pitch.

6.1.5 Serial Communication

Telemetry data was sent and received on-board the quadrotor using a XBee Series 1 RF Module [106] from Digi International, where RF represents Radio Frequency. This module communicates with the APM 2.5 using the SPI. This data was sent and received at the base station computer by a paired XStick USB Adapter [107] shown in fig. 6.6, where USB represents Universal Serial Bus. This module was used to receive position data from the base station, position data which was extracted from stereo vision images. The module was also utilized to send other data to the base station, such as attitude calculations, accelerometer measurements, and data which was utilized for code or hardware troubleshooting.

6.2 Cameras

The cameras which were used in this work were mounted on the bottom of the quadrotor in horizontal stereo configuration as shown in fig. 6.7. They are mounted like this to mimic the stereo vision of human eyes which allows for intuitive analysis of the images. The data captured by the cameras is transmitted as an analogue signal by a wireless transmitter to receivers mounted at the base station computer. The wireless transmitters and receivers are discussed in the following subsections.

6.2.1 Image Capturing

The device which was used to capture images from the environment is a CMOS camera module made by SparkFun Electronics [108]. The camera lenses were capable of capturing a field of view (FoV) of $130°$. The CMOS detector inside the camera is a PC1089K [109]; it contains an array of pixels with dimensions of 728×488. The detector is configured to output a composite analogue signal which is capable of 30 frames per second (fps).

6.2.2 Wireless Transmitter/Receiver

The transmitters mounted on the quadrotor were responsible for sending visual data to the base station computer. The transmitters used were two model TS-351 wireless transmitters from

FIGURE 6.6: Base station computer with XStick (circled in red) and video receivers (circled in yellow).

BosCam [110]. These transmitters operated at $5.8GHz$ and could be configured to operate on one of sixteen possible channels. The power the transmitters required for operation was $200mW$. Cloverleaf-style antennas were used on the transmitters and receivers. The receivers (shown in fig. 6.6) are connected to the base station computer via a composite-to-USB video converter from StarTech [111]. The base station computer is discussed in the following section.

6.3 Base Station Computer

The base station computer was a Lenovo IdeaCentre K430. It contained an i7-3770 Central Processing Unit (CPU), $12GB$ of Random Access Memory (RAM), and a NVidia GeForce GT640 Graphics Processing Unit (GPU). The XStick, discussed in section 6.1.5, was insterted into one of the front-facing USB ports to ensure line of sight transmission between the base station and the quadrotor. The software used on the base station was written in C++ language. The software libraries used were the Open Computer Vision (OpenCV) libraries, version 4.0.0, and the NVidia GPU Computing Toolkit, version 10.1.

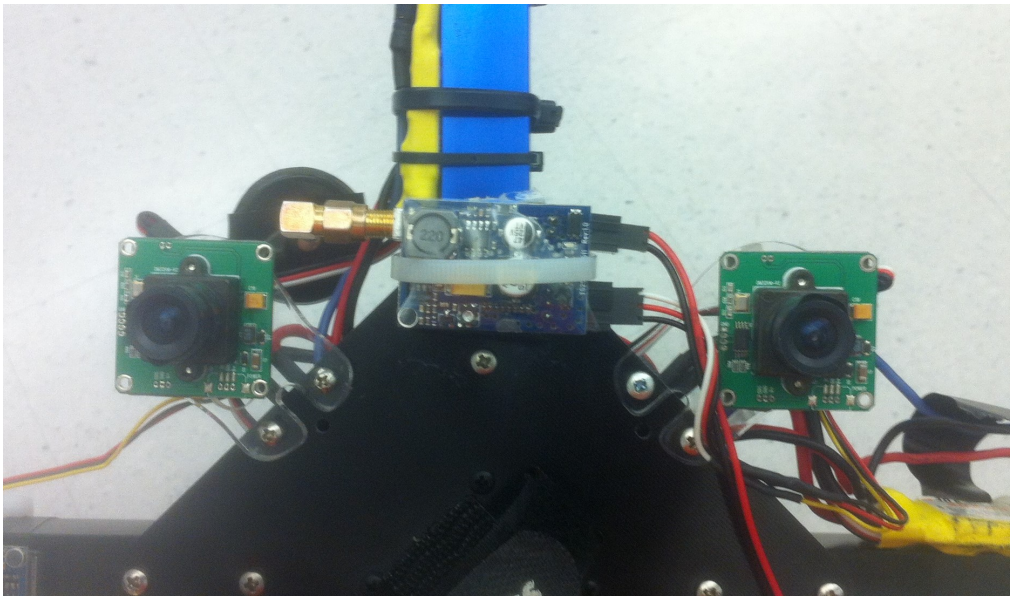


FIGURE 6.7: Cameras mounted on the bottom of the quadrotor in horizontal stereo.

Chapter 7

Experimental Platform: Sensor Calibration

Each sensor was calibrated before its use since electronics are prone to biases which must be compensated for in order to improve the accuracy and consistency of the measurements. For example, accelerometers and gyroscopes are prone to slowly varying or constant measurement biases. Magnetometers, which measure ambient magnetic fields, cannot distinguish between the Earth's magnetic field (which was used as the reference magnetic field) and locally induced magnetic fields. The lenses of cameras have a curved shape in order to capture a wider field of view; unfortunately, this curvature results in what is referred to as lens distortion when capturing images. These aforementioned inaccuracies must therefore be compensated for via calibrations. The following discussion details the calibration techniques applied for each of the quadrotor's sensors and the results which were acquired.

7.1 Accelerometer

The accelerometer measured the instantaneous linear acceleration of the quadrotor. The accelerometer measurements are expressed as

$$\mathbf{a}_{\mathcal{B}} = R^{\top} \left(\mathbf{a}_{\mathcal{I}} - g\hat{\mathbf{e}}_z \right) + \mathbf{b}_a + \boldsymbol{\eta}_a, \tag{7.1}$$

where $\mathbf{a}$ is linear acceleration, $\mathcal{B}$ as a subscript represents a vector expressed in the body-fixed frame, R is the rotation matrix, $\mathcal{I}$ as a subscript represents a vector expressed in the inertial frame, g is the gravity constant, $\hat{\mathbf{e}}_z$ is a unit vector which represents the down axis of the inertial frame, $\mathbf{b}_a$ is a constant bias term of the accelerometer, and $\boldsymbol{\eta}_a$ is additive white Gaussian noise (AWGN) affecting the accelerometer measurements. To compensate for the bias term, $\mathbf{b}_a$, the drone was kept stationary at the identity orientation, *i.e.* $\mathbf{a}_{\mathcal{B}} = [0\ 0\ 0]^T$ and $R = I_3$, and the bias estimate was calculated as

$$\mathbf{b}_a = \frac{1}{N} \sum_{i=1}^{N} \mathbf{a}_{\mathcal{B},i} + g\hat{\mathbf{e}}_z, \tag{7.2}$$

where N is some number of samples and i is the sample index. This bias value was then subtracted from accelerometer measurements to compensate for this initial offset.

7.2 Gyroscope

The gyroscope measured the instantaneous angular velocity of the quadrotor. The gyroscope measurements are expressed as

$$\boldsymbol{\Omega}_{\mathcal{B}} = R^{\top}\boldsymbol{\Omega}_{\mathcal{I}} + \mathbf{b}_{\omega} + \boldsymbol{\eta}_{\omega} \tag{7.3}$$

where $\boldsymbol{\omega}$ is angular velocity, $\mathbf{b}_{\omega}$ is a constant bias term of the gyroscope, and $\boldsymbol{\eta}_{\omega}$ is AWGN affecting the gyroscope measurements. To compensate for the bias term, $\mathbf{b}_{\omega}$, the drone was kept stationary at the identity orientation and the bias estimate was calculated as

$$\mathbf{b}_{\omega} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{\Omega}_{\mathcal{B}}(i)\,. \tag{7.4}$$

Similar to the accelerometer, this bias value was then subtracted from gyroscope measurements to compensate for this initial offset.

7.3 Magnetometer

The magnetometer measured the ambient magnetic field. The magnetometer measurements are expressed as

$$\mathbf{m}_{\mathcal{B}} = DR^{\top}\mathbf{m}_{\mathcal{I}} + \mathbf{b}_{m} + \boldsymbol{\eta}_{m}, \tag{7.5}$$

where $\mathbf{m}$ is the measured ambient magnetic field, D is the distortion of the magnetic field, $\mathbf{b}_{m}$ is a constant bias term of the magnetometer, and $\boldsymbol{\eta}_{m}$ is AWGN affecting magnetometer measurements. The distortion was determined using the method described in [43], [86], [112], [113]. The distortion matrix, $D \in \mathbb{R}^{3\times 3}$ is given as

$$D = \begin{bmatrix} \varepsilon_1 & 0 & 0 \\ \varepsilon_2\sin(\delta_1) & \varepsilon_2\cos(\delta_1) & \\ \varepsilon_3\sin(\delta_2)\cos(\delta_3) & \varepsilon_3\sin(\delta_3) & \varepsilon_3\cos(\delta_2)\cos(\delta_3) \end{bmatrix} \tag{7.6}$$

where ε_i is a total scale error and δ_i is a sensor misalignment angle, for $i = \{1, 2, 3\}$. If the noise is neglected, then eq. (7.5) can be written as

$$R^{\top}\mathbf{m}_{\mathcal{I}} = D^{-1}\left(\mathbf{m}_{\mathcal{B}} - \mathbf{b}_{m}\right), \tag{7.7}$$

where

$$D^{-1} = \begin{bmatrix} D_1 & 0 & 0 \\ D_2 & D_3 & 0 \\ D_4 & D_5 & D_6 \end{bmatrix}, \tag{7.8}$$

and

$$D_1 = \frac{1}{\varepsilon_1} \tag{7.9}$$

$$D_2 = -\frac{\tan\delta_1}{\varepsilon_1} \tag{7.10}$$

$$D_3 = \frac{1}{\varepsilon_2 \cos\delta_1} \tag{7.11}$$

$$D_4 = \frac{\tan\delta_1 \tan\delta_3 - \tan\delta_2 \cos\delta_2}{\varepsilon_1 \cos\delta_2} \tag{7.12}$$

$$D_5 = -\frac{\tan\delta_3}{\varepsilon_2 \cos\delta_1 \cos\delta_2} \tag{7.13}$$

$$D_6 = \frac{\varepsilon_1 \cos\delta_2}{\cos\delta_3}. \tag{7.14}$$

Taking the norm of eq. (7.7) results in

$$C_1 m_{\mathcal{B}_x}^2 + C_2 m_{\mathcal{B}_x} m_{\mathcal{B}_y} + C_3 m_{\mathcal{B}_x} m_{\mathcal{B}_z} + C_4 m_{\mathcal{B}_y}^2 + C_5 m_{\mathcal{B}_y} m_{\mathcal{B}_z} + C_6 m_{\mathcal{B}_z}^2 + C_7 m_{\mathcal{B}_x} + C_8 m_{\mathcal{B}_y} + C_9 m_{\mathcal{B}_z} = C_{10} \tag{7.15}$$

where the coefficients, C_i, are nonlinear functions of $\boldsymbol{\varepsilon}$, $\boldsymbol{\delta}$, $\mathbf{b}$, and $\|\mathbf{m}_{\mathcal{I}}\|$. The value of $\|\mathbf{m}_{\mathcal{I}}\|$ was found using the global magnetic field calculator in [114]. A set of N data points from the magnetometer are collected. Rewriting the system as

$$\mathbb{M}\mathbb{C} = \mathbb{I} \tag{7.16}$$

where

$$\mathbb{M} = \begin{bmatrix} m_{\mathcal{B}_{x,1}}^2 & m_{\mathcal{B}_{x,1}} m_{\mathcal{B}_{y,1}} & \cdots & m_{\mathcal{B}_{z,1}} \\ \vdots & \vdots & \ddots & \vdots \\ m_{\mathcal{B}_{x,N}}^2 & m_{\mathcal{B}_{x,N}} m_{\mathcal{B}_{y,N}} & \cdots & m_{\mathcal{B}_{z,N}} \end{bmatrix}, \quad \mathbb{C} = \begin{bmatrix} C_1/C_{10} \\ \vdots \\ C_9/C_{10} \end{bmatrix}, \quad \mathbb{I} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}. \tag{7.17}$$

Isolating $\mathbb{C}$ in eq. (7.16) yields

$$\mathbb{C}_{est} = \left(\mathbb{M}^T \mathbb{M}\right)^{-1} \mathbb{M}^T \mathbb{I}, \tag{7.18}$$

where $\mathbb{C}_{est}$ is a least-squares solution for $\mathbb{C}$. This system of nine nonlinear equations, with nine unknowns ($\boldsymbol{\varepsilon}$, $\boldsymbol{\delta}$, and $\mathbf{b}$), can then be solved numerically. Planar plots, and a three dimensional plot, of the uncalibrated and calibrated data are shown in fig. 7.1 The values obtained for this experiment were:

$$\boldsymbol{\delta} = \begin{bmatrix} 0.8949 \\ 0.8767 \\ 0.8423 \end{bmatrix}, \quad \boldsymbol{\varepsilon} = \begin{bmatrix} -0.0387 \\ 0.0200 \\ -0.0412 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0.0611 \\ 0.0165 \\ 0.0777 \end{bmatrix}. \tag{7.19}$$

These values were then substituted into eqs. (7.9) to (7.14) to be used in eq. (7.7) as the values of D^{-1}.

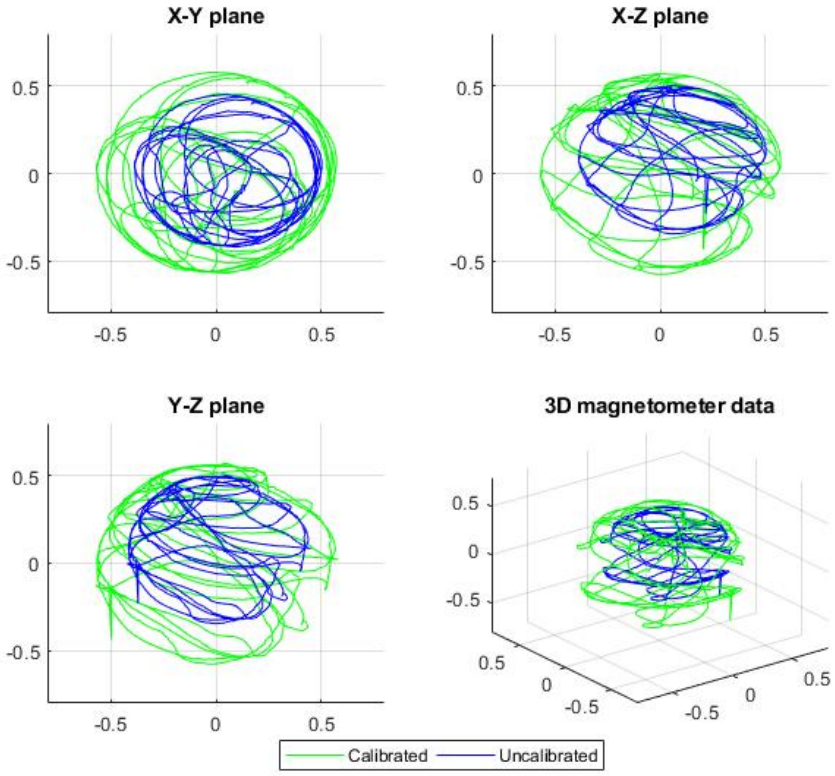


FIGURE 7.1: Magnetometer calibration results.

7.4 Cameras

Cameras suffer from inherent distortion caused by the way that light is bent by the camera lens. To correct for this distortion, inherent parameters of the camera must be determined. A method similar to Tsai [58] was used from the OpenCV libraries to determine the intrinsic parameters of the camera. This is discussed in terms of individual camera calibration and in terms of stereo camera calibration in the following subsections.

7.4.1 Individual Camera Calibration

The first step from [58] was to find a rigid body transformation between the inertial frame $\mathcal{I}$ and the camera frame $\mathcal{C}$:

$$\begin{bmatrix} x_{\mathcal{C}} \\ y_{\mathcal{C}} \\ z_{\mathcal{C}} \end{bmatrix} = {}^{\mathcal{I}}_{\mathcal{C}}R \begin{bmatrix} x_{\mathcal{I}} \\ y_{\mathcal{I}} \\ z_{\mathcal{I}} \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}. \tag{7.20}$$

The rotation matrix, ${}^{\mathcal{I}}_{\mathcal{C}}R$, and translation vector, $\mathbf{T} = [T_x, T_y, T_z]$, are known as extrinsic parameters; they are parameters which are external to the camera and affect how captured points are expressed as coordinates. The rotation matrix is defined such that

$$\mathbf{x}_{\mathcal{C}} = {}^{\mathcal{C}}_{\mathcal{I}}R^{\top}\, \mathbf{x}_{\mathcal{I}}. \tag{7.21}$$

The OpenCV calibration does not provide these parameters and they must be determined independently. The pinhole camera projection showing the rotation and translation from $\mathcal{C}$ to $\mathcal{I}$ is presented in fig. 7.2 [115]. Due to how the cameras were mounted on the quadrotor, a simpli-

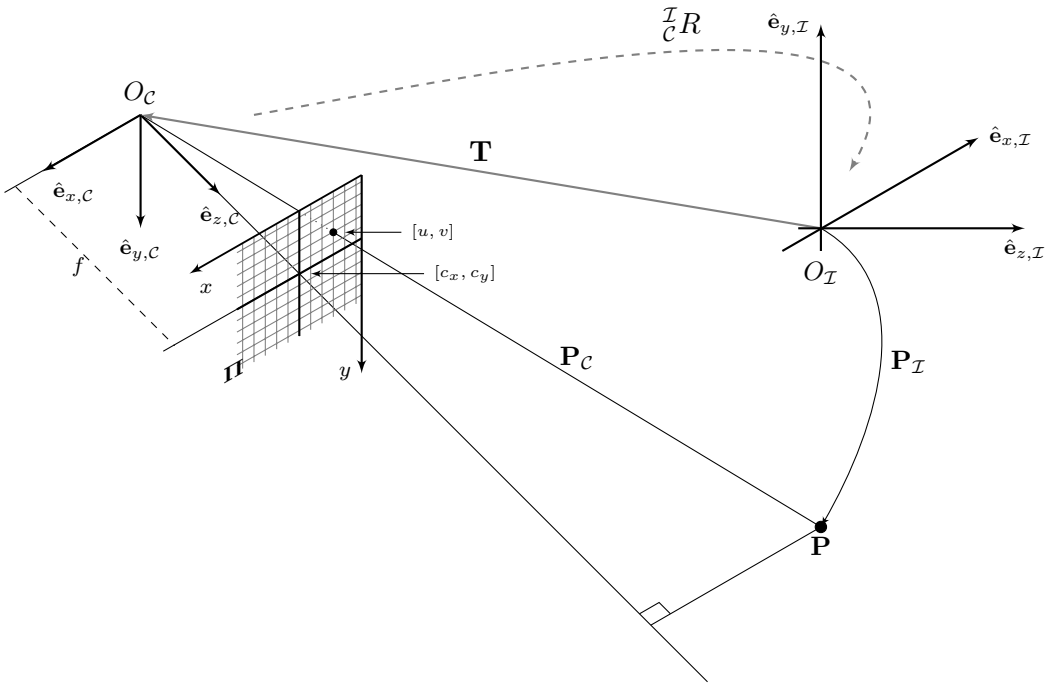


FIGURE 7.2: Pinhole camera projection.

fied approach was taken. Since the cameras rotate and translate with the quadrotor, the attitude determined by the quadrotor could be used to transform the quadrotor's position from $\mathcal{B}$ to $\mathcal{I}$. Therefore, it was sufficient to find

$$ {}^{\mathcal{C}}_{\mathcal{B}}R = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad {}^{\mathcal{C}}_{\mathcal{B}}T = \begin{bmatrix} 0.057 \\ -0.057 \\ 0.055 \end{bmatrix}, \tag{7.22} $$

which transformed $\mathbf{p}_{\mathcal{C}}$ to $\mathbf{p}_{\mathcal{B}}$.

Next, the intrinsic (internal) parameters of the camera were determined. This was done using a perspective transformation with pinhole camera geometry from the three-dimensional camera coordinates to the ideal image coordinate:

$$ u = f\frac{x}{z}, \tag{7.23a} $$

$$ v = f\frac{y}{z}, \tag{7.23b} $$

where $\mathbf{p_u} = [u, v, 1]^\mathsf{T}$ is the homogeneous position vector of ideal, undistorted pixel coordinates in the image, f is the focal length of the camera which is defined as the distance from the ocular center to the image plane in pixels, and $[x, y, z]^\mathsf{T}$ are the 3D coordinates of the detected point

with respect to $\mathcal{C}$. The third step was to determine the radial lens distortion such that

$$X_d D = u, \tag{7.24a}$$

$$Y_d D = v, \tag{7.24b}$$

where $[X_d, Y_d]$ are the detected image coordinate, and

$$D = 1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6 + \cdots, \tag{7.25}$$

where κ_i, $i \in \mathbb{N}$ are the distortion parameters. The final calibration step was to transform from the detected image coordinates to the computer image coordinates by

$$X_f = s X_d + c_x, \tag{7.26a}$$

$$Y_f = X_d + c_y, \tag{7.26b}$$

where s is a scale factor defined by the ratio between the length and width of a CMOS element and $[c_x, c_y]$ are the coordinates of the center of the image in pixels. In [58], the center of the image buffer was considered to be the image center for this calibration step; however, these coordinates can be changed to translate the undistorted image in the computer display. Using the OpenCV libraries and a series of images showing a known pattern, the intrinsic parameters of the cameras were determined and are shown in table 7.1. The intrinsic calibration procedure was performed for the left and right cameras independently.

TABLE 7.1: Individual intrinsic camera parameters.

Parameter	Left Camera	Right Camera
f	456.4	461.5
s	1.059	1.059
c_x	331.4	341.6
c_y	211.4	244.4
κ_1	-0.5176	-0.4976
κ_2	0.3257	0.2730
κ_3	-0.1186	-0.0791

7.4.2 Stereo Camera Calibration

Shown in fig. 7.3 is a generalised version of data being captured from a scene in two viewpoints. A line drawn through the optical center of one viewpoint, C_1, and the optical center of the second viewpoint, C_2, is called the baseline. The points where this line intersects the image planes are called the epipoles. Lines which pass through a detected image point and the epipole are called epipolar lines. Stereo camera calibration has two goals:

1. Rotate the image planes such that the epipoles are at infinity; and
2. Translate the image planes such they are coplanar and the epipolar lines of each plane are collinear.

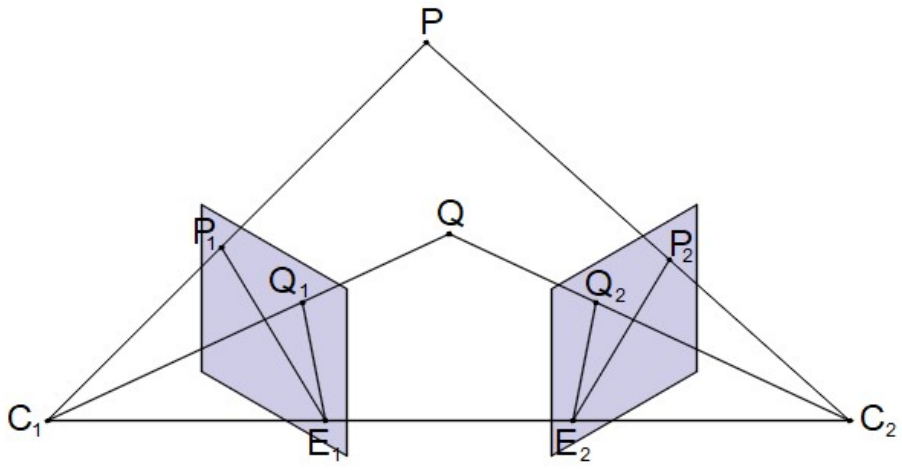


FIGURE 7.3: Generalized imaging of a scene using two viewpoints.[49]

A rectified pair of image planes with collinear epipolar lines is shown in Figure 7.4. Having collinear epipolar lines is useful because it allows faster matching between the two views using only horizontal scanlines. Appropriate scanlines can also be calculated if the fundamental matrix of the two viewpoint system is known. The fundamental matrix relates homogeneous image coordinates between the two images and can be determined using the normalized 8-point algorithm [59]–[62]. The fundamental matrix is defined through

$$\mathbf{x}'^{\top} F \mathbf{x} = 0, \tag{7.27}$$

where $\mathbf{x} = [u, v, 1]^{\top}$ is the homogeneous coordinates of a detected point in the first image, $\mathbf{x}' = [u', v', 1]^{\top}$ is the homogeneous coordinates of a matching point in the second image, and

$$F \in \{\mathbb{R}^{3\times 3} | \text{ rank}(F) = 2\} \tag{7.28}$$

is the fundamental matrix. If a point in the first image and F are known, then

$$\mathbf{l}' = F\mathbf{x} \tag{7.29}$$

describes a line, $\mathbf{l}'$, in the second image upon which the matching point will lie. Observing fig. 7.3, if the fundamental matrix between the two image frames were known, then the line upon which P_2 lies can be calculated using the homogeneous coordinates of P_1 in eq. (7.29). To determine the values of F, eq. (7.28) is first rewritten as a linear equation:

$$\mathbf{u}_i^{\top} \mathbf{f} = 0, \tag{7.30}$$

where

$$\mathbf{u}_i = [u_i u_i', v_i u_i', u_i', u_i v_i', v_i v_i', v_i', u_i, v_i, 1]^{\top}, \tag{7.31a}$$

$$\mathbf{f} = [F_{11}, F_{12}, F_{13}, F_{21}, F_{22}, F_{23}, F_{31}, F_{32}, F_{33}]^{\top}, \tag{7.31b}$$

and F_{ij} is the element of F at row i and column j. Given n point matches, eq. (7.28) can be stacked to obtain

$$U_n \mathbf{f} = 0, \tag{7.32}$$

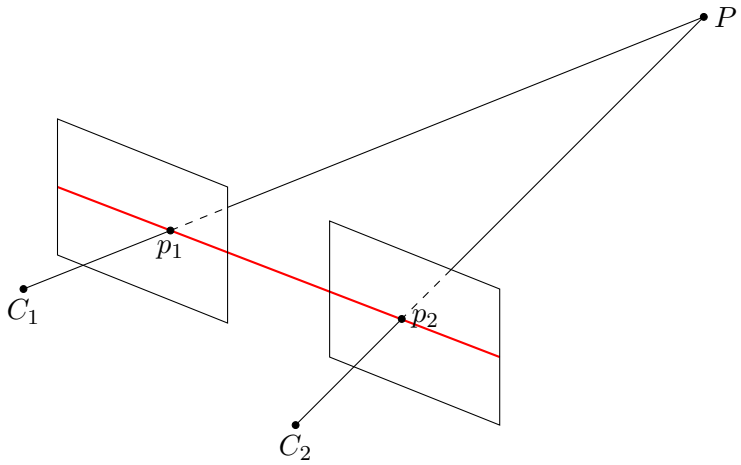


FIGURE 7.4: Rectified imaging of a scene using two viewpoints.[115]

where

$$U_n = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_n \end{bmatrix}^\top. \tag{7.33}$$

This set of linear equations can then be used to solve the for values of F, up to a scale, using a least-squares solution. For more details, see [60].

The stereo vision system was rectified using the OpenCV libraries and a known pattern which was captured simultaneously by the left and right cameras. The rotation matrices to rotate the image planes of the left and right views to be coplanar were found to respectively be

$$R_l = \begin{bmatrix} 0.998 & -0.002 & 0.07 \\ 0.003 & 0.9999 & -0.011 \\ -0.07 & 0.011 & 0.997 \end{bmatrix}, \qquad R_r = \begin{bmatrix} 0.997 & 0.061 & 0.050 \\ -0.061 & 0.998 & 0.011 \\ -0.049 & -0.014 & 0.999 \end{bmatrix}, \tag{7.34}$$

and the fundamental matrix was found to be

$$F = \begin{bmatrix} -8.86 \times 10^{-7} & -6.20 \times 10^{-6} & 1.09 \times 10^{-}2 \\ 1.97 \times 10^{-4} & 2.11 \times 10^{-5} & -2.49 \times 10^{-1} \\ 7.89 \times 10^{-4} & 2.37 \times 10^{-1} & 1 \end{bmatrix}. \tag{7.35}$$

The camera calibration also modified the scale and positioning of the images displayed by the computer to align the scanlines which makes the epipolar lines collinear. The modified intrinsic parameters of the stereo vision system are shown in table 7.2. The intrinsic camera parameters for individual and stereo calibration were used as inputs for the "initUndistortRectifyMap" OpenCV function [116] which calculated remapping matrices. These remapping matrices were then applied to the captured image matrices which allowed rectification of the images for feature detection, feature matching, and position determination.

TABLE 7.2: Stereo intrinsic camera parameters.

Parameter	Left Camera	Right Camera
f	458.9	458.9
s	1.059	1.059
c_x	280.0	280.0
c_y	170.8	170.8
κ_1	-0.5176	-0.4976
κ_2	0.3257	0.2730
κ_3	-0.1186	-0.0791

Chapter 8

Position and Velocity Estimation and Control Using Stereo Vision and Kalman Filtering

This chapter details the linear position and velocity control of the quadrotor. A block-flow diagram of the overall control system is shown in fig. 8.1. In section 8.1, the position estimation using stereo computer vision is detailed; the methods of filtering stereo vision point matches and the geometry necessary for position determination of the quadrotor are discussed. The discrete implementation of a Kalman filter is presented in section 8.2. Next, section 8.3 details the determination of the control variables and how the position is ultimately controlled through manipulation of the attitude dynamics. Finally, in section 8.4 the performance of the implemented position controller using stereo computer vision to estimate the position of a quadrotor is shown.

8.1 Position Estimation Using Stereo Computer Vision

The position of the quadrotor was estimated using the stereo vision cameras mounted on the bottom of the quadrotor which faced downwards. Within the images, an object which closely matched the desired star contour and enclosed the greatest number of pixels was utilised to determine its position using the centroid of the contour. These centroids were used to determine the position of the quadrotor relative to the detected contour using simple triangulation. To calculate the depth of the observed centroid from the quadrotor, the geometry of the observation must be established. The geometry of the camera, including the observed point and the captured point is

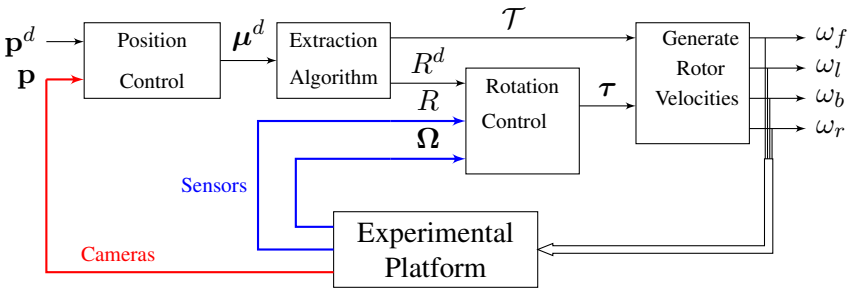
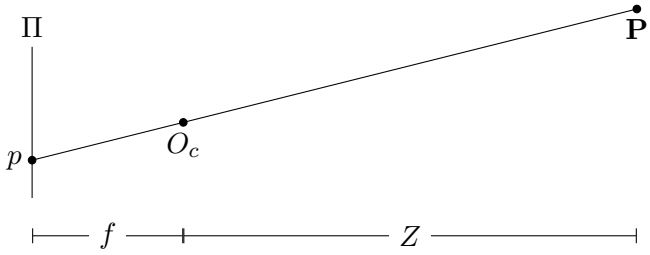
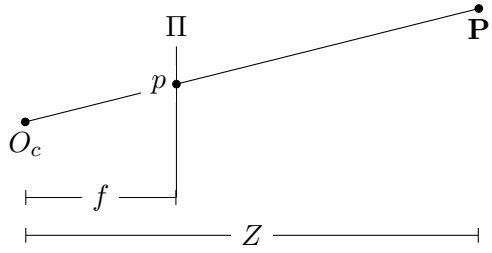


FIGURE 8.1: Block diagram of the quadrotor's control system.

called the camera projection. The most commonly used camera projection is the pinhole camera projection. It is the most common projection because of its simplicity. The simplicity is due to the linear geometry which avoids nonlinearities that could be caused by the lens. The pinhole camera method treats the camera lens as a single point to avoid nonlinear equations which would be caused by the curvature of light on its way to the detector. These nonlinearities are accounted for in the camera calibration which is discussed in detail in section 7.4. The geometry of the pinhole camera projection is shown in fig. 8.2. Intuitively, the image capture plane can be moved in front of the camera lens to avoid the need of inverting the image to determine the pixel coordinates of the captured image feature. In fig. 8.2, Π is the image capture plane, $\mathbf{p} = [u, v, 1]^\mathsf{T}$ is the



(A) A simplification of the camera's physical layout.



(B) The utilised geometry of the pinhole camera projection.

FIGURE 8.2: The geometry of the pinhole camera projection where Π is the image capture plane, p is the detected point in the image, O_c is the center of mass of the camera, $\mathbf{P}$ is the detected physical point, f is the focal length of the camera, and Z is the calculated depth of the detected point.

vector representing the homogeneous coordinates of the point in the image detector expressed in pixels, $\mathbf{P} = [X, Y, Z]^\mathsf{T}$ is the point of interest in the environment relative to the camera expressed in metres in the camera referenced frame $\mathcal{C}$, 0_c is the origin of the camera coordinate frame, also called the ocular center, and is at the center of mass of the camera lens, and f is the focal length of the camera in pixels. Due to the geometry of the pinhole camera projection, the ratio

$$\frac{X}{p} = \frac{Z}{f} \tag{8.1}$$

cannot be relied upon to determine the distance of a point from the camera because as long as the point $\mathbf{P}$ lies along the line passing through $\overline{O_c\mathbf{P}}$, the above ratio cannot be determined. This is what causes the need of having two cameras. Two well-calibrated cameras with coplanar image planes can be used to perform simple triangulation. Shown in fig. 8.3 is the geometry of calibrated stereo cameras using the pinhole camera geometry where the subscripts l and r denote features of the left and right camera's pinhole camera projections, respectively, and b represents the physical distance between the centres of mass of the camera lenses, is known as the baseline,

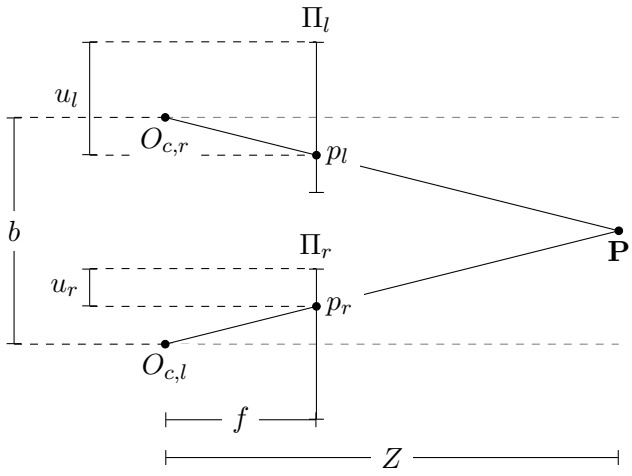


FIGURE 8.3: Stereo pinhole camera geometry.

and is expressed in metres. Using the shown geometry, similar triangles can be constructed to determine the depth of the point using a simple triangular ratio. The first triangle is created by the detected point and the camera lens centres. The second triangle uses only internal aspects of the cameras. The image planes are translated so that they share the same area of the plane they occupy and the ocular centres become one point but the values of u_l and u_r are not changed. The second triangle is then created by the points p_l, p_r, and O_c (O_{c_l} and O_{c_r} are a single point). The ratio created by the similar triangles is therefore:

$$\frac{Z}{b} = \frac{f}{u_l - u_r}, \tag{8.2}$$

which can be solved to determine the image depth as:

$$Z = \frac{fb}{u_l - u_r}. \tag{8.3}$$

After calculating the depth of the image, the remaining two dimensions of the three-dimensional position of the cameras relative to the observed scene can be solved using a single camera and similar triangles. The left camera was arbitrarily chosen. The values for X and Y can then be found as:

$$X = \frac{Z\,(u_l - c_{l,x})}{f} \tag{8.4a}$$

$$Y = \frac{Z\,(v_l - c_{l,y})}{f}. \tag{8.4b}$$

To see how these ratios were constructed, see fig. 7.2.

The image depth was found using a single known object matched in the left and right camera images. To determine the quadrotor's horizontal position coordinates, this object was also used. For this work, the four-pointed star shown in fig. 8.4 was used as the known object. The shape was detected by finding closed contours in one image of the stereo pair. The detected closed contours were then compared to a stored ideal contour by performing a correlation between the

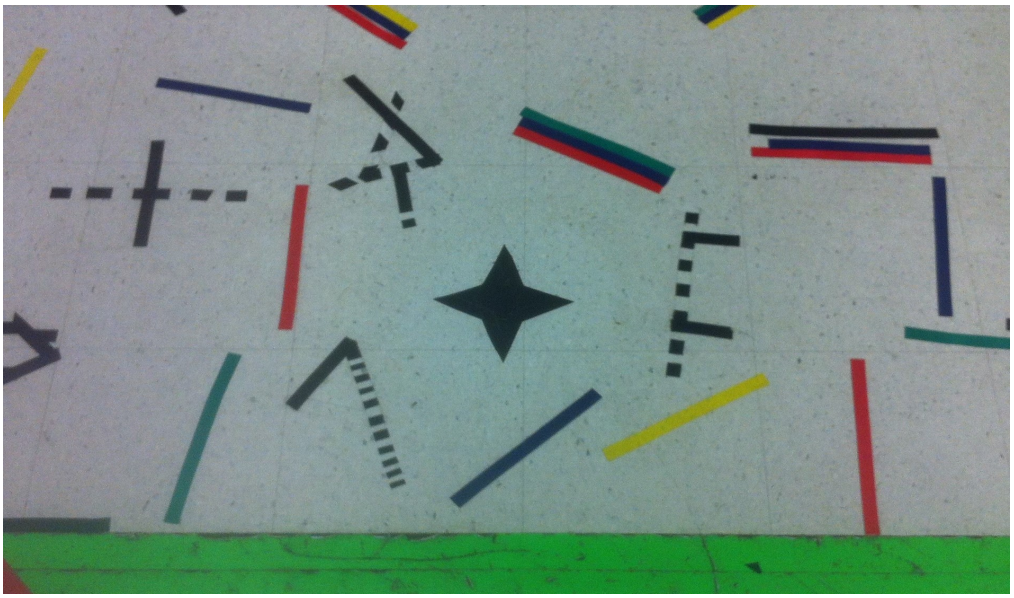


FIGURE 8.4: Four-pointed star used for horizontal position reference.

detected contours and the stored contour. The first step to finding a contour which could be matched with the star was to detect all the contours within the image. The contours in the image frame were detected using the following algorithm:

1. Convert the image to greyscale.
2. Invert the image.
3. Threshold the image so that regions which are currently very bright (originally very dark) are solid white shapes and the reset of the image is black.
4. Perform morphological operations on the image to smooth edges.
5. Use the OpenCV function "findContours" on the image to detect closed shape contours.

The OpenCV function "findContours" uses the line detection algorithm from [117]. The "findContours" function finds the contours of white shapes in the image, not dark shapes, which is why the greyscale image was inverted.

Due to how the CMOS image sensors within the cameras capture an image, during high velocity manoeuvres the detected image could be distorted. The movement of the camera during the $16.67ms$ between the camera capturing the even and odd numbered lines would cause the image to become distorted. This caused the contours to appear as much more complex than they truly were. To smooth the objects in the thresholded image, a sequence of morphological operations were performed. First, the shapes within the image were eroded; this means pixels with fewer than four neighbouring white pixels would be set to black. Next, the white objects within the image were dilated twice. Dilation sets any pixel with three or more neighbouring white pixels to white. Finally, the image was eroded a second time. It was found that these morphological operations allowed the contour of the star to be detected even under fast manoeuvres. The contours were expressed by a minimal vector of points.

Once the contours within the image were detected, a filtering process was performed to eliminate any contours that would be poor matches. The contours were filtered out based on the following five criteria:

1. The contour must be at least two pixels away from the edge of the image.
2. The contour must contain more than 350 pixels.
3. The ratio of the height to the width of the contour must be between 0.7 and 0.7^{-1}.
4. The absolute difference between the number of points which describe the desired contour and the number of points required to describe the detected contour must be less than or equal to one.
5. The sum of absolute differences between the inverse of the first seven Hu invariant moments [118] must be less than 0.5 and greater than 10^{-7}.

Contours too close to the edge of the image were discarded because the contour of an object partially within the image frame could have the necessary geometry to provide a stronger correlation to the star's shape. This constraint was imposed after finding that the detected position would step approximately $0.5m$ between image frames. It was found that most of the false positives were filtered out and successful detection of the star was significantly increased after contours near the edge of the image were filtered. Contours which enveloped less than 350 pixels were chosen to be discarded because salt noise (random single white pixels) were found to have strong correlations with the desired object due to the salt noise having perfect symmetry. Also, the pixel area of the star's contour was found to be greater than 350 pixels when the quadrotor was at a distance of approximately $2.6m$, the chosen maximum height due to hanging lights in the experiment area. The hight:width ratios were chosen to be approximately square; the value 0.7 was chosen to ensure that at relatively extreme rotations for the application the star would still be detected. The number of points contained within the minimal contour vector was the fourth criteria. The difference between the number of points for the ideal star (eight points each describing a vertex of the star) and the detected contour should be minimized. An absolute difference of one was chosen because the star's contour was not guaranteed to have exactly eight points describing it so some leeway was allowed (mostly described by eight points and sometimes nine). Finally, the Hu invariants were used to compare contours. The contour matching algorithm from the OpenCV libraries uses the Hu invariant image moments to compare contours. Hu invariant moments are used because they are invariant to orientation and scale differences between two objects. The bounds of the sum of absolute differences between the Hu invariant moments were chosen as they are through testing. The lower bound of 10^{-7} was chosen because salt noise was providing false positives. The upper bound of 0.5 was found empirically. If the star was successfully detected, the position of the cameras relative to the observed scene could be easily determined.

The above image processing was performed on the base station computer due to the processing power that was required for the stereo image rectification and contour detection. The hardware used made integration of the microcontroller with the cameras difficult since the cameras output an analogue signal which the transmitter and receiver discussed in section 6.2.2 handled. The software on the base station estimated the position of the camera relative to the detected features. The duration of time required to perform this algorithm was transmitted to the quadrotor along with the estimated position vector of the camera. A loop of the base station software was approximately 30*ms*, the duration of time between frame captures of the cameras. The position was

then transformed from the image coordinate frame to the inertial frame by:

$$\mathbf{p}_{\mathcal{I}} = {}^{\mathcal{I}}_{\mathcal{B}}R\left({}^{\mathcal{B}}_{\mathcal{C}}R\left(\mathbf{p}_{\mathcal{C}} + {}^{\mathcal{B}}_{\mathcal{C}}T\right)\right), \tag{8.5}$$

where $\mathbf{p}_{\mathcal{I}}$ is the position of the quadrotor in $\mathcal{I}$, ${}^{\mathcal{I}}_{\mathcal{B}}R$ is a rotation matrix which rotates a vector from $\mathcal{B}$ to $\mathcal{I}$, ${}^{\mathcal{B}}_{\mathcal{C}}R$ is a rotation matrix which rotates a vector from $\mathcal{C}$ to $\mathcal{B}$, $\mathbf{p}_{\mathcal{C}}$ is the position of the quadrotor in $\mathcal{C}$, and ${}^{\mathcal{B}}_{\mathcal{C}}T$ is a linear transformation from the origin of $\mathcal{C}$ to $\mathcal{B}$. The transformed position and time were used to fully estimate the position using the Kalman filter described in the next section.

8.2 Discrete Kalman Filter

The data from multiple sensors must be fused on the quadrotor to estimate the position. The Kalman filter was used in this work because the data was used to estimate the linear position and velocity. The Kalman filter was implemented on a discrete system which necessitated that the dynamics be discretized. The discretized version is based on the work from [119]. From eq. (5.1), the translational dynamics of the quadrotor are:

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \mathbf{u}, \end{aligned} \tag{8.6}$$

with

$$\mathbf{u} \coloneqq g\hat{\mathbf{e}}_z + \hat{R}\mathbf{a}_{\mathcal{B}}, \tag{8.7}$$

where $\hat{R}$ is the estimated rotation matrix determined using the method described in chapter 4 and $\mathbf{a}_{\mathcal{B}}$ is the accelerometer measurement. The state-space representation of the system is given by:

$$\begin{aligned} \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{v}} \end{bmatrix} &= \begin{bmatrix} 0 & I_3 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} + \begin{bmatrix} 0 \\ I_3 \end{bmatrix} \mathbf{u} \\ \mathbf{y} &= \begin{bmatrix} I_3 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} \end{aligned} \tag{8.8a}$$

$$\begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\mathbf{u} \\ \mathbf{y} &= C\mathbf{x}, \end{aligned} \tag{8.8b}$$

where $\mathbf{x} = [\mathbf{p}^{\mathsf{T}}, \mathbf{v}^{\mathsf{T}}]^{\mathsf{T}}$. Discretizing eq. (8.8b) yields:

$$\begin{aligned} \mathbf{x}_{k+1} &= F\mathbf{x}_k + G\mathbf{u}_k \\ \mathbf{y}_k &= H\mathbf{x}_k, \end{aligned} \tag{8.9}$$

where

$$F = \begin{bmatrix} I_3 & TI_3 \\ 0 & I_3 \end{bmatrix}, \quad G = \begin{bmatrix} \frac{1}{2}T^2 I_3 \\ TI_3 \end{bmatrix}, \quad H = \begin{bmatrix} I_3 & 0 \end{bmatrix}, \tag{8.10}$$

$\mathbf{x}_k$ is the system state vector at time t_k, $\mathbf{u}_k$ is the linear virtual acceleration of the quadrotor expressed in $\mathcal{I}$ at time t_k, $\mathbf{y}_k$ is a measurement at time t_k, and T is the elapsed time between iterations of the Kalman filter.

The model used for the linearized Kalman filter is as follows:

$$\textit{A priori} \text{ estimate} : \begin{cases} \hat{\mathbf{x}}_{k+1}^{-} = F\hat{\mathbf{x}}_k + G\mathbf{u}_k \\ P_{k+1}^{-} = FP_kF^{\mathsf{T}} + Q_k \end{cases} \tag{8.11a}$$

$$\textit{A posteriori} \text{ update} : \begin{cases} K_k = P_{k+1}^{-}H^{\mathsf{T}}\left(HP_{k+1}^{-}H^{\mathsf{T}} + E_k\right)^{-1} \\ \hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_k^{-} + K_k\left(\mathbf{y}_k - H\hat{\mathbf{x}}_{k+1}^{-}\right) \\ P_{k+1} = \left(I_6 - K_kH\right)P_{k+1}^{-}, \end{cases} \tag{8.11b}$$

where P_k is the estimate covariance matrix at time t_k, Q is the process covariance matrix at time t_k, and E_k is the measurement covariance matrix at time t_k. A hyphen as a superscript denotes the value as an *a priori* estimate. The *a priori* estimate equations operate at a much higher frequency than the *a posteriori* update equations. The *a posteriori* equations are calculated only when a new measurement has been received and therefore act as a filter of the measurements. The calculations were simplified to increase the speed of the algorithm by assuming the variances of each axis were independent from one another. This assumption allowed $P_k \in \mathbb{R}^{2\times2}$, $Q_k \in \mathbb{R}^{2\times2}$, $E_k \in \mathbb{R}$, and $K_k \in \mathbb{R}^2$ reducing the number of calculations required dramatically.

The process and measurement covariance matrices for each axis are given by

$$Q_k = \begin{bmatrix} \sigma_{q,p}^2 T & 0 \\ 0 & \sigma_{q,v}^2 T \end{bmatrix} \tag{8.12a}$$

$$E_k = \sigma_r^2, \tag{8.12b}$$

where σ^2 represents a particular variance. The variances for Q were estimated and then updated between experiments to improve results. The variances for R were determined by calculating the measurement variance of the camera system from multiple static points with known location and averaging those variances.

A conceptualization of the Kalman filter is shown in fig. 8.5. The *a priori* estimates cycle continuously and whenever a measurement has been received, the switch closes after the state has been estimated for that cycle and then the measurement is filtered.

8.3 Position Controller

Consider the following translational dynamics of the quadrotor:

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \boldsymbol{\mu} \end{aligned} \tag{8.13}$$

where $\boldsymbol{\mu} = g\hat{\mathbf{e}}_z - \frac{\mathcal{T}}{m}R^{\top}\hat{\mathbf{e}}_z$.

The position controller operates using a hierarchical method. The attitude controller operates at a much higher frequency than the position controller. The output of the position controller is used as the control input for the attitude controller. For the design of the position controller, the linear

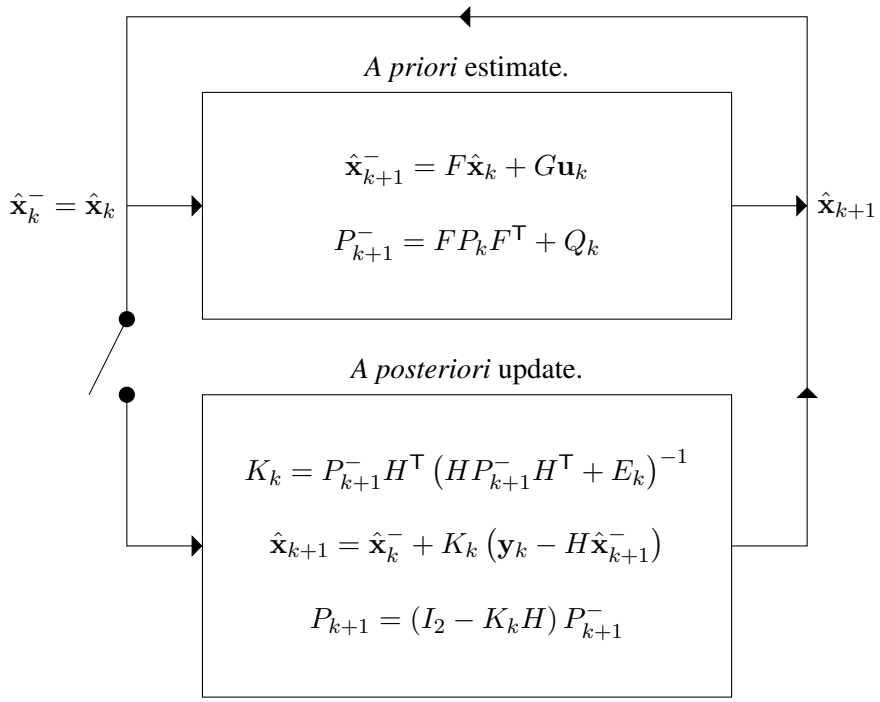


FIGURE 8.5: Kalman filter block diagram.

dynamical errors are defined as:

$$\begin{aligned}\tilde{\mathbf{p}} &\coloneqq \mathbf{p} - \mathbf{p}^d \\ \tilde{\mathbf{v}} &\coloneqq \mathbf{v} - \dot{\mathbf{p}}^d,\end{aligned} \tag{8.14}$$

where $\mathbf{p}^d$ is the desired position and $\dot{\mathbf{p}}^d$ is the desired linear velocity. Taking the derivative of the errors yields:

$$\begin{aligned}\dot{\tilde{\mathbf{p}}} &= \dot{\mathbf{p}} - \dot{\mathbf{p}}^d \\ \dot{\tilde{\mathbf{v}}} &= \dot{\mathbf{v}} - \ddot{\mathbf{p}}^d.\end{aligned} \tag{8.15}$$

The virtual acceleration is then substituted into the error equations for the velocity derivative:

$$\begin{aligned}\dot{\tilde{\mathbf{p}}} &= \dot{\mathbf{p}} - \dot{\mathbf{p}}^d \\ \dot{\tilde{\mathbf{v}}} &= \boldsymbol{\mu} - \ddot{\mathbf{p}}^d.\end{aligned} \tag{8.16}$$

The virtual acceleration error is defined as:

$$\tilde{\boldsymbol{\mu}} \coloneqq \boldsymbol{\mu} - \boldsymbol{\mu}^d, \tag{8.17}$$

where $\boldsymbol{\mu}^d = g\hat{\mathbf{e}}_z - \frac{\mathcal{T}}{m}R_d^{\top}\hat{\mathbf{e}}_z$. The virtual acceleration in eq. (8.16) is substituted with $\boldsymbol{\mu}^d + \tilde{\boldsymbol{\mu}}$ to achieve:

$$\begin{aligned}\dot{\tilde{\mathbf{p}}} &= \dot{\mathbf{p}} - \dot{\mathbf{p}}^d \\ \dot{\tilde{\mathbf{v}}} &= \boldsymbol{\mu}^d + \tilde{\boldsymbol{\mu}} - \ddot{\mathbf{p}}^d.\end{aligned} \tag{8.18}$$

For simplicity, it was assumed that the desired virtual acceleration closely followed the virtual acceleration and that $\tilde{\boldsymbol{\mu}} \approx 0$. Therefore:

$$\begin{aligned} \dot{\tilde{\mathbf{p}}} &= \dot{\mathbf{p}} - \dot{\mathbf{p}}^d \\ \dot{\tilde{\mathbf{v}}} &= \boldsymbol{\mu}^d - \ddot{\mathbf{p}}^d. \end{aligned} \tag{8.19}$$

It was desired for $\boldsymbol{\mu}^d$ to be *a priori* bounded, therefore it was designed as follows:

$$\boldsymbol{\mu}^d = \ddot{\mathbf{p}}^d - K_p \tanh(\tilde{\mathbf{p}}) - K_v \tanh(\tilde{\mathbf{v}}), \tag{8.20}$$

where

$$K_p = \begin{bmatrix} k_{p,x} & 0 & 0 \\ 0 & k_{p,y} & 0 \\ 0 & 0 & k_{p,z} \end{bmatrix}, \qquad K_v = \begin{bmatrix} k_{v,x} & 0 & 0 \\ 0 & k_{v,y} & 0 \\ 0 & 0 & k_{v,z} \end{bmatrix}, \tag{8.21}$$

all diagonal elements of K_p and K_v are positive real values, and the hyperbolic functions act element-wise on the vectors. To prove the asymptotic stability of the desired virtual acceleration, consider the following positive definite Lyapunov function candidate:

$$\dot{\mathcal{V}} = \tilde{\mathbf{v}}^\top K_p \tanh(\tilde{\mathbf{p}}) + \tilde{\mathbf{v}}^\mathsf{T} \dot{\tilde{\mathbf{v}}} \tag{8.22}$$

whose time derivative is

$$\dot{\mathcal{V}} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} K_p \tilde{\mathbf{v}} \tanh(\tilde{\mathbf{p}}) + \tilde{\mathbf{v}}^\mathsf{T} \dot{\tilde{\mathbf{v}}}. \tag{8.23}$$

In view of eq. (8.19) and eq. (8.20), the time derivative becomes

$$\dot{\mathcal{V}} = -\tilde{\mathbf{v}}^\mathsf{T} K_v \tanh(\tilde{\mathbf{v}}), \tag{8.24}$$

which is negative semi-definite which guarantees that $\tilde{\mathbf{v}}$ is bounded. The convergence of $\tilde{\mathbf{p}}$ to zero can be shown through LaSalle's Invariance Theorem. Setting eq. (8.24) to zero, then:

$$\dot{\mathcal{V}} = 0 \implies \tilde{\mathbf{v}} = 0 \implies \dot{\tilde{\mathbf{v}}} = 0. \tag{8.25}$$

The position error can then be shown to converge by substituting eq. (8.20) into eq. (8.19) to obtain:

$$\dot{\tilde{\mathbf{v}}} = -K_p \tanh(\tilde{\mathbf{p}}) - K_v \tanh(\tilde{\mathbf{v}}), \tag{8.26}$$

then setting to zero the states shown in eq. (8.25) yields:

$$\dot{\tilde{\mathbf{v}}} = 0 = -K_p \tanh(\tilde{\mathbf{p}}) - K_v \tanh(0) \implies \tilde{\mathbf{p}} = 0, \tag{8.27}$$

since $\tanh(0) = 0$.

To extract the desired orientation, R_d, and thrust, $\mathcal{T}$, the following procedure was performed:

$$\frac{\mathcal{T}}{m} R_d^\top \hat{\mathbf{e}}_z = g\hat{\mathbf{e}}_z - \boldsymbol{\mu}^d, \tag{8.28}$$

with $\boldsymbol{\mu}$ given in eq. (8.20). The norm of eq. (8.28) was then taken yielding:

$$\left\| \frac{\mathcal{T}}{m} R_d^\top \hat{\mathbf{e}}_z \right\| = \| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}^d \|. \tag{8.29}$$

Finally, eq. (8.29) was solved for $\mathcal{T}$:

$$\mathcal{T} = m\| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}^d \|, \tag{8.30}$$

since

$$\left\| \frac{\mathcal{T}}{m} R_d^\top \hat{\mathbf{e}}_z \right\| = \sqrt{\left(\frac{\mathcal{T}}{m} R_d^\top \hat{\mathbf{e}}_z \right)^\top \left(\frac{\mathcal{T}}{m} R_d^\top \hat{\mathbf{e}}_z \right)} = \frac{\mathcal{T}}{m} \sqrt{\hat{\mathbf{e}}_z^\top R_d R_d^\top \hat{\mathbf{e}}_z} = \frac{\mathcal{T}}{m}, \tag{8.31}$$

since $RR^\top = I_3$ and $\hat{\mathbf{e}}^\mathsf{T}\hat{\mathbf{e}} = 1$.

With the thrust known, the control torques must then be found. First, eq. (8.30) is substituted into eq. (8.28) and rearranged to yield:

$$R_d \frac{g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d}{\| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d \|} = \mathbf{e}_z, \tag{8.32}$$

which has the form

$$R_d \mathbf{u} = \mathbf{v}, \tag{8.33}$$

where $\mathbf{u}$, $\mathbf{v} \in \mathbb{R}^3$, $\|\mathbf{u}\| = \|\mathbf{v}\| \neq 0$, and $\mathbf{u} \neq -\mathbf{v}$. The expression given by eq. (8.33) is mathematically equivalent to

$$\begin{bmatrix} 0 \\ \mathbf{v} \end{bmatrix} = Q_d^{-1} \odot \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} \odot Q_d, \tag{8.34}$$

where Q_d is the desired unit quaternion associated with the rotation matrix. Given the restrictions placed on $\mathbf{u}$ and $\mathbf{v}$, the solution found [17] for Q_d that satisfies eq. (8.34) is given by

$$q_{d0} = \frac{1}{\|\mathbf{u}\|} \sqrt{\frac{\|\mathbf{u}\|^2 + \mathbf{u}^\mathsf{T}\mathbf{v}}{2}} \tag{8.35a}$$

$$\mathbf{q}_d = \frac{1}{\|\mathbf{u}\|} \sqrt{\frac{1}{2(\|\mathbf{u}\|^2 + \mathbf{u}^\mathsf{T}\mathbf{v})}}\, sk(\mathbf{v})\, \mathbf{u}. \tag{8.35b}$$

The proof of eq. (8.35) can be found in [17]. Substituting

$$\mathbf{u} = \frac{g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d}{\| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d \|}, \quad \mathbf{v} = \hat{\mathbf{e}}_z \tag{8.36}$$

into eq. (8.35) then yields

$$q_{d0} = \sqrt{\frac{1}{2}\left(1 + \frac{g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d}{\| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d \|} \right)} \tag{8.37a}$$

$$\mathbf{q}_d = \frac{1}{2\| g\hat{\mathbf{e}}_z - \boldsymbol{\mu}_d \| q_{d0}} \begin{bmatrix} \mu_{d3} \\ -\mu_{d1} \\ 0 \end{bmatrix}, \tag{8.37b}$$

which provides the desired quaternion that was used to find the control torques. The attitude error was then defined as:

$$\tilde{Q} \coloneqq Q_d^{-1} \odot Q. \tag{8.38}$$

The attitude error was used in the model independent control law from [32] given by:

$$\boldsymbol{\tau} = -\alpha \tilde{\mathbf{q}} - \Gamma \boldsymbol{\Omega}, \tag{8.39}$$

where $\alpha \in \mathbb{R} > 0$ and $\Gamma \in \left\{ \mathbb{R}^{3\times 3} \mid \Gamma = \Gamma^T > 0 \right\}$. The proof of global asymptotic stability of eq. (8.39) is shown in [32]. The thrust and torque are then used to determine the propeller speeds through

$$\bar{\boldsymbol{\omega}} = M^{-1}\mathbb{T}, \tag{8.40}$$

from eq. (5.9), where $\bar{\boldsymbol{\omega}}$ is defined in eq. (5.7), M^{-1} is defined in eq. (5.8), and $\mathbb{T}$ is defined in eq. (5.5).

8.4 Experimental Results

Implementation results for a position hold command are presented here. To perform a position hold, the quadrotor was manually controlled to fly above its target and then the autonomous flight routine was initiated. The control gains used in the hovering experiment are shown in table 8.1.

TABLE 8.1: Control gains.

Parameter	Value
k_1	1
k_2	0.2
k_3	0.03125
k_4	0.00625
k_b	16
Δ	0.03
α	3.0
Γ	$I_3[0.625, 0.625, 0.5]^\top$
Q	$I_2 T$
K_p	$I_3[3.625, 3.625, 3.625]^\top$
K_v	$I_3[11.875, 11.875, 3.5]^\top$

For the following results, $\mathbf{p}^d = [0, 0, -1.5]^\top$. In fig. 8.6, the position can be seen to converge to the neighbourhood of the desired values. The x and y positions vary by $\pm 20cm$ with standard deviations of $8.5cm$. The altitude (y position) of the quadrotor converges to the set point after approximately one minute. The spikes that appear in the altitude were caused by transmission errors of the image frames. These errors could be eliminated by moving the image processing on board the quadrotor.

In fig. 8.7, the data spikes due to frame errors were removed to better show how the quadrotor's 2D positions. The 2D plots in fig. 8.7 are each shown as viewed from the negative remaining axis,

i.e. the X-Y plot (top left) is viewed from the negative z-axis, the X-Z plot (top right) is viewed from the negative y-axis, and the Y-Z plot (bottom left) is viewed from the negative x-axis.

In fig. 8.8, there are velocity spikes similar to fig. 8.6 which are also due to frame errors. The velocity of the quadrotor is otherwise consistent.

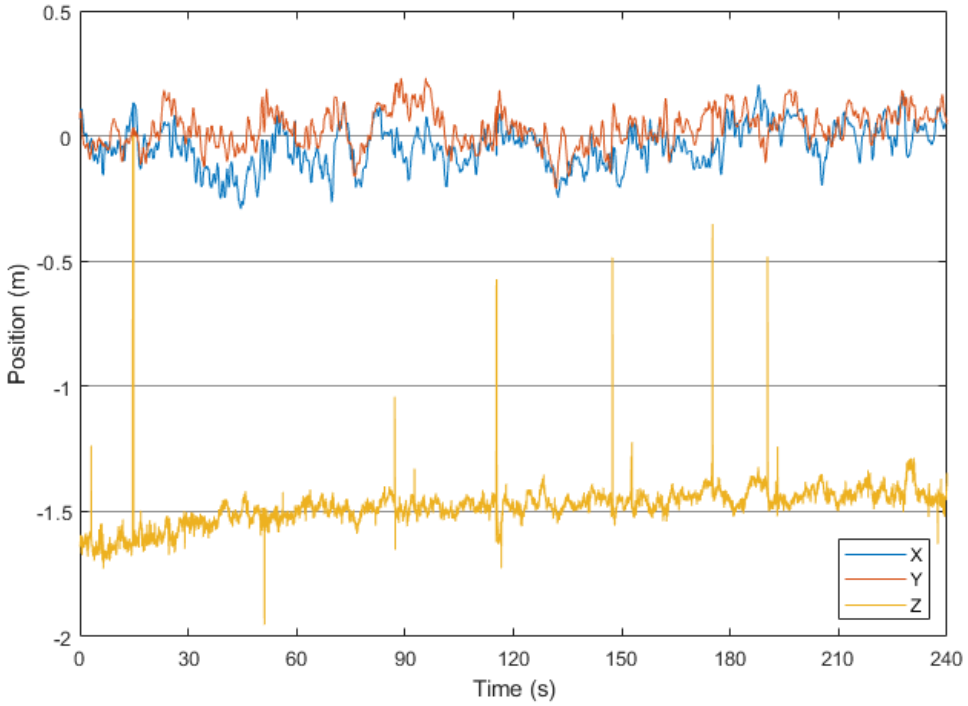


FIGURE 8.6: Position of the quadrotor in each Cartesian axis vs. time expressed in $\mathcal{I}$.

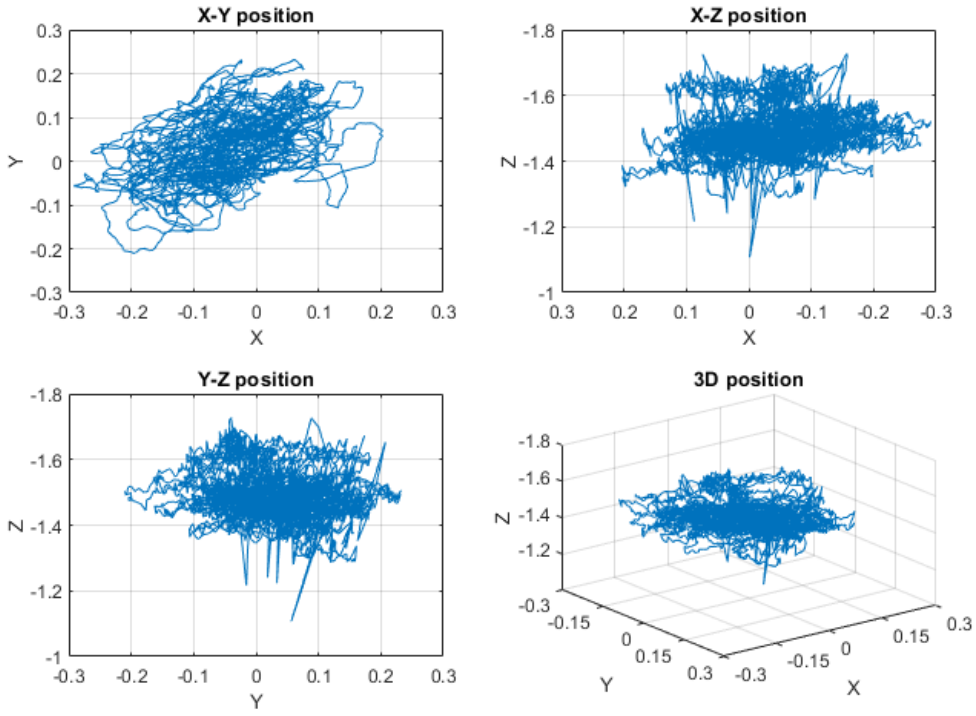


FIGURE 8.7: 3D position from autonomous flight using stereo vision for position estimation in metres.

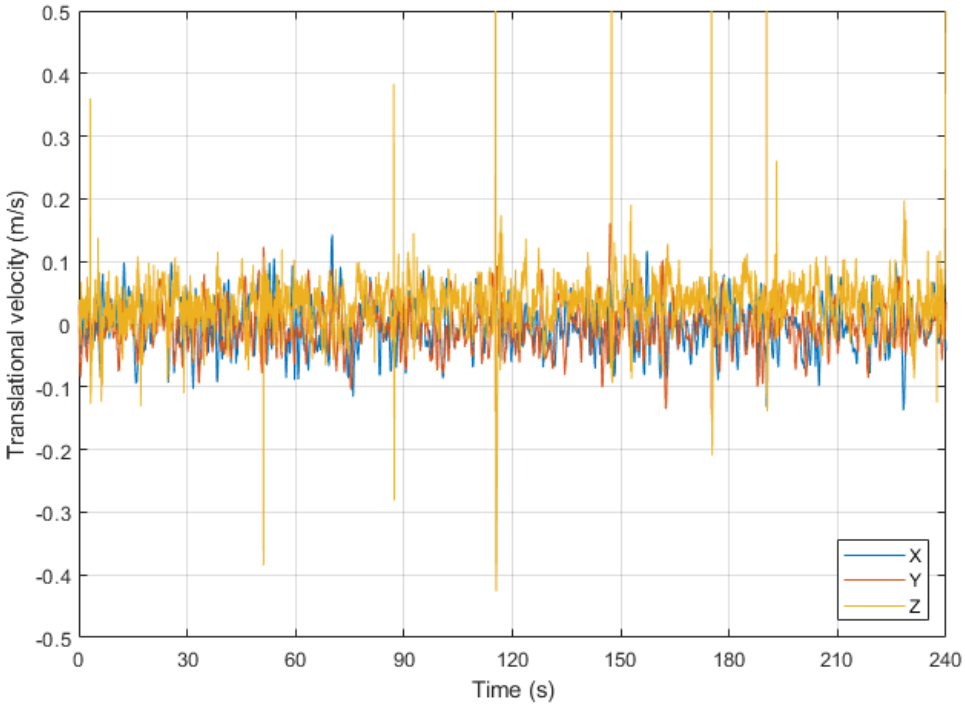


FIGURE 8.8: Velocity of the quadrotor in each Cartesian axis vs. time expressed in $\mathcal{I}$.

Chapter 9

Conclusion and Future Work

In this thesis a stereo computer vision system for position determination and control was designed and implemented. A position tracking control scheme which relied on IMU and stereo computer vision measurements was successfully implemented on a quadrotor UAV.

The attitude estimation technique used in this work was a complementary filter; it was chosen because it is a computationally efficient method of estimating the nonlinear dynamics. The "conditioned observer" complementary filter was chosen due to its decoupling of the roll and pitch dynamics from its yaw dynamics and for its anti-integral windup capabilities.

A Kalman filter based linear position and velocity estimator was implemented using stereo computer vision and accelerometer data. The position was estimated using stereo matching and triangulation to determine the depth and contour matching to determine the horizontal position. The covariance values for the Kalman filter were determined empirically.

Recommendations for continuation of this work are:

1. If a base station computer is desired, a more powerful GPU should be used to increase the speed of image processing. This would also allow higher resolution cameras to be used.
2. Implement a stereo vision system which does not require a base station computer so as to synchronise the position and IMU measurements and to ensure that image frames are not lost or distorted due to their wireless transmission.
3. A method that detects and tracks a general feature should be used to increase the robustness of position estimation and reduce the computational load caused by image analysis.
4. A smaller drone would be desirable to reduce risk to lab personnel, to reduce the inertial of the drone allowing for a more agile platform, and to reduce the air movement in the lab creating less environmental disturbance.
5. A frame whose landing legs are affixed to the central structure and/or motor arms with less play should be used; this would prevent hard landings from changing the resting orientation of the centrally mounted sensors.

Bibliography

[1] Y. Huang, W. C. Hoffmann, Y. Lan, W. Wu, and B. K. Fritz, "Development of a spray system for an unmanned aerial vehicle platform", *Appl. Eng. Agriculture*, vol. 25, no. 6, pp. 803–809, 2009.

[2] R. L. Finn and D. Wright, "Unmanned aircraft systems: Surveillance, ethics and privacy in civil applications", *Comput. Law Security Review*, vol. 28, no. 2, pp. 184–194, 2012.

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

[4] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin, "The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC)", in *Proc. 23rd Digit. Avionics Syst. Conf.*, IEEE, vol. 2, 2004, 12.E.

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

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

[7] F. L. Markley, "Attitude determination using vector observations and the singular value decomposition", *J. Astronautical Sci.*, vol. 36, no. 3, pp. 245–258, 1988.

[8] G. Wahba, "A least squares estimate of satellite attitude", *SIAM Review*, vol. 7, no. 3, pp. 409–409, 1965.

[9] R. E. Kalman, "A new approach to linear filtering and prediction problems", *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[10] J. L. Farrell, "Attitude determination by Kalman filtering", *Automatica*, vol. 6, no. 3, pp. 419–430, 1970.

[11] J. L. Crassidis, F. L. Markley, and Y. Cheng, "Survey of nonlinear attitude estimation methods", *J. Guidance Contr. Dynamics*, vol. 30, no. 1, pp. 12–28, 2007.

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

[13] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Nonlinear complementary filters on the special orthogonal group", *IEEE Trans. Automat. Contr.*, vol. 53, no. 5, pp. 1203–1217, 2008.

[14] D. Lee, H. J. Kim, and S. Sastry, "Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter", *Int. J. Contr. Automat. Syst.*, vol. 7, no. 3, pp. 419–428, 2009.

[15] M.-D. Hua, "Attitude estimation for accelerated vehicles using GPS/INS measurements", *Contr. Eng. Practice*, vol. 18, no. 7, pp. 723–732, 2010.

[16] P. Martin and E. Salaün, "Design and implementation of a low-cost observer-based attitude and heading reference system", *Contr. Eng. Practice*, vol. 18, no. 7, pp. 712–722, 2010.

[17] A. Roberts and A. Tayebi, "On the attitude estimation of accelerating rigid-bodies using GPS and IMU measurements", in *2011 50th IEEE Conf. Decision Contr. and European Contr. Conf.*, IEEE, 2011, pp. 8088–8093.

[18] M.-D. Hua, K. Rudin, G. Ducard, T. Hamel, and R. Mahony, "Nonlinear attitude estimation with measurement decoupling and anti-windup gyro-bias compensation", *IFAC Proc. Volumes*, vol. 44, no. 1, pp. 2972–2978, 2011.

[19] M.-D. Hua, G. Ducard, T. Hamel, and R. Mahony, "Introduction to nonlinear attitude estimation for aerial robotic systems", *Aerospace Lab*, AL08–04, 2014.

[20] M.-D. Hua, G. Ducard, T. Hamel, R. Mahony, and K. Rudin, "Implementation of a nonlinear attitude estimator for aerial robotic vehicles", *IEEE Trans. Contr. Syst. Technol.*, vol. 22, no. 1, pp. 201–213, 2014.

[21] W. E. Frye and E. V. B. Stearns, "Stabilization and attitude control of satellite vehicles", *ARS J.*, vol. 29, no. 12, pp. 927–931, Dec. 1959.

[22] R. H. Olds, "Attitude control and station keeping of a communication satellite in a 24-hour orbit", *AIAA J.*, vol. 1, no. 4, pp. 852–858, Apr. 1963.

[23] H. B. Kennedy, "A Gyro Momentum Exchange Device for Space Vehicle Attitude Control", *AIAA J.*, vol. 1, no. 5, pp. 1110–1118, May 1963.

[24] J. R. Wertz, *Spacecraft attitude determination and control*. Springer Science & Business Media, 2012, vol. 73.

[25] J. Li and Y. Li, "Dynamic analysis and PID control for a quadrotor", in *2011 IEEE Int. Conf. Mechatron. Automat.*, IEEE, 2011, pp. 573–578.

[26] I. D. Cowling, J. F. Whidborne, and A. K. Cooke, "Optimal trajectory planning and LQR control for a quadrotor UAV", in *Int. Conf. Contr.*, 2006.

[27] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The flying machine arena", *Mechatron.*, vol. 24, no. 1, pp. 41–54, 2014.

[28] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, "Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle", *J. Field Robot.*, vol. 33, no. 4, pp. 431–450, 2016.

[29] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor", in *Proc. 2005 IEEE Int. Conf. Robot. Automat.*, IEEE, 2005, pp. 2247–2252.

[30] A Benallegue, A Mokhtari, and L Fridman, "Feedback linearization and high order sliding mode observer for a quadrotor UAV", in *Proc. 2006 Int. Workshop Variable Structure Syst.*, IEEE, 2006, pp. 365–372.

[31] R. Mahony, T. Hamel, J. Trumpf, and C. Lageman, "Nonlinear attitude observers on SO (3) for complementary and compatible measurements: A theoretical study", in *Proc. 48th IEEE Conf. Decision Contr. held jointly with 2009 28th Chinese Contr. Conf.*, IEEE, 2009, pp. 6407–6412.

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

[33] E. Abbott and D. Powell, "Land-vehicle navigation using GPS", *Proc. IEEE*, vol. 87, no. 1, pp. 145–162, 1999.

[34] C. R. Carlson, J. C. Gerdes, and J. D. Powell, "Practical position and yaw rate estimation with GPS and differential wheelspeeds", in *Proc. AVEC 6th Int. Symp.*, 2002.

[35] Global Positioning System Wing. (Jul. 2010). Interface Specification IS-GPS-200 Revision E, [Online]. Available: https://media.defense.gov/2016/Jul/26/2001583113/-1/-1/1/IS%20GPS%20200E.PDF.

[36] J. L. Crowley and P. Reignier, "Asynchronous control of rotation and translation for a robot vehicle", *Robot. Autonomous Syst.*, vol. 10, no. 4, pp. 243–251, 1992.

[37] C. R. Carlson, J. C. Gerdes, and J. D. Powell, "Practical position and yaw rate estimation with GPS and differential wheelspeeds", in *Proc. AVEC 6th Int. Symp.*, 2002.

[38] F. Chenavier and J. L. Crowley, "Position estimation for a mobile robot using vision and odometry", in *Proc. 1992 IEEE Int. Conf. Robot. Automat.*, IEEE, 1992, pp. 2588–2593.

[39] J. L. Crowley, "World modeling and position estimation for a mobile robot using ultrasonic ranging", in *Proc., 1989 Int. Conf. Robot. Automat.*, IEEE, 1989, pp. 674–680.

[40] K. Yu and I. Oppermann, "Performance of UWB position estimation based on time-of-arrival measurements", in *2004 Int. Workshop Ultra Wideband Syst. held jointly with Conf. Ultra Wideband Syst. Technol.*, IEEE, 2004, pp. 400–404.

[41] S. Gezici, "A survey on wireless position estimation", *Wireless Personal Communications*, vol. 44, no. 3, pp. 263–282, 2008.

[42] S. Gupte, P. I. T. Mohandas, and J. M. Conrad, "A survey of quadrotor unmanned aerial vehicles", in *2012 Proc. IEEE Southeastcon*, IEEE, 2012, pp. 1–6.

[43] Z. Sedor, "UWB localization for autonomous indoor position control of VTOL UAVs", Master's thesis, Lakehead University, 2018.

[44] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation", *Computational Vision and Active Perception Laboratory, Fiskartorpsv*, vol. 15, p. 2002, 2002.

[45] A. J. Koivo and N. Houshangi, "Real-time vision feedback for servoing robotic manipulator with self-tuning controller", *IEEE Trans. Syst. Man Cybern.*, vol. 21, no. 1, pp. 134–142, 1991.

[46] J. Qian and J. Su, "Online estimation of image Jacobian matrix by Kalman-Bucy filter for uncalibrated stereo vision feedback", in *Proc. 2002 IEEE Int. Conf. Robot. Automat.*, IEEE, vol. 1, 2002, pp. 562–567.

[47] P. I. Corke, *Visual Control of Robots: High-Performance Visual Servoing*. Taunton, Somerset, UK: Research Studies Press Ltd., 1996.

[48] B. Cyganek and J. P. Siebert, *An introduction to 3D computer vision techniques and algorithms*. John Wiley & Sons, 2011.

[49] E. R. Davies, *Computer and machine vision: theory, algorithms, practicalities*. Academic Press, 2012, ch. 18 Image Transformations and Camera Calibration, pp. 478–503.

[50] L. Matthies, T. Kanade, and R. Szeliski, "Kalman filter-based algorithms for estimating depth from image sequences", *Int. J. Comput. Vision*, vol. 3, no. 3, pp. 209–238, 1989.

[51] S. Hutchinson, G. D. Hager, and P. I. Corke, "A tutorial on visual servo control", *IEEE Trans. Robot. Automat.*, vol. 12, no. 5, pp. 651–670, 1996.

[52] N. Hollinghurst and R. Cipolla, "Uncalibrated stereo hand-eye coordination", *Image Vision Computing*, vol. 12, no. 3, pp. 187–192, 1994.

[53] S. D. Jones, C. Andresen, and J. L. Crowley, "Appearance based process for visual navigation", in *Proc. 1997 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, vol. 2, 1997, pp. 551–557.

[54] E. Malis, "Survey of vision-based robot control", *ENSIETA European Naval Ship Design Short Course, Brest, France*, vol. 41, p. 46, 2002.

[55] T. Ohno, A. Ohya, and S. Yuta, "Autonomous navigation for mobile robots referring pre-recorded image sequence", in *Proc. 1996 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, vol. 2, 1996, pp. 672–679.

[56] A. Dev, B. Krose, and F. Groen, "Navigation of a mobile robot on the temporal development of the optic flow", in *Proc. 1997 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, vol. 2, 1997, pp. 558–563.

[57] A. Bernardino and J. Santos-Victor, "Visual behaviours for binocular tracking", *Robot. Autonomous Syst.*, vol. 25, no. 3-4, pp. 137–146, 1998.

[58] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses", *IEEE J. Robot. Automat.*, vol. 3, no. 4, pp. 323–344, 1987.

[59] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections", *Nature*, vol. 293, pp. 133–135, Sep. 1981.

[60] R. I. Hartley, "In defence of the 8-point algorithm", in *Proc. IEEE Int. Conf. Comput. Vision*, IEEE, 1995, pp. 1064–1070.

[61] R. I. Hartley, "In defense of the eight-point algorithm", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 6, pp. 580–593, Jun. 1997.

[62] Z. Zhang, "Determining the epipolar geometry and its uncertainty: A review", *Int. J. Comput. Vision*, vol. 27, no. 2, pp. 161–195, 1998.

[63] S. T. Barnard and M. A. Fischler, "Computational stereo", SRI International, Menlo Park, CA: Artificial Intelligence Center, Tech. Rep., 1982.

[64] H. P. Moravec, "Obstacle avoidance and navigation in the real world by a seeing robot rover", Robotics Inst., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep., Sep. 1980.

[65] J. F. Canny, "Finding edges and lines in images", Artif. Intell. Lab., MIT, Cambridge, MA, Tech. Rep., Jun. 1983.

[66] J. Canny, "A computational approach to edge detection", *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.

[67] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[68] M. Björkman, N. Bergström, and D. Kragic, "Detecting, segmenting and tracking unknown objects using multi-label MRF inference", *Comput. Vision Image Understanding*, vol. 118, pp. 111–127, 2014.

[69] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection", in *European Conf. Comp. Vision*, Springer, 2006, pp. 430–443.

[70] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)", *Comput. Vision Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[71] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: An efficient alternative to SIFT or SURF.", in *Int. Conf. Comput. Vision*, Citeseer, vol. 11, 2011, p. 2.

[72] B. K. Horn and B. G. Schunck, "Determining optical flow", *Artificial Intell.*, vol. 17, no. 1-3, pp. 185–203, 1981.

[73] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques", *Int. J. Comput. Vision*, vol. 12, no. 1, pp. 43–77, 1994.

[74] W. Hoff and N. Ahuja, "Surfaces from stereo: Integrating feature matching, disparity estimation, and contour detection", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 2, pp. 121–136, 1989.

[75] R. Mahony and T. Hamel, "Image-based visual servo control of aerial robotic systems using linear image features", *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 227–239, 2005.

[76] P. I. Corke, "Visual control of robot manipulators–a review", in *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, World Scientific, 1993, pp. 1–31.

[77] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, "Divergent stereo for robot navigation: Learning from bees", in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, IEEE, 1993, pp. 434–439.

[78] D. B. Gennery, "A stereo vision system for an autonomous vehicle.", in *Int. Joint Conf. Artificial Intell.*, vol. 2, 1977, p. 576.

[79] D.-J. Kim, R. Lovelett, and A. Behal, "Eye-in-hand stereo visual servoing of an assistive robot arm in unstructured environments", in *2009 IEEE Int. Conf. Robot. Automat.*, IEEE, 2009, pp. 2326–2331.

[80] O. Amidi, Y. Mesaki, and T. Kanade, "Research on an autonomous vision-guided helicopter", *AIAA J.*, 1994.

[81] O. Amidi, "An autonomous vision-guided helicopter", PhD thesis, Carnegie Mellon University, 1996.

[82] P. Corke, "An inertial and visual sensing system for a small autonomous helicopter", *J. Robot. Syst.*, vol. 21, no. 2, pp. 43–51, 2004.

[83] E. D. Dickmanns, "A general dynamic vision architecture for UGV and UAV", *Applied Intell.*, vol. 2, no. 3, pp. 251–270, 1992.

[84] O. Shakernia, R. Vidal, C. S. Sharp, Y. Ma, and S. Sastry, "Multiple view motion estimation and control for landing an unmanned aerial vehicle", in *Proc. 2002 IEEE Int. Conf. Robot. Automat.*, IEEE, vol. 3, 2002, pp. 2793–2798.

[85] J. M. Hintze, "Autonomous landing of a rotary unmanned aerial vehicle in a non-cooperative environment using machine vision", Master's thesis, Brigham Young University-Provo, 2004.

[86] M. Wang, "Attitude control of a quadrotor UAV: Experimental results", Master's thesis, Lakehead University, 2015.

[87] S. Hrabar, G. S. Sukhatme, P. Corke, K. Usher, and J. Roberts, "Combined optic-flow and stereo-based navigation of urban canyons for a UAV", in *2005 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2005, pp. 3309–3316.

[88] J. Byrne, M. Cosgrove, and R. Mehra, "Stereo based obstacle detection for an unmanned air vehicle", in *Proc. 2006 IEEE Int. Conf. Robot. Automat.*, IEEE, 2006, pp. 2830–2835.

[89] T Hamel and R Mahony, "Pure 2D visual servo control for a class of under-actuated dynamic systems", in *Proc. 2004 IEEE Int. Conf. Robot. Automat.*, IEEE, New Orleans, LA, USA, Apr. 2004.

[90] B. Herisse, F.-X. Russotto, T. Hamel, and R. Mahony, "Hovering flight and vertical landing control of a VTOL unmanned aerial vehicle using optical flow", in *2008 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, IEEE, 2008, pp. 801–806.

[91] M. D. Schuster, "A survey of attitude representations", *J. Astronautical Sci.*, vol. 41, no. 4, pp. 439–517, Oct. 1993.

[92] H. Schaub and J. L. Junkins, in *Analytical Mechanics of Space Systems*, 1st ed. Reston, VA: AIAA, 2003, ch. Rigid Body Kinematics, pp. 71–105.

[93] N. A. Chaturvedi, A. K. Sanyal, and N. H. McClamroch, "Rigid-body attitude control using rotation matrices for continuous singularity-free control laws", *IEEE Control. Syst. Mag.*, vol. 31, no. 3, pp. 30–51, Jun. 2011.

[94] T. Hamel, R. Mahony, R. Lozano, and J. Ostrowski, "Dynamic modelling and configuration stabilization for an X4-flyer", *IFAC Proc. Volumes*, vol. 35, no. 1, pp. 217–222, 2002.

[95] 3DRobotics. (Aug. 2013). Quad-C DIY Assembly Instructions.

[96] ArduPilot. (Jan. 2017). Archived:APM 2.5 and 2.6 Overview, [Online]. Available: http://ardupilot.org/copter/docs/.

[97] 3DRobotics. (Mar. 2012). A2830 Out Runner Brushless Motor Instruction, [Online]. Available: https://3dr.com/wp-content/uploads/2017/03/A2830-Out-Runner-Brushless-Motor-Instructions-1.pdf.

[98] M. H. Sadraey, *Aircraft Performance: An Engineering Approach*. CRC Press, Jan. 2017, ch. Engine Performance, p. 125.

[99] B. Becker, "Understanding propeller pitch", *Boating Life*, vol. 2, no. 6, p. 18, 1998.

[100] 3DRobotics. (Mar. 2017). ESC 20 Amp with SimonK | 3DR, [Online]. Available: https://3dr.com/support/articles/esc_20_amp_with_simonk/.

[101] APC Propellers. (Feb. 2016). APC Propeller Speed Specification, [Online]. Available: https://49u9ei2todm6vjegs45ceqp1-wpengine.netdna-ssl.com/wp-content/uploads/2017/06/APC-Propeller-RPM-Limits.pdf.

[102] Invensense. (Aug. 2013). MPU-6000 and MPU-6050 Product Specification Revision 3.4, [Online]. Available: https://store.invensense.com/datasheets/invensense/MPU-6050_DataSheet_V3%204.pdf.

[103] Honeywell. (Feb. 2013). 3-Axis Digital Compass IC HMC5883L, [Online]. Available: https://cdn-shop.adafruit.com/datasheets/HMC5883L_3-Axis_Digital_Compass_IC.pdf.

[104] Invensense. (Aug. 2013). MPU-6000 and MPU-6050 Register Map and Descriptions Revision 4.2, [Online]. Available: https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf.

[105] Flysky Technology co. (Jul. 2017). FS-TH9X User Manual, [Online]. Available: https://static1.squarespace.com/static/5bc852d6b9144934c40d499c/t/5c62387fe2c48311c8948a4a/1549941012543/FS-TH9X+User+Manual+20170721.pdf.

[106] Digi International. (Sep. 2009). Datasheet: XBee/Xbee-PRO RF Modules, [Online]. Available: https://www.sparkfun.com/datasheets/Wireless/Zigbee/XBee-Datasheet.pdf.

[107] Digi International. (2018). XStick - USB to Xbee Wireless Pan Adapter For Laptops Datasheet, [Online]. Available: https://www.digi.com/pdf/ds_xstick.pdf.

[108] SparkFun Electronics. (Mar. 2013). Datasheet: 32KM, [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/LightImaging/32KM_spec.docx.pdf.

[109] PixelPlus Co., Ltd. (2010). Datasheet: PC1089K, [Online]. Available: http://www.alliedsens.com/uploads/2017/03/20/PC1089K.pdf.

[110] BosCam. (Feb. 2012). Datasheet: TS 351, [Online]. Available: http://shinwa2000.com/downloads/Boscam%20RC805%20-%205.8Ghz%208%20Channel%20AV%20Receiver.pdf.

[111] StarTech. (Jan. 2019). Technical specifications: S-Video/Composite to USB Video Capture Cable w/ TWAIN and Mac Support, [Online]. Available: https://www.startech.

com/media/products/SVID2USB23/PDFs/SVID2USB23_Datasheet.pdf.

[112] C. C. Foster and G. H. Elkaim, "Extension of a two-step calibration methodology to include nonorthogonal sensor axes", *IEEE Trans. Aerosp. Electron. Syst.*, vol. 44, no. 3, pp. 1070–1078, 2008.

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

[114] (Feb. 2017). Magnetic field calculator, [Online]. Available: https://geomag.nrcan.gc.ca/calc/mfcal-en.php.

[115] B. Cyganek and J. P. Siebert, *An introduction to 3D computer vision techniques and algorithms*. John Wiley & Sons, 2011.

[116] OpenCV. (Jul. 2011). initUndistortRectifyMap function.

[117] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following", *Comput. Vision Graphics Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

[118] M.-K. Hu, "Visual pattern recognition by moment invariants", *IRE Trans. Inf. Theory*, vol. 8, no. 2, pp. 179–187, 1962.

[119] A. Kelly, "A 3D state space formulation of a navigation kalman filter for autonomous vehicles", Robotics Inst., Carnegie-Mellon Univ., Pittsburgh, PA, Tech. Rep., May 1994.