2014-02-11

Development of Tailsitter Hover Estimation and Control

Jason M. Beach
Brigham Young University - Provo

Development of Tailsitter Hover Estimation and Control

Jason M. Beach

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

Brigham Young University

February 2014

ABSTRACT

Development of Tailsitter Hover Estimation and Control

Jason M. Beach
Department of Mechanical Engineering, BYU
Master of Science

UAVs have become an essential tool in many market segments, particularly the military where critical intelligence can be gathered by them. A tailsitter aircraft is a platform whose purpose is to efficiently merge the range and endurance of fixed-wing aircraft with the VTOL capabilities of rotorcraft and is of significant value in applications where launch and recovery area is limited or the use of launch and recovery equipment is not desirable.

Developing autopilot software for a tailsitter UAV is unique in that the aircraft must be autonomously controlled over a much wider range of attitudes than conventional UAVs. Assumptions made in conventional estimation and control algorithms are not valid for tailsitter aircraft because of routine operation around gimbal lock. Quaternions are generally employed to overcome the limitations Euler angles; however, adapting the attitude representation to work at a full range of attitudes is only part of the solution. Kalman filter measurement updates and control algorithms must also work at any orientation. This research presents several methods of incorporating a magnetometer measurement into an extended Kalman filter. One method combines magnetometer and accelerometer sensor data using the solution to Wahba's problem to calculate an overall attitude measurement. Other methods correct only heading error and include using two sets of Euler angles to update the estimate, using quaternions to determine heading error and Euler angles to update the estimate, and using only quaternions to update the estimate.

Quaternion feedback attitude control is widely used in tailsitter aircraft. This research also shows that in spite of its effective use in spacecraft, using the attitude error calculated via quaternions to drive flight control surfaces may not be optimal for tailsitters. It is shown that during hover when heading error is present, quaternion feedback can cause undesired behavior, particularly when the heading error is large. An alternative method for calculating attitude error called resolved tilt-twist is validated, improved, and shown to perform better than quaternion feedback.

Algorithms are implemented on a commercially available autopilot and validation is performed using hardware in loop simulation. A custom interface is used to receive autopilot commands and send the autopilot simulated sensor information.

The final topic covered deals with the tailsitter hovering in wind. As the tailsitter hovers, wind can cause the tailsitter to turn such that the wind is perpendicular to the wings. Wind tunnel data is taken and analyzed to explain this behavior.

Keywords: tailsitter, unmanned air vehicle, magnetometer, Kalman filter, estimation, quaternion, attitude control

ACKNOWLEDGMENTS

This thesis would not be complete without first taking the opportunity to express gratitude to those whose efforts have made it possible. First, to my graduate committee; Dr. McLain and Dr. Beard, who have guided much of the research performed and Dr. Bowman who allowed me to gain experience with RC aircraft in his lab while trying to find a research problem. I am also grateful to the other members of the the tailsitter team, Matt Argyle and Nathan Edwards, and to Stephen Morris and the MLB Company. I cannot imagine a better research project to work on and I am grateful for them allowing me to be a part of their team.

There are also many members of the MAGICC lab who have helped in one way or another– Jesse Wynn and Dallin Briggs for their help in building the Y-Bat, Justin Mackay for answering an endless stream of random questions as I tried to figure things out and to Dan Koch and those who helped me to move our motion capture system outside to allow us to obtain the data we did.

Funding from ASEE and the SMART scholarship allowed me to focus on research while providing for my family and is appreciated.

Most importantly, I would like to thank my wife, Heather, for supporting my educational goals while taking care of our twin boys Michael and Ethan. Her sacrifices in keeping me clothed and fed and maintaining a semblance of order in our home were an essential component of this work. It is my hope that this work will be an example to Michael and Ethan of what can be accomplished with effort and hard work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

$\mathbf{A}$	Dynamic system state transition matrix
$\mathbf{a}^{[\ ]}$	Acceleration vector expressed in the [ ] frame
$\tilde{\mathbf{a}}$	MEKF attitude error
$\mathbf{B}$	Dynamic system input Jacobian matrix
$\mathbf{b}^{[\ ]}$	Magnetic vector expressed in the [ ] frame
$\{\mathbf{b}\}$	Set of vectors expressed in body frame for solution to Wahba's problem
b	Wing span
$\mathbf{C}$	Dynamic system output matrix
C_{ℓ,δ_v}	Control vane lift coefficient slope
$\mathrm{c}_{[\ ]}$	Cosine of the angle [ ]
$\mathbf{e}$	Unit length axis of rotation
f	External applied force
$\mathbf{g}$	Normalized acceleration vector
g	Gravity
k	Tunable gain
$\mathbf{L}$	Kalman gain matrix
M	Externally applied moment
m	Mass
$\mathbf{P}$	Estimation error covariance matrix
p	Angular velocity about the body x-axis
$\mathbf{Q}$	Process noise covariance matrix
q	Angular velocity about the body y-axis
$\mathbf{R}_{[\ ]}$	Measurement covariance matrix for the [ ] measurement.
$\mathbf{R}_a^b$	A rotation matrix that rotates the coordinate frame a to frame b
$\{\mathbf{r}\}$	Set of vectors expressed in reference frame for solution to Wahba's problem
$r_{[\ ]}$	The [ ] element of a rotation matrix
r	Angular velocity about the body z-axis
S	Gain scheduler
S_v	Control vane surface area
$\mathrm{s}_{[\ ]}$	Sine of the angle [ ]
T_s	Sample period
$\mathbf{u}$	Dynamic system input vector
u	Inertial velocity of the airframe, projected onto the body x-axis
v	Inertial velocity of the airframe, projected onto the body y-axis
w	Inertial velocity of the airframe, projected onto the body z-axis
x	Curve fit input angle
$\mathbf{x}$	Dynamic system state vector
V_b	Air velocity over control vanes
V_w	Wind velocity magnitude

	Greek Symbols
β	Wind angle relative to tailsitter
χ_w	Inertial referenced wind direction
δ	Magnetic declination
δ_a	Aileron control effort
δ_e	Elevator control effort
δ_r	Rudder control effort
δ_t	Throttle control effort
η	Attitude quaternion
$\boldsymbol{\eta}$	The vector part of the quaternion η
$\boldsymbol{\omega}$	Angular velocity vector
ϕ	Euler angle rotation about the body x-axis
ψ	Euler angle rotation about the body z-axis. Without subscript, generically refers to vehicle heading
ψ_m	Magnetic heading
$\sigma_{[\ ]}$	Noise term with standard deviation $\sigma_{[\ ]}$
Θ	Angle of rotation
θ	Euler angle rotation about the body y-axis

	Subscripts, superscripts, and other indicators
$\mathbf{R}$	Bold capital letters are matrices
$\mathbf{v}$	Bold lower case letters are vectors
$\mathbf{v}^a$	A vector referenced to and expressed in the coordinate frame a
$\mathbf{v}^a_b$	A vector referenced to the b coordinate frame and expressed in the a coordinate frame
$\dot{[\ ]}$	indicates the first derivative of [ ] with respect to time
$\ddot{[\ ]}$	indicates the second derivative of [ ] with respect to time
$[\ ]^*$	indicates a measured value. Unless otherwise indicated, expressed in the body frame
$[\ ]^d$	indicates a desired or commanded value of [ ]
$\hat{[\ ]}$	indicates an estimated value of [ ]
$\tilde{[\ ]}$	indicates [ ] is an error value
$[\ ]^v$	indicates [ ] is expressed in the vehicle frame
$[\ ]^{v1}$	indicates [ ] is expressed in the vehicle-1 frame
$[\ ]^{v2}$	indicates [ ] is expressed in the vehicle-2 frame
$[\ ]^h$	indicates [ ] is expressed in the hover frame
$[\ ]^{h1}$	indicates [ ] is expressed in the hover-1 frame
$[\ ]^{h2}$	indicates [ ] is expressed in the hover-2 frame
$[\ ]^b$	indicates [ ] is expressed in the body frame
$[\ ]_0$	is the scalar component of the quaternion [ ]
$[\ ]_x$	is the x vector component of [ ]
$[\ ]_y$	is the y vector component of [ ]
$[\ ]_z$	is the z vector component of [ ]
$[\ ]^+$	is the value of [ ] immediately following an update
$[\ ]^-$	is the value of [ ] immediately preceding an update

CHAPTER 1. INTRODUCTION

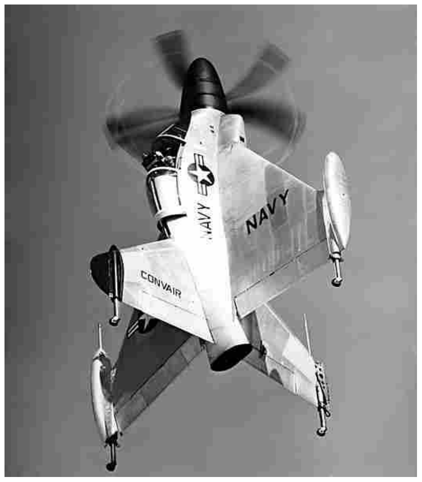
Unmanned Air Vehicle (UAV) use has become widespread in many market segments, particularly the military where their use enhances the gathering of intelligence. One challenge that exists for current fixed-wing UAVs is the launch and recovery equipment required for their use. This challenge is somewhat overcome by rotorcraft; however, they require more power than their fixed-wing counterparts which limits their range and endurance. An aircraft that could merge the efficiency of a fixed-wing aircraft with the vertical takeoff and landing (VTOL) capabilities of rotorcraft would be of tremendous value.

The tailsitter aircraft does just that. It can take off vertically with no special launch equipment, transition to level flight to perform its mission and then transition back to hover flight for landing, with no recovery equipment. To be viable, however, it must be robust to external disturbances during critical phases of flight, particularly landing, and must be safe to vehicle operators. Most research has focused on the transition between the vertical and horizontal flight regimes and has been on vehicles with exposed propellers at the nose of the vehicle.

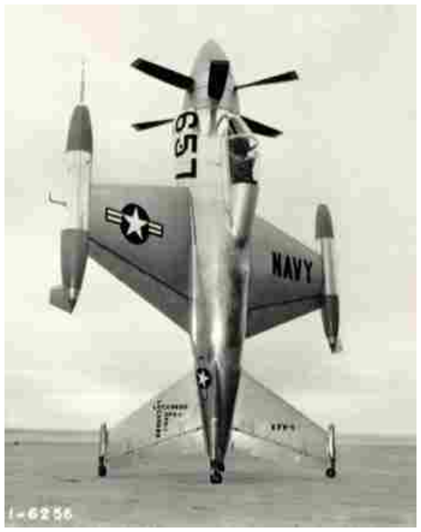
1.1 Background

Tailsitter research can be traced back to 1944 with the Focke-Wulf *Triebflügel*, shown in Figure 1.1(a). Intended to protect important German factories and other sensitive areas not located by airstrips in World War II, a complete prototype never made it off the drawing board, ironically due to Allied forces bombing the facility where it was being developed. In 1951, Lockheed and Convair were awarded contracts to design a VTOL aircraft from which the XFY *Pogo*, Figure 1.1(b), and the XFV *Salmon*, Figure 1.1(c), were built and successfully flown. Both had the major operational disadvantages of being relatively slow in level flight (as compared to jet aircraft of that time) and being difficult to land because of pilot orientation. Development of vectored thrust aircraft such as the Hawker Harrier in the 1960's provided the needed VTOL capability and curtailed the need for manned tailsitter aircraft.

(a) Focke-Wulf *Triebflügel*



(b) Convair XFY *Pogo*

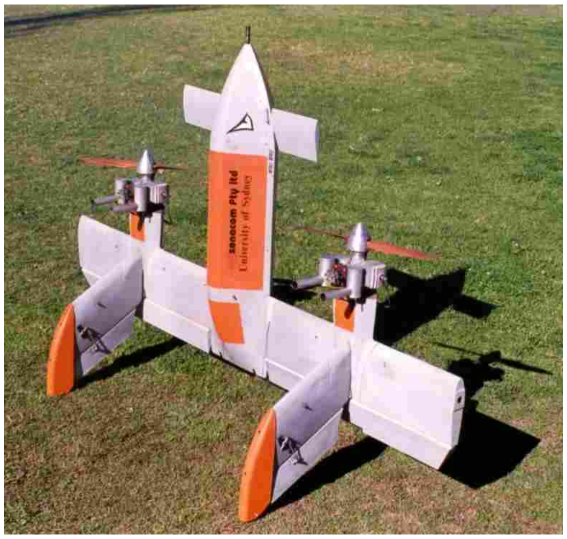


(c) Lockheed XFV *Salmon*

Figure 1.1: Manned tailsitter aircraft.

Today as UAV use has increased, tailsitters have once again emerged as a viable VTOL design since they are mechanically much simpler than vectored thrust aircraft and pilot orientation is not an issue. One difficulty in autonomously controlling a tailsitter is the highly nonlinear aerodynamics that occur in the transition between vertical and horizontal flight. Considerable research has been done to address this including that found in [1–17]. Much of this research has been performed using conventional fixed-wing aircraft that cannot actually perform vertical takeoffs or landings, but can perform transition maneuvers once airborne [6, 11, 13, 14, 16]. Matsumoto [16] proposes an algorithm to reject large disturbances while hovering, but does not perform outdoor flight testing. This work is explored in more depth in Section 5.1.2.

A particularly thorough treatment of tailsitter work has been performed at the University of Sydney on a tailsitter called the T-Wing [1–5, 12], shown in Figure 1.2(a). Extensive flight testing is reported in [12]. It is the only literature found to date that contains specific tailsitter hover flight test data in windy conditions. One of the most recent publications on tailsitter aircraft details



(a) University of Sydney/Sonacom T-Wing

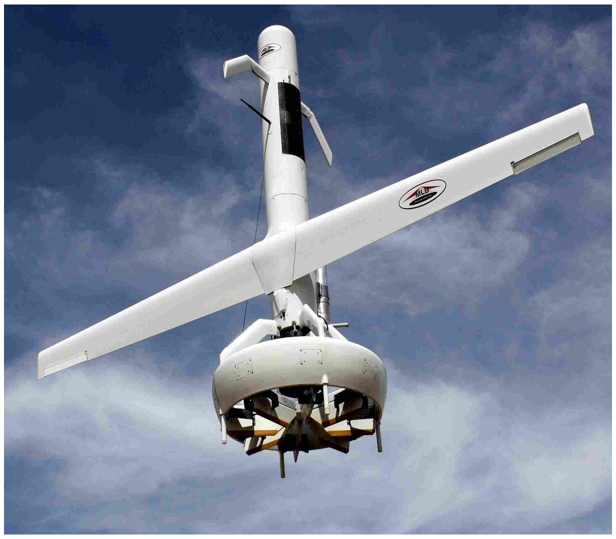

(b) MLB V-Bat (Courtesy of the MLB Company)

Figure 1.2: Unmanned tailsitter aircraft.

research performed at the Korea Advanced Institute of Science and Technology (KAIST) [18]. In their work, they perform extensive dynamic modeling on a tailsitter and propose an $\mathcal{L}_1$ adaptive controller. Simulation and flight testing are also reported.

Another notable contribution is that by Knoebel [9], a former Brigham Young University (BYU) student. His work in developing a quaternion estimator and model reference adaptive controller provided much of the foundation for tailsitter work at BYU that has occurred since, including this work. Osborne [10] studied the transitions between hover and level flight. Millet [15] looked at performing precision tailsitter landings using computer vision. Hogge [17] detailed the overall design of a tailsitter airframe.

More recently, BYU has worked with the MLB company [19] to design the control algorithms of a new tailsitter design called the Vertical-Bat or V-Bat, shown in Figure 1.2(b). In phase I of this project, the V-Bat was designed and built with functional control algorithms for all flight regimes. A MATLAB simulator was also built and some limited flight testing occurred. Now in phase II, the control algorithms are being implemented on a scale model of the V-Bat.

1.2 Contributions

The primary challenge with tailsitter aircraft is adapting current autopilot estimation and control algorithms to work at all vehicle attitudes. The research performed makes the following contributions:

- **Kalman Filter Magnetometer Measurement Update** - The traditional method for using a magnetometer relies on the use of Euler angles to determine heading. Several alternative methods that work at all vehicle attitudes are compared. Simulation and hardware testing results are given.

- **Alternative Method of Computing Attitude Error Validated and Improved** - Using quaternions to calculate attitude error is widely used for attitude control in tailsitter aircraft because gimbal lock is not an issue. This research shows that during hover, using the vector part of an attitude error quaternion can cause unexpected and undesired behavior when large heading errors are present. An alternate method called resolved tilt-twist is analyzed, implemented in hardware and examined for its suitability.

- **Autonomous Hover Control Demonstrated** - Hardware testing was performed on a commercially available UAV. This platform is different in that it is specifically designed as a tailsitter aircraft and that the motor is located much closer to control surfaces. While this increases the vehicle's control authority, particularly in descent, it makes the vehicle virtually impossible to control in hover without the use of feedback control.

- **Wind Analysis Performed** - As the tailsitter hovers, wind can cause the tailsitter to turn such that the wind is perpendicular to the wings. Wind tunnel data was taken and analyzed to explain this behavior.

1.3 Document Outline

This document proceeds as follows: Chapter 2 reviews basic concepts on vehicle attitude representation that are used throughout this work. Chapter 3 gives an overview of some of the development tools that were used in this research including simulators and hardware platforms. Chapter 4 compares different ways to correct heading error in the vehicle attitude estimate that are not affected by gimbal lock. Chapter 5 examines vehicle attitude control and looks in-depth at calculating attitude error using quaternions and resolved tilt-twist. Chapter 6 examines hover behavior in the presence of wind and provides wind tunnel data and analysis of this behavior. Chapter 7 summarizes the research performed and gives recommendations for future work. Lastly, Appendix A provides conversions between different attitude representations and Appendix B reviews Wahba's problem and a solution to it.

CHAPTER 2. REVIEW OF ATTITUDE REPRESENTATIONS

Attitude representation at its most fundamental level consists of representing the rotation from an inertial coordinate frame to a body-fixed coordinate frame. Euler angles are the easiest and most physically intuitive method for representing vehicle attitude; however, the well documented gimbal lock problem limits their suitability for a tailsitter type aircraft. The unit-norm quaternion is the lowest dimension, globally nonsingular attitude representation [20]. Quaternions also have computational advantages as noted in [21]. These qualities make the quaternion an attractive method of attitude representation for a tailsitter aircraft.

Each of the representations start with an inertial coordinate frame fixed at an arbitrary location. A copy of this frame is translated to the tailsitter center of mass and is referred to as the vehicle frame [22]. Both the inertial and vehicle frames are oriented such that the x-axis points

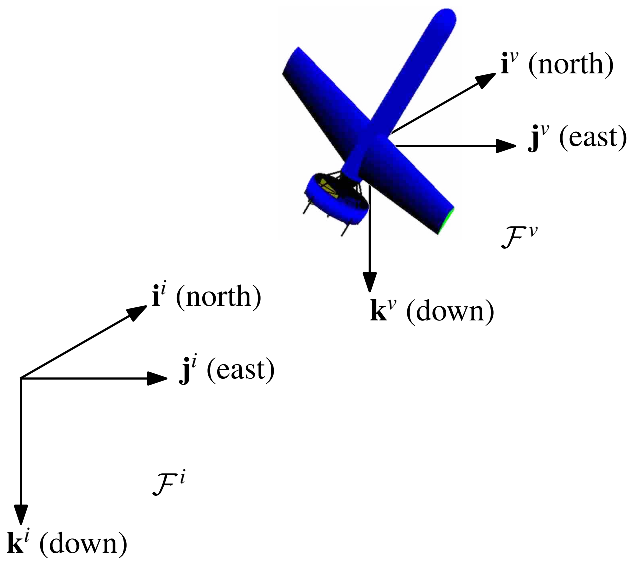


Figure 2.1: Inertial frame ($\mathcal{F}^i$) and vehicle coordinate frame ($\mathcal{F}^v$).

north, y-axis points east and z-axis points to the center of the earth. Because of this, rotations referenced to the vehicle frame are identical to rotations referenced to the inertial frame.

This chapter reviews important concepts in how the rotation from the vehicle frame to the body frame is represented. Section 2.1 gives a brief overview of rotation matrices. Section 2.2 discusses level flight Euler angles and some of their limitations. Section 2.3 derives a set of Euler angles useful for hover flight of a tailsitter. Section 2.4 reviews basic concepts of quaternions. The chapter is concluded with an overview of Wahba's problem in Section 2.5.

2.1 Rotation Matrices

The most basic way of representing the rotation between an inertial coordinate frame and a body-fixed coordinate frame is the rotation matrix. A rotation matrix is a 3×3 unitary matrix that when multiplied by can either express a vector in a new coordinate frame or rotate a vector within the current coordinate frame. In the context of attitude representation, of primary interest is being able to relate vectors expressed in the vehicle frame to sensor measurements expressed in the body frame. As a useful example, the axes of vehicle coordinate frame can be expressed in terms of the body frame by

$$\begin{bmatrix} \mathbf{i}_v^b & \mathbf{j}_v^b & \mathbf{k}_v^b \end{bmatrix} = \mathbf{R}_v^b \begin{bmatrix} \mathbf{i}^v & \mathbf{j}^v & \mathbf{k}^v \end{bmatrix}, \tag{2.1}$$

where $\mathbf{i}^v$, $\mathbf{j}^v$ and $\mathbf{k}^v$ are the unit vectors of the vehicle coordinate frame and $\mathbf{R}_v^b$ is the rotation matrix that rotates the vehicle frame to the body frame. Thus, $\mathbf{i}_v^b$, $\mathbf{j}_v^b$ and $\mathbf{k}_v^b$ are the axes of the vehicle coordinate frame expressed in the body coordinate frame. The arrangement of $\mathbf{i}^v$, $\mathbf{j}^v$ and $\mathbf{k}^v$ in Equation 2.1 are such that they form the identity matrix. This leads to an observation that the columns of $\mathbf{R}_v^b$ are the vehicle frame axes expressed in the body-fixed frame.

Since a rotation matrix is unitary, the inverse of the matrix or the inverse rotation is simply its transpose, or

$$\mathbf{R}_b^v = \mathbf{R}_v^{b\top}. \tag{2.2}$$

This allows us to express the axes of the body frame in the vehicle coordinate frame by

$$\begin{bmatrix} \mathbf{i}_b^v & \mathbf{j}_b^v & \mathbf{k}_b^v \end{bmatrix} = \mathbf{R}_b^v \begin{bmatrix} \mathbf{i}^b & \mathbf{j}^b & \mathbf{k}^b \end{bmatrix}. \tag{2.3}$$

A similar observation can be made: the columns of $\mathbf{R}_b^v$ are the body axes expressed in the vehicle frame. Noting from Equation 2.2 that the columns of $\mathbf{R}_b^v$ are the rows of $\mathbf{R}_v^b$, the rows of $\mathbf{R}_v^b$ are also the body axes expressed in the vehicle frame. These observations are useful and used in future sections.

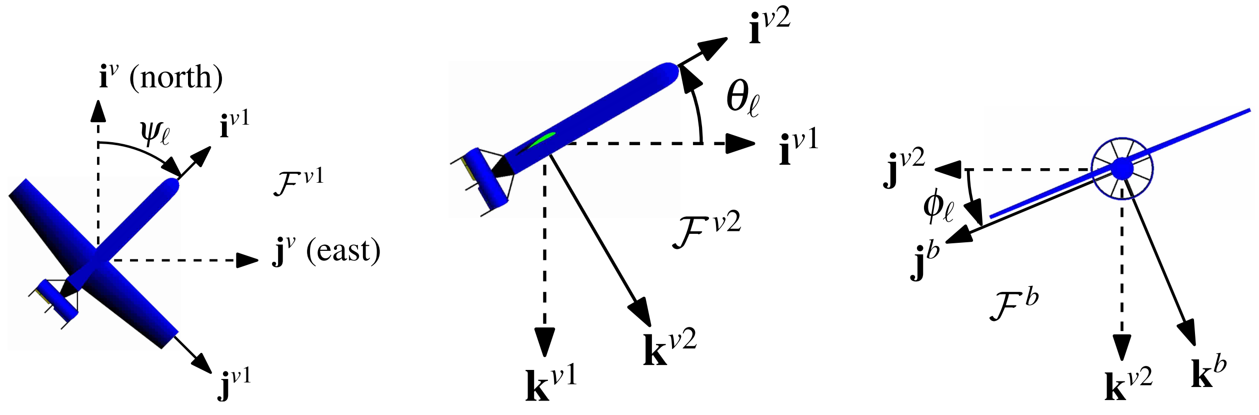
2.2 Level Flight Euler Angles

Euler angles represent three consecutive rotations about body-fixed axes with the 3-2-1 rotation sequence being the most commonly used in aircraft literature. When performing these rotations several intermediate coordinate system definitions are needed and are now defined. These rotations are [21]:

(a) Rotate the vehicle frame about the $\mathbf{k}^v$ axis by the heading angle, ψ_ℓ, into the vehicle-1 frame as shown in Figure 2.2(a).

(b) Rotate the vehicle-1 frame about the $\mathbf{j}^{v1}$ axis by the elevation angle, θ_ℓ, into the vehicle-2 frame as shown in Figure 2.2(b).

(c) Rotate the vehicle-2 frame about the $\mathbf{i}^{v2}$ axis by the bank angle, ϕ_ℓ, into the body frame as shown in Figure 2.2(c).



(a) Vehicle-1 frame (b) Vehicle-2 frame (c) Body frame

Figure 2.2: Level flight Euler angle coordinate frames.

Phillips [21] carefully notes that the heading, elevation and bank angles are similar to, but subtly different than the yaw, pitch and roll angles. Roll, pitch and yaw are rotations about the body axes $\mathbf{i}^b$, $\mathbf{j}^b$ and $\mathbf{k}^b$, respectively, whereas the bank, elevation and heading angles are about the intermediate frames $\mathbf{i}^{v2}$, $\mathbf{j}^{v1}$ and $\mathbf{k}^v$.

Mathematically these three rotations are represented by

$$\mathbf{R}_v^{v1}(\psi_\ell) = \begin{bmatrix} \cos\psi_\ell & \sin\psi_\ell & 0 \\ -\sin\psi_\ell & \cos\psi_\ell & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2.4}$$

$$\mathbf{R}_{v1}^{v2}(\theta_\ell) = \begin{bmatrix} \cos\theta_\ell & 0 & -\sin\theta_\ell \\ 0 & 1 & 0 \\ \sin\theta_\ell & 0 & \cos\theta_\ell \end{bmatrix}, \tag{2.5}$$

$$\mathbf{R}_{v2}^{b}(\phi_\ell) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_\ell & \sin\phi_\ell \\ 0 & -\sin\phi_\ell & \cos\phi_\ell \end{bmatrix}. \tag{2.6}$$

With sine and cosine abbreviated as “s” and “c,” respectively, these are combined to form

$$\mathbf{R}_v^b(\phi_\ell,\theta_\ell,\psi_\ell) = \mathbf{R}_{v2}^b(\phi_\ell)\mathbf{R}_{v1}^{v2}(\theta_\ell)\mathbf{R}_v^{v1}(\psi_\ell), \tag{2.7}$$

$$= \begin{bmatrix} c_{\theta_\ell} c_{\psi_\ell} & c_{\theta_\ell} s_{\psi_\ell} & -s_{\theta_\ell} \\ s_{\phi_\ell} s_{\theta_\ell} c_{\psi_\ell} - c_{\phi_\ell} s_{\psi_\ell} & s_{\phi_\ell} s_{\theta_\ell} s_{\psi_\ell} + c_{\phi_\ell} c_{\psi_\ell} & s_{\phi_\ell} c_{\theta_\ell} \\ c_{\phi_\ell} s_{\theta_\ell} c_{\psi_\ell} + s_{\phi_\ell} s_{\psi_\ell} & c_{\phi_\ell} s_{\theta_\ell} s_{\psi_\ell} - s_{\phi_\ell} c_{\psi_\ell} & c_{\phi_\ell} c_{\theta_\ell} \end{bmatrix}. \tag{2.8}$$

$\mathbf{R}_v^b(\phi_\ell,\theta_\ell,\psi_\ell)$ is the Euler angle based rotation matrix representing the rotation of the vehicle frame to the body frame. The subscript on $\mathbf{R}$ indicates the coordinate frame being rotated and the superscript indicates the frame being rotated to. The subscript on the angles distinguishes between the standard level flight Euler angles and the hover Euler angles described in the following section.

It is well known that for this rotation sequence, when $\theta_\ell = 90°$, one degree of freedom is lost in a condition called gimbal lock. In this state, bank angle and heading are ambiguous. To provide some physical intuition and overcome some of the limits of being close to gimbal lock, an alternate set of Euler angles is now derived.

2.3 Hover Euler Angles

The hover coordinate frame is located at the tailsitter center of mass with the x-axis pointing up, the y-axis pointing east and the z-axis pointing north. Note that this is simply the vehicle frame rotated by 90° about the $\mathbf{j}^v$ axis.

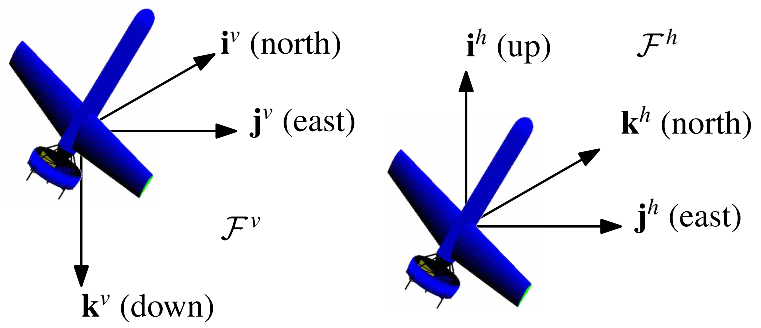


Figure 2.3: The hover coordinate frame ($\mathcal{F}^h$) is simply the vehicle frame ($\mathcal{F}^v$) rotated by 90° about the $\mathbf{j}^v$ axis.

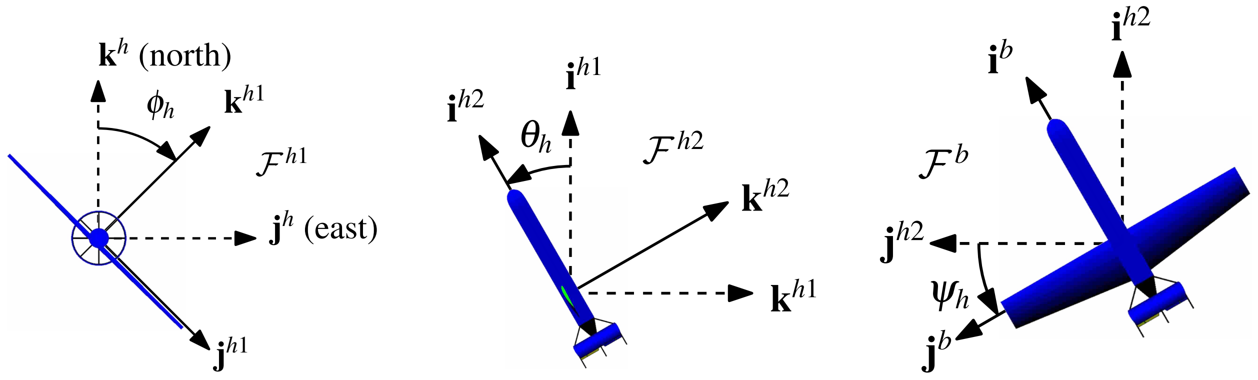
The complete rotation sequence is

(a) Rotate the vehicle frame 90° about the $\mathbf{j}^v$ axis into the hover frame, shown in Figure 2.3.

(b) Rotate the hover frame about the $\mathbf{i}^h$ axis by the heading angle, $-\phi_h$, into the hover-1 frame as shown in Figure 2.4(a). The minus sign is necessary to maintain heading sense (i.e., so that $\phi_h = 90°$ still results in the belly pointing east).

(c) Rotate the hover-1 frame about the $\mathbf{j}^{h1}$ axis by the elevation angle, θ_h, into the hover-2 frame as shown in Figure 2.4(b).

(d) Rotate the hover-2 frame about the $\mathbf{k}^{v2}$ axis by the bank angle, ψ_h, into the body frame as shown in Figure 2.4(c).



(a) Hover-1 frame (b) Hover-2 frame (c) Body frame

Figure 2.4: Hover flight Euler angle coordinate frames.

Mathematically this is represented by

$$\begin{aligned}\mathbf{R}_v^b(\phi_h,\theta_h,\psi_h) &= \mathbf{R}_{h2}^b(\psi_h)\mathbf{R}_{h1}^{h2}(\theta_h)\mathbf{R}_h^{h1}(-\phi_h)\mathbf{R}_v^h,\\ &= \begin{bmatrix} c_{\psi_h} & s_{\psi_h} & 0\\ -s_{\psi_h} & c_{\psi_h} & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix} c_{\theta_h} & 0 & -s_{\theta_h}\\ 0 & 1 & 0\\ s_{\theta_h} & 0 & c_{\theta_h}\end{bmatrix}\begin{bmatrix} 1 & 0 & 0\\ 0 & c_{\phi_h} & -s_{\phi_h}\\ 0 & s_{\phi_h} & c_{\phi_h}\end{bmatrix}\begin{bmatrix} 0 & 0 & -1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{bmatrix},\\ &= \begin{bmatrix} -s_{\phi_h}s_{\psi_h} - c_{\phi_h}s_{\theta_h}c_{\psi_h} & c_{\phi_h}s_{\psi_h} - s_{\phi_h}s_{\theta_h}c_{\psi_h} & -c_{\theta_h}c_{\psi_h}\\ c_{\phi_h}s_{\theta_h}s_{\psi_h} - s_{\phi_h}c_{\psi_h} & c_{\phi_h}c_{\psi_h} + s_{\phi_h}s_{\theta_h}s_{\psi_h} & c_{\theta_h}s_{\psi_h}\\ c_{\phi_h}c_{\theta_h} & s_{\phi_h}c_{\theta_h} & -s_{\theta_h}\end{bmatrix}. \qquad (2.9)\end{aligned}$$

Like any other sequence of three rotations, this too is affected by gimbal lock when the elevation angle $\theta_h = \pm 90°$. To cover a full range of attitudes, it is necessary to switch between level and hover Euler angles depending on elevation angle. Because of this, Euler angles are still not suitable for some uses such as the tailsitter attitude estimate state vector. They can be used to provide intuition about the vehicle attitude. Also, it is important to note that in both level and hover Euler angles, we have chosen to maintain ϕ, θ and ψ as rotations about the respective x, y and z axes. Doing so means that when using the level Euler angles, ψ_ℓ defines heading, but in hover mode ϕ_h does.

2.4 Quaternions

Instead of three separate rotations, the quaternion expresses attitude in terms of a single rotation. The *Euler-Rodrigues symmetric parameters* are related to a unit-length axis of rotation $\mathbf{e}$ and an angle of rotation Θ by

$$\eta = \begin{bmatrix} \eta_0 \\ \boldsymbol{\eta} \end{bmatrix} = \begin{bmatrix} \cos\frac{\Theta}{2} \\ \mathbf{e}\sin\frac{\Theta}{2} \end{bmatrix}. \tag{2.10}$$

The quaternion has two parts–a scalar component and a 3-element vector component. There is variation in the literature on whether the scalar component is expressed as the first or last element of the quaternion. Here we have chosen to place it as the first component and to express the vector part in bold. As such, the identity quaternion or the quaternion that represents no rotation is $\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^\top$. We have also chosen to follow the original Hamiltonian definition for quaternion algebra [23]. Accordingly, the rotation matrix based on a quaternion is given by

$$\mathbf{R}_v^b(\eta) = \begin{bmatrix} \eta_0^2 + \eta_x^2 - \eta_y^2 - \eta_z^2 & 2(\eta_x\eta_y + \eta_z\eta_0) & 2(\eta_x\eta_z - \eta_y\eta_0) \\ 2(\eta_x\eta_y - \eta_z\eta_0) & \eta_0^2 - \eta_x^2 + \eta_y^2 - \eta_z^2 & 2(\eta_y\eta_z + \eta_x\eta_0) \\ 2(\eta_x\eta_z + \eta_y\eta_0) & 2(\eta_y\eta_z - \eta_x\eta_0) & \eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2 \end{bmatrix}. \tag{2.11}$$

A quaternion must be unit-norm, $|\eta| = \eta_0^2 + |\boldsymbol{\eta}|^2 = \eta_0^2 + \eta_x^2 + \eta_y^2 + \eta_z^2 = 1$ to represent a rotation. Successive rotations are easily expressed as a single quaternion through either quaternion multiplication or the composition operator

$$\eta'' = \eta\eta' = \eta' \otimes \eta = \begin{bmatrix} \eta_0'\eta_0 - \boldsymbol{\eta}' \cdot \boldsymbol{\eta} \\ \eta_0'\boldsymbol{\eta} + \eta_0\boldsymbol{\eta}' - \boldsymbol{\eta}' \times \boldsymbol{\eta} \end{bmatrix}, \tag{2.12}$$

with $\otimes$ denoting the more common notation of composition. The difference between quaternion multiplication and composition is simply the order of the operands. In Equation 2.7, the rotation sequence occurs right to left–$\mathbf{R}_v^{v1}$ is the first rotation, $\mathbf{R}_{v1}^{v2}$ is next, followed by $\mathbf{R}_{v2}^{b}$. As with most quaternion operations, multiplication is not commutative so composition was defined to swap the operands so that the order of rotations was also right to left.

In Equation 2.12, η'' represents the overall rotation given by η followed by η'. This can also be written as

$$\eta'' = \eta' \otimes \eta = \left[\eta^{\otimes}\right] \eta', \tag{2.13}$$

where

$$\left[\eta^{\otimes}\right] = \begin{bmatrix} \eta_0 & -\eta_x & -\eta_y & -\eta_z \\ \eta_x & \eta_0 & -\eta_z & \eta_y \\ \eta_y & \eta_z & \eta_0 & -\eta_x \\ \eta_z & -\eta_y & \eta_x & \eta_0 \end{bmatrix}. \tag{2.14}$$

In this form, it can be seen that if $|\eta| = 1$, then $\left[\eta^{\otimes}\right]$ is unitary and $\left[\eta^{\otimes}\right]^{\top} \left[\eta^{\otimes}\right] = \mathbf{I}$. This is useful in cases where η' is an unknown quaternion and must be solved for:

$$\begin{aligned} \eta'' &= \left[\eta^{\otimes}\right] \eta', \\ \eta' &= \left[\eta^{\otimes}\right]^{\top} \eta''. \end{aligned} \tag{2.15}$$

Quaternions have the advantage of being singularity free and are more computationally efficient; however, they are not without their challenges. Perhaps the greatest difficulty in working with them is a lack of simple physical intuition on the rotation a given quaternion represents. Another challenge is the unit-norm constraint. Since a quaternion that represents a rotation must be unit-norm, the operations of addition and subtraction are not meaningful. Lastly, any rotation can be represented by two quaternions, η and $-\eta$. This non-uniqueness is referred to as double coverage. Double coverage does not pose an issue in the strict mathematical operations of quaternion algebra, but does need to be accounted for in applications such as quaternion feedback attitude control. The typical way of dealing with double coverage is to simply keep the scalar element positive. If the scalar element becomes negative, negating all four elements of the quaternion maintains the same rotation, while protecting against double coverage.

2.5 Wahba's Problem

One method that is used in this work to determine the rotation from one coordinate frame to another is the solution to Wahba's problem. Wahba's problem, first posed in 1965 by Grace Wahba, seeks to find the optimal rotation (in a least squares sense) between a set of vector observations expressed in a reference coordinate frame and those same observations expressed in a body-fixed coordinate frame. The reference frame vectors are typically available from some sort of model such as the World Magnetic Model [24] or database such as a star map. They are what would be expected to be observed by on-board sensors if the body frame was aligned with the reference frame. The body frame vectors are typically those observed by on-board sensors. A solution to Wahba's problem is given in Appendix B.

It is convenient to express the solution, $\mathbf{R}_{\text{opt}}$, as a quaternion using Equations A.10-A.14. Throughout this work, computing the solution to Wahba's problem and converting the solution to a quaternion is represented by

$$\eta = \text{wahba}(\{\mathbf{r}\}, \{\mathbf{b}\}, \mathbf{a}), \tag{2.16}$$

where $\{\mathbf{r}\}$ is the set vectors expressed in the reference frame, $\{\mathbf{b}\}$ is the set of vectors expressed in the body frame and $\mathbf{a}$ is a vector of weights.

CHAPTER 3. DEVELOPMENT TOOLS

Several development tools were crucial to the progress of this project. This chapter describes these tools. Section 3.1 describes a quaternion visualizer that provided a way to easily view the different attitude representations and the conversions between them. Section 3.2 goes over the simulators used to validate the algorithms before they were implemented in hardware. Section 3.3 then describes the hardware platforms used. Finally, Section 3.4 discussed the motion capture system used to gauge algorithm performance.

3.1 Quaternion Visualizer

One of the challenges in working with quaternions is visualizing what a given quaternion represents. Each individual Euler angle has independent, physical meaning. For example, in the 3-2-1 rotation sequence, ψ has independent meaning; it is the amount of rotation about the inertial z-axis. With quaternions, this is not the case. The meaning of each parameter by itself is difficult to discern. As a group, their meaning is somewhat easier to see for simple cases, but in general it is a challenge. To assist in this, a Graphical User Interface (GUI) was developed and is shown in Figure 3.1.

The original intent of the GUI was to associate a quaternion with a given set of Euler angles and display the rotation in graphical form. Other functions include showing the effects of quaternion operations, such as composition, showing attitude error and playing back data files from the autopilot and motion capture system.

3.1.1 Quaternion Input

The primary input for the GUI is an estimated attitude quaternion. It can be input directly or through conversion from Euler angles or hover Euler angles. All three representations are displayed simultaneously and a change to any one of them updates the others. The methods

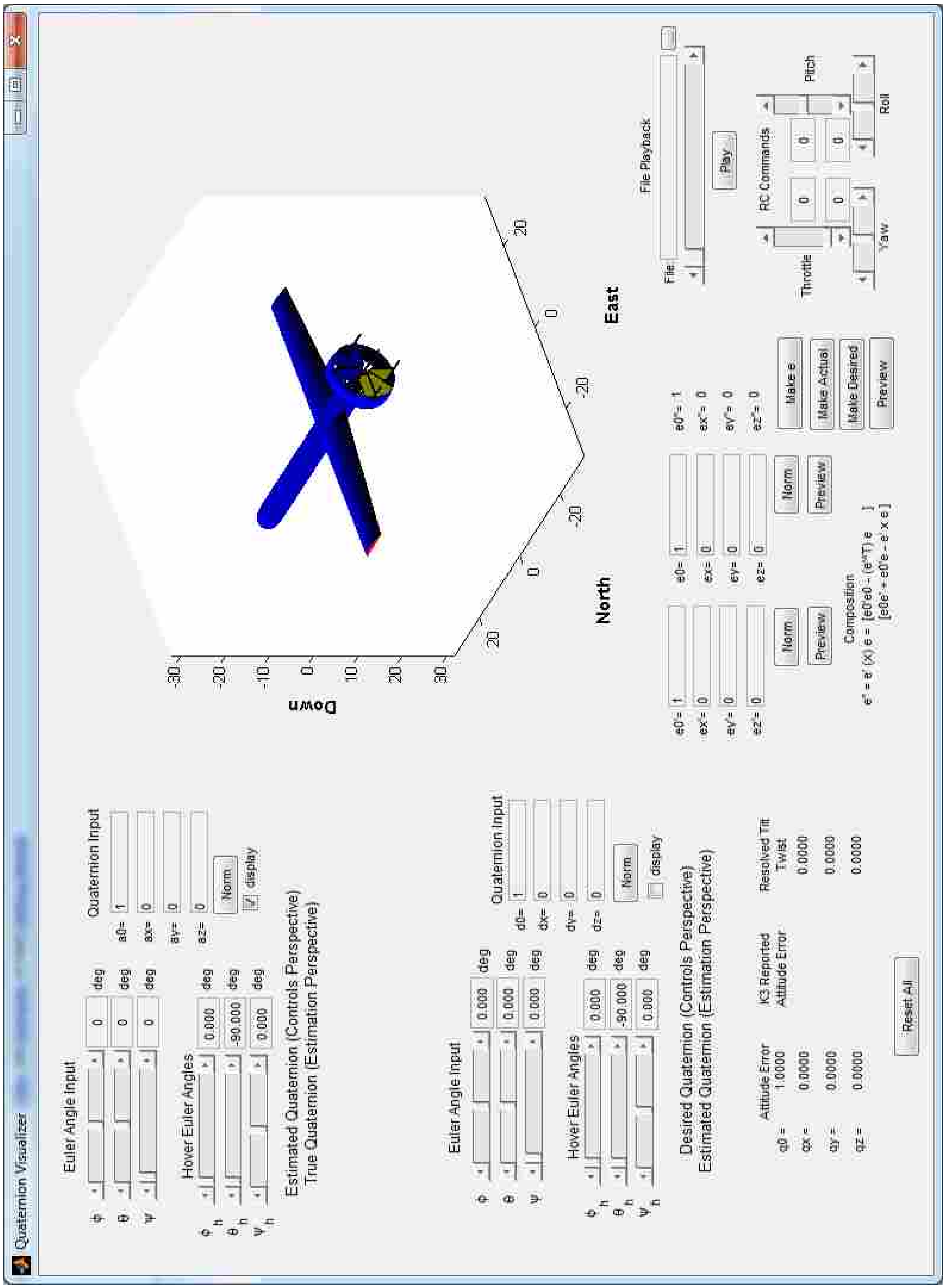


Figure 3.1: Quaternion visualizer GUI

for converting between representations is detailed in Appendix A. In general, the quaternion displayed is constrained to be unit-norm, the exception being when it is input directly. When input directly, the four elements are normalized to update the other representations and determine the outputs, but the inputs themselves are not updated with normalized values since this would make inputting them very cumbersome. After all the elements are input, a button is available to display the unit-norm quaternion.

The secondary input is a desired or true attitude quaternion. This similarly has level flight and hover flight Euler angle inputs or can be directly entered and functions similarly to the actual quaternion input. It is possible to input two additional quaternions, e and e′. These quaternions have no specific definition and are used to allow the user to see the effects composing quaternions.

3.1.2 Remote Control Inputs

To assist with the initial gain tuning of the tailsitter autopilot inner control loops, it was convenient to be able to generate a desired attitude quaternion via a remote control (RC) transmitter. This made it possible for an individual to send commands to the airplane without the use of a ground control station (GCS). The transmitter aileron, elevator and rudder control stick positions are mapped to a desired ϕ_h, θ_h, and ψ_h which are then converted to a desired quaternion by Equation A.6.

To verify the mixing of the control sticks, sliders were used to simulate the RC sticks. The resulting quaternion is automatically transferred to the desired quaternion input making it easy to verify the mixing worked as intended.

3.1.3 File Playback

The GUI was also capable of taking a file that contained a time sequence of two quaternions and animating the graphical representations of those quaternions in approximate real time. Optionally, data for the RC transmitter sticks could also be provided. In that case, the RC sliders are set accordingly.

3.1.4 Output

The visualizer's primary output is a tailsitter display of either the estimated, desired, or composed ($e' \otimes e$) quaternion. All three quaternions can be displayed simultaneously or individually. Other outputs include attitude error and the composed quaternion $e'' = e' \otimes e$.

3.2 Simulators

Two simulators were used to validate solutions to development issues. The primary simulator used was a 6-DOF tailsitter simulator developed during phase I of the tailsitter project [25]. The other was a more generic aircraft simulator used to compare estimation methods. The tailsitter simulator is Simulink-based and models the tailsitter dynamics as well as the autopilot guidance and control algorithms. It is possible to run this simulator in a hardware-in-loop (HIL) mode in which the simulated sensor output is generated and sent to an actual autopilot and the autopilot commands are fed back to the simulator.

3.2.1 Estimation Simulator

To test estimation algorithms a simpler UAV simulator based on [22] was used. Since the goal of these algorithms was to demonstrate various methods of incorporating a magnetometer sensor into an attitude estimate, no autopilot guidance and control functionality was incorporated. Details of each component that was incorporated is now given.

Simulator Input

The inputs to the simulator are the four primary flight controls: aileron, elevator, throttle and rudder.

Vehicle Dynamics

The translational equations of motion are

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{p}_d \end{bmatrix} = \mathbf{R}_v^b(\eta) \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{3.1}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}, \tag{3.2}$$

where f_x, f_y and f_z are the external forces applied respectively in the x, y and z body-frame axes. The rotational equations of motion are

$$\begin{bmatrix} \dot{\eta}_0 \\ \dot{\eta}_x \\ \dot{\eta}_y \\ \dot{\eta}_z \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} \begin{bmatrix} \eta_0 \\ \eta_x \\ \eta_y \\ \eta_z \end{bmatrix}, \tag{3.3}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6 \left(p^2 - r^2\right) \\ \Gamma_7 pq - \Gamma_1 qr \end{bmatrix} + \begin{bmatrix} \Gamma_3 l + \Gamma_4 n \\ \frac{1}{J_y} \\ \Gamma_4 l + \Gamma_8 n \end{bmatrix}, \tag{3.4}$$

where

$$\Gamma \triangleq J_x J_z - J_{xz}^2, \tag{3.5}$$

$$\Gamma_1 \triangleq \frac{J_{xz}(J_x - J_y + J_z)}{\Gamma}, \tag{3.6}$$

$$\Gamma_2 \triangleq \frac{J_z(J_z - J_y) + J_{xz}^2}{\Gamma}, \tag{3.7}$$

$$\Gamma_3 \triangleq \frac{J_z}{\Gamma}, \tag{3.8}$$

$$\Gamma_4 \triangleq \frac{J_{xz}}{\Gamma}, \tag{3.9}$$

$$\Gamma_5 \triangleq \frac{J_z - J_x}{\Gamma}, \tag{3.10}$$

$$\Gamma_6 \triangleq \frac{J_{xz}}{J_y}, \tag{3.11}$$

$$\Gamma_7 \triangleq \frac{J_x(J_x - J_y) + J_{xz}^2}{\Gamma}, \tag{3.12}$$

$$\Gamma_8 \triangleq \frac{J_x}{\Gamma}. \tag{3.13}$$

Sensor Models

Three sensors were simulated: a 3-axis rate-gyro, a 3-axis accelerometer and a 3-axis magnetometer. The 3-axis rate-gyro is simulated by adding zero-mean, Gaussian noise to the true angular velocity states provided by the simulator. The simulated rate-gyro sensor output is

$$\boldsymbol{\omega}^* = \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \sigma_{gyro}. \tag{3.14}$$

This assumes that biases in the rate-gyro measurement are removed. A more accurate model would contain a bias term that drifts; however, in the research performed, any method to account for this would be common to all algorithms and do nothing to distinguish one from another.

Acceleration sensed by an aircraft can be expressed mathematically as

$$\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \frac{d}{dt}\mathbf{v}^b + \boldsymbol{\omega}^b \times \mathbf{v}^b - \mathbf{R}_v^b \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}, \tag{3.15}$$

where $\frac{d}{dt}\mathbf{v}^b$ is the translational acceleration of the vehicle, $\boldsymbol{\omega}^b \times \mathbf{v}^b$ is the Coriolis acceleration and $\mathbf{R}_v^b \begin{bmatrix} 0 & 0 & g \end{bmatrix}^\top$ is gravity. In component form this is

$$\begin{aligned} a_x &= \dot{u} + qw - rv + 2g(\eta_0\eta_y - \eta_x\eta_z), \\ a_y &= \dot{v} + ru - pw - 2g(\eta_0\eta_x + \eta_y\eta_z), \\ a_z &= \dot{w} + pv - qu - g(\eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2). \end{aligned} \tag{3.16}$$

In the simulation, since external forces are calculated as part of the vehicle dynamics, it is convenient to not calculate the translational accelerations and instead substitute in Equation 3.2. Doing so simplifies Equation 3.16 to

$$\begin{aligned} a_x &= \frac{f_x}{m} + 2g(\eta_0\eta_y - \eta_x\eta_z), \\ a_y &= \frac{f_y}{m} - 2g(\eta_0\eta_x + \eta_y\eta_z), \\ a_z &= \frac{f_z}{m} - g(\eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2). \end{aligned} \tag{3.17}$$

Ideally, when using the accelerometer to determine vehicle tilt, the measurement would contain only the acceleration due to gravity. On an actual autopilot, GPS velocity measurements can be rotated to the body frame and used in conjunction with rate-gyro measurements to subtract Coriolis acceleration. Since GPS measurements were not simulated, the simulated accelerometer sensor output was corrected by subtracting the Coriolis acceleration. Adding zero-mean, Gaussian noise makes the simulated output to be

$$\mathbf{a}^* = \begin{bmatrix} \frac{f_x}{m} + 2g(\eta_0\eta_y - \eta_x\eta_z) - (qw - rv) \\ \frac{f_y}{m} - 2g(\eta_0\eta_x + \eta_y\eta_z) - (ru - pw) \\ \frac{f_z}{m} - g(\eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2) - (pv - qu) \end{bmatrix} + \sigma_{accel}. \tag{3.18}$$

On an actual autopilot, the corrected accelerometer measurements are

$$\mathbf{a}^* = \begin{bmatrix} a_x^* - (qw - rv) \\ a_y^* - (ru - pw) \\ a_z^* - (pv - qu) \end{bmatrix}. \tag{3.19}$$

In Provo, Utah, the earth's magnetic field is $\mathbf{b}^v \approx \begin{bmatrix} 21.030 & 4.348 & 47.304 \end{bmatrix}^\top$ μT. The magnetometer output is simulated by rotating this vector into the body frame using the true attitude quaternion and adding zero-mean, Gaussian noise:

$$\mathbf{b}^* = \mathbf{R}_v^b \mathbf{b}^v + \sigma_{mag}. \tag{3.20}$$

Simulator Output

The main simulator output is a 3-D plot of the vehicle and its flight path that gives an overall view of what the aircraft is doing. A variety of other plots are available to view sensor output and aircraft attitude. The simulator is also capable of recording data, similar to what would be available from an autopilot.

3.3 Hardware Platforms

This section briefly describes the hardware platforms used in testing.

3.3.1 Electric V-Bat

The MLB Company provided BYU with an electric powered, .78 scale model of the V-Bat, called the eV-Bat or electric V-Bat. The mature control algorithms will be implemented and tested on this aircraft.

3.3.2 BYU Scale V-Bat

For initial testing, an approximate .66 scale model of the eV-Bat was built. The goal for this aircraft was to allow initial validation of control algorithms without risk of damaging the eV-Bat. It is called the "Y-Bat" to differentiate between it and the eV-Bat. Both models are shown in Figure 3.2. Their dimensions are given in Table 3.1.

Table 3.1: Dimensions of the eV-Bat and Y-Bat.

	eV-Bat	Y-Bat
Total Airframe Length (m)	1.43	1.30
Wingspan (m)	1.93	1.56
Root Chord (m)	0.362	0.295
Tip Chord (m)	0.216	0.175
Fuselage Diameter (m)	0.152	0.127
Duct Outer Diameter (m)	0.483	0.406
Duct Inner Diameter (m)	0.381	0.336
Mass (kg)	5.47	3.98

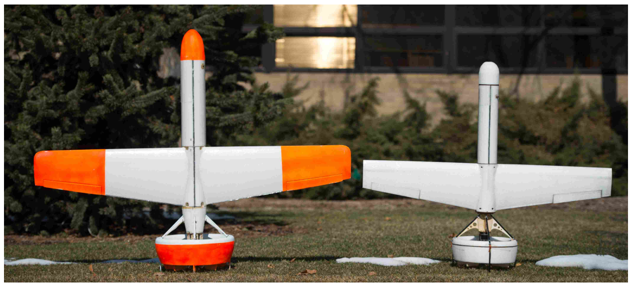


Figure 3.2: eV-Bat and Y-Bat models.

3.3.3 Kestrel Autopilot

The autopilot chosen for use in the V-Bat is the commercially available Lockheed-Martin Procerus Technologies Kestrel v3.0 autopilot, shown in Figure 3.3. The primary reason for choosing this platform was the maturity of the autopilot firmware and available ground control station (GCS). Although the firmware is mature, it currently only supports fixed-wing and multi-rotor aircraft. It was anticipated significant modifications would be needed to the firmware to support a tailsitter type aircraft. These modifications are addressed in future sections. The following section addresses some minor hardware modifications that were needed.

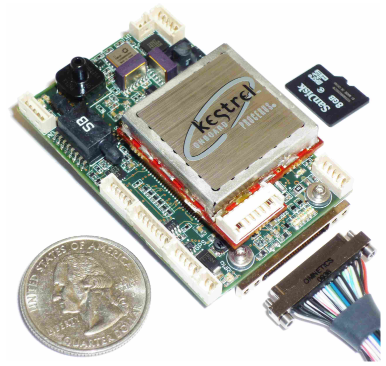


Figure 3.3: Kestrel v3 autopilot. Photo courtesy of Lockheed-Martin Procerus Technologies.

Servo Output

Basic flight of the V-Bat requires a total of ten servo outputs–one for each of the eight control vanes, one for the wing ailerons and one for throttle. One initial issue discovered was that of the eleven available servo outputs, three of them were limited to outputting a 3.3 V pulse width modulation (PWM) signal. A PWM level-shift board, shown in Figure 3.4, was built to increase the amplitude of these three outputs to the 5 V expected by most servos. This board also doubled as a power distribution board to all of the servos.

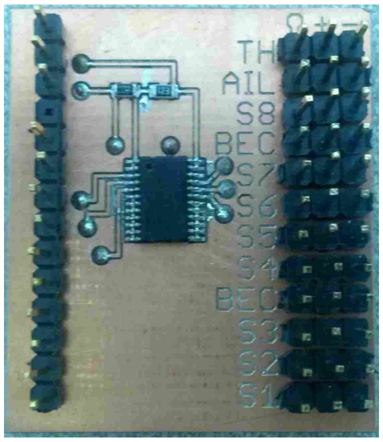


Figure 3.4: PWM level shift board. In addition to boosting the 3.3 V PWM servo signals to 5 V, this board also doubles as a servo power distribution board.

GPS Antenna Considerations

The standard GPS patch antenna, shown on the receiver board in Figure 3.5, works well when it is parallel to the ground, but not well when it is perpendicular to the ground. To overcome this, an omnidirectional antenna was used. It was mounted at 45° to ensure maximum reception in both level and hover flight modes.

3.4 Motion Capture

To provide truth data which estimates of the attitude could be compared to, a Motion Analysis motion capture system with eight cameras was used. The system produced a quaternion referenced to its own XYZ coordinate system. This quaternion was rotated by 180° about its inertial x-axis to transform it into a quaternion referenced to the north-east-down (NED) coordinate system that was aligned approximately with the NED system used by the autopilot.

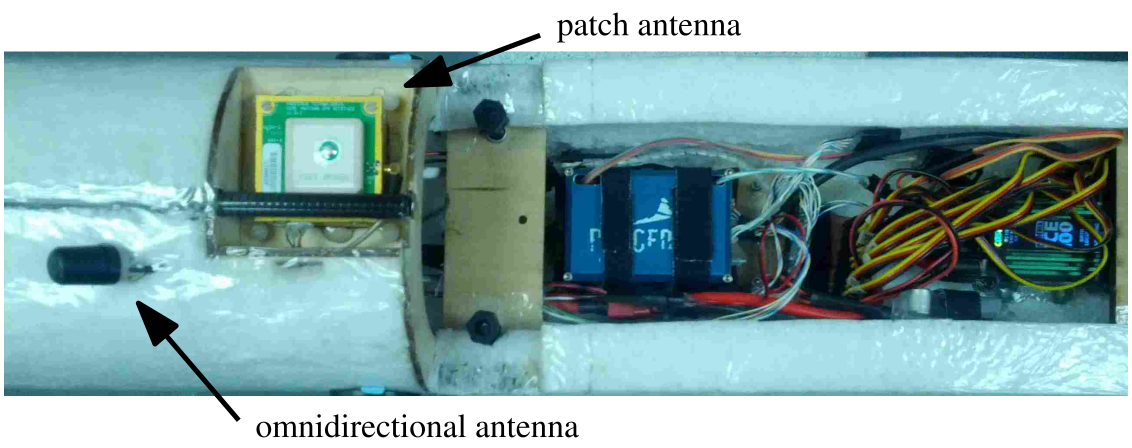


Figure 3.5: Kestrel autopilot installation. The omnidirectional GPS antenna is shown on the far left with the GPS receiver board and patch antenna just to its right.

3.4.1 Coordinate System Alignment

The Kestrel NED coordinate system is referenced to true north using magnetometer measurements. The XYZ coordinates of the motion capture system have no absolute reference to true north. During calibration of the motion capture system, efforts were made to align the XYZ coordinate system with true north. To account for the possibility of misalignment in this effort, the vehicle was kept perfectly still for the first five seconds of each test where it was assumed that the estimated and true quaternions were equal. Differences between the two were assumed to be a constant misalignment bias. To calculate this bias, it was desired to average several data-points to minimize any noise that was present; however, averaging a quaternion or rotation matrix is not well defined. Simply averaging the quaternion from several data points does not guarantee a good bias correction term. To determine the bias the problem is cast as Wahba's problem.

To cast finding the bias as Wahba's problem, the vectors used are the axes of the tailsitter, expressed in the vehicle frame. They are conveniently obtained by converting each quaternion to a rotation matrix using Equation 2.11. The true quaternions, η, provided by the motion capture data provides the body-fixed vectors $\{\mathbf{b}\}$ in equation B.2. The estimated quaternions, $\hat{\eta}$, recorded by the autopilot are the reference vectors $\{\mathbf{r}\}$. Each rotation matrix provides three vectors, so the total number of vectors used to solve Wahba's problem is the number of data points times three. For a single data point the vectors are

$$\left\{\mathbf{b}\right\} \triangleq \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3\end{bmatrix} = \mathbf{R}_v^b(\eta)^\top,$$
$$\left\{\mathbf{r}\right\} \triangleq \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3\end{bmatrix} = \mathbf{R}_v^b(\hat{\eta})^\top.$$

Using Equation 2.16, the bias correction is then

$$\eta_{bc} = \text{wahba}(\{\mathbf{r}\},\{\mathbf{b}\},\mathbf{a}), \tag{3.21}$$

with the vectors being uniformly weighted. The bias was then removed by

$$\hat{\eta}' = \eta_{bc} \otimes \hat{\eta}. \tag{3.22}$$

3.4.2 Data Synchronization

One challenge in comparing the data recorded by the autopilot and that provided by the motion capture system was time synchronization. A more expensive instrumentation data system could provide a common time reference for all recorded measurements; however, this was not possible. Getting the data synchronized correctly is crucial, since misalignments of one tenth of a second easily produce apparent heading errors of 40° when the heading is changing quickly. To assist with synchronizing the data, the airplane was pitched 90° immediately following the five-second pause to take bias-correction data. This introduced an easily identifiable feature in the data that would allow the data to be synchronized during post-processing. A disadvantage of this method is that it can hide any delay in the estimated data.

CHAPTER 4. HEADING ESTIMATION COMPARISON

A tailsitter aircraft naturally requires an attitude estimator that is functional at a wide range of attitudes. In addition to choosing an attitude representation that is free of singularities, sensor measurements must also be singularity free. The typical way of interpreting a magnetometer measurement uses Euler angles to determine heading [22]. By itself, this method breaks down when the tailsitter is in hover because of gimbal lock.

This chapter examines different methods for incorporating a 3-axis magnetometer measurement into an attitude estimate that are functional throughout a full range of vehicle attitudes and is organized as follows: Section 4.1 briefly reviews the Extended Kalman Filter (EKF) and discusses a number of methods to incorporate a magnetometer measurement into it. Section 4.2 discusses a simple Multiplicative Extended Kalman Filter (MEKF) to overcome EKF limitations with respect to quaternions. Sections 4.3 and 4.4 present the simulation and hardware testing that has occurred and preliminary results.

4.1 Extended Kalman Filter

The EKF has become a relatively standard tool in attitude estimation since it performs well, is well-documented and relatively straight-forward to implement. Two strategies are examined for incorporating the magnetometer measurement into the EKF. Using a magnetometer to measure only heading does not use the vertical component of the magnetic vector. The first strategy seeks to take advantage of the additional information and uses the full magnetic vector in conjunction with the accelerometers to determine attitude. The second strategy uses the magnetometer to only determine heading. Section 4.1.1 gives an overview of how the EKF works and how it is implemented. For the first strategy, Section 4.1.2 presents two methods that use the full magnetic vector to update the EKF estimate. Section 4.1.3 then presents three methods that use the second strategy and use the magnetometer only to update the heading.

4.1.1 EKF Overview

The EKF implemented is known as a continuous-discrete Kalman filter because it continuously propagates the estimated states forward in time and then at discrete points updates the estimate with sensor measurements. Since we are interested in estimating vehicle attitude, the four elements of the quaternion η are used for the state vector $\mathbf{x}$. The states are propagated forward in time using the quaternion time derivative [26] by

$$\mathbf{x} \triangleq \eta, \tag{4.1}$$

$$\mathbf{u} \triangleq \omega^* = \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{4.2}$$

$$\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u}) = \frac{1}{2}\begin{bmatrix} 0 \\ \omega^* \end{bmatrix} \otimes \eta, \tag{4.3}$$

where p, q and r are the body angular velocities as obtained from the rate gyros. It is assumed that the gyro biases have been estimated off-line. When the states are propagated forward in time, the estimate covariance must also be updated. The differential equation governing the estimate covariance is

$$\dot{\mathbf{P}} = \mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^\top + \mathbf{B}\mathbf{R}_g\mathbf{B}^\top + \mathbf{Q}, \tag{4.4}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the linearized state transition and input matrices, defined to be

$$\mathbf{A} \triangleq \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}} = \frac{1}{2}\begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix}, \tag{4.5}$$

$$\mathbf{B} \triangleq \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}} = \frac{1}{2}\begin{bmatrix} -\eta_x & -\eta_y & -\eta_z \\ \eta_0 & -\eta_z & \eta_y \\ \eta_z & \eta_0 & -\eta_x \\ -\eta_y & \eta_x & \eta_0 \end{bmatrix}. \tag{4.6}$$

$\mathbf{R}_g$ is the rate-gyro covariance matrix and $\mathbf{Q}$ is the process noise matrix. Euler integration can be used to actually implement equations 4.1-4.4 by

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + f(\hat{\mathbf{x}}^-,\mathbf{u})T_s, \tag{4.7}$$

$$\mathbf{P}^+ = \mathbf{P}^- + \left(\mathbf{A}\mathbf{P}^- + \mathbf{P}^-\mathbf{A}^\top + \mathbf{B}\mathbf{R}_g\mathbf{B}^\top + \mathbf{Q}\right)T_s, \tag{4.8}$$

where T_s is the update interval of the filter. Care should be taken since numerical integration techniques typically do not enforce the unit-norm requirement of the quaternion. As a result $\hat{\eta}$ should be normalized after integration takes place to eliminate numerical errors. Also, to avoid issues surrounding double coverage, the scalar component should be kept positive.

It is easy to see that if the estimate is only propagated forward in time, $\hat{\mathbf{x}}$ will drift because of integration and rate-gyro measurement errors. To correct this, additional sensor measurements are incorporated at arbitrary intervals by

$$\mathbf{L}_i = \mathbf{P}^-\mathbf{C}_i^\top(\mathbf{C}_i\mathbf{P}^-\mathbf{C}_i^\top + \mathbf{R}_i)^{-1}, \tag{4.9}$$

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_i(y_i^* - \mathbf{C}_i\hat{\mathbf{x}}^-), \tag{4.10}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_i\mathbf{C}_i)\mathbf{P}^-, \tag{4.11}$$

where $\mathbf{L}_i$ is the Kalman gain of sensor i, $\mathbf{C}_i$ is the linearized measurement output matrix defined as

$$\mathbf{C}_i \triangleq \frac{\partial h(\mathbf{x})_i}{\partial \mathbf{x}}, \tag{4.12}$$

$\mathbf{R}_i$ is the sensor's covariance matrix and y_i^* is the measurement. The steps to implement an EKF are summarized in Algorithm 1.

Since there are i sensors available to make measurement updates, one of the first decisions that must be made is how the measurement updates will be incorporated. Incorporating sensor

Algorithm 1 Quaternion based four-state EKF
Input: Filter update period T_s, measured body-rates p, q, and r, process covariance matrix $\mathbf{Q}$ and measurement covariance matrix $\mathbf{R}$

```
 1: Initialize $\hat{\mathbf{x}} = \eta_{init}, \mathbf{P} = \mathbf{P}_{init}$
 2:
 3: For each filter update period
 4: $\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + f(\hat{\mathbf{x}}^-, \mathbf{u})T_s$
 5: $\mathbf{A} = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}$
 6: $\mathbf{B} = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}$
 7: $\mathbf{P}^+ = \mathbf{P}^- + T_s\left(\mathbf{A}\mathbf{P}^- + \mathbf{P}^-\mathbf{A}^\top + \mathbf{B}\mathbf{R}_g\mathbf{B}^\top + \mathbf{Q}\right)$
 8: $\hat{\mathbf{x}} = \frac{\hat{\mathbf{x}}}{||\hat{\mathbf{x}}||}$                              ▷ Normalize quaternion
 9: if $\hat{\mathbf{x}}(1) < 0$ then, $\hat{\mathbf{x}} = -\hat{\mathbf{x}}$                    ▷ Check for double coverage
10:
11: if measurement $y_i$ is received from sensor $i$ then
12:     $\mathbf{C}_i = \frac{\partial h(\mathbf{x})_i}{\partial \mathbf{x}}$
13:     $\mathbf{L}_i = \mathbf{P}^-\mathbf{C}_i^\top(\mathbf{C}_i\mathbf{P}^-\mathbf{C}_i^\top + \mathbf{R}_i)^{-1}$
14:     $\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_i(y_i^* - \mathbf{C}_i\hat{\mathbf{x}}^-)$
15:     $\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_i\mathbf{C}_i)\mathbf{P}^-$
16:     $\hat{\mathbf{x}} = \frac{\hat{\mathbf{x}}}{||\hat{\mathbf{x}}||}$
17:     if $\hat{\mathbf{x}}(1) < 0$ then, $\hat{\mathbf{x}} = -\hat{\mathbf{x}}$
18: end if
```

measurements individually gives flexibility in how often each individual measurement update occurs and what triggers it to happen (at a fixed rate or by an interrupt, for example). Accuracy may also be gained since that measurement can be incorporated as soon as it is received. Combining data from multiple sensors into a single measurement update allows more complex updates to occur, but comes at the expense of having to delay incorporating one measurement while waiting for data from the other sensors. One way to mitigate this is to sample the sensors much faster than the frequency of the measurement update.

This decision fits well with the two strategies for incorporating magnetometer measurements. When using the full magnetometer measurement, it is easier to combine it with an accelerometer measurement to produce an “attitude” measurement that is used to update the estimate. This is demonstrated in the next section. Conversely, if using the magnetometer to only determine heading, it is easier to have the magnetometer and accelerometer perform their own individual measurement updates. This is shown in Section 4.1.3.

4.1.2 Full Attitude Measurement Update

Two methods are presented in this section that combine the accelerometer and magnetometer measurements using the solution to Wahba's problem (Appendix B). The solution is then used as an overall attitude measurement which is then used to update the estimate.

Wahba Method 1

The first method that implements the solution to Wahba's problem uses the corrected accelerometer and magnetometer measurements for the body-fixed vectors $\{\mathbf{b}\}$. The reference vectors $\{\mathbf{r}\}$ are $\mathbf{g}^v = \begin{bmatrix} 0 & 0 & -9.81 \end{bmatrix}^\top$ m/s^2 and $\mathbf{b}^v = \begin{bmatrix} 21.053 & 4.520 & 47.689 \end{bmatrix}^\top$ μT, although the reference magnetic field would obviously change depending on the location in the world. The World Magnetic Model [24] can be implemented to provide this. A heavier weight was given to the magnetic vector due to the possibility that the accelerometer measurement contained accelerations other than gravity, but it is not clear how significant this was in computing the solution since only two vectors were used. The attitude measurement is given by

$$\eta^* = \text{wahba}\left(\left\{\mathbf{g}^v \quad \mathbf{b}^v\right\}, \left\{\mathbf{a}^* \quad \mathbf{b}^*\right\}, \begin{bmatrix} 0.25 & 0.75 \end{bmatrix}\right). \tag{4.13}$$

To further protect the estimate from measurements that are affected by accelerations other than gravity, confidence in the measurement was reduced when the norm of the accelerometer components was not equal to gravity [26]. This is done by modifying the measurement covariance matrix $\mathbf{R}_w$ to

$$\mathbf{R}'_w = \mathbf{R}_w \left(1 + k \left|1 - \frac{|\mathbf{a}^*|}{g}\right|\right), \tag{4.14}$$

where k is a tunable gain. One disadvantage of this is that the entire measurement is penalized when $|\mathbf{a}^*| \neq g$, even though the magnetic vector is not affected. Noting that the measurement output matrix $\mathbf{C}_w = \mathbf{I}$, the estimate is updated by

$$\mathbf{L}_w = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}'_w)^{-1}, \tag{4.15}$$

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_w(\eta^* - \hat{\mathbf{x}}^-), \tag{4.16}$$

and the covariance is updated by

$$\mathbf{P}^{+} = (\mathbf{I} - \mathbf{L}_w)\mathbf{P}^{-}. \tag{4.17}$$

Wahba Method 2

One of the issues with Wahba Method 1 is that the entire measurement was penalized when the magnitude of the accelerometer measurement was not equal to gravity. To penalize only the accelerometer it is necessary to add additional vectors when determining the solution of Wahba's problem. The inertial axes and current estimate provide this. This does not provide an absolute reference but serves to constrain the third degree of freedom. The reference and body-frame vectors are then

$$\{\mathbf{r}\} = \left\{\mathbf{g}^v \quad \mathbf{b}^v \quad \mathbf{I}_{3\times 3}\right\},$$
$$\{\mathbf{b}\} = \left\{\mathbf{a}^* \quad \mathbf{b}^* \quad \mathbf{R}_v^b\right\},$$

with the matrices being decomposed into column vectors. Instead of inflating the measurement covariance matrix, the accelerometer is penalized by reducing its weight when computing the solution. The penalty is

$$z \triangleq \frac{1}{1 + k\left|1 - \frac{|\mathbf{a}^*|}{g}\right|}, \tag{4.18}$$

and is bounded between zero and one. Since the sum of the weights must be one, they were divided three ways: the magnetometer was given a fixed weight of 1/2, the accelerometer was weighted at $z/2$ and each of the vectors from the current estimate received a weight of $(1-z)/6$ or $\mathbf{z} = \begin{bmatrix}0.5 & z/2 & (1-z)/6 & (1-z)/6 & (1-z)/6\end{bmatrix}$. The attitude measurement is then

$$\eta^* = \text{wahba}\left(\{\mathbf{r}\}, \{\mathbf{b}\}, \mathbf{z}\right). \tag{4.19}$$

Similar to the Wahba Method 1, the measurement update is

$$\mathbf{L}_w = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}_w)^{-1}, \tag{4.20}$$

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_w(\eta^* - \hat{\mathbf{x}}^-), \tag{4.21}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_w)\mathbf{P}^-. \tag{4.22}$$

4.1.3 Heading Only Magnetometer Measurement Update

This section focuses on how to incorporate the magnetometer measurement separately from the accelerometer. The accelerometer measurement update is discussed first and is common to all algorithms in this section. After the accelerometer measurement update is presented, several ways to incorporate the magnetometer are given:

1. A "pure Euler angle" method that converts the estimated attitude quaternion to either level flight or hover flight Euler angles to determine heading error and perform what we call the compass Kalman update.

2. A "mixed quaternion Euler angle" method which calculates heading error using quaternions but then uses the compass Kalman update from above.

3. A "pure quaternion" method that uses quaternions to determine heading error and perform the Kalman update.

Accelerometer Measurement Update

This measurement update uses the corrected accelerometer output from Equation 3.18 and is based on [26]. The normalized gravity vector is expressed in the body frame using the current estimate by

$$\mathbf{g}^b = \mathbf{R}_v^b(\hat{\eta}) \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^\top, \tag{4.23}$$

and is compared to the normalized gravity vector measurement, which assuming no translational acceleration, is the normalized accelerometer measurement

$$\mathbf{g}^* = \frac{\mathbf{a}^*}{|\mathbf{a}^*|}. \tag{4.24}$$

The error axis and angle of rotation are derived from the cross product and dot product of these vectors

$$\mathbf{e}_a = \frac{\mathbf{g}^* \times \mathbf{g}^b}{|\mathbf{g}^* \times \mathbf{g}^b|}, \tag{4.25}$$

$$\Theta_a = \cos^{-1}\left(\mathbf{g}^* \cdot \mathbf{g}^b\right). \tag{4.26}$$

The error expressed as a quaternion is

$$\tilde{\eta} = \begin{bmatrix} \cos\frac{\Theta_a}{2} & \mathbf{e}_a \sin\frac{\Theta_a}{2} \end{bmatrix}^\top. \tag{4.27}$$

To get an attitude measurement the error is combined with the current estimate by

$$\eta^* = \tilde{\eta} \otimes \hat{\mathbf{x}}. \tag{4.28}$$

To account for the possibility that the measured gravity vector has been distorted by accelerations other than gravity, the accelerometer measurement is penalized when the norm of accelerometer measurement is not equal to gravity. Similar to the Wahba Method 1, the inflated measurement covariance matrix is

$$\mathbf{R}'_a = \mathbf{R}_a \left(1 + k\left|1 - \frac{|\mathbf{a}^*|}{g}\right|\right). \tag{4.29}$$

Noting that the measurement output matrix $\mathbf{C}_a$ is identity, the estimate is updated by

$$\mathbf{L}_a = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}'_a)^{-1}, \tag{4.30}$$

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_a(\eta^* - \hat{\mathbf{x}}^-), \tag{4.31}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_a)\mathbf{P}^-. \tag{4.32}$$

Pure Euler Angle Magnetometer Update

To cover a full range of attitudes two sets of Euler angles are required to avoid gimbal lock. In level flight the pure Euler angle method is the traditional and most intuitive way to use a magnetometer–as a digital compass that gives an absolute heading measurement. Determining heading and updating the estimate using level flight Euler angles is discussed first, followed by using hover Euler angles.

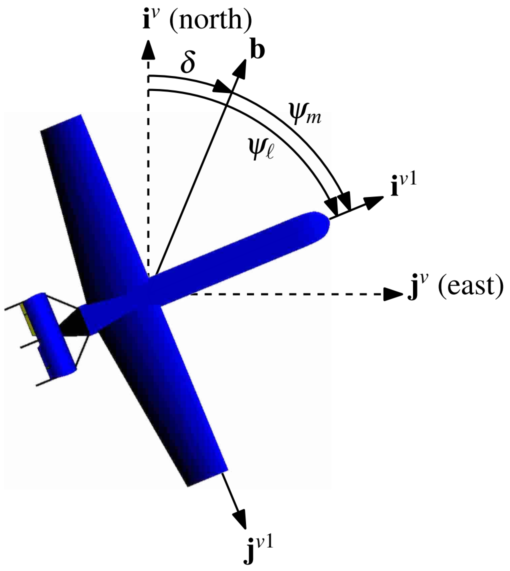


Figure 4.1: Magnetic vector in the vehicle-1 frame. Not visible is the possibly substantial vertical component.

The magnetic vector measured by the magnetometer, $\mathbf{b}^*$, is expressed in terms of the body frame. To get heading the magnetic vector must be expressed in terms of the vehicle-1 frame so that heading is the only rotation between magnetic north and $\mathbf{b}^*$ as shown in Figure 4.1. Since the quaternion represents attitude as a single rotation it is not possible to use it directly to express $\mathbf{b}^*$ in the vehicle-1 frame. To do this Equation A.2 is used to convert the estimated quaternion to Euler angles which can then be used to make the rotation [22]:

$$\mathbf{b}^{v_1} = \mathbf{R}_{v2}^{v1}(\theta_\ell)\mathbf{R}_{b}^{v2}(\phi_\ell)\mathbf{b}^*,$$

$$= \begin{bmatrix} c_{\theta_\ell} & s_{\phi_\ell} s_{\theta_\ell} & c_\phi s_{\theta_\ell} \\ 0 & c_{\phi_\ell} & -s_{\phi_\ell} \\ -s_{\theta_\ell} & s_{\phi_\ell} c_{\theta_\ell} & c_\phi c_{\theta_\ell} \end{bmatrix} \mathbf{b}^*. \tag{4.33}$$

Magnetic heading is then given by

$$\psi_m = -\operatorname{atan2}\left[b_y^{v1}, b_x^{v1}\right]. \tag{4.34}$$

The negative sign is necessary because the coordinate frame was rotated, not the magnetic vector. If the magnetic vector expressed in the vehicle frame is known, declination can be determined by

$$\delta = \operatorname{atan2}\left[b_y^{v}, b_x^{v}\right]. \tag{4.35}$$

True heading and heading error are then

$$\psi = \psi_m + \delta, \tag{4.36}$$

$$\tilde{\psi} = \psi - \psi_\ell, \tag{4.37}$$

where ψ_ℓ is the estimated heading obtained when the current estimate is converted to Euler angles.

The compass Kalman update that updates the attitude estimate based on $\tilde{\psi}$ requires a formal definition of heading, traditionally Equation 2.4, or the angle between the vehicle and body x-axes as a result of a rotation about the vehicle z-axis. Conveniently, the elements of $\mathbf{R}_v^b$ can be used to express this definition mathematically since the rows of $\mathbf{R}_v^b$ represent the body axes expressed in the vehicle frame

$$h_{b,\ell}(\hat{\mathbf{x}}) = \operatorname{atan2}\left[\mathbf{R}_v^b(1,2),\ \mathbf{R}_v^b(1,1)\right]. \tag{4.38}$$

The output matrix is formed by taking the Jacobian of equation 4.38 giving

$$\frac{\partial h_{b,\ell}}{\partial \eta_0} = \frac{2\eta_z \mathbf{R}_v^b(1,1)}{\mathbf{R}_v^b(1,1)^2 + \mathbf{R}_v^b(1,2)^2},$$

$$\frac{\partial h_{b,\ell}}{\partial \eta_x} = \frac{2\eta_y \mathbf{R}_v^b(1,1)}{\mathbf{R}_v^b(1,1)^2 + \mathbf{R}_v^b(1,2)^2},$$
$$\frac{\partial h_{b,\ell}}{\partial \eta_y} = \frac{2\eta_x \mathbf{R}_v^b(1,1) + 4\eta_y \mathbf{R}_v^b(1,2)}{\mathbf{R}_v^b(1,1)^2 + \mathbf{R}_v^b(1,2)^2},$$
$$\frac{\partial h_{b,\ell}}{\partial \eta_z} = \frac{2\eta_0 \mathbf{R}_v^b(1,1) + 4\eta_z \mathbf{R}_v^b(1,2)}{\mathbf{R}_v^b(1,1)^2 + \mathbf{R}_v^b(1,2)^2},$$
$$\mathbf{C}_b = \frac{\partial h_b}{\partial \eta} = \begin{bmatrix} \frac{\partial h_b}{\partial \eta_0} & \frac{\partial h_b}{\partial \eta_x} & \frac{\partial h_b}{\partial \eta_y} & \frac{\partial h_b}{\partial \eta_z} \end{bmatrix}. \quad (4.39)$$

The compass Kalman update is then performed by

$$\mathbf{L}_b = \mathbf{P}^{-}\mathbf{C}_b^\top \left(\mathbf{C}_b \mathbf{P}^{-}\mathbf{C}_b^\top + \mathbf{R}_b\right)^{-1}, \quad (4.40)$$

$$\hat{\mathbf{x}}^{+} = \hat{\mathbf{x}}^{-} + \mathbf{L}_b \tilde{\psi}, \quad (4.41)$$

$$\mathbf{P}^{+} = \left(\mathbf{I} - \mathbf{L}_b \mathbf{C}_b\right) \mathbf{P}^{-}. \quad (4.42)$$

As in other cases, after the update occurs the quaternion is normalized and corrected for double coverage. If not used in conjunction with hover Euler angles, this update has several problems when the elevation angle, θ_ℓ, is near 90°. First, it is impossible to determine ϕ_ℓ accurately for expressing the magnetic measurement in the vehicle-1 frame. Second, small deviations in elevation angle cause large swings in heading. For example, if $\theta_\ell = 90°$ with zero bank or heading, equation 4.38 gives the correct heading of zero. If elevation angle is increased to 91°, either actually or by measurement noise, the heading will swing to 180°. Additionally, when $\theta_\ell = 90°$, $\mathbf{R}_v^b(1,1) = \mathbf{R}_v^b(1,2) = 0$, which is indicative that using the body x-axis to define heading at that elevation angle is ambiguous. This also causes $\mathbf{C}_b$ and $\mathbf{L}_b$ to be zero, resulting in the filter always rejecting the measurement input.

It is not possible to define heading in a manner that is valid at all attitudes; a second definition is needed. When the tailsitter is in hover, heading is defined as the direction the body z-axis points as a result of a rotation about the vehicle z-axis. Similar to level flight, magnetic heading, ψ_m, is determined by expressing the magnetometer measurement in to the hover-1 frame by

$$\mathbf{b}^{h1} = \mathbf{R}_b^{h1}\mathbf{b}^{*}, \quad (4.43)$$

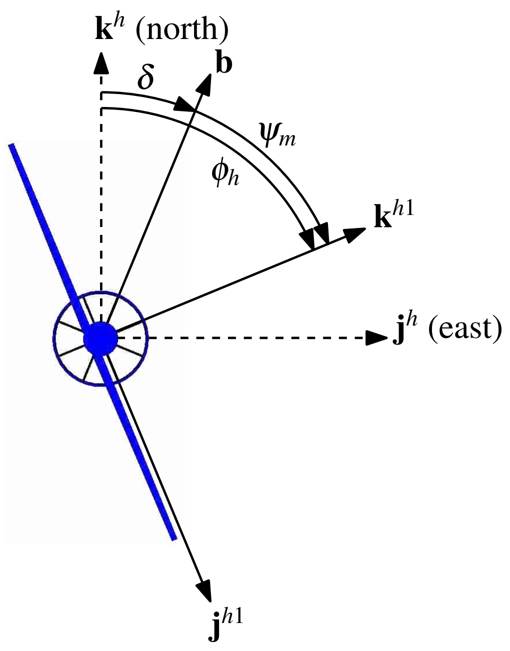


Figure 4.2: Magnetic vector in the hover-1 frame. As in the level case, the magnetic vector may have a substantial vertical component.

$$= \begin{bmatrix} c_{\theta_h} c_{\psi_h} & -c_{\theta_h} s_{\psi_h} & s_{\theta_h} \\ s_{\psi_h} & c_{\psi_h} & 0 \\ -s_{\theta_h} c_{\psi_h} & s_{\theta_h} s_{\psi_h} & c_{\theta_h} \end{bmatrix} \mathbf{b}^*, \tag{4.44}$$

$$\psi_m = -\operatorname{atan2}\left[b_y^{h1}, b_z^{h1}\right]. \tag{4.45}$$

True heading is calculated using Equation 4.36; however, since ϕ_h governs heading while hovering, the heading error given by 4.37 is instead

$$\tilde{\psi} = \psi - \phi_h. \tag{4.46}$$

Since heading is now defined using the z-axis, the nonlinear measurement output equation is

$$h_{b,h}(\mathbf{x}, \mathbf{u}) = \operatorname{atan2}\left[\mathbf{R}_v^b(3,2),\ \mathbf{R}_v^b(3,1)\right], \tag{4.47}$$

making the partial derivatives for the output matrix $\mathbf{C}_b$

$$\begin{aligned}
\frac{\partial h_{b,h}}{\partial \eta_0} &= \frac{-2\left(\eta_x \mathbf{R}_v^b(3,1) + \eta_y \mathbf{R}_v^b(3,2)\right)}{\mathbf{R}_v^b(3,1)^2 + \mathbf{R}_v^b(3,2)^2}, \\
\frac{\partial h_{b,h}}{\partial \eta_x} &= \frac{-2\left(\eta_0 \mathbf{R}_v^b(3,1) + \eta_z \mathbf{R}_v^b(3,2)\right)}{\mathbf{R}_v^b(3,1)^2 + \mathbf{R}_v^b(3,2)^2}, \\
\frac{\partial h_{b,h}}{\partial \eta_y} &= \frac{2\left(\eta_z \mathbf{R}_v^b(3,1) - \eta_0 \mathbf{R}_v^b(3,2)\right)}{\mathbf{R}_v^b(3,1)^2 + \mathbf{R}_v^b(3,2)^2}, \\
\frac{\partial h_{b,h}}{\partial \eta_z} &= \frac{2\left(\eta_y \mathbf{R}_v^b(3,1) - \eta_x \mathbf{R}_v^b(3,2)\right)}{\mathbf{R}_v^b(3,1)^2 + \mathbf{R}_v^b(3,2)^2}.
\end{aligned}$$

The compass Kalman update in Equations 4.39-4.42 is then used to update the estimate.

Similar to the level flight Euler angles, a singularity occurs when $\theta_h = 90°$. To avoid this the two representations are used together, switching between them at a given point. Chosen somewhat arbitrarily, when $\theta_\ell \leq 60°$, the level flight Euler angles are used to update the attitude, and above that the hover flight Euler angles are used.

Mixed Quaternion Euler Angle Measurement Update

It would be convenient if there were a way to determine heading error without having to convert to Euler angles and use a switch to avoid gimbal lock. Fortunately, there is; however, it requires that heading error, $\tilde{\psi}$, is calculated directly without actually using heading. Heading can be calculated separately with Equations A.2 and A.6 if desired for user display. This method was initially developed in [9] but is here adapted to work within an EKF framework and to slightly improve computational efficiency and ease of implementation.

Instead of expressing $\mathbf{b}^*$ in the vehicle-1 frame, $\mathbf{b}^*$ is expressed in the vehicle frame and projected onto the vehicle $i^v j^v$-plane, something that can be done directly with the attitude quaternion by

$$\mathbf{b}^{\hat{v}} = \frac{\mathbf{P_{xy}}[\mathbf{R}_v^b(\hat{\eta})]^\top \mathbf{b}^*}{|\mathbf{P_{xy}}[\mathbf{R}_v^b(\hat{\eta})]^\top \mathbf{b}^*|}, \tag{4.48}$$

$$\mathbf{P_{xy}} \triangleq \operatorname{diag}[1,1,0]. \tag{4.49}$$

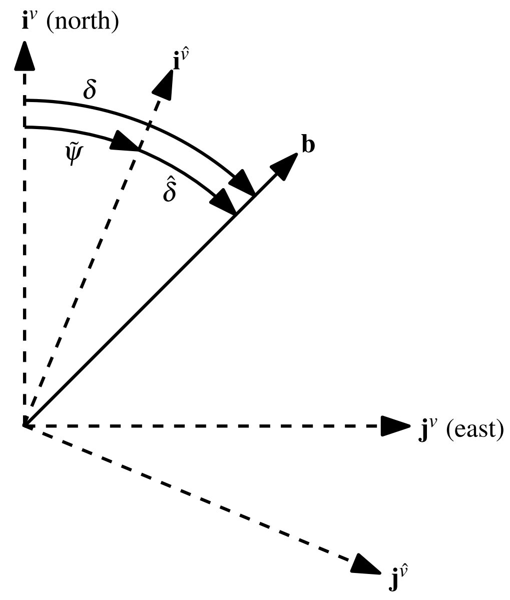


Figure 4.3: Heading error in the vehicle frame. Heading error is the difference between the actual declination at a given location and the estimated declination.

$\mathbf{P_{xy}}$ is a projection matrix that zeros the bottom row of $\mathbf{R}_v^b$, thus projecting $\mathbf{b}^*$ onto the horizontal plane. This can be implemented more efficiently by simply omitting the multiplication of that row.

Because the estimated quaternion is used, Equation 4.48 actually expresses the magnetometer measurement in the *estimated* vehicle frame, denoted by $\hat{v}$. In Figure 4.3, $\mathbf{i}^{\hat{v}}$ and $\mathbf{j}^{\hat{v}}$ are the estimated north and east axes. The estimated angle of declination is

$$\hat{\delta} = \text{atan2}\left[b_y^{\hat{v}}, b_x^{\hat{v}}\right]. \tag{4.50}$$

Heading error is simply the difference between the true and estimated values of declination or

$$\tilde{\psi} = \delta - \hat{\delta}. \tag{4.51}$$

This heading error is then used directly in equation 4.41 and has the advantages of being computationally faster and not requiring use of a switch to determine heading error. The compass Kalman update is then used to update the attitude estimate; however, a switch is still required to determine the output matrix $\mathbf{C}_b$ when calculating the Kalman gain. This is why this method is called the “mixed quaternion Euler angle method.”

One advantage of this algorithm over that given in [9] is that the algorithm can begin to correct the heading estimate before δ is known (such as when the autopilot is first powered on and reference to GPS and WMM data is not yet available) by assuming it is zero. Doing so will reference the heading to magnetic north instead of true north, but as soon as δ is known it can be added in to account for declination.

Pure Quaternion Magnetometer Update

To fully avoid using a switch a way must be developed to update the estimate without using Euler angles. This is possible if the heading error from Equation 4.51 is combined with the attitude estimate to form a quaternion measurement which is then used to update the estimate. Heading error expressed as a quaternion is

$$\tilde{\eta}_b = \begin{bmatrix} \cos\frac{\tilde{\psi}}{2} & 0 & 0 & \sin\frac{\tilde{\psi}}{2} \end{bmatrix}^\top . \tag{4.52}$$

To form an attitude measurement, this error is composed with the current estimate by

$$\eta^* = \hat{\eta}^- \otimes \tilde{\eta}_b, \tag{4.53}$$

where the order of composition is reversed because the heading error is expressed in the vehicle frame. In this case, the measurement output matrix $\mathbf{C}_b$ is identity, simplifying the Kalman update to

$$\mathbf{L}_b = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}_b)^{-1}, \tag{4.54}$$

$$\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- + \mathbf{L}_b(\eta^* - \hat{\mathbf{x}}^-), \tag{4.55}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L})\mathbf{P}^-. \tag{4.56}$$

The primary advantage of this method is that it works at all attitudes without a switch. Computationally, calculating heading error is faster, but the Kalman update is slower because of the matrix inversion required.

4.2 Multiplicative Extended Kalman Filter

The EKF in general has two conceptual problems when it comes to estimating a quaternion. The first is that the Kalman update does not account for the unit-norm constraint of the quaternion. This is illustrated in Equations 4.52-4.55 in that $\tilde{\eta}_b \neq \eta^* - \hat{\mathbf{x}}^-$. Another issue pointed out by [20] is that because of the quaternion unit-norm constraint, the 4×4 covariance matrix is redundant and provides no additional information over a 3×3 covariance matrix. The Multiplicative Extended Kalman Filter (MEKF) was designed to address these two issues. The MEKF implemented was developed in [26]; however the implementation presented here uses a magnetometer in lieu of a GPS.

A key concept to the MEKF is expressing error in the attitude estimate by

$$\tilde{\mathbf{a}} \triangleq \Theta \mathbf{e}, \tag{4.57}$$

where Θ and $\mathbf{e}$ are respectively the error angle and axis of rotation. Provided that $\mathbf{e}$ is unit-norm, all information about attitude error is recovered and expressed as an error quaternion by [20]

$$\delta\eta(\tilde{\mathbf{a}}) \triangleq \begin{bmatrix} \cos\frac{|\tilde{\mathbf{a}}|}{2} \\ \frac{\tilde{\mathbf{a}}}{|\tilde{\mathbf{a}}|}\sin\frac{|\tilde{\mathbf{a}}|}{2} \end{bmatrix}. \tag{4.58}$$

This works because $|\tilde{\mathbf{a}}| = \Theta$.

4.2.1 Time Update

The MEKF is called an indirect Kalman filter because the quaternion is not formally part of the estimator state vector. Instead, the attitude error, $\tilde{\mathbf{a}}$, is the estimator state vector, or

$$\mathbf{x} \triangleq \tilde{\mathbf{a}}, \tag{4.59}$$

$$\mathbf{u} \triangleq \omega^* = \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{4.60}$$

Similar to the standard EKF, the time update consists of propagating both the estimator state vector and covariance matrix forward in time. The differential equation governing the attitude error is

$$\dot{\tilde{\mathbf{a}}} = f(\mathbf{x},\mathbf{u}) = -\omega \times \tilde{\mathbf{a}}. \tag{4.61}$$

The linearized state transition matrix is then

$$\mathbf{A} \triangleq \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix}. \tag{4.62}$$

Assuming the biases in the rate gyros are removed, the noise on the rate gyros is zero mean and the attitude error is zero at the beginning of the time update, the expected value of the attitude error after the time update is zero. The assumption that the attitude error is zero at the beginning of the time update is important and is justified by "resetting" the attitude error to zero at the end of each measurement update. Because of this, the state vector itself does not require integration.

As is the case of the EKF, the differential equation governing the covariance matrix is

$$\dot{\mathbf{P}} = \mathbf{AP} + \mathbf{PA}^\top + \mathbf{Q}, \tag{4.63}$$

or implemented discretely using Euler integration is

$$\mathbf{P}^+ = \mathbf{P}^- + \left(\mathbf{AP}^- + \mathbf{P}^-\mathbf{A}^\top + \mathbf{Q}\right) T_s. \tag{4.64}$$

An estimate of the attitude quaternion is maintained separately outside of the state vector and is propagated forward in time separately by integrating the quaternion derivative. From Equation 4.3 this is

$$\hat{\boldsymbol{\eta}}^+ = \hat{\boldsymbol{\eta}}^- + \frac{T_s}{2}\begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix}\hat{\boldsymbol{\eta}}^-. \tag{4.65}$$

4.2.2 Accelerometer Measurement Update

The MEKF accelerometer measurement update is similar to that presented in Section 4.1.3. The error angle and axis of rotation from Equations 4.25 and 4.26 are used to calculate the attitude error vector by

$$\tilde{\mathbf{a}}_a^- = \Theta_a \mathbf{e}_a. \tag{4.66}$$

Confidence in the accelerometer measurement is reduced as before by inflating the accelerometer covariance matrix, $\mathbf{R}_a$, by

$$\mathbf{R}_a' = \mathbf{R}_a\left(1 + k\left|1 - \frac{|\mathbf{a}^*|}{g}\right|\right), \tag{4.67}$$

where k is a tunable gain.

The attitude error before the measurement is assumed to be zero and the measurement output matrix, $\mathbf{C}_a$, is identity, allowing the Kalman gain and the attitude error update equations to reduce to

$$\mathbf{L}_a = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}_a')^{-1}, \tag{4.68}$$

$$\tilde{\mathbf{a}}_a^+ = \mathbf{L}_a\tilde{\mathbf{a}}_a^-, \tag{4.69}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_a)\mathbf{P}^-. \tag{4.70}$$

The attitude error is then reset to zero by updating the quaternion using composition

$$\hat{\eta}^+ = \delta\eta(\tilde{\mathbf{a}}_a^+) \otimes \hat{\eta}^-. \tag{4.71}$$

4.2.3 Magnetometer Measurement Update

The method for calculating heading error from Equation 4.51 ties in well here. Since heading is always the rotation about the vehicle z-axis, the attitude error vector is

$$\tilde{\mathbf{a}}_b^- = \begin{bmatrix} 0 & 0 & \tilde{\psi} \end{bmatrix}^\top . \tag{4.72}$$

The Kalman update equations are similar to those used by the accelerometers:

$$\mathbf{L}_b = \mathbf{P}^-(\mathbf{P}^- + \mathbf{R}_b)^{-1}, \tag{4.73}$$

$$\tilde{\mathbf{a}}_b^+ = \mathbf{L}_b\tilde{\mathbf{a}}_b^-, \tag{4.74}$$

$$\mathbf{P}^+ = (\mathbf{I} - \mathbf{L}_b)\mathbf{P}^-. \tag{4.75}$$

Resetting the error to zero is accomplished by

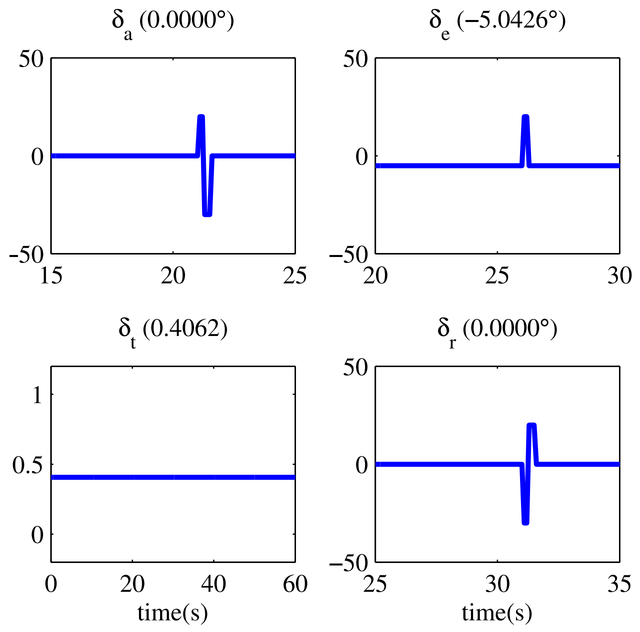
$$\hat{\eta}^+ = \hat{\eta}^- \otimes \delta\eta(\tilde{\mathbf{a}}_b^+). \tag{4.76}$$

The order of composition is reversed to account for the error being expressed in the vehicle frame.

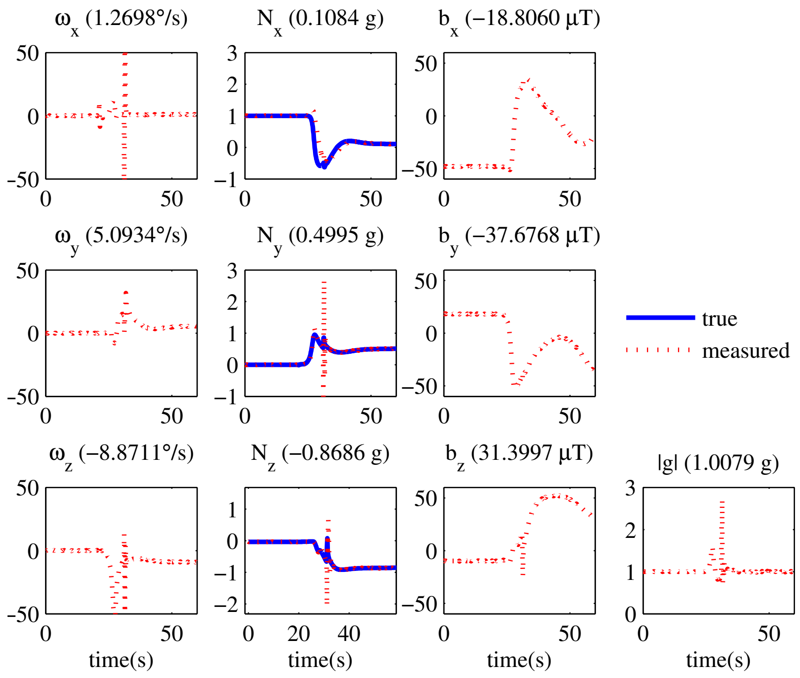
4.3 Simulation

The simulation results of each of these methods is now presented. During testing, the initial elevation angle of the tailsitter was approximately 90° and a random heading was chosen. For comparison and the results presented here, the initial heading was arbitrarily fixed at 250°. At the start of the simulation the vehicle was trimmed to allow time for the estimator to converge. Approximately 20 seconds into the simulated flight a benchmark maneuver consisting of an aileron doublet, an elevator impulse, and a rudder doublet was performed. Each was separated by five seconds as shown in Figure 4.4(a).

Since no autopilot control was implemented, the benchmark maneuver had the effect of tipping the aircraft over into a descending spiral. This exposes the estimator to a variety of attitudes as well as conditions that cause the accelerometers to measure more than just gravity. The resulting



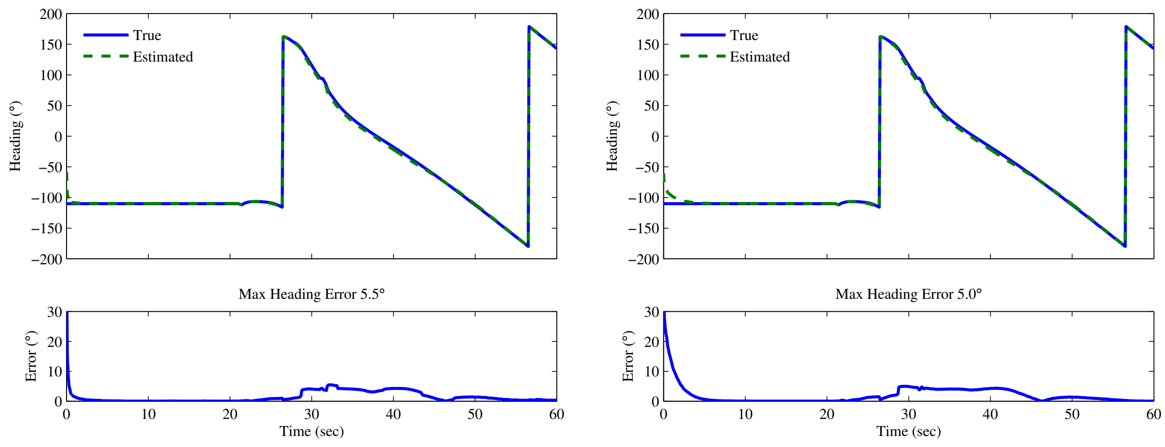
(a) Benchmark manuever. Left to right, top to bottom is the aileron, elevator, throttle and rudder inputs.



(b) Sensor output

Figure 4.4: Benchmark maneuver and sensor output of simulation. The sensor output, in columns from left to right, is the rate gyro, normalized accelerometer and magnetometer output. Magnitude of the measured gravity vector is also shown and is of primary interest since it shows how much the gravity measurement is contaminated with other accelerations.

sensor output is shown in Figure 4.4(b). From approximately 25 seconds to 40 seconds the vehicle experiences accelerations that distort the true gravity vector.



(a) Wahba Method 1 (b) Wahba Method 2

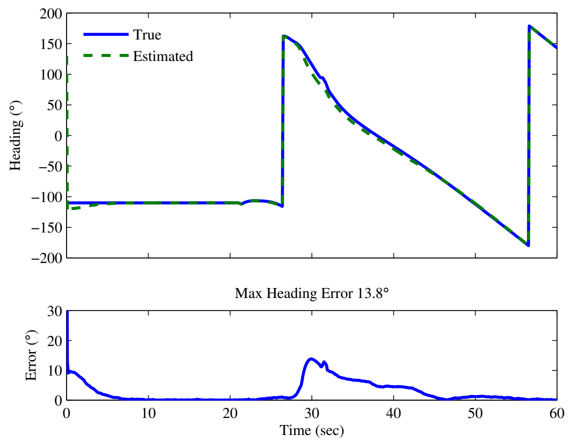
Figure 4.5: Wahba Method simulation results

The way an EKF is tuned has a significant effect on how the filter performs. For comparison, efforts were made to keep as many parameters common as possible between the different filters. It is also possible to tune the filter such that it becomes overconfident and always discards the measurement update. This occurs when the diagonals of the covariance matrix **P** become very small. To prevent this the filters were tuned so that the diagonals of **P** converged to values on the order of 1×10^{-2}.

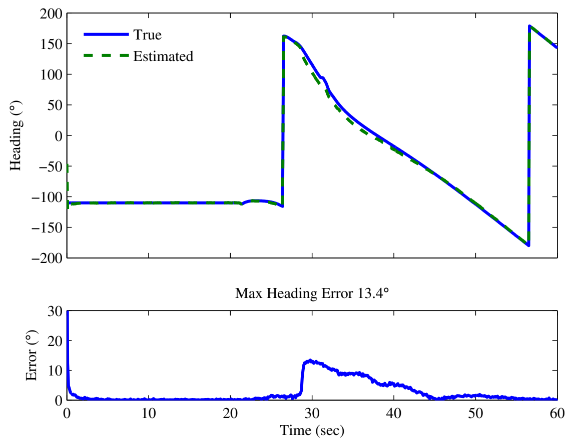
Figures 4.5-4.7 show the simulation results. In the heading portion of the plots, a switch from hover Euler angles to level Euler angles is evident at 26.5 s. At 56.5 s the heading wraps at -180°. The maximum heading error that occurred after 10 s into the simulation is also shown.

Wahba Method 1 and 2 both performed well. Incorporating the estimate in Wahba method 2 essentially acts like a low-pass filter and slowed the filter's convergence; however, it did not provide significant improvement during the high-g maneuver as anticipated.

The two methods that used Euler angles had the highest overall maximum error. The mixed Euler method has the advantage of having the computationally fastest measurement update of the EKFs tested, primarily because it requires only a single trigonometric function to compute heading error. The pure quaternion and MEKF approaches had the lowest maximum error.

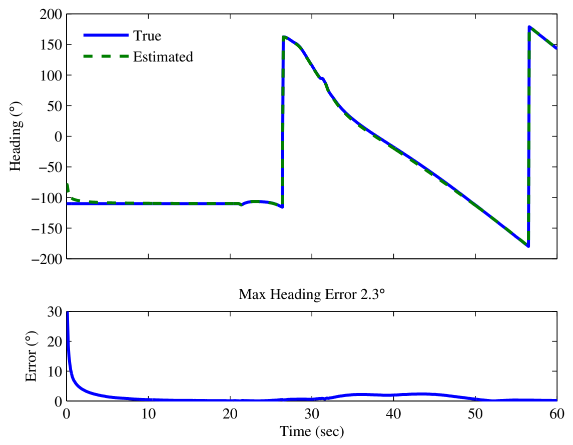


(a) Pure Euler Method

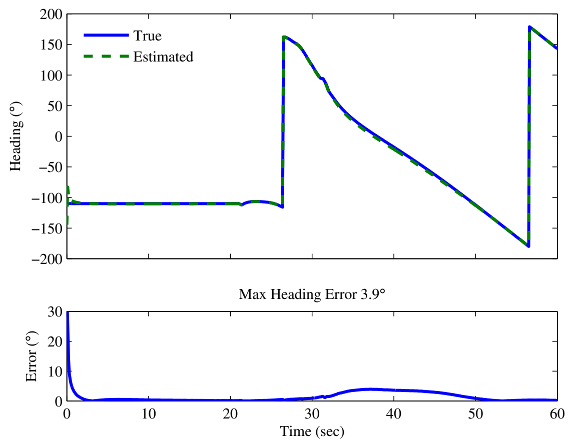


(b) Mixed Euler Method

Figure 4.6: Pure- and mixed-Euler simulation results



(a) Pure Quaternion Method



(b) MEKF

Figure 4.7: Pure quaternion and MEKF simulation results

4.4 Hardware Results

To test these algorithms in hardware, the source code of the Kestrel autopilot was modified to incorporate the pure Euler, mixed Euler, pure quaternion and MEKF algorithms. Because the first three of these are fundamentally similar to the estimator used by the autopilot (all based on an EKF), only the magnetometer update was modified. A software flag set via the ground control station during flight was used to pick which of the three algorithms was used.

The goal in testing was to compare the quaternion estimated by the autopilot to one measured by the motion capture system. Tests were conducted outdoors to allow access to the GPS and magnetometer data needed for attitude estimation. To ensure ample illumination of the motion capture targets, tests were conducted at night.

For the testing reported here, the tailsitter was hovered approximately three feet off the ground. An attitude control loop has been implemented and is capable of keeping the vehicle upright; however, for safety, the aircraft was tethered to an overhead cable.

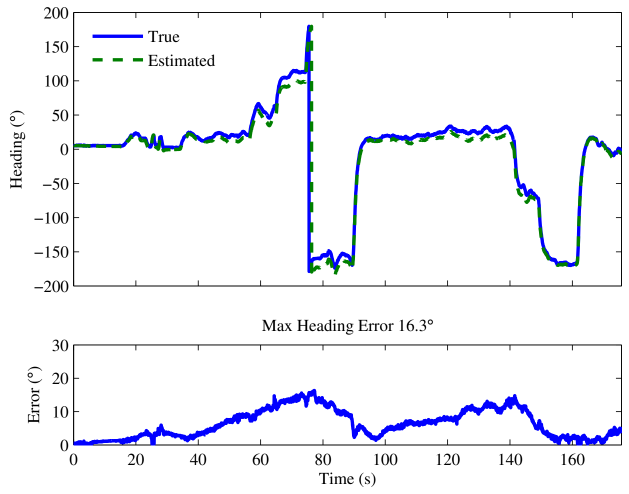


Figure 4.8: Flight using the pure Euler angle method to correct attitude. This is currently the default magnetometer update method.

Several test flights were conducted. In each one data was logged by the autopilot at 20 Hz and by the motion capture system at 100 Hz. Before data logging was started the aircraft was left stationary to allow the autopilot estimator to reach steady state. Once data recording was started the autopilot was allowed to remain stationary for several more seconds to provide data to determine and remove the bias mentioned in Section 3.4.1. The data feature mentioned in Section 3.4.2 was then introduced followed by advancing the throttle to begin the flight. During each test several heading commands were given to the autopilot via the ground control station.

In each plot, true and estimated heading as calculated by equation A.6 are given. Heading error is shown separately for emphasis. It is important to note that errors in the elevation and bank angle estimates can contribute to heading error.

Figure 4.8 shows data where heading was updated using the pure Euler method. A maximum of 16° of heading error was recorded for this flight. This is somewhat larger than expected

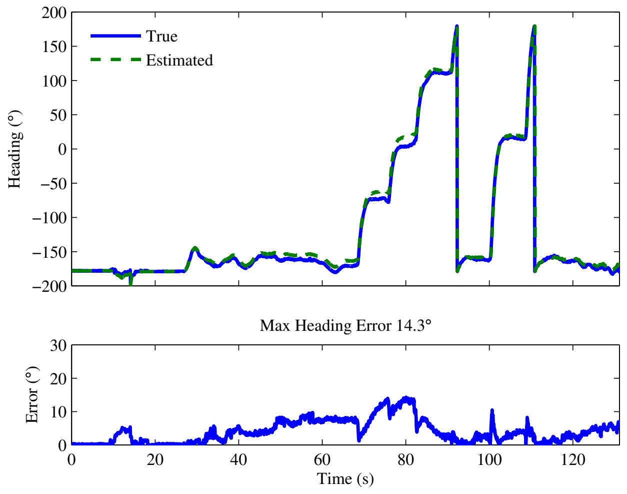


Figure 4.9: Flight using quaternion to calculate heading error and the Euler angle Kalman update. Numerically this appears to be identical to the pure Euler method but is faster.

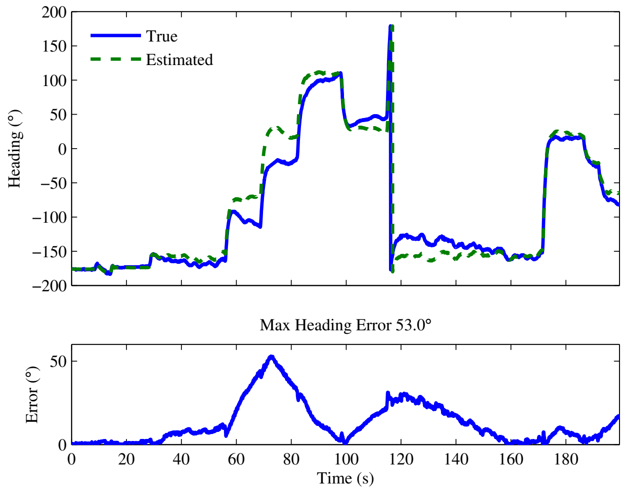


Figure 4.10: Flight using quaternion to calculate heading error and perform estimate update. The large error is mainly attributed to not re-tuning the filter.

and is initially attributed to electromagnetic interference from the battery cables. After this data was taken the battery cables were twisted and located as far from the magnetometer as possible, although this was already somewhat done for these tests.

Figure 4.9 shows data using the mixed quaternion Euler approach and with 14° max error, performed similarly to the previous method. Of the three EKF based methods tested, this was computationally the fastest.

The final set of data recorded was obtained using the quaternion measurement update. It was noted during simulation that the covariance matrix values that worked well for the above methods did not work well here. Good performance was obtained in simulation once these values were refined. Tuning and future testing should produce a much lower maximum error.

Because the MEKF is fundamentally different from the EKF it was necessary to implement a complete MEKF routine that ran along side of the EKF routine. Both routines are able to be run simultaneously; however, the filter was not yet tuned and ready for evaluation when the above testing took place.

4.5 Conclusion

This chapter has presented several estimation schemes that are functional at a full range of vehicle attitudes. Two strategies were employed. The first strategy sought to use the entire magnetic vector in conjunction with gravity. The second strategy used only the horizontal component of the magnetic vector to update vehicle heading. Simulation results for all algorithms have been given. Testing of a subset of these algorithms on an actual tailsitter aircraft was also performed and results were given.

CHAPTER 5. HOVER CONTROL

The overall autopilot architecture designed for the V-Bat is given in [25] and shown in Figure 5.1. The flight mode controller is made up of controllers for the different flight regimes

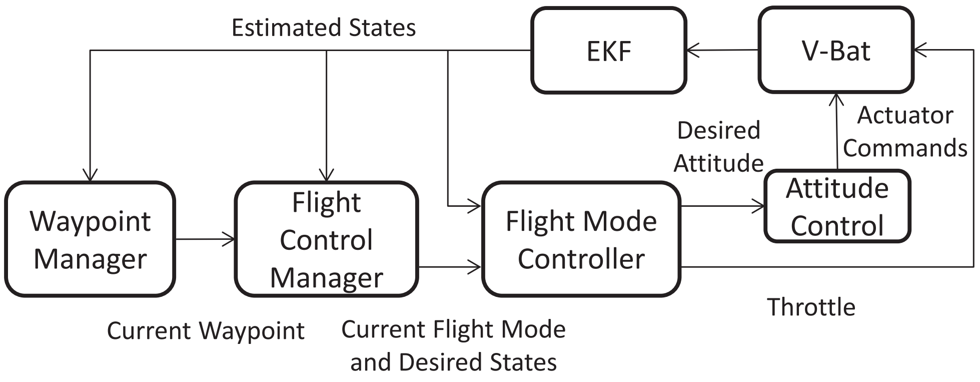


Figure 5.1: Planned V-Bat Control Architecture.

such as hover, level flight and the transitions. This chapter focuses on attitude control and the hover flight portion of the flight mode controller. Section 5.1 discusses the inner-loop attitude control and some of the issues discovered during implementation. Sections 5.2 and 5.3 briefly address position and altitude control. Section 5.4 briefly describes guidance modes needed for testing. The chapter concludes with testing results in Section 5.5.

5.1 Attitude Control

In the design of the control architecture, one goal was to have an inner-loop attitude control approach that was common to all flight regimes. This avoids having to switch between attitude control schemes in the middle of a transition and any associated discontinuities and transients introduced by them. Quaternion feedback attitude control is widely used in spacecraft and is a natural choice for the tailsitter because it works at all attitudes and is computationally efficient. Section 5.1.1 describes quaternion attitude control and an issue that was discovered during its implementation. Section 5.1.2 then details an alternative to quaternion feedback control called resolved tilt-twist angle control.

5.1.1 Quaternion Control

Quaternion attitude control exploits the fact that the four elements of a quaternion are related to an axis and angle of rotation. To do this an error quaternion is defined to be the rotation needed to get from the current estimated attitude, $\hat{\eta}$, to some desired attitude, η^d or

$$\eta^d = \tilde{\eta} \otimes \hat{\eta}. \tag{5.1}$$

From Equation 2.15, this rotation can be solved for by

$$\tilde{\eta} = \left[\hat{\eta}^{\otimes}\right]^{\top} \eta^d. \tag{5.2}$$

The vector component of $\tilde{\eta}$ defines the axis of rotation to get from the current attitude to the desired attitude, expressed in the body frame [9]. Because of this each component of the error can be used directly to drive a flight control surface: the x-component drives the roll effort, the y-component drives the pitch effort, and the z-component drives the yaw effort or

$$\delta_a = k_p\tilde{\eta}_x - k_d p + k_i \int \tilde{\eta}_x \, \mathrm{d}t, \tag{5.3}$$

$$\delta_e = k_p\tilde{\eta}_y - k_d q + k_i \int \tilde{\eta}_y \, \mathrm{d}t, \tag{5.4}$$

$$\delta_r = k_p\tilde{\eta}_z - k_d r + k_i \int \tilde{\eta}_z \, \mathrm{d}t, \tag{5.5}$$

where p, q, and r are the body angular rates given by rate-gyros and the control efforts δ_a, δ_e and δ_r are mapped to their respective control surfaces such that a positive control effort results in a positive moment about that axis.

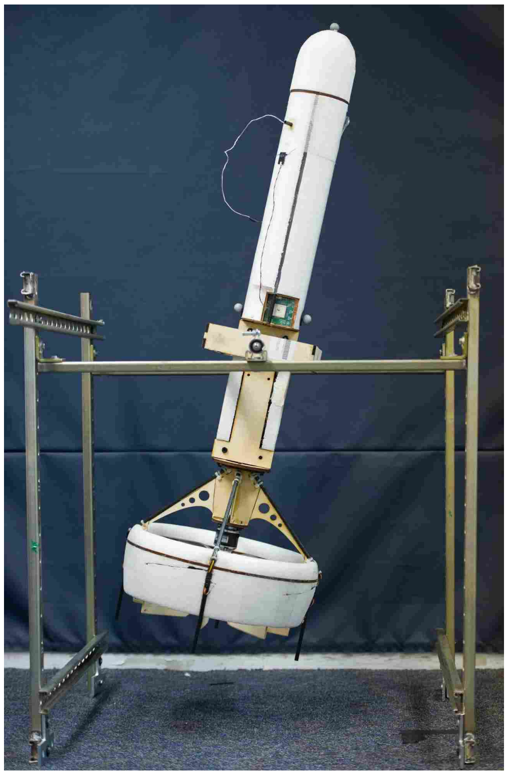


Figure 5.2: Y-Bat 2-axis gimbal.

Simulation of this indicated it would work well. Initial hardware testing of this took place with the Y-Bat placed in a 2-axis gimbal, shown in Figure 5.2. This gimbal allowed relatively free rotation in the body y- and z-axes while preventing rotation about the body x-axis.

Testing took place indoors where the magnetometer measurements were unreliable. Because of this and the constraints of the gimbal, the x-axis (roll) gains were set to zero. In spite of this, when the vehicle was close to hover orientation it would begin to wobble around the vertical position.

Investigation revealed that even though the x-axis gains were zero, the error in the body x-axis impacted the error in the other body axes. This is most easily shown in Figure 5.3. Given a desired attitude of $\eta^d = \begin{bmatrix} 0.7071 & 0 & 0.7071 & 0 \end{bmatrix}^\top$, which is simply the vehicle pitched to 90° ($\theta_h = 0°$) and belly facing north, and an estimated attitude pitched down slightly to $\theta_h = -10°$ and an arbitrary heading. The intuitive control command in all cases would be some pitch effort to correct the pitch error and some roll effort to correct the heading error. No yaw (z-axis) correction is needed.

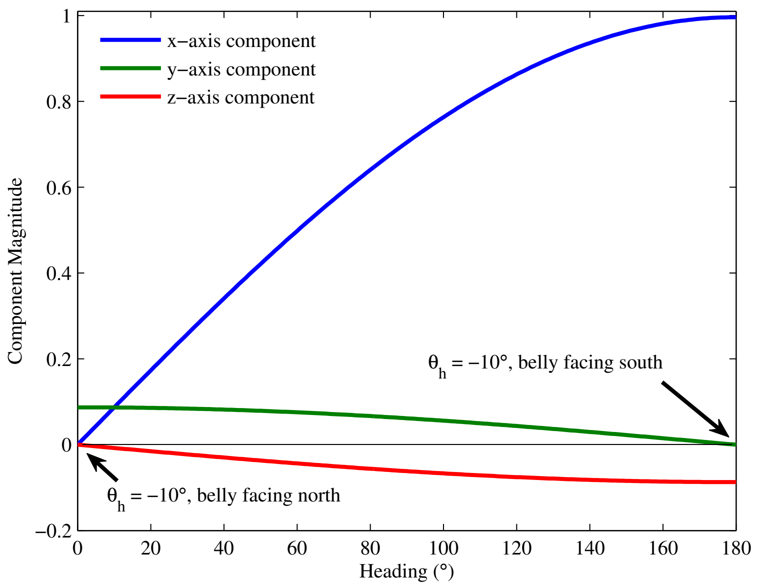


Figure 5.3: Quaternion error vector components given a desired attitude of the nose pointing up and belly facing north ($\theta_h = 0°$, $\phi_h = 0°$) and a estimated attitude of $\theta_h = -10°$ and heading (ϕ_h) ranging from 0°-180°. Since the desired heading is zero, heading and heading error have the same magnitude. Bank angle (ψ_h) in all cases is zero.

As expected in Figure 5.3, the x-axis component magnitude increases as heading error increases. When the current heading is zero the pitch error appears fully in the y-axis as expected; however, as the current heading (and thus heading error) is increased to 180° the error shifts to the body z-axis so no correction in pitch is made. This gradual shift was not anticipated. This phenomenon is the presumed cause of the wobble around the vertical position, particularly because motion in the x-axis was constrained, preventing that error from being driven to zero.

5.1.2 Resolved Tilt-Twist Angle Control

A literature search revealed that Matsumoto, et al [16] found this same phenomenon. They propose a solution called resolved tilt-twist angle control or simply RTT that decomposes the error into a tilt error and a twist error. The work in this section validates and builds upon their work by making several algorithmic improvements and incorporating the RTT control algorithm on the V-Bat. Deriving the required control effort makes heavy use of the fact that the columns of $\mathbf{R}_v^b$ are the vehicle axes expressed in the respective body frame and that the rows of $\mathbf{R}_v^b$ are the axes of the respective body frame expressed in the vehicle frame as explained in Section 2.1.

Derive Tilt Error

To start, the estimated and desired attitudes must be expressed as rotation matrices using equation 2.11. The attitude error is then defined as the rotation required to get from the estimated coordinate frame to the desired coordinate frame or

$$\mathbf{R}_v^b(\eta^d) = \tilde{\mathbf{R}}_v^b \mathbf{R}_v^b(\hat{\eta}). \tag{5.6}$$

Solving for $\tilde{\mathbf{R}}_v^b$ gives

$$\tilde{\mathbf{R}}_v^b = \mathbf{R}_v^b(\eta^d)\mathbf{R}_v^b(\hat{\eta})^\top = \begin{bmatrix} \tilde{r}_{11} & \tilde{r}_{12} & \tilde{r}_{13} \\ \tilde{r}_{21} & \tilde{r}_{22} & \tilde{r}_{23} \\ \tilde{r}_{31} & \tilde{r}_{32} & \tilde{r}_{33} \end{bmatrix}, \tag{5.7}$$

where $\tilde{r}_{mn}$ denotes the (m,n) element of the respective rotation matrix which in this case is $\tilde{\mathbf{R}}_v^b$. The tilt error consists of an error about the body y-axis and body z-axis. Two cases are considered to derive the error in these axes. The first case illustrates the y-axis (pitch) error and the second shows the z-axis (yaw) error.

If the desired attitude is pointing straight up with the belly facing north (ϕ_h, θ_h and ψ_h all 0°), and the estimated attitude was pitched 10° (ϕ_h, θ_h and ψ_h are 0°, 10° and 0°) then the three rotation matrices are

$$\mathbf{R}_v^b(\eta^d) = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{R}_v^b(\hat{\eta}) = \begin{bmatrix} -0.17 & 0 & -0.98 \\ 0 & 1 & 0 \\ 0.98 & 0 & -0.17 \end{bmatrix}, \quad \tilde{\mathbf{R}}_v^b = \begin{bmatrix} 0.98 & 0 & 0.17 \\ 0 & 1 & 0 \\ -0.17 & 0 & 0.98 \end{bmatrix}. \tag{5.8}$$

Since the columns of $\mathbf{R}_v^b(\eta^d)$ and $\mathbf{R}_v^b(\hat{\eta})$ represent the axes of the vehicle frame expressed in some other coordinate frame (namely, the desired body frame and estimated body frame), $\tilde{\mathbf{R}}_v^b$ also expresses the axes of the vehicle frame in another coordinate frame, which is here referred to as the error frame. Comparing the columns of $\tilde{\mathbf{R}}_v^b$ in Equation 5.8 with the frames shown in Figure 5.4, shows that this is indeed that case. It can also be seen that if the estimated frame is rotated by the error frame, the result is the desired frame.

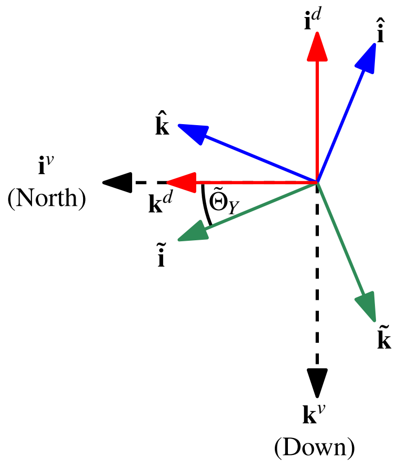


Figure 5.4: Tilt error about the body y-axis. For the desired attitude, ϕ_h, θ_h and ψ_h are all 0°, so that the nose is pointing up and the belly is facing north. For the estimated attitude ϕ_h, θ_h and ψ_h are 0°, 10°, and 0°.

Since the columns of $\tilde{\mathbf{R}}_v^b$ express the vehicle axes in the error frame, using the x-axis column (or the vehicle x-axis expressed in error frame) to determine the tilt error as originally done in [16] has an undesired side-effect. Heading error shows up in the error frame as a rotation about the error frame x-axis. This makes sense, because in hover, heading is changed by rotating the tailsitter about the hover frame x-axis; however, this changes the x-axis column of $\tilde{\mathbf{R}}_v^b$ and using it to define the tilt error would need to be accounted for as they do in the last step of their process. It is more convenient to instead use $\tilde{\mathbf{i}}$, the x-axis of the error frame expressed in the vehicle frame, or the top row in $\tilde{\mathbf{R}}_v^b$, to determine tilt error. Expressing the x-axis of the error frame in the vehicle frame has the property of being invariant with respect to heading error. This is because rotating a coordinate frame about an axis does not change that axis, and heading errors show up as rotations about the error x-axis. Given this, the y-axis error is given as

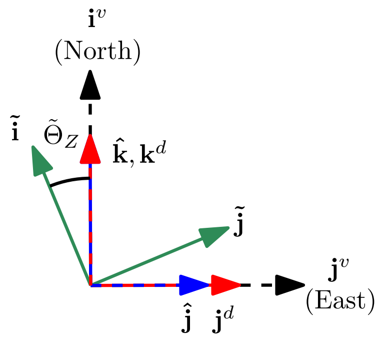
$$\tilde{\Theta}_Y = -\operatorname{atan2}\left[\tilde{r}_{13}, \tilde{r}_{11}\right]. \tag{5.9}$$

The negative sign accounts for the needed direction of rotation. From Figure 5.4 it can be seen that without the negative sign, Equation 5.9 would return a positive value; however, a negative rotation about the body y-axis is needed to correct the vehicle attitude.

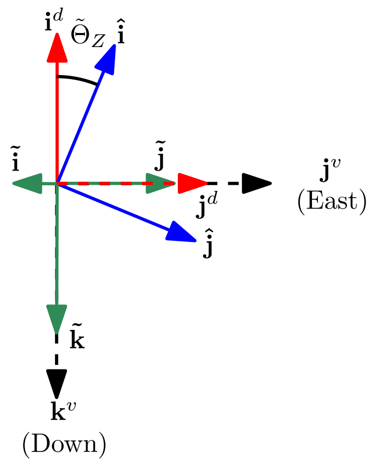
To derive the z-axis tilt error consider the same desired attitude of the tailsitter pointing straight up with the belly facing north and a new estimated attitude with a bank angle of 10° (ϕ_h, θ_h and ψ_h are 0°, 0° and 10°). In this case the rotation matrices are

$$\mathbf{R}_v^b(\eta^d) = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{R}_v^b(\hat{\eta}) = \begin{bmatrix} 0 & 0.17 & -0.98 \\ 0 & 0.98 & 0.17 \\ 1 & 0 & 0 \end{bmatrix}, \quad \tilde{\mathbf{R}}_v^b = \begin{bmatrix} 0.98 & -0.17 & 0 \\ 0.17 & 0.98 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5.10}$$

These coordinate frames are shown in Figure 5.5. The error frame is simply the vehicle frame



(a) View looking down



(b) View looking north

Figure 5.5: Tilt error about body z-axis

rotated about its z-axis. Rotating the estimated frame about its body z-axis results in the desired body frame. Again using the x-axis row of $\tilde{\mathbf{R}}_v^b$ to determine the angle of rotation gives

$$\tilde{\Theta}_Z = \operatorname{atan2}\left[\tilde{r}_{12}, \tilde{r}_{11}\right]. \tag{5.11}$$

Similar to the previous case, a negative rotation about the body z-axis is needed to correct the attitude for this case; however, unlike the previous case, the sign of Θ_Z is already accounted for in the angle, which can be seen in Figure 5.5(a) and Equation 5.10.

Derive Twist Error

The strategy for deriving the twist error is to find an intermediate coordinate frame with the same amount of twist error as the estimated body frame, but whose x-axis is aligned with the x-axis of the desired frame. The twist error is the angle of rotation about the x-axis of this intermediate frame required to align the remaining axes.

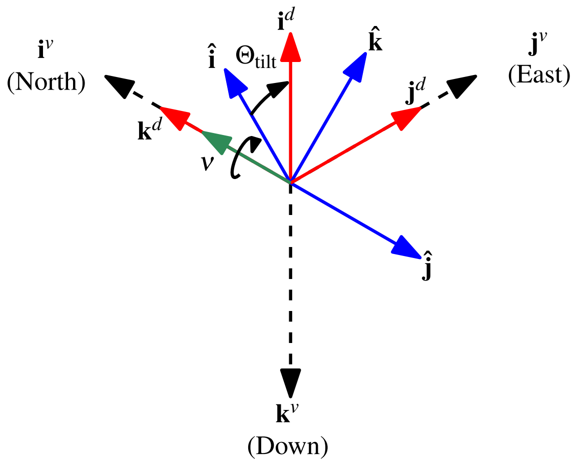


Figure 5.6: Tilt error compensation. In this case, the desired frame is still pointing up and belly facing north. The estimated frame has the belly pointing east and pitched up slightly (ϕ_h, θ_h and ψ_h are 90°, 10°, and 0°). A -10° rotation about the estimated body y-axis would align the x-axes.

To illustrate this case, we consider the scenario shown in Figure 5.6 where the estimated attitude is belly pointing east and slightly pitched up to $\theta_h = 10°$. For this,

$$\mathbf{R}_v^b(\eta^d) = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{R}_v^b(\hat{\eta}) = \begin{bmatrix} 0 & -0.17 & -0.98 \\ -1 & 0 & 0 \\ 0 & 0.98 & -0.17 \end{bmatrix}, \quad \tilde{\mathbf{R}}_v^b = \begin{bmatrix} 0.98 & 0 & 0.17 \\ -0.17 & 0 & 0.98 \\ 0 & -1 & 0 \end{bmatrix}. \quad (5.12)$$

To determine the rotation that would align the x-axes, the axes of both frames must be expressed in a common coordinate frame, the vehicle frame being the easiest to use. This is done using equation 2.11 and again noting that the rows of $\mathbf{R}_v^b$ represent the body axes expressed in the vehicle frame:

$$\begin{bmatrix} \mathbf{i}^d \\ \mathbf{j}^d \\ \mathbf{k}^d \end{bmatrix} \triangleq \mathbf{R}_v^b \left(\eta^d\right), \tag{5.13}$$

$$\begin{bmatrix} \hat{\mathbf{i}} \\ \hat{\mathbf{j}} \\ \hat{\mathbf{k}} \end{bmatrix} \triangleq \mathbf{R}_v^b \left(\hat{\eta}\right). \tag{5.14}$$

The angle of rotation needed to align the axes is determined from the dot product of the two x-axes or

$$\Theta_{\text{tilt}} = \cos^{-1}\left(\mathbf{i}^d \cdot \hat{\mathbf{i}}\right). \tag{5.15}$$

The unit-length axis of rotation is given by

$$\boldsymbol{\nu} = \frac{\hat{\mathbf{i}} \times \mathbf{i}^d}{|\hat{\mathbf{i}} \times \mathbf{i}^d|}, \tag{5.16}$$

which for this scenario, gives $\boldsymbol{\nu} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^\top$. Because the vectors used are expressed in the vehicle frame, the axis of rotation is also expressed in the vehicle frame. For the rotation to occur correctly, this axis must be expressed in the frame that is to be rotated, or in other words, the estimated body frame $\mathbf{R}_v^b(\hat{\eta})$. This is done by

$$\boldsymbol{\nu}^b = \mathbf{R}_v^b(\hat{\eta})\boldsymbol{\nu}. \tag{5.17}$$

For this case, the axis of rotation expressed properly is $\boldsymbol{\nu}^b = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top$, which is correctly the negative y-axis of the estimated body frame. Rodrigues' rotation formula [27] gives the rotation matrix that rotates a vector by a given angle about an arbitrary axis. Using $\boldsymbol{\nu}^b$ and Θ_{tilt} this gives

$$\mathbf{R}_{\nu^b} = \begin{cases} \mathbf{I} + [\boldsymbol{\nu}^{b\times}]\sin(\Theta_{\text{tilt}}) + [\boldsymbol{\nu}^{b\times}]^2[1 - \cos(\Theta_{\text{tilt}})], & \Theta_{\text{tilt}} \neq 0 \\ \mathbf{I} & \Theta_{\text{tilt}} = 0 \end{cases}, \tag{5.18}$$

where

$$[\boldsymbol{\nu}^{b\times}] = \begin{bmatrix} 0 & -\nu_z^b & \nu_y^b \\ \nu_z^b & 0 & -\nu_x^b \\ -\nu_y^b & \nu_x^b & 0 \end{bmatrix}. \tag{5.19}$$

It is important to note that the rotation matrix obtained from Equation 5.18 rotates a vector within a coordinate frame. We instead need to rotate the coordinate frame. This rotation matrix is obtained by transposing $\mathbf{R}_{\nu^b}$ or by simply changing the angle of rotation to $-\Theta_{\text{tilt}}$. The latter method has the advantage of being computationally faster and easier to implement. With this, Equation 5.18 becomes

$$\mathbf{R}_{\nu^b}^{\top} = \begin{cases} \mathbf{I} - [\boldsymbol{\nu}^{b\times}]\sin(\Theta_{\text{tilt}}) + [\boldsymbol{\nu}^{b\times}]^2[1 - \cos(\Theta_{\text{tilt}})], & \Theta_{\text{tilt}} \neq 0 \\ \mathbf{I} & \Theta_{\text{tilt}} = 0 \end{cases}. \tag{5.20}$$

The condition in Equations 5.18 and 5.20 is required because the axis of rotation becomes ill-defined as the cross product of $\hat{\mathbf{i}}$ and $\mathbf{i}^d$ goes to zero. The cross product is zero whenever these two axes are aligned, or in other words, when $\Theta_{\text{tilt}} = 0$. This condition is more appropriate than that originally given in [16] since the presence (or absence) of twist error does not effect whether or not the x-axes are aligned. $\tilde{\mathbf{R}}_\nu^b$ is only equal to identity when there is zero tilt and zero twist error. This condition has the added advantage of being easier to implement in programming languages such as C. The coordinate frame with the aligned x-axis is then given by

$$\mathbf{R}^a = \mathbf{R}_{\nu^b}^{\top}\mathbf{R}_\nu^b(\hat{\eta}). \tag{5.21}$$

The twist error is derived by decomposing this into row vectors, or

$$\begin{bmatrix} \mathbf{i}^a \\ \mathbf{j}^a \\ \mathbf{k}^a \end{bmatrix} \triangleq \mathbf{R}^a, \tag{5.22}$$

and then determining the angle of rotation by

$$\Theta_{\text{twist}} \triangleq \cos^{-1}\left(\mathbf{k}^a \cdot \mathbf{k}^d\right), \tag{5.23}$$

as shown in Figure 5.7. To determine the direction of rotation, Θ_{sign} is defined as

$$\Theta_{\text{sign}} \triangleq \cos^{-1}\left(\mathbf{j}^a \cdot \mathbf{k}^d\right), \tag{5.24}$$

making the twist error

$$\tilde{\Theta}_X = \begin{cases} -\Theta_{\text{twist}} & \Theta_{\text{sign}} \leq \frac{\pi}{2} \\ \Theta_{\text{twist}} & \Theta_{\text{sign}} > \frac{\pi}{2} \end{cases}. \tag{5.25}$$

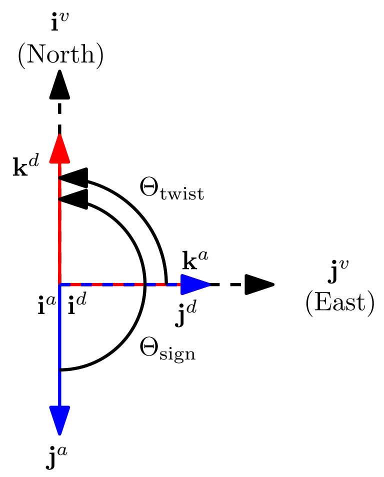


Figure 5.7: Twist angle definition.

Feedback Control

Feedback control is accomplished by

$$\delta_a = k_p\tilde{\Theta}_X - k_d p + k_i \int \tilde{\Theta}_X \,\mathrm{d}t, \tag{5.26}$$

$$\delta_e = k_p\tilde{\Theta}_Y - k_d q + k_i \int \tilde{\Theta}_Y \,\mathrm{d}t, \tag{5.27}$$

$$\delta_r = k_p\tilde{\Theta}_Z - k_d r + k_i \int \tilde{\Theta}_Z \,\mathrm{d}t, \tag{5.28}$$

where the control efforts are again mapped to their respective control surfaces such that a positive control effort results in a positive moment about that axis. Figure 5.8 shows the error calculated based on the resolved tilt-twist method given the same scenario as in Figure 5.3. The desired attitude is nose pointing up with belly facing north. The estimated attitude is pitched down slightly to $\theta_h = -10°$ with a varying heading. As opposed to the quaternion attitude error computation, the RTT method pitch error is constant and the heading error linearly increases, both as would be expected.

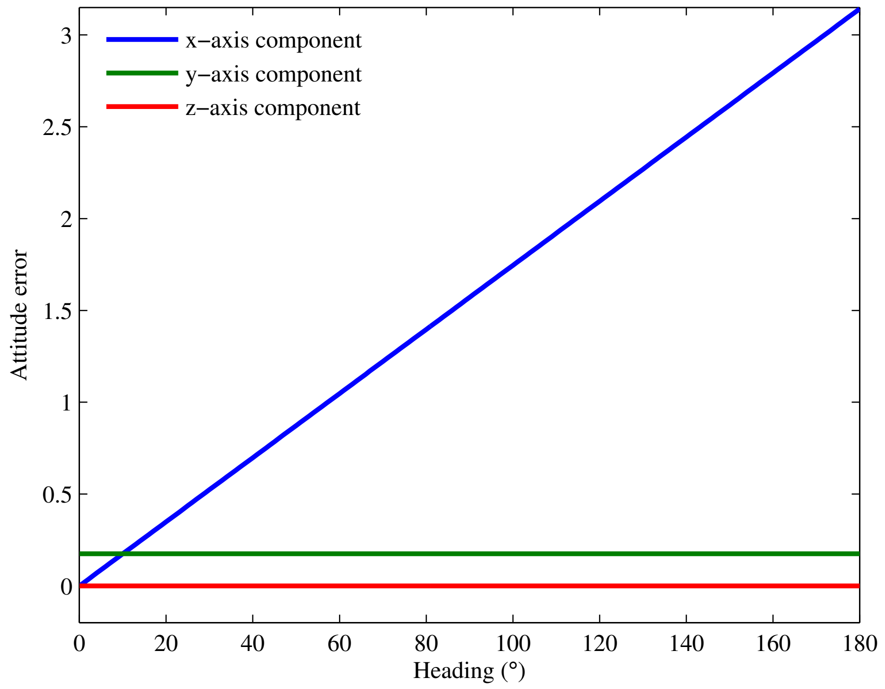


Figure 5.8: Resolved tilt-twist output given a similar scenario as the quaternion attitude control in Figure 5.3.

RTT is summarized in Algorithm 2 and has been implemented in hardware. It was used to stabilize and change the heading of the Y-Bat for the results presented in Section 4.4. Additional results are given in Section 5.5.

Algorithm 2 Resolved Tilt-Twist Control
Input: The estimated vehicle attitude, expressed as a rotation matrix, $R_v^b(\hat{\eta})$ and the desired vehicle attitude, expressed as a rotation matrix, $R_v^b\left(\eta^d\right)$.

```
 1: Calculate Tilt Error
 2: $\tilde{\mathbf{R}}_v^b \triangleq \mathbf{R}_v^b\left(\eta^d\right)\mathbf{R}_v^b(\hat{\eta})^\top$    ▹ Calculate attitude error
 3: $\tilde{\Theta}_Y = -\operatorname{atan2}[\tilde{r}_{13}, \tilde{r}_{11}]$    ▹ Calculate pitch tilt error
 4: $\tilde{\Theta}_Z = \operatorname{atan2}[\tilde{r}_{12}, \tilde{r}_{11}]$    ▹ Calculate yaw tilt error
 5:
 6: Calculate twist error
 7: $\hat{\mathbf{i}} \triangleq \begin{bmatrix}\hat{r}_{11} & \hat{r}_{12} & \hat{r}_{13}\end{bmatrix}^\top$    ▹ Estimated $x$-axis
 8: $\mathbf{i}^d \triangleq \begin{bmatrix}r_{11}^d & r_{12}^d & r_{13}^d\end{bmatrix}^\top$    ▹ Desired $x$-axis
 9: $\mathbf{k}^d \triangleq \begin{bmatrix}r_{31}^d & r_{32}^d & r_{33}^d\end{bmatrix}^\top$    ▹ Desired $z$-axis
10: $\Theta_{\text{tilt}} = \cos^{-1}\left(\hat{\mathbf{i}} \cdot \mathbf{i}^d\right)$    ▹ Define total tilt error
11: if $\Theta_{\text{tilt}} = 0$ then
12:     $\mathbf{R}_{\nu^b} = \mathbf{I}$    ▹ $x$-axes are already aligned
13: else
14:     $\boldsymbol{\nu} = \frac{\hat{\mathbf{i}} \times \mathbf{i}^d}{|\hat{\mathbf{i}} \times \mathbf{i}^d|}$    ▹ Rotation axis
15:     $\boldsymbol{\nu}^b = \mathbf{R}_v^b(\hat{\eta})\boldsymbol{\nu}$    ▹ Rotation axis expressed in estimated frame
16:     $\mathbf{R}_{\nu^b}^\top = \mathbf{I} - [\boldsymbol{\nu}^{b\times}]\sin(\Theta_{\text{tilt}}) + [\boldsymbol{\nu}^{b\times}]^2[1 - \cos(\Theta_{\text{tilt}})]$    ▹ Rotation matrix to align axes
17: end if
18: $\mathbf{R}^a = \mathbf{R}_{\nu^b}^\top \mathbf{R}_v^b(\hat{\eta})$    ▹ Align $x$-axes
19: $\mathbf{j}^a \triangleq \begin{bmatrix}r_{21}^a & r_{22}^a & r_{23}^a\end{bmatrix}^\top$    ▹ $y$-axis of aligned frame
20: $\mathbf{k}^a \triangleq \begin{bmatrix}r_{31}^a & r_{32}^a & r_{33}^a\end{bmatrix}^\top$    ▹ $z$-axis of aligned frame
21: $\Theta_{\text{twist}} \triangleq \cos^{-1}\left(\mathbf{k}^a \cdot \mathbf{k}^d\right)$    ▹ Calculate twist error
22: $\Theta_{\text{sign}} \triangleq \cos^{-1}\left(\mathbf{j}^a \cdot \mathbf{k}^d\right)$    ▹ Define $\Theta_{\text{sign}}$
23: $\tilde{\Theta}_X = \begin{cases} -\Theta_{\text{twist}} & \Theta_{\text{sign}} \leq \frac{\pi}{2} \\ \Theta_{\text{twist}} & \Theta_{\text{sign}} > \frac{\pi}{2} \end{cases}$    ▹ Rotate correct direction
24: $\delta_a = k_p\tilde{\Theta}_X - k_d p + k_i \int \tilde{\Theta}_X \, \mathrm{d}t$
25: $\delta_e = k_p\tilde{\Theta}_Y - k_d q + k_i \int \tilde{\Theta}_Y \, \mathrm{d}t$
26: $\delta_r = k_p\tilde{\Theta}_Z - k_d r + k_i \int \tilde{\Theta}_Z \, \mathrm{d}t$
```

5.1.3 Gain Scheduling

Because the effectiveness of the control surfaces is influenced by airspeed and throttle setting it is necessary to schedule the controller output for δ_a, δ_e, or δ_r (generically referred to here as δ_i) to obtain consistent performance at a variety of airspeeds and throttle settings. The simplest way to schedule the controller output with airspeed is to scale it by $\frac{1}{V^2}$; however, as airspeed gets small the control output grows rapidly and without bound unless externally saturated. Combining this with throttle in $\frac{1}{k_v V^2 + k_t \delta_t^2}$ where k_v and k_t are tunable gains offers better behavior since it is very unlikely throttle and airspeed will be simultaneously zero; however, there is still no bound on the output unless an external limit function is used. A sample output of this is shown in Figure 5.9.

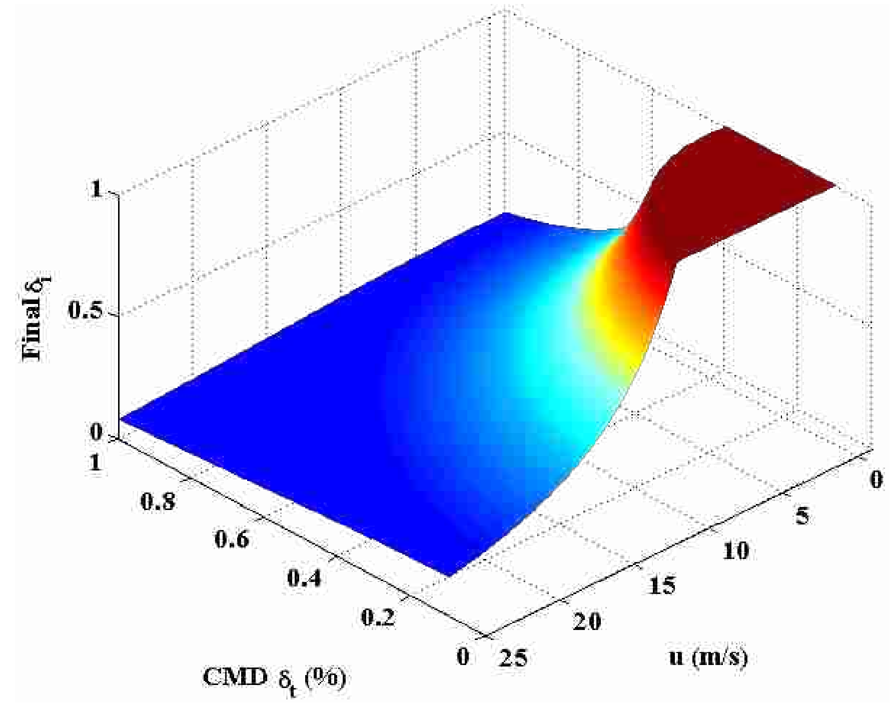
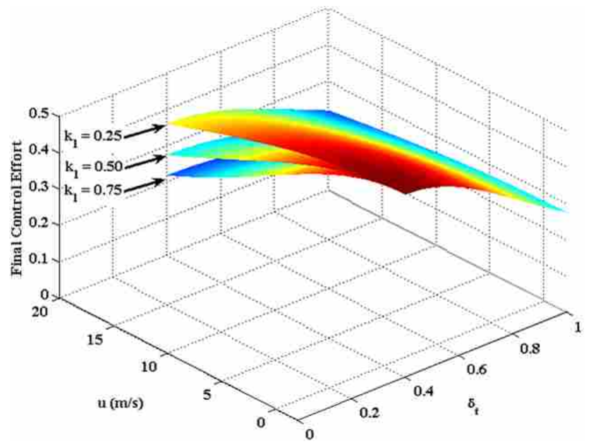
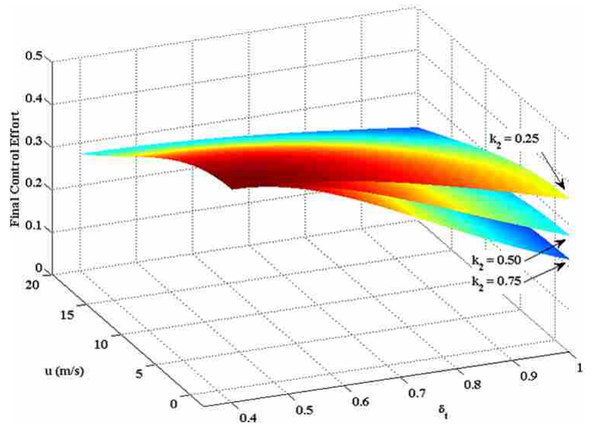


Figure 5.9: Non-normalized inverse gain schedule. This plot shows the scheduled output for various airspeeds and throttle settings for an initial input of $\delta_i = 0.5$. The output is saturated at $\delta_i = 1$.

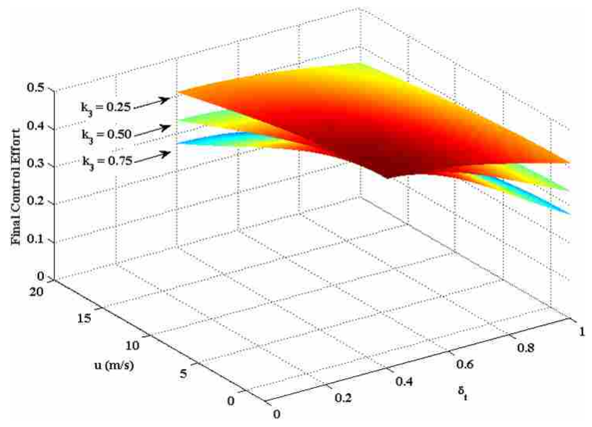
Another method which we refer to as the "normalized exponential gain schedule" is shown in Figure 5.10. To calculate the schedule the airspeed and throttle commands are normalized and



(a) Effect of k_1. Increasing k_1 increases the effect u has on the schedule. In this plot k_2 is held constant at 0.5.



(b) Effect of k_2. Increasing k_2 increases the effect δ_t has on the schedule. In this plot k_1 is held constant at 0.5.



(c) Effect of k_3. Increasing k_3 increases the overall effect of the schedule. In this plot k_1 and k_2 are held constant at 0.5.

Figure 5.10: Normalized exponential gain schedule. In both cases the unscheduled input is $\delta_i = 0.5$. No output saturation is used. Red areas mean the schedule has had the least effect (most control output). Deep blue regions are where the schedule has reduced the output the most.

blended together by

$$A \triangleq 1 + k_1 \left(\frac{u - u_{min}}{u_{max} - u_{min}} \right)^2 + k_2 \left(\frac{\delta_t - \delta_{t,min}}{1 - \delta_{t,min}} \right)^2 \qquad \begin{array}{c} u_{min} \leq u \leq u_{max} \\ 0 \leq \delta_{t,min} \leq \delta_t \leq 1 \end{array}, \tag{5.29}$$

where k_1 and k_2 are positive gains that determine the effect u and δ_t have on the schedule with larger gains causing a greater effect. The schedule is then calculated by

$$S \triangleq \left(\frac{1}{A} \right)^{k_3} \qquad k_3 > 0, \tag{5.30}$$

where k_3 determines the overall effect the schedule has on the control effort δ_i–the larger it is the more effect the schedule has. When $k_3 = 0$, the schedule is shut off. The schedule is applied to the control efforts by

$$\delta_i^+ = S\delta_i^-. \tag{5.31}$$

The schedule was designed so that it never increases the input control; it only decreases it based on conditions. Another goal with its design was to allow gains to be tuned individually so that changing one gain does not affect another gain. This arrangement makes it so that the controller gains can be tuned when operating close to $\delta_{t,min}$ and u_{min} when the schedule has little or no effect. At high throttle and airspeeds the commanded efforts are adjusted by only changing the schedule parameters–not the controller gains. It is assumed that $\delta_{t,max} = 1$. At this setting and when $u = u_{max}$ the schedule has the greatest effect. As such, these values can somewhat be used to tune the shape of the schedule given the above constraints.

5.2 Position Control

The position control used during testing is now presented. The basic idea of the hover position control is that the aircraft tilts in the direction that it wants to move in. This controller can be divided into three parts.

The first part of the controller determines the desired translational ground speed by using the horizontal position error. The position error is

$$\tilde{\mathbf{p}} = \mathbf{p}^d - \hat{\mathbf{p}}, \tag{5.32}$$

where $\hat{\mathbf{p}} = \begin{bmatrix}\hat{p}_n & \hat{p}_e\end{bmatrix}^\top$ is the current horizontal position and $\mathbf{p}^d = \begin{bmatrix}p_n^d & p_e^d\end{bmatrix}^\top$ is the desired horizontal position. The desired horizontal ground velocity magnitude is computed by

$$\dot{x}^d = k_{\dot{x}_1}\frac{2}{\pi}\tan^{-1}\left(\frac{k_{\dot{x}_2}}{k_{\dot{x}_1}}|\tilde{\mathbf{p}}|\right), \tag{5.33}$$

where the inverse tangent is used as a saturation function with $k_{\dot{x}_1}$ as the maximum desired velocity. The parameter $k_{\dot{x}_2}$ determines how fast the transition between extreme values occurs. Increasing it makes the transition occur faster. The velocity direction is given by

$$\psi_{\dot{x}}^d = \text{atan2}\left[\tilde{p}_e, \tilde{p}_n\right]. \tag{5.34}$$

This portion of the controller is summarized in Figure 5.11.

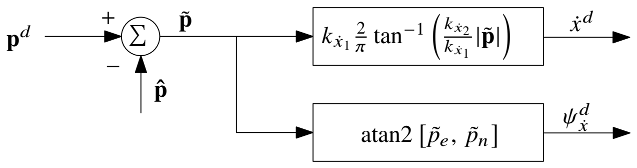

$\mathbf{p}^d$
+
Σ
−
$\hat{\mathbf{p}}$
$\tilde{\mathbf{p}}$
$k_{\dot{x}_1}\frac{2}{\pi}\tan^{-1}\left(\frac{k_{\dot{x}_2}}{k_{\dot{x}_1}}|\tilde{\mathbf{p}}|\right)$
$\dot{x}^d$
atan2 $[\tilde{p}_e, \tilde{p}_n]$
$\psi_{\dot{x}}^d$

Figure 5.11: The first section of the hover flight horizontal position controller takes the horizontal position error and computes a desired horizontal velocity.

The second section of the controller takes the desired horizontal velocity and modifies it to reduce the steady state tracking error of the desired translational velocity. This modification also improves the performance of the position controller in wind. This modification is accomplished by first converting the desired translational velocity into a desired north and east velocity

$$\dot{p}_n^d = \dot{x}^d\cos(\psi_{\dot{x}}^d) - k_d\dot{p}_n, \tag{5.35}$$

$$\dot{p}_e^d = \dot{x}^d\sin(\psi_{\dot{x}}^d) - k_d\dot{p}_e. \tag{5.36}$$

The translational velocity error in the inertial frame is computed by

$$\tilde{\dot{p}}_n = \dot{p}_n^d - \hat{\dot{p}}_n, \tag{5.37}$$

$$\tilde{\dot{p}}_e = \dot{p}_e^d - \hat{\dot{p}}_e, \tag{5.38}$$

where the current estimated velocities are given by

$$\hat{\dot{p}}_n = \left(\eta_x^2 + \eta_0^2 - \eta_y^2 - \eta_z^2\right) u + 2\left(\eta_x\eta_y - \eta_z\eta_0\right) v + 2\left(\eta_x\eta_z + \eta_y\eta_0\right) w, \tag{5.39}$$

$$\hat{\dot{p}}_e = 2\left(\eta_x\eta_y + \eta_z\eta_0\right) u + \left(\eta_y^2 + \eta_0^2 - \eta_x^2 - \eta_z^2\right) v + 2\left(\eta_y\eta_z - \eta_x\eta_0\right) w. \tag{5.40}$$

The scaled desired translational velocity is computed by

$$\dot{y}^d = \sqrt{\left(\tilde{\dot{p}}_n - k_d\ddot{p}_n + k_i \int \tilde{\dot{p}}_n\right)^2 + \left(\tilde{\dot{p}}_e - k_d\ddot{p}_e + k_i \int \tilde{\dot{p}}_e\right)^2}, \tag{5.41}$$

$$\psi_{\dot{y}}^d = \operatorname{atan2}\left[\dot{p}_e - k_d\ddot{p}_e + k_i \int \tilde{\dot{p}}_e,\ \tilde{\dot{p}}_n - k_d\ddot{p}_n + k_i \int \tilde{\dot{p}}_n\right], \tag{5.42}$$

where $\dot{y}^d$ is the magnitude of the modified desired translational velocity and $\psi_{\dot{y}}^d$ is the direction. The inertial velocities are used in Equation 5.41 instead of body frame velocities so that if the aircraft's heading changes, the integrated error still compensates for wind correctly. The accelerations $\ddot{p}_n$ and $\ddot{p}_e$ are obtained by rotating the accelerometer measurements back to the vehicle frame.

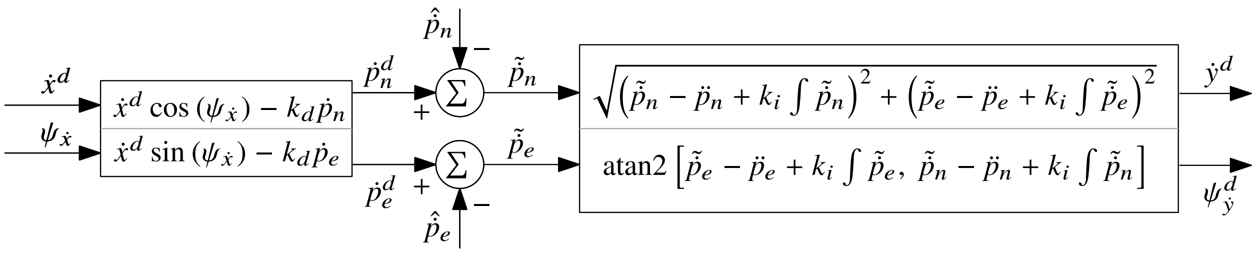

$\dot{x}^d$
$\psi_{\dot{x}}$
$\dot{x}^d \cos(\psi_{\dot{x}}) - k_d\dot{p}_n$
$\dot{x}^d \sin(\psi_{\dot{x}}) - k_d\dot{p}_e$
$\dot{p}_n^d$
$\dot{p}_e^d$
$\hat{\dot{p}}_n$
$\hat{\dot{p}}_e$
Σ
+
−
$\tilde{\dot{p}}_n$
$\tilde{\dot{p}}_e$
$\sqrt{\left(\tilde{\dot{p}}_n - \ddot{p}_n + k_i \int \tilde{\dot{p}}_n\right)^2 + \left(\tilde{\dot{p}}_e - \ddot{p}_e + k_i \int \tilde{\dot{p}}_e\right)^2}$
$\operatorname{atan2}\left[\tilde{\dot{p}}_e - \ddot{p}_e + k_i \int \tilde{\dot{p}}_e,\ \tilde{\dot{p}}_n - \ddot{p}_n + k_i \int \tilde{\dot{p}}_n\right]$
$\dot{y}^d$
$\psi_{\dot{y}}^d$

Figure 5.12: The second section of the hover flight horizontal position controller modifies the desired horizontal velocity to reduce steady state error and the effects of wind.

A limit to the integrated velocity error is added to handle the integrator wind up. Using Euler's method, the error in north and east velocity is integrated by

$$\tilde{p}_n^{i+} = \tilde{p}_n^{i-} + T_s \tilde{\dot{p}}_n, \tag{5.43}$$

$$\tilde{p}_e^{i+} = \tilde{p}_e^{i-} + T_s \tilde{\dot{p}}_e. \tag{5.44}$$

To saturate the integration effort we ensure that

$$\left\| \begin{bmatrix} \tilde{p}_n^i & \tilde{p}_e^i \end{bmatrix} \right\|_2 \leq L, \tag{5.45}$$

where $\|\cdot\|_2$ is the L_2 norm and L is the saturation limit. After $\tilde{p}_n^i$ has been updated by equation 5.43 it is saturated by

$$\tilde{p}_n^i = \begin{cases} \tilde{p}_n^i, & v \leq L \\ \frac{L\tilde{p}_n^i}{v}, & \text{else} \end{cases} \tag{5.46}$$

where

$$v = \sqrt{\tilde{p}_n^{i\,2} + \tilde{p}_e^{i\,2}} \tag{5.47}$$

is the magnitude of the integrated velocity error vector. The east component of velocity is similarly updated. One nice feature of this method is that we can set a hard limit on the integrator effect on the overall velocity instead of just the components of the velocity. In addition, the saturation only changes the magnitude of the integration vector and not its direction.

The third section of the controller, shown in Figure 5.13, takes the modified translational velocity and the desired heading and computes a desired attitude. First, the modified translational velocity is converted into a desired tilt angle by

$$\Theta_{\text{tilt}}^d = k_{\Theta_1} \frac{2}{\pi} \tan^{-1}\left(\frac{k_{\Theta_2}}{k_{\Theta_1}} \dot{y}^d \right), \tag{5.48}$$

where k_{Θ_1} is the maximum tilt angle and k_{Θ_2} is a scaling factor. The desired tilt angle and its direction are then converted into a quaternion by

$$\eta_{\text{tilt}}^{d} = \begin{bmatrix} \cos\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ \cos\left(\psi_{\dot{y}}\right)\sin\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ \sin\left(\psi_{\dot{y}}\right)\sin\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ 0 \end{bmatrix}. \tag{5.49}$$

The desired heading is also converted to a quaternion, which also takes into account the hover aspect of the attitude, or in other words that the nose of the tailsitter should be pointed up, by

$$\eta_{\text{h}}^{d} = \begin{bmatrix} \frac{1}{\sqrt{2}}\cos\left(\frac{\psi_h^d}{2}\right) \\ \frac{1}{\sqrt{2}}\sin\left(\frac{\psi_h^d}{2}\right) \\ \frac{1}{\sqrt{2}}\cos\left(\frac{\psi_h^d}{2}\right) \\ -\frac{1}{\sqrt{2}}\sin\left(\frac{\psi_h^d}{2}\right) \end{bmatrix}. \tag{5.50}$$

The two quaternions are composed to arrive at the desired attitude,

$$\eta^{d} = \eta_{\text{tilt}}^{d} \otimes \eta_{\text{h}}^{d}, \tag{5.51}$$

which is sent to the attitude controller.

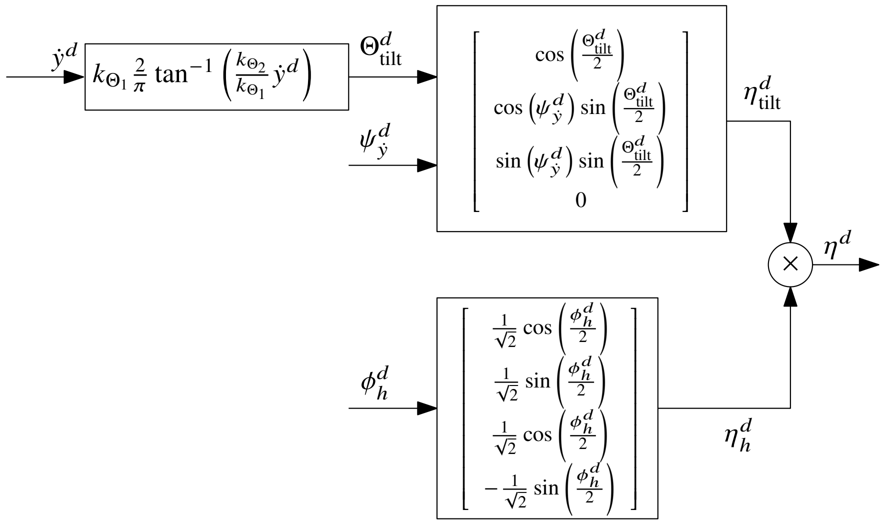

$\dot{y}^d$
$k_{\Theta_1}\frac{2}{\pi}\tan^{-1}\left(\frac{k_{\Theta_2}}{k_{\Theta_1}}\dot{y}^d\right)$
Θ_{tilt}^d
$\psi_{\dot{y}}^d$
$\begin{bmatrix} \cos\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ \cos\left(\psi_{\dot{y}}^d\right)\sin\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ \sin\left(\psi_{\dot{y}}^d\right)\sin\left(\frac{\Theta_{\text{tilt}}^{d}}{2}\right) \\ 0 \end{bmatrix}$
η_{tilt}^d
$\times$
η^d
ϕ_h^d
$\begin{bmatrix} \frac{1}{\sqrt{2}}\cos\left(\frac{\phi_h^d}{2}\right) \\ \frac{1}{\sqrt{2}}\sin\left(\frac{\phi_h^d}{2}\right) \\ \frac{1}{\sqrt{2}}\cos\left(\frac{\phi_h^d}{2}\right) \\ -\frac{1}{\sqrt{2}}\sin\left(\frac{\phi_h^d}{2}\right) \end{bmatrix}$
η_h^d

Figure 5.13: Third section of the hover flight horizontal position controller. Converts the desired translational velocity into a tilt angle and computes the desired attitude.

5.3 Altitude Control

The thrust controller has been slightly modified from what is shown in Figure 5.1 to add a altitude rate outer loop. The strategy is that the outer loop operates on the commanded altitude h^d and current altitude $\hat{h}$ to determine altitude error and determine a corrective climb rate $\dot{h}^d$. This is critical because an excessive descent rate during landing could lead to a loss of control authority and possible crash. A hard limit is placed on the commanded climb rate to ensure an excessive descent rate is not commanded.

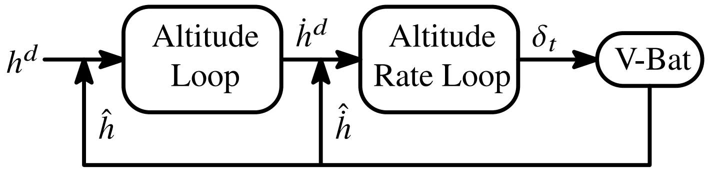


Figure 5.14: Altitude control scheme. The outer loop computes altitude error and determines a desired climb/descent rate. The rate loop controls throttle to obtain this rate.

5.3.1 Altitude Loop

The outer loop determines altitude error and computes the desired altitude rate by

$$\tilde{h} = h^d - \hat{h}, \tag{5.52}$$

$$\dot{h}^d = \text{sat}\left[k_p\tilde{h} - k_d\hat{\dot{h}}, L_\ell, L_u\right], \tag{5.53}$$

where $\tilde{h}$ is the altitude error. To ensure an excessive descent rate is not commanded, Equation 5.53 includes a saturation function, with L_u and L_ℓ being the upper and lower limits. An inverse tangent function could be used to perform this (as in Equation 5.33); however, this forces the upper and lower limits of the output to have the same magnitude and is not desirable in this case.

Typically, integrators are used in outer loops to account for steady-state error in the inner loops. It is generally not good practice to have integrators on the inner loops since they slow the inner loop dynamics. However, having the integrator on the outer loop makes it difficult to enforce a hard limit on commanded climb or descent rates. Because of this, the integrator is placed in the inner loop. An additional integrator is not required on the outer loop.

One issue is how to obtain $\hat{h}$ and $\hat{\dot{h}}$. If a combined pitot/static probe is used to obtain airspeed data, using the static port to measure barometric pressure is likely to give poor results. This is particularly true in hover where the pitot probe is perpendicular to the horizontal plane. This orientation could allow either airflow from vehicle translation or wind to blow directly into the static port of the pitot tube. A better way would be to vent the barometric sensor to the interior of the UAV, covering it with foam to protect it from any wind currents that might be present within the UAV body.

5.3.2 Altitude Rate Loop

Once the desired altitude rate is determined, throttle is used to obtain this rate by

$$\tilde{\dot{h}} = \dot{h}^d - \dot{h}, \tag{5.54}$$

$$\delta_t = k_p \tilde{\dot{h}} - k_d \ddot{x} + k_i \int \tilde{\dot{h}} \, \mathrm{d}t + \delta_{t,trim}, \tag{5.55}$$

where $\tilde{\dot{h}}$ is the climb rate error and $\delta_{t,trim}$ is the feed-forward term or the throttle setting needed to maintain zero climb rate. The derivative term is $\ddot{x}$ is obtained from subtracting gravity from the x-axis accelerometer or

$$\ddot{x} \triangleq a_x^* - 2g(\hat{\eta}_0 \hat{\eta}_y - \hat{\eta}_x \hat{\eta}_z). \tag{5.56}$$

The feed-forward term is used to counteract gravity and minimizes the amount the integrator effort required to hover. This in turn minimizes the performance penalties introduced by the integrator. A typical way to implement the feed-forward term is to schedule it based on pitch, or in other words, as pitch is reduced so is the feed-forward term. This approach is more applicable when there is another source of lift generation, such as during forward flight. It is problematic for tailsitters when descending because if the vehicle begins to tip due to lack of control authority, this schedule would reduce the feed-forward term leading to further loss of control authority. The most straight-forward solution is to keep the feed-forward term constant. During hover flight, another possible way to is actually increase the feed-forward term as the pitch moves away from $\theta_h = 0°$. This would be based on keeping the vertical component of thrust equal to weight or

$$T \cos\theta_h = mg, \tag{5.57}$$

where T is the thrust produced by the motor. One slight difficulty with this is that thrust is not linearly related to throttle position.

5.4 Guidance for Hardware Testing

When determining how to implement these loops on the actual tailsitter, there were several considerations including how to tune each loop separately and how to account for certain loops not being active. One of the issues when initially trying to tune the altitude loop on the Y-Bat is that the vehicle position would drift fairly significantly since position feedback control was not yet operational, even though the vehicle was maintaining a vertical attitude. The following guidance test modes allow an RC pilot to maintain control when various feedback loops are not active:

1. **RC Mode** - this mode gives a remote control pilot the most control over the vehicle and provides only rate-damping control. It is intended only for emergency use to get the vehicle into an attitude that can then be stabilized by the autopilot.

2. **FBW Mode** - Fly-by-wire mode. This mode adds tilt attitude control to RC Mode and is intended to be the default go-to mode to get the vehicle out of trouble.

3. **Tune Mode** - This mode has several sub-modes that add increasing levels of autonomy and is intended to allow loops to be tuned individually.

5.4.1 RC Mode

RC mode allows a human pilot to take over control of the vehicle. The autopilot only provides rate damping about the three vehicle axes. It was debated on whether or not to have the RC sticks behave differently when in hover versus when in level flight. It was decided that horizontal and vertical movement of the left stick would always be linked respectively to rotations about the body z-axis and throttle control. Horizontal and vertical movements of the right stick would always be linked to rotations about the body x and y-axes. This is normal for airplanes in level flight, but slightly unusual for hover flight. It was felt consistency was more important so that the pilot would never have to figure out what mode they are in.

5.4.2 FBW Mode

FBW mode also allows a human pilot to take control of the vehicle, but with a higher level of autonomy. In this mode the elevator RC stick command (δ_e^c), the rudder RC stick command (δ_r^c) and the current heading ($\hat{\phi}_h$), are transformed into a desired attitude quaternion by

$$\phi_h = \hat{\phi}_h, \tag{5.58}$$

$$\theta_h = \delta_e^c, \tag{5.59}$$

$$\psi_h = \delta_r^c, \tag{5.60}$$

and Equation A.8. Doing this allows the RC pilot to control the tilt angle. Including the current heading is necessary so that the "heading" of the desired quaternion is always aligned with the heading of the vehicle. Without this, the desired quaternion would always have the belly facing north, resulting in unexpected behavior. The desired quaternion is then sent to the inner (attitude) loop. Lastly, in FBW mode, the RC throttle stick command is linked directly with the vehicle throttle. No feedback control is used.

5.4.3 Tune Mode

This mode is used to tune the position and altitude control loops. This mode has 12 sub-modes–four position modes and three altitude modes that can be used in any combination. Each mode progressively adds autonomy to the previous mode and are set by a flag in the ground control station (GCS). The position modes are

1. **FBW Mode** - in this mode position control functions exactly as it does when in the FBW mode described in Section 5.4.2-the RC pilot has tilt attitude control and rate-damping on heading.

2. **YRate Mode** - in this mode a GCS user directly enters the scaled translational velocity magnitude $\dot{y}^d$ and direction $\psi_{\dot{y}}^d$ instead of the autopilot computing them by Equations 5.41 and 5.42.

3. **XRate Mode** - in this mode a GCS user directly enters the desired translational velocity magnitude $\dot{x}^d$ and direction $\psi_{\dot{x}}^d$ instead of the autopilot computing them by Equations 5.33 and 5.34.

4. **Position Mode** - in this mode, a GCS user directly enters the desired north and east coordinates in meters.

The altitude modes are

1. **FBW Mode** - in this mode altitude control functions exactly as it does in the FBW mode described in Section 5.4.2- the RC throttle stick command is passed directly to the motor.

2. **Altitude Rate Mode** - in this mode a GCS user directly enters a desired climb rate.

3. **Altitude Mode** - in this mode a GCS user directly enters the desired altitude.

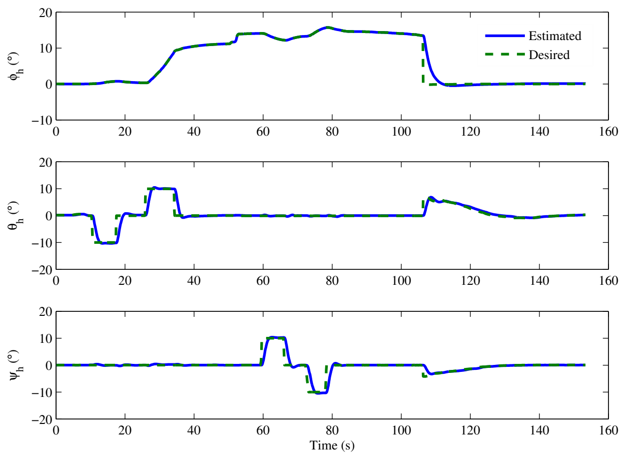
5.5 Hardware in Loop Simulation

For debugging and algorithm validation, hardware-in-the-loop simulation tests were conducted. All autopilot control and estimation functions were performed by a Kestrel autopilot. The autopilot sent the simulator control commands and the simulator returned sensor information to the autopilot.

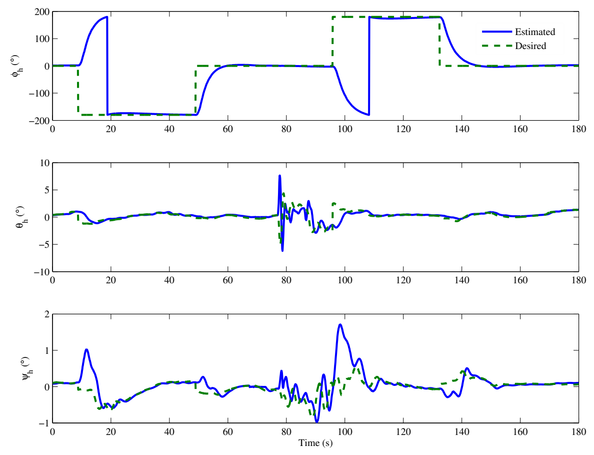
Initial RTT performance was tested by using the FBW guidance mode described in Section 5.4. Figure 5.15(a) shows the result of commanding tilt steps. At approximately $t = 105$s, the position guidance mode was enabled driving ϕ_h to zero and position to 0 m north and 0 m east.

To demonstrate roll performance the XRate guidance mode was used since it allowed us to issue arbitrary heading commands. Figure 5.15(b) shows several 180° heading steps while a translational velocity was commanded to be zero.

Figure 5.16 shows additional velocity commands. In this scenario, the tailsitter was initially still with it's belly facing north. It was commanded a velocity of 1 m/s north. Once that had stabilized, it was commanded a velocity of 1 m/s south and heading of 180°.

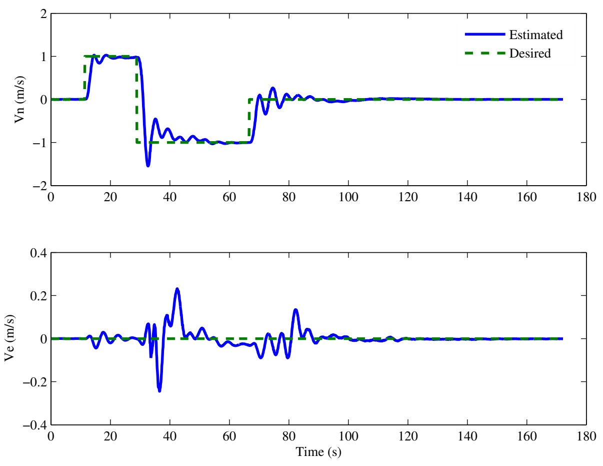


(a) Pitch and yaw steps

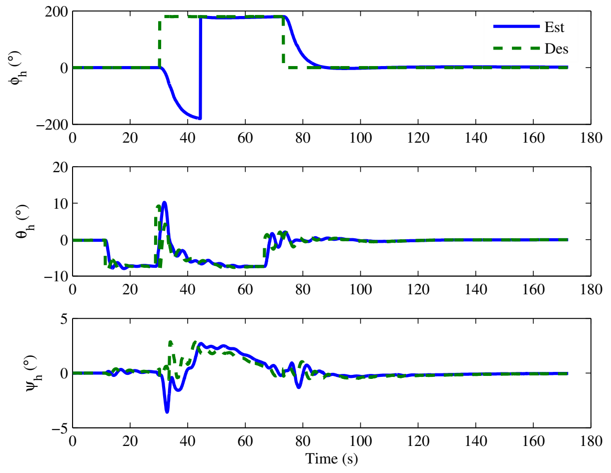


(b) Roll steps

Figure 5.15: Attitude steps using RTT



(a) Translational velocity



(b) Attitude

Figure 5.16: Controlling translational velocity using XRate mode and RTT

Work in these higher level loops is ongoing, as evidenced by the oscillations in Figure 5.16. In spite of this, the attitude loop performed well, allowing the vehicle to complete the maneuver.

The final test presented here was for the tailsitter to do the following: climb to an altitude of 10 m, fly to the following north-east coordinates (10, 0), (10, 10), (10, 0), and (0, 0) and then descend back to an altitude of zero. The data presented was logged by the Kestrel autopilot.

Figure 5.17 shows the route flown. Overall the route was flown well. Some of the velocities are a little lower that what would be desirable on the actual eV-Bat, so there is still some work to do in tuning gains and setting limits. Also, the waypoints were changed manually. Waypoint sequencing is not used yet.

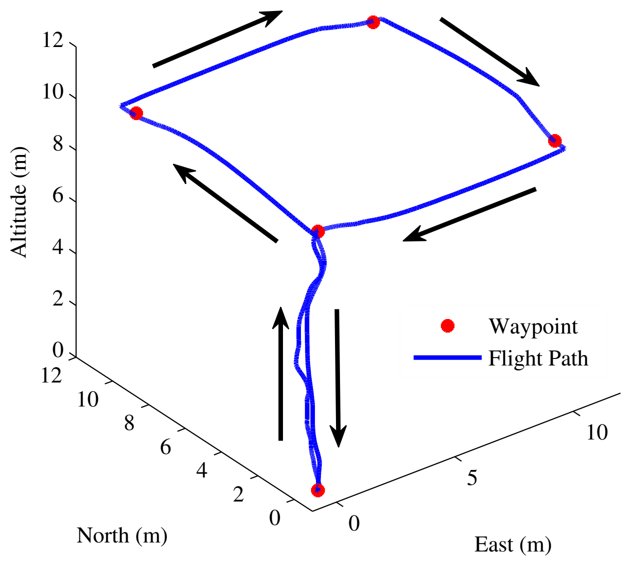
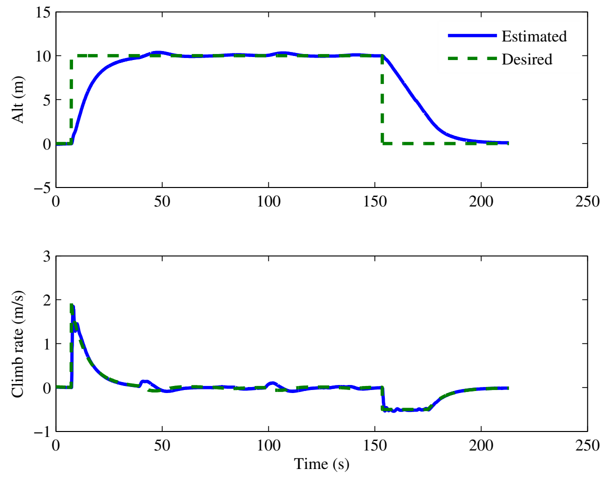
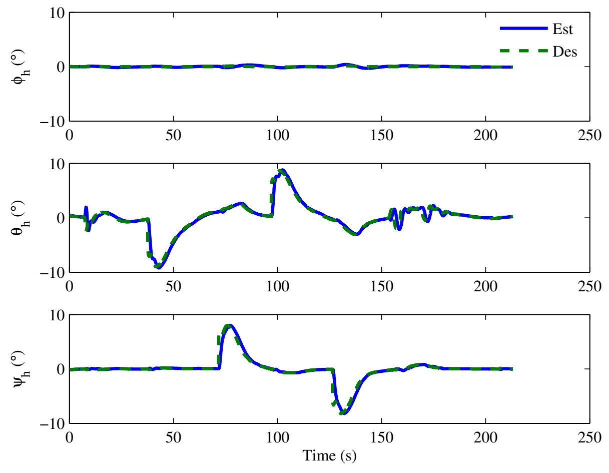


Figure 5.17: Test route flown for HIL simulation

Figure 5.18(a) shows how closely the altitude and climb rate followed their commanded values. It would most likely be desirable to increase the climb performance to get to the commanded altitude faster. Also, a low descent rate was commanded to ensure adequate control authority. Additional testing will determine if that value can be increased. Figure 5.18(b) shows the attitude loop performance with the attitude error being determined by the resolved tilt-twist method. Both performed well. Figure 5.19 shows the position and translation velocity performance. The position control performed well, but could probably be improved. The velocity control also performed as expected, even though the commanded values were lower than what would be desirable in actual hardware. This is where performance needs to increase the most.

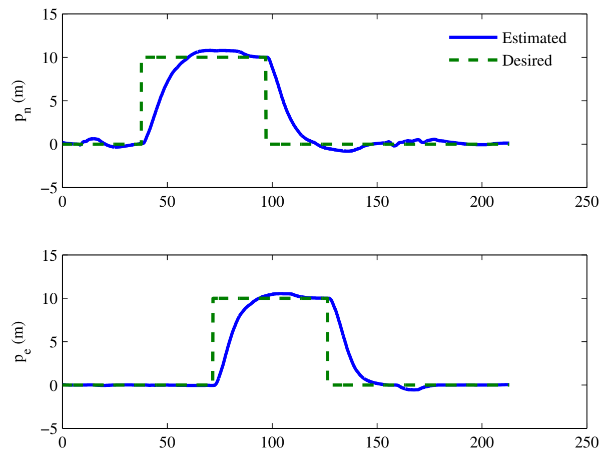


(a) Altitude

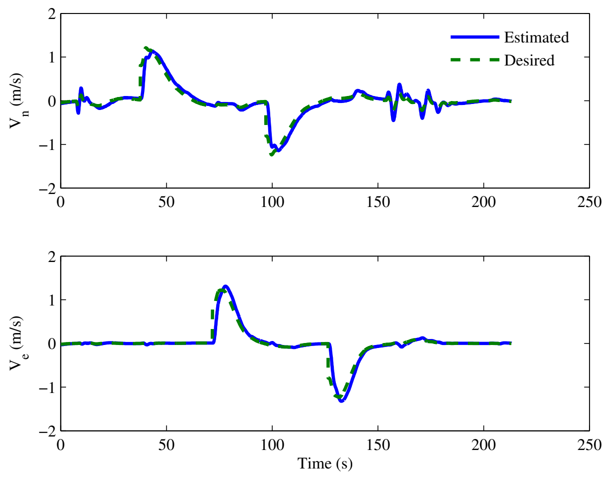


(b) Attitude

Figure 5.18: HIL route altitude and attitude control



(a) Position



(b) Velocity

Figure 5.19: HIL route position and velocity control

5.6 Conclusion

This chapter discussed why quaternion feedback may not be the best attitude control for a tailsitter and presented the RTT method as an alternative. Position and altitude control have also been outlined. The chapter concluded with hardware-in-the-loop test results.

CHAPTER 6. WIND ANALYSIS

One aspect of this research was to characterize the response of the V-Bat when hovering in wind. Intuitively it makes sense that without any control input, any wind would cause the V-Bat to rotate such that the wings are perpendicular to the wind. This chapter presents an analysis that was performed to show this behavior mathematically. The analysis required knowledge of the roll moment (moment about the body x-axis) generated at a given wind incidence angle, β, shown in Figure 6.1. Because a good analytical model is difficult to obtain due to the complex 3-D flow involved, experimental data was collected and used to characterize the roll moment as a function of β. The collected data is presented in Section 6.1. The experimental data is curve fit in Section 6.2 and the dynamic model is defined in Section 6.3. The model is analyzed to explain this behavior in Section 6.4.

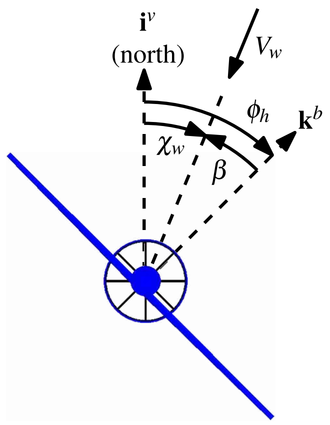


Figure 6.1: Wind tunnel angle definitions. Heading is given by ϕ_h. The wind direction χ_w is the angle between true north and the wind vector. β is the wind incidence angle.

To start, the angles χ_w and β must be formally defined. Heading, ϕ_h, has previously been defined as a left-handed rotation about the hover frame x-axis and that definition is maintained. The wind direction is the angle between true north and the direction the wind is coming from and is expressed as a right-handed rotation about the inertial z-axis. The wind incidence angle β is the angle between the body z-axis and the direction the wind is coming from and is expressed as a right-handed rotation about the body x-axis. The angles are related to each other by

$$\beta = \phi_h - \chi_w. \tag{6.1}$$

For wind tunnel testing χ_w is assumed to be zero.

6.1 Wind Tunnel Data

The goal of the wind tunnel data was to obtain data that could be curve fit and analyzed. For this, the roll moment generated by wind was measured as the model shown in Figure 6.2 was rotated a full 360°. The model is a 1/10 scale model of the full-size V-Bat built by Nathan Edwards. Testing took place in BYU's large wind tunnel which has a four feet wide by two feet tall test section. The force balance used is capable of measuring normal and axial forces and moments in all three axes. Data was taken in 5° increments at a wind speed of approximately 43 ft/s.

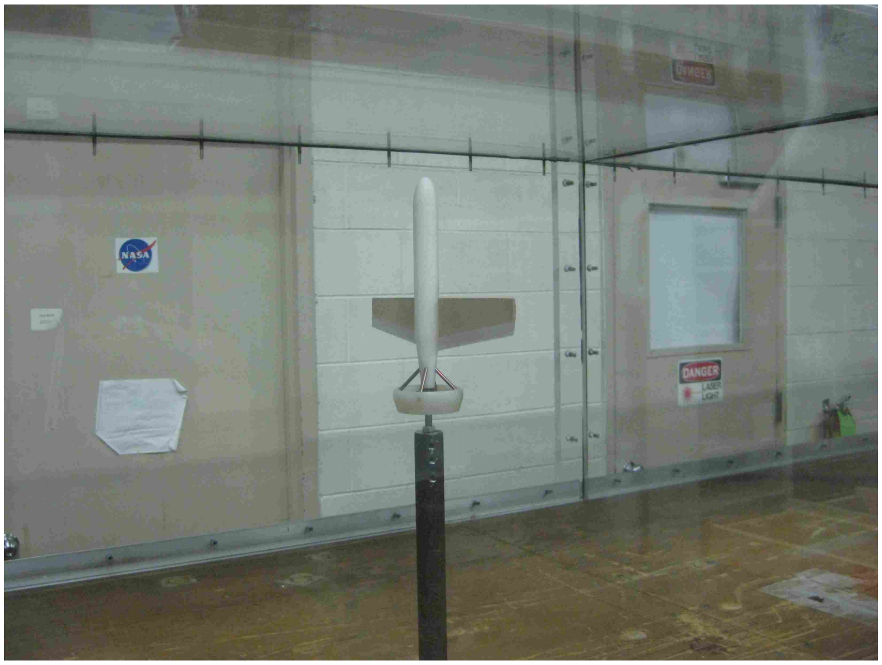


Figure 6.2: Wind tunnel model at $\beta = 0°$. At $\beta = 90°$, the belly of the tailsitter would be facing the wall shown in the photo.

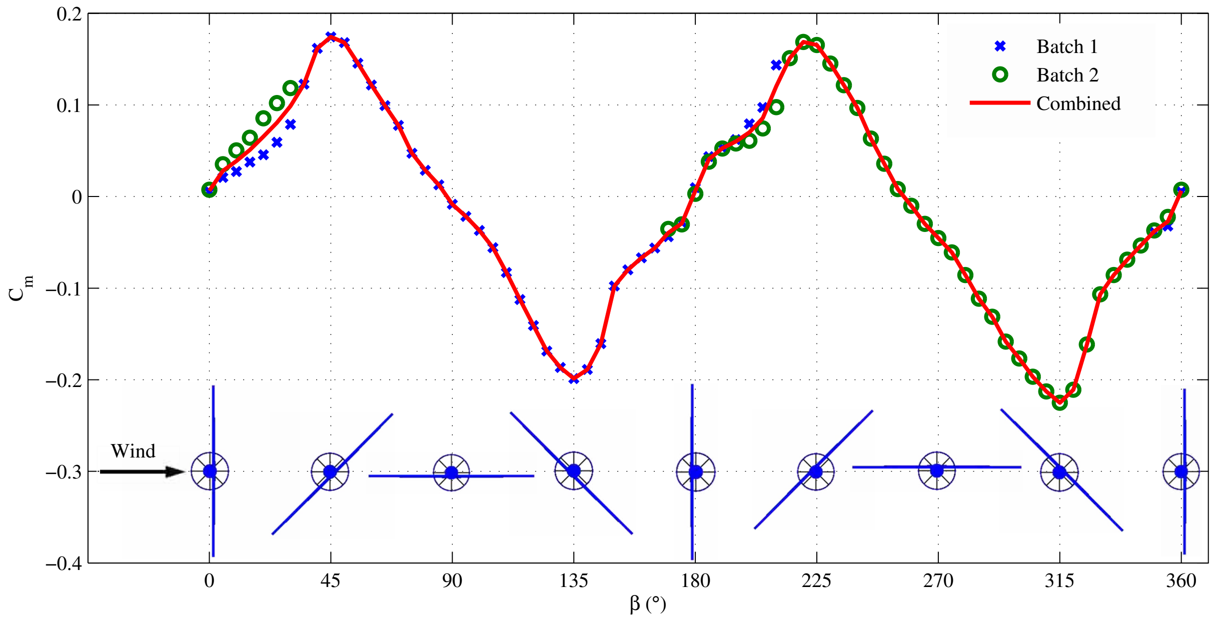


Figure 6.3: Roll moment coefficient data.

Data was collected in two batches. The first batch collected data from a wind incidence angle of 350° to 210° and the second from 170° to 30°. This provided overlap to be used in combining the data sets. The data obtained is shown in Figure 6.3. The measured roll moment and the roll moment coefficient are related by

$$M_w(\beta) = \frac{1}{2}\rho V_w^2 S_w b C_m(\beta). \tag{6.2}$$

The data was non-dimensionalized by solving for $C_m(\beta)$,

$$C_m(\beta) = \frac{2M_w(\beta)}{\rho V_w^2 S_w b}, \tag{6.3}$$

and using the parameters given in Table 6.1. Because of the orientation of the wind with respect to the wing, wing span was used as the characteristic length, rather than wing chord.

The only place the data really looks questionable is right around 270°. At this location the data indicates there is a comparatively large negative moment. Intuitively, the moment should be roughly the same magnitude as at 90° but opposite in direction. The low peak at 315° is interesting, but makes sense if the air is catching the fuselage causing a larger moment to be generated. It is

Table 6.1: Wind tunnel and scale model parameters.

ρ	Air Density	0.0021	slug/ft^3
V_w	Wind speed magnitude	43	ft/s
S_w	Wing area	0.0983	ft^2
b	Wing span	9.6	inch
M_w	Roll moment produced by wind		in-lbs

also interesting that the data is not symmetric about the plot x-axis–slightly larger moments are generated at 135° and 315° than at 45° and 225°.

6.2 Data Curve Fit

The data shown in Figure 6.3 is shaped approximately as a triangle wave giving a nice model to start with. The Fourier series of a triangle wave with period T is

$$f(x) = \frac{8}{\pi^2} \sum_{n=1,3,5\ldots}^{\infty} \frac{(-1)^{(n-1)/2}}{n^2} \sin\left(\frac{2n\pi x}{T}\right). \tag{6.4}$$

For curve fitting, this needs to be generalized to allow an arbitrary amplitude A, phase shift φ and offset d and to allow the angle, period and phase shift to be input as degrees. Doing so gives

$$f(A,\varphi,T,d,x) = A\frac{8}{\pi^2} \sum_{n=1,3,5\ldots}^{\infty} \frac{(-1)^{(n-1)/2}}{n^2} \sin\left(\frac{2n\pi(x+\varphi)}{T}\right) + d. \tag{6.5}$$

In addition to simplifying the equation, truncating the higher order terms has the effect of rounding the peaks of the wave allowing for a better fit. After some experimentation it was determined that using the first three terms works well. This reduces equation 6.5 to

$$f(A,\varphi,T,d,x) = A\frac{8}{\pi^2}\left[\sin\left(\frac{2\pi(x+\varphi)}{T}\right) - \frac{1}{9}\sin\left(\frac{6\pi(x+\varphi)}{T}\right) + \frac{1}{25}\sin\left(\frac{10\pi(x+\varphi)}{T}\right)\right] + d. \tag{6.6}$$

Two curve fits were performed. In both cases the parameters were the amplitude, phase, period and offset; however, in the second fit the period was constrained to be 180°. The model error is given by

$$E(A,\varphi,T,d) = \sum_{\beta=0}^{360} \left[M(\beta) - f(A,\varphi,T,d,\beta)\right]^2 \tag{6.7}$$

where $M(\beta)$ is the measured moment at a given wind incidence angle. E was minimized using a Generalized Reduced Gradient optimization method. The optimal parameters are given in Table 6.2 and the resulting models plotted in Figure 6.4.

Table 6.2: Roll moment curve fit parameters.

	A	T	φ	d	E
No constraint	0.1938	176.1	-1.338	-0.01417	0.01695
T = 180° constraint	0.1910	180.0	2.718	-0.01391	0.02424

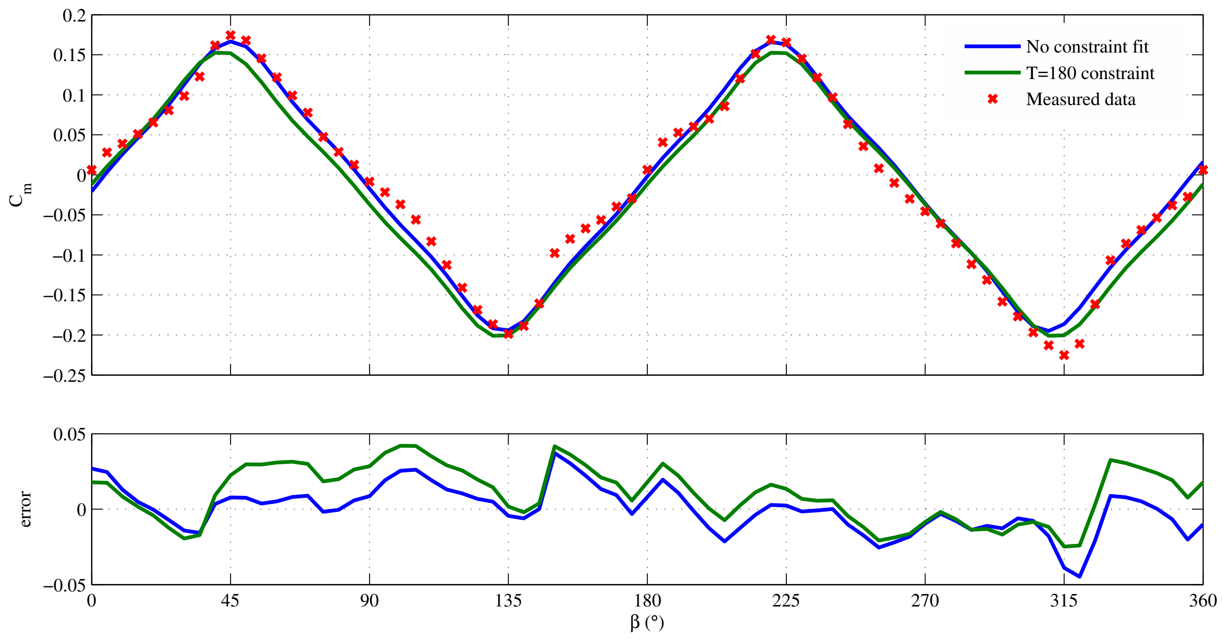


Figure 6.4: Roll moment curve fit.

Relaxing the period constraint allowed for a better overall curve fit but comes at the expense of having a discontinuity at zero. It was hoped that constraining the period would move the moment at the cardinal angles closer to zero, but it actually makes it slightly worse at most of those angles; however, it does remove the discontinuity at zero. It is assumed that constraining the period did not reduce the error at the cardinal angles as intended because the measured data was not perfectly symmetric about zero. Because the constrained model does not have the discontinuity at zero, it is considered the best model and is used in the analysis that follows:

$$C_m(\beta) = 0.1910\frac{8}{\pi^2}\left[\sin\left(\frac{2\pi(\beta+2.718)}{180}\right) - \frac{1}{9}\sin\left(\frac{6\pi(\beta+2.718)}{180}\right) + \frac{1}{25}\sin\left(\frac{10\pi(\beta+2.718)}{180}\right)\right] - 0.01391. \tag{6.8}$$

6.3 Model Definition

For the analysis performed, the full tailsitter dynamics were simplified considerably. Translational accelerations and velocities were assumed zero. The only moment taken into account in addition to that caused by the wind is the roll effort generated by the control vane deflection, δ_v, given by

$$M_{\delta_v} = \frac{1}{2}\rho V_b^2 S_v l_v C_{\ell,\delta_v}\delta_v, \tag{6.9}$$

where a positive roll effort is assumed to produce a positive moment. It is assumed that the roll moment is generated entirely by the control vanes since, in steady hover, the ailerons produce no

Table 6.3: Atmospheric and eV-Bat parameters used in analysis.

C_{ℓ,δ_v}	Change in control vane lift coefficient as a result of vane deflection	5.73	rad^{-1}
J_{xx}	Tailsitter moment of inertia about the x-axis	0.3745	kg m^2
S_v	Control vane surface area	0.128	m^2
S_w	Wing surface area	0.558	m^2
V_b	Air velocity over control vanes	5.0	m/s
V_w	Wind speed magnitude	6.0	m/s^2
b	Wing span	1.93	m
b_{damp}	Air damping coefficient	0.15	N s m
l_v	Control vane moment arm	0.124	m
ρ	Air density	1.069	kg/m^3

moment. The physical parameters used are given in Table 6.3. The equation of motion is

$$\ddot{\phi}_h = \frac{1}{J_{xx}} \left[M_{\delta_v} - M_w(\beta) - b_{damp}\dot{\phi}_h \right], \tag{6.10}$$

where $M_w(\beta)$ is reconstructed from Equation 6.2 using values from Table 6.3.

To determine the stability of this system it is necessary to linearize Equation 6.10. The states are first defined to be $\mathbf{x} \triangleq \begin{bmatrix} \phi_h & \dot{\phi}_h \end{bmatrix}^\top$ and the input to be $u = [\delta_v]$. V_w and χ_w could be considered systems inputs, but under the assumption that they are approximately constant they can be omitted. It is also assumed that the sensors available for feedback are a rate-gyro for $\dot{\phi}_h$ and magnetometer for ϕ_h. The nonlinear state-space model is defined as

$$\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u}) = \begin{bmatrix} \dot{\phi}_h \\ \ddot{\phi}_h \end{bmatrix} = \begin{bmatrix} \dot{\phi}_h \\ \frac{1}{J_{xx}} \left[M_{\delta_v} - M_w(\beta) - b_{damp}\dot{\phi}_h \right] \end{bmatrix}, \tag{6.11}$$

$$\mathbf{y} \triangleq g(\mathbf{x},\mathbf{u}) = \begin{bmatrix} \phi_h \\ \dot{\phi}_h \end{bmatrix}. \tag{6.12}$$

The system is linearized by computing the Jacobians of $f(\mathbf{x},\mathbf{u})$ and $g(\mathbf{x},\mathbf{u})$ and evaluating them at a desired equilibrium point or

$$\mathbf{A} = \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x},\mathbf{u}) \bigg|_{\mathbf{x}_{eq},\mathbf{u}_{eq}}, \tag{6.13}$$

$$\mathbf{B} = \frac{\partial}{\partial \mathbf{u}} f(\mathbf{x},\mathbf{u}) \bigg|_{\mathbf{x}_{eq},\mathbf{u}_{eq}}, \tag{6.14}$$

$$\mathbf{C} = \frac{\partial}{\partial \mathbf{x}} g(\mathbf{x},\mathbf{u}) \bigg|_{\mathbf{x}_{eq},\mathbf{u}_{eq}}. \tag{6.15}$$

The Jacobian of $f(\mathbf{x},\mathbf{u})$ is

$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x},\mathbf{u}) = \begin{bmatrix} 0 & 1 \\ \frac{\partial \ddot{\phi}_h}{\partial \phi_h} & -\frac{b_{damp}}{J_{xx}} \end{bmatrix} \Bigg|_{\mathbf{x}_{eq},\mathbf{u}_{eq}}, \tag{6.16}$$

where

$$\frac{\partial \ddot{\phi}_h}{\partial \phi_h} = -\frac{\rho V_w^2 S_w b}{2 J_{xx}} \left(\frac{\partial C_m(\beta)}{\partial \phi_h} \right), \tag{6.17}$$

and

$$\frac{\partial C_m(\beta)}{\partial \phi_h} = A \frac{16}{\pi T} \left[\cos\left(\frac{2\pi(\beta + \varphi)}{T} \right) - \frac{1}{3} \cos\left(\frac{6\pi(\beta + \varphi)}{T} \right) + \frac{1}{5} \cos\left(\frac{10\pi(\beta + \varphi)}{T} \right) \right]. \tag{6.18}$$

Equilibrium points of β =0° and β =90° were chosen for analysis. At 0°, the system matrices are

$$\mathbf{A}_0 = \begin{bmatrix} 0 & 1 \\ -0.2552 & -0.4005 \end{bmatrix} \qquad \mathbf{B}_0 = \begin{bmatrix} 0 \\ 3.244 \end{bmatrix}. \tag{6.19}$$

At 90°, they are

$$\mathbf{A}_{90} = \begin{bmatrix} 0 & 1 \\ 0.2552 & -0.4005 \end{bmatrix} \qquad \mathbf{B}_{90} = \begin{bmatrix} 0 \\ 3.244 \end{bmatrix}. \tag{6.20}$$

In both cases the output matrix **C** is the identity matrix.

Intuitively, the derivative of a triangle wave is a square wave, so we would expect $\frac{\partial C_m(\beta)}{\partial \phi_h}$ to have that basic shape. Equation 6.18 appears to have this form. The derivative of a perfect triangle wave does not exist at the wave peaks which is indicated by the edge of the square wave. Since the peaks of the wave in the curve fit were rounded, the derivative does exist at these points; however, its slope is close to being vertical. This near-discontinuity is shown in Figure 6.5 and the analysis shows that this is the dividing line between the stable and unstable modes.

6.4 Stability Analysis

Several tests of stability were performed. First, Lyapunov, or internal stability, was examined. Following that, controllability, stabilizability, observability and detectability are determined for the chosen equilibrium points.

6.4.1 Lyapunov Stability

Lyapunov stability is concerned with the stability of the solution to a system's dynamic equations around a point of equilibrium. A system is stable in the Lyapunov sense if all solutions to the system's dynamic equations that start sufficiently close to an equilibrium point converge to that equilibrium point. It is most easily determined by the eigenvalue test. This test states that in order to be stable, the real part of all eigenvalues of **A** must be negative.

The characteristic equation of **A** is given by $\det(\lambda \mathbf{I} - \mathbf{A}) = \lambda^2 + \frac{b_{damp}}{J_{xx}}\lambda - \frac{\partial \ddot{\phi}_h}{\partial \phi_h}$. Since this is a second order system, the quadratic formula is useful in determining the eigenvalues

$$\lambda = \frac{-\frac{b_{damp}}{J_{xx}} \pm \sqrt{\left(\frac{b_{damp}}{J_{xx}}\right)^2 + 4\frac{\partial \ddot{\phi}_h}{\partial \phi_h}}}{2}. \tag{6.21}$$

If $\frac{\partial \ddot{\phi}_h}{\partial \phi_h} = 0$, then the eigenvalues will be $\lambda = \begin{bmatrix} 0 & -\frac{b_{damp}}{J_{xx}} \end{bmatrix}$. From this, one can see that $\frac{\partial \ddot{\phi}_h}{\partial \phi_h} < 0$ is a necessary and sufficient condition for the eigenvalues to be in the open left-half plane and, hence, the system will be stable in the Lyapunov sense if this condition is satisfied. From Equation 6.17, it can be seen that J_{xx}, ρ, V_a^2, S_w, and b are always positive and only scale the amplitude of $\frac{\partial \ddot{\phi}_h}{\partial \phi_h}$, but do not change its sign. From this we can see that whenever $\frac{\partial C_m(\beta)}{\partial \phi_h}$ is positive, $\frac{\partial \ddot{\phi}_h}{\partial \phi_h}$ is negative. This leads to the more general condition of stability of $\frac{\partial C_m(\beta)}{\partial \phi_h} > 0$. The simplest way to show when this condition is met is to plot $\frac{\partial C_m(\beta)}{\partial \phi_h}$ for several values of β. This function is shown in Figure 6.5.

Several insights can be gained from Figure 6.5. First and foremost, changing β so that the wind blows close to parallel to the wings drives $\frac{\partial C_m(\beta)}{\partial \phi_h}$ to be negative, meaning there is an eigenvalue in the right-half plane. This means that as long as wind blows approximately perpendicular to the wings, the system will be internally stable. This makes sense–if the vehicle is at equilibrium with the wind blowing normal to its belly ($\beta = 0°$), when perturbed, it tends to return to $\beta = 0°$. When the vehicle is at equilibrium around $\beta = 90°$, perturbations will cause it to move away from $\beta = 90°$ to a stable equilibrium at either $\beta = 0°$ or $\beta = 180°$. Table 6.4 gives the eigenvalues of **A** at various equilibrium values of β.

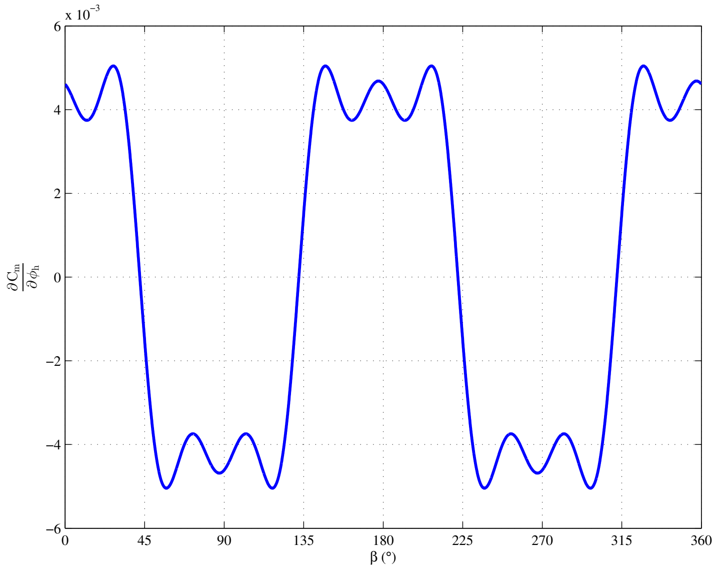


Figure 6.5: Roll moment derivative. All eigenvalues of the linearized system in Equation 6.16 have negative real parts whenever the roll moment derivative shown here is positive. The system eigenvalues are positive from approximately $43° \leq \beta \leq 133°$ and $223° \leq \beta \leq 313°$, or in other words, as the wind becomes parallel with the wings. The oscillations around the peaks are likely an artifact of the curve fit and truncation of the higher order terms.

Table 6.4: Eigenvalues of **A** at various trim points.

$\beta(°)$	Eigenvalues (**A**)	Closed loop eigenvalues (**A − BK**)
0	$-0.2003 \pm 0.4637j$	−0.7074, −32.43
42	−0.02341, −0.3771	−0.7072, −32.44
43	0.04985, −0.4504	−0.7072, −32.44
90	0.3431, −0.7437	−0.7071, −32.44

6.4.2 Controllability

Controllability is defined as the ability of the system to take any initial state to zero in finite time. The rank test was used to determine this. This test states that a system is controllable if and only if the controllability matrix, $\mathcal{C} = [\mathbf{B}\ \mathbf{AB}]$, is full rank.

$$\mathcal{C}_0 = \begin{bmatrix} 0 & 3.244 \\ 3.244 & -1.299 \end{bmatrix}, \qquad \mathcal{C}_{90} = \begin{bmatrix} 0 & 3.244 \\ 3.244 & -1.299 \end{bmatrix} \tag{6.22}$$

The controllability matrix is the same at both equilibrium positions and in both cases the rank is 2. The system as modeled is controllable at both equilibrium points.

6.4.3 Observability

A system is observable if and only if every non-zero initial state results in a non-zero output. To determine this the rank test was also used. Similar to controllability, a system is observable if and only if its observability matrix, $\mathcal{O} = [\mathbf{C}\ \mathbf{CA}]^\top$, is full rank.

$$\mathcal{O}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ -0.2552 & -0.4005 \end{bmatrix}, \qquad \mathcal{O}_{90} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0.2552 & -0.4005 \end{bmatrix} \tag{6.23}$$

The observability matrix in both cases is full rank. This system is observable.

6.4.4 Stabilizability

A system is stabilizable if all uncontrollable components of it are asymptotically stable. Since neither system has any uncontrollable components, they are both stabilizable.

6.4.5 Detectability

A system is detectable if all unobservable states are asymptotically stable. Since both systems are observable, they are detectable.

6.5 Conclusion

A simplified dynamic model of the tailsitter in hover was created from first principles and wind tunnel data. This model was analyzed for stability and other system characteristics were discussed. The analysis mathematically supports the observation that the tailsitter tends to weather vane so that either its belly or its back is normal to the wind while hovering.

CHAPTER 7. CONCLUSIONS AND RECOMMENDATIONS

Although a tailsitter aircraft poses unique autopilot control challenges, solving them is well worth the effort. This platform has the potential to greatly simplify the use of UAVs for end users by eliminating the need for special launch and recovery equipment while not sacrificing range or mission duration.

7.1 Conclusions

The primary contribution of this work is to automate the hover control of a commercially viable tailsitter UAV. This includes developing and testing several methods to incorporate a magnetometer into autopilot estimation algorithms that are fully functional at all attitudes. Of the methods given, the pure quaternion and MEKF were shown to perform the best.

Another significant contribution is the improvement and validation of the resolve tilt-twist method for calculating attitude error. It has been shown that this method of computing attitude error works better for tailsitter aircraft than using conventional quaternion error. Position and altitude control have also been demonstrated. Simulation and hardware testing has taken place to gain confidence in the effectiveness and robustness of implemented algorithms.

7.2 Future Work

While this work has covered basic hovering functionality there is still much that can be done to further this research including:

- **Improve MEKF Estimator** - The MEKF presented estimated only the attitude quaternion and made assumptions on how GPS could be used to correct accelerometer measurements and that the rate-gyro sensor biases were already known. It would be useful to relax these assumptions and to develop it into a more comprehensive state estimation solution estimating among other things position, velocities and rate-gyro biases.

- **Implementation of Outer Loop Optimal Control** - The outer-loop control inplemented uses PID control. It would be interesting and possibly beneficial to see if using optimal control in the outer loops would increase controller performance and robustness.
- **Algorithm Integration** - The work performed here was somewhat exclusive to hover control; however, it was done with the goal of being integrated into a more complete flight control system that encompasses transitions and level flight. The obvious next big step is to actually implement that more complete flight control system in hardware, so that the tailsitter is fully autonomous in all flight regimes.

REFERENCES

[1] R. H. Stone and K. C. Wong, “Preliminary Design of a Tandem-Wing Tail-Sitter UAV Using Multi-Disciplinary Design Optimisation,” in *International Aerospace Congress*, Sydney, Feb. 1997, pp. 707–720. 2, 3

[2] R. H. Stone and G. Clarke, “The T-Wing: A VTOL UAV for Defense and Civilian Applications,” in *UAV Australia Conference*, Melbourne, Feb. 2001. 2, 3

[3] ——, “Optimization of Transition Maneuvers for a Tailsitter Unmanned Air Vehicle,” in *Australian International Aerospace Congress*, Canberra, Mar. 2001. 2, 3

[4] R. H. Stone, “Control Architecture for a Tail-sitter Unmanned Air Vehicle,” *Control Conference, 2004. 5th Asian*, vol. 2, pp. 736–744, 2004. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1426743 2, 3

[5] ——, “The T-wing Tail-sitter Unmanned Air Vehicle: From Design Concept to Research Flight Vehicle,” *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 218, no. 6, pp. 417–433, Jan. 2004. [Online]. Available: http://pig.sagepub.com/lookup/doi/10.1243/0954410042794920 2, 3

[6] W. Green and P. Oh, “A MAV That Flies Like an Airplane and Hovers Like a Helicopter,” in *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.* Monterey, CA: IEEE, 2005, pp. 693–698. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1511063 2, 3

[7] E. Johnson and M. Turbe, “Modeling, Control, and Flight Testing of a Small Ducted Fan Aircraft,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, vol. 29, no. 4. Reston, Virigina: American Institute of Aeronautics and Astronautics, Aug. 2005. [Online]. Available: http://hdl.handle.net/1853/35883 2

[8] A. Frank, J. S. McGrew, M. Valenti, D. Levine, and J. P. How, “Hover, Transition, and Level Flight Control Design for a Single-Propeller Indoor Airplane,” in *AIAA Guidance, Navigation, and Control Conference*, Hilton Head, SC, 2007. 2

[9] N. B. Knoebel, “Adaptive Quaternion Control of a Miniature Tailsitter UAV,” Master’s thesis, Brigham Young University, Dec. 2007. [Online]. Available: http://contentdm.lib.byu.edu/cdm/ref/collection/ETD/id/1197 2, 3, 39, 41, 53

[10] S. R. Osborne, “Transitions between Hover and Level Flight for a Tailsitter UAV,” Master’s thesis, Brigham Young University, Dec. 2007. [Online]. Available: http://contentdm.lib.byu.edu/cdm/ref/collection/ETD/id/1445 2, 3

[11] E. N. Johnson, A. Wu, J. C. Neidhoefer, S. K. Kannan, and M. A. Turbe, “Flight-Test Results of Autonomous Airplane Transitions Between Steady-Level and Hovering Flight,” *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 2, pp. 358–370, Mar. 2008. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.29261 2, 3

[12] R. H. Stone, P. Anderson, C. Hutchison, A. Tsai, P. Gibbens, and K. C. Wong, “Flight Testing of the T-Wing Tail-Sitter Unmanned Air Vehicle,” *Journal of Aircraft*, vol. 45, no. 2, pp. 673–685, Mar. 2008. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.32750 2, 3

[13] K. Kita, A. Konno, and M. Uchiyama, “Transitions between Level Flight and Hovering of a Tailsitter Vertical Takeoff and Landing Aerial Robot,” *Advanced Robotics*, vol. 24, no. 5/6, pp. 763–782, 2010. 2, 3

[14] R. Naldi and L. Marconi, “On Robust Transition Maneuvers for a Class of Tail-sitter Vehicles,” in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2010, pp. 358–363. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5717170 2, 3

[15] P. T. Millet, “Vision-Based Precision Landings of a Tailsitter UAV,” Master’s thesis, Brigham Young University, Apr. 2010. [Online]. Available: http://contentdm.lib.byu.edu/cdm/ref/collection/ETD/id/1987 2, 3

[16] T. Matsumoto, K. Kita, R. Suzuki, A. Oosedo, K. Go, Y. Hoshino, A. Konno, and M. Uchiyama, “A Hovering Control Strategy for a Tail-sitter VTOL UAV that Increases Stability Against Large Disturbance,” *2010 IEEE International Conference on Robotics and Automation*, pp. 54–59, May 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5509183 2, 3, 55, 57, 61

[17] J. V. Hogge, “Development of a Miniature VTOL Tailsitter Unmanned Aerial Vehicle,” Master’s thesis, Brigham Young University, Aug. 2008. [Online]. Available: http://contentdm.lib.byu.edu/cdm/ref/collection/ETD/id/1377 2, 3

[18] Y. Jung, S. Cho, and D. H. Shim, “A Comprehensive Flight Control Design and Experiment of a Tail-Sitter UAV,” in *AIAA Guidance, Navigation, and Control (GNC) Conference*. Reston, Virginia: American Institute of Aeronautics and Astronautics, Aug. 2013, pp. 1–23. [Online]. Available: http://arc.aiaa.org/doi/pdf/10.2514/6.2013-4992 3

[19] MLB Company. [Online]. Available: http://spyplanes.com/products-v-bat/ 4

[20] F. L. Markley, “Attitude Error Representations for Kalman Filtering,” *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 2, pp. 311–317, Mar. 2003. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/2.5048 6, 42

[21] W. F. Phillips, *Mechanics of Flight*, 2nd ed. Hoboken, New Jersey: John Wiley & Sons, Inc, 2010. 6, 8, 9, 96, 98

[22] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton, New Jersey: Princeton University Press, 2012. [Online]. Available: http://uavbook.byu.edu/ 6, 18, 27, 35

[23] J. B. Kuipers, *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*. Princeton, New Jersey: Princeton University Press, 1999. 12

[24] World Magnetic Model. [Online]. Available: http://www.ngdc.noaa.gov/geomag/WMM/DoDWMM.shtml 14, 31

[25] M. Argyle, R. Beard, and S. Morris, "The Vertical Bat Tail-sitter: Dynamic Model and Control Architecture," *American Control Conference*, pp. 806–811, 2013. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6579935 18, 52

[26] J. K. Hall, N. B. Knoebel, and T. W. McLain, "Quaternion Attitude Estimation for Miniature Air Vehicles Using a Multiplicative Extended Kalman Filter," *2008 IEEE/ION Position, Location and Navigation Symposium*, pp. 1230–1237, 2008. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4570043 28, 31, 33, 42

[27] S. Belongie. "Rodrigues' Rotation Formula." From *MathWorld*–A Wolfram Web Resource, created by E. W. Weisstein. [Online]. Available: http://mathworld.wolfram.com/RodriguesRotationFormula.html 60

[28] F. L. Markley, "Attitude Determination Using Vector Observations and the Singular Value Decomposition," *The Journal of the Astronautical Sciences*, vol. 36, no. 3, pp. 245–258, 1988. [Online]. Available: http://www.control.auc.dk/~tb/best/aug23-Bak-svdalg.pdf 100

APPENDIX A. ATTITUDE REPRESENTATION CONVERSIONS

This appendix reviews the conversion between each of the attitude representations mentioned in this work.

A.1 Level Flight Euler Angle - Quaternion Conversion

The conversion between the standard 3-2-1 Euler angles and the quaternion is well known and derivations are given in [21]. For level-flight Euler angles to quaternion, the relationship is

$$\begin{bmatrix} \eta_0 \\ \eta_x \\ \eta_y \\ \eta_z \end{bmatrix} = \begin{bmatrix} \mathrm{c}_{\frac{\phi_\ell}{2}} \mathrm{c}_{\frac{\theta_\ell}{2}} \mathrm{c}_{\frac{\psi_\ell}{2}} + \mathrm{s}_{\frac{\phi_\ell}{2}} \mathrm{s}_{\frac{\theta_\ell}{2}} \mathrm{s}_{\frac{\psi_\ell}{2}} \\ \mathrm{s}_{\frac{\phi_\ell}{2}} \mathrm{c}_{\frac{\theta_\ell}{2}} \mathrm{c}_{\frac{\psi_\ell}{2}} - \mathrm{c}_{\frac{\phi_\ell}{2}} \mathrm{s}_{\frac{\theta_\ell}{2}} \mathrm{s}_{\frac{\psi_\ell}{2}} \\ \mathrm{c}_{\frac{\phi_\ell}{2}} \mathrm{s}_{\frac{\theta_\ell}{2}} \mathrm{c}_{\frac{\psi_\ell}{2}} + \mathrm{s}_{\frac{\phi_\ell}{2}} \mathrm{c}_{\frac{\theta_\ell}{2}} \mathrm{s}_{\frac{\psi_\ell}{2}} \\ \mathrm{c}_{\frac{\phi_\ell}{2}} \mathrm{c}_{\frac{\theta_\ell}{2}} \mathrm{s}_{\frac{\psi_\ell}{2}} - \mathrm{s}_{\frac{\phi_\ell}{2}} \mathrm{s}_{\frac{\theta_\ell}{2}} \mathrm{c}_{\frac{\psi_\ell}{2}} \end{bmatrix}. \tag{A.1}$$

To convert a quaternion to level-flight Euler angles, the expression is

$$\begin{bmatrix} \phi_\ell \\ \theta_\ell \\ \psi_\ell \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}\left[2\left(\eta_0\eta_x + \eta_y\eta_z\right), \eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2\right] \\ \sin^{-1}\left[2\left(\eta_0\eta_y - \eta_x\eta_z\right)\right] \\ \operatorname{atan2}\left[2\left(\eta_0\eta_z + \eta_x\eta_y\right), \eta_0^2 + \eta_x^2 - \eta_y^2 - \eta_z^2\right] \end{bmatrix}. \tag{A.2}$$

A.2 Hover Flight Euler Angle - Quaternion Conversion

The conversion between quaternions and hover Euler angles is derived by setting Equation 2.9 equal to Equation 2.11

$$\begin{bmatrix} -\mathrm{s}_{\phi_h}\,\mathrm{s}_{\psi_h} - \mathrm{c}_{\phi_h}\,\mathrm{s}_{\theta_h}\,\mathrm{c}_{\psi_h} & \mathrm{c}_{\phi_h}\,\mathrm{s}_{\psi_h} - \mathrm{s}_{\phi_h}\,\mathrm{s}_{\theta_h}\,\mathrm{c}_{\psi_h} & -\mathrm{c}_{\theta_h}\,\mathrm{c}_{\psi_h} \\ \mathrm{c}_{\phi_h}\,\mathrm{s}_{\theta_h}\,\mathrm{s}_{\psi_h} - \mathrm{s}_{\phi_h}\,\mathrm{c}_{\psi_h} & \mathrm{c}_{\phi_h}\,\mathrm{c}_{\psi_h} + \mathrm{s}_{\phi_h}\,\mathrm{s}_{\theta_h}\,\mathrm{s}_{\psi_h} & \mathrm{c}_{\theta_h}\,\mathrm{s}_{\psi_h} \\ \mathrm{c}_{\phi_h}\,\mathrm{c}_{\theta_h} & \mathrm{s}_{\phi_h}\,\mathrm{c}_{\theta_h} & -\mathrm{s}_{\theta_h} \end{bmatrix} = \begin{bmatrix} \eta_0^2 + \eta_x^2 - \eta_y^2 - \eta_z^2 & 2(\eta_x\eta_y + \eta_0\eta_z) & 2(\eta_x\eta_z - \eta_0\eta_y) \\ 2(\eta_x\eta_y - \eta_0\eta_z) & \eta_0^2 - \eta_x^2 + \eta_y^2 - \eta_z^2 & 2(\eta_0\eta_x + \eta_y\eta_z) \\ 2(\eta_0\eta_y + \eta_x\eta_z) & 2(\eta_y\eta_z - \eta_0\eta_x) & \eta_0^2 - \eta_x^2 - \eta_y^2 + \eta_z^2 \end{bmatrix}. \tag{A.3}$$

Elements of the third row and third column of each of these matrices are then used to determine the conversions. For ϕ_h this is

$$\frac{\mathrm{s}_{\phi_h}\,\mathrm{c}_{\theta_h}}{\mathrm{c}_{\phi_h}\,\mathrm{c}_{\theta_h}} = \frac{2(\eta_y\eta_z - \eta_0\eta_x)}{2(\eta_0\eta_y + \eta_x\eta_z)}. \tag{A.4}$$

Solving for ϕ_h then gives

$$\phi_h = \operatorname{atan2}\left[\eta_y\eta_z - \eta_0\eta_x, \eta_0\eta_y + \eta_x\eta_z\right]. \tag{A.5}$$

A similar process is used to solve for θ_h and ψ_h. The resulting relationships are

$$\begin{bmatrix} \phi_h \\ \theta_h \\ \psi_h \end{bmatrix} = \begin{bmatrix} \operatorname{atan2}\left[\eta_y\eta_z - \eta_0\eta_x, \eta_0\eta_y + \eta_x\eta_z\right] \\ \sin^{-1}\left[-\eta_0^2 + \eta_x^2 + \eta_y^2 - \eta_z^2\right] \\ \operatorname{atan2}\left[\eta_0\eta_x + \eta_y\eta_z, \eta_0\eta_y - \eta_x\eta_z\right] \end{bmatrix}. \tag{A.6}$$

To express the hover Euler angles as a single quaternion requires the use of the following intermediate quaternion definitions:

$$\eta_v \triangleq \begin{bmatrix} \frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \end{bmatrix}^\top$$
$$\eta_{\phi_h} \triangleq \begin{bmatrix} \cos\left(\frac{\phi_h}{2}\right) & \sin\left(-\frac{\phi_h}{2}\right) & 0 & 0 \end{bmatrix}^\top$$

$$\eta_{\theta_h} \triangleq \begin{bmatrix}\cos\left(\frac{\theta_h}{2}\right) & 0 & \sin\left(\frac{\theta_h}{2}\right) & 0\end{bmatrix}^\top$$
$$\eta_{\psi_h} \triangleq \begin{bmatrix}\cos\left(\frac{\psi_h}{2}\right) & 0 & 0 & \sin\left(\frac{\psi_h}{2}\right)\end{bmatrix}^\top .$$

Note that these are the quaternion equivalents of the four intermediate matrices used in equation 2.9. They are combined using composition by

$$\eta = \eta_{\psi_h} \otimes \left(\eta_{\theta_h} \otimes \left(\eta_{\phi_h} \otimes \eta_v\right)\right). \tag{A.7}$$

Simplifying the resulting vector gives

$$\eta = \frac{\sqrt{2}}{2}\begin{bmatrix}(\mathrm{c}_{\frac{\theta_h}{2}} - \mathrm{s}_{\frac{\theta_h}{2}})(\mathrm{c}_{\frac{\phi_h}{2}}\,\mathrm{c}_{\frac{\psi_h}{2}} - \mathrm{s}_{\frac{\phi_h}{2}}\,\mathrm{s}_{\frac{\psi_h}{2}}) \\ (\mathrm{c}_{\frac{\theta_h}{2}} + \mathrm{s}_{\frac{\theta_h}{2}})(\mathrm{c}_{\frac{\phi_h}{2}}\,\mathrm{s}_{\frac{\psi_h}{2}} - \mathrm{s}_{\frac{\phi_h}{2}}\,\mathrm{c}_{\frac{\psi_h}{2}}) \\ (\mathrm{c}_{\frac{\theta_h}{2}} + \mathrm{s}_{\frac{\theta_h}{2}})(\mathrm{c}_{\frac{\phi_h}{2}}\,\mathrm{c}_{\frac{\psi_h}{2}} + \mathrm{s}_{\frac{\phi_h}{2}}\,\mathrm{s}_{\frac{\psi_h}{2}}) \\ (\mathrm{c}_{\frac{\theta_h}{2}} - \mathrm{s}_{\frac{\theta_h}{2}})(\mathrm{c}_{\frac{\phi_h}{2}}\,\mathrm{s}_{\frac{\psi_h}{2}} + \mathrm{s}_{\frac{\phi_h}{2}}\,\mathrm{c}_{\frac{\psi_h}{2}})\end{bmatrix}. \tag{A.8}$$

A.3 Rotation Matrix - Quaternion Conversion

Equation 2.11 expresses a quaternion as a rotation matrix. The conversion from rotation matrix to quaternion is given in [21] and is done by first numbering the elements of the rotation matrix as

$$\mathbf{R}_v^b = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33}\end{bmatrix}. \tag{A.9}$$

The quaternion element of greatest magnitude is then determined by

$$\eta_{max}^2 \triangleq \max\begin{bmatrix}\eta_0^2, & \eta_x^2, & \eta_y^2, & \eta_z^2\end{bmatrix}. \tag{A.10}$$

This provides a way to avoid division by an element that is close to zero and obtain the most numerically stable solution. The quaternion is computed by one of the following:

If $\eta_{max}^2 = \eta_0^2$,

$$\begin{aligned}
\eta_0 &= \sqrt{1 + r_{11} + r_{22} + r_{33}}/2 \\
\eta_x &= (r_{23} - r_{32})/4\eta_0 \\
\eta_y &= (r_{31} - r_{13})/4\eta_0 \\
\eta_z &= (r_{12} - r_{21})/4\eta_0
\end{aligned} \tag{A.11}$$

If $\eta_{max}^2 = \eta_x^2$,

$$\begin{aligned}
\eta_x &= \sqrt{1 + r_{11} - r_{22} - r_{33}}/2 \\
\eta_0 &= (r_{23} - r_{32})/4\eta_x \\
\eta_y &= (r_{12} + r_{21})/4\eta_x \\
\eta_z &= (r_{31} + r_{13})/4\eta_x
\end{aligned} \tag{A.12}$$

If $\eta_{max}^2 = \eta_y^2$,

$$\begin{aligned}
\eta_y &= \sqrt{1 - r_{11} + r_{22} - r_{33}}/2 \\
\eta_0 &= (r_{31} - r_{13})/4\eta_y \\
\eta_x &= (r_{12} + r_{21})/4\eta_y \\
\eta_z &= (r_{23} + r_{32})/4\eta_y
\end{aligned} \tag{A.13}$$

If $\eta_{max}^2 = \eta_z^2$,

$$\begin{aligned}
\eta_z &= \sqrt{1 - r_{11} - r_{22} + r_{33}}/2 \\
\eta_0 &= (r_{12} - r_{21})/4\eta_z \\
\eta_x &= (r_{31} + r_{13})/4\eta_z \\
\eta_y &= (r_{23} + r_{32})/4\eta_z
\end{aligned} \tag{A.14}$$

APPENDIX B. WAHBA'S PROBLEM

Wahba's problem consists of finding the best rotation (in a least squares sense) that rotates a set of n vector observations expressed in a reference frame to a set of those same vector observations expressed in a body-fixed frame. The cost function minimized to find the optimal rotation is

$$J(\mathbf{R}_r^b) = \frac{1}{2}\sum_{i=1}^{n} a_i|\mathbf{b}_i - \mathbf{R}_r^b\mathbf{r}_i||^2 \tag{B.1}$$

where $\mathbf{r}$ is the set of vectors observed in the reference frame, $\mathbf{b}$ is the set of vectors observed in the body frame and a is a set of weights such that $\sum_{i=1}^{n} a_i = 1$. Markley in [28] derives a solution to this using the Singular Value Decomposition. To incorporate this

1. Compute the matrix $\mathbf{B}$ as

$$\mathbf{B} \equiv \sum_{i=1}^{n} a_i\mathbf{b}_i\mathbf{r}_i^\top . \tag{B.2}$$

2. Decompose $\mathbf{B}$ using the SVD,

$$\mathbf{USV}^\top = \text{svd}\left[\mathbf{B}\right]. \tag{B.3}$$

3. Compute the intermediate matrix

$$\mathbf{D} = \text{diag}\begin{bmatrix}1 & 1 & \det(\mathbf{U})\det(\mathbf{V})\end{bmatrix}. \tag{B.4}$$

4. The optimal rotation is given by

$$\mathbf{R}_{\text{opt}} = \mathbf{UDV}^\top . \tag{B.5}$$