State Estimation in Unmanned Aerial Vehicles: Theoretical Approaches for Increasing Accuracy

by
Nicholas Johan Minnaar

Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering (Mechatronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Dr Willie Smit

December 2017

Declaration

By submitting this report electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: December 2017 .....

Abstract

The utility of unmanned aerial vehicles (UAVs) are enhanced by their ability to navigate accurately. Unfortunately, it is not possible to capture the states which describe a UAV's position and orientation in a measurement system without some degree of uncertainty. The work presented here resolves this issue by developing strategies for generating accurate navigational outputs from the measurements taken on board an UAV.

The three strategies that are ultimately developed address fundamental misnomers that are related to the processing of navigational measurement data. The first strategy relates to the choice of state variable when exogenous signals are available in the UAV's measurement matrix. It is suggested that these signals be used to update error estimates related to other sensors instead of acting as observations of internal system states. The second strategy addresses a common theoretical conflict which arises in UAV applications where accelerometers are used for orientation observations. It is suggested that redundant accelerometers be added to the measurement system and a strategy for distilling orientation rate information from this redundant information is also provided. The final strategy which is provided addresses the numerical errors which slowly disintegrate the orientation estimates of a UAV. It is suggested that a singular value decomposition be leveraged to restore the welfare of the orientation estimates. All the proposals which are made improve the navigational performance of the UAV and are validated through simulation.

Uittreksel

Die nut, aanwending en gebruik van Hommeltuie word verbeter met hul direkte vermoë om meer akkuraat te kan navigeer. Ongelukkig is dit nie moontlik om die staat van die veranderlikes in terme van posisie en orientasie te bepaal sonder ‘n mate van metingsonsekerheid nie. Hierdie voorgestelde werk bied ‘n oplossing deur drie verskillende strategie te ontwikkel om meer akkurate navigasie data te genereer uit die Hommeltuig se bestaande, aanboord metingsstelsel.

Die drie strategieë behandel die verwerkings metodes van die navigasie metingsdata wat die foute veroorsaak gedurende die prosesering, vooruitskatting en bepaling van staatskattings. Die eerste strategie hou verband met die voorkeur van veranderlike state sodra eksogene seine ook teenwoordig is in die meetingsmatriks van die Hommeltuig. Daar word voorgestel dat hierdie seine aangewend word om foutberamings (wat verband hou met die insete van ander sensors) eerder voordurend op te dateer in plaas van om die interne staat veranderlikes op te dateer. Die tweede strategie spreek ‘n algemene teoretiese teenstrydigheid aan by Hommeltuig aanwendings wat onstaan sodra versnelingsmeters gebruik word vir orienteringswaarnemings. Dit word voorgestel dat addisionele versnellingsmeters by die meetstelsel gevoeg word en ‘n strategie om die oortollige inligting te gebruik om orienteringsberamings te bereken. Die finale gegewe strategie spreek die numeriese foute aan wat die orientasie beramings van die Hommeltuig stelselmatig laat disintegreer. Daar word voorgestel dat ‘n enkelvoudige afgeleide waarde (“ Singular Value Decomposition”) aangewend word om die welstand van die orientasieberamings te herstel. Elk van hierdie drie unieke voorstelle verbeter die akkuraatheid van navigasie vermoe van ‘n Hommeltuig en is deur middel van simmulasie bewys.

Acknowledgments

First and foremost, I would like to thank my supervisor, Dr WJ Smit, for his guidance and patience throughout this project. I am also thankful to the National Research Foundation for their financial support. Thank you to my two wonderful parents for equipping me with everything I would ever need in life to make a success. I would also like to thank my friends, their support has been greatly appreciated. Lastly, Natasha Solari, without your unending love and support - this thesis would not have been possible.

Dedications

In this thesis, as in life, I have learnt a lot about the world around me from the models I have used which describe it. So I dedicate this work to my greatest [role] model, my greatest teacher, from whom I have learned so much - Dad, this one is for you.

Contents

Declaration i

Abstract ii

Uittreksel iii

Acknowledgments iv

Dedications v

Contents vi

List of Figures ix

Nomenclature xii

1 Introduction 1
- 1.1 Project Motivation . . . 2
- 1.2 Aim and Objectives . . . 5
- 1.3 Scope . . . 6
- 1.4 Methodology . . . 7
- 1.5 Layout . . . 8

2 Literature Review 9
- 2.1 Dead Reckoning . . . 9
- 2.2 Position Fixing . . . 11
- 2.3 Integrated Navigation . . . 12
  - 2.3.1 Traditional Kalman Filter . . . 12
  - 2.3.2 Extended Kalman Filter (EKF) . . . 13
  - 2.3.3 Unscented Kalman Filtering (UKF) . . . 13
  - 2.3.4 Applications of other filters . . . 14
- 2.4 Theoretical Conflicts . . . 14

2.5 Literature Dissection 16
2.6 Conclusion 17

3 A Virtual Test Bench **18**
3.1 Co-Simulation Architecture 20
3.2 State Variable Conventions 22
3.2.1 The Rotation Matrix 24
3.2.2 The Direction Cosine Matrix (DCM) 25
3.2.3 Euler Angles 27
3.3 The Plant 29
3.3.1 Linear Dynamics 33
3.3.2 Rotational Dynamics 33
3.4 The State Estimator 35
3.4.1 A Stochastic Approach 35
3.4.2 A Deterministic Approach 36
3.4.3 The Kalman Filter 38
3.4.4 Derivation of Kalman Filter Equations 39
3.4.5 Steps of the Kalman Filter Algorithm 41
3.5 Conclusion 44

4 The Resolution of Theoretical Conflicts **45**
4.1 Measurement Model 46
4.2 Algorithm for solving Angular Rate 48
4.3 Conclusion 51

5 Reducing Numerical Errors **53**
5.1 Kinematics of Rotational Motion 54
5.2 Orthonormalization 56
5.3 The Singular Value Decomposition 57
5.4 Conclusion 61

6 Results and Discussion **63**
6.1 Strategies for the use of Exogenous Signal based Sensors . . . 63
6.2 A Theoretical Conflicts 67
6.3 Numerical Errors 70

7 Conclusions **76**
7.1 General Conclusions 77
7.2 The Utilization of Exogenous Sensor Signals 78
7.3 The Resolution of a Theoretical Conflict 78
7.4 The Preservation of Orthogonality 79

7.5 Difficulties and Shortfalls 80
7.6 Recommendations for Future Work 80

Appendices **81**

A MotionSolve Brochure **82**

B MotionView Brochure **85**

C Activate Brochure **88**

D Co-simulation Technical Documentation **91**

E Linear Algebra Theorem **96**

F Force Assumptions **97**

G Controller Structure **101**

List of References **103**

List of Figures

1.1 An unmanned aerial vehicle 2
1.2 Four tiers that constitute project motive 3
1.3 Diagram showing operating principle of concentrated solar power using heliostat mirrors 4
1.4 A novel heliostat calibration concept proposed by STERG members 5
1.5 The three main tiers of this thesis 8

2.1 A typical navigation system 10

3.1 Chapter 3 is represented by the darkened section of the image and forms the first of the three main pillars which make up the thesis. 18
3.2 Chapter 3 is modularised into the sections shown. 19
3.3 The MotionView preprocessor user interface. 21
3.4 High level schematic model of simulations performed in this thesis. 22
3.5 An illustration of the co-simulation concept. 22
3.6 North East Down frame of reference 23
3.7 A 3-Rotation of the e_1 unit vector from the standard basis of $\mathbb{R}^3$ 24
3.8 A 3-Rotation of the the standard basis of $\mathbb{R}^3$ 26
3.9 A 3-2-1 rotation sequence 28
3.10 Body-fixed coordinate frame 30
3.11 A free body diagram for a quadcopter 31
3.12 UAV Mechanics 32
3.13 Deterministic Estimator 37
3.14 Kalman Filter 43

4.1 A depiction of the context and content of Chapter 4 46
4.2 General Acceleration Coordinate Frames 47
4.3 Kalman Filter Structure with proposed redundant accelerometer scheme 51

5.1 Chapter 5 is represented by the darkened section of the image and forms the third of the three main pillars which make up the thesis. . . . 53
5.2 The corresponding coordinates of a point in a coordinate frame and an infintesimally rotated frame . . . 55
5.3 Discrete coordinates which have been wrapped with a surface to form a randomly oriented sphere . . . 58
5.4 A transformation of the surface wrapped discrete coordinates that represent a unit sphere . . . 59
5.5 Singular Value Decomposition of matrix (square and real with positive determinant) transformation . . . 60
5.6 The singular values of the transformation matrix are forced to unity when orthogonalizing it according to equation 5.3.1 . . . 62

6.1 Section 6.1 utilizes the theory presented Chapter 3 to present strategies for utilizing exogenous sensors . . . 64
6.2 A schematic representing the simulated sampling scheme of the sensors . . . 65
6.3 Altitude signals for the UAV when given a step reference input (full order estimator) . . . 65
6.4 Altitude signals for the UAV when given a step reference input (this estimator structure includes an error state) . . . 66
6.5 Altitude Bias signals for the UAV when given a step reference input (this estimator structure includes an error state) . . . 67
6.6 Section 6.2 utilizes the theory presented Chapter 4 to address errors related to theoretical conflicts . . . 68
6.7 The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion . . . 69
6.8 Section 6.3 utilizes the theory presented Chapter 5 to address errors related to numerical error . . . 70
6.9 The ground truth data of an analytical solution for precession motion . . . 71
6.10 The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion . . . 73
6.11 The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion (25Hz propagation updates) . . . 74

F.1 Blade Flapping . . . 98

G.1 The structure of the UAV controller 102

Nomenclature

Constants

g	9.81	m/s^2
m	2.26	kg

Variables

x	Positional State	m
y	Positional State	m
z	Altitude	m
$\dot{x}$	Linear Velocity State	m/s
$\dot{y}$	Linear Velocity State	m/s
$\dot{z}$	Altitude Velocity	m/s
ψ	Roll	rad
ϕ	Pitch	rad
θ	Yaw	rad
$\dot{\psi}$	Roll Rate	rad/s
$\dot{\phi}$	Pitch Rate	rad/s
$\dot{\theta}$	Yaw Rate	rad/s
ω	Angular Rate	rad/s
a	Linear Acceleration	m/s^2
$\dot{\omega}$	Angular Acceleration	rad/s^2
f	Force	N
M	Moment	Nm

Vectors and Tensors

e	Basis Vector
b	Basis Vector
v	Process Noise Vector
w	Unmodeled Dynamic Input Vector
y	Output Vector
z	Measurement Vector
A	Feedback Gain
C	Cosine Matrix
F	Continuous System Model
G	Input Matrix
H	Measurement Model
I	Inertia Tensor
I	Unity Matrix
K	Kalman Filter Gain
L	Angular Momentum
M	Matrix Ricatti Equation
P	Error Covariance Matrix
P	Coordinate Matrix
Q	Noise Covariance Matrix
R	Rotation Matrix
R	Measurement Noise Co variance
R	Linear Acceleration
U	Left Singular Vectors
V	Right Singular Vectors
X	Measurement Matrix
Ψ	Discrete System Model
ρ	Positional Vector with reference to centre of mass
ω	Concatenation of three orthogonal angular rates
$\sum$	Singular Value Matrix
$\Delta\vartheta$	Infinitesimal Direction Cosine Matrix

Subscripts

1	Body Aligned Axis
1	Unmodeled Dynamics
2	Body Transverse Axis
3	Body Vertical Axis
k	Current Epoch
$k-1$	Previous Epoch
m	Measured
x	North Direction
y	East Direction
z	Vertical Direction
T	Total
ψ	Roll Axis
ϕ	Pitch Axis
θ	Yaw Axis

Superscripts

T	Transpose
$\hat{}$	Estimated
$+$	Updated
$-$	Propagated (but not yet updated)

Acronyms

AT	Aerial Triangulation
CAD	Computer Aided Design
CSP	Concentrated Solar Power
CV	Computer Vision
DCM	Direct Cosine Matrix
EKF	Extended Kalman Filter
FBD	Free Body Diagram
GPS	Global Positioning System
IMU	Inertial Measurement Unit
MS	MotionSolve
MV	MotionView
NED	North East Down
RTK	Real-Time Kinematic
SDRE	State-Dependent Riccati Equation
SN	Satellite Navigation
STERG	Solar Thermal Energy Research Group
SVD	Singular Value Decomposition
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter

Chapter 1

Introduction

"To measure is to know" – this is the mandate of famed physicist and engineer, Lord Kelvin. The essential idea behind this mandate is that if we can sense some quantity associated with a physical entity, then we may gain some objective knowledge about that entity. This idea is not new in science nor engineering, but still presents us with the same challenge as it did to all those scientists/ engineers that have gone before us – it says nothing of quality. In some simple cases, the intuitive approach to deal with this challenge is adequate: take high quality measurements to gain high fidelity knowledge about the entity you are sensing. In most other cases, as we will see, a deeper understanding of the mapping between measurement data and objective knowledge is necessary to improve the associated quality.

The mechanism which maps the measured quantity into objective knowledge is usually simple. For example, we may formulate an objective knowledge about the amount of matter contained in a physical entity by weighing it. The value that the scale will register varies linearly with the amount of matter contained i.e. if object A weighs twice as much as object B, then it also contains double the amount of matter. The mapping mechanism is not always linear as it is in this example though, and sometimes it is not even clear what quantities we should measure to gain a suitable objective knowledge of the physical entity which we are studying.

Unmanned aerial vehicles (UAVs) such as the one shown in Figure 1.1 are physical entities which are difficult to describe objectively, specifically in terms of their motion. They are rigid bodies that are fully described by six quantities which relate to their position and orientation in space. We would think that it should be simple to measure these six quantities, but as we will see, it is not always possible to do this uniquely, accurately or directly. In addition, our knowledge of the UAV does not vary linearly with the measurements we take on it, and we must exploit our knowledge of the

very mechanisms which link these quantities to correct for the previously mentioned contingencies. Thus, the hunt for high fidelity objective knowledge concerning the motion of a UAV forms the central theme of this thesis.

Figure 1.1: An unmanned aerial vehicle (Minnaar, 2015)

1.1 Project Motivation

The motivation for this project is built upon four tiers, shown in Figure 1.2. This section will begin at the top tier which, at first, is seemingly disparate from the main theme of the thesis [1]. The lowest tier on the other hand, aligns perfectly with the main theme, and is discussed last. The second and third tiers of the motivation will make the connection, and will hence appear in the middle of this section. It will be helpful to refer back to Figure 1.2 in order to follow the motivation, as the layout of this section corresponds with the layout of the tiers in the figure.

UAVs are a class of aircraft which are remotely piloted and were an important technological innovation which now serves multiple markets. They

[1] Central theme of this thesis: The hunt for high fidelity objective knowledge concerning the motion of a UAV

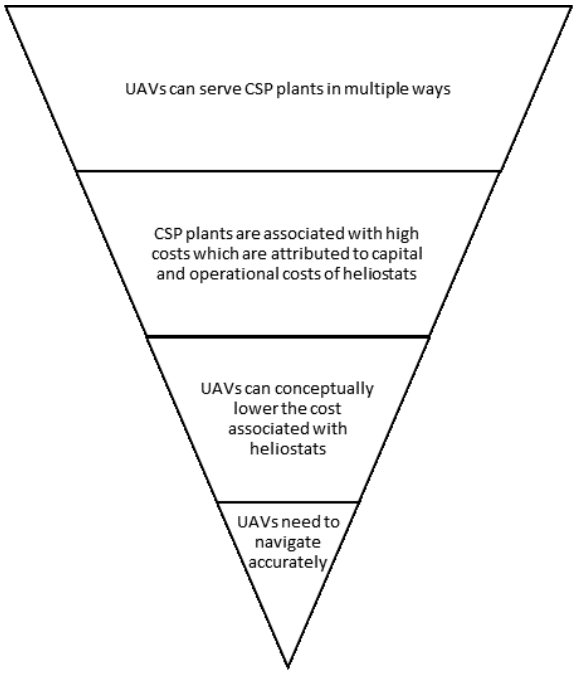


Figure 1.2: Four tiers that constitute project motive

can be found in photography, aerial surveying, surveillance, search-and-rescue assistance, and conservation patrolling to name but a few conventional applications. In addition, members of the Solar Thermal Energy Research Group (STERG) have identified a need for UAVs in the field of concentrated solar power (CSP). Specifically, tasks that can be undertaken by UAVs in CSP plants include monitoring (security and data logging), maintenance (cleaning and scheduling), and heliostat control (calibrating and realigning heliostat mirrors). The task of heliostat control, in particular, has potential to greatly impact the cost associated with a CSP plant.

To understand the potential economic benefits that UAVs may bring to CSP, we must first understand the most basic ingredient of a CSP plant, the heliostat. Heliostats are the mirrors that reflect the sun's rays to a central point, where heat is converted to electrical power through a heat engine (see Figure 1.3). The problem with heliostats is that they lose their direction as time moves on due to environmental factors. Every so often, the heliostats are recalibrated so that they are once again directing rays towards the central receiver. This recalibration process is inevitable to some degree, but can be performed less frequently when the heliostats are built to drift less rapidly. Unfortunately, these constructions come at a higher manufacturing cost (Lock, 2016), so the operational cost of frequent recalibration has been

replaced by a larger capital cost in building the CSP plant. The implication of harnessing the suns energy using heliostats is thus that either a large capital cost or a high operational cost must be incurred.

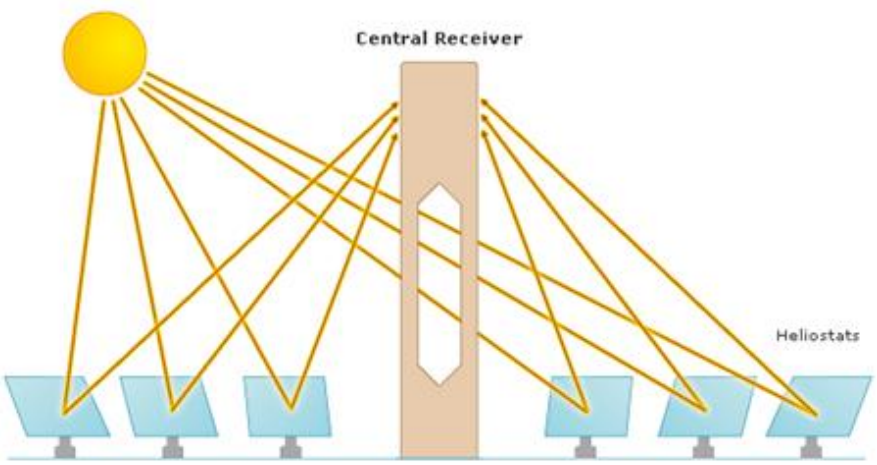


Figure 1.3: Diagram showing operating principle of concentrated solar power using heliostat mirrors (Abengoa Solar, 2012)

STERG members have conceptualized a heliostat calibration method which could be undertaken autonomously by UAVs. This method could potentially improve the currently achievable recalibration frequencies required for each heliostat. Since the heliostats would receive frequent recalibration, they would not need to be manufactured precisely, and this serves to drive down the manufacturing costs of the heliostats (and hence, the capital cost of a CSP plant [2]) without increasing the operational costs of recalibration. Figure 1.4 represents the calibration method in concept.

The suncopter project, which is a branch of research that falls under the Solar Thermal Energy Research Group (2015), consists of researchers that are working to make the utilisation of UAVs in heliostat calibration viable. Although UAVs have been levergaed in CSP plants before (UAS Vision, 2016), they have not been utilised for the same task which they are envisaged to take on here – heliostat calibration. Considering the work which has already been done within the group will provide context for the motivation of this research.

[2]The capital costs of the UAVs themselves would be recuperated by the manufacturing costs of heliostats because of the sheer volume of heliostats which are needed in a CSP plant.

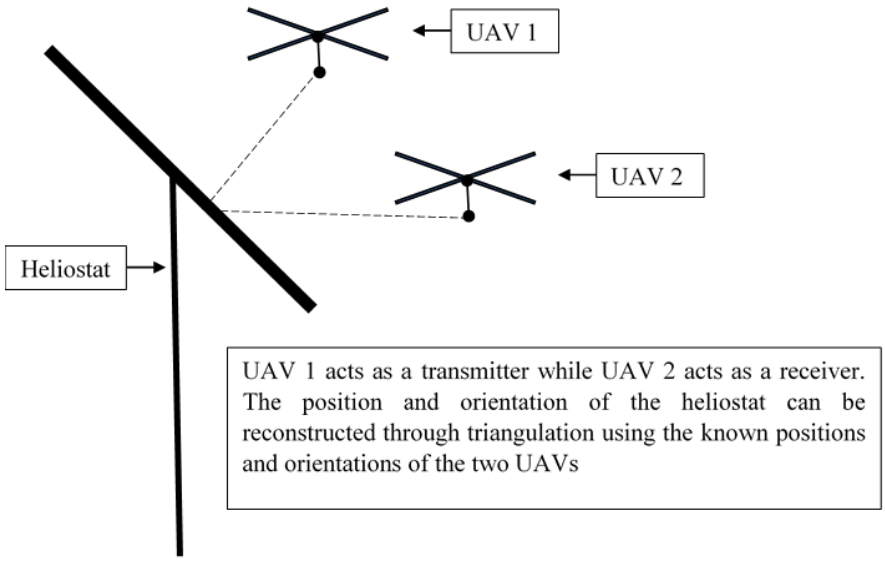


Figure 1.4: A novel heliostat calibration concept proposed by STERG members

Starting in 2015, one researcher studied the relationships between errors in heliostat calibration and their associated tracking accuracy (Zheng, 2017). This is a fundamental starting point for the development of a new heliostat calibration methodology. From this methodology, which would rely heavily on UAVs for automation, it will be possible to specify the required navigational accuracy. Lock (2016) developed a computer vision (CV) system which could augment an existing UAV in anticipation that its current navigational accuracy would not be good enough to meet the criteria of this as-yet undeveloped methodology. His research was conducted in tandem with the work done by Zheng, and while the performance of the CV system was good, some practical issues limited its operational range. The current research enters the picture at this point, with the motive of producing strategies for increasing the navigational accuracy of a UAV.

1.2 Aim and Objectives

Recall from the introduction that the central theme for this thesis is *the hunt for high fidelity knowledge concerning the motion of UAVs*. This theme rests on the assumption that UAVs do not currently navigate accurately enough, and the motivation for this was provided in Section 1.1. The aim of this research is now clear and follows the motive of the project: *to find strategies*

for increasing the navigational accuracy of a UAV [3]. The aim will be achieved by meeting the following research objectives:

1. Identify the main causes of inaccuracy in UAV navigation using available literature.
2. Identify solutions which address the inaccuracies identified in the second objective (again using available literature).
3. Propose solutions which address inaccuracies that have been identified in the first objective but which have not been adequately addressed in the second objective.
4. Build a virtual test bench (leveraging available simulation technology) which can be manipulated in order to test the strategies which will have been proposed in the third objective.
5. Assess the effectiveness of the proposed strategies in increasing the navigational accuracy of the virtual UAV.

1.3 Scope

The term "navigation" encompasses two concepts. The first one relates to the determination of a vehicle's position, orientation, velocity, and angular velocity. The second navigation concept, on the other hand, refers to the planning and maintenance strategies which would pilot a vehicle from one point to another (Groves, 2013). This thesis is concerned only with the first concept, and a replacement of terminology will be used to reflect this scope throughout the remainder of the thesis. Specifically, since the position, orientation, velocity, and angular velocity of a vehicle is also referred to as its "state", the term "state estimation" should be understood to replace "navigation" in the sense described by the first concept which was provided. The control system, obstacle avoidance, and path planning components of a UAV navigation system all fall outside of the scope of this thesis. It is noted, however, that some of these aspects have already been addressed by researchers within STERG (Coetzee, 2017; van Breda, 2016).

Furthermore, it is noted that this thesis is mostly theoretical, and that the practical implementation of the proposed strategies in a physical UAV falls outside the scope of the project. Instead, simulations are leveraged

[3] The term "navigational accuracy" is broad, and the scope has thus been defined in Section 1.3.

to assess and validate the proposals which have been reasoned based on the relevant theory, and the practical implementation is left as a task for future researchers to carry out. Lastly, the research is limited to the class of UAVs known as quadcopters. These quadcopters are defined by their method of actuation, which consists of four motor/rotor pairs that are fixed along a common plane and produce thrust which is aligned. The magnitude of the thrust resulting from motor/rotor pairs are strategically imbalanced to cause moments about the UAV's center of mass, which results in rotation and subsequent translation of the aircraft (an image of a quadcopter was provided in Figure 1.1).

1.4 Methodology

The methodology for achieving the overarching aim of the project (to find strategies for increasing the navigational accuracy of a UAV) will commence with a survey of the literature (Chapter 2), from which current barriers regarding navigational accuracy will be identified. It shall be shown that solutions have been developed to address some but not all of these barriers. Subsequent chapters (Chapters 3, 4 and 5) will provide solutions for the outstanding barriers and the necessary theory will also be developed.

The utility of the proposed solutions will be evaluated using simple and highly idealised simulation tests that allows for all variables to be observed and controlled (the results of these tests will be provided in Chapter 6). Any effects that are observed in the navigation solution would have a derivable cause in such an auditable environment. Conversely, there are effects that would confound the effects of observable/controllable variables if elaborate experimental tests were performed since not all variables could be knowingly accounted for. This confounding phenomena would make it difficult to separate cause and effect since elaborate tests include many root causes that are associated with similar effects and which are propagated through the system and ultimately mixed.

1.5 Layout

The pertinent chapters of this thesis will contain the image shown in Figure 1.5, with a single one of its elements highlighted. The highlighted elements serve as bookmarks and serve to enrich the readers insight concerning the context of that particular chapter within the thesis. The thesis will first work its way through the top three tiers of the figure before eventually addressing the questions which arise at the bottom of each of the tiers in Chapter 6. The thesis does also contain two chapters which do not reference the figure at all because the information provided by the chapters only provide context or conclusions that support the main tiers of the project. The literature review and conclusion chapters thus do not contain the figure, and are sand wedged on either side of the chapters that make up the figure.

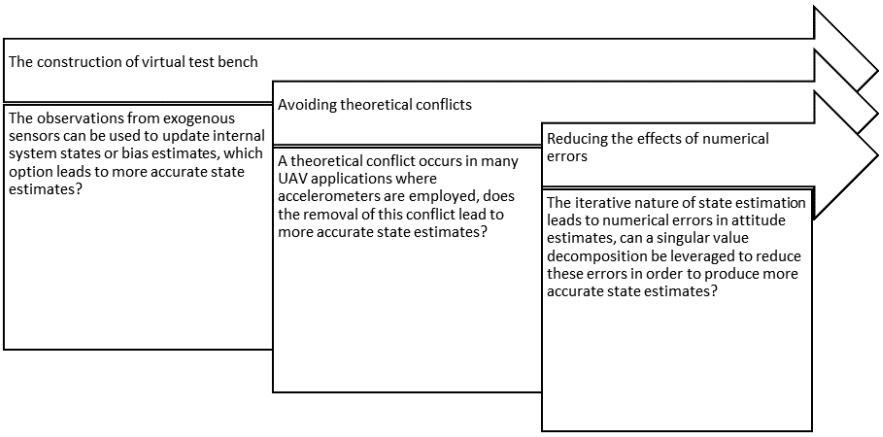


Figure 1.5: The three main tiers of this thesis

Chapter 2

Literature Review

It is the objective of this literature review to seek information which will help answer the question,"How can an UAV estimate its *pose*(position and orientation) accurately? " Although this objective is geared towards the isolation of key sources of inaccuracy in UAVs (which can later be addressed with appropriate strategies), the review has been enriched with topics surrounding navigation systems in general. The reader will find this broader literature review useful in gaining an adequate background which contextualizes the remaining chapters of the thesis. Figure 2.1 illustrates a typical navigation system where the sensors have been grouped according to their navigation philosophies (dead reckoning and position-fixing).

The review has been divided into four categories. The first two categories are related to the fundamental methods of *navigation* which are used in UAV applications, namely *dead reckoning* and *position fixing*; the third category is related to *integrated navigation*, which can be viewed as a hybrid of the two previously mentioned methods of navigation; and the fourth category is related to literature which deals with UAV *dynamics*. Note that, unless stated otherwise, the UAV platform in each citation is a quadrotor helicopter (also known as a quadcopter).

2.1 Dead Reckoning

Dead reckoning is a method of navigation whereby a relative signal is integrated in order to obtain the absolute navigational solution (Groves, 2013). Two common examples used in UAV platforms are gyroscopes and accelerometers (the output of both sensors must be integrated with respect to time to obtain attitude/positional information, respectively). The two advantages afforded from systems based on dead reckoning are 1) they can form a self-

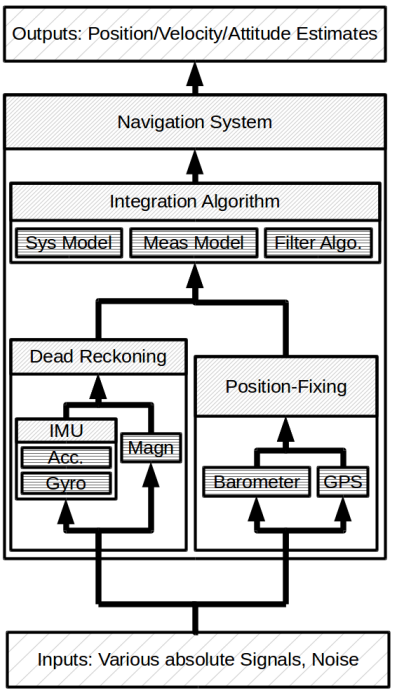


Figure 2.1: A typical navigation system

contained navigation system, and 2) high measurement update rates are possible. The two disadvantages of a system based on dead reckoning, on the other hand, are that 1) drift inevitably arises from the accumulation of errors in the process of signal integration, and 2) The navigational solution must be initialized.

An inertial measurement unit is a typical navigation unit which leverages the principle of dead reckoning and can be found in the vast majority of UAV applications. They usually comprise three accelerometers, which measure specific force in three orthogonal axes, and three gyroscopes (gyros) which measure angular rate with respect to three orthogonal axes. The IMU can be used to form a standalone Inertial Navigation System (INS) which would employ a strapdown architecture (This is when the IMU is rigidly fixed to the vehicle in some configuration). This illustrates the first advantage of a navigation system based on dead reckoning principals. (i.e. the systems are self contained - they do not rely on the availability of external reference signals).

2.2 Position Fixing

Position fixing is a method of navigation whereby the coordinates of known external entities are used to infer the coordinates of the vehicle being navigated. A common example used in UAV platforms, to be described in the paragraph that follows, is a Gloabl Positioning System (GPS). The advantages afforded by a system based on position fixing are 1) They do not suffer from a measurement drift , and 2) the navigation outputs do not need to be initialized. Notice that these advantages are the antonyms of the disadvantages found in a dead reckoning based system. The disadvantages of a system based on position fixing are 1) they rely on the availability of an identifiable external signal (so they do not possess the ability to be self-contained), and 2) relatively large amounts of data must be stored and processed, so slower update rates are inevitable. These two advantages can be viewed as the antonyms of the advantages afforded from a navigation system based on dead reckoning. What follows is a recount of the literature where authors have studied a UAV navigation system based on position fixing.

Satellites which orbit the earth may be used to fix the position of an object near the earths surface. The satellites broadcast signals of varying frequencies which allows them to be distinguished. This data contains information about the time the signal was sent, as well as information about the orbit of that particular satellite. Signals from multiple satellites may be received by a vehicle on the earths surface and a navigation solution may be computed from this data. This technique is common to UAV applications, and a list of authors who have implemented global positioning systems (GPSs) would be exhaustive. The problem with SN is that errors arise in the signals as a result of some physical phenomena in their transmission. These errors limit the obtainable accuracy of the SN system to the meter level.

To mitigate this problem, a geographic location may be surveyed and a reference station may be set up with the aforementioned errors calibrated out. These baseline signals may be used to resolve the errors in a UAV which is in close proximity (more or less 20 km) to the reference station. A common system based on this differential navigation is a real-time kinematic (RTK) positioning system. This advanced method of SN increases the obtainable accuracy in the navigation solution by an order of magnitude, with some authors predicting a positioning accuracy of just 3 cm Rieke *et al.*.

Li *et al.* (2013) have developed a real-time capable direct geo-referencing system in which they compute a navigation solution using a GPS (corrected by a RTK reference). The authors report a positional accuracy of 5 cm and an attitude accuracy of 0.5 deg. Although their positional accuracy solution disagrees by 40% with the aforementioned author, the general agreement is

that an accuracy within single digit centimeter level is possible. This range of accuracy is further consolidated by newer commercial grade sensors which can be found on the market. For instance, *SwiftNavigation* have developed integrated navigation systems that boast an accuracy within the 2-15 centimeter range.

2.3 Integrated Navigation

The presence of *uncertainty* in the navigation measurement data is inevitable. This fact complicates UAV navigation and in addition, each method of navigation that has been explored above has exhibited some practical advantages and disadvantages. It is the goal of integrated navigation to irridicate some uncertainty from the navigational outputs, in addition to complementing the other technologies onboard the system. Integrated navigation encompasses the process of estimating the most probable navigation states using the data obtained from multiple sensors as well as the dynamics of the system.

Various filtering techniques have been applied to UAV navigation in an attempt to achieve this state estimation. State estimation in UAVs is dominated by the family of Kalman filters (Extended Kalman Filters are used most abundantly while Unscented Kalman filters also exert their presence in the literature). What follows is a recount of the work done in the domain of integrated navigation in UAVs. The literature has been subdivided, according to the method of filtering. It is the goal of this sub section to illustrate the dominance of Extended Kalamn filtering in UAV navigation systems.

2.3.1 Traditional Kalman Filter

The most basic integrated navigation system might include the combined measurements from a 1-axis gyroscope, a 3-axis accelerometer and a GPS. This combined integration was investigated by Yoo *et al.* (2003). The study serves in providing the lower end of achievable performance in terms of navigation precision. The authors find that a positional accuracy of 5 m was attainable, while a attitude precision of 15 degrees was achieved.

Nagai *et al.* (2009) have combined measurement data from a laser scanner, two charge-coupled device (CCD) cameras, a GPS and a 6 degree-of-freedom IMU. The test bed used is a 330 kg (plus 100 kg payload) helicopter UAV whose purpose is one of geo-referencing. The experimental test work reveals that ground points may be directly geo-referenced with an error in the 10-30 cm range(Swift Navigation[Online]). While not many details are provided, it should be realized that the payload in this UAV eradicates any

constraint on computational power, since more resources may be installed without effecting vehicle dynamics too greatly. The applicability of the work done by this author in the current research is thus to demonstrate primarily that in particular applications - computation resources is not a constraining factor. Nonetheless, the need for a filter with increased complexity should first be critically assessed because it might not be necessary (the strategies proposed in this thesis are indeed computationally expensive, so the reader is urged to keep in mind the lack of constraint which is posed by such a demand in certain applications).

2.3.2 Extended Kalman Filter (EKF)

The number of states which are estimated in a navigational system is not unique. The two popular approaches are to perform 1) error state estimation, where measurements from the different sources are used to estimate the measurement errors and subsequently update the navigational outputs and 2) full state estimation in which a dynamic model of the system is included in order to estimate the navigational outputs which also get used in successive updates. Although the EKF is abundantly implemented, many assumptions are often made.

Han and Lee make the case for updating a GPS/IMU navigation system using aerial triangulation(AT) techniques applied using digital images. The authors have only simulated their tests, but have found that the AT updates reduce the positional and attitude error estimates. Choi and Lee (2011) continue to experimentally verify the simulated data previously produced. The authors report that georeferenced ground data could be obtained with an accuracy of +- 4 cm. This shows great affinity with the proposed project in terms of accuracy, but ceratinly not in terms of budget. The study is reported to be pat of a real-time aerial monitoring system which has the support of 6 million US dollars and a 5 year period of development behind it. Since real-time georeferncing is not the objective of the proposed study, these daunting figures should be regarded lightly. Nonetheless, the study confirms that updating a GPS/IMU navigation system with AT undoubtidly increases the accuracy of the navigation system, and is an avenure worth exploring in the currently proposed research.

2.3.3 Unscented Kalman Filtering (UKF)

Oh (2010) has combined the measurement data obtained from an IMU, GPS and magnetometer (3-axis). The author performed no physical experiments, but have evaluated the performance of their filter in a realistic simulation

environment (this is the same approach which will be followed in this thesis) against that of an existing EKF. The author acknowledges the short coming of the UKF, that is computational cost. The acceptance of this shortcoming is motivated with an apparent increased accuracy.

Gross *et al.* (2012) consider the effect of the filtering scheme on the performance of the navigation system in producing its outputs. The athours implement three different filters for comparison purposes. A linear estimator know as the Acceleration Vector Attitude Estimator (AVAE) is the first to go under the microscope. The advantages afforded by this estimator are that it offers a low computational complexity. Its disadvantage is one relative to the non-linear estimators (a EKF and a UKF), and that is accuracy. The two non-linear filters yielded similar performance in terms of accuracy, but the UKF inherited significantly more computational complexity than the EKF. This study suggests that an EKF would be the filter of choice in the proposed study.

2.3.4 Applications of other filters

Nemra and Aouf (2010) fuse INS/GPS data using a State-Dependant Riccati Equation (SDRE) navigation filter. They propose this filter as an alternative to an EKF, and provide quantitve comparisons between the two following simulations. The simulations show that the SDRE filter outperforms the EKF, owing to its capaility of handling issues related to linearization. The implementation of such a filter may be problematic though, and the authors have provided a comparison of computation times between the two filters. This comparison indicates that the SDRE filter takes almost three times as long as the EKF to perform 100 iterations of the simulated trajectory, and so is indicative of computationally intensive filtering.

2.4 Theoretical Conflicts

The following section forms an excerpt from an accepted but not yet published proceeding produced by the author during this thesis (Minnaar and Smit, 2017). Six states are necessary to fully describe the motion of any dynamic rigid body. Accelerometers may be used in the estimation of the linear states of such a body (position and velocity) through double integration Kowalczuk and Merta (2015). In addition, since accelerometers measure specific force (i.e. they contain the effects of gravity), they have also been used in the estimation of vehicle attitude Bourke *et al.* (2011). A theoretical conflict arises, however, when accelerometers are used to estimate both lin-

ear velocity and vehicle attitude simultaneously Leishman *et al.* (2014). A single triad of accelerometers is therefore inadequate in fully describing the dynamics of a rigid body.

The attitude of a vehicle has traditionally been observed using a single triad of gyroscopes coupled with a single step of integration Xue *et al.* (2009), thereby resolving the theoretical conflict previously mentioned. It is desirable, however, to replace gyros with accelerometers in the estimation of rotational velocity since accelerometers are cheaper to manufacture, don't have large growing static biases, and sport a wider dynamic range of operation Mingli *et al.* (2006). A large body of literature has been driven by this need to replace gyros with accelerometers Cao and Zu (2010).

Chen *et al.* (1994) proposed an array of 6 single axis accelerometers in which it became possible to sense the rotational acceleration of a body. Specifically, the accelerometers were configured so that the net centripetal accelerations in the measurement system become zero. Through integration of the tangential accelerations, the authors showed that it is possible to obtain the linear states and rotation states of a system simultaneously using this accelerometer array. This is known as the integration approach.

Wang *et al.* (2003) highlighted the fact that integrating noisy signals to obtain system states would result in growing biases in the states of the system, and suggested the addition of 3 accelerometers to the configuration proposed by Chen *et al.* (1994). The redundant acceleration information exposes the net centripetal accelerations in the system and can be used to directly solve the angular velocity without integration. This is known as the algebraic approach and places a bound on the errors in the state estimates. Other authors have suggested array configurations with even higher counts of accelerometers Cardou *et al.* (2011).

Although redundant acceleration information may alleviate some of the functional difficulties posed by the integration approach, redundant accelerometers increases the cost of navigation system. This is a contradiction to one of the main drivers to develop accelerometer arrays in the first place. This thesis will attempt to bridge the gap between the integral and algebraic approaches, using six single axis accelerometers while still solving the angular velocity of the system algebraically. This is done through the inclusion of a singular value decomposition of the difference in point acceleration measurements made at two distinct locations in the system. A detailed description will be provided in Chapter 4

2.5 Literature Dissection

The opening section of the literature review drew a distinction between navigation systems that rely on position-fixing and dead reckoning principles. It was shown that dead-reckoning sensor like inertial measurement units rely on signal integration in order to produce navigation outputs. There are disadvantages to this, and one in particular will be addressed in Chapters 5. Specifically, it will be shown that numerical errors in the integration process spoil the navigation solution and proposals for mitigating this are developed. The results that stem from the solutions developed in Chapter 5 will be provided in Chapter 6. The following body of literature presented some of the research that has been done in the area of state estimation. The author was not able to discern from the literature how signals from position fixing sensors such should be included into navigation systems where certain estimates are already made from dead-reckoning philosophies. Since there is not a unique way to include position-fixing sensors in these scenarios, Chapter 3 will develop estimation theory and Chapter 6 will present results that address this ambiguity. The last body of literature draws attention to the fact that a theoretical conflict is commonly raised when accelerometers are used for determining the orientation parameters of a UAV. This theoretical conflict will be discussed and resolved in Chapter 4. The navigational improvements that results from resolving this theoretical conflict will be documented in Chapter 6.

2.6 Conclusion

The literature review endeavored to provide context for the theory that will follow. In essence, three difficulties have been taken from the literature for further investigation. Firstly, it is not clear what the guideline is for the integration of various sensors i.e. how should the measurements of two sensors measuring the same quantity be blended? Should one sensor be used to estimate the error in the other or should both sensors be used to estimate the same internal state? Although there are a myriad of sensor combinations that have been applied in the literature - this question is still unclear. The theory to answer this question will be developed in Chapter 3, while a strategy will be validated through simulation in Chapter 6. Secondly, a theoretical conflict was identified in the literature, and this will be the focus of Chapter 4. A third issue was not discussed explicitly in this chapter but has been found in open-source code for a popular UAV autopilot. The issue is that the DCM which describes the attitude of a vehicle is supposed to be orthonormal - but drifts from this condition due to numerical errors. This issue will be detailed in Chapter 5.

Chapter 3

A Virtual Test Bench

The proposals made in this thesis are assessed leveraging simulation technology. The models which are needed to facilitate this are provided in this chapter (Figure 3.1 represents the relevant position of the current chapter within the whole thesis), along with the relevant theory. It will begin by giving a system wide perspective, focusing on the description of the interaction between the various components that make up the system model. The constituents of the system model are then examined in further detail, and their inner mechanics are revealed.

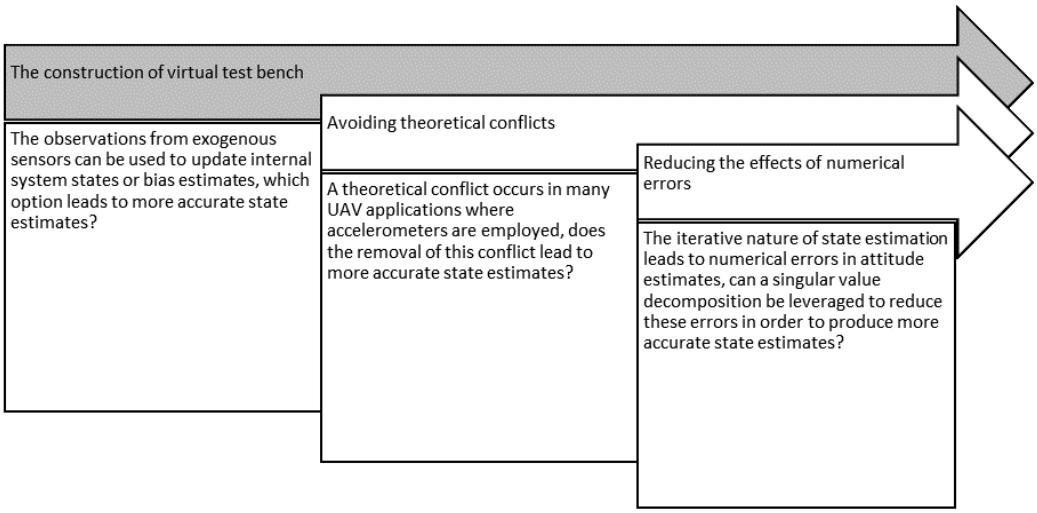


Figure 3.1: Chapter 3 is represented by the darkened section of the image and forms the first of the three main pillars which make up the thesis.

The models which are developed in this chapter will later be referred to as the virtual test bed and will, as alluded to above, be used to assess all navigation strategies that are later developed. Each theoretical concept depends on a knowledge of the one developed immediately prior. Figure 3.2 below assists in contextualising each of the concepts, so as to orient the reader within the chapter. First, a description is given regarding the variables that make up the navigation solution. In the case of attitude variables, it will be shown that a unique choice of variables is not possible. Hence, the description of Euler angles is developed - they are the variables which are used in this project to describe the vehicle attitude. Following the state variable description, the models which govern their time evolution will be developed under the heading "Plant dynamics". Lastly, the state estimator which is responsible for generating a best estimate of the state during motion will be developed - it relies on plant dynamics models.

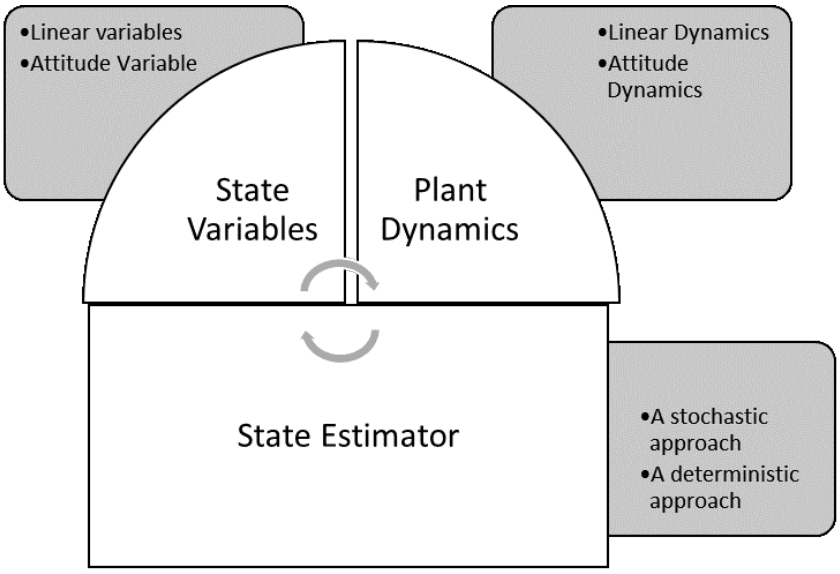


Figure 3.2: Chapter 3 is modularised into the sections shown.

3.1 Co-Simulation Architecture

The science and engineering communities have benefited greatly from the advent of computer simulation, with many popular solver technologies having now already been in commercial development for upwards of twenty years. A challenging situation in computer simulation arises when there is a requirement to couple multiple technologies which individually cater to different simulation domains. This approach, which is known as co-simulation, allows each dedicated simulation technology to generate a solution for a particular epoch. The two technologies then exchange input and output data at these epochs, before independently continuing to generate a solution for the next epoch. This thesis will leverage such a scheme, using two dedicated simulation technologies. One is used for multi-body dynamics analysis, and is responsible for calculating the motion of a virtual quadcopter. The other simulation technology is responsible for solving the remaining mixture of algebraic and differential equations which make up the control system and state estimator. Each technology is briefly described below, and context is provided for the role that the software fulfills. The section concludes by describing how the two software environments are interfaced and used to perform a co-simulation.

MotionSolve (MS), a popular multi-body dynamics solution, is used in this thesis to simulate the motion of a quadcopter. The user interacts with the software by generating an input deck which contains information regarding the system inputs, physical layout of the components in the system, and the constraints which couple some of the components together (and hence, also certain degrees of freedom in the system). The software uses this input deck to generate the equations of motion for the system, and then subsequently computes results by solving those equations (See Appendix A for more information on MS). The input deck to MS is created using its corresponding pre-processor, MotionView (MV). This graphical user interface allows for the parametric modeling of the quadcopter, and convenient generation of all other required inputs in the deck. The environment, as well as the physical CAD model of the virtual quadcopter are shown in Figure 3.3. More information regarding MV can be found in Appendix B.

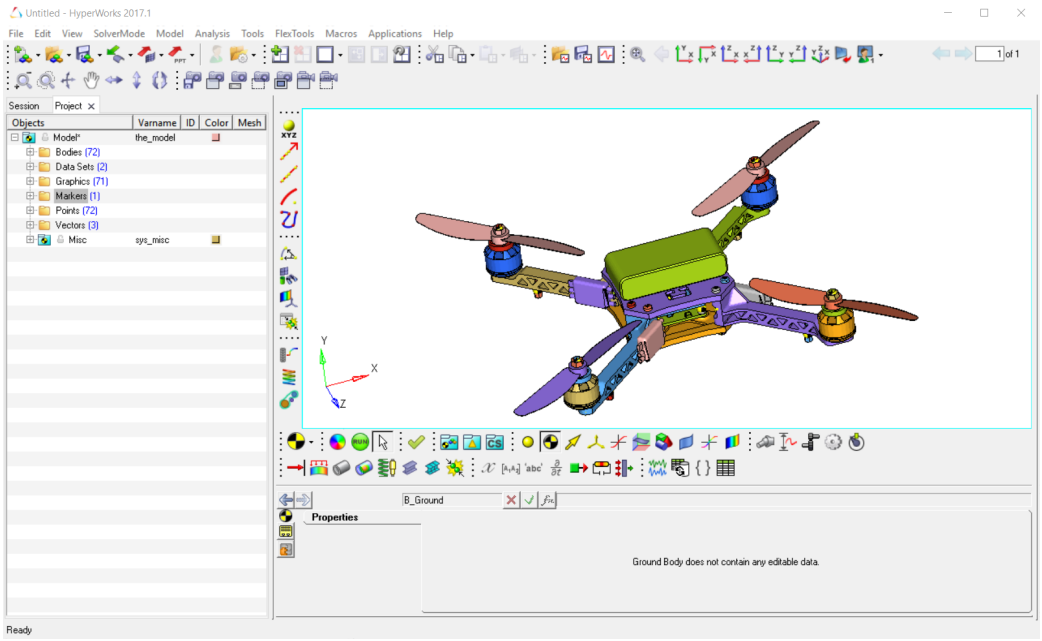


Figure 3.3: The MotionView preprocessor user interface.

Activate, which is a product of the solidThinking (sT) company, is a schematic modeling environment which allows users to treat built-in libraries as "black boxes" while building their simulation models. A user can connect these "black boxes" together to build up complex simulations. Another particularly important capability of this tool (within the context of this thesis) is that it supports interfaces with 3rd party products for performing co-simulation. These interfaces are implemented as easy-to-use block components, ready to be connected to the rest of the "black boxes" which are present in a particular simulation. The modeling paradigm used in sT Activate is ideally suited to building models for the state estimator and control system (more information on the product can be found in Appendix C). Figure 3.4 illustrates the high-level schematic model which represents the simulations that utilise the models which are developed in this chapter. The remaining sections in the chapter will describe sections of the model which are embedded in this system model (so are lower in the hierarchy).

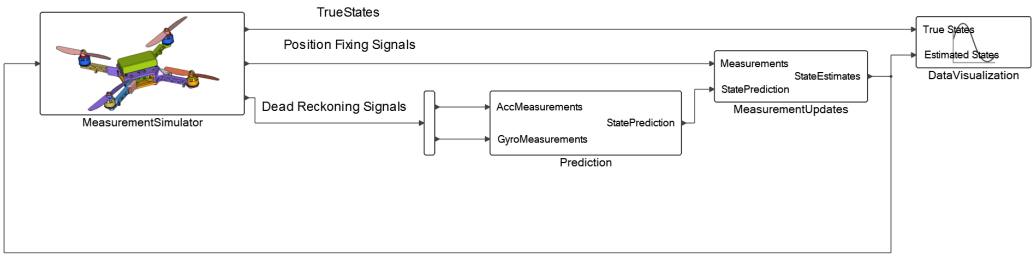


Figure 3.4: High level schematic model of simulations performed in this thesis.

The co-simulation between MS and sT Activate is a parallel process which is driven by the latter product, acting as the master simulator. The interface between the two products is responsible for piping the outputs of sT Activate into the runtime (solver) variables of MS. The outputs of MS are also handled by the interface, which are piped into sT Activate. The data exchange between the two products through the interface occurs at discrete time instants, and a Gauss-Seidel iteration method is employed to deal with the coupling of the equations from each domain (more details can be found in an excerpt of the sT Activate documentation which is provided in Appendix D). Figure 3.5 represents the co-simulation scheme, where the simulation interface has been embedded inside the sT Activate product.

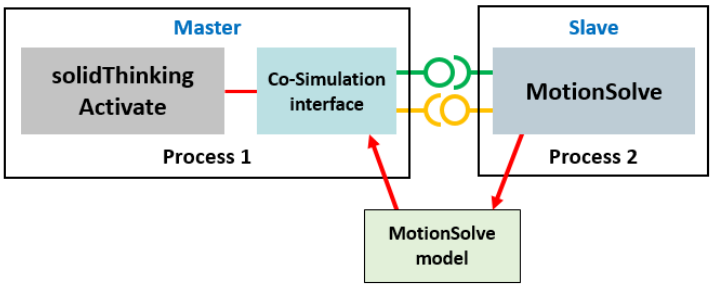


Figure 3.5: An illustration of the co-simulation concept.

3.2 State Variable Conventions

Physical entities such as a quadcopter contain dynamic properties that describe its motion. These dynamic properties include, for example, the position and velocity of the vehicle and are referred to as state variables. We form the vehicle state by casting these state variables into a vector. In this

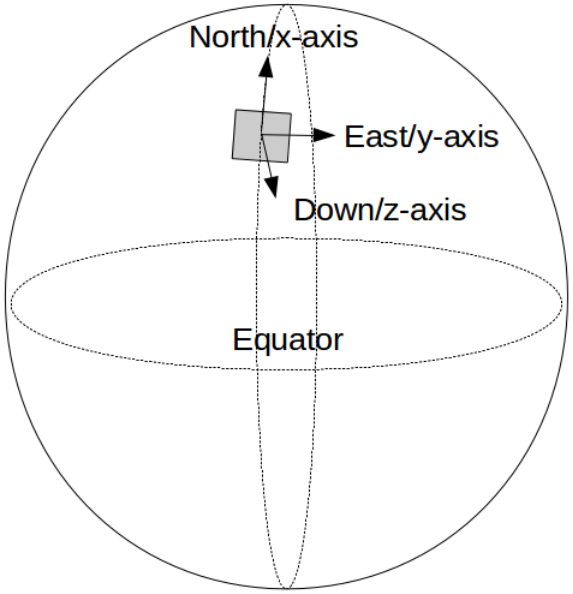


Figure 3.6: North East Down frame of reference

section, each of the state variables contained in the UAV state vector will be described. This state vector is shown below.

$$x = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\phi} & \dot{\psi} & \dot{\theta} & x & y & z & \phi & \psi & \theta \end{bmatrix}^T \tag{3.2.1}$$

The most easily interpreted states are the ones which describe the linear motion of the UAV. The state variables highlighted below comprise the linear dynamics of the UAV.

$$x = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\phi} & \dot{\psi} & \dot{\theta} & x & y & z & \phi & \psi & \theta \end{bmatrix}^T \tag{3.2.2}$$

The first three highlighted state variables are the components of the UAV linear velocity while the last three highlighted state variable are the positional coordinates of the UAV's center of gravity. The $\{x, y, z\}$ components make reference to a coordinate frame which is fixed to the earth surface at the initial takeoff position of the UAV. The x and y axes are aligned with the north and east directions, respectively. The z axis is normal to the earths surface and points in the direction of the earths center. This frame of reference is depicted in Figure 3.10 and is known in the literature as the North East Down (NED) frame of reference.

The state variables which were not highlighted in equation 3.2.2 comprise the rotational dynamics of the UAV's orientation. The difficulty in dealing

with the rotational dynamics of any object is that there are multiple ways to describe the orientation of the body. One parameterization approach that has become popular are the Euler angles of the body. The remainder of this section is split up into three subsections which provide the background that is necessary to describe the Euler angles of a body.

First a matrix transformation is derived which transforms an input vector according to some sequence of rotations. Next, a matrix transformation is derived which may be used to swap between the referenced coordinate systems of a vector. It shall be shown that these two transformations share a relationship and that they provide us with a method for quantifying the attitude of the UAV through the Euler angles, which constitutes the final subsection. Th remainder of this section will reference the axes of an orthogonal coordinate system with the subscripts $\{1, 2, 3\}$ and not the subscripts $\{x, y, z\}$ in order to simplify notation.

3.2.1 The Rotation Matrix

Consider Figure 3.7 which shows the effect of a rotation about the e_3 axis on the e_1 unit vector. This is termed a 3-rotation.

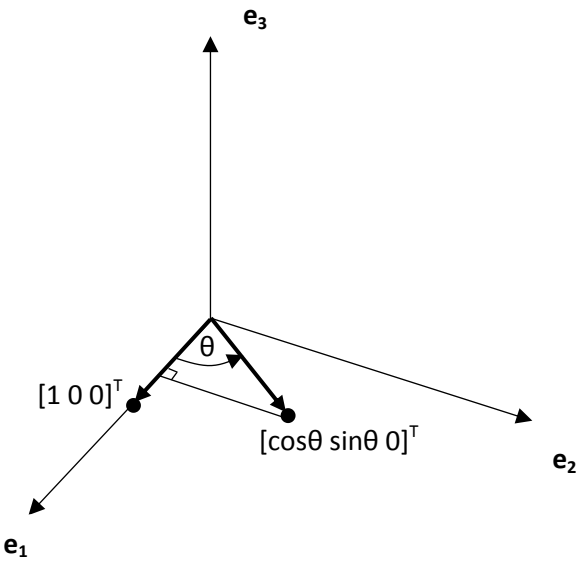


Figure 3.7: A 3-Rotation of the e_1 unit vector from the standard basis of $\mathbb{R}^3$

Figures similar to that of the one above can be constructed to show the effects of a 3-rotation on the e_2 and e_3 unit vectors. The relationships below

summarizes the findings. Let

$$\begin{aligned} R_\theta \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} &= \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \\ R_\theta \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} &= \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix} \\ R_\theta \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \end{aligned} \tag{3.2.3}$$

where R_θ is the rotation matrix we seek to find. It useful to apply a theorem from linear algebra which allows one to find an unknown matrix transformation by concatenating its effects on the unit vectors of a standard basis into a matrix as done below. The theorem used and its proof are provided in Appendix E.

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2.4}$$

This rotation matrix can now be used to determine the coordinate vector of an arbitrary vector in $\mathbb{R}^3$ which has been rotated about the e_3 unit vector (expressed as a linear combination of the standard basis vectors). The derivation which produced equation 3.2.4 may be repeated in order to find the matrix transformations which corresponds to a 1-rotation

$$R_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{3.2.5}$$

and a 2-rotation.

$$R_\psi = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \tag{3.2.6}$$

The symbols ϕ and ψ have been used to denote the angles of rotation around the e_1 and e_2 axes,respectively.

3.2.2 The Direction Cosine Matrix (DCM)

The rotation matrix may be used to find the coordinates of a vector that has been rotated. However, we would also like to find a matrix transformation

which rotates the reference axes and not the vector itself. Consider Figure 3.8 which depicts two bases of $\mathbb{R}^3$. The basis consisting of the vectors $\{b_1, b_2, b_3\}$ represent a basis whose orientation differs from the $\{e_1, e_2, e_3\}$ by a 3-rotation.

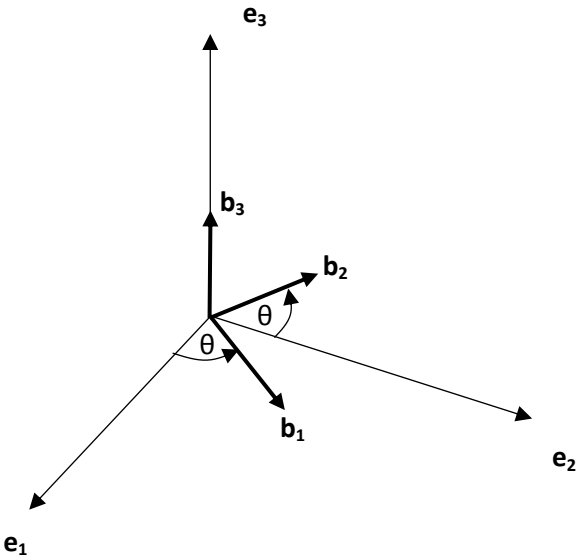


Figure 3.8: A 3-Rotation of the the standard basis of $\mathbb{R}^3$

To find the matrix transformation which facilitates the desired basis change, we express each e_i vector in terms of the b_i vectors as follows:

$$\begin{aligned} e_1 &= \cos\theta b_1 + \cos(90+\theta)b_2 + \cos 90 b_3 \\ e_2 &= \cos(90-\theta)b_1 + \cos\theta b_2 + \cos 90 b_3 \\ e_3 &= \cos 90 b_1 + \cos 90 b_2 + \cos 0 b_3 \end{aligned} \tag{3.2.7}$$

Casting the above relationships into a matrix format and making use of some trigonometric identities, we find a transformation that facilitates the desired basis change.

$$C_\theta = \begin{bmatrix} \cos\theta & \sin\theta & \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2.8}$$

This is known as the Direction Cosine Matrix (DCM). The same derivation procedure may be performed in order to obtain the DCMs for 1- and 2-rotations of the standard basis. For a 1-rotation of the standard basis, the

DCM becomes:

$$C_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{3.2.9}$$

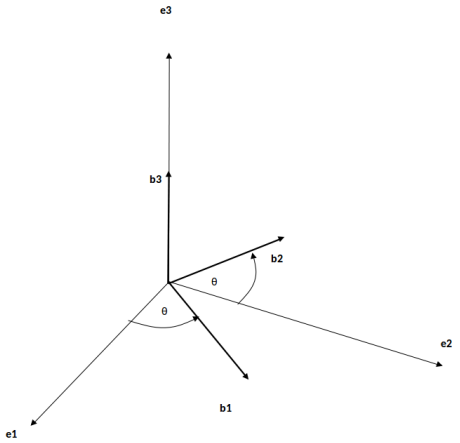
and for a 2-rotation of the standard basis, the direction cosine matrix becomes:

$$C_\psi = \begin{bmatrix} \cos\psi & 0 & -\sin\psi \\ 0 & 1 & 0 \\ \sin\psi & 0 & \cos\psi \end{bmatrix} \tag{3.2.10}$$

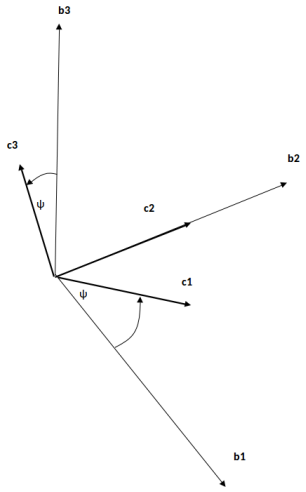
We have again used symbols ϕ and ψ to denote angles of rotation around the e_1 and e_2 axes, respectively. Notice that the DCM derived above are the transpose of the rotation matrices derived in equations 3.2.4, 5.1.2 and 3.2.6. Moreover, we make the observation that $R_i C_i = I$, where I is the identity matrix and the subscript $i \in \theta, \phi, \psi$. This means that $R_i^{-1} = C_i = R_i^T$. Similarly, $C_i^{-1} = R_i = C_i^T$. We may conclude that transforming the coordinates of a vector is as simple as applying a direction cosine matrix or rotation matrix to the original coordinate vector (depending on which basis change is desired).

3.2.3 Euler Angles

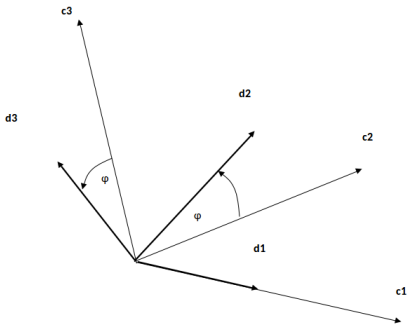
Section 3.2.2 showed how a basis change could be performed for the cases when the new basis is a simple 1-, 2- or 3-rotation of the standard basis vectors. However, we would like to find a transformation which changes the basis of a vector to one which represents an arbitrarily orientated set of axes. Consider the process depicted in Figure 3.9. Firstly, the standard basis has undergone a 3-rotation. The new set of axes then undergoes a 2-rotation. Finally, the preceding set of axes undergoes a 1-rotation.



(a) 3-Rotation of the standard basis of $\mathbb{R}^3$



(b) 2-Rotation of the basis vectors $\{b_1, b_2, b_3\}$



(c) 1-Rotation of the basis vectors $\{c_1, c_2, c_3\}$

Figure 3.9: A 3-2-1 rotation sequence

It is clear that the final basis of $\{d_1, d_2, d_3\}$ is arbitrarily oriented in $\mathbb{R}^3$. Thus we have observed that any orientation in $\mathbb{R}^3$ can be described by the three angles θ, ψ and ψ. These angles are called *Euler Angles* and we note that we have used a 3-2-1 rotation sequence[1]. Of course, we could have used a different rotation sequence and in fact, there are twelve of them (some being more popular than the rest). We carefully notice that each successive set of axes can be found from transforming the preceding basis.

$$\begin{aligned}
\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} &= C_\theta \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \\
\begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix} &= C_\psi \begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix}^T = C_\psi C_\theta \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T \\
\begin{bmatrix} d_1 & d_2 & d_3 \end{bmatrix} &= C_\psi \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^T = C_\phi C_\psi C_\theta \begin{bmatrix} e_1 & e_2 & e_3 \end{bmatrix}^T
\end{aligned} \tag{3.2.11}$$

We may combine the matrix transformations since we are not interested in the coordinate systems of the intermediate transformations.

$$C_{321} = C_\phi C_\psi C_\theta \tag{3.2.12}$$

We note that the subscript 321 which designates the rotation sequence is the reverse order of the subscripts ϕ, ψ and θ which correspond to the angles of rotation (a consequence of premultiplication). Performing the matrix multiplications yields[2].

$$C_{321} = \begin{bmatrix} \mathrm{c}(\theta)\mathrm{c}(\psi) & \mathrm{s}(\theta)\mathrm{c}(\psi) & -\mathrm{s}(\psi) \\ \begin{matrix}-\mathrm{s}(\theta)\mathrm{c}(\phi)\\ +\mathrm{c}(\theta)\mathrm{s}(\psi)\mathrm{s}(\phi)\end{matrix} & \begin{matrix}\mathrm{c}(\theta)\mathrm{c}(\phi)\\ +\mathrm{s}(\theta)\mathrm{s}(\psi)\mathrm{s}(\phi)\end{matrix} & \mathrm{c}(\psi)\mathrm{s}(\phi) \\ \begin{matrix}\mathrm{s}(\theta)\mathrm{s}(\phi)\\ +\mathrm{c}(\theta)\mathrm{s}(\psi)\mathrm{c}(\phi)\end{matrix} & \begin{matrix}-\mathrm{c}(\theta)\mathrm{s}(\phi)\\ +\mathrm{s}(\theta)\mathrm{s}(\psi)\mathrm{c}(\phi)\end{matrix} & \mathrm{c}(\psi)\mathrm{c}(\phi) \end{bmatrix} \tag{3.2.13}$$

In the context of the work presented in this thesis, ϕ is referred to as the *roll*, ψ is referred to as the *pitch*, and θ is referred to as the *yaw*. If we consider a non-inertial frame which is attached to the body of a UAV as shown in Figure 3.10, then the Euler angles which align the body axis system of the UAV with the one depicted in Figure 3.10 describe the orientation of the UAV. It is these orientation parameters that make up the variables of the state vector which were not highlighted in Equation 3.2.2.

3.3 The Plant

The forces acting upon the UAV and their corresponding effects are described in the introductory paragraphs of this section. The dynamics of the plant are then described in both linear and rotational motion.

[1] The angles denoting a 3-2-1 rotation sequence are also known as *aircraft* angles, since this sequence has become popular in aviation spheres

[2] We have let $\mathrm{c}(x) = \cos(x)$ and $\mathrm{s}(x) = \sin(x)$ in order to save space.

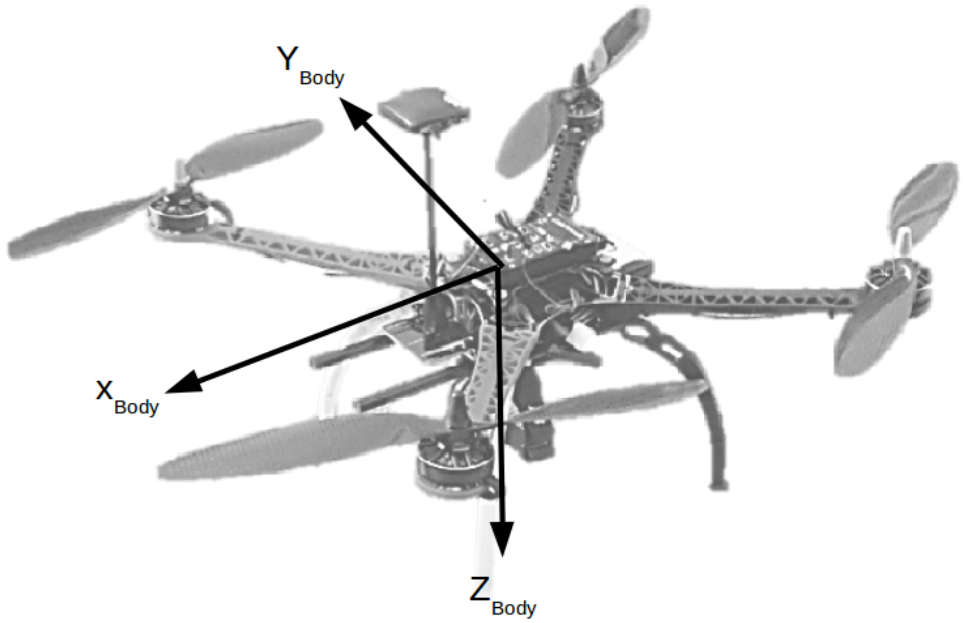


Figure 3.10: Body-fixed coordinate frame

Figure 3.11 provides a free body diagram (FBD) where it can bee seen that six external forces act on the UAV. The four forces shown in red are thrust forces (one from each motor). The force shown in blue comprises various drag components which have been lumped into one drag force. Note that this force does not necessarily act through the UAV's center of mass (CM). The last force, shown in green, is due to the weight of the UAV and acts towards the center of the earth. Each of the forces described above (excluding the force due to gravity) are non-linear functions of time. Please see Appendix F for a description of some of the assumptions which are employed, together with a tabular summary of the assumptions which have been made to deal with the non-linear effects of the forces.

Each rotor induces a thrust force and a torque (around its rotation center) which is dependent on the rotational speed of the rotor. The vehicle motion may then be controlled by adjusting the speed of the rotors. There are four motions that the UAV may undergo based on the differential speeds of its rotors. Consider Figure 3.12, which demonstrates the mechanics of the UAV while neglecting external forces such as drag or one which would be imposed by a gravitational field. Increasing the thrust of all four rotors by an equal amount will cause the UAV to lift in the Z_{Body} direction, this is shown

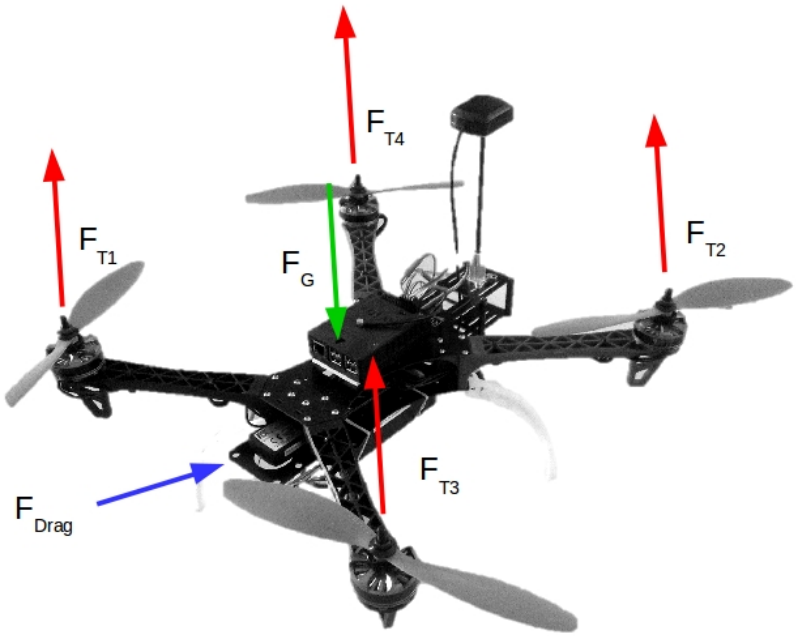
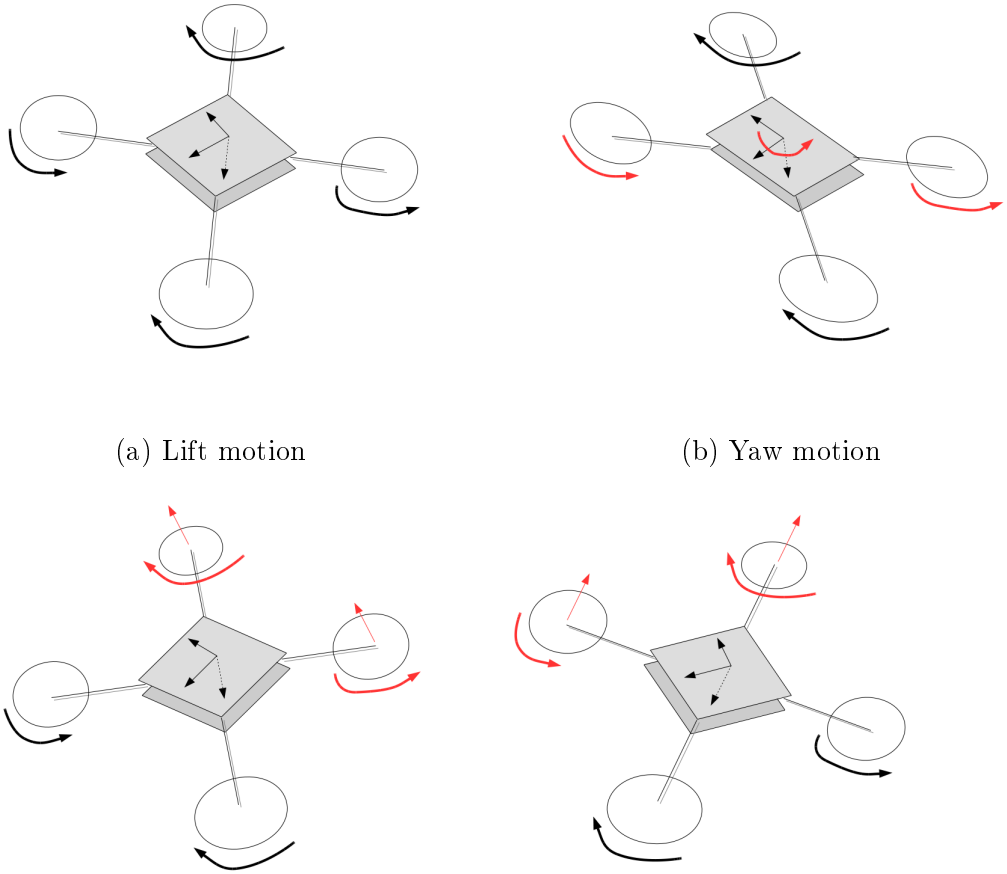


Figure 3.11: A free body diagram for a quadcopter

in Figure 3.12a. Furthermore, we may control the yaw of the vehicle (its rotational motion around Z_{Body}) by increasing the speed of opposite rotors in order to perturb the balance of angular momentum possessed by the UAV. This is illustrated in Figure 3.12b. The remaining unique UAV motions are termed pitch and roll. These motions may be achieved by increasing the thrust of two adjacent rotors equally, while decreasing the thrust of the remaining two adjacent motors equally. These actions cause the UAV to rotate about its Y_{Body} and X_{Body} axes, depending on which adjacent rotors are paired. These motions are illustrated in Figure 3.12c and Figure 3.12d.

Ideally, we would like to be able to command the UAV state vector to follow any arbitrary path in its state space. Unfortunately, the UAV is able to achieve only four unique motions (as described in the paragraph above) in spite of the fact that we have chosen 12 state variables to represent the



(a) Lift motion

(b) Yaw motion

(c) Pitch motion

(d) Roll motion

Figure 3.12: UAV Mechanics

state of our system [3]. The UAV is therefore under-actuated because it has more degrees of freedom than it has actuators. The state space of the UAV therefore becomes constrained based on the UAV states that we choose to control. For example, we may choose to control the UAV pitch, but then we cannot control the motion of the UAV in the X_{Body} direction (unless we do so in an open loop fashion by relating the amount of roll to the distance traveled in the X_{Body} direction). Since the control of the UAV is beyond the

[3]We could have defined our state vector differently of course, but no matter what representation we choose, the UAV has six degrees of freedom (3 rotational and 3 linear translational)

scope of this project, it is necessary only to know that the evolution of the state vector is constrained by the mechanics of the UAV.

3.3.1 Linear Dynamics

In this section, we derive the linear dynamics equations of the UAV and formulate them in a state space format. Firstly, we may apply Netwon's law of motion in order to model the linear dynamics of a UAV assuming that it is a rigid body. Note that we make use of the inertial frame for the reference.

$$f = ma \tag{3.3.1}$$

where f is the force associated with the acceleration a of mass m. Then we formulate the linear dynamics equation as multiple first order differential equations to account for the three independent directions into which the quantities may be separated. Finally, we cast these differential equations into the state space framework. This is shown below.

$$\begin{bmatrix} \ddot{x} \\ \dot{x} \\ \ddot{y} \\ \dot{y} \\ \ddot{z} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ x \\ \dot{y} \\ y \\ \dot{z} \\ z \end{bmatrix} + \begin{bmatrix} \frac{f_x}{m} \\ 0 \\ \frac{f_y}{m} \\ 0 \\ \frac{f_z}{m} \\ 0 \end{bmatrix} \tag{3.3.2}$$

3.3.2 Rotational Dynamics

For the moment, let us consider our frame of reference to be a body fixed non-inertial frame so that we may apply the rotational equation of motion for a rigid body to the UAV.

$$M = \dot{L} \tag{3.3.3}$$

Where $\dot{L}$ is the rate of change of the angular momentum for the UAV. Note that our point of reference must be the UAVs centre of mass in equation 3.3.3. To apply the rotational equation of motion given by equation 3.3.3, we must obtain the angular momentum of the UAV and then derive it with respect to time. First we obtain the angular momentum of the UAV.

$$L = I \cdot \omega \tag{3.3.4}$$

Where I is the inertia matrix of the UAV. To find the time rate of change for the angular momentum, we derive 3.3.4 with respect to time.

$$\dot{L} = I \cdot \dot{\omega} + \dot{I} \cdot \omega \tag{3.3.5}$$

It can be shown that 3.3.5 simplifies as given below.

$$\dot{L} = I \cdot \dot{\omega} + \omega \times I \cdot \omega \tag{3.3.6}$$

Finally we may combine equations 3.3.3 and 3.3.6 and replace the cross product operation with a skew symmetric matrix to form the rotational dynamic equation of motion for the UAV.

$$M = I \cdot \dot{\omega} + \left[\tilde{\omega}\right] \cdot I \cdot \overline{\omega} \tag{3.3.7}$$

Once again, we must manipulate the equation so that it is appropriately formulated. This is shown below.

$$\dot{\omega} = -I^{-1}\tilde{\omega}I\omega + I^{-1}M \tag{3.3.8}$$

If we choose our body axes conveniently so that the products of inertia vanish in the above equations then we obtain our *Euler equations*.

$$\begin{aligned}
\dot{\omega}_1 &= \frac{1}{I_{11}}(M_1 - (I_{33} - I_{22})\omega_2\omega_3) \\
\dot{\omega}_2 &= \frac{1}{I_{22}}(M_2 - (I_{11} - I_{33})\omega_3\omega_1) \\
\dot{\omega}_3 &= \frac{1}{I_{33}}(M_3 - (I_{22} - I_{11})\omega_1\omega_2)
\end{aligned} \tag{3.3.9}$$

We cannot cast these angular dynamics relationships into the same state-space format as we did with the linear dynamics (because the relationship between the state variable vector and its derivative is nonlinear) so we make use of the more general representation.

$$\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = f\left(\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \phi \\ \psi \\ \theta \end{bmatrix}, M \right) \tag{3.3.10}$$

It should be understood that $f()$ represents a function in Equation 3.3.10, and does not refer to the UAV mass. It is important to note that Equation 3.3.10 has raised a concern. Specifically, the angular rate portion of the vector is referencing the body fixed frame and not the global inertial reference frame, as is the case with the rest of the vector. We could convert the angular rates so that they do reference the global frame. In that case, the states pertaining to the UAV's orientation would be $\begin{bmatrix} \dot{\phi} & \dot{\psi} & \dot{\theta} & \phi & \psi & \theta \end{bmatrix}^T$, as desired.

$$\begin{aligned}\omega_1 &= \dot{\phi} - \dot{\theta}\sin\psi \\ \omega_2 &= \dot{\theta}\cos\psi\sin\phi + \dot{\psi}\cos\phi \\ \omega_3 &= \dot{\theta}\cos\psi\cos\phi - \dot{\psi}\sin\phi\end{aligned} \tag{3.3.11}$$

However, we will not perform the conversion step during this thesis. The reason that the conversion will not be performed will be detailed in Chapter 5, but for now we offer the simple explanation that the Euler rates and body rates become equal under infinitesimal angles, so that equation 3.3.10 can be rewritten.

$$\begin{bmatrix}\dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \\ \omega_1 \\ \omega_2 \\ \omega_3\end{bmatrix} = f\left(\begin{bmatrix}\dot{\phi} \\ \dot{\psi} \\ \dot{\theta} \\ \phi \\ \psi \\ \theta\end{bmatrix}, M\right) \tag{3.3.12}$$

3.4 The State Estimator

We have previously argued (in the introductory chapter) that there is not always a clear relationship between the measurements we take on an entity and its internal states. We have also alluded to the fact that the measurements themselves are not free from uncertainty. It is the goal of this section to build contingencies for these two inescapable facts. We first discuss two traditional approaches for estimating the states of a system. We then show how these two approaches are combined in a way that leverages the model of the system and the knowledge of the uncertainties in the measurements to produce the best possible estimates of the system states.

3.4.1 A Stochastic Approach

Assume that sensors are being used to measure dynamic properties of a UAV from which its states must be calculated. Since each sensor contains some uncertainty, its measurements are typically reported as done below.

$$x' = \bar{x} \pm u_{\bar{x}}(P\%) \tag{3.4.1}$$

where $\bar{x}$ represents the most probable estimate of x' and $\pm u_{\bar{x}}$ represents the uncertainty interval in that estimate at some level of probability, $P\%$. It is

now possible to estimate a state variable pertaining to the UAV if we know its relationship with the measured dynamic property.

$$x = f(x') \tag{3.4.2}$$

For example, a barometer may be used to estimate the UAV altitude. In this case, the UAV's altitude is a function of the barometers pressure reading according to the barometric formula (Figliola and Beasley, 2011). Suppose that we are making use of a sensor which produces measurements containing an unacceptable level of uncertainty ($\pm u_{\bar{x}}$). Then from a stochastic viewpoint, we may follow one of two approaches in order to reduce the uncertainty in our estimate of the vehicle state.

1. We may replace the sensor being used with one which produces measurements with lower uncertainty.

2. We may add another sensor with comparable uncertainty so that we may observe the dynamic property being measured for a second time. Combining these observations increases the probability of that measurement (this is called sensor fusion).

The second of these routes is preferable because of its financial feasibility (commercial grade sensors have comparable uncertainty when they fall in similar price ranges, but if one would like to acquire a sensor with lower uncertainty - it would be a more expensive sensor). It turns out that there is more than one way to combine the measurements in order to exploit the redundant information when following this approach. In this thesis, we make use of the Kalman Filter algorithm, a topic which will be covered in Section 3.4.3.

3.4.2 A Deterministic Approach

We may leverage the deterministic properties of the UAV in order to estimate the its states. Neglecting process noise (e.g., a sudden gust of wind) for the moment, we may estimate the state of a vehicle by constructing a full-order model of the plant dynamics.

$$\dot{\hat{x}} = F\hat{x} + Gu \tag{3.4.3}$$

Note that we could use this model to estimate the state of the vehicle at any time if the initial state of the system is attainable. However, if the initial conditions are not correct or the model is inaccurate, then the estimated states will diverge from the true states of the system. In order to avoid this

divergence, we may study the dynamics of the error in our state estimates. First, we choose to define the error in the estimate to be

$$\tilde{x} = x - \hat{x} \tag{3.4.4}$$

Where $\tilde{x}$ is the error, x is the true state, and $\hat{x}$ is the estimate of the state.Then the dynamics of the error can be found by substituting equation 3.4.3 and the model for the true dynamics of the system into the time derivative of equation 3.4.4.

$$\dot{\tilde{x}} = F\tilde{x} \tag{3.4.5}$$

For a stable F, our error will converge to zero, but the rate of convergence will depend on the natural dynamics of the system. In order to attain faster estimates, we feedback the difference between the measured and estimated outputs. This scenario is depicted in Figure 3.13.

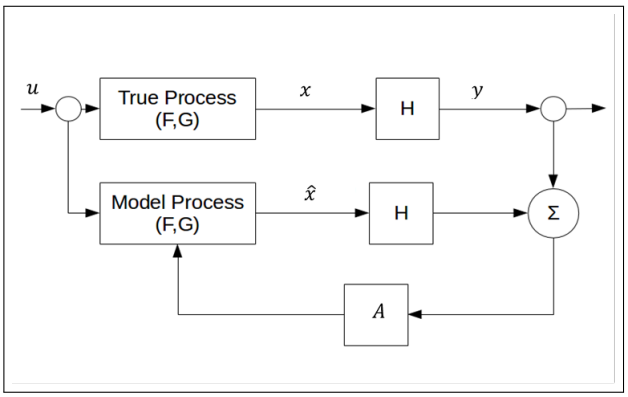


Figure 3.13: Deterministic Estimator (Franklin *et al.*, 2010)

The equation describing the dynamics of the estimated states can then be formulated as done below.

$$\dot{\hat{x}} = F\hat{x} + Gu + A(y - H\hat{x}) \tag{3.4.6}$$

We can again obtain the equation for the dynamics of the estimate error like before, this time replacing equation 3.4.3 with equation 3.4.6.

$$\dot{\tilde{x}} = (F - AH)\tilde{x} \tag{3.4.7}$$

Now we may choose A so that $F - AH$ has stable and reasonably fast eigenvalues. Note that we have assumed that F, G and H are identical in the physical plant and the computer implementation of the estimator. In addition, we have assumed no noise to be present during the measurements. To

address the first assumption, we choose to represent the true plant dynamics with an additional term which accounts for unmodeled dynamics as well as possible process noise.

$$\dot{x} = Fx + Gu + G_1 w \tag{3.4.8}$$

Furthermore, to address the second assumption, we represent the measurement equation with a sensor noise term.

$$y = Hx + v \tag{3.4.9}$$

The estimator error equation with these additional inputs becomes

$$\dot{\tilde{x}} = (F - AH)\tilde{x} + G_1 w - Av \tag{3.4.10}$$

Notice that the sensor noise is multiplied by A but the process noise is not. Therefore, if A is very small, the effect of the sensor noise is removed but the estimators dynamics will be too slow. This means that the estimate error will not reject the effects of w very well and so the estimator would then not perform well in an uncertain plant. Conversely, if A is large, the estimator response will be fast and process noise will be rejected, but there will be large errors due to sensor noise. Choosing the gain A is clearly a trade-off.

3.4.3 The Kalman Filter

In the previous two subsections, two approaches were described for estimating the states of a dynamic system. Each approach had its caveats: in the stochastic approach, it was alluded that there are multiple ways to combine measurement data from multiple sensors; in the deterministic approach, it was described how the computation for the gain of the feedback A must balance the dynamic response of the estimator and the bandwidth of the measurement noise. The Kalman filter is a state estimation algorithm which removes both difficulties by exploiting both state estimation approaches simultaneously. It is an iterative algorithm which harnesses stochastic properties of the sensors and other process noise in order to compute an optimal feedback gain K for the deterministic estimator.

The elements of the filter are described below and then a new subsection will be dedicated to the derivation of the filter equations. The algorithm for implementing the Kalman filter will also be provided thereafter. We note that some of the notation used in this section will be different from that used in Section 3.4.2 to account for the fact that the Kalman filter is implemented in a digital computer where discrete time approximations must be used. For example, the gain notation A has been replaced with the symbol K. Now with a road map of this section in hand, the elements of the Kalman filter are summarized.

(a) The state vector(x): A set of parameters (states) describing the system.

(b) Error covariance matrix(P): Represents uncertainties in state estimates and the degree of correlation between those errors in the estimates.

(c) The system model(Φ): This describes how the Kalman filter states vary with time. We have replaced F to explicitly remind the reader of the fact that we are working with its discrete analog.

(d) The system noise covariance matrix (Q) This describes the statistics of the process noise and also how the estimation error covariance matrix varies with time.

(e) The measurement vector(z): A set of simultaneous measurements of properties of the system. We have replaced y to explicitly remind the reader of the fact that we are working with its discrete analog.

(f) Measurement noise covariance (R): This describes the statistics of noise on the measurements.

(g) The measurement model(H): Describes how the measurement vector varies as a function of the true state vector in the absence of measurement noise. This matrix remains the same regardless of a continuous or discrete state formulation, and so it retains the same symbol.

3.4.4 Derivation of Kalman Filter Equations

This subsection derives the matrix equations which are recursively evaluated during the Kalman filtering algorithm. Consider the case where the UAV contains no control action. Descritizing equation 3.4.8, we find the equation for the plant dynamics.

$$x_k = \phi_k x_{k-1} + w_k \tag{3.4.11}$$

Furthermore, descritizing equation 3.4.6, we find the dynamics of our state estimate.

$$\hat{x}_k = \phi_k \hat{x}_{k-1} + K_k(z_k - H\phi_k \hat{x}_{k-1}) \tag{3.4.12}$$

Where

$$z_k = Hx_k + v_k \tag{3.4.13}$$

Lastly, descritizing equation 3.4.4 and substituting the above descritizations, we find the dynamics of the state estimate error.

$$\tilde{x}_k = \phi_k x_{k-1} + w_k - \phi_k \hat{x}_{k-1} - K_k(H\phi_k x_{k-1} + Hw_k + v_k - H\phi_k \hat{x}_{k-1}) \tag{3.4.14}$$

We simplify this equation by combining similar terms and making use of the definition for the state estimate error.

$$\tilde{x}_k = (I - K_k H)\phi_k \tilde{x}_{k-1} + (I - K_k H)w_k - K_k v_k \tag{3.4.15}$$

We have essentially arrived at the discretized version of equation 3.4.9, and are faced with the same dilemma of choosing the gain which does not over amplify the measurement noise, but at the same time is also not so low that it does not sufficiently raise the dynamics of the state error. To derive K_k, we first define the covariance of the error.

$$\begin{aligned} P_k &= E[\tilde{x}_k \tilde{x}_k^T] \\ &= E[(I - K_k H)\{\phi_k \tilde{x}_{k-1}\tilde{x}_{k-1}^T \phi_k^T + w_k w_k^t\}(I - K_k H)^T + K_k v_k v_k^T K_k] \end{aligned} \tag{3.4.16}$$

Furthermore, we make the following definition so that we can more compactly write the error covariance.

$$M_k = \phi_k P_{k-1} \phi_k^T + Q_k \tag{3.4.17}$$

Where $Q_k = E[w_k w_k^T]$. We can then formulate the error covariance in its general form.

$$P_k = (I - K_k H)M_k(I - K_k H)^T + K_k R_k K_k^T \tag{3.4.18}$$

Where $R_k = E[v_k v_k^T]$. We now choose to resolve our previous dilemma of choosing K by opting to choose it in such a way that it minimizes the variance of the error in our state estimates. Furthermore, we know that these variances appear on the diagonal of the error covariance matrix. We then choose to derive the trace of equation 3.4.18 with respect to the gain K_k, set the result to zero, and solve K_k.

$$K_k = M_k H^T (H M_k H^T + R_k)^{-1} \tag{3.4.19}$$

Finally, we may substitute this optimal gain into the general error covariance equation.

$$P_k = (I - K_k H)M_k \tag{3.4.20}$$

Equations 3.4.17, 3.4.19, and 3.4.20 make up the three equations which are evaluated in each iteration of the Kalman filtering algorithm.

3.4.5 Steps of the Kalman Filter Algorithm

Implementing the equations of Section 3.4.4 is most easily done by splitting the Kalman filter algorithm into two phases. We will refer to these phases as *system propagation* and *measurement update*. These two phases make up 10 steps per iteration. Each of these phases is described below, and the steps of the algorithm are provided thereafter.

During system propagation, known properties of the system are used to predict forward the state vector and error covariance matrix from the last point of validity (the last measurement) to the current time which corresponds to the new set of measurements. The first two steps of the Kalman filtering algorithm are to calculate the deterministic system model and the system noise model (note that these two steps may only need to be evaluated once in the case of a linear system, but this is not the case with a UAV). The third step is to use these models to to bring the state vector up to date. The fourth step is to evaluate M_k using equation 3.4.17. These four steps conclude the system propagation phase.

The measurement update phase serves to incorporate the new measurement information in order to update the state vector estimate and error covariance estimate. The fist two steps of this new phase (steps 5 and 6 in the overall algorithm) are to calculate the deterministic and noise parts of the measurement model. The seventh step, *gain computation*, calculates the Kalman gain matrix using equation 3.4.19. This is used to optimally weight the correction to the state vector. The gain is calculated based on the the uncertainty of the current state estimates and the amount of noise in the measurements. The eighth step formulates the measurement vector. The ninth step, *measurement update*, updates the state estimates to incorporate the measurement data weighted with the Kalman gain. Finally, the error covariance matrix is updated to account for the new information that has been incorporated into the state vector. The algorithm is provided below.

1. Calculate the transition matrix, Φ.
2. Calculate the system noise covariance matrix, Q.
3. Propagate the state estimate vector from $\hat{x}_{k-1}^{+}$ to $\hat{x}_{k}^{-}$.
4. Calculate M_k.
5. Calculate the measurement matrix, H_k.
6. Calculate the measurement noise covariance matrix, R_k.
7. Calculate the Kalman gain matrix, K_k.

8. Formulate the measurement, z_k.
9. Update the state vector estimate from $\hat{x}_k^-$ to $\hat{x}_k^+$.
10. Update the error covariance matrix from P_k^- to P_k^+.

Figure 3.14 below presents the final component of the virtual test bench, the Kalman filter. It embeds the algorithm which was presented above into a block based simulation model which is supported by sT Activate (see Section 3.1). Note that some of the blocks represent a superblock which themselves contain submodels. The submodels of Figure 3.14 are outlined by dotted borders.

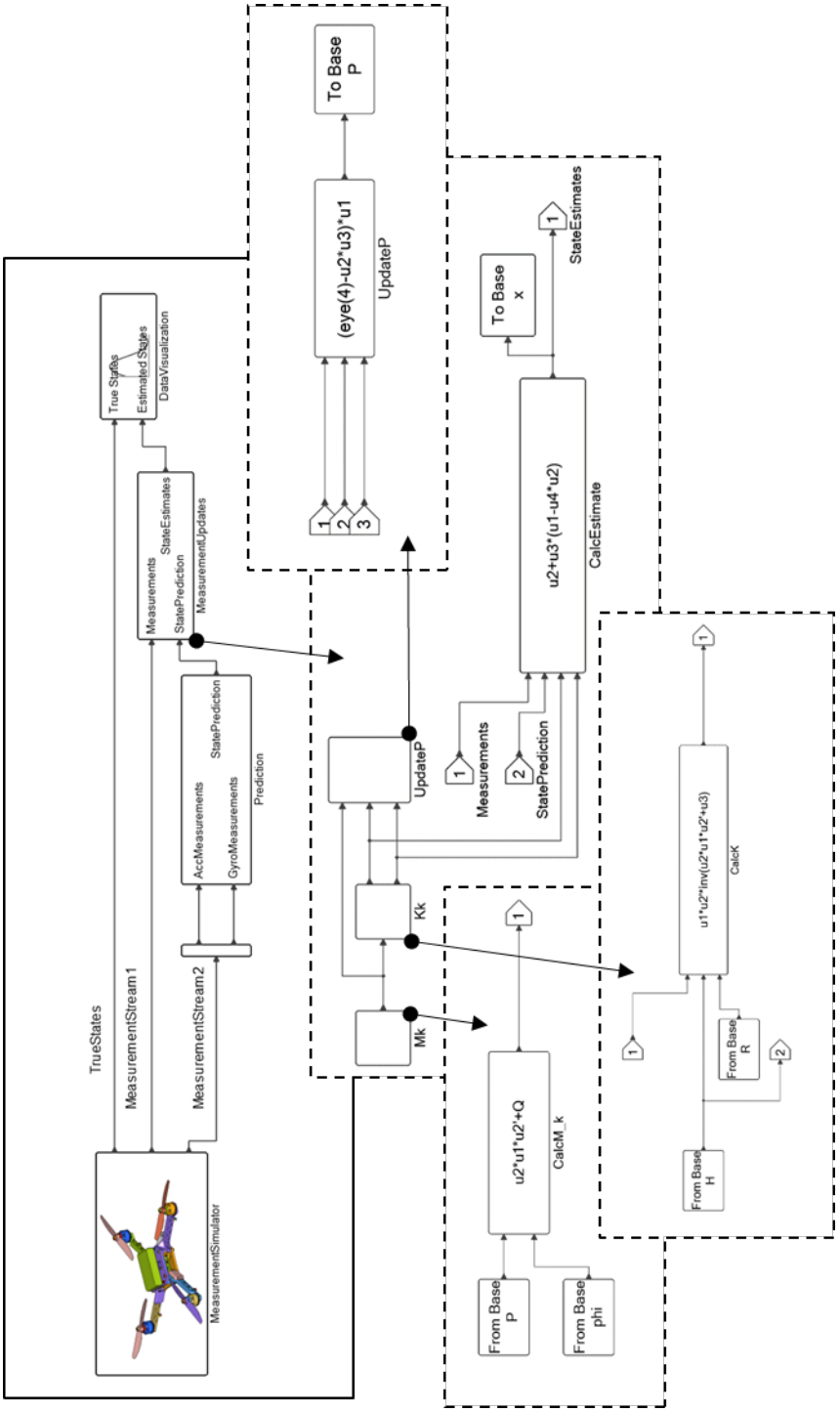


Figure 3.14: Kalman Filter

3.5 Conclusion

The focus of this chapter has been to establish a virtual test bench which can be modified in order to resolve the sources of inaccuracy that were previously identified (see Chapter 2) in quadcopter state estimation. The software infrastructure which is used was described in Section 3.1 and consists of a multi-body dynamics solution (MotionSolve), a block based modeller (sT Activate), and a pre-processor (MotionView) which compliments the multi-body dynamic solution. The multi-body dynamics solution is used to simulate *truth* data, while the plant dynamics models which have been developed in Section 3.3 constitute a metamodel of these plant dynamics. This metamodel is implemented within the Kalman filter as a part of the *system propagation* phase. The block modeling paradigm supported by sT Activate was used to implement the *measurement update* phase of the Kalman filter. This chapter has documented the fulfillment of the fourth objective of the project, and the next two chapters will modify the baseline test bench which was developed in order to fulfill the remaining objective.

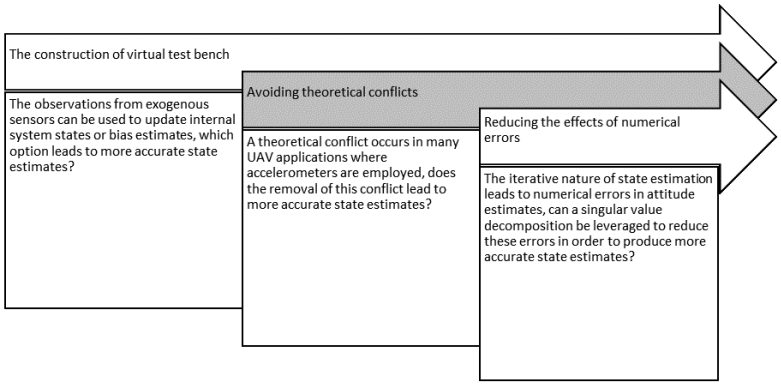
Chapter 4

The Resolution of Theoretical Conflicts

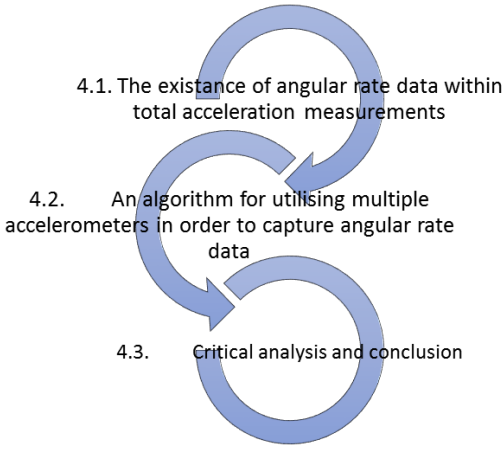
Accelerometers form the basis of many UAV navigation systems, and are often used as tilt sensors. In this scheme, the gravity vector is shared among the three axes of an accelerometer triad, with the weighting factor on each axis being a function of the body orientation. Consequently, it is possible to use inverse trigonometric functions to obtain the body orientation from accelerometer measurements. Unfortunately, it must be assumed that the UAV itself has no acceleration if this scheme is used. A pragmatic solution to deal with this is to pass the accelerometer measurements through a low pass filter.

On the other hand, it can be useful to integrate acceleration data in order to estimate the velocity or position of a body. The difficulty here is that the orientation of the UAV must be known so that the gravity vector may be removed from the accelerometer measurements. The common approach to deal with this difficulty is to integrate gyroscope signals in order to obtain UAV orientation. It is fairly common, however, to ignore the theoretical violation and use accelerometers for both orientation and positional updates. The results achieved in practice are acceptable for low fidelity applications. In this chapter, the authors propose augmenting an accelerometer triad with two more accelerometer triads and show how the measurements can be used to obtain attitude rate data which can be integrated towards orientation. The result is a resolution of the theoretical conflict which has been described (the simultaneous assumption that the body is not accelerating and that it is indeed accelerating). Figure 4.1a provides the reader with a bookmark for the current chapter within the context of the thesis, and Figure 4.1b provides the contribution of each section to the overarching goal of this chapter (to resolve a common theoretical conflict by utilising multiple accelerometers in

order to detect the angular rate of a UAV).



(a) Chapter 4 is represented by the darkened section of the image and forms the second of the three main pillars which make up the thesis.



(b) Chapter 4 is modularised into the sections shown.

Figure 4.1: A depiction of the context and content of Chapter 4

4.1 Measurement Model

The goal of this section is to convince the reader that pertinent angular rate data resides in the measurements taken from accelerometers. We begin our argument by introducing the general equation for acceleration in a non-inertial frame of reference. We continue to show, theoretically, how angular

rate data may be filtrated from the difference in synchronized accelerometer readings when we apply this equation to the measurements. This theory will be further leveraged in Section 4.2 to solve the angular rate of our inverted pendulum without the need for integration or redundant accelerometers. Consider Fig. 4.2 during the explanation of the theory that follows.

The measurements taken from an accelerometer will encapsulate the following terms if not placed on the centre of mass of the object under investigation (only the first term remains if the accelerometer is placed at the centre of mass).

$$a_T = \ddot{R} + (\dot{\omega} \times \rho) + [\omega \times (\omega \times \rho)] + \ddot{\rho}_r + 2\omega \times \dot{\rho}_r \tag{4.1.1}$$

where a_T is the total acceleration measured. The various terms have the following physical interpretations. The first term ($\ddot{R}$) is the inertial acceleration of **O'**, the origin of the non-inertial frame. The second term ($\dot{\omega} \times \rho$) can be considered as a tangential acceleration (it is a function of the angular acceleration $\dot{\omega}$ and the vector connecting the accelerometer and the body centre of mass ρ). The third term ($\omega \times (\omega \times \rho)$) is a centripetal acceleration directed toward an axis of rotation through **O'**. These first three terms constitute the acceleration of point P if it were fixed in the xyz frame. The last two terms modify the acceleration equation so that it captures the kinematics of point P when it is not fixed within the xyz frame. Specifically, the fourth term ($\ddot{\rho}_r$) is the acceleration of **P** relative to the xyz frame, that is, the acceleration of the particle, as recorded by instruments fixed in the xyz frame and rotating with it. The final term ($2\omega \times \dot{\rho}_r$) is the Coriolis acceleration due to a velocity relative to the rotating frame Greenwood (2006).

Notice that the second and third terms house information regarding the angular rate of the object. These two terms are also perpendicular. Then

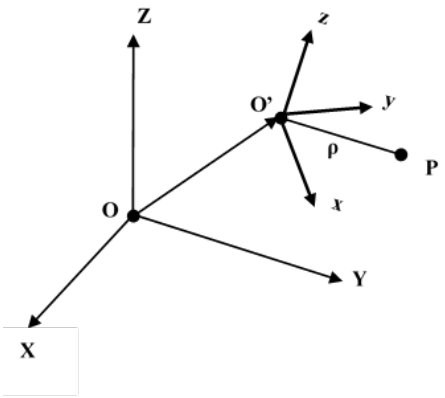


Figure 4.2: General Acceleration Coordinate Frames

the last two terms are only non-zero if the accelerometer moves relative to the body frame while the body is itself accelerating. Let us now assume that two triads of accelerometers are fixed to a moving rotating body but are not situated at the same location. Their axes are, however aligned. This assumption is only made for ease at this stage as the axes can be digitally aligned through a calibration process so that they do not need to be physically aligned. The accelerations measured by the devices will therefore comprise the first 3 terms of the equation 4.1.1, resolved in the axes of the measurement system. The same component will populate the first term of both triads, so using simple subtraction:

$$\Delta a_m = \dot{\tilde{\omega}}(\rho_1 - \rho_2) + \tilde{\omega}(\tilde{\omega}(\rho_1 - \rho_2)) \tag{4.1.2}$$

Where $\Delta a_m \in R^3$. We have replaced the cross product with the equivalent skew symmetric matrix $\tilde{\omega}$. The difference measured by the two accelerometers would comprise the accelerations which are due to the tangential and centripetal accelerations of the body. We know that these two components are perpendicular, as mentioned earlier. The differential measurement at a single epoch can therefore be spanned by a 2-dimensional subspace.

$$\Delta a_m = span\{u_1, u_2\} \tag{4.1.3}$$

Projecting the 3D measurements onto this 2-dimensional subspace:

$$\Delta a_m = |\dot{\tilde{\omega}}(\rho_1 - \rho_2)|u_1 + |\tilde{\omega}(\tilde{\omega}(\rho_1 - \rho_2))|u_2 = t1 + t2 \tag{4.1.4}$$

we can now use this projection to solve the angular rate of the body algebraically:

$$\omega = \sqrt{\frac{t2}{|(\rho_1 - \rho_2)u_2|}} \tag{4.1.5}$$

4.2 Algorithm for solving Angular Rate

The theory provided in Section 4.1 is applied to solve the angular rate of the UAV. In order to do this, a method to project accelerometer reading onto a lower dimensional subspace is needed. A suitable method is also required to identify which of the new basis vectors are aligned with the UAV's centripetal acceleration. We address each of these difficulties in this section.

The singular value decomposition (SVD) is the perfect tool for finding the basis vectors which span a data set, and hence address the first of the difficulties previously raised. This mathematical tool will be revisited in

Chapter 5, and so in order to avoid duplication, the reader is urged to consult it or more elaborate texts such as Poole (2011) for details. For now, it is only necesarry to knopw that the SVD can be used to find orthogonal data sets in highly dimensional data. Thus, the algorithm for projecting Δa_m onto a 2D subspace can be summarized:

- Make 3 independent and synchronized 3D observations of the point accelerations at three distinct points on the body.
- Subtract the acceleration measurement of one accelerometer from the other for each independent but synchronized observation.
- Assemble a matrix from the vectors obtained from the above steps

$$X = \begin{bmatrix} \Delta a_{1-2} & \Delta a_{2-3} & \Delta a_{1-3} \end{bmatrix} \tag{4.2.1}$$

- Perform a SVD on the observations matrix X to obtain the 2-dimensional subspace which spans the centripetal/ tangential acceleration pair

$$X = U\Sigma V^* \tag{4.2.2}$$

Where

$$U = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \tag{4.2.3}$$

$$\Sigma = \text{diag}(|\dot{\tilde{\omega}}(\rho_1 - \rho_2)|, |\tilde{\omega}(\tilde{\omega}(\rho_1 - \rho_2))|, 0) \tag{4.2.4}$$

Note that $|(\rho_1 - \rho_2)| = |(\rho_1 - \rho_3)| = |(\rho_2 - \rho_3)|$ and that V^* is not necessary, which is why its value is not provided here. Unfortunately, the order of the vectors in U and the corresponding singular values in Σ depend on which component has the larger magnitude (hence, it becomes a challenge to identify the centripetal acceleration). We may, however, exploit our a priori knowledge of the system motion to identify the centripetal and tangential basis vectors in U. In Chapter 3, it was shown that the state space of a UAV is constrained. In other words, there are only certain possible motions that would propagate the UAV's state at one epoch to a state at another. These permissible motions are governed by the equations that describe the dynamics of the vehicle. Theoretically speaking, it is possible then to combine the state estimate of the UAV at the most recent epoch with the UAV's equations of motion in order to predict what the tangential and centripetal accelerations might be at the next epoch. This prediction can be used to establish a relationship between the tangential/centripetal accelerations i.e. Should the tangential acceleration have a larger magnitude, or is the centripetal acceleration larger? Finally, this relationship can be applied to the actual differential

acceleration measurements in order to identify the two components. The remainder of this section applies this principal to the specific case of a UAV, completing the algorithm which has thus far been general in its development (i.e. the algorithm may be applied to other dynamic systems as shown by Minnaar and Smit (2017)).

The centripetal and tangential acceleration components of the UAV will be constantly changing in direction and magnitude. The driving force behind this change is the thrust which acts upon the UAV (ignoring the effects of all other forces for simplification). The following heuristic algorithm leverages this physical fact in order to identify the centripetal acceleration in the differential measurements and use it to solve the angular rate of the UAV algebraically:

- Feed commanded thrust into equation 3.3.9 to predict magnitude and direction of centripetal and normal components in following time step.
- Resolve the magnitude and direction of the above into the accelerometer coordinate system
- The column of U that corresponds with the largest singular value should be identified as the acceleration component which had the largest magnitude in the prediction. Leverage this fact to screen the singular values and pick out the one corresponding to the centripetal acceleration.
- Use equation 4.1.5 to solve the angular rate of the UAV algebraically
- Correct the handedness of the angular rate estimates by comparison with the predictions

It is important to note that the measurements which will be taken in a physical system will contain some degree of noise. This is an important eventuality to replicate in the simulation environment since it effects the robustness of the relationship screening step in the algorithm above. Consequently, the two triads of virtual accelerometers which are simulated in this work are contaminated with noise corresponding to a power spectral density of $400\frac{\mu g}{\sqrt{Hz}}$ (MPU-9250 Product Specification). The axes of the accelerometer triads are aligned and placed randomly on the virtual UAV with a known spatial vector between them (they could be randomly orientated with respect to one another, as this misalignment could be calibrated). Lastly, it will be necessary to adapt the Kalman filter in order to include the navigation scheme proposed here. Figure 4.3 illustrates the flow chart that encapsulates the suggestions which were made in this chapter. Notice that the low pass filtering scheme is not abandoned altogether, but that scheme shall only be used in maneuvers with low dynamics.

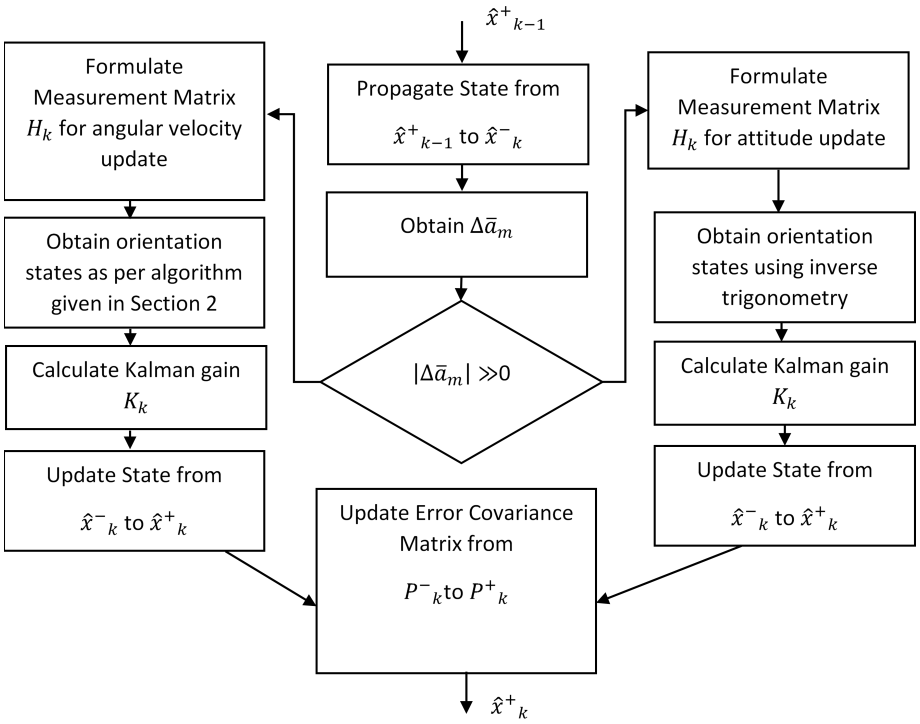


Figure 4.3: Kalman Filter Structure with proposed redundant accelerometer scheme

4.3 Conclusion

In this section, an algorithm was suggestion in which three or more accelerometers that are equidistant on the UAV may be used to algebraically determine the angular rate of a UAV. Results for simulations which are performed to validate the algorithm are presented in Chapter 6. An iteration of the main findings in this Chapter is now provided.

Firstly, it was alluded that if an accelerometer itself is not accelerating and that it is arbitrarily tilted, the device measures gravitational acceleration distributed among the three axes that makeup the accelerometer triad. It is thus possible to use inverse trigonometry to find out from the measurement data what the orientation of the device is. Although this is an elegant solution for finding the orientation of the vehicle with accelerometers when zero acceleration is assumed, it cannot be used during dynamic maneuvers. It is possible to lessen the effect of body acceleration by passing the accelerometer measurements through a low pass filter with a very low cutoff frequency and then still applying the logic which was just discussed, but it is not an elegant

solution (although it works rather well in practice).

To help overcome the theoretical conflict that arises when using accelerometers for orientation estimation despite non-zero accelerations, it is common to employ gyroscopes which allow one to observe the angular rate of the body in question. These signals may then be integrated towards absolute orientation. Gyroscopes do pose some disadvantages which accelerometers do not though (e.g., high cost, low operating range and growing biases).

This chapter explored an alternate possibility in which accelerometers also be used to find the orientation of the body but without violating any of the theoretical assumptions previously. Towards this end, it was argued that the total acceleration which is measured by accelerometers is made up of 5 components. Of particular interest were the tangential and centripetal accelerations which are the cause of placing an accelerometer off center. The components were interesting because they are both functions of orientation derivatives. It was argued that if these terms could be isolated and exposed, that it would be possible to integrate towards body orientation.

An algorithm was developed in order to perform the acceleration component isolation and exposure programatically (the authors identified in Chapter 2 performed the isolation physically with particular accelerometer layouts). The method involves the comparison of three different accelerometer triad measurements with one another. The accelerometers, which are placed at different locations around the UAV's centre of mass but are equidistant from one another, produce unique measurements that differ only by their centripetal and tangential components. Utilising the Singular Value Decomposition makes it possible to isolate and expose these components, which can be subsequently integrated towards an orientation solution.

Chapter 5

Reducing Numerical Errors

This chapter will detail the numerical effects which get propagated in parameterizing the orientation of the UAV. First it is shown that angular rate data is a defined quantity and not the derivative of some other function. For this reason it is necessary to propagate a DCM (as opposed to integrating it). Numerical errors arise during this propagation, and these numerical errors ultimately destroy the orthogonality of the DCM. A solution for the eventuality is then provided through the use of the singular value decomposition.

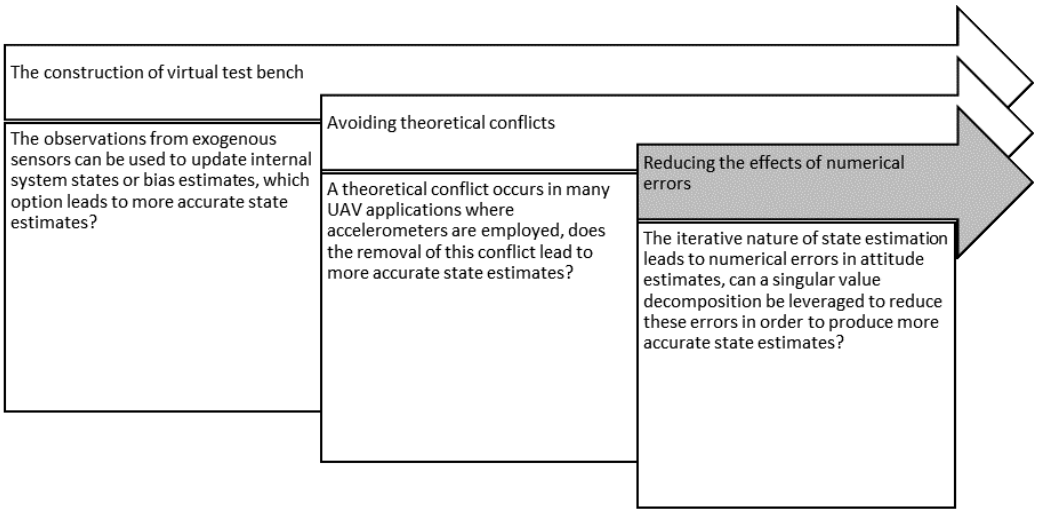


Figure 5.1: Chapter 5 is represented by the darkened section of the image and forms the third of the three main pillars which make up the thesis.

5.1 Kinematics of Rotational Motion

This section develops the kinematic and dynamic relationships which are necessary for the purposes of this thesis.

We highlight an important consequence of our definition for matrix multiplication - it does not possess the property of commutativity. It should be apparent then that the DCM and rotation matrix for different rotation sequences will not be equal (contemplate equation 3.2.12 and what happens if the order of multiplication changes). The result of this is that coordinate systems that have been rotated by finite angles cannot be represented as vector operations. In turn, the implication is that there is no vector which can be differentiated in order to obtain the rotation rates. This is contrary to a position vector which can be differentiated with respect to time in order to obtain a position rate.

We address the shortcomings of our orientation parameterization by analysing infinitesimal rotations. Consider the following equation, which is the result of invoking the *small angle assumption* to simplify equation 3.2.13.

$$\mathbf{C}_{321} = \begin{bmatrix} 1 & \delta\theta & -\delta\psi \\ -\delta\theta & 1 & \delta\phi \\ \delta\psi & -\delta\phi & 1 \end{bmatrix} \tag{5.1.1}$$

where δ represents an infinitesimal such that $\delta\theta$, $\delta\phi$ and $\delta\psi$ are all infinitesimal angles. It is a remarkable fact that by simplifying the DCM for any rotation sequence by invoking the small angle assumption, we will obtain the result in equation 5.1.1(Infinitesimal rotations are commutative). We can now find a vector to express infinitesimal rotations, and their rates. We start by manipulating equation 5.1.1:

$$\mathbf{C}_{321} = \mathbf{I} - \begin{bmatrix} 0 & -\delta\theta & \delta\psi \\ \delta\theta & 0 & -\delta\phi \\ -\delta\psi & \delta\phi & 0 \end{bmatrix} \tag{5.1.2}$$

It follows that

$$\begin{aligned} \mathbf{C}_{321}^{T} &= \mathbf{I} + \begin{bmatrix} 0 & \delta\theta & -\delta\psi \\ -\delta\theta & 0 & \delta\phi \\ \delta\psi & -\delta\phi & 0 \end{bmatrix} \\ &= \mathbf{I} + \left[\Delta\vartheta\right] \end{aligned} \tag{5.1.3}$$

Now we consider the coordinate vector of a point as its position is changed through a rotation. We will consider two coordinate frames, one of which is the reference frame and the other which been rotated an infintesimal as the object changed position. We will denote the coordinate vector with a

superscript $'$ in the rotated(object) coordinate frame and no superscript in the reference coordinate frame. Furthermore, the subscripts i and f denote initial and final positions. Contemplate Figure 5.2 to convince yourself that $\overline{x}'_f = \overline{x}_i$.

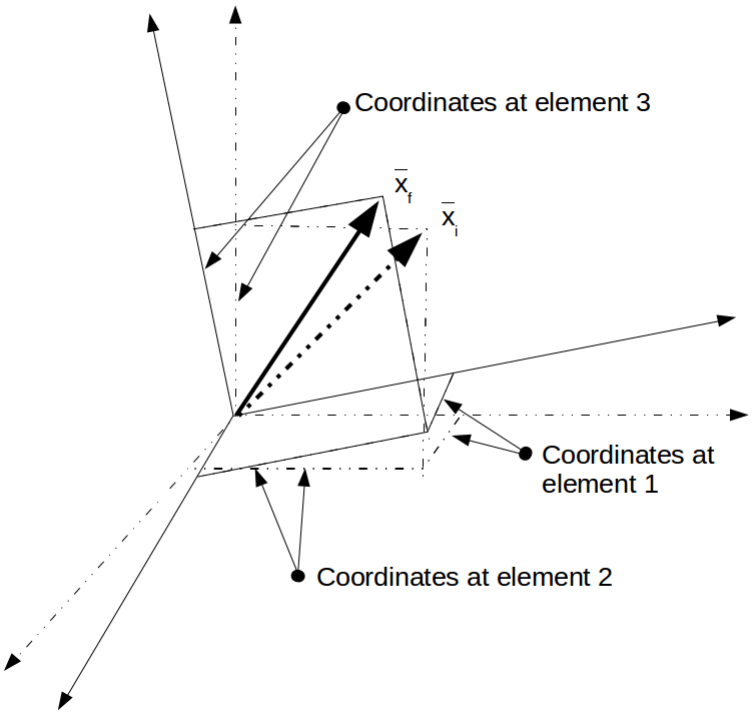


Figure 5.2: The corresponding coordinates of a point in a coordinate frame and an infintesimally rotated frame

Then we may define the change in coordinates as

$$\overline{\Delta x}' = \overline{x}'_f - \overline{x}'_i = \overline{x}_i - \overline{x}'_i \tag{5.1.4}$$

Then by noting that $\overline{x}_i = \mathbf{C}^T \overline{x}'_i$, we may simplify equation 5.1.4 as done below.

$$\overline{\Delta x}' = (\mathbf{I} + \left[\Delta\vartheta\right])\overline{x}'_i - \overline{x}'_i = \left[\Delta\vartheta\right] \overline{x}'_i \tag{5.1.5}$$

We may choose to write equation 5.1.5 as a cross product operation if we recognize that it is a column vector representation. This is shown below.

$$\overline{\Delta x}' = \overline{\Delta\vartheta} \times \overline{x}' \tag{5.1.6}$$

where

$$\overline{\Delta\vartheta} = \overline{\Delta\vartheta}_1 e'_1 + \overline{\Delta\vartheta}_2 e'_2 + \overline{\Delta\vartheta}_3 e'_3 \tag{5.1.7}$$

Since we are interested in the time rate of change for the position vector, we divide equation 5.1.7 by Δt and take the limit as $\Delta t \to 0$. We define the resulting quantity as the angular velocity.

$$\begin{aligned} \overline{\omega} &= \lim_{\Delta t \to 0} \frac{\overline{\Delta\vartheta}}{\Delta t} \\ &= \omega_1 e'_1 + \omega_2 e'_2 + \omega_3 e'_3 \end{aligned} \tag{5.1.8}$$

Finally, we may calculate the time rate of change for a position vector due to a rotation.

$$\dot{\overline{x}}' = \overline{\omega} \times \overline{x}' \tag{5.1.9}$$

We emphasize that the angular velocity is a defined quantity and is not the derivative of another vector.

5.2 Orthonormalization

Let D_0 be the exact DCM. Now let S be the set of all orthonormal matrices in $R^{3\times3}$ and A is simply all matrices from $R^{3\times3}$ (orthonormal or not). D_0 will belong to S while the DCM that we compute from equation 5.1.9 will give us a matrix that lies within A (let us use X' to denote the matrix which is the result of equation 5.1.9). We would like to find the best approximation of D_0 which is also in S by orthonormalizing X'. Since there isn't a unique way of finding such a matrix, we introduce some mathematical rigor and choose to find $\overline{X}$ so that it minimizes $\|D_0 - X\|$ for all $X \in S$. We formulate this objective mathematically.

$$\min_{X\in S} W \triangleq \min_{X\in S} (\sum_i \sum_j w_{ij}^2)^{0.5} \tag{5.2.1}$$

Where

$$W = N(D_0 - X) \tag{5.2.2}$$

And N represents some function which returns the norm of the input matrix. Let us choose N to be the euclidean norm. Then leveraging the definition of matrix multiplication (and recalling that the trace of a matrix is equal to the sum of its diagonal elements), we may reformulate equation 5.2.1.

$$\min_{X\in S} W = \min_{X\in S} tr(W^T W)^{0.5} \tag{5.2.3}$$

Our challenge now is to find the orthonormalizing function which takes X' from $A \in R^{3\times3}$ to $\overline{X}$ in $S \in R^{3\times3}$ while satisfying equation 5.2.3. One solution is to make use of the singular value decomposition, which was introduced in section 5.3. The details of the logical arguments which support the method were described by Mao (1986), who show that the SVD of a matrix produces such an orthonormal solution.

5.3 The Singular Value Decomposition

The singular value decomposition (SVD) factorizes an input matrix into three components, as shown in equation 5.3.1. The application of the decomposition in this thesis was provided in section 5.2 (it was used to orthogonalize the DCM). This section is intended only to provide the fundamental properties of the SVD components and leverage them to provide a physical interpretation of each. The explanations will hopefully augment the analytical argument for "why a SVD should be used to orthogonalize a DCM" (provided in section 5.2) with a physical interpretation.

$$A = U\Sigma V^T \tag{5.3.1}$$

It can be shown that U and V^T are orthogonal (Poole, 2011). Furthermore, Σ is real and square if the input matrix is square with a full rank[1]. Matrices that exhibit these special properties will be of concern later during this section, and so the descriptions which follow are concerned exclusively with this class of matrices. Firstly, in order to aid our visualization of the descriptions which follow, consider a compilation of coordinate points which have been wrapped with a surface to form a perfect sphere which is randomly oriented.

Now consider a square, real and positive determinant matrix to be a transformation which acts upon this collection of wrapped coordinate points. The result is as shown below.

Where in this case we have

$$F = \begin{bmatrix} 0.7725 & 1.6665 & -0.1763 \\ 0.6929 & -0.4093 & 0.2299 \\ 0.1836 & 0.1812 & -1.1348 \end{bmatrix} \tag{5.3.2}$$

In order to gain and understanding of this transformation, the matrix multiplication is segmented according to the SVD of F.

[1]The number of columns in the input matrix must be equal to the column space of the matrix

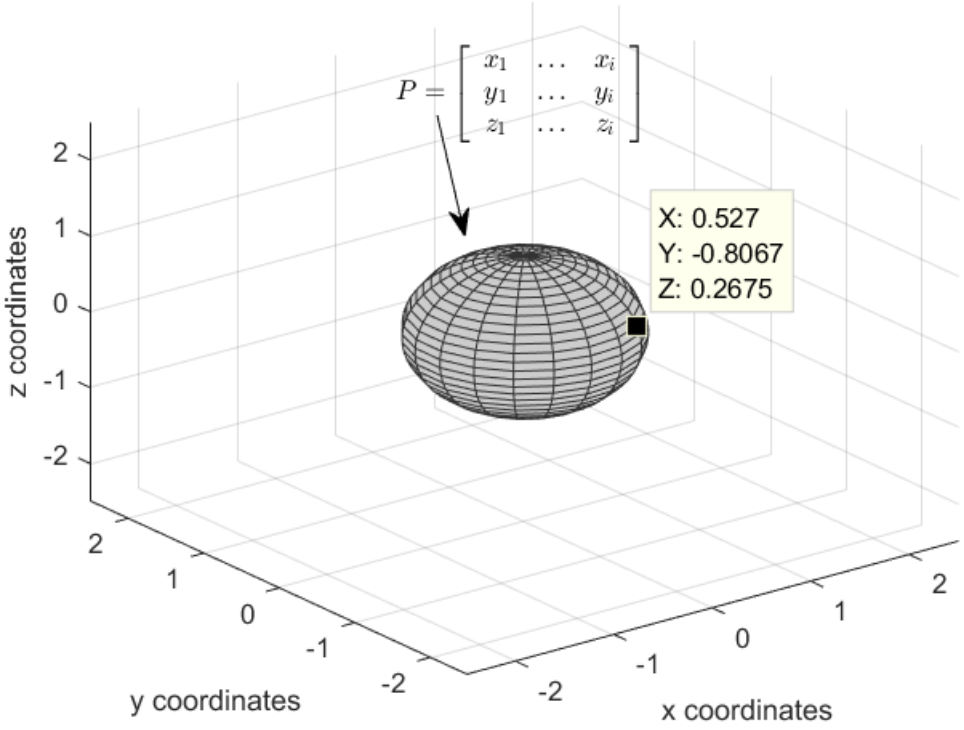


Figure 5.3: Discrete coordinates which have been wrapped with a surface to form a randomly oriented sphere

$$\begin{aligned} P_{transformed} &= UP_2 \\ P_2 &= \Sigma P_1 \\ P_1 &= V^T P \end{aligned} \tag{5.3.3}$$

These equations should be regarded in conjunction with Figure 5.5 which will illustrate the physical interpretations of the matrix transformation. The interpretation is composed of a rotation, followed by a scaling, followed once again by another rotation. The two rotations at the beginning and end of the sequence are unsurprising, since U and V are orthogonal matrices in $R^{3\times3}$ [2]. More specifically, Figure 5.5a depicts a resolution of the coordinate points into an orthonormal basis of the domain associated with the matrix transformation F. Figure 5.5c, on the other hand, depicts a rotation of the

[2]Recall that the rotation matrices (which were developed in Chapter 3) are also orthogonal and contained in $R^{3\times3}$

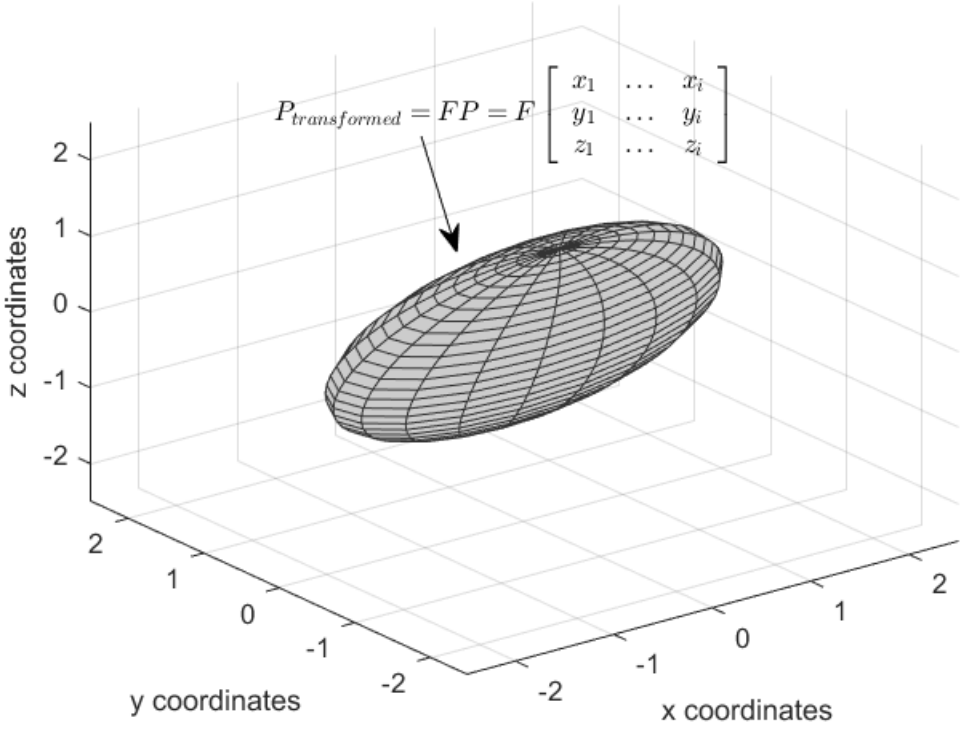
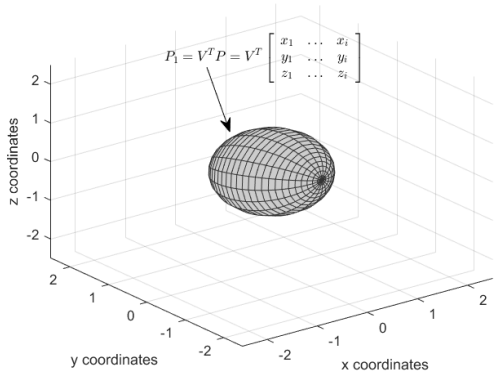


Figure 5.4: A transformation of the surface wrapped discrete coordinates that represent a unit sphere

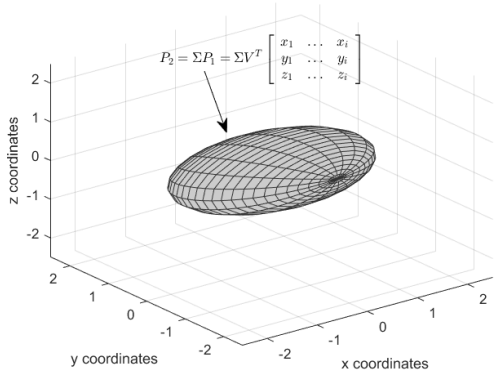
coordinate points to the final image [3] of the coordinate points under the matrix transformation F. Figure 5.5b depicts the intermediate step which scales the coordinate points along the principal axes which form this orthonormal basis.

The geometric interpretation of a matrix transformation which was provided in Figure 5.5 is useful for understanding the effects of perturbations in rotation matrices. Situations like these arise during the numerical propagation of an aircraft's DCM which leads to a loss of orthogonality (see Section 5.1). To understand these perturbations, first assume that the DCM being investigated is orthonormal, as it should be. Applying a SVD to this DCM can be interpreted as follows.

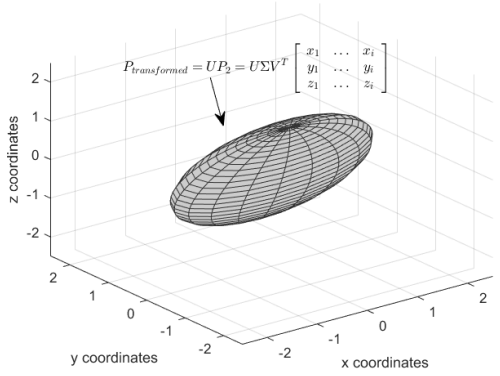
[3] The word "image" here should be understood in the context of linear algebra where it is defined as the unique vector in the codomain of a transformation which is associated with the corresponding input vector



(a) The first matrix multiplication resolves the coordinate points into the principal axes of the matrix transformation



(b) The second multiplication scales the coordinate points along the principal axes of the matrix transformation



(c) The last matrix multiplication rotates the coordinate points, concluding the matrix transformation

Figure 5.5: Singular Value Decomposition of matrix (square and real with positive determinant) transformation

- V^T is a permutation matrix. A linear combination of the columns of a rotation matrix forms an orthonormal basis for its domain. Since the columns have unit magnitude, any permutation of these columns forms the orthonormal basis of the transformation.

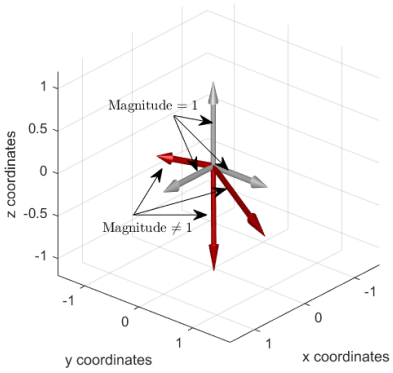
- Σ is an identity matrix since a DCM is orthonormal.

- U is the rotation matrix itself, with its columns having possibly been reordered and negated. The premultiplication of ΣV^T by U is the final mapping which brings the input vector into the final image of the transformation. The result of ΣV^T is simply a permutation matrix according to the previous items of this list.

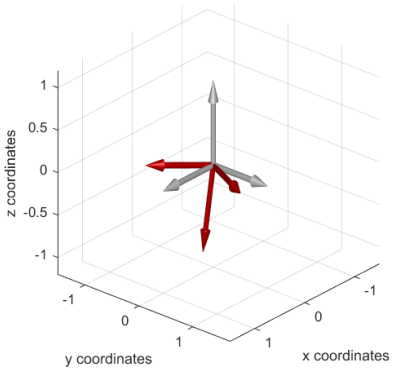
Now the effects of perturbations on a rotation matrix are contemplated in terms of it's SVD elements. The perturbations spoil the orthonormal characteristic of the rotation matrix. Physically this means that the UAV coordinate axes in the image of the rotation transformation have possibly been rescaled and rotated towards or away from one another. This is not physically possible since the UAV is treated as a rigid body. The orientation estimates thus loose accuracy as a result of the perturbations in the rotation matrix. In this thesis, it is proposed that a SVD be performed on the DCM at each iteration where updated orientation estimates are generated and that the DCM be recomposed with it's singular values forced to unity. The effect of this orthonormalization is exhibited in Figure 5.6.

5.4 Conclusion

In this section, it was shown the the DCM looses its orthogonality through the process of DCM propagation. The three state estimates that are particularly sensitive to numerical errors are thus the ones that describe the absolute attitude of the UAV, since these parameters are interpreted from the DCM. Fortunately, there is a priori knowledge to be utilised in restoring the DCM and thereby cleansing it of its numerical errors. These properties can be enforced in the DCM by means of a SVD which breaks a matrix down into the product of three other matrices, each of which tells us something about the source matrix. The simulations which validate the orthonormalized DCM through singular value decomposition are provided in Chapter 6.



(a) The image of the standard basis of $R^{3\times3}$ under a matrix transformation may be elongated/shortened



(b) All vectors remain unity in the image of the standard basis of $R^{3\times3}$ under a matrix transformation which has been orthogonalized

Figure 5.6: The singular values of the transformation matrix are forced to unity when orthogonalizing it according to equation 5.3.1

Chapter 6

Results and Discussion

6.1 Strategies for the use of Exogenous Signal based Sensors

Exogenous signal based sensors such as GPS/barometers are a staple sensor in many outdoor UAV navigation systems. This class of sensors is not mandatory for UAV navigation, however, since many UAVs are able to navigate solely using the measurements from inertial navigation sensors. This raises an important question,"what is the best way to augment an inertial navigation system with exogenous signal based sensors? '' The reason that this question is raised is that there is not a unique way to include the measurement. It is possible to, for example, use an exogenous signal to estimate the bias on an inertial sensor (error state estimation). It is equally possible to use the exogenous signal to update an internal state of the UAV through a Kalman filter (full state estimation). In this section, the two alternatives are explored. Figure 6.1 provides context for the results discussed in this section. It is reiterated that the theory regarding these results is presented in Chapter 3.

Instead of reconstructing the full state of a simulated UAV motion, a single state will be estimated in order to discuss the alternative uses for the exogenous signals of the measurement system. The representative state that is estimated in the following discussion is the positional state pertaining to altitude. Specifically, the virtual test bench is provided with a step input in its altitude reference. The simulation which follows combines a triad of accelerometers, a barometer, a GPS and a metamodel of the system dynamics in a Kalman Filter architecture. The sensors are intentionally corrupted with random noise (all of the sensors) and bias (only the barometer is corrupted with this source of uncertainty) and also provide outputs at varying

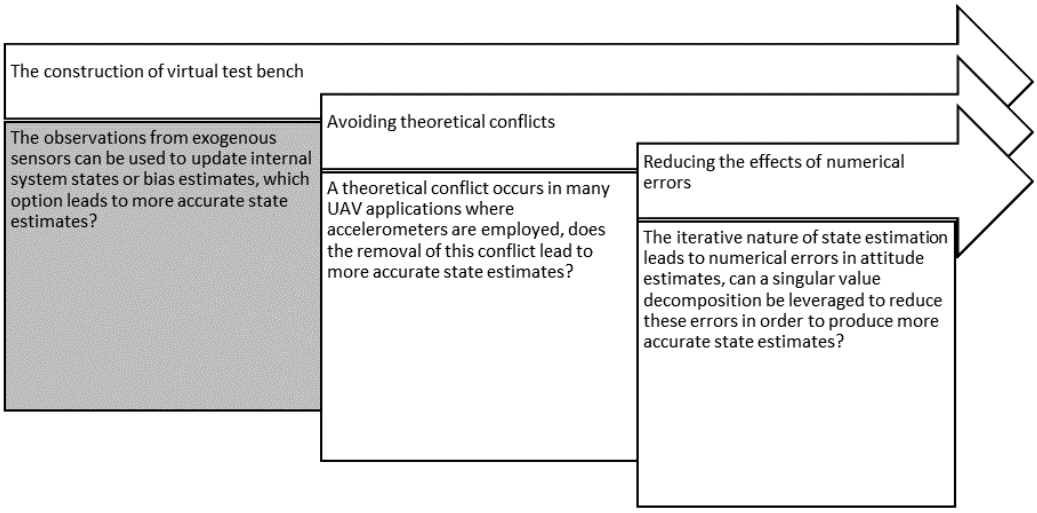


Figure 6.1: Section 6.1 utilizes the theory presented Chapter 3 to present strategies for utilizing exogenous sensors

frequencies. Figure 6.2 depicts the control block which activates the sampling functions on each sensor. From the figure, it can be seen that the accelerometers are sampled most frequently (200Hz) and are used as a reference for activating the other sensors. The GPS is sampled least frequently (5Hz) and the barometer is sampled in between the two extremes (100 Hz). Furthermore, even though the measurements taken by the GPS and barometer are multiples of the accelerometer frequency, the measurements are not synchronized. It was important to simulate this unsynchronized behavior in the measurement stream since in physical implementations, the behavior would certainly occur. Note, however, that the possible loss of GPS signals has not been simulated.

In the case of full state estimation, the GPS signals are used to make observation of the altitude, as are the barometer measurements. Figure 6.3 presents a graph of the ground truth signal alongside the estimated and measured signals. Notice that that the estimates slowly drift away from the true signal due to the fact the the barometer updates include a bias which is not being estimated and that the barometer signals arrive more frequently than the GPS signals do. The parameters of the Kalman filter could be tuned to pronounce or soften this effect, but it will remain to some degree because of the structure of the state estimator (full model state but with no error states).

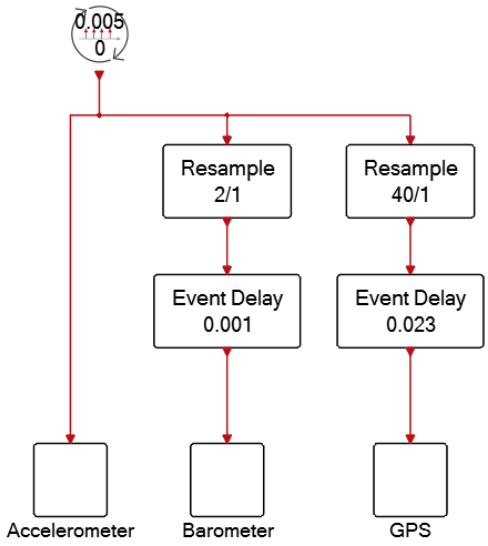


Figure 6.2: A schematic representing the simulated sampling scheme of the sensors

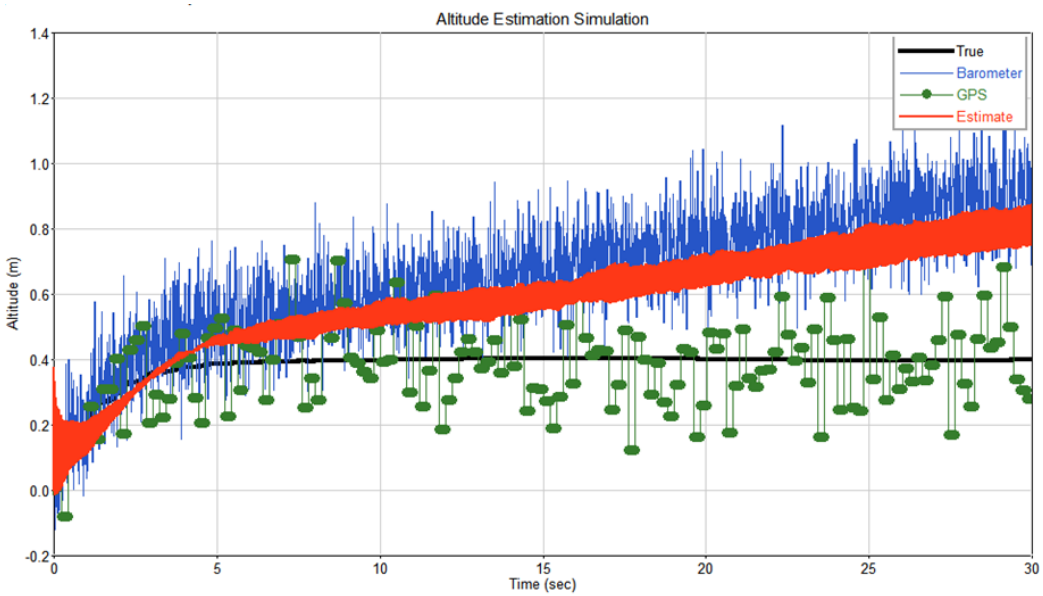


Figure 6.3: Altitude signals for the UAV when given a step reference input (full order estimator)

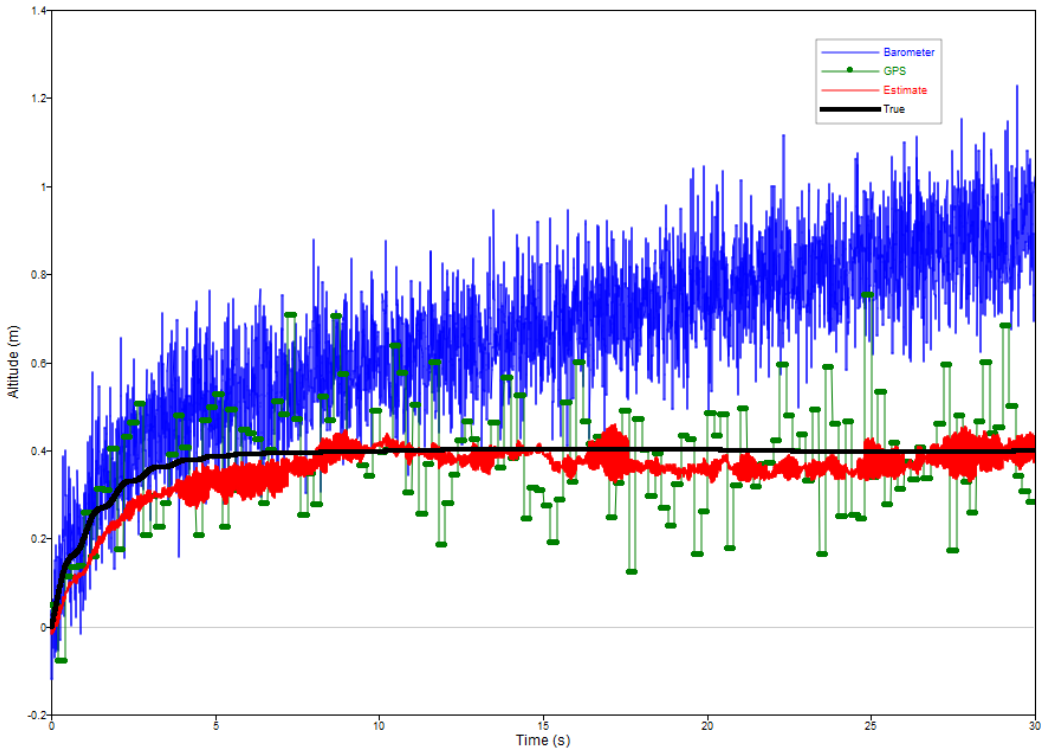


Figure 6.4: Altitude signals for the UAV when given a step reference input (this estimator structure includes an error state)

A second method for augmenting this particular navigation system with GPS measurements is to use them for the generation of estimates in barometer bias. Figure 6.4 presents the altitude state of the virtual UAV (when it is commanded with a step input) when this method is employed. The figure illustrates that although the barometer measurements (which are updating the altitude estimate) drift, the altitude estimate itself does not. Furthermore, the estimated signal also contains less random noise than any one of the sensors alone. Of course, this behavior can be tuned through the Kalman filter parameters - but the essence of the figure is to illustrate that the biases in the measurement system may be removed if states are constructed to capture those biases.

Figure 6.5 shows a plot of the true bias in the barometer as well as the estimate of the bias. The performance of the estimator in predicting the bias is remarkable considering that it only presumes the bias is a first order model, but is otherwise clueless regarding the magnitude of the time dependence. It is noted that the state vector has been appended with two additional states that represent the bias in the barometer(the actual state itself and

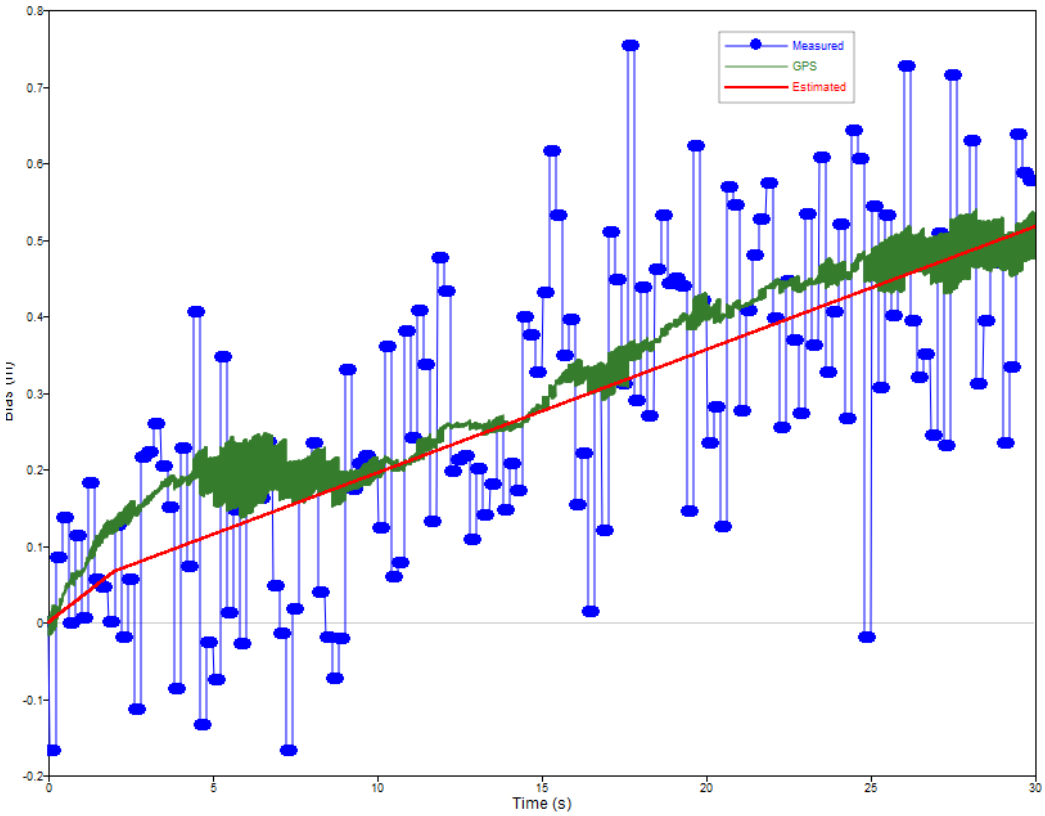


Figure 6.5: Altitude Bias signals for the UAV when given a step reference input (this estimator structure includes an error state)

its first order time derivative). Of course, a barometer may have a more complicated bias profile in practice. A first order model would be adequate for capturing the bias even in such a scenario due to the high update rates and an assumption that the drift exhibits a low time dependence.

6.2 A Theoretical Conflicts

The solutions which are presented in Figure 6.7 seem unremarkable at first glance. No artificial noise has been introduced into the system, and yet there appears to be noise in the Euler angles which are calculated using the algorithm which was presented in Chapter 4. Furthermore, the roll angle which is obtained from the algorithm which was provided seems to give errors of up to 400% (even if just momentarily). Yet the results are indeed remarkable, because these angular rate values were calculated algebraically - using only accelerometer data. The useful fact about the observed behavior in the plots

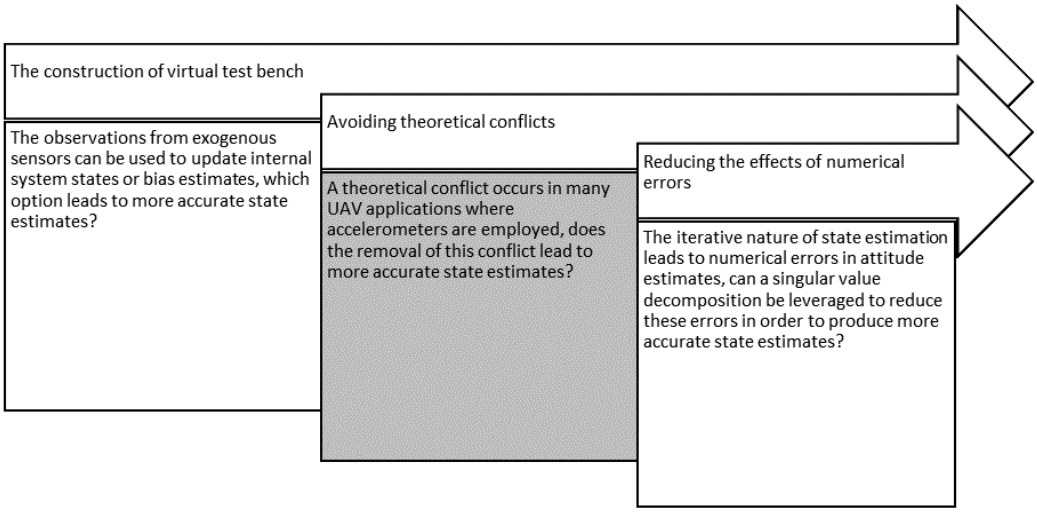
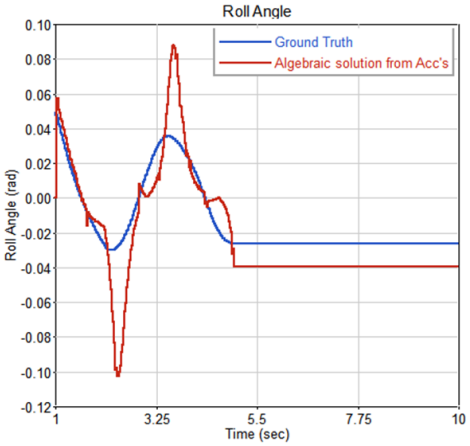
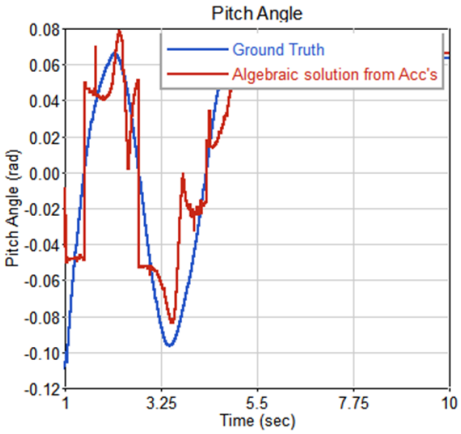


Figure 6.6: Section 6.2 utilizes the theory presented Chapter 4 to address errors related to theoretical conflicts

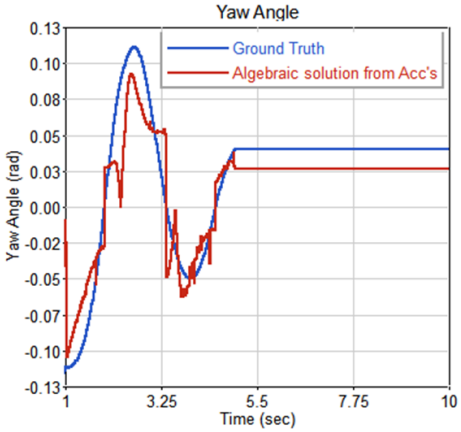
presented in Figure 6.7 is that no bias is present. This is because bias was not simulated, as it does not affect accelerometers adversely like it does with gyroscopes. In affect then, this measurement technique for the angular rate may be used as a mechanism to estimate bias on gyroscopes in a measurement integration system. Of course, the measurement technique may be used on its own too, since the errors shown are low even in their worst case scenario (the roll error is 4 degrees at its peak). It is noted that gyroscopes themselves measure acceleration, Coriolis acceleration specifically. The principle of obtaining angular rate from linear acceleration is therefore not new - although the technique presented in Chapter 4 exploits centripetal acceleration for the calculation of angular rate. The author has discussed this technique in more detail (Minnaar and Smit, 2017), where an inverted pendulum acted as the vessel on which to test the technique. Useful results from that study suggested that the separation distance between the accelerometers should be maximized in order to reduce the sensitivity of the system towards random noise.



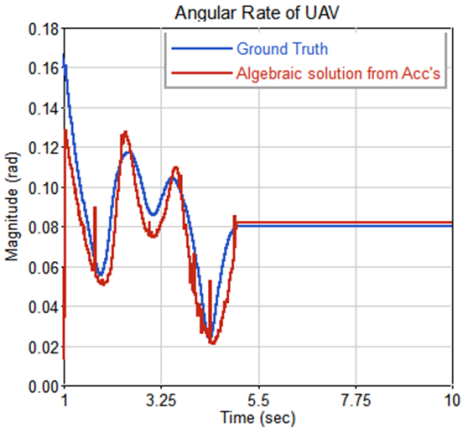
(a) An error sweep for the roll angle of a body undergoing precession motion



(b) An error sweep for the yaw angle of a body undergoing precession motion



(c) An error sweep for the pitch angle of a body undergoing precession motion



(d) An error sweep for the pitch angle of a body undergoing precession motion

Figure 6.7: The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion

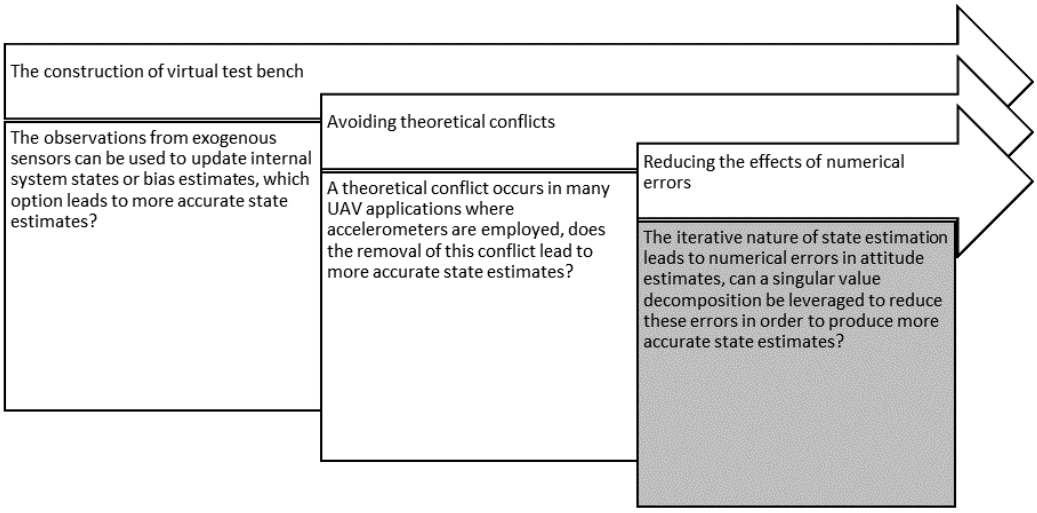


Figure 6.8: Section 6.3 utilizes the theory presented Chapter 5 to address errors related to numerical error

6.3 Numerical Errors

Numerical errors degrade the quality of attitude estimates. Chapter 5, however, provided the theory for conditioning the direction cosine matrix (DCM) in order to reduce the effects of numerical errors (Figure 6.8 contains a bookmark). Specifically, it was shown how the singular value decomposition of the DCM could be leveraged. The virtual experiments which were used to test the theory, as well as the corresponding results, are presented below.

Unlike virtual experiments done in section 6.1 and 6.2, it was not possible to use the virtual test bench (developed in Chapter 3) for experimentation in this section. The reason for this is that the author recognized that the integration process used in simulation to generate the "ground truth" data also leads to an erroneous DCM. The mechanism which drives the errors in the simulated DCM is the same as it would be in a physical UAV's autopilot - numerical errors in the propagation from one epoch to another. Although the errors are lower than what they would be in a commercial autopilot (due to the fact that a 64bit technology is used to generate ground truth data), they are present nonetheless. Fortunately, there is an analytical solution for a free-rotational motion called precession. The details of this motion are unimportant, it is only necessary to know that an analytical solution exists

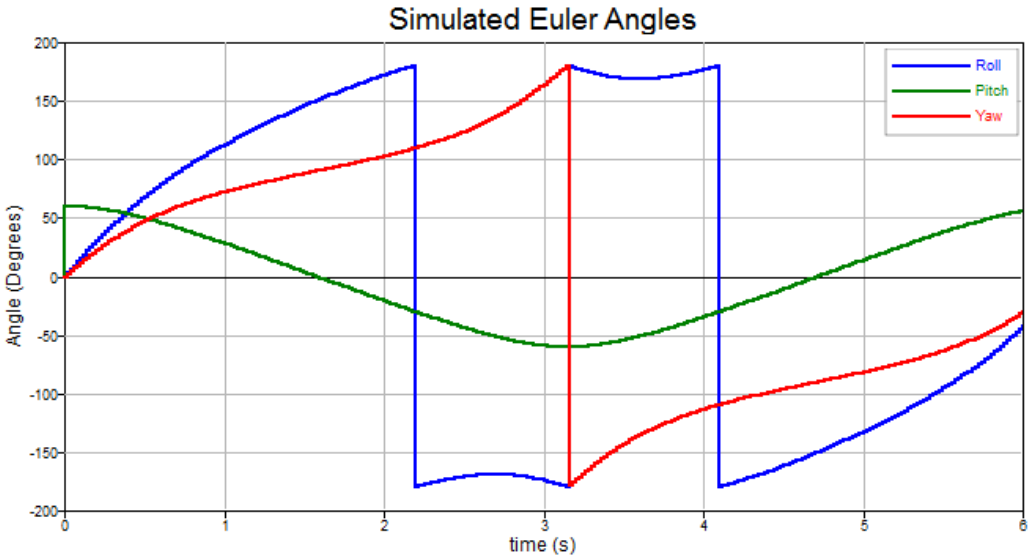


Figure 6.9: The ground truth data of an analytical solution for precession motion

for it. The corresponding solutions are provided below (Janota *et al.*, 2015).

$$\begin{aligned}
\omega_1 &= A \\
\omega_2 &= A\sin(At) \\
\omega_3 &= A\cos(At) \\
\phi &= At + \arctan 2(\sin(\psi_0)\sin(At), \cos(\psi_0)) \\
\psi &= \arcsin(\sin(\psi_0)\cos(At)) \\
\theta &= \arctan 2(\sin(At), \cos(\psi_0)\cos(At))
\end{aligned} \tag{6.3.1}$$

Recall that it is the numerical error in the propagation of the DCM itself that is a concern in this section - not the accuracy of the angular rate data (i.e. noise in the measurement data is not the symptom being treated in this section). The ground truth data of the angular rate is thus sampled and used to propagate a DCM without first adding artificial noise to the samples. The ground truth data for the motion which is simulated is presented in Figure 6.9. Notice that the angles are represented in the range from $-\pi$ to $-\pi$.

Figure 6.10 presents the errors for the three Euler angles. These errors represent the maximum difference between the true Euler angles and the Euler angles which are obtained from a propagated DCM. Furthermore, the reader will notice that the dependent variable in these plots is frequency as opposed to time (the reason for this is that the DCM propagation simula-

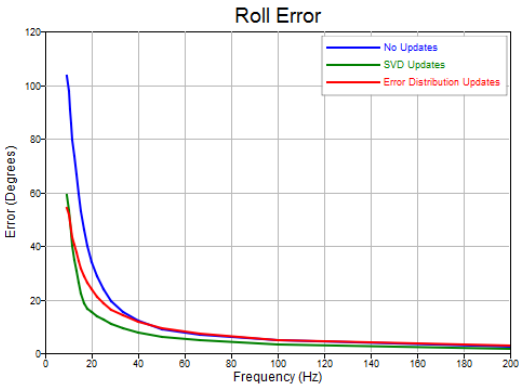
tions were performed for various sampling frequencies). This means that each subfigure contains data which was generated from multiple simulation (one simulation per frequency). Consequently, a maximum error refers to the maximum error over an entire simulation. Moreover, notice that two additional plots appear on each error figure. The first represents the Euler angle errors if no orthogonalization occurs and the second is an alternate orthogonalization technique (called the error distribution method). These additional plots are provided for comparison.

It is clear from Figure 6.10 that the SVD updates improve the accuracy of the DCM propagation, especially at low frequencies. It is also clear that the benefit of the SVD update method diminishes at higher frequencies. It is thus suggested that there is a balance of accuracy and computational effort to be found for a given hardware architecture. Moreover, the pitch angle is not accurately obtained through the DCM propagation at higher or lower frequencies, and it is suggested that it is mandatory to perform at least one of the DCM propagation correction methods in order to obtain the roll angle with some level of precision. The alternative is to propagate the DCM at a very high frequency, and this is limited by the available computational resources.

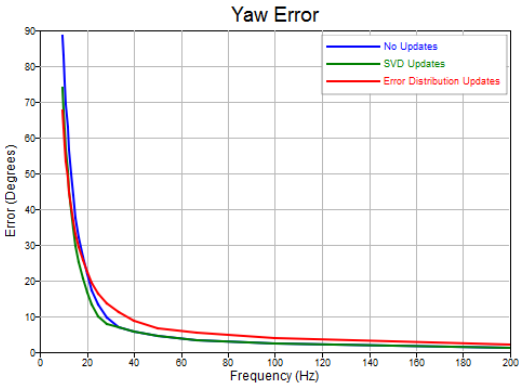
It was previously alluded that the error plots in Figure 6.10 were created from the information gathered over many simulations (each frequency in the plots represents an entire simulation). Although the benefit of orthogonalization has already been established, the single simulations which were used to generate the metadata reveal additional benefits. Figure 6.11 provides representative error data of such a single simulation at 25Hz update intervals.

It is evident that the Euler angle estimates from a DCM deteriorate as the simulation continues when no orthogonalization takes places. This makes sense, as the numerical errors are compounded at each propagation of the DCM. However, it is clear that when orthogonalization occurs, that the errors in the estimates seem to be bounded. This means that if simulations of 60 seconds were used to produce the metadata for Figure 6.10, the entire plot which represents the DCM propagation with no orthogonalization would have been shifted upwards on the vertical axis, while the other two plots would have remained in tact. showed that the SVD orthogonalization technique improves the DCM accuracy at a single epoch but it is evident from Figure 6.11 that the greatest advantage of DCM orthogonalization is that it removes the compounding effect of numerical errors.

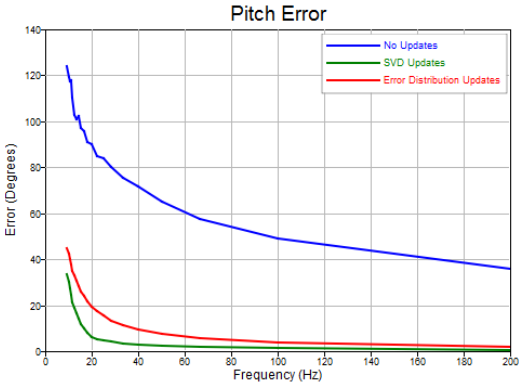
Figure 6.11 also illustrates that the numerical errors have a periodic behavior over the simulation. Recall that the motion which is simulated is periodic too, and so this suggests that the numerical errors have a depen-



(a) An error sweep for the roll angle of a body undergoing precession motion

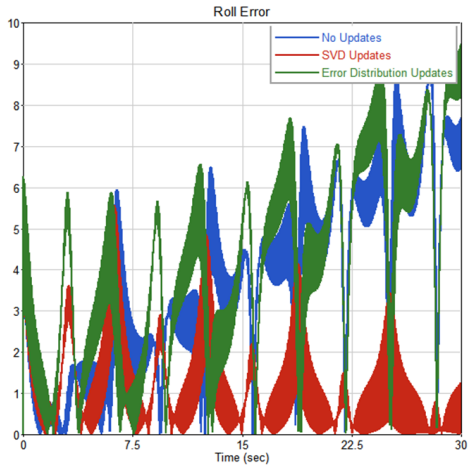


(b) An error sweep for the yaw angle of a body undergoing precession motion

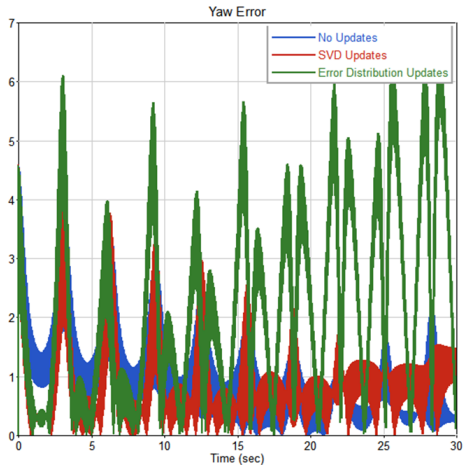


(c) An error sweep for the pitch angle of a body undergoing precession motion

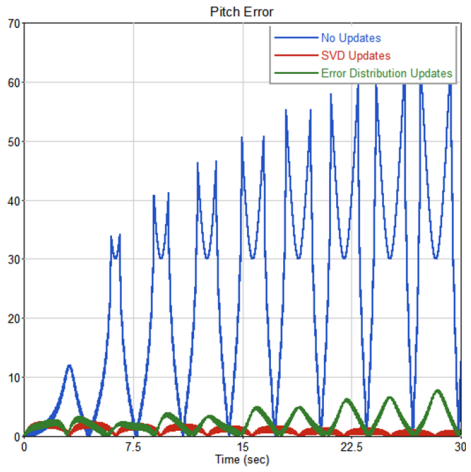
Figure 6.10: The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion



(a) The error profile for the roll angle of a body undergoing precession motion



(b) The error profile for the yaw angle of a body undergoing precession motion



(c) The error profile for the pitch angle of a body undergoing precession motion

Figure 6.11: The Euler angle errors when DCM propagation is performed in order to track the orientation of a body undergoing precession motion (25Hz propagation updates)

dency on the orientation of the vehicle. This conclusion was also drawn by Lock (2016), who attempted to estimate the states of a UAV using computer vision techniques. The results presented in Figure 6.11 thus enrich that particular conclusion which was drawn by Lock (2016), because they illustrate that it is not only computer vision techniques that suffer from orientation dependent errors in the state estimates - direct body rate measurement systems which utilize a DCM would suffer from these errors too.

Chapter 7

Conclusions

The act of measurement can never be done precisely. Consequently, experiments which are done in the real world can only ever be done with the same uncertainty as the measurement devices which are employed. It is therefore difficult to categorically attribute the behavior of observed measurements with a particular navigation strategy that has been employed. This thesis sought to mitigate the afore mentioned eventuality by replacing physical experiments with virtual ones. A virtual test bench was built in which ground truth data was generated for the subsequent scrutiny of suggested navigation strategies. Using this paradigm, it is impossible to attribute observations in simulation data with anything other than the physics or mathematics which caused it. The aim of this thesis was to find strategies for increasing the navigational accuracy of a UAV. In the end, three causes for navigation errors were isolated and a strategy was proposed to mitigate each of these causes. The first strategy attempts to deal with the utilization of exogenous signals from sensors that operate on position fixing principles. The second strategy details a theoretical error that is common in UAV applications when accelerometers are utilized, and suggests a resolution strategy which ultimately provides a way of correcting gyroscope bias. The last strategy which was proposed deals with the destruction of orthogonality in the direction cosine matrix (a parameterization which assists in describing the attitude of the UAV), and specifically how this can be prevented. Each of these strategies require conclusions of their own, and so this chapter has been sectioned accordingly. Some general conclusions on this work are also packaged in section 7.1, and are provided as a preface. Additionally, the thesis is enriched with section 7.5 which serves as both a warning and acknowledgment of the shortfalls of the project. Finally, the section conclusions pertaining to the three strategies which were suggested are complimented with recommendations for future work that are provided in section 7.6.

7.1 General Conclusions

The reader may be concerned that the lack of physical experiments infer that the strategies which were generated in this thesis are not validated. Two arguments are provided to support the view that this is not the case. Firstly, in the case of the first two strategies which were proposed (a strategy for utilising exogenous sensors and a strategy for resolving a common theoretical conflict) the validity of the suggestion did not rely on the validity of the ground truth data which was generated by the virtual test bench. Secondly, in the case of the final strategy (an orthonormalization technique), the proposals were evaluated using analytical models for precession motion. The following comments describe how the project objectives were explicitly met. Firstly, it was necessary to assess available literature in order to identify avenues worth investigating in the hunt for high fidelity knowledge concerning the motion of a UAV. This objective has been met in Chapter 2 with a comprehensive literature review that has been sectioned according to the inaccuracies that are addressed later throughout the thesis (the chapter was also enriched with general navigation information to provide context for the thesis). It was also necessary to identify if solutions already exists within the literature, and this objective was also achieved within Chapter 2. Thirdly, it was necessary to propose solutions for addressing the inaccuracies which were identified but not adequately addressed by the literature. Chapters 2, 4, and 5 all contribute to the fulfillment of this objective. Although the chapters mostly contain elements of the necessary theory which already exists in the literature, the chapters aim to package the theory in a way that is specific to UAV applications, and quadcopters in particular. Next, it was necessary to construct a virtual test bench which could serve as a platform to scrutinize the strategies generated in the previous objective. This objective was achieved in Chapter 3. Lastly, it was necessary to assess the effectiveness of the proposed strategies in increasing the navigation accuracy of a UAV. This objective was solely fulfilled in Chapter 6. The following section will provide more detailed conclusions in this regard. The fulfillment of all five of these objectives lead to the conclusion that the project has achieved its stated aim which was to find suitable strategies for increasing the navigational accuracy of a UAV.

7.2 The Utilization of Exogenous Sensor Signals

The use of position-fixing sensors in UAV navigation systems is desirable since they do not generally suffer from growing biases. There is not one unique way to leverage the measurements from these sensors, however, and the two alternatives that were examined in this thesis are 1) that the signals be used as observers of internal UAV system states and 2) that the signals be used to estimate the bias on one of the other measurements in the system. The representative use case that was illustrated was a virtual UAV that needed to estimate its altitude under a step reference input. Three virtual sensors were place which included accelerometers, a barometer and a GPS. It was found that setting the GPS measurements as observers for the barometer bias yielded better estimates than setting the GPS measurements as observers for the altitude itself. This is due to the fact that the bias which was introduced to the barometer measurements were successfully mitigated by approach two while the influence of the bias in the barometers were still pronounced in approach one. It is noted that the measurements were not synchronized, nor were they sampled at the same frequency. Since barometer measurements arrived more frequently than GPS measurements, it is expected that the barometer measurements would dominate the GPS measurements as an altitude observer.

7.3 The Resolution of a Theoretical Conflict

A theoretical conflict is raised when accelerometers are used to estimate the attitude of a UAV and its linear states simultaneously. This is true because in order to estimate the vehicle attitude from accelerometer readings, it has traditionally been assumed that the accelerometer triad is static. On the other hand, in order to integrate accelerometer signals in order to obtain velocity or position, it must be fundamentally assumed that the accelerometer is accelerating, hence the conflict. Despite this misnomer, it has been demonstrated that the pragmatic approach produces acceptable results - the pragmatic approach being to ignore the conflict altogether. Chapter 4 built a case for the resolution of this theoretical conflict. The chapter showed how accelerometers could be used to measure angular rate even when the assumption of static motion is removed. Furthermore, Chapter 6 proved that the approach was capable of measuring transient attitude information during simulated UAV motion. Of particular importance was that the body rates measured would not be subject to a growing bias, as is the case with gyro-

scopes. This means that the accelerometer based attitude rate measurement strategy could replace gyroscopes. Alternately, the strategy could be used to augment gyroscope based body rate measurement systems by updating bias estimates of the gyroscope sensors. It is worth noting that the author has had a paper accepted for a proceeding as a result of the work done in this section (Minnaar and Smit, 2017).

7.4 The Preservation of Orthogonality

The loss of orthogonality in the direction cosine matrix (DCM) leads to a loss in accuracy on the euler angle estimates which are translated from the DCM. This was shown in Chapter 5 alongside a description for the source of the loss in orthogonality (numerical errors in the propagation of the DCM). No unique solution exists for addressing this eventuality, and so a method was proposed which leverages the singular value decomposition (SVD). While this method of re-orthogonalization is not new in general, no literature exists which examines the orthogonalization technique in a simulation environment. Rather, existing literature describes the benefit of the the technique in terms of a single orthogonalization iteration. Meanwhile, Chapter 6 has shown that the strategy of orthonormalization has a compounding effect, and that it bounds the error that can be expected on attitude estimates (whereas the error is unbounded for DCMs that are continually propagated but not orthogonalized). Chapter 5 continued to argue that a popular commercial autopilot still relies on an alternate solution for orthogonalization. That technique was evaluated against the one suggested here, and it was shown that the SVD orthogonalization provides superior accuracy over the older strategy - although the benefit diminishes at higher sampling frequencies. This also suggests that at lower frequencies, the numerical effects are not as great as the violation in the assumption of small angles in propagating the DCM - some deformation is required to match observations with ground truth data. It was speculated that the commercial autopilot leveraged the alternate orthogonalization technique due to its computational requirements, which are less. The strategy proposed here may or may not be applicable in a practical application and it is left as a suggestion that future work investigate this.

7.5 Difficulties and Shortfalls

Three project shortfalls/ difficulties are identified. Firstly, since the effects of the controller were not of concern in this thesis, some basic decoupled proportional integral derivative (PID) controllers was used to command the UAV state vector. The structure of the controller was alluded, but a schematic representation can be found in Appendix G. The parameters of this controller were not investigated extensively, and a reader interested in performing these simulation may want to direct more time into the design or modification of the control parameters. Secondly, it was discovered that the co-simulation interface that exists between the dynamics solver and the schematic based block diagram solver does not handle continuous simulation data as expected. For this reason, all simulation equations were discretized (inclusive of the controller). Lastly, the Kalman filter algorithm which has been used to integrate the measurements was adequately derived in Chapter 3. The results, however, did not focus on the effects on the tune-able parameters of the filter. It is left as a suggestion that the interested reader pursue an investigation into these effects.

7.6 Recommendations for Future Work

The project has been driven from a theoretical point of view, and validated using simulation technology. One natural recommendation for future work is then to extrapolate the work which was done into physical experiments. The argument for doing this is not so that the strategies which have been suggested may be validated, since that was done with simulation. Rather, the author suggests that there will be practical hurdles that would need to be overcome in order to actually implement these strategies in a physical autopilot. For example, the singular value decomposition has been a workhorse of the suggested strategies. The computational requirements of this mathematical decomposition may supersede the need for more intensive resources than those currently available in commercial navigation systems. This hurdle would need to be addressed in a physical implementation (it is possible that the resources available in current navigation systems already meet the computation requirements of these strategies, and if not, may do so in the future due to the nature of continued product development).

Appendices

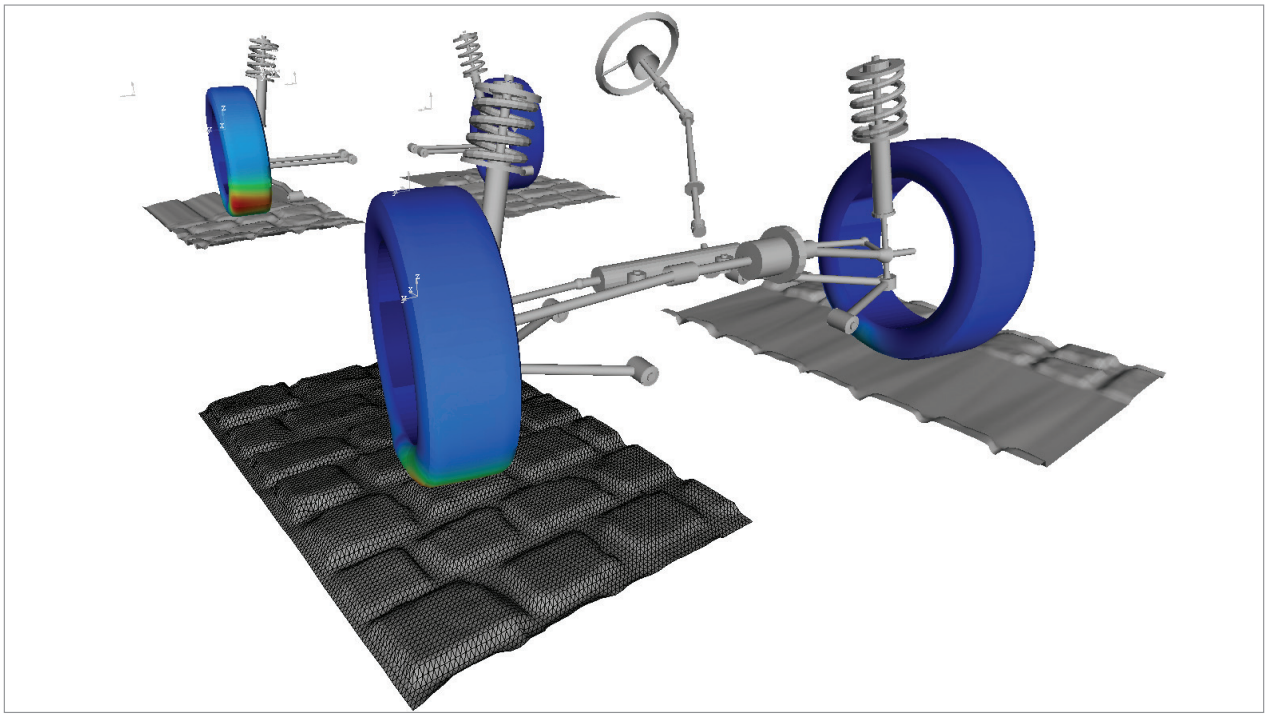
Appendix A

MotionSolve Brochure

MotionSolve®

Multi-body Dynamics Simulation

Altair | HyperWorks®



MotionSolve is an integrated solution to analyze and optimize multi-body systems. MotionSolve offers powerful modeling, analysis, visualization, and optimization capabilities for simulating complex systems. You can perform kinematic, dynamic, static, quasi-static, linear, and vibration analyses. MotionSolve helps you to understand and improve the performance of your product.

Product Highlights

- Comprehensive multi-body solution to optimize mechanical system performance
- Easily model, analyze, evaluate, and optimize your mechanical system
- Validated across several automotive, aerospace, and general machinery applications
- Extensively correlated to test data through partnership with customers

Learn more:
altairhyperworks.com/motionsolve

Benefits

Reduce Product Development Time
Build simple models early in the design phase and add complexity as the design evolves. MotionSolve supports a large set of modeling elements and a variety of analysis methods to facilitate this. Through simulation you avoid time consuming physical testing and get to the right answer earlier.

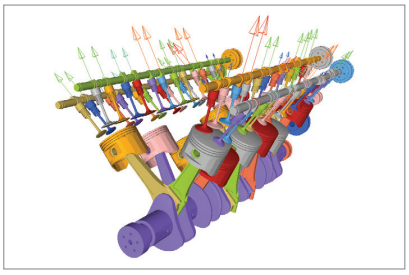
Improve Product Quality
Build multi-body models that have the fidelity to capture phenomena of interest to you and accurately solve the underlying equations to characterize product behavior. Examine the product behavior to determine if the design meets your need.

Accelerate Product Innovation
Evaluate the behavior of complex systems in realistic settings. Perform design of experiments (DOE) and stochastic simulation to characterize and optimize product performance. Use the loads computed by MotionSolve for component weight, strength and other performance optimization.

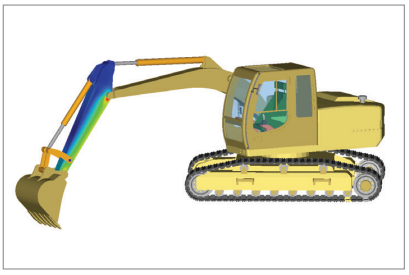
Reduce Design and Manufacturing Risk
Through simulation evaluate a wide variety of alternative concepts and designs very quickly and choose the best design. Moreover, as the design evolves easily validate updated designs with models that have already been built.

Modeling Capabilities
MotionSolve supports a rich set of modeling elements that allows you to build multi-body systems with the desired degree of complexity. MotionSolve is an open platform that offers built-in integration with CAD, FE, Controls, 1D simulation, CFD, and Optimization.
MotionSolve models routinely include:

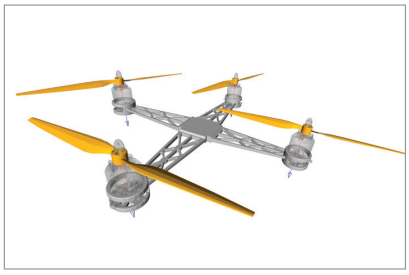
- 2D and 3D rigid bodies
- Linear and nonlinear flexible bodies



2D contact between cam and follower



Excavator flexbody simulation



Unmanned Aerial Vehicle dynamics analysis

- Lower- and higher-pair constraints
- Linear and nonlinear force connectors
- General 2D and 3D contact based on CAD geometry
- Contact between deformable curves / surfaces
- Joint friction, limits and slop
- Motion inputs
- Transfer functions and state matrices
- Splines for inputting test data
- Event sensors
- Generic nonlinear algebraic and differential equations
- User-defined elements to model non-standard phenomena

Analysis Capabilities
With MotionSolve, you can determine whether a system meets its desired requirements. You can study the dynamic behavior of a system, compute its vibration characteristics, assess the performance of control systems in the system, perform packaging studies and generate realistic loads to predict component life and damage. If these built-in analyses are not adequate, you can create and use your own analyses scripts.

MotionSolve provides many options for analyzing system behavior:
- Implicit/explicit, stiff/non-stiff and DAE/ODE based methods of numerical integration
- Static/quasi-static solvers to compute static equilibrium configurations and loads
- Automatic detection and removal of redundant constraints
- Kinematic analysis for motion driven systems
- Linear analysis with ABCD export, eigenvalue computation and modal energy distribution tables
- Co-simulation to solve multi-physics problems
- Custom analyses specified in user-subroutines

Vehicle Dynamics, Durability & NVH Solutions
MotionSolve provides a comprehensive solution for the automotive market. For a complete description, please see the automotive brochure. MotionSolve contains a library of parametric automotive components that help you build subsystems. With support for TNO Delft-Tyre, FTire, CD Tire and the OpenCRG standard, MotionSolve provides tires and roads of varying fidelity for your applications. Templates for common subsystems such as suspensions, steering and leaf springs are available. Wizards guide you step-by-step to quickly assemble a car or truck. Simple user interfaces permit you to define and run static, dynamic or steady state events. Automated reports allow you to quickly assess system performance. You can extend all of the above to facilitate workflows specific to your organization. For instance, you can have Excel drive the Model-Analyze-Evaluate-Improve workflow. With these core capabilities you can perform suspension design and analysis, evaluate vehicle dynamics, assess controller behavior, conduct rough road simulations for estimating component durability and study the NVH characteristics of your vehicle.

General Machinery & Mechanism Solutions
Mechanisms are used in all industries. MotionSolve provides comprehensive contact capabilities that enable you to quickly build and accurately analyze complex systems that may contain thousands of contacts. As with automotive solutions, you can also create a library of parametric components, templates for systems and use wizards to build models and run simulations. For more information about general machinery and mechanism solutions, please see the general machinery brochure.

1D, Controls and Mechatronics Solutions
MotionSolve provides state-of-the-art integration to 1D and controls software so you can reuse validated MotionSolve models in this context also.
- Early in the design phase, you can import linearized multibody models from MotionSolve as state matrices (ABCD) into your controls package and perform the control design.
- Later, in the evaluation phase, you can import high fidelity MotionSolve models into Matlab, Simulink or solidThinking/Activate® to evaluate the controller. In the 1D or controls environment you connect the systems so they can exchange signals at run time. Then you run a simulation of the entire system to evaluate how well the system is performing. Alternatively, MotionSolve can import Simulink Coder C code as user subroutines and perform the same simulation.
- MotionSolve supports the FMI/FMU 2.0 protocol so you can include a wide variety of models that have been developed elsewhere.

HyperWorks Integration
With MotionSolve, HyperWorks delivers a complete multibody simulation environment. You can:
- Easily build multi-body models in MotionView® as well as in HyperMesh®
- Review system results as animations and plots with HyperView® and HyperGraph®
- Run custom scripts for complex post-processing with solidThinking Compose®
- Improve system fidelity by generating reduced flex-bodies with the OptiStruct®
- Perform component optimization in OptiStruct® with loads computed by MotionSolve
- Couple with AcuSolve® to analyze multi-body systems where fluid effects are important
- Connect to solidThinking Activate® to design and validate mechatronics systems
- Use HyperStudy® to perform system level DOE, optimization and stochastic studies

Altair **Altair Engineering, Inc., World Headquarters** 1820 E. Big Beaver Rd., Troy, MI 48083-2031 USA • (P) +1.248.614.2400 • (F) +1.248.614.2411 • www.altair.com • info@altair.com

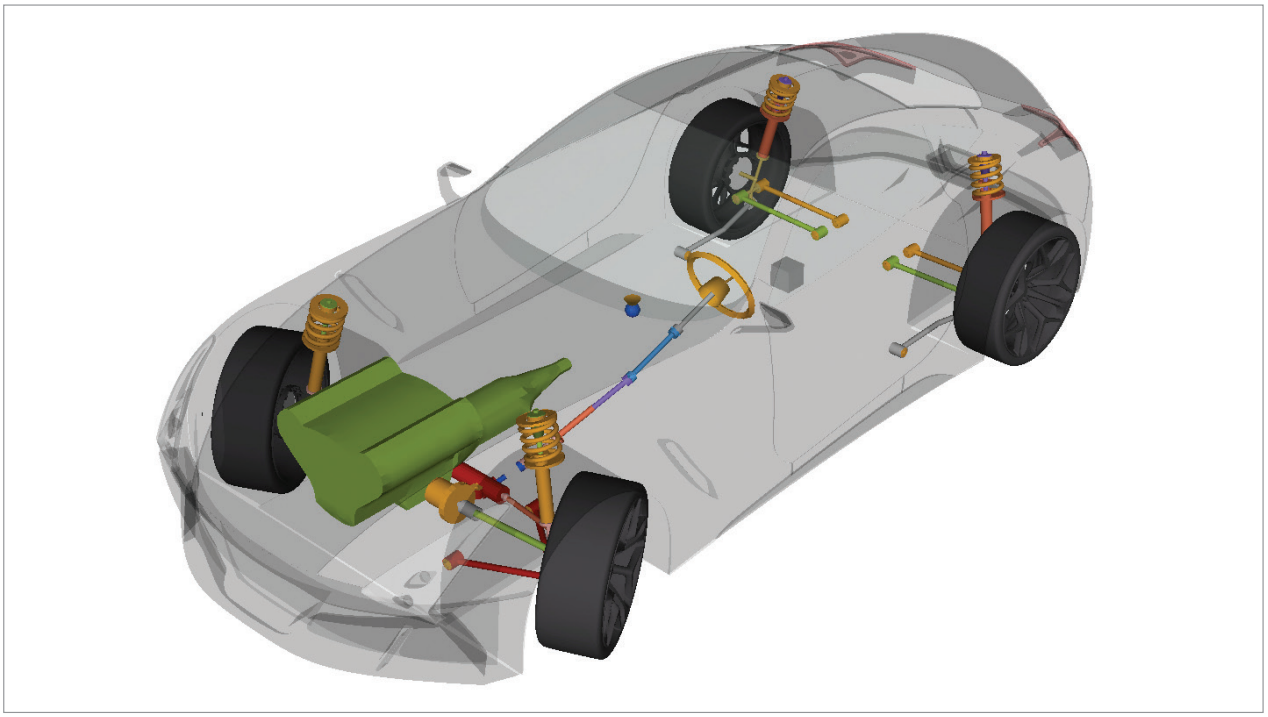
02.2017

Appendix B

MotionView Brochure

MotionView®

Multi-body Systems Modeling



Altair MotionView is a user-friendly and intuitive multi-body systems modeling environment. Its built-in parametric modeling capability and hierarchical modeling language allows users to quickly build, analyze, and improve mechanical system designs even before physical prototypes are available. In conjunction with MotionSolve, MotionView provides the perfect solution for your multi-body dynamics simulation needs.

Product Highlights

- Intuitive, solver neutral environment for multi-body systems modeling
- Hierarchical modeling
- Built-in parametric modeling for efficient studies of model variations
- User extensible GUI and data model to support product customization
- Automated assembly for complex systems

Learn more:
altairhyperworks.com/motionview

Benefits

Accelerate Product Innovation
With MotionView you can build parametric models, easily assess alternative designs and choose the best design to meet your objectives.

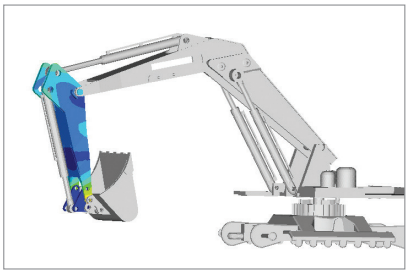
Reduce Product Design Time and Cost
You can evaluate products early in the development cycle with MotionView's physics-based simulation capabilities. Furthermore, you can build a model once, validate it and reuse it in many different contexts.

Improve Product Quality
MotionView allows you to easily conduct what-if analyses and stochastic simulations to characterize product behavior. You can use this information to mitigate the effects of manufacturing variations on product performance.

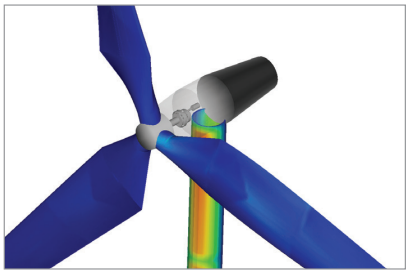
Enforce Corporate Quality Standards
MotionView can capture your company's know-how as repeatable processes to ensure usage consistency. You can customize the user interface to meet your needs, use automation capabilities to eliminate repetitive procedural tasks and standardize and share models, data and results with others in your organization.

The Modeling Environment
MotionView contains many capabilities designed to simplify the creation of complex mechanical models.

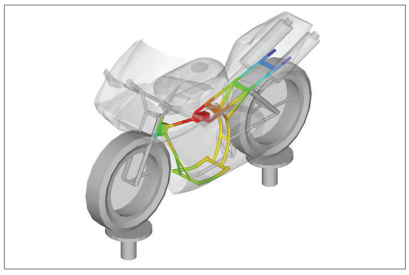
- Easy import of CAD geometry to create the system model; all popular formats are supported
- Import neutral geometry formats such as Parasolid, STEP or IGES
- A hierarchical modeling language to easily build complex models
- Parametrics to facilitate downstream DOE, optimization and design studies



Excavator arm flexbody simulation



Wind turbine flex-body simulation



Motorcycle ride analysis

- Built-in support for symmetry planes to minimize input
- Conditional logic to enable multiple topology configurations within a single system model
- Automated system assembly via a wizard that reduces model assembly to just a few mouse clicks
- Easy import of test data into the multi-body system model
- Comprehensive modeling support for MotionSolve and Adams

Intuitive User Interface

MotionView's intuitive user interface allows both experienced and novice engineers to build and analyze multi-body systems rapidly.

- Built-in workflows simplify and standardize mechanical systems modeling
- A modern user interface with context menus in the graphics window allows for intuitive software usage
- A project browser with context sensitive menus, search and filtering options ensures easy model navigation
- A wide variety of graphically accessible tools to create, modify and manipulate models easily

Automation and Customization

MotionView is completely customizable. You can modify MotionView to meet your needs.

- Build custom objects with MotionView's unique Model Definition Language
- Create custom panels and menus to graphically create custom objects
- Send models to solvers and retrieve results
- Use scripting to automate repetitive modeling tasks and minimize mouse actions
- Export component loads in FE & fatigue formats for downstream component design, strength, fatigue, and optimization calculations
- Generate reports to communicate system performance with others in your team

Easy Flex-body Generation and Usage

MotionView provides a simple yet powerful set of tools to create flexible bodies in your model.

- Easily import reduced finite element models to represent flexible bodies or build nonlinearly flexible systems
- Perform error checking to identify and diagnose modeling errors
- Connect flexible bodies to a multi-body system model
- Convert a rigid component to flexible and vice-versa
- Mirror flexible bodies about a plane of symmetry to simplify modeling

A Comprehensive End-to-end Solution

MotionView supports the MODEL—ANALYZE—REVIEW—OPTIMIZE paradigm of use for multi-body systems. In a single environment you can perform all of your tasks – no need to switch between products.

- MODEL: Create or assemble complex multi-body models graphically
- ANALYZE: Send a validated model to a multi-body solver to run a simulation. MotionSolve and ADAMS are natively supported
- REVIEW: Analyze and correlate simulation results to test data, compute performance metrics, plot results, view animations, create and publish reports summarizing model behavior
- OPTIMIZE: Execute DOE, optimization, and stochastic studies through HyperStudy® to understand system behavior; optimize components with OptiStruct®

Automotive Solutions

MotionView provides a broad set of tools for car and truck modeling. A comprehensive library of higher-level, automotive-specific modeling entities such as tires, roads, drivers, springs, bushings, bump-stops, dampers are available for building vehicle models. You may add your own components to the built-in set. In addition, MotionView supports model and task assembly wizards. With just a few mouse clicks you can assemble a fully parametric vehicle with your selection of front and rear suspension topologies, a full IC engine powertrain, choice of tires, smooth and rough roads and simulate any of the standard suspension and driving events. Component loads can be sent for downstream strength or durability analysis. Simulation reports are automatically generated. For more information, please refer to the automotive brochure.

General Machinery Solutions

With MotionView you can import CAD and FE geometry to quickly build your system. All popular formats are supported. Component mass and inertia properties are automatically computed. 2D and 3D contact is easily specified between complex geometry shapes. 2D shapes can be extracted from the CAD geometry. You can also import data from CSV files to create "hard points" in any coordinate system of your choice. You can use specialized tools for modeling generalized joints that include compliance, friction, limits and slop; quickly construct belt-pulley systems, gear systems, cables, pulleys and winches. These core capabilities may be used to quickly assemble a system and perform your analysis of choice. For more information, please refer to the general machinery brochure.

Altair **Altair Engineering, Inc., World Headquarters** 1820 E. Big Beaver Rd., Troy, MI 48083-2031 USA • (P) +1.248.614.2400 • (F) +1.248.614.2411 • www.altair.com • info@altair.com

02.2017

Appendix C

Activate Brochure

ACTIVATE™
solidThinking

solidThinking Activate enables product creators, system simulation and control engineers to model, simulate and optimize multi-disciplinary systems. By leveraging model based development Activate's users can ensure that all design requirements are successfully met while also identifying system level problems early in the design process. Activate's intuitive block diagram environment empowers users to rapidly build demonstrations of how real world systems function and easily experiment with new ideas without any need to build prototypes

Product Highlights

- Model based development of hybrid systems
- Construct hierarchical, parameterized multi-disciplinary models
- Mix signal-based and physical (Modelica) components in the same diagram
- Easily extensible, built-in block libraries including library management
- Model exchange or co-simulation through the Functional Mock-Up interface
- Co-simulation with multi-body dynamics
- Compile models into executable code

Learn more:
solidThinking.com/Activate

Benefits

Improve system level performance
Simulate and improve the dynamic behavior of any multi-disciplinary system using Activate. Activate makes it easy to model, simulate and validate smart systems where users can incorporate functions of sensing, actuation, and control coming from diverse components.

Design for robustness
Model based development using Activate provides an efficient approach for establishing a common framework for communication throughout the design process. Perform what-if analyses at the system level to quickly test several designs and investigate the interactions of all components in a system.

Gain functional insight early
Activate empowers users to identify system level problems early in the design process while ensuring that all the design requirements are met. Activate provides its users with a standard set of predefined blocks that can easily be combined to model systems.

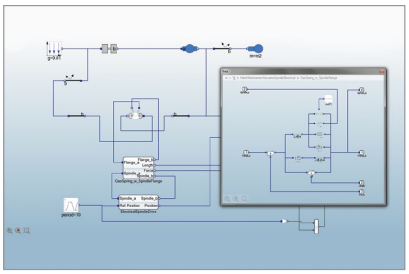
Activate users can easily leverage the large library of Modelica physical components to further describe the plant and the controller.

Capabilities

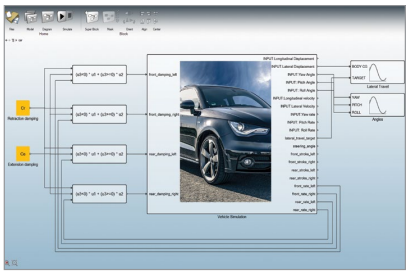
Build diagrams intuitively
- Drag, drop and connect paradigm to rapidly construct models
- Multiple window configuration with the ability to modify diagrams between windows using the drag-and-drop and copy-and-paste operations
- Support for concurrent loading of multiple models in a session

Hybrid modeling
Model and simulate continuous and discrete dynamic systems.

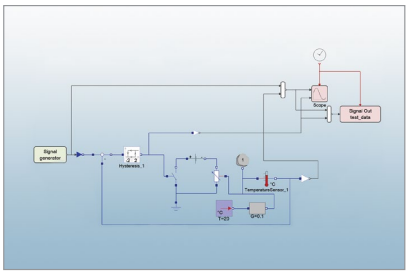
Multi-disciplinary modeling
Real-world systems are multi-domain in nature. Activate allows users to model and simulate the combined system behavior of real world systems with support for multiple domains such as Mechanical, Electrical, and more.



Physical component Modeling of a hatch Mechanism – (Mechanical/Electrical modeling)



State-of-the-art co-simulation with Multi-body Dynamics



Room temperature control system with Modelica components

Hierarchical & parametric modeling

- Build hierarchical component-based models of the real world system using signal based and physical modeling libraries
- Mix signal based and physical modeling blocks in the same model
- When modeling large or complex systems, easily create super blocks by encapsulating multiple blocks in a diagram into a single block. Super blocks are modular, reusable, can be masked, and fundamentally behave like regular blocks allowing users more flexibility
- Since a model can be hierarchical and parameters can be defined at different levels, Activate provides an all available parameters option which lets users navigate in a diagram and get a report of all parameters that are known or defined at a current level
- Generate C-code directly from your model

Built-in block-based model libraries

Activate includes a large variety of predefined blocks that are available in a library system of palettes. Users can also create their own custom blocks in C or math scripts and save them to new or existing libraries.

- Signal Generators
- Signal Viewers
- Signal Importers
- Signal Exporters
- Signal Conversions
- Signal Properties
- Math Operstions
- Dynamic
- Hybrid
- Routing
- Logical Operations
- Activation Operations
- Matrix Operations
- Lookup Tables
- Ports
- Buffers
- Bus Operations
- Optimization
- Cosimulation
- FlipFlops
- Custom Blocks

Physical Component Modeling Using Modelica

Easily extend the capability of Activate using Modelica. A better way to model physical components is to use implicit blocks in which the behavior of the blocks is specified through symbolic equations.

Modelica, which is a standard in component level modeling is supported natively in Activate for acausal modeling.

Library Management

Easily create components and assemble custom applications. Use Activate's library manager to create and edit custom libraries. Activate also provides an IDE along with API functions for users to further leverage library management.

Hybrid Simulator

Activate's simulator provides users with several high performance numerical solvers that accurately and robustly solve dynamic systems including continuous, discrete-time and event based behaviors.

Solver Type	Stiffness	Solver Name
Fixed step -size	Non-stiff ODE	Forward Euler Explicit Trapezoidal Classical Runge Kutta Runge-Kutta
	Stiff ODE	Backward Euler Implicit Trapezoidal
Varible step-size	Non-stiff ODE	CVODE-BDF-Functional CVODE-ADAMS Functional DOPRI (Dormand-prince)
	Stiff ODE	Lsode CVODE-BDF-NEWTON CVODE-ADAMS-NEWTON RADAU-IIA for ODE CPODE
	DAE	IDA RADAUV-IIA for DAE DASKR

Optimization

Formulate optimization problems to improve the system parameters and design robust control strategies via:

BOBYQA optimizer block

- This optimization block can be used directly in a model and doesn't require any external calling function/link up
- Cascade multiple optimization blocks to formulate max-min and min-max problems
- Graphical optimization tool - the simplest way to formulate and solve optimization problems
- Script based optimization - a powerful mechanism for solving general optimization problems where the cost and constraints may be obtained from a combination of Activate simulation results and math scripts

Model exchange and Co-simulation via Functional Mock-up Interface (FMI)

Activate supports FMI 2.0 standard for both model exchange and co-simulation of dynamic systems including the ability to import and export FMU (Functional Mock-up Unit).

Co-simulation with Multi-body Dynamics

The co-simulation interface lets users simulate a complex system that includes a multi-body system (MBS) and one or more control subsystems. In order to effectively simulate the entire system, the MBS is simulated with a Multi-body simulation solver while the control subsystem is simulated with solidThinking Activate.

Linearization

Activate allows users to create linear models from Activate blocks by linearization. The operating point can be computed either by running the simulation at a given time instant or by computing a steady-state point by imposing constraints on inputs, outputs, states and state derivatives.

Compiling models into executable code

Activate supports code generation for system performance & IP protection.

Appendix D

Co-simulation Technical Documentation

Chapter 12

Co-Simulation with Multi-body Simulation

12.1 Introduction

Modelling complex heterogeneous systems in engineering usually leads to hybrid systems of differential and algebraic system of equations with discrete-time equations. Such complex multi-disciplinary systems cannot often be modelled and simulated in one simulation tool alone. Subsystem models are often available only for a specific simulation tool. For example special tools for CFD (Computational Fluid Dynamics) models or integrated circuit systems. In many situations, the sub-systems shall be simulated with the simulator which suits best for the specific domain. Thus for the simulation of multi-disciplinary models it is often reasonable or even necessary to couple different simulation tools with each other or with real world system components. Co-simulation is a general approach for the joint simulation of models developed with different tools where each tool treats one part of a modular coupled problem. The data exchange (input and output variables, status information) between subsystems is restricted to discrete communication points. In the time between two communication points, the subsystems are solved independently from each other by their individual solvers.

Master-slave is a common method in Co-simulation. In a master-slave approach the slave simulates the sub-model whereas the master is responsible for both coordinating the overall simulation as well as transferring data. The slaves are the simulation tools, which are prepared to simulate their subtask. The slaves are able to communicate data, execute control commands and return status information.

Several MSplant blocks representing different MotionSolve models can be instantiated in an Activate model.

12.2 Co-Simulation with MBS

The Co-simulation interface lets you simulate a complex system that includes a multi-body system (MBS) and one or more control subsystems. In order to effectively simulate the entire system, the MBS is simulated with MotionSolve [1] while the control subsystem is simulated with solidThinking Activate.

[1] The installation of HyperWorks MotionSolve software is required for co-simulation. MotionSolve is a multi-body modeling, analysis, visualization and optimization solution for performing multi-disciplinary simulations that include kinematics and dynamics, statics and quasi-statics, linear and vibration studies, stress and durability, loads extraction, co-simulation, effort estimation and packaging synthesis.

In the **Activate**-MotionSolve interface, **Activate** is the master and MotionSolve (MS) is slave. The co-simulation interface has been implemented in a block called **MS Plant** which is available in the `Advanced/CoSimulation` palette, as shown in Fig.12.1. **MS Signals** is a variant of **MS Plant** block that displays the name of MS signals at the input/output ports of the blocks.

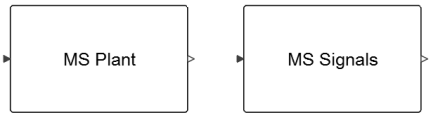


Figure 12.1: Blocks in **Activate** to simulate a MotionSolve model.

The block provides a small set of C++ functions to implement the **Activate**-MotionSolve interface. The path of the MotionSolve model is given to MS Plant and MotionSolve solver is called to simulate the model. In order to perform the simulation, it is necessary to have MotionSolve installed on the machine. This process is shown in fig.12.2.

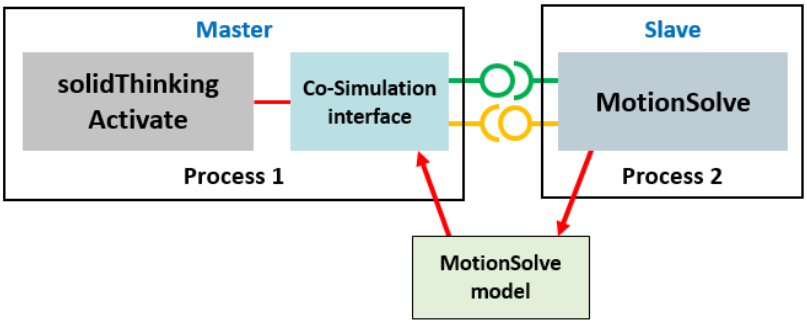


Figure 12.2: Co-simulation between **Activate** and MotionSolve.

The co-simulation connection variables to define the inputs/outputs from the MotionSolve model are defined within the Control_PlantInput and Control_PlantOutput MotionSolve statements, which is a list of runtime (solver) variables for inputs and outputs to the mechanical plant, respectively. These also have settings for the hold_order, sampling_period, and offset_time.

In co-simulation, after instantiation and initialization of the MotionSolve solver, inputs to the MotionSolve model is provided and the output of the MotionSolve block is requested. In order to explain the way a co-simulation is performed, consider a model composed of two simulators as shown in Fig.12.3.

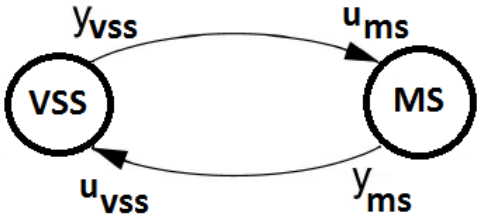


Figure 12.3: Co-simulation between **Activate** and MotionSolve

If the output of each model is computed as following equations:

$$\begin{cases} y_{vss} &= F_{vss}(y_{ms}) \\ y_{ms} &= F_{ms}(y_{vss}) \end{cases}$$

in order to co-simulate this system, this equation should be solved at each communication instant. F_i functions are usually complex, non-linear, and time-dependent relationships, as a result, iterative methods are the only choice for solving such equations. Three common methods used for solving them are Gauss-Seidel and Gauss-Jacobi and Newton methods. The Gauss-Seidel iteration method is represented as follows.

$$\begin{aligned} y_{vss}^{i+1} &= F_{vss}(y_{ms}^{i}) \\ y_{ms}^{i+1} &= F_{ms}(y_{vss}^{i+1}) \\ i &= 0, 1, 2, ... \end{aligned}$$

The Gauss-Jacobi iteration method is defined as

$$\begin{aligned} y_{vss}^{i+1} &= F_{vss}(y_{ms}^{i}) \\ y_{ms}^{i+1} &= F_{ms}(y_{vss}^{i}) \\ i &= 0, 1, 2, ... \end{aligned}$$

These iterative methods are simple fixed-point methods and their convergence(at most linear) depends on F_i functions. If the Fixed Point converges very slowly or diverge the Newton-type methods can be used. These methods are, however, more complicated, because in each step the Jacobian must be computed which results in higher number of simulator calls. Neither **Activate** nor MotionSolve support roll back in time, as result, the method which is used actually in **Activate** is GS1, i.e., the Gauss-Seidel method with only one iteration. This method has been illustrated in Fig.12.4;

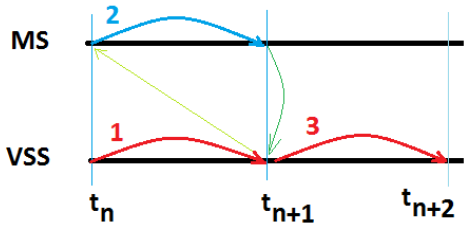


Figure 12.4: Cosimulation method used in **Activate**.

The **Activate** simulator performs the following steps for the co-simulation and synchronisation between slaves, as shown in 12.4.

1. The **Activate** simulator advances the time from t_n to t_{n+1}, and updates the output of all blocks (except slaves).
2. Slaves are called one by one and asked to advance their time from t_n to t_{n+1} and their output is requested at t_{n+1}.
3. The **Activate** simulator takes a new step from t_{n+1} to t_{n+2} using the new output value of slaves computed at t_{n+1}.

4. goto 1

Usually in order to get a correct result, the communication step-size (time between two communication instants) should not be larger than the smallest time-constant of slaves. Often smaller communication step-size results in better convergence but longer simulation time.

12.3 MSplant block parameters

The MSplant block implements co-simulation interface between **Activate** and MotionSolve. Simulations with this block require that Altair MotionSolve solver is installed. The path to the solver and its licensing must be set within **Activate** preferences panel, section Co-Simulation.

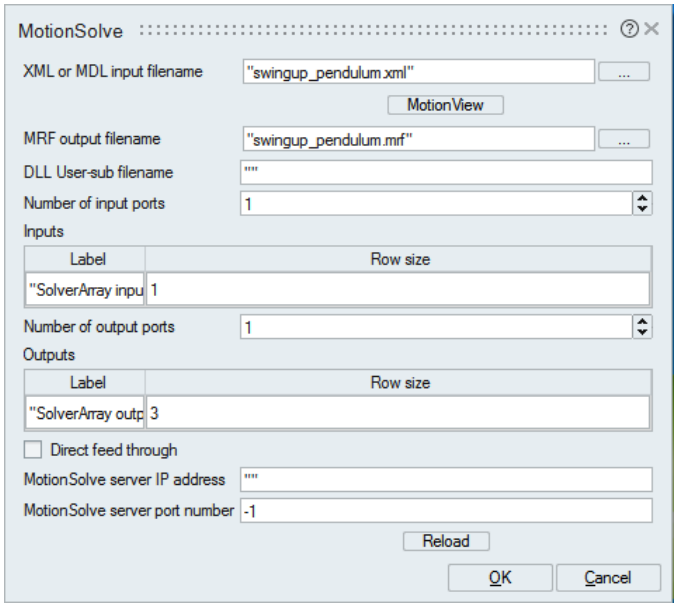


Figure 12.5: Graphical user interface of the MSplant block.

- **XML or MDL input filename:** In this field the name of the MotionSolve model (with extension .xml) or MotionView (with extension .mdl) can be given. The filenames can be either given by an absolute path, or a relative path. If **Activate** and MotionSolve models are in the same folder, no path is required. In the later case the absolute path is obtained from where the **Activate** model is located. If a MotionView is given the (.mdl) file is automatically converted into a MotionSolve (.xml) file.
- **Launch MotionView:** If a MotionView (.mdl) file is provided as the input MotionSolve filename, it is possible to visualize and edit the model in MotionView by clicking on this button. Note: When MotionView is open **Activate** remains inactive.
- **MRF output filename:** This field indicates the name of the MotionView result filename. Note that the folder where is file should be generated is not write-protected. If the corresponding feature is active in the MotionSolve (.xml) model, the MotionView result file (.mrf) will be converted into an H3D file (.h3d). The .mrf and .h3d files can be visualized in MotionView.

Appendix E

Linear Algebra Theorem

Theorem. *Let* $\{\overline{\mathbf{e_1}}, \overline{\mathbf{e_2}}, \cdots, \overline{\mathbf{e_n}}\}$ *be the standard basis vectors in* $\mathbb{R}^n$*. Then the matrix which maps a vector in* $\mathbb{R}^n$ *to a vector in* $\mathbb{R}^m$ *is given by an* $m \times n$ *matrix*

$$\mathbf{A} = \begin{bmatrix} T(\overline{\mathbf{e_1}}) & T(\overline{\mathbf{e_2}}) & \cdots & T(\overline{\mathbf{e_n}}) \end{bmatrix}$$

Proof. Let $\overline{\mathbf{w}}$ be a vector in $\mathbb{R}^n$. The vector can be resolved into components along its standard basis vectors, so we have $\overline{\mathbf{w}} = w_1\overline{\mathbf{e_1}} + w_2\overline{\mathbf{e_2}} + \cdots + w_n\overline{\mathbf{e_n}}$. Then

$$\begin{aligned} T(\mathbf{w}) &= T(w_1\overline{\mathbf{e_1}} + w_2\overline{\mathbf{e_2}} + \cdots + w_n\overline{\mathbf{e_n}}) \\ &= w_1T(\overline{\mathbf{e_1}}) + w_2T(\overline{\mathbf{e_2}}) + \cdots + w_nT(\overline{\mathbf{e_n}}) \\ &= \begin{bmatrix} T(\overline{\mathbf{e_1}}) & T(\overline{\mathbf{e_2}}) & \cdots & T(\overline{\mathbf{e_n}}) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} \\ &= \mathbf{A}\overline{\mathbf{w}} \end{aligned}$$

□

Appendix F

Force Assumptions

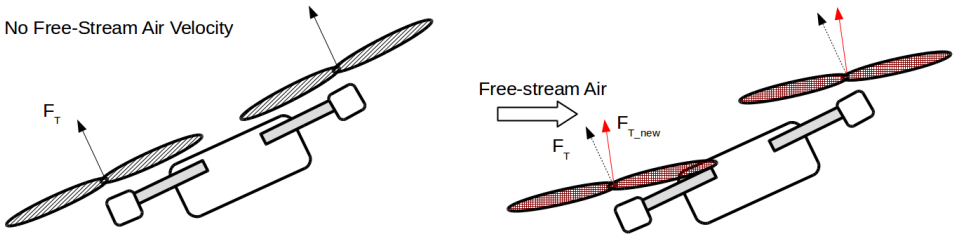
In order to understand the assumptions which were made during this thesis regarding thrust, it is first necessary to understand the fundamentals of this force. This appendix describes the mechanics of the thrust forces which act on a quadcopter in detail. The goal of the appendix is to expose the reader to an understanding of the assumptions which were made regarding the thrust.

Hoffmann *et al.* (2007) and Hoffmann *et al.* (2011) identified that the model usually employed for the thrust produced by a rotor is based on the explicit assumption that the vehicle is in a hovering state. The authors identified four separate aerodynamic effects that influence the flight dynamics (when the vehicle is in motion) which had previously not been taken into account when designing flight controllers, two of which pertain to the thrust force, and will thus be described here.

The first effect is due to blade flapping during translational flight. Blade flapping occurs because the lift produced on an advancing blade is higher than the lift produced on a retreating blade if the vehicle experiences a non-zeros free stream velocity. The periodic blade flapping causes the plane of the rotor to be tilted away from the normal vector with which the corresponding motor is aligned. This scenario is depicted in Figure F.1. This first aerodynamic effect was found to effect the control performance of the vehicles attitude. This aerodynamic effect is assumed to be negligible in this thesis.

The second aerodynamic effect which was identified was the change in rotor thrust **magnitude** due to translational flight. Using the principle of conservation of momentum, the authors showed that the free-stream velocity of air surrounding the vehicle may reduce the thrust produced by the rotors. We will also develop the same mathematical arguments here to expose this secondary aerodynamic effect.

Consider one fundamental fact and one fundamental definition. Firstly, the rotor generates thrust by displacing a volume of air. Secondly, a defini-



(a) A hypothetical scenario where the UAV has just tilted and experiences no free-stream air

(b) The UAV has begun transversing and is subject to a relative free-stream air even if there is no wind

Figure F.1: Blade Flapping

tion: the free stream velocity is defined as the velocity of the air surrounding the UAV, denoted with v_∞.

It is well understood that the rotor changes the speed of the air at the rotor inlet to a higher velocity at the rotor outlet. The volume of air displaced by the rotors therefore changes with free-stream air velocity since this is the velocity of the air at the rotor inlet. This means that the thrust produced by the rotors is dependent on the free-stream air velocity, according to the fundamental fact previously stated. The common assumption that the thrust force is dependent on the rotor speed only then becomes invalid under two conditions, following this argument. The first is when the free-stream velocity of the air surrounding the UAV is none zero. The second is when the UAV translates, thereby resulting in a relative non-zero free stream velocity of the air.

To see this, we derive the thrust forces for the ideal[1] UAV motion. First consider the power produced by the motor in a hovering vehicle state. This power is the product of the thrust force and the change in air speed which is induced by the motors (relative to the free stream velocity v_∞).

$$P_h = F_T v_h \tag{F.0.1}$$

The principle of conservation of momentum can be used to derive a relationship for the air velocity change induced by the rotors. This is given below.

$$v_i = \frac{v_h^2}{\sqrt{(v_\infty \cos\alpha)^2 + (v_i - v_\infty \sin\alpha)^2}} \tag{F.0.2}$$

[1]Ideal in the sense that no vortices are present

where α is the angle of attack of the rotor plane with respect to the free stream velocity. This equation becomes invalid under large angles of attack since the rotors begin to induce excesive vortexes. Assuming then that our UAV is oriented for a small angle of attack, the thrust generated for a certain power input can then be computed as done below.

$$F_T = \frac{P}{v_i - v_\infty \sin\alpha} \tag{F.0.3}$$

Note that a numerical solution will usually be employed to solve v_i (given in equation F.0.2) for use in the above equation[2]. From this equation, it is clear that the thrust force of a rotor is dependent on it's angle of attack with the free-stream velocity of air, as well as the magnitude of the velocity of the free-stream. This aerodynamic effect too is assumed to be negligible in this thesis.

Table F.1 summarizes all the assumptions which are employed to deal with the complexities that arises from the non-linearity inherent in the forces that act upon a UAV. Note that this appendix has provided arguments for the first two assumptions only. The table also provides conditions under which these assumptions are no longer valid, for interest sake. In this thesis, we have assumed that the assumptions always hold.

[2]We obtain exact solutions for the thrust force under extreme angles of attack. These motions correspond to $\cos\alpha = 0$ in equation F.0.2. We find

$$v_i = -\frac{v_c}{2} \pm \sqrt{(\frac{v_c}{2})^2 + v_h^2}$$

Assumption	**Conditions for invalidity**
Thrust is proportional to motor speed only	The thrust produced by a rotor is proportional to the free-stream air velocity
Thurst force remains perpindicular to UAV surface	There is Variation in lift across the rotor when the free-stream air velocity is non-zero.
No frontal drag on UAV sides	The component of free stream velocity which is perpindicular to sides of UAV may be significant
No shear drag on top or bottom surface of UAV	The Component of free-stream air velocity parallel to surface may be non-zero.
Applied moment from frontal drag is zero	The resultant force of the frontal drag doesnt act through the UAV cm.

Table F.1: Summary of common force assumptions

Appendix G

Controller Structure

The controller which was used in the virtual test bench was alluded, so the structure of the controller is provided here in schematic form.

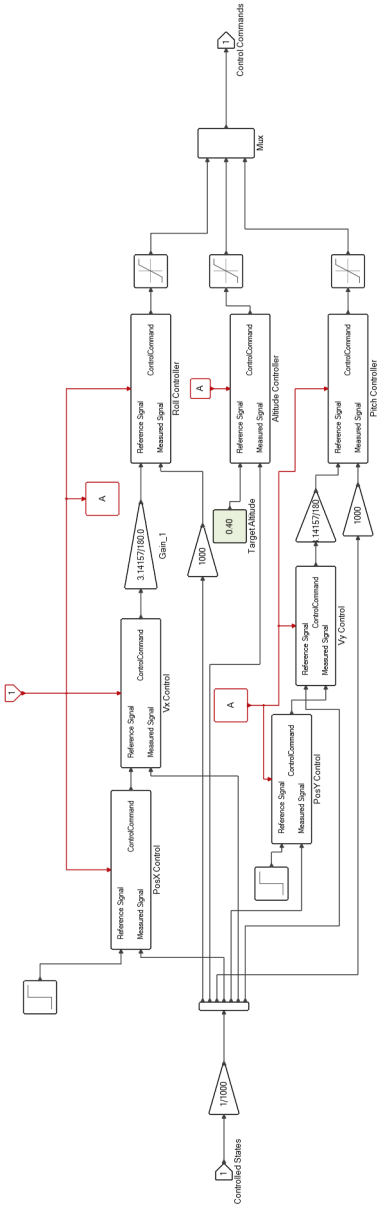


Figure G.1: The structure of the UAV controller

List of References

Abengoa Solar (2012). Heliostat fields [Online]. Available at: `http://www.abengoasolar.com/` [2017, August 08].

Bourke, A.K., O'Donovan, K., Clifford, A., Olaighin, G. and Nelson, J. (2011 Aug). Optimum gravity vector and vertical acceleration estimation using a tri-axial accelerometer for falls and normal activities. In: *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 7896–7899. ISSN 1094-687X.

Cao, Y. and Zu, J. (2010 Oct). Review of the gyroscope free strap-down inertial navigation system. In: *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, vol. 6, pp. V6–413–V6–417. ISSN 2161-9069.

Cardou, P., Fournier, G. and Gagnon, P. (2011 Oct). A nonlinear program for angular-velocity estimation from centripetal-acceleration measurements. *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 5, pp. 932–944. ISSN 1083-4435.

Chen, J.-H., Lee, S.-C. and DeBra, D.B. (1994). Gyroscope free strapdown inertial measurement unit by six linear accelerometers. *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 2, pp. 286–290.

Choi, K. and Lee, I. (2011). Real-time georeferencing of image sequence acquired by a UAV multi-sensor system. In: *2011 International Workshop on Multi-Platform/Multi-Sensor Remote Sensing and Mapping, M2RSM 2011*. ISBN 9781424494040. ISSN 20724292.

Coetzee, J.S. (2017). *Obstacle Avoidance with a Multicopter in a Dynamic Two-Dimensional Environment*. Master's thesis, Mechanical Engineering, University of Stellenbosch, Stellenbosch, South Africa.

Figliola, R.S. and Beasley, D.E. (2011). *Theory and Design for Mechanical Measurements*, chap. 9. John Wiley & Sons (Asia) Pte Ltd, Asia.

Franklin, G., Powell, J. and Emami-Naeini, A. (2010). *Feedback control of dynamic systems*, chap. 9. Pearson.

Greenwood, D. (2006). *Advanced Dynamics*. Cambridge University Press. ISBN 9780521029933.
Available at: `https://books.google.com/books?id=r2CSj1A-zWQC`

Gross, J.N., Gu, Y., Rhudy, M.B., Gururajan, S. and Napolitano, M.R. (2012). Flight-test evaluation of sensor fusion algorithms for attitude estimation. *IEEE Transactions on Aerospace and Electronic Systems*. ISSN 00189251.

Groves, P. (2013). *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems, Second Edition:*. Artech House. ISBN 9781608070053.

Han, J.-H. and Lee, I. (). Position and Atti GPS, IM.

Hoffmann, G.M., Huang, H., Waslander, S.L. and Tomlin, C.J. (2007). Quadrotor Helicopter Flight Dynamics and Theory and Experiment. In: *AIAA Guidance, Navigation and Control Conference and Exhibit*.

Hoffmann, G.M., Huang, H., Waslander, S.L. and Tomlin, C.J. (2011). Precision flight control for a multi-vehicle quadrotor helicopter testbed. *Control Engineering Practice*. ISSN 09670661.

Janota, A., Simak, V., Nemec, D. and Hrbcek, j. (2015). Improving the precision and speed of euler angles computation from low cost rotation sensor data. *Sensors*, vol. 15, no. 3, pp. 7016–7039. ISSN 1424-8220.

Kowalczuk, Z. and Merta, T. (2015 July). Evaluation of position estimation based on accelerometer data. In: *2015 10th International Workshop on Robot Motion and Control (RoMoCo)*, pp. 246–251.

Leishman, R.C., Macdonald, J.C., Beard, R.W. and McLain, T.W. (2014 Feb). Quadrotors and accelerometers: State estimation with an improved dynamic model. *IEEE Control Systems*, vol. 34, no. 1, pp. 28–41. ISSN 1066-033X.

Li, D.R., Liu, Y. and Yuan, X.X. (2013). Image-based self-position and orientation method for moving platform. *Science China Information Sciences*. ISSN 1674733X.

Lock, J.C. (2016). *Pose Error Estimation of a Quadcopter in the Outdoors*. Master's thesis, Mechanical Engineering, University of Stellenbosch, Stellenbosch, South Africa.

Mao, J. (1986). Optimal orthonormalization of the strapdown matrix by using singular value decomposition. *Computers & Mathematics with Applications*, vol. 12, no. 3, pp. 353 – 362. ISSN 0898-1221.

Mingli, D., Qingdong, Z., Qi, W. and Changhong, W. (2006 Aug). Feasibility analysis of accelerometer configuration of non-gyro micro inertial measurement unit. In: *2006 CES/IEEE 5th International Power Electronics and Motion Control Conference*, vol. 3, pp. 1–4.

Minnaar, N. (2015 January 5). Quadcopter (Personal Photo).

Minnaar, N. and Smit, W. (2017 February). Removing accelerometer redundancy in non-gyro inertial measurement unit. Unpublished Proceeding.

MPU-9250 Product Specification (2013 8). *MPU-6000 and MPU-6050 Product Specification*. InvenSense Inc. Rev. 3.4.

Nagai, M., Chen, T., Shibasaki, R., Kumagai, H. and Ahmed, A. (2009). UAV-borne 3-D mapping system by multisensor integration. *IEEE Transactions on Geoscience and Remote Sensing*. ISSN 01962892.

Nemra, A. and Aouf, N. (2010). Robust INS/GPS Sensor Fusion for UAV Localization Using SDRE Nonlinear Filtering. *IEEE SENSORS JOURNAL*, vol. 10, no. 4.

Oh, S.M. (2010). Multisensor fusion for autonomous UAV navigation based on the unscented Kalman filter with sequential measurement updates. In: *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. ISBN 9781424454266.

Poole, D. (2011). *Linear Algebra: A Modern Introduction*. Brooks/Cole. ISBN 9780538735445.
Available at: `https://books.google.com/books?id=y-4CSgAACAAJ`

Rieke, M., Foerster, T., Geipel, J. and Prinz, T. (). High-precision Positioning and Real-time Data Processing of UAV-Systems. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Solar Thermal Energy Research Group (2015). The suncopter project [Online]. Available at: `http://sterg.sun.ac.za/researchsuncopter/` [2017, August 08].

Swift Navigation[Online] (2017). Available at: `https://www.swiftnav.com/piksi-multi`, [2017, November 19].

UAS Vision (2016). Drones cut cost of thermographic pv panel inspections [Online]. Available at: http://www.uasvision.com/2016/09/15/drones-cut-cost-of-thermographic-pv-panel-inspections/ [2017, August 08].

van Breda, R.J. (2016). *Vector Field Histogram Star Obstacle Avoidance System for Multicopters.* Master's thesis, Mechanical Engineering, University of Stellenbosch, Stellenbosch, South Africa.

Wang, Q., Ding, M. and Zhao, P. (2003 Nov). A new scheme of non-gyro inertial measurement unit for estimating angular velocity. In: *Industrial Electronics Society, 2003. IECON '03. The 29th Annual Conference of the IEEE*, vol. 2, pp. 1564–1567 Vol.2.

Xue, L., Yuan, W., Chang, H. and Jiang, C. (2009 Jan). Mems-based multi-sensor integrated attitude estimation technology for mav applications. In: *2009 4th IEEE International Conference on Nano/Micro Engineered and Molecular Systems*, pp. 1031–1035.

Yoo, C.-s., Ahn, L.-k. and Ver QOm, H. (2003). Low Cost GPS/INS Sensor Fusion System for UAV Navigation.

Zheng, X. (2017). *An Error Propagation Model of a Heliostat.* Master's thesis, Mechanical Engineering, University of Stellenbosch, Stellenbosch, South Africa.