2008-12-22

Attitude Estimation and Maneuvering for Autonomous Obstacle Avoidance by Miniature Air Vehicles

James K. Hall
Brigham Young University - Provo

ATTITUDE ESTIMATION AND MANEUVERING FOR AUTONOMOUS OBSTACLE AVOIDANCE BY MINIATURE AIR VEHICLES

by

James K. Hall

A dissertation submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Mechanical Engineering

Brigham Young University

April 2009

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a dissertation submitted by

James K. Hall

This dissertation has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Timothy W. McLain, Chair
Date	Randal W. Beard
Date	Clark N. Taylor
Date	Mark B. Colton
Date	W. Jerry Bowman

ABSTRACT

ATTITUDE ESTIMATION AND MANEUVERING FOR AUTONOMOUS OBSTACLE AVOIDANCE BY MINIATURE AIR VEHICLES

James K. Hall

Department of Mechanical Engineering

Doctor of Philosophy

Utilizing the Euler-Rodrigues symmetric parameters (attitude quaternion) to describe vehicle orientation, we develop a multiplicative, nonlinear (extended) variation of the Kalman filter (MEKF) to fuse data from low-cost sensors. The sensor suite is comprised of gyroscopes, accelerometers, and a GPS receiver. In contrast to the common approach of using the complete vehicle attitude as the quantities to be estimated, our filter states consist of the three components of an attitude error vector. In parallel with the time update of the attitude error estimate, we utilize the gyroscope measurements for the time propagation of the attitude quaternion. The accelerometer and the GPS sensors are used independently for the measurement update portion of the Kalman filter. For both sensors, a vector arithmetic approach is used to determine the attitude error vector. Following each measurement update, a multiplicative reset operation moves the attitude error information from the filter state into the attitude estimate. This reset operation utilizes quaternion algebra to implicitly maintain the unity-norm constraint. We demonstrate the effectiveness of our attitude estimation

algorithm through flight simulations and flight tests of aggressive maneuvers such as loops and small-radius circles.

We implement an approach to aerobatic maneuvering for miniature air vehicles (MAVs) using time-parameterized attitude trajectory generation and an associated attitude tracking control law. We designed two methodologies, polynomial and trigonometric, for creating functions that specify pitch and roll angles as a function of time. For both approaches, the functions are constrained by the maneuver boundary conditions of aircraft position and velocity. We construct a trajectory tracking feedback control law to regulate aircraft orientation throughout the maneuvers. The trajectory generation algorithm was used to construct several maneuvers and trajectory tracking control law successfully executed the maneuvers in the flight simulator. In addition to the simulation results, MAV flight tests verified the performance of the maneuver generation and control.

To achieve obstacle avoidance maneuvering, the time parameterized trajectories were converted to spatially parameterized paths, which allowed for inertial reference frame position error to be included in the control law feedback loop. We develop a novel method to achieve the spatial parameterization using a prediction and correction approach. Additionally, the first derivative of position of the desired path is modified using a corrective parameter scheme prior to being used in the control. Using the path position error and the corrected derivative, we utilize a unit-norm quaternion framework to implement a proportional-derivative (PD) control law. This control law was demonstrated in simulation and hardware on maneuvers designed specifically to avoid obstacles, namely the Immelmann and the Close-Q, as well as a basic loop.

ACKNOWLEDGMENTS

This document is the culmination of the most challenging experience of my life. I would not have succeeded had it not been for the generous assistance and support I received along the way. It is entirely appropriate to here mention specific individuals.

The members of my research committee have been instrumental in my success. Dr. McLain did a fine job of providing guidance without dictating or coddling. The energy and creativity of Dr. Beard was inspiring and contributed greatly to the research. Dr. Taylor combines a great mind with a welcoming, personable research approach that are worthy of emulation. I particularly appreciate Dr. Bowman succeeding Dr. Bons in providing the Air Force perspective to the committee.

Among my fellow MAGICCians are very dear friends, some of whom were absolutely essential for my success. The first MAGICC labber I met was Brandon Call, whose amazing computer skills helped salvage my first semester grades and enabled the remainder of my research. Bryce Ready is a great teacher who has helped me better understand computer code, Kalman filtering and economics. Andrew Eldredge and Nate Knoebel provided a beginning and direction for my research. Ryan Holt and Justin Bradley were my traveling buddies and sounding boards for the first two years of my BYU time. My hardware results were made possible through the remarkable skills of Lieutenant Jared Yates. Additionally, the friendship and keen minds of Derek Kingston, Blake Barber, Joe Jackson, Dave Johansen, Neil Johnson, Brett Millar, Andres Rodrigues, Jacob Bishop, and Travis Millett have contributed to my BYU experience.

Throughout the past three years my wife has more than compensated for my absence at home. She has been my constant companion and my tolerant sounding

board, my loyal friend and my number one fan. Without her unwavering love and support, I surely would have failed.

Like the Mormon handcart pioneers who came to know God through their trials, my relationship with God has grown as I have struggled through this experience. Each day He has given the needed light and inspiration and strength. Every morning I pleaded for help, every day he filled my cup, and every evening I poured out my heart in thanksgiving. I now have a more sure knowledge that His patience and intelligence are infinite.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The use of miniature air vehicles[1] (MAVs) for aerial surveillance has increased dramatically in recent years as various technologies such as low-cost sensors and cameras have been developed and integrated. The MAVs flown by the Brigham Young University (BYU) Multi-AGent Intelligent Coordination and Control (MAGICC) lab have been demonstrated in applications such as forest fire perimeter tracking, military tactical reconnaissance, and rural search and rescue where rapid area searches are desired. Many of these missions require the MAVs to operate in urban or mountainous environments in which obstacles can be encountered in the flight path. Thus, much research has been conducted to develop autonomous obstacle avoidance technologies for use while traversing specified search regions.

For autonomous aircraft, the problem of avoiding obstacles has been addressed using several different approaches. In [1], a small, unmanned helicopter was used to autonomously explore an unknown urban environment by generating real-time, planar trajectories to navigate around obstacles. The work reported in [2] demonstrated the ability to generate horizontal paths for an aircraft flying among obstacles, and [3, 4] demonstrated the ability to plan and control planar trajectories to avoid obstacles. The common element in the autonomous obstacle avoidance literature is the extension of ground robot methods to aircraft, thereby imposing an artificial constraint on the maneuver space. By developing the ability to autonomously fly aerobatic maneuvers,

[1]We define a miniature air vehicle as having a mass between 0.5 and 4 kilograms and having a wingspan between 0.5 and 2 meters.

we make possible the exploitation of an aircraft's inherent ability to operate in three-dimensions.

For obstacle avoidance maneuvers, there exists a minimum possible turn radius that provides a lower bound on the spatial operating envelope for a given aircraft. In Anderson's [5] discussion of aircraft turning performance, he derives an expression for turn radius as a function of the square of flight velocity, V_∞^2, divided by the load factor, n (defined as the ratio of lift to weight). Hence, for a given aircraft, the minimum turn radius is fixed by the maximum load factor and the minimum flight velocity. This fixed lower-bound on the maneuver envelope can be enhanced by maneuvering in three dimensions rather than just two. For instance, by executing a climbing, three-dimensional turn, the aircraft trades airspeed for altitude, which maintains the total energy of the aircraft while reducing the turn radius. Additionally, the projection of a three dimensional maneuver onto the horizontal plane would be significantly smaller than a corresponding level turn.

1.2 Problem Description

One challenge for autonomous maneuvering aircraft is determining vehicle orientation with respect to an inertial reference frame. Intrinsic to this attitude estimation problem is the mathematical system used to represent attitude. In the aircraft community, the standard attitude representation system uses the body-axis Euler angles, ϕ, θ, and ψ. However, the ability to autonomously fly aerobatic maneuvers requires a system, such as the Euler-Rodrigues symmetric parameters (attitude quaternion), that avoids the mathematical singularities and non-unique attitude representations inherent to the Euler angle system [6].

The problem of aircraft attitude estimation is fundamentally one of fusing data from various sensors in a way that accounts for uncertainty in the measurements and the dynamic model. Although it lacks the guaranteed optimality of the linear Kalman filter, the extended Kalman filter (EKF) can be used to filter various sensors and the dynamic model of a nonlinear system. However, the structure of the attitude quaternion makes it awkward for usage in a standard EKF. Thus, we developed and implemented a multiplicative EKF (MEKF) which used a three-component parameterization of attitude error as the filter state. The attitude quaternion was propagated and updated in parallel with the MEKF.

Another challenge to achieving autonomous aerobatic maneuver avoidance is defining the desired path. When operating in an urban environment, it is conceivable that the aircraft will encounter blind alleys or T-intersections that would result in a crash if the MAV were confined to maneuvering in the horizontal plane. In contrast to helicopters, fixed-wing aircraft have kinematic constraints, specifically airspeed, which must be met when defining aerobatic maneuvers. Combining the airspeed constraints with desired position and orientation in the framework of continuously differentiable functions allows aerobatic obstacle avoidance maneuvers to be specified.

To use aerobatic maneuvers to avoid obstacles, the paths from which the attitude trajectories were generated need to be parameterized in space rather than time so that position error can be calculated and included in the feedback loop. However, including position error in the control law requires knowing the distance from the aircraft to the desired path, which, in turn, requires parameterizing the path as a function of position rather than time. Given the error in position and the error in the first derivative of position allows the development of a proportional-derivative (PD) control law. Building the control law in a unit-norm quaternion framework enables the controller actuation commands to be effected directly in the body reference frame of the aircraft.

To aid in the development of the aircraft attitude estimation algorithm and obstacle avoidance maneuvering control technologies, a Matlab MAV flight simulation was developed. The algorithms developed and refined using this Matlab tool were then imported into the Kestrel autopilot code and further refined in the Aviones flight simulator. Once the algorithms were satisfactorily refined and tested in the simulations, the code can be programmed directly onto the MAV autopilot computer processor [7, 8] for flight validation. Using this development environment, the quaternion MEKF attitude estimation scheme, the trajectory tracking control law, and the quaternion path following controller were developed and tested.

1.3 Related Work

The ability to autonomously fly aerobatic maneuvers to avoid obstacles requires research in three broad categories: 1) MAV attitude estimation, 2) maneuver trajectory generation and tracking, and 3) maneuver path following. The problem of estimating orientation in the three-dimensional world has existed since humans first started exploring the world. Maps, compasses, sextants, and stars were the tools of early explorers to help them determine their heading. For MAVs, the attitude estimation challenge requires fusing the data from several different tools. Our suite of attitude sensors includes low-cost gyroscopes and accelerometers with the information from the GPS and pressure sensors. This challenge is made difficult due to the noise and drift inherent in these sensors, the lack of direct attitude measurements, and the disparate operating frequencies of the various sensors. In addition to the challenge of obtaining a single attitude estimate from multiple sensors, the aerobatic maneuvering vehicle needs a singularity-free attitude representation, such as the Euler-Rodrigues Symmetric Parameters (referred to herein as the attitude quaternion). However, the axis-angle, unit-norm structure of the attitude quaternion creates difficulties for use as the system state for a Kalman filter. One of the seminal works in this arena is [9] where the multiplicative extended Kalman filter (MEKF) was first introduced for spacecraft applications.

The ability to autonomously fly aerobatic maneuvers requires an understanding of the kinematic constraints of the aircraft in question. These constraints are different for helicopters compared to aircraft. For example, aircraft must maintain a minimum airspeed and have a minimum turn radius whereas the ability to hover allows helicopters to maneuver in a completely different spatial flight envelope. For the MAGICC lab fixed-wing MAVs, aerobatic maneuvers must have a continuous, smooth transition from steady, level-flight to the maneuver and another smooth transition from the maneuver back to level flight. These kinematic constraints create maneuver boundary conditions that must be met by the trajectory generation algorithm. Additionally, the trajectory generation algorithm needs to be able to calculate desired MAV attitude as a function of time that does not exceed the dynamic limits of the aircraft for thrust and turning rate. Whereas fifth-order polynomials can be used to fly single-axis maneuvers such as aileron rolls or loops, a more sophisticated algorithm that meets kinematic constraints is required to generate simultaneous pitch and roll angles during a full three-dimensional maneuvers such as barrel rolls. The ground breaking work on aerobatic maneuvering reported in [10] gives great insight into modeling aircraft flight dynamics using a differentially flat system of equations. The challenge of the trajectory tracking control law is to achieve a balance between the roll and pitch attitude to obtain acceptable tracking performance for three-dimensional attitude trajectories. This balancing is necessary due to the pitch and roll coupling during three-dimensional maneuvers; a small error in roll angle can cause the pitch angle to take the aircraft in a very wrong direction.

The foremost challenge of following a curving path in space is determining the point on the path that corresponds to the aircraft's current location and the value of the spatial parameter that corresponds to this point on the path. This point is the minimum distance from the aircraft to the path and the magnitude of this distance indicates the amount of error in the path following; thus, it is of singular importance when regulating to a curved path as was demonstrated in [11]. Calculating this point on the path and the associated distance is particularly difficult when the path crosses itself as when performing a loop maneuver. A related challenge is calculating the derivative of the path as a function of space rather than time. In the trajectory generation algorithm, the derivative was calculated with respect to time; however, for path following the path derivatives must be derived with respect to the spatial parameter found in conjunction with the desired path that is minimum distance from the aircraft location. Finally, the proportional and derivative errors are calculated in the inertial reference frame, transformed into the path and body reference frames, built into separate unit-norm quaternions, and then melded into a single unit-norm, correction quaternion (see [12]).

1.4 Contributions

The combination of singularity-free attitude estimation, maneuver path definition and location, and the path following control law are enabling technologies necessary for autonomous obstacle avoidance maneuvering for MAVs. The specific contributions of this research are the following:

- the development of a multiplicative extended Kalman filter (MEKF) for fusing MAV-specific sensor data;
- the formulation of the methodology for directly computing the attitude error quaternion for use in the MEKF for fixed-wing MAVs;
- the creation of a structured methodology for defining and generating aerobatic maneuver paths using continuous, smooth polynomial and trigonometric functions and boundary conditions for position and velocity;
- the formulation of a prediction-correction approach to determining the path location and parameterization variable associated with the aircraft instantaneous location; and
- the generalization of a quaternion control approach from tracking a fixed position in inertial space to following a three-dimensional path.

As a secondary effect beyond enabling aerobatic maneuvering, the improved accuracy of the aircraft attitude estimation enables better geo-location from the MAV platform and improves altitude hold performance during turns. In addition to flying familiar maneuvers such as the aileron roll, loop and Immelmann[2], a completely new maneuver, dubbed the Close-Q, specific to obstacle avoidance in urban terrain has been defined and flight tested. Additionally, the Matlab MAV flight simulation tool was adapted to aid in the initial development and testing of new control laws.

[2]Named for Max Immelmann, German pilot in World War I.

1.5 Outline

The research presented herein can be categorized under three general topics: attitude estimation, time-parameterized maneuver trajectory generation and tracking, and path following maneuver control. However, before discussing the research, Chapter 2 will present the various tools we used to develop and test MAV control laws. The topic of attitude estimation will be presented in Chapter 3. The construction of maneuver trajectories based on boundary conditions will be discussed in Chapter 4. We conclude with a discussion in Chapter 5 of the conversion of the trajectory to a path and the quaternion path following control law. Some conclusions and a discussion of possible future research are presented in Chapter 6. Disclaimer: The views expressed in this document are those of the author and do not necessarily reflect the official policy or position of the Air Force, the Department of Defense, or the U.S. Government.

Chapter 2

MAV Development Tools

The attitude estimation and autonomous maneuvering research presented herein was conducted in the BYU MAGICC lab using various flight simulation software environments as well as flight test hardware. These tools have been created and refined by MAGICC lab researchers during the past several years.

2.1 Matlab Flight Simulation

The first tool is a Matlab Simulink environment that uses the full six degree-of-freedom (6-DOF) model of an aircraft from [13]. This flight simulation environment includes autopilot and plotting functions (see Figure 2.1), which enables the rapid development and testing of new concepts for controlling autonomous aircraft.

To facilitate autonomous aerobatic maneuvering research, we modified the first-order differential equations for attitude from the Euler angle model to the attitude quaternion kinematic model. With this change, the number of equations in the complete aircraft dynamic model increase from twelve to thirteen, but the attitude estimation is free of singularities. Additionally, the drag polar information from [14] was used to update the drag model parameters.

One of the advantages of the Matlab development tool is the ease with which various flight parameters can be plotted for further review. The sample snapshot of a single data run for an aggressive aerobatic maneuver is shown in Figure 2.2. The legend for each of the various outputs is not included so as to conserve space for the pertinent information.

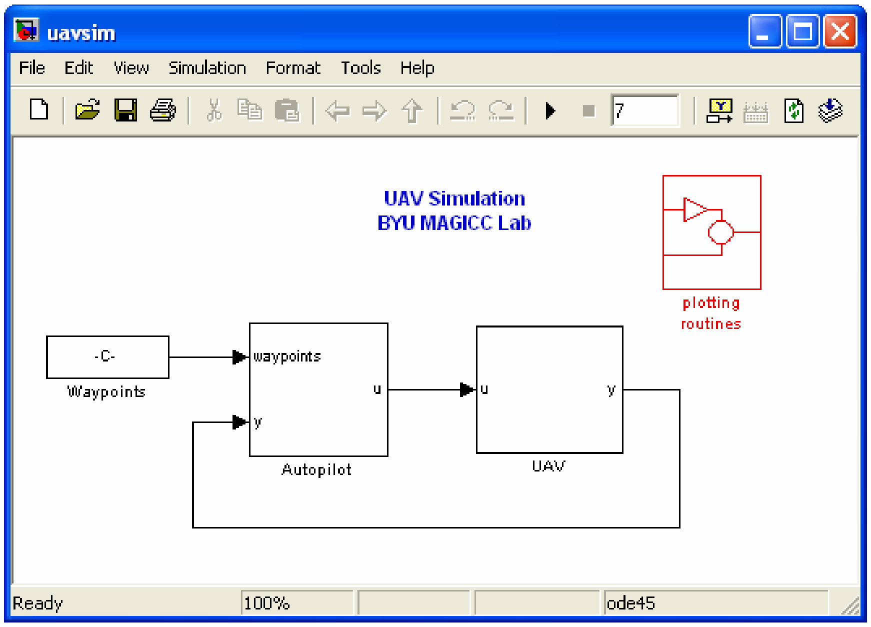


Figure 2.1: Matlab Simulink UAV and Autopilot. The image shows the simulation interface for Matlab Simulink model of the autopilot and generic unmanned air vehicle (UAV). This environment provides a means to rapidly test new aircraft control concepts.

Once a concept has been developed and tested in the Matlab environment, translating the computer code into the appropriate autopilot library is quite straight forward.

2.2 Autonomous Flight Software Tools

The Virtual Cockpit is a software program that was developed at BYU to interface with MAVs in simulation and flight. This interaction can include tuning controller gains, commanding waypoints with the associated altitude and airspeed, and monitoring aircraft status. The Virtual Cockpit has the ability to communicate with the MAV in flight or with an emulated MAV in a flight simulator program.

The Virtual Cockpit presents vital telemetry data of the MAV on a heads-up display. It also displays a geo-referenced terrain map, on which the operator can point-and-click, defining a flight path for the MAV to follow. The current position of

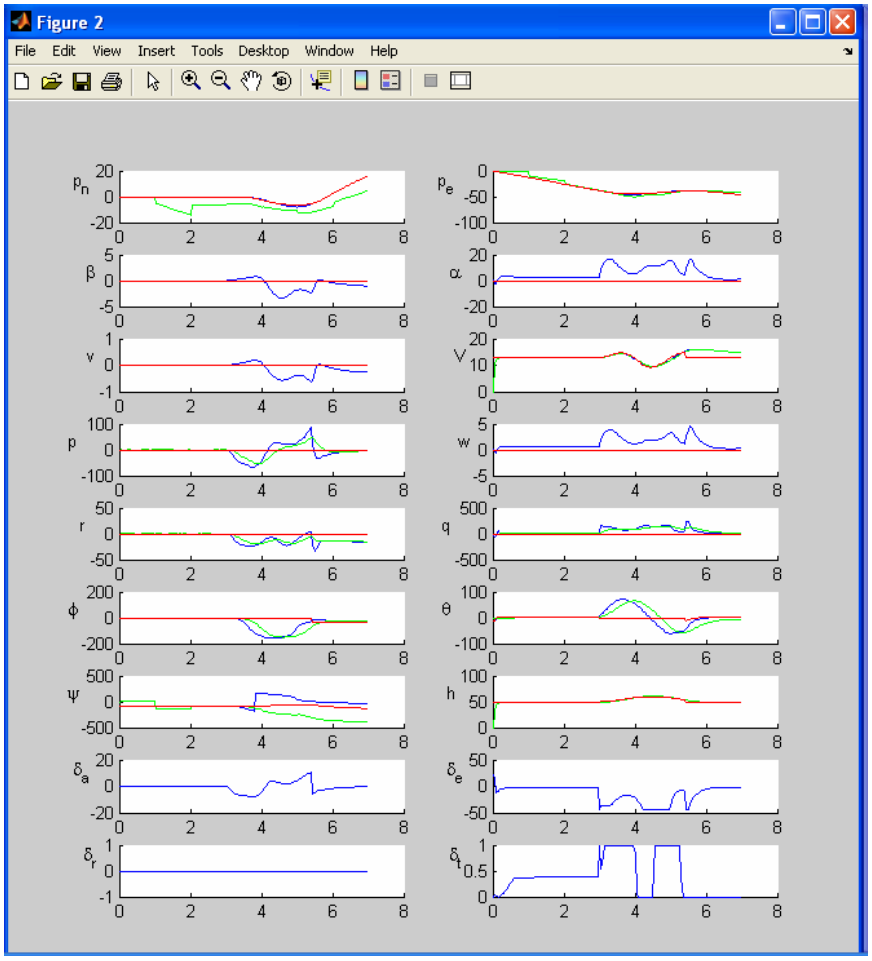
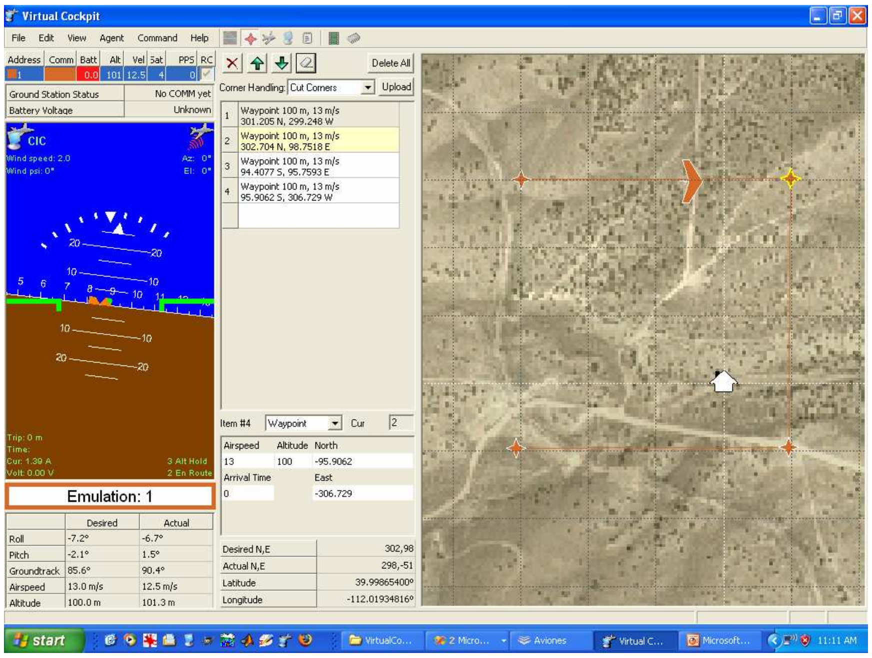


Figure 2.2: Output from the Matlab UAV Simulation. The output data from a simulated Close-Q aerobatic maneuver. Of particular interest in this data set was the deflections of the ailerons, elevators and throttle (δ_a, δ_e, and δ_t).

the MAV is also displayed on the map allowing the operator to monitor the progress of the MAV along the path. A screen shot of these displays is shown in Figure 2.3.

The Virtual Cockpit contains a framework for accessing autopilot variables, which can be used for in-flight parameter and gain tuning. Additional features include the ability to stream video, record important telemetry data, and coordinate with multiple agents. The ability of Virtual Cockpit to interface with both the simulator and the actual aircraft allows for smoother development and testing of control algorithms.

The simulation environment used in conjunction with the Virtual Cockpit is called Aviones. It was developed at BYU and the open source code is available for



Address	Comm	Batt	Alt	Vel	Sat	PPS	RC
1		0.0	101	12.5	4	0	✓

Ground Station Status	No COMM yet
Battery Voltage	Unknown

Wind speed: 2.0
Wind psi: 0°
Az: 0°
El: 0°
Trip: 0 m
Time:
Cur: 1.39 A
Volt: 0.00 V
3 Alt Hold
2 En Route

Emulation: 1

	Desired	Actual
Roll	-7.2°	-6.7°
Pitch	-2.1°	1.5°
Groundtrack	85.6°	90.4°
Airspeed	13.0 m/s	12.5 m/s
Altitude	100.0 m	101.3 m

Corner Handling: Cut Corners

1	Waypoint 100 m, 13 m/s 301.205 N, 299.248 W
2	Waypoint 100 m, 13 m/s 302.704 N, 98.7518 E
3	Waypoint 100 m, 13 m/s 94.4077 S, 95.7593 E
4	Waypoint 100 m, 13 m/s 95.9062 S, 306.729 W

Item #4 Waypoint Cur 2

Airspeed	Altitude	North
13	100	-95.9062
Arrival Time		East
0		-306.729

Desired N,E	302,98
Actual N,E	298,-51
Latitude	39.99865400°
Longitude	-112.01934816°

Figure 2.3: The Virtual Cockpit Map Page. A snapshot of the Virtual Cockpit map page shows an overhead view of the waypoint path over the local map, along with a list of waypoints and the artificial horizon of the heads-up display.

download from SourceForge: http://sourceforge.net/projects/aviones. The engine that drives the Aviones flight simulation are the thirteen, first-order, nonlinear differential equations for the motion of an aircraft. These equations allow for six degrees of freedom for the aircraft, including translation in the three orthogonal directions and rotations about the three axes in inertial space. These thirteen equations can be divided into four sets: 1) force, 2) moment, 3) navigation, and 4) attitude kinematics. We define the key variables in vector format, where $\begin{bmatrix} u & v & w \end{bmatrix}^T$ is the vector of velocities in the body-frame directions $\begin{bmatrix} \mathbf{X}_b & \mathbf{Y}_b & \mathbf{Z}_b \end{bmatrix}^T$, respectively. The vector $\begin{bmatrix} p & q & r \end{bmatrix}^T$ contains the angular rates about the body-frame axes, the vector $\begin{bmatrix} e_0 & e_x & e_y & e_z \end{bmatrix}^T$ is made up of the four components of the attitude quaternion, the components of the vector $\begin{bmatrix} F_x & F_y & F_z \end{bmatrix}^T$ are the summation of the aerodynamic and thrust forces in the body-frame directions, g is the local acceleration due to gravity, and finally m is the aircraft mass. Using this nomenclature, the three force equations can be written as

$$\begin{aligned} \dot{u} &= rv - qw + g\left(2e_xe_z - 2e_0e_y\right) + \frac{F_x}{m} \\ \dot{v} &= pw - ru + g\left(2e_ye_z + 2e_0e_x\right) + \frac{F_y}{m} \\ \dot{w} &= qu - pv + g\left(e_0^2 - e_x^2 - e_y^2 + e_z^2\right) + \frac{F_z}{m}. \end{aligned} \tag{2.1}$$

The three moment equations use the vector of torques $\begin{bmatrix} T_x & T_y & T_z \end{bmatrix}^T$ about the three body-frame axes and the associated moments of inertia $\begin{bmatrix} J_x & J_y & J_z & J_{xz} \end{bmatrix}^T$ in the calculation of the change in angular rates, as

$$\begin{aligned} \dot{p} &= \frac{J_{xz}\left(J_x - J_y + J_z\right)pq - \left(j_z^2 - J_zJ_y + J_{xz}^2\right)qr + J_zT_x + J_{xz}T_z}{J_xJ_z - J_{xz}^2} \\ \dot{q} &= \frac{\left(J_z - J_x\right)pr - J_{xz}p^2 + J_{xz}r^2 + T_y}{J_y} \\ \dot{r} &= \frac{\left(J_x^2 - J_xJ_y + J_{xz}^2\right)pq - \left(J_{xz}J_x - J_{xz}J_y + J_{xz}J_z\right)qr + J_{xz}T_x + J_xT_z}{J_xJ_z - J_{xz}^2}. \end{aligned} \tag{2.2}$$

The translation and rotation information calculated in the force and moment equations can be used to propagate the aircraft position using the navigation equations

$$\begin{aligned} \dot{p}_N &= u\left(e_0^2 + e_x^2 - e_y^2 - e_z^2\right) + v\left(2e_ye_x - 2e_0e_z\right) + w\left(2e_xe_z - 2e_0e_y\right) + V_{\text{wind}_N} \\ \dot{p}_E &= u\left(2e_ye_x + 2e_0e_z\right) + v\left(e_0^2 - e_x^2 + e_y^2 - e_z^2\right) + \left(2e_ye_z - 2e_0e_x\right) + V_{\text{wind}_E} \\ \dot{p}_D &= u\left(2e_xe_z - 2e_0e_y\right) + v\left(2e_ye_z + 2e_0e_x\right) + w\left(e_0^2 - e_x^2 - e_y^2 + e_z^2\right), \end{aligned} \tag{2.3}$$

where V_{wind_N} and V_{wind_E} are the components of the modeled wind. Finally, the angular rates can be used to determine the aircraft spatial orientation from the kinematic equations. However, for the purposes of the aerobatic maneuvering research, we changed the equations of motion in Aviones to propagate the attitude quaternion parameters rather than Euler angles. The four attitude-quaternion equations can be written as

$$\dot{e}_0 = \frac{1}{2}\left(-e_xp - e_yq - e_zr\right)$$

$$
\begin{aligned}
\dot{e}_x &= \frac{1}{2}\left(e_0 p - e_z q + e_y r\right) \\
\dot{e}_y &= \frac{1}{2}\left(e_z p + e_0 q - e_x r\right) \\
\dot{e}_z &= \frac{1}{2}\left(-e_y p + e_x q + e_0 r\right).
\end{aligned}
\tag{2.4}
$$

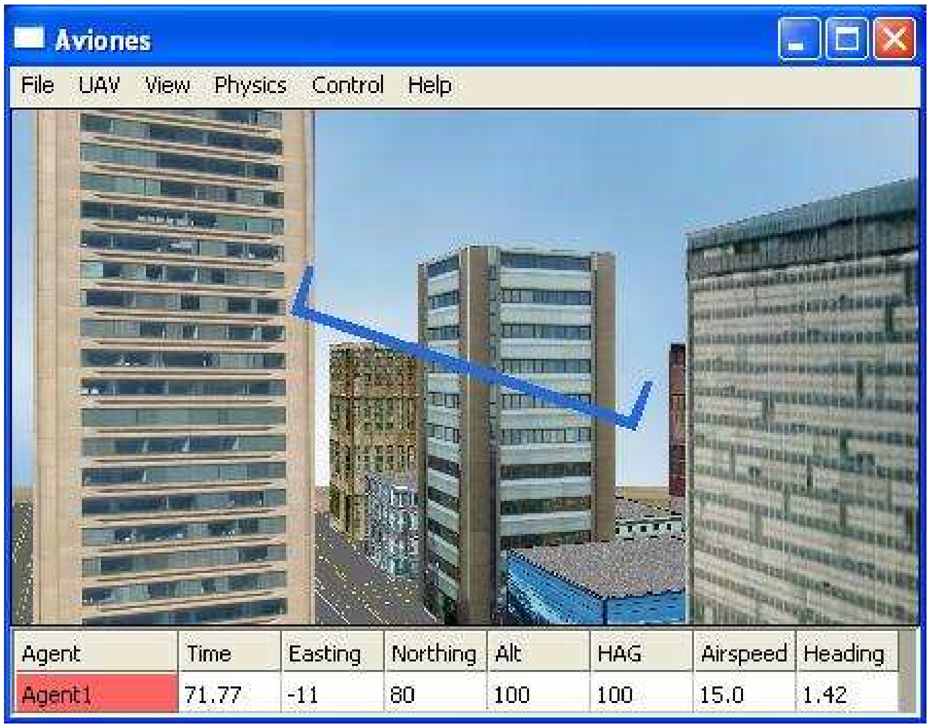
In Aviones, this set of thirteen differential equations are propagated forward in time using a fourth-order Runge-Kutta numerical integration scheme.

Using the motion information from the integration functions, Aviones renders MAVs flying above three-dimensional, geo-referenced terrain maps, and also has the ability to import city files. A screen shot of a MAV flying through a city in Aviones is shown in Figure 2.4. Aviones supports the simulation of multiple UAVs, as well as different types of aircraft with their associated physics models, and sensor models. It also allows alteration of aircraft parameters such as lift and drag coefficients, as well as environmental factors such as gravitational fields and wind.

The distinguishing feature of Aviones is the autopilot emulation framework that allows code written for the autopilot to be compiled as a dynamically linked library (DLL). This DLL is then loaded into the simulated environment for testing. Aviones communicates with the Virtual Cockpit over a TCP/IP socket as though the simulated aircraft were a physical aircraft communicating via radio. The physics engine of Aviones provides state information to the autopilot DLL and replaces the sensors of the actual autopilot with models. The Virtual Cockpit and Aviones tools enable algorithm development in a controlled environment prior to flight testing.

2.3 Flight Hardware

The control algorithms developed in the Matlab Simulink tool and refined in the Aviones simulation environment can be validated in the crucible of flight testing. The primary component of the flight tests is the MAV, which is an expanded polypropylene (EPP) flying wing with carbon fiber rods bonded into the wings for strength and stiffness, as shown in Figure 2.5. The MAVs are battery powered, with thrust coming from a speed controlled, brushless electric motor.



Agent	Time	Easting	Northing	Alt	HAG	Airspeed	Heading
Agent1	71.77	-11	80	100	100	15.0	1.42

Figure 2.4: The Aviones Flight Simulation Environment. This environment can include buildings and roads in its simulated world.

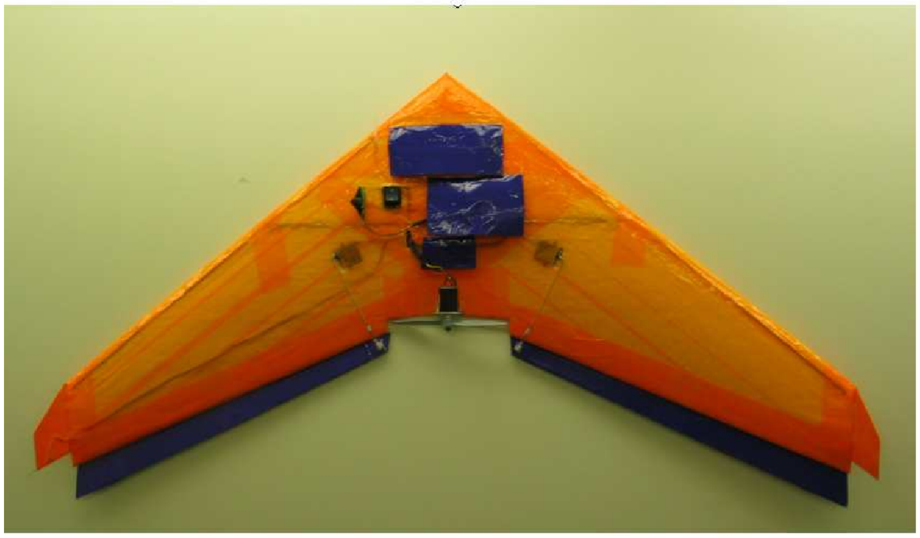


Figure 2.5: The Flight Test MAV. The MAV used in the flight testing of the attitude estimation and aerobatic maneuvering control laws.

The hardware that distinguishes the MAV from a radio controlled aircraft is the autopilot shown in Figure 2.6. The Kestrel Autopilot was developed at BYU and includes a main processor, onboard sensors, and communication ports for external servos, modem, and sensors. The Kestrel Autopilot is equipped with a 29 MHz Rabbit microcontroller with 512K Flash and 512K RAM. The integrated autopilot sensors include gyroscopes, accelerometers, and pressure sensors for measuring altitude and airspeed.

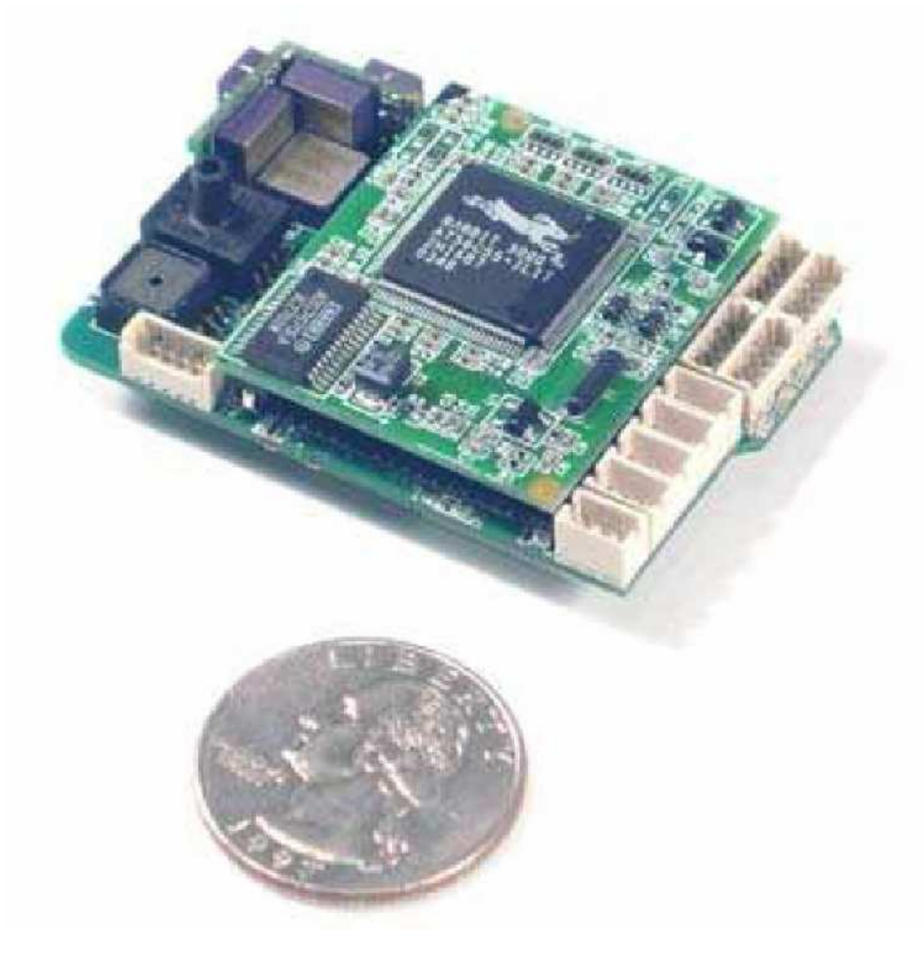


Figure 2.6: The Kestrel Autopilot. Version 2.2 of the autopilot is shown with a quarter for scale purposes.

To provide position and course measurements even during maneuvering, a GPS receiver with a 4 Hz update rate and an active, omnidirectional antenna is used as an additional onboard sensor. The GPS receiver and antenna can be seen in Figure 2.7.

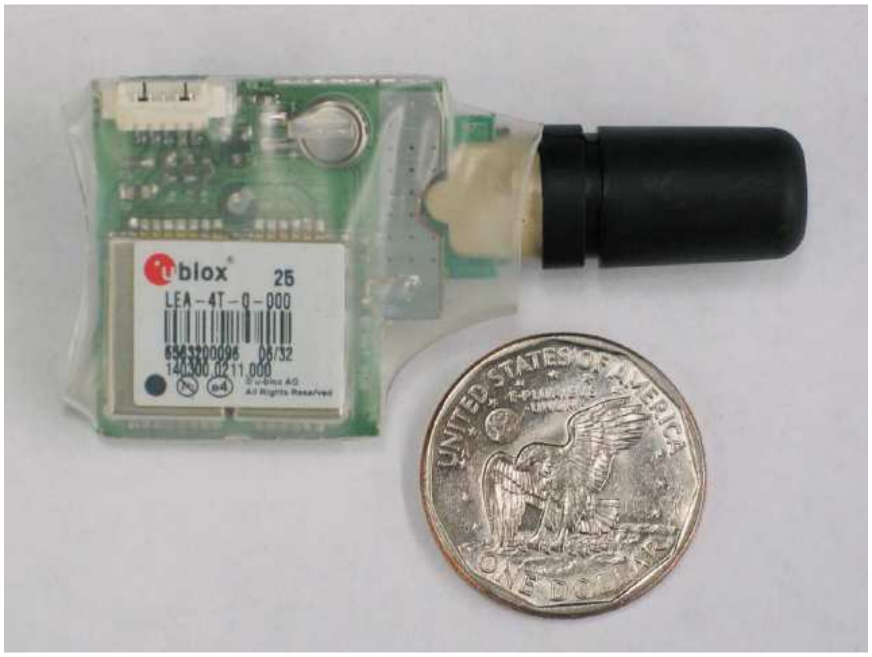


Figure 2.7: The GPS Receiver and Antenna. The MAV obtains position and course information from the GPS receiver.

The communication link from the ground station to the aircraft is accomplished using two 900 MHz wireless modems, one on the aircraft and the other in the communications box attached to the ground station. The communications box is equipped with circuitry and connections for a radio control aircraft transmitter. Thus, a flying MAV can be controlled by the ground station computer running Virtual Cockpit or by a pilot using the radio control transmitter. This arrangement allows human intervention to override the autopilot should a dangerous situation arise.

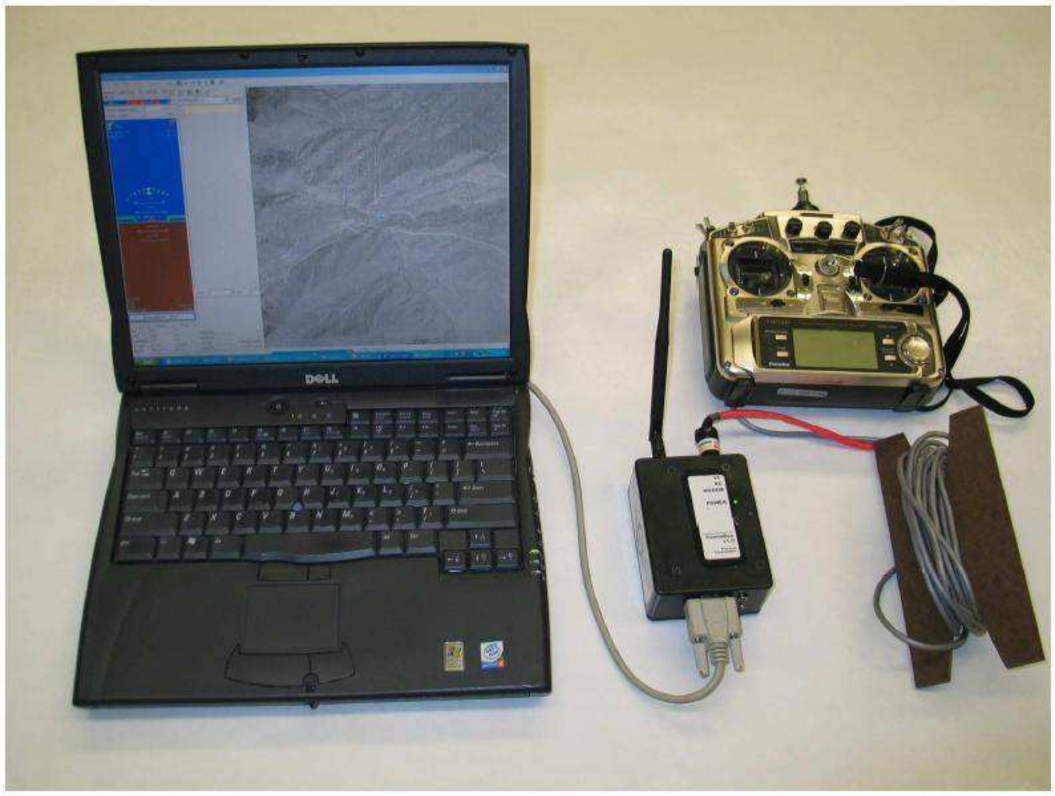


Figure 2.8: MAV Flight Test Ground Station. The laptop computer is running Virtual Cockpit and is connected to the radio control transmitter through a communications box wherein resides the 900 MHz wireless modem.

Table 2.1: MAGICC Lab Aircraft Specifications

CHARACTERISTIC	SPECIFICATION
Processor Speed	29 MHz
Memory	512K flash & 512 RAM
GPS	4Hz update rate
Communication	900 MHz Wireless Modem
Size	1.2 m span, 0.3 m chord
Weight	1.5 kg

2.4 Summary

The BYU MAGICC lab has developed an effective set of tools to develop and demonstrate innovative control technologies for MAVs. The high fidelity of the Matlab and Aviones software environments aids in the refinement of the methodologies prior to flight validation.

Chapter 3

Attitude Estimation

The first step in developing obstacle avoidance maneuvering for MAVs is to create an attitude estimation approach that continues to function regardless of the aircraft attitude. The challenge of estimating the attitude, or orientation, with respect to an inertial reference frame has been a topic of research for many years.

3.1 Background

Many researchers have addressed the aircraft attitude estimation problem [15, 16, 17] by fusing data from various sensors. The current MAGICC lab attitude estimation system [18] uses a complementary filter (see [19]) with the filter gain scaled as a function of the angular rates; this algorithm is referred to as the Variable Gain Observer (VGO). A method of data fusion that uses sensor noise statistics to improve estimation accuracy is the Kalman filter [20, 21, 22], which can be decomposed into two sequential parts: the time update and the measurement update. The MEKF [9, 23, 24, 25], which uses multiplication rather than addition in the measurement update, has been applied to the attitude estimation problem, particularly when the vehicle orientation is expressed using the attitude quaternion [6, 26, 27]. For our attitude estimation algorithm, we developed an implementation of the MEKF that fuses data from a low-cost suite of gyroscopes, accelerometers and GPS.

Although the attitude quaternion provides a singularity-free representation, its unique, mathematical structure creates difficulties for use as the system state for a Kalman filter. One approach is to define attitude error as the filter state vector [28] and propagate the attitude quaternion separately. We utilize the Euler axis/angle parametrization of attitude error for our state vector and propagate our attitude estimate based on output from the gyroscopes and updates from the measured attitude error vector.

The measurement update portion of the MEKF provides a correction to the attitude quaternion propagated using the gyroscopes. The task of determining attitude from vector observations is often referred to as Wahba's problem [29], and requires the measurement of at least two, non-collinear vectors. Many different approaches have been used to solve Wahba's problem as applied to spacecraft attitude estimation [25, 28, 30], and recently a quaternion-based EKF approach was demonstrated for manned aircraft [31]. For our MAV attitude estimation scheme, the two vector observations will be gravity and heading, derived from accelerometer and GPS measurements, respectively. However, the distinct attributes of the GPS and accelerometer measurements creates a need to employ the incremental update methodology [32, 18]. Thus, we developed an intelligent algorithm to incrementally update the attitude estimate using the MEKF, where the filter state vector is the Euler axis and angle parametrization of attitude error and the attitude quaternion is propagated in parallel.

3.2 Attitude Representation

The attitude estimation problem is to determine the orientation of the vehicle body reference frame (subscript b) with respect to an inertial reference frame (subscript i). Although many different attitude representation systems (see [6]) can be used to specify the orientation of a moving vehicle to some inertial reference frame, only two are used in our research: Euler angles and the attitude quaternion.

3.2.1 Euler Angles

In the aeronautical community, Euler angles (ϕ, θ, and, ψ) are the primary attitude representation system used to describe the orientation of the body-frame with respect to the inertial-frame. The Euler angles provide an intuitive approach for visualizing the various coordinate reference frames, facilitating understanding of key concepts, and presenting the flight simulation results, as shown in Figures 3.1-3.3. Of particular pertinence for understanding the aircraft orientation during maneuvers are the pitch angle (Figure 3.2) and the roll angle (Figure 3.3).

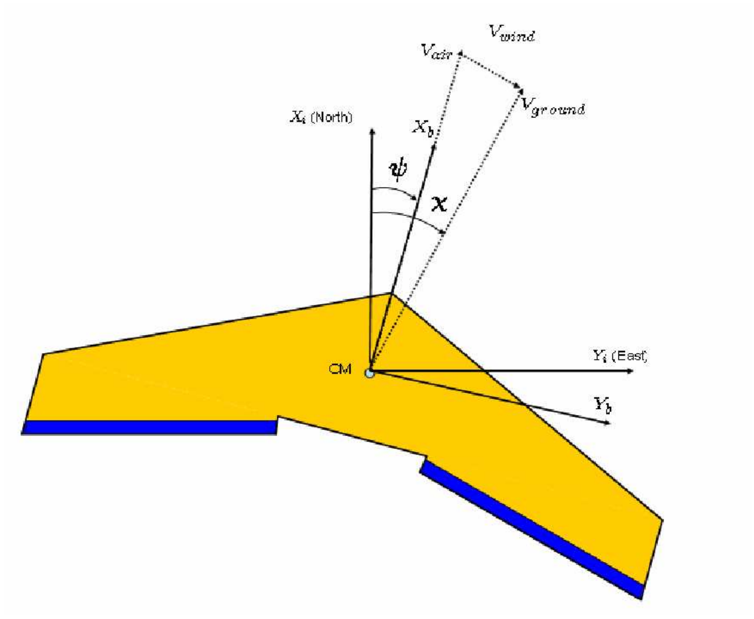


Figure 3.1: MAV Top View. The attitude angle for heading, ψ, is measured from local north to the nose of the aircraft. The course angle, χ, indicates the ground track of the aircraft relative to the inertial frame **x**-axis (north).

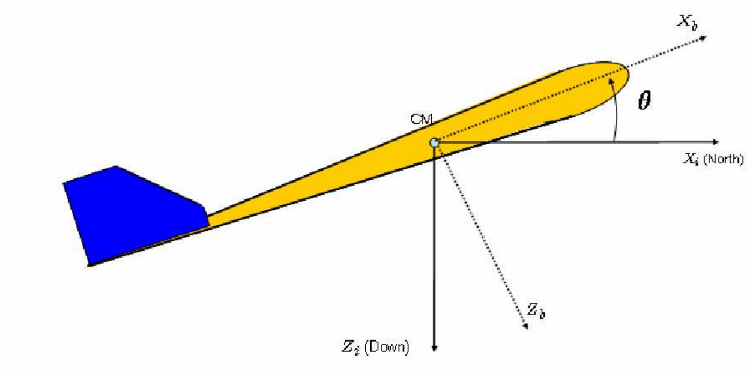


Figure 3.2: MAV Side View. The aircraft pitch angle, θ, is measured from an intermediate inertial-frame horizontal plane rotated by ψ to the line extending from the aircraft center of mass out the right wing. Positive rotation is right-handed sense.

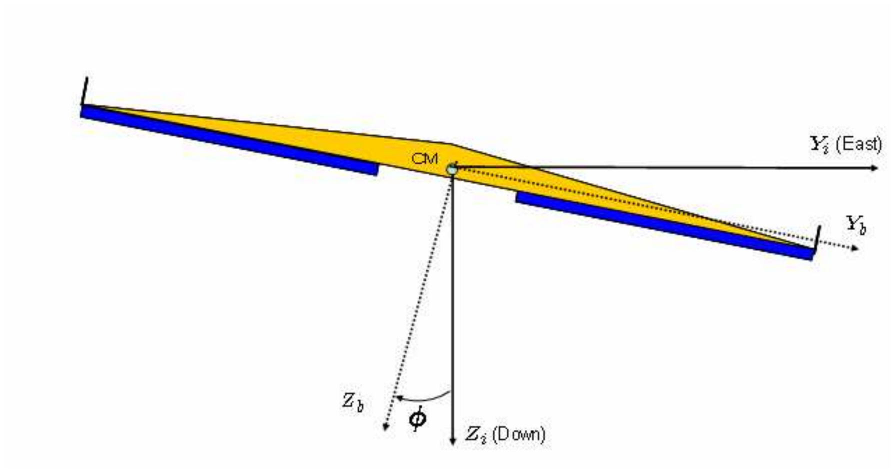


Figure 3.3: MAV Rear View. The aircraft roll angle, ϕ, is measured relative to an intermediate inertial-frame axis that has been rotated by ψ and θ. Positive rotation is in the right-handed sense.

Fundamental to developing maneuver trajectories, paths, and control laws is the ability to represent body-frame vectors in the inertial reference frame and vice versa. The transformation of vectors from one frame to the other can be accomplished using the Euler angle formulation of the rotation matrix, which can be written as

$$\mathbf{R}_{(\phi\theta\psi)} = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \end{bmatrix}, \tag{3.1}$$

where

$$\begin{aligned}
\mathbf{c}_1 &= \begin{bmatrix} \cos\theta\cos\psi \\ -\cos\phi\sin\psi + \sin\phi\sin\theta\cos\psi \\ \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \end{bmatrix} \\
\mathbf{c}_2 &= \begin{bmatrix} \cos\theta\sin\psi \\ \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi \\ -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi \end{bmatrix} \\
\mathbf{c}_3 &= \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix}.
\end{aligned}$$

Conversely, if we know the rotation matrix, we can find the separate Euler angles using the following three equations extracted from the components of Equation 3.1

$$\theta = -\sin^{-1}\left[\mathbf{c}_3(1)\right] \tag{3.2}$$

$$\phi = \tan^{-1}\left[\mathbf{c}_3(2), \mathbf{c}_3(3)\right] \tag{3.3}$$

$$\psi = \tan^{-1}\left[\mathbf{c}_2(1), \mathbf{c}_1(1)\right]. \tag{3.4}$$

The primary motivation for using the attitude quaternion rather than the Euler angles is to avoid the well-known singularity in the Euler angle kinematic equations. This singularity is can be readily observed in the equations relating angular rates to Euler angle rates,

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \frac{\sin\theta\sin\phi}{\cos\theta} & \frac{\sin\theta\cos\phi}{\cos\theta} \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{3.5}$$

where the $(\cos\theta)$ term in the denominator goes to 0 as pitch angle approaches 90 degrees. In the literature, this singularity results in a phenomenon often referred to as gimbal lock.

3.2.2 Attitude Quaternion

The attitude quaternion $(\mathbf{e})$ is defined as a rotation about a specific axis arranged as a column vector of four parameters, which can be written as,

$$\mathbf{e} = \begin{bmatrix} e_0 \\ e_x \\ e_y \\ e_z \end{bmatrix} \equiv \begin{bmatrix} \cos\left(\frac{\Theta}{2}\right) \\ E_x\sin\left(\frac{\Theta}{2}\right) \\ E_y\sin\left(\frac{\Theta}{2}\right) \\ E_z\sin\left(\frac{\Theta}{2}\right) \end{bmatrix}. \tag{3.6}$$

The first parameter, e_0, is a scalar quantity related to the angle of rotation, Θ, and the parameters e_x, e_y, and e_z are the scaled components of the unit-length axis of rotation, $\mathbf{E} = [E_x \, E_y \, E_z]^T$, in the inertial reference frame. One essential attribute of the attitude quaternion is that the Euclidean norm is equal to unity,

$$\begin{aligned} \|\mathbf{e}\| &= \sqrt{e_0^2 + e_x^2 + e_y^2 + e_z^2} \\ &= \sqrt{\cos^2\left(\frac{\Theta}{2}\right) + \left(E_x^2 + E_y^2 + E_z^2\right)\sin^2\left(\frac{\Theta}{2}\right)} \\ &= \sqrt{\cos^2\left(\frac{\Theta}{2}\right) + (1)\sin^2\left(\frac{\Theta}{2}\right)} \\ &= 1. \end{aligned}$$

This is in contrast to the strict mathematical definition of the quaternion as being a column vector with four parameters.

The unity-norm constraint on the attitude quaternion allows the formulation of a rigid-body rotation matrix from the four attitude quaternion parameters. This matrix is written as

$$\mathbf{R}(\mathbf{e})_{b \leftarrow i} = [\mathbf{c}_1 \; \mathbf{c}_2 \; \mathbf{c}_3], \tag{3.7}$$

where the three column vectors can be expressed as,

$$\begin{aligned} \mathbf{c}_1 &= \begin{bmatrix} e_0^2 + e_x^2 - e_y^2 - e_z^2 \\ 2(e_x e_y - e_z e_0) \\ 2(e_x e_z + e_y e_0) \end{bmatrix} \\ \mathbf{c}_2 &= \begin{bmatrix} 2(e_x e_y + e_z e_0) \\ e_0^2 - e_x^2 + e_y^2 - e_z^2 \\ 2(e_y e_z - e_x e_0) \end{bmatrix} \\ \mathbf{c}_3 &= \begin{bmatrix} 2(e_x e_z - e_y e_0) \\ 2(e_y e_z + e_x e_0) \\ e_0^2 - e_x^2 - e_y^2 + e_z^2 \end{bmatrix}. \end{aligned}$$

This matrix transforms vectors represented in the inertial frame to representations in the body frame. For example, given an inertial frame vector, v_i, the expression

$$v_b = \mathbf{R}(\mathbf{e})_{b\leftarrow i} v_i \tag{3.8}$$

yields the same vector expressed in the body reference frame. Because the rotation matrix is unitary, its transpose can be used to transform vectors in the body frame to the inertial frame,

$$\begin{aligned} v_i &= \mathbf{R}(\mathbf{e})^T_{b\leftarrow i} v_b \\ &= \mathbf{R}(\mathbf{e})_{i\leftarrow b} v_b. \end{aligned} \tag{3.9}$$

The attitude quaternion can also be used to rotate vectors from one reference frame to another by first defining the quaternion conjugate, $\mathbf{e}^*$, as,

$$\mathbf{e}^* = \begin{bmatrix} e_0 \\ -e_x \\ -e_y \\ -e_z \end{bmatrix}. \tag{3.10}$$

The vector transformation is achieved using the formula

$$v_i = \mathbf{e} \otimes v_i \otimes \mathbf{e}^*. \tag{3.11}$$

The unity-norm constraint is implicitly enforced by adhering to the rules of quaternion algebra when composing two quaternions. The composition of two quaternions, $\mathbf{e}$ and $\mathbf{e}'$, is written as

$$\mathbf{e}'' = \mathbf{e} \otimes \mathbf{e}'$$

$$= \begin{bmatrix} e_0e'_0 - e_xe'_x - e_ye'_y - e_ze'_z \\ e_0e'_x + e_xe'_0 - e_ye'_z + e_ze'_y \\ e_0e'_y + e_xe'_z + e_ye'_0 - e_ze'_x \\ e_0e'_z - e_xe'_y + e_ye'_x + e_ze'_0 \end{bmatrix}. \tag{3.12}$$

If $\mathbf{e}$ and $\mathbf{e}'$ both meet the unity-norm constraint, the quaternion resulting from their composition will also be unity-norm. The composition of two attitude quaternions represents two successive rotations, whereas the quaternion resulting from the composition is a single rotation that produces the same transformation. For example, if $\mathbf{e}'$ rotates from coordinate system 1 to 2 and $\mathbf{e}$ rotates coordinate system 2 to 3, then the composed quaternion rotates directly from 1 to 3. In equation form, this can be written as

$$\mathbf{e}''_{3\leftarrow 1} \;=\; \mathbf{e}_{3\leftarrow 2} \otimes \mathbf{e}'_{2\leftarrow 1}, \tag{3.13}$$

where the order of composition is defined so that the multiplication of the associated rotation matrices yields the same total rotation, which is expressed as

$$\mathbf{R}\left(\mathbf{e}''\right)_{3\leftarrow 1} \;=\; \mathbf{R}\left(\mathbf{e}\right)_{3\leftarrow 2} \mathbf{R}\left(\mathbf{e}'\right)_{2\leftarrow 1}. \tag{3.14}$$

Given this physical interpretation for the composition of attitude quaternions, it is evident that composition is not commutative, $\mathbf{e} \otimes \mathbf{e}' \neq \mathbf{e}' \otimes \mathbf{e}$.

One can readily see the primary advantage of the attitude quaternion compared to the Euler angles by comparing Equation (3.5) with the differential equation for the attitude quaternion kinematics, which can be written as

$$\dot{\mathbf{e}} \;=\; \frac{1}{2} \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \otimes \mathbf{e}, \tag{3.15}$$

where $\boldsymbol{\omega}$ is the vector of body-frame angular rates $[p\ q\ r]^T$ from the three gyroscopes (one for each body-frame axis). Not only does the attitude quaternion avoid mathematical singularities, Equation (3.15) is bi-linear in the attitude quaternion parameters and the gyroscope measurements.

One of the key concepts utilized in our MEKF algorithm is the ability to construct an error quaternion from an Euler attitude error vector [28]. The three components of the Euler attitude error vector are defined as the product of an orthogonal unit vector, $\mathbf{E}_e$, and rotation angle, Θ_e(measured in radians) ,

$$\mathbf{a}_e = \Theta_e \mathbf{E}_e = \begin{bmatrix} a_{ex} \\ a_{ey} \\ a_{ez} \end{bmatrix}. \tag{3.16}$$

From the Euler attitude error vector, we can construct a unity-norm error quaternion, $\delta\mathbf{e}\left(\mathbf{a}_e\right)$, according to the formulation,

$$\delta\mathbf{e}\left(\mathbf{a}_e\right) = \begin{bmatrix} \cos\left(\frac{\|\mathbf{a}_e\|}{2}\right) \\ \left(\frac{a_{ex}}{\|\mathbf{a}_e\|}\right)\sin\left(\frac{\|\mathbf{a}_e\|}{2}\right) \\ \left(\frac{a_{ey}}{\|\mathbf{a}_e\|}\right)\sin\left(\frac{\|\mathbf{a}_e\|}{2}\right) \\ \left(\frac{a_{ez}}{\|\mathbf{a}_e\|}\right)\sin\left(\frac{\|\mathbf{a}_e\|}{2}\right) \end{bmatrix}, \tag{3.17}$$

where the range of the error angle is $\Theta_e \in \begin{bmatrix} 0, & 2\pi \end{bmatrix}$.

3.3 Time Update

For the purposes of completeness and clarity, the standard EKF time update equations are presented first, followed by an explanation of how these have been adapted to solve the MAV attitude estimation problem.

3.3.1 Standard EKF Time Update

Using the notation for the continuous-discrete EKF from [33], the standard form of the dynamic system of equations is written as

$$\dot{\mathbf{x}} = f(\mathbf{x},\mathbf{u},t) + \mathbf{G}\mathbf{w}(t) \tag{3.18}$$

$$\mathbf{z}(k) = h(\mathbf{x},k) + \mathbf{v}(k). \tag{3.19}$$

The differential equation for the covariance matrix, $\mathbf{P}$, is written as

$$\dot{\mathbf{P}} = \mathbf{F}\mathbf{P} + \mathbf{P}\mathbf{F}^T + \mathbf{G}\mathbf{Q}\mathbf{G}^T, \tag{3.20}$$

where $\mathbf{G}$ is the cross-correlation matrix, and $\mathbf{Q}$ is the covariance of the stationary, zero-mean process noise, $\mathbf{w}(t)$. For nonlinear systems, the state transition matrix, $\mathbf{F}$, is computed at each time step from the Jacobian of Equation (3.18), as

$$\mathbf{F} = \frac{\partial f(\mathbf{x},\mathbf{u},t)}{\partial \mathbf{x}}. \tag{3.21}$$

3.3.2 MEKF Time Update

For the MAV attitude estimation problem, we use an attitude error model which makes two key assumptions: 1) the static bias for each gyroscope have been estimated offline and 2) the transient biases are negligible. Thus our dynamic model of attitude error can be written as

$$\dot{\mathbf{a}}_e = -\boldsymbol{\omega} \times \mathbf{a}_e + \mathbf{w}(t), \tag{3.22}$$

where $\mathbf{w}(t)$ is the stationary, white noise on the gyroscopes, $\boldsymbol{\omega}$ is the body-frame angular rates from the gyroscopes, and $\mathbf{a}_e$ (the Euler attitude error vector) is the state vector. When implemented on the autopilot, the discrete time propagation of the state can be accomplished using Euler integration. Because of the assumptions of known static biases and zero-mean noise, the expected value of the attitude error after the time update is equal to zero $(\hat{\mathbf{a}}_e(k+1) = [0])$.

With the system model given in Equation (3.22), the state update matrix, $\mathbf{F}$, used in the propagation of the covariance matrix, is written as

$$\mathbf{F} = \frac{\partial f(\mathbf{x}, \mathbf{u}, t)}{\partial \mathbf{x}} = -\hat{\boldsymbol{\omega}}(k), \tag{3.23}$$

where $\hat{\boldsymbol{\omega}}(k)$ is defined as a skew-symmetric matrix with elements equal to the body-frame angular rates, written as

$$\hat{\boldsymbol{\omega}}(k) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}. \tag{3.24}$$

Making the standard assumption that the process noise, $\mathbf{w}(t)$, is an uncorrelated, stationary, zero-mean, Gaussian process, the cross-correlation matrix, $\mathbf{G}$, in Equation (3.20), is the identity matrix. Thus, the discrete Euler integration update equations are written as,

$$\hat{\mathbf{a}}_e(k+1) = 0 \tag{3.25}$$

$$\mathbf{P}(k+1) = \mathbf{P}(k) + T_s\Big(\mathbf{F}(k)\mathbf{P}(k) + \mathbf{P}(k)\mathbf{F}(k)^T + \mathbf{Q}\Big), \tag{3.26}$$

where T_s is the integration time step.

Concurrent with the propagation of the state and the covariance matrix, we also propagate the attitude quaternion using Euler integration of Equation (3.15),

$$\mathbf{e}(k+1) = \mathbf{e}(k) + \frac{T_s}{2}\left(\begin{bmatrix} 0 \\ \boldsymbol{\omega}(k) \end{bmatrix} \otimes \mathbf{e}(k)\right). \tag{3.27}$$

Following the time propagation of the attitude quaternion, the unity-norm constraint can be explicitly enforced by dividing each of the parameters by the norm of the attitude quaternion, as

$$\mathbf{e}(k+1) \leftarrow \frac{\mathbf{e}(k+1)}{\|\mathbf{e}(k+1)\|}. \tag{3.28}$$

3.4 Measurement Update

The discussion of the measurement update is also divided into a presentation of the standard EKF formulation followed by a discussion of the necessary modifications to create the MAV-specific version of the MEKF.

3.4.1 Standard EKF Measurement Update

In the standard EKF, the measurement update portion is divided into three parts: calculating the Kalman gain, updating the covariance matrix, and estimating the new system state. The key relationship for the EKF measurement update step is the measurement output matrix, $\mathbf{H}(k)$, linearized at each iteration as

$$\mathbf{H}(k) = \frac{\partial h(\mathbf{x}, k)}{\partial \mathbf{x}}. \tag{3.29}$$

The equation for calculating the Kalman gain matrix can be written as

$$\mathbf{K}(k) = \mathbf{P}(k)^{-} \mathbf{H}(k)^{T} \left(\mathbf{R} + \mathbf{H}(k) \mathbf{P}(k)^{-} \mathbf{H}(k)^{T}\right)^{-1}, \tag{3.30}$$

where $\mathbf{R}$ represents the applicable sensor noise covariance and $\mathbf{P}(k)^{-}$ is the previously calculated covariance matrix. The Kalman gain matrix, $\mathbf{K}(k)$, is used to calculate the new covariance matrix, according to the equation

$$\mathbf{P}(k)^{+} = (\mathbf{I} - \mathbf{K}(k) \mathbf{H}(k)) \mathbf{P}(k)^{-}. \tag{3.31}$$

In the final step of the standard EKF, $\mathbf{K}(k)$ is used to scale the state error prior to summation with the previous state estimate to determine the new state estimate, according to

$$\hat{\mathbf{x}}(k)^{+} = \hat{\mathbf{x}}(k)^{-} + \mathbf{K}(k) \left[\mathbf{z}(k) - \mathbf{H}\hat{\mathbf{x}}(k)^{-}\right]. \tag{3.32}$$

3.4.2 MEKF Measurement Update

Incremental Update

Our attitude estimation algorithm uses an observation of the heading vector derived from the GPS and an estimation of the wind vector (see [34]), along with an observation of the gravity vector derived from the accelerometers, to provide the two non-collinear vectors needed for attitude measurement. However, these two observations should be treated independently for two reasons: disparate sensor operating frequencies and different observation reference frames. Hence, our first modification to the standard EKF is an implementation of the incremental update concept [32], which allows the measurement update to be completed independently for each sensor.

The first motive for pursuing the incremental update concept is that the accelerometers and the GPS operate at different frequencies. The accelerometers are analog devices that continuously sense gravity, allowing our attitude estimation algorithm to be updated with each iteration of the autopilot code, nominally 50 Hz. Conversely, the GPS data update occurs at a fixed rate of 4 Hz, meaning that GPS-derived heading information is used to update the attitude estimate only when new data is available.

Additionally, the observations from the GPS and accelerometers do not occur in the same reference frame. The GPS measurements are made in the inertial frame and accelerometers measure in the body frame. This is important because the order of quaternion composition is reversed for the body frame compared to the inertial frame.

Euler Attitude Error Vector

Although several methods exist for calculating attitude error [31, 25], our approach builds the Euler attitude error vector directly from observed vectors and their predicted values. In generic notation, the measured and predicted vectors can be denoted as $\mathbf{v}_m$ and $\mathbf{v}_p$. Calculating the cross product of the normalized measurement and predicted vectors according to

$$\mathbf{E}_e = \frac{\mathbf{v}_p}{\|\mathbf{v}_p\|} \times \frac{\mathbf{v}_m}{\|\mathbf{v}_m\|}, \tag{3.33}$$

yields a unit-vector rotation axis around which the attitude estimate needs to be rotated to correct the error. Next, the error angle is found from the dot product of the same two vectors using the expression,

$$\Theta_e = \cos^{-1}\left(\frac{\mathbf{v}_p}{\|\mathbf{v}_p\|} \cdot \frac{\mathbf{v}_m}{\|\mathbf{v}_m\|}\right). \tag{3.34}$$

The Euler attitude error vector is the product of the unit vector and the angle, written as

$$(\mathbf{a}_e)_m = \Theta_e \mathbf{E}_e. \tag{3.35}$$

Because the time propagated expected value of the Euler attitude error vector is defined to be zero ($\hat{\mathbf{a}}_e^- = 0$), the state update equation (3.32) becomes

$$\left[\; \hat{\mathbf{a}}_e^+ \;\right] = \left[\; 0 \;\right] + \mathbf{K}\left[(\mathbf{a}_e)_m - 0\right]. \tag{3.36}$$

Additionally, since the measured Euler attitude error vector is created from the sensor observations, the measurement output matrix, $\mathbf{H}(k)$, is a 3×3 identity matrix.

We complete the measurement update by implementing the reset operation, which moves the attitude error information from the state vector into the attitude quaternion. The first step in the reset operation is to map the Euler attitude error vector into an error quaternion according to Equation (3.17). Next, we compose the error quaternion with the previous estimate of the attitude quaternion,

$$\hat{\mathbf{e}}^+ = \delta\mathbf{e}\left(\hat{\mathbf{a}}_e^+\right) \otimes \hat{\mathbf{e}}^-, \tag{3.37}$$

where the order of composition is reversed if the measurements have been made in the inertial reference frame. By correcting the attitude quaternion with the error information, the attitude estimate is assumed to be accurate and therefore, the expected value of the error is zero. Thus, the state vector is reset to zero prior to each iteration of the MEKF algorithm. Given the generalized MEKF measurement update procedure outlined above, we now present our sensor-specific implementation.

Accelerometer Measurement Update

When in steady-level flight, the accelerometers provide an indication of the gravity vector, which can be compared with the inertial $\mathbf{z}$-axis rotated into the body frame, $(\mathbf{z}_i)_b$, using the rotation matrix (Equation (3.7) based on the previous attitude quaternion). Given these two body-frame vectors, the accelerometer-based attitude error axis and error angle are computed as

$$\mathbf{E}_{eA} = \frac{\mathbf{g}}{\|\mathbf{g}\|} \times (\mathbf{z}_i)_b \tag{3.38}$$

$$\Theta_{eA} = \cos^{-1}\left(\frac{\mathbf{g}}{\|\mathbf{g}\|} \cdot (\mathbf{z}_i)_b\right), \tag{3.39}$$

where the normalized $\mathbf{g}$ is the observed direction of the gravity vector. The error axis and angle are then substituted into Equation (3.35) to calculate $(\mathbf{a}_e)_A$. The Kalman gain matrix, $\mathbf{K}(k)$, is calculated based on the measurement noise matrix, $\mathbf{R}$, applicable to the accelerometers. $\mathbf{K}(k)$ is used in Equation (3.31) to calculate the new covariance matrix, $\mathbf{P}(k)^+$, and in Equation (3.36) to scale the Euler attitude error vector, $\hat{\mathbf{a}}_e^+$. Finally, the reset operation is accomplished by first mapping $\hat{\mathbf{a}}_e^+$ into an error quaternion using Equation (3.17) and then moving the attitude error information into the attitude estimate from the previous time update according to Equation (3.37). Once the attitude estimate has been corrected, the state vector is reset to zero ($\hat{\mathbf{a}}_e = 0$) in preparation for the next iteration.

GPS Measurement Update

The GPS portion of the measurement update is accomplished using a heading vector, $\boldsymbol{\psi}_G$, derived from the GPS measurement of course angle and the wind vector

[34]. To find the heading error, the heading observation is compared to our predicted heading. This prediction of heading is obtained by projecting the body reference frame $\mathbf{x}$-axis onto the inertial $\mathbf{x}-\mathbf{y}$ plane using the transpose of the rotation matrix based on the previous attitude quaternion. Again, the vector cross product and dot product are used to calculate the error axis and angle, as

$$\begin{aligned} \mathbf{E}_{eG} &= \frac{(\mathbf{x}_b)_i}{\| (\mathbf{x}_b)_i \|} \times \boldsymbol{\psi}_G && (3.40) \\ \Theta_{eG} &= \cos^{-1}\left[\frac{(\mathbf{x}_b)_i}{\| (\mathbf{x}_b)_i \|} \cdot \boldsymbol{\psi}_G\right], && (3.41) \end{aligned}$$

where $(\mathbf{x}_b)_i$ needs to be normalized after projection onto the $\mathbf{x}-\mathbf{y}$ plane. The GPS-based Euler attitude error vector, $(\mathbf{a}_e)_G$, is then calculated. This vector is scaled by the $\mathbf{K}(k)$ calculated based on the GPS noise measurement matrix to get the new $\hat{\mathbf{a}}_e^+$. As with the accelerometer measurement, this updated attitude error vector is mapped into a unity-norm error quaternion and composed with the previous attitude quaternion. Note, however, that the order of composition,

$$\hat{\mathbf{e}}^+ = \hat{\mathbf{e}}^- \otimes \delta\mathbf{e}\left(\hat{\mathbf{a}}_e^+\right), \qquad (3.42)$$

is reversed from the accelerometer measurement update due to the heading vectors being compared in the inertial reference frame. Moving the attitude error information into the attitude quaternion allows the state vector to be reset prior to the next iteration of the algorithm.

Accelerometer Measurement Noise Tuning

When not in steady-level flight, accelerometers sense linear, angular, and Coriolis accelerations in addition to the acceleration of gravity. This degradation of the observed gravity vector was addressed in [31] by turning off the measurement update whenever the gravity measurement was corrupted beyond a certain threshold. Our approach is to scale each component of the accelerometer measurement noise matrix, $\mathbf{R}$, to account for the decreased confidence in the accelerometer measurements. The equation we use for computing the scaled, diagonal components of **R** can be written as

$$\mathbf{R} \;=\; \mathbf{R}\left[1+\mathbf{k}\left(1-\|\mathbf{g}\|\right)^2\right], \tag{3.43}$$

where **k** is a vector of tunable gains and **g** is the accelerometer measurements. Thus, during steady-level flight, the norm of the accelerometer measurements (measured in gravity units) should be nearly equal to one, making the difference zero and the components of **R** unchanged. Conversely, while maneuvering, the difference will be non-zero, effectively reducing confidence in the accelerometer measurement values.

Induced Heading Error Correction

One challenge of the incremental update methodology is that the accelerometer measurement update can induce an error in the heading portion of the attitude estimate. Noting that the gravity vector should align with the inertial **z**-axis, the attitude estimate correction provided by the accelerometers provides no heading information. Thus, any change in the heading estimate created during the accelerometer portion of the measurement update is erroneous.

The aircraft heading vector is defined as the body-frame **x**-axis projected onto the inertial $\mathbf{x}-\mathbf{y}$ plane, denoted as $\left(\mathbf{x}_b^-\right)_i$ or $\left(\mathbf{x}_b^+\right)_i$, where the superscript indicates the heading before or after the accelerometer measurement update. After normalizing these two vectors, an Euler attitude error vector for the induced heading error can be calculated according to

$$\mathbf{E}_{eI} \;=\; \frac{\left(\mathbf{x}_b^+\right)_i}{\|\left(\mathbf{x}_b^+\right)_i\|} \times \frac{\left(\mathbf{x}_b^-\right)_i}{\|\left(\mathbf{x}_b^-\right)_i\|} \tag{3.44}$$

$$\Theta_{eI} \;=\; \cos^{-1}\left[\frac{\left(\mathbf{x}_b^+\right)_i}{\|\left(\mathbf{x}_b^+\right)_i\|} \cdot \frac{\left(\mathbf{x}_b^-\right)_i}{\|\left(\mathbf{x}_b^-\right)_i\|}\right] \tag{3.45}$$

$$\left(\mathbf{a}_e\right)_I \;=\; \Theta_{eI}\mathbf{E}_{eI}. \tag{3.46}$$

We convert $\mathbf{a}_{eI}$ into the error quaternion according to Equation (3.17) and compose this error quaternion with the attitude estimate from the accelerometer update, according to

$$\hat{\mathbf{e}}^{+} = \hat{\mathbf{e}}^{-} \otimes \delta\mathbf{e}\left(\mathbf{a}_e\right)_I , \tag{3.47}$$

where the order of composition is as shown in Equation (3.42) since the heading error is in the inertial reference frame.

In conclusion, our MEKF approach enforces the unity-norm constraint on the attitude quaternion by using the multiplicative reset operation to move the attitude error information from the state to the attitude estimate. Additionally, the incremental update approach allows the attitude estimation problem to be decomposed for disparate sensor information rates and reference frames. To validate our attitude estimation algorithm, flight simulations were run using the Aviones[1] software.

3.5 Results

The simulation results were obtained with the Aviones software providing the true attitude for comparison with the MEKF and VGO estimation schemes. However, much difficulty was experienced in obtaining definitive hardware results due to the lack of truth data for the flying MAV.

3.5.1 Simulation Results

We conducted simulations to refine and demonstrate the quaternion MEKF attitude estimation using the Aviones software to generate truth data for each of the attitude parameters. Additionally, the VGO data was included to provide an additional indication of the MEKF performance. Loop maneuvers were executed to exercise the attitude estimation algorithm in a situation where the Euler angle singularity would be experienced. Additionally, small diameter orbits were flown to

[1]The Aviones flight simulation software is available at `http://sourceforge.net/projects/avoines`

determine performance in an extended-duration turning maneuver. During both maneuvers, the simulator attitude information (truth) was used in the feedback loop, thereby isolating the MEKF attitude estimation scheme from the control of the aircraft. In the plots, the attitude quaternion is converted to Euler angles to clarify the presentation.

The results for a loop maneuver, commanded using a fifth-order trajectory in pitch angle (see Section 4.2), are shown in Figures 3.4 and 3.5. The vectors in Figure 3.4 show the MEKF estimate of pitch angle at various positions during the loop. The inward pointing vectors demonstrate the actual pitch angle required to track the commanded pitch angle. Note that the vectors are not plotted at regular time intervals due to both the variation in the loop speed of the autopilot code and changing airspeed. Referring to Figure 3.5, the MEKF estimate of pitch angle is plotted with the commanded pitch angle and the simulator pitch angle. Throughout the loop maneuver, the MEKF algorithm was able to consistently provide an accurate attitude estimate compared to the simulator truth data without experiencing the mathematical singularity problems associated with the Euler angle attitude representation. An interesting result can be seen in Figure 3.5, where between 4 and 5 seconds the simulator and MEKF both lead the desired angle; this is due to slight overshoot in the feedback loop of the control law rather than an error in the attitude estimate.

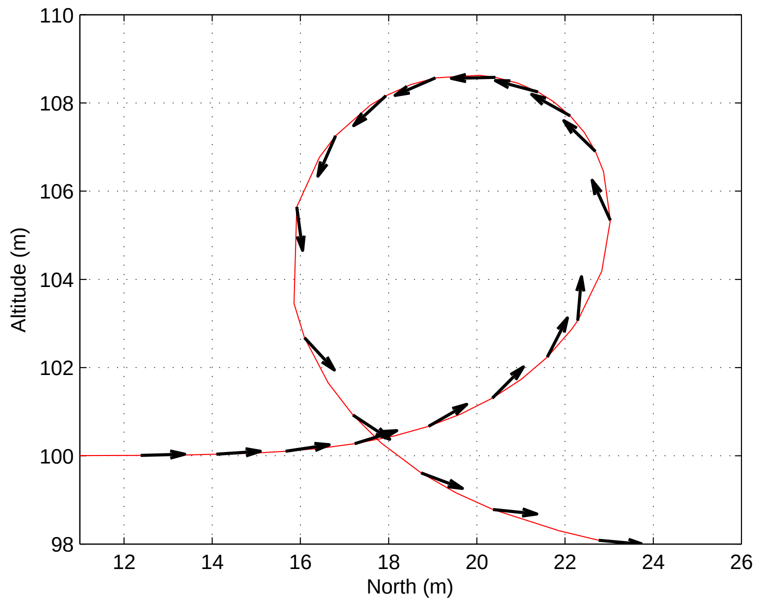


Figure 3.4: Loop Maneuver Pitch Angle. The MEKF estimate of pitch angle is shown plotted as a function of position during a loop maneuver.

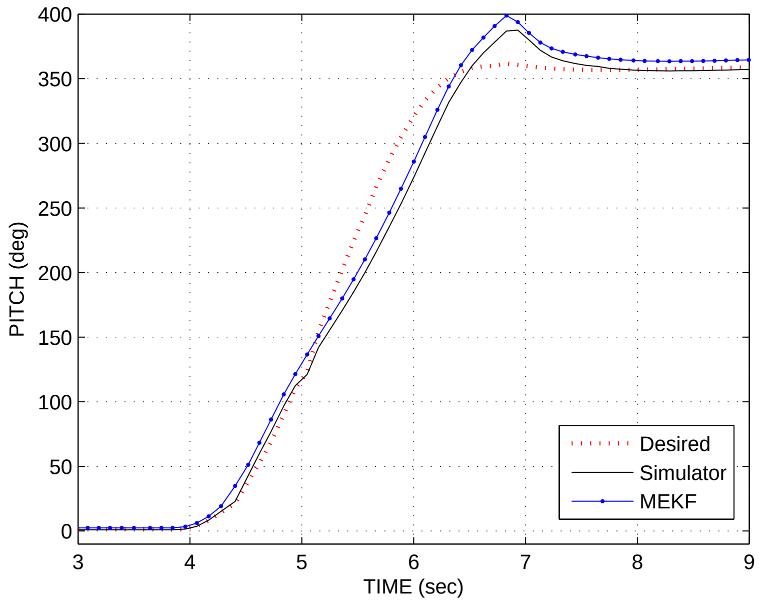


Figure 3.5: MEKF Pitch and Simulator True Pitch. The MEKF estimate of pitch angle is compared to the true pitch angle from the simulator and the commanded pitch angle from the maneuver controller.

The plot in Figure 3.6 shows the variation of the Kalman gains during the loop maneuver. Note that our formulation of the MEKF creates a diagonal Kalman gain matrix where the gains for both pitch and roll are identical due to the noise characteristics of the accelerometers. It can be seen that the accelerometer measurement noise tuning has caused the gains to be significantly reduced during the maneuver. By closely examining the Kalman gain matrix formula in Equation (3.30) and measurement covariance matrix in Equation (3.31), one would expect that a decrease in the Kalman gains for the accelerometer would result in a decreased covariance matrix, which would then produce a positive variation for the GPS-related Kalman gain. This variation indicates an increased relative confidence in the accuracy of the measured heading angle.

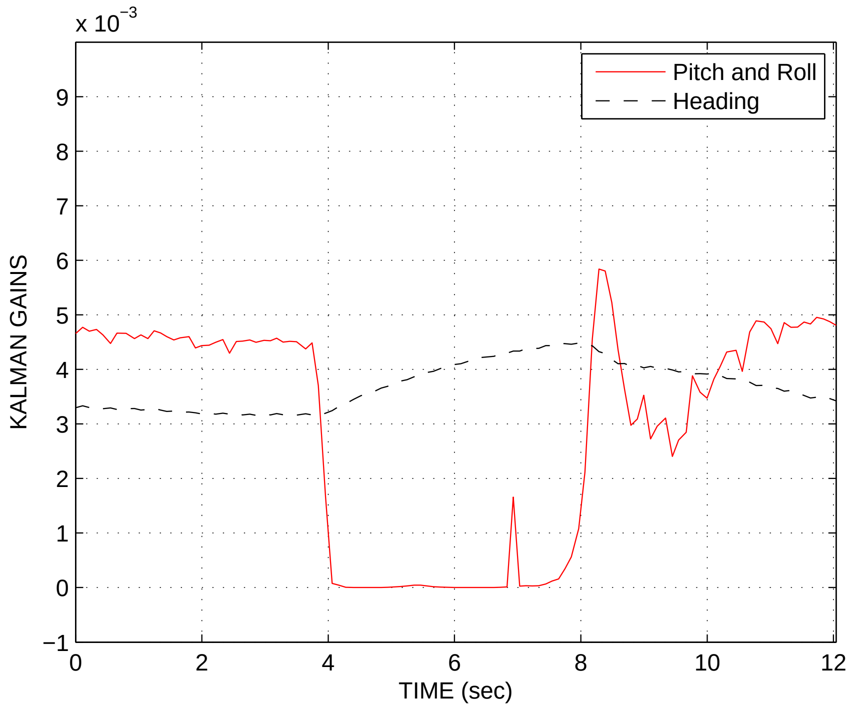


Figure 3.6: MEKF Kalman Gains. The MEKF Kalman gains show the effect of scaling the accelerometer measurement noise matrix during the loop maneuver.

The challenge of tracking attitude while flying small radius orbits was handled quite effectively by our MEKF attitude estimation scheme. The vectors shown in

Figure 3.7 indicate the heading angle at each position of the orbit as estimated by the MEKF algorithm. The plots in Figure 3.8 cover three complete circuits of a 40 meter orbit, demonstrating accurate tracking of the simulator pitch and roll angles by the MEKF attitude estimation algorithm. Examination of the pitch and roll angle plots shows that the MEKF performs significantly better than the VGO. The inability to correctly estimate attitude during turns can negatively effect geo-referencing and altitude-hold functions of the autopilot.

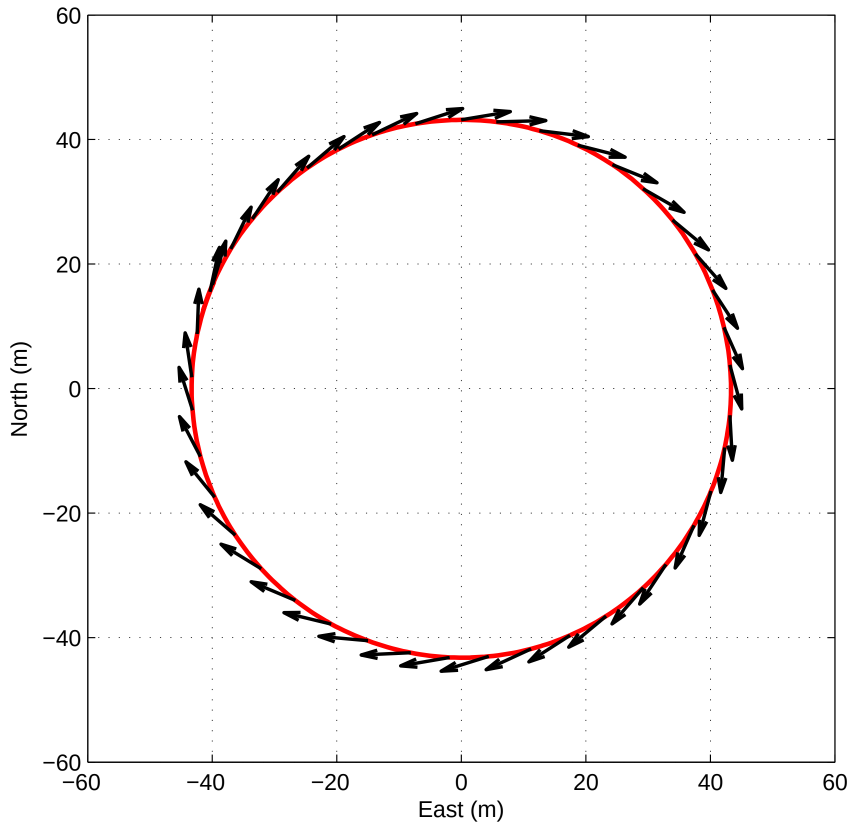


Figure 3.7: MEKF Heading Angle. The plot shows the estimated heading angle of the MEKF while flying a 40-meter radius circle.

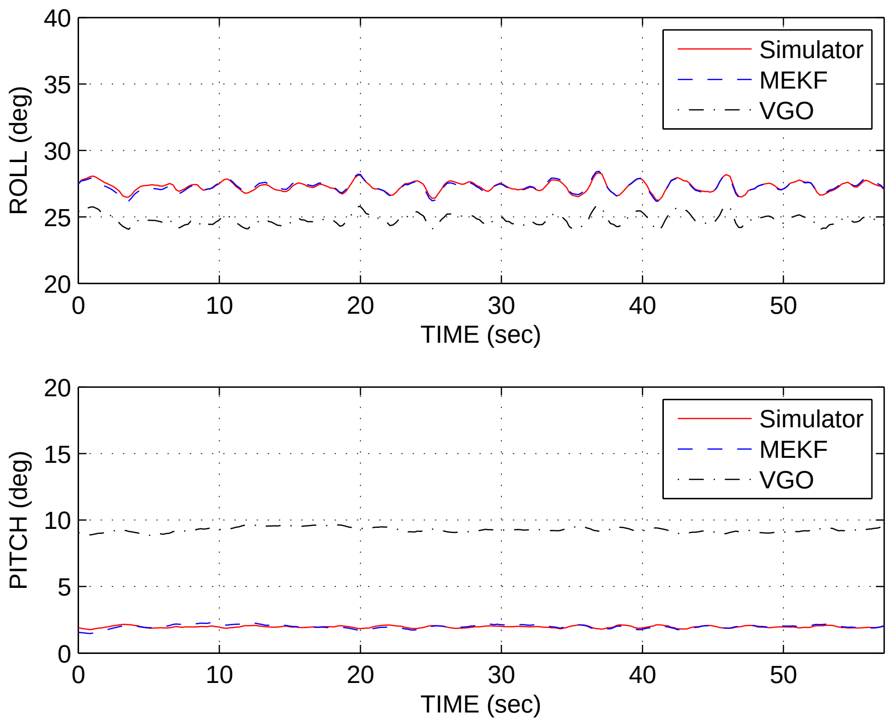


Figure 3.8: MEKF Pitch and Roll Angles. The plots show the tracking of the MEKF compared to both the true attitude from the simulator and the VGO while flying the 40-meter radius circle.

3.5.2 Hardware Results

The hardware results were found through ground experiments and flight tests. The purpose of the ground tests was to demonstrate that the MEKF attitude estimation algorithm functions well using real, rather than modeled, sensors. For the ground tests, we built an experimental piece of equipment (shown in Figure 3.9), which allowed the autopilot, including gyroscopes and accelerometers (not GPS), to be mounted in a fixture and rotated about a single axis at a time. The static portion of the test equipment was marked at increments of ten degrees and the rotating fixture was given a zero-rotation marking. Truth data was provided by a potentiometer with a accuracy of ±1 degree against which the MEKF attitude estimation scheme was compared.

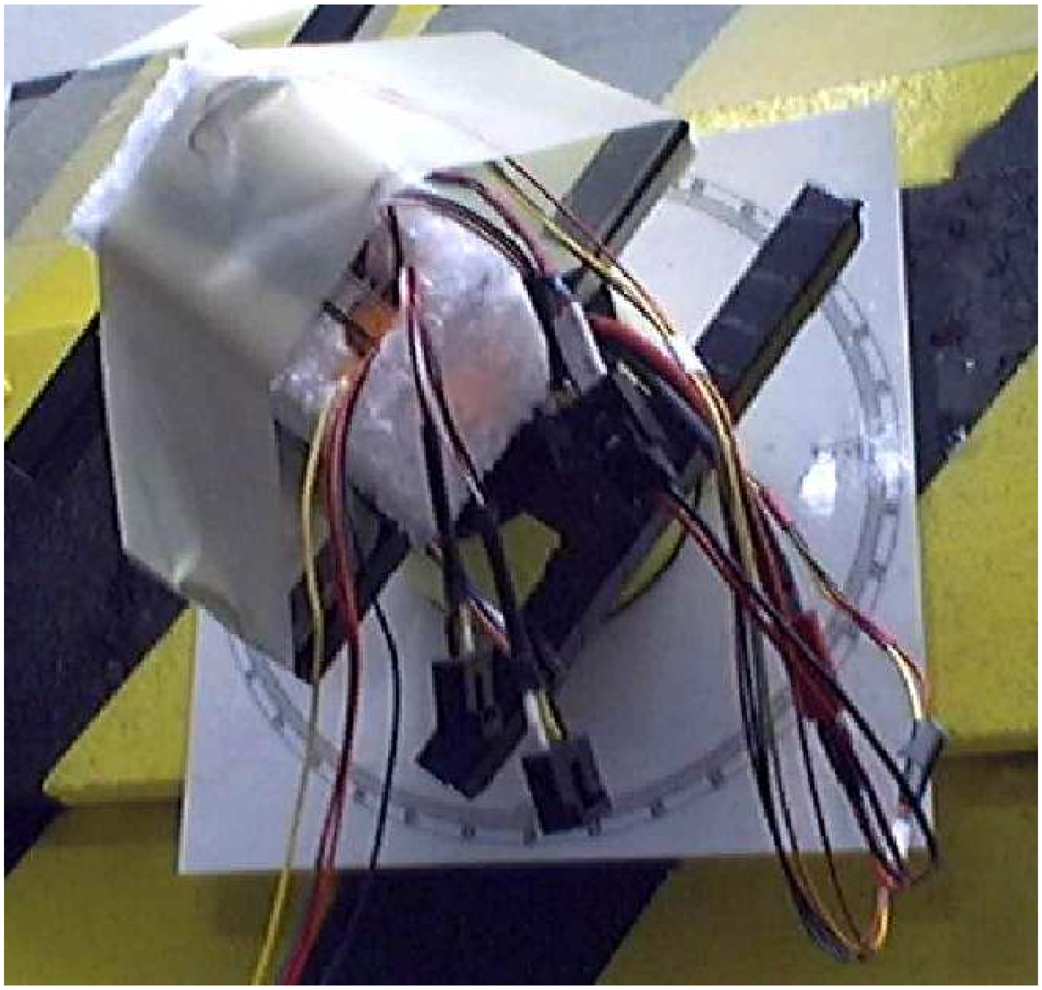


Figure 3.9: Attitude Ground Test Fixture. The experimental apparatus provides truth data for testing the MEKF on the autopilot hardware. The fixture is shown mounted so that the axis of rotation is parallel to the floor, allowing for rotations in pitch or roll angle. The axis of rotation is changed by mounting the autopilot on the fixture in specific orientations. The pitch angle testing orientation is shown.

For the first experiment, the autopilot was mounted on the apparatus fixture so that rotations would change pitch angle only. For this experiment, the fixture was turned one revolution so as to mimic the angular sweep of a loop maneuver. The MEKF attitude estimate, the dotted line in Figure 3.10, tracked the motion of the fixture quite accurately, thus validating its performance on the autopilot hardware. Note that for the attitude quaternion, pitch angle is defined on the range $\theta \in \left[-\frac{\pi}{2}, \quad \frac{\pi}{2} \right]$ and therefore the output from the MEKF attitude estimate was unwrapped onto the domain of the fixture and then scaled into units of degrees.

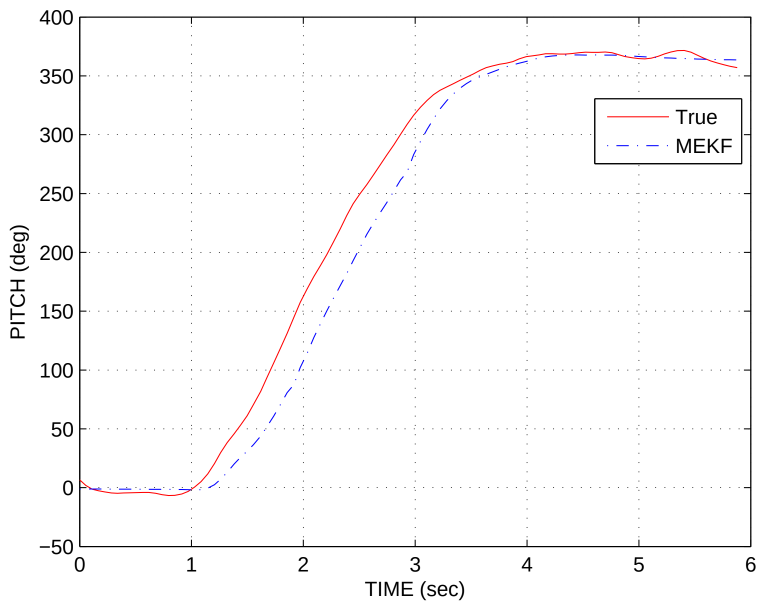


Figure 3.10: MEKF Pitch Angle Ground Test. The output angle for pitch was plotted against the truth data generated by the test apparatus.

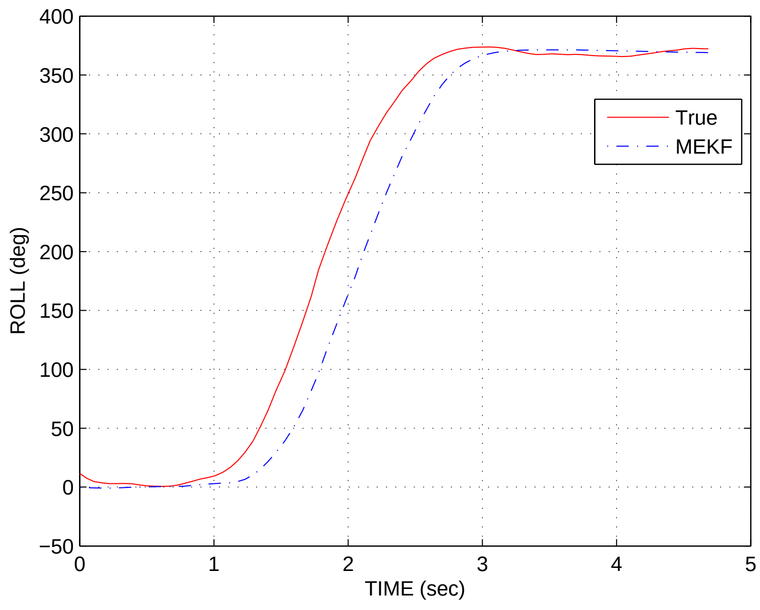


Figure 3.11: MEKF Roll Angle Ground Test. The plot shows the roll attitude estimation and truth data using the ground test equipment.

For the second experiment, the mounting of the autopilot was changed so that rotating the fixture would change roll angle only. To test roll angle tracking performance, the fixture was rotated one full revolution mimicking an aileron roll maneuver. The plots in Figure 3.11 show the MEKF attitude estimation moving directly to desired rotation and holding that angle. The roll angle output of the attitude quaternion is defined as $\phi \in \begin{bmatrix} -\pi, & \pi \end{bmatrix}$; the data plotted in Figure 3.11 has been mapped onto $\begin{bmatrix} 0, & 2\pi \end{bmatrix}$ and converted from radians to degrees. The ground testing demonstrated that the MEKF provides an accurate attitude estimation using the real sensors with all of the inherent bias and noise.

Finally, the autopilot was mounted in an aircraft and flown using the MEKF attitude estimation scheme in the feedback loop. Due to the lack of truth data during flight tests, Figure 3.12 shows the MEKF attitude estimation scheme compared to the desired pitch and roll angles generated by the autopilot and the attitude estimation produced by the VGO. The autopilot was running in waypoint-following and altitude-hold modes, which means that pitch was being used to compensate for errors in altitude and roll was used to correct errors in heading. Although this data lacks an indication of the true attitude, we can assert that the MEKF attitude estimation was sufficiently accurate and robust to allow the MAV to successfully fly and navigate continuously for more than twenty minutes.

Although not a strict truth metric, comparing the estimated pitch angle to the MAV altitude provides an indication of the performance of the pitch angle portion of the attitude estimate. If the aircraft altitude sensor indicates the MAV is above the desired altitude (100 meters in this case), then the autopilot should command a negative pitch angle; deviations below the desired altitude should result in positive pitch angles. As can be seen in Figure 3.13, the MEKF-derived estimate of the pitch angle behaves as expected, decreasing when altitude goes above 100 meters and increasing when the altitude goes below. Hence, the MEKF algorithm provides an attitude estimate that is accurate enough to allow stable altitude hold.

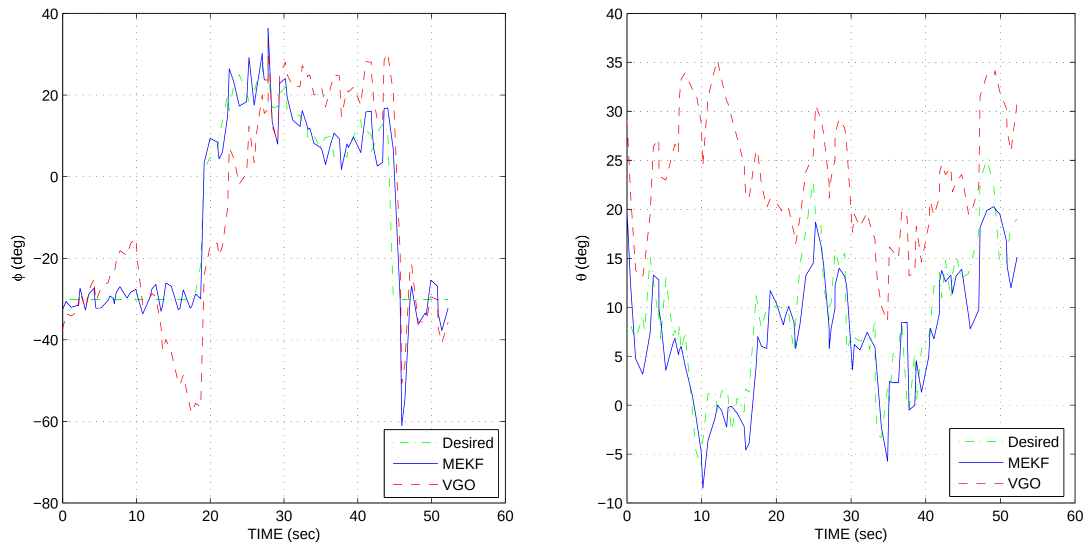


Figure 3.12: MEKF Angle Flight Data. A snapshot of flight data comparing the autopilot desired angles with the MEKF and the VGO attitude estimate. The MEKF data was being used in controlling the MAV.

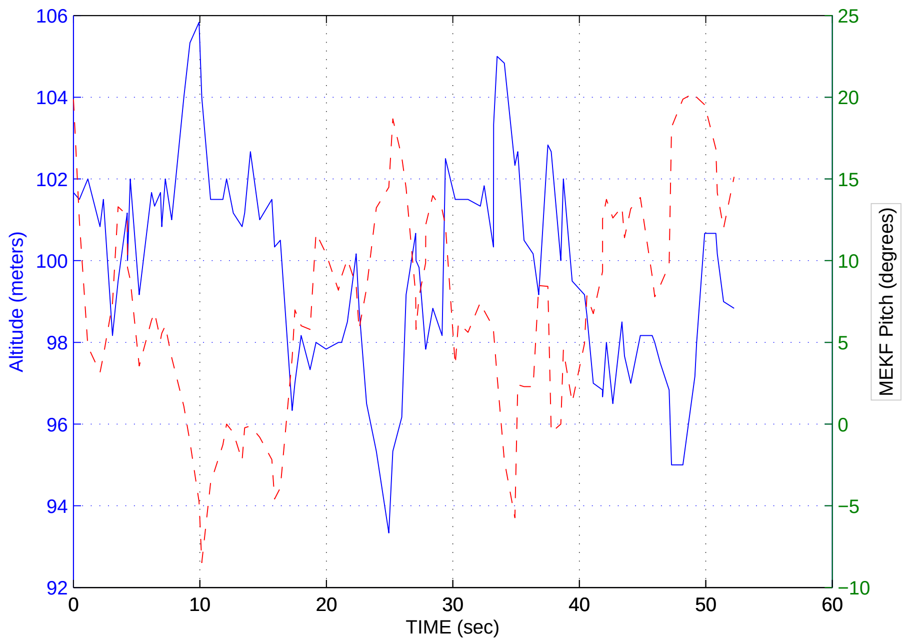


Figure 3.13: MEKF Altitude Flight Data. The MEKF pitch changes to return the MAV to the desired altitude indicating that the attitude estimation is behaving according to expectations.

3.6 Summary

The MEKF attitude estimation scheme adapts the quaternion attitude representation system to MAVs by implementing several innovations to account for MAV specific sensors and environmental factors. The flight simulations demonstrate the MEKF to be highly accurate in attitude tracking; the ground tests of the hardware show that the MEKF algorithm functions properly when using real sensors; and the flight results confirm that the MEKF can be used in actual flight conditions. Additionally, the quaternion MEKF attitude estimation allows for the development of full-envelope maneuvers.

Chapter 4

Trajectory Maneuvers

Having developed a full-envelope attitude estimation scheme, the next step is to develop a methodology for creating desired maneuvers. Our approach is to develop a maneuver definition methodology that specifies the desired attitude as a function of time and then develop a control law that tracks this trajectory.

4.1 Background

Aircraft aerobatic maneuvers are defined as a transition from one steady-state flight condition to another by passing through a sequence of desired locations with an associated desired attitude [35, 36]. Therefore, the maneuvers need to meet boundary conditions for position and heading at the beginning and end so as to smoothly transition to and from the maneuver. With these smooth transitions, maneuvers can connect segments of steady flight and the concept of a library of maneuvers, as presented in [37] for helicopters, would be desirable for autonomous, fixed-wing aircraft. In [38], each maneuver in the library was constructed using flight data to build polynomial splines of position and orientation. A fifth-order polynomial [39] can be used to generate trajectories in pitch angle or roll angle individually, that have smooth transitions from steady to maneuvering flight. This method can be utilized to generate trajectories for loop and aileron roll maneuvers, and by connecting the loop and aileron roll maneuvers in series, one can command an Immelmann or Split-S. Similar polynomial-based aircraft maneuver trajectories were demonstrated in [40, 41] where higher-order polynomials were generated using virtual arcs and aircraft thrust histories. Similarly, horizontal trajectories for obstacle avoidance were generated using stream functions in [42]. For maneuvers that incorporate simultaneous variation of the pitch and roll angles, a methodology for obtaining the aircraft rotation matrix from the first and second derivatives of desired position, as presented in [10], can be exploited. This approach is based on assuming that the aircraft velocity is always aligned with the body-frame $\mathbf{x}$-axis, an assumption that allows a differentially flat mathematical model [43, 44, 45, 46] of aircraft flight to be derived.

We develop an approach for constructing aerobatic maneuvers using continuously differentiable functions subject to constraints on position and velocity at the beginning and ending of the maneuver. Applying the maneuver boundary conditions, we demonstrate the ability to create many different aerobatic maneuvers, including basic maneuvers such as aileron rolls and loops, as well as several MAV-specific obstacle avoidance maneuvers. Each of these maneuvers can be commanded by generating the desired attitude at each instant in time according to continuous functions that meet the boundary conditions of initial and final position and velocity.

4.2 Attitude Trajectories

The maneuvers in our library were created using both fifth-order polynomials and trigonometric functions. The motivation for applying both approaches is that the aileron roll maneuver cannot be built as a trigonometric function. An idealized aileron roll never deviates from a single heading, altitude, or velocity, meaning that the aircraft is not experiencing any linear accelerations; because there are no linear accelerations, the kinematic constraints allow flight in a straight line at any roll rate. Thus, the polynomial trajectory in roll angle is the correct way to generate the commands for an aileron roll maneuver.

4.2.1 Polynomial Functions

A time-parameterized trajectory for a loop maneuver and an aileron roll maneuver can be generated using a fifth-order polynomial function [39] in pitch angle or roll angle, respectively. These polynomial functions have the attractive property of being easily generated at each time step in the autopilot code, and the desired angle and its first derivative are continuous, smooth functions. For the aileron roll maneuver, the functions for the desired roll angle and desired roll rate are

$$\begin{aligned}\phi_d(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \qquad (4.1)\\ \dot{\phi}_d(t) &= a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4, \qquad (4.2)\end{aligned}$$

where the subscript d indicates the desired value. For a lateral-axis-only maneuver (aileron roll), the derivative of the roll angle, $\dot{\phi}$, is approximately equal to the angular rate about the body-frame $\mathbf{x}$-axis, denoted as p. The polynomial coefficients $(a_n,\ n = 1, ..., 5)$ are calculated from the boundary conditions on the roll angle and its first two derivatives, according to the equations:

$$\begin{aligned}a_0 &= \phi_0\\ a_1 &= \dot{\phi}_0\\ a_2 &= \frac{\ddot{\phi}_0}{2}\\ a_3 &= \frac{20(\phi_f - \phi_0) - \left(8\dot{\phi}_f + 12\dot{\phi}_0\right)t_f + \left(\ddot{\phi}_f - 3\ddot{\phi}_0\right)t_f^2}{2t_f^3}\\ a_4 &= \frac{30(-\phi_f + \phi_0) + \left(14\dot{\phi}_f - 16\dot{\phi}_0\right)t_f - \left(2\ddot{\phi}_f - 3\ddot{\phi}_0\right)t_f^2}{2t_f^4}\\ a_5 &= \frac{12(\phi_f - \phi_0) - \left(6\dot{\phi}_f + 6\dot{\phi}_0\right)t_f + \left(\ddot{\phi}_f - \ddot{\phi}_0\right)t_f^2}{3t_f^5},\end{aligned}$$

where the subscript 0 indicates an initial condition $(t = 0)$ and subscript f indicates a final condition $(t = t_f)$.

With this formulation, an aileron roll maneuver can be flown by generating the desired roll angle and first derivative as a function of time and specifying that $\phi_0 = 0$ and $\phi_f = 2\pi$. Thus, the trajectory in roll angle completes a full revolution and the aircraft tracking these angles will fly an aileron roll maneuver. The aggressiveness of the maneuver is specified by t_f, where a smaller final time results in faster turning. For the MAGICC lab MAVs, there is not a method to make an accurate analytical prediction of the maneuver envelope; hence, the limits on t_f were found through multiple trials of simulated flight maneuvers.

Similarly, a loop maneuver can be flown by following a time-parameterized fifth-order polynomial function for pitch angle and its time derivative, which can be written as

$$\theta_d(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{4.3}$$

$$\dot{\theta}_d(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4, \tag{4.4}$$

The loop is a purely longitudinal-mode maneuver, implying that $\dot{\theta}_d$ is equal to the body-frame angular rate about the $\mathbf{y}$-axis, denoted as q. By tracking the attitude trajectory for desired pitch angle ($\theta_d = 0 \rightarrow 2\pi$), the aircraft will fly a loop maneuver, where the maneuver aggressiveness is again specified by t_f. As with the aileron roll maneuver, the range of t_f was found by trial and error using the flight simulator.

The individual aileron roll and loop maneuvers can be combined sequentially to create more interesting maneuvers such as the Immelmann. Fighter pilots developed the Immelmann maneuver to trade airspeed for altitude while reversing directions; it is accomplished by flying a half loop ($\theta_d = 0 \rightarrow \pi$) followed by a half roll ($\phi_d = -\pi \rightarrow 0$).

As mentioned previously, the aggressiveness of these maneuvers is fully specified by the maneuver duration, t_f, the limits of which are dictated by the turning performance and the thrust characteristics of each individual aircraft. The primary shortcoming of the polynomial attitude trajectories is the inability to specify a maneuver that is a simultaneous combination of pitch and roll angles. Therefore, a novel approach was developed that uses trigonometric functions to describe time-parameterized paths from which a desired attitude can be calculated at each instant in time.

4.2.2 Trigonometric Functions

Building a three-dimensional aerobatic flight maneuver requires simultaneous knowledge of both pitch and roll angles, which can be derived from functions of position and its derivatives in the inertial $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$-axes. These time-parameterized trigonometric functions can be created using boundary conditions for position and velocity at the start and finish of the maneuver.

Time-Parameterized Position Functions

For the purposes of MAV obstacle avoidance, we devised a three-dimensional aerobatic maneuver that we refer to as the Close-Q (see Figure 4.1), which can be thought of as a loop with a quarter twist. This maneuver would be used for making a 90-degree turn in a minimum amount of horizontal space. The Close-Q boundary conditions are aircraft position at maneuver start–level flight pointing in any compass direction and in the exact same position at the end of the maneuver, pointing in a direction ± 90 degrees from the initial heading, depending on the desired final direction. The example boundary conditions shown in Figure 4.1 are aircraft position at maneuver start with wings-level, pointing north and in the exact same position at the end of the maneuver, with wings level, pointing in the negative east direction. The vectors in Figure indicate the aircraft body-frame $\mathbf{x}$ and $\mathbf{z}$-axes.

Given these boundary conditions, the vector of functions specifying desired position for a Close-Q maneuver can be written as

$$\mathbf{p}_d = \begin{bmatrix} \frac{1}{2}r\sin(\omega t) + r\sin\left(\frac{1}{2}\omega t\right) \\ \frac{1}{2}r\sin(\omega t) - r\sin\left(\frac{1}{2}\omega t\right) \\ -r + r\cos(\omega t) \end{bmatrix}, \tag{4.5}$$

where r is the maneuver radius; t is the maneuver time, which varies from $(0 \rightarrow 2\pi/\omega)$; and ω is the maneuver rate, which is the ratio of nominal flight velocity and maneuver radius $(\omega = V/r)$. Note that the $\mathbf{z}$-axis function is defined according to the positive-down convention and should not be confused with altitude, which is defined as positive

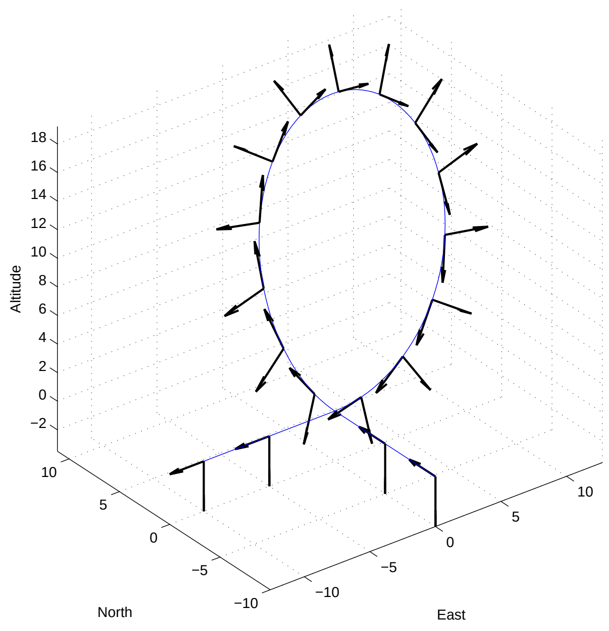


Figure 4.1: Close-Q Maneuver Path. The Close-Q maneuver can be flown to minimize the projected footprint when making a 90-degree turn.

up. The maneuver aggressiveness variable for the trigonometric functions is the maneuver radius, r. As with the polynomial trajectories, the range of r is limited by the aircraft flight envelope and was computed experimentally. By analyzing the functions in each inertial frame axis individually, as shown in Figure 4.2, we can gain insight into the construction of the maneuver. Note that by adding trigonometric functions, the $\mathbf{x}_i$ position starts and ends at zero, but the slope at $t = 0$ is unity, indicating that all of the airspeed is in the $\mathbf{x}_i$ direction at the start of the maneuver. Similarly, by subtracting the same two trigonometric functions for $\mathbf{y}_i$ position, the maneuver again starts and ends at zero, but the slope of the position curve at $t = 0$ is zero and at $t = t_f$, the slope is unity, indicating that all of the airspeed at the end of the maneuver is in the $\mathbf{y}_i$ direction. The plot of $\mathbf{z}_i$ position starts and ends at zero, with zero slope at both instants in time.

Given this framework of position and velocity constraints, one may derive a functional description for many desired maneuvers, such as loops, barrel rolls, etc.

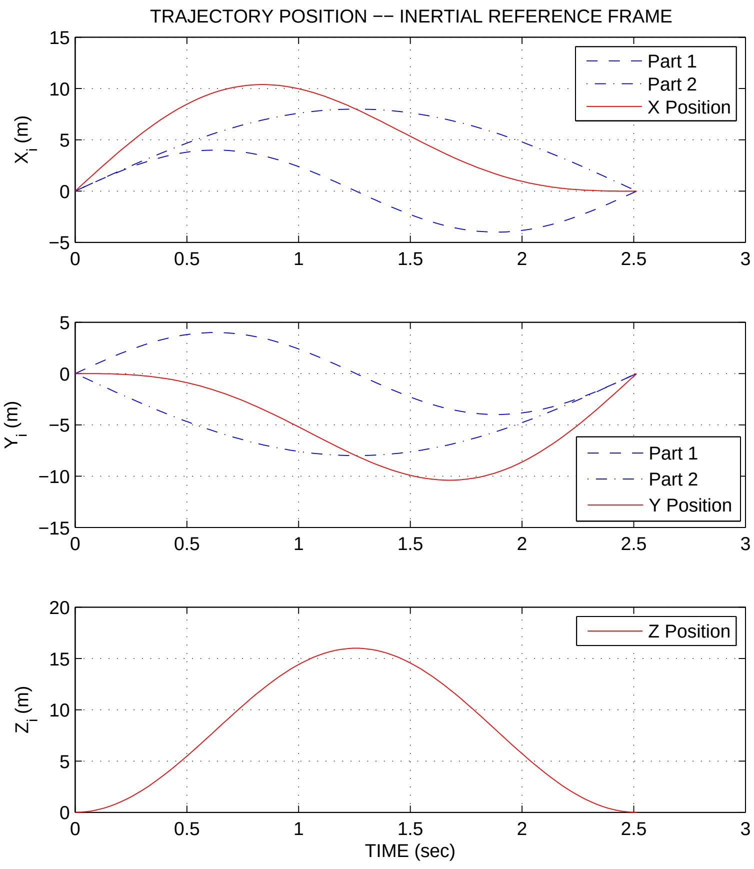


Figure 4.2: Close-Q Manuever Path Functions. The trigonometric functions for position in each inertial frame axis are constructed by adding various functions to meet the boundary condition constraints. Note that the sign is reversed on the $\mathbf{z}_i$ plot to give a more intuitive view of the third component of the maneuver.

Additionally, the Close-Q maneuver can be used to execute a right-angle turn to the left by simply changing the sign on the **y**-axis function.

Generating Combined Pitch and Roll Trajectories

The process of generating attitude trajectories that simultaneously specify pitch and roll angles involves the following steps: 1) determine a trigonometric function that specifies the desired time-parameterized position in inertial space and meets the boundary conditions for position and velocity; 2) find the first and second derivatives of this function and ensure attitude boundary conditions are satisfied; 3) calculate the desired body-to-inertial reference frame rotation matrix; 4) determine the desired pitch and roll angles; and 5) compute the desired body-frame angular rates. The derivation of this process is based on the assumption that the aircraft velocity vector is aligned with the body-frame **x**-axis and aircraft kinematics. These kinematic constraints are specifically smooth transitions to and from steady, level flight, maximum and minimum limits on airspeed, and limits on the curvature of turns.

Step 1 – Time-Parameterized Position Function This is accomplished by first specifying the desired, inertial reference frame, **x**, **y**, and **z**-axis positions as continuously differentiable functions of time subject to the boundary conditions for position and velocity. Using the Close-Q maneuver as an example, the position functions are as shown in Equation (4.5). As noted earlier for the Close-Q maneuver, each axis position function starts and ends at the current aircraft north-east-altitude position, in keeping with the position boundary conditions. The velocity boundary conditions are met in Step 2.

Step 2 – Position Function Time Derivatives Taking the first and second derivatives of the Close-Q position functions (4.5) yields

$$\dot{\mathbf{p}}_d = \begin{bmatrix} \frac{1}{2}V\cos(\omega t) + \frac{1}{2}V\cos\left(\frac{1}{2}\omega t\right) \\ \frac{1}{2}V\cos(\omega t) - \frac{1}{2}V\cos\left(\frac{1}{2}\omega t\right) \\ V\sin(\omega t) \end{bmatrix} \tag{4.6}$$

and

$$\ddot{\mathbf{p}}_d = \begin{bmatrix} -\frac{1}{2}\omega V\sin(\omega t) - \frac{1}{4}\omega V\sin\left(\frac{1}{2}\omega t\right) \\ -\frac{1}{2}\omega V\sin(\omega t) + \frac{1}{4}\omega V\sin\left(\frac{1}{2}\omega t\right) \\ \omega V\cos(\omega t) \end{bmatrix}, \tag{4.7}$$

where V is the nominal airspeed of the aircraft. Examining $\dot{\mathbf{p}}_d$, one can see that at time $t = 0$ the velocity is all in the inertial frame **x**-axis and at time $t = t_f$ the velocity is in the **y**-axis, thus meeting the boundary conditions for velocity. Using the expressions for the inertial components of velocity and acceleration allows the derivation of the tangent and normal vector of our position curve. This fact will be exploited in the next step.

Step 3 - Rotation Matrix from First and Second Derivatives The procedure for finding the rotation matrix $\left(\mathbf{R} = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \end{bmatrix}\right)$ is to find the first and third columns of the matrix using kinematic relationships and then use the vector product to find the remaining column. The first column of the rotation matrix is found by enforcing the assumption that the velocity vector aligns with the body-frame $\mathbf{x}$-axis. This implies that $\mathbf{c}_1$ (a unit-length vector) points in the direction of the aircraft velocity vector.

Rotation Matrix, Column 1 ($\mathbf{c}_1$) The first column, $\mathbf{c}_1$, of the rotation matrix can be derived from the expression for the velocity vector alignment assumption. Given an airspeed, V, and an inertial-frame unit vector $\hat{\mathbf{e}}_x = \begin{bmatrix} 1, & 0, & 0 \end{bmatrix}^T$, the assumption that the velocity vector aligns with the $\mathbf{x}$-axis can be written as

$$\dot{\mathbf{p}} = V\mathbf{R}\hat{\mathbf{e}}_x, \tag{4.8}$$

which can be expanded to yield

$$\dot{\mathbf{p}} = V \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Performing the multiplication and solving for $\mathbf{c}_1$ yields

$$\mathbf{c}_1 = \frac{\dot{\mathbf{p}}}{V}, \tag{4.9}$$

where $V = \|\dot{\mathbf{p}}\|$ is the airspeed of the MAV.

Rotation Matrix, Column 3 ($\mathbf{c}_3$) To find the third column, $\mathbf{c}_3$, we use the dynamic expression for the second derivative of the inertial reference frame position vector, $\mathbf{p}$, which can be written as,

$$\ddot{\mathbf{p}} = \mathbf{g} + \mathbf{R}\mathbf{a}_b, \tag{4.10}$$

where $\mathbf{g}$ is the inertial frame gravity vector $\left(\begin{bmatrix} 0, & 0, & g \end{bmatrix}^T\right)$, $\mathbf{R}$ is the body-to-inertial rotation matrix, and $\mathbf{a}_b$ is the vector of body-frame accelerations. Differentiating the rotation matrix orthogonality constraint $\left(\mathbf{I} = \mathbf{R}\mathbf{R}^T\right)$ yields a kinematic relationship for the time-rate-of-change of the rotation matrix,

$$\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\omega}}, \tag{4.11}$$

where $\hat{\boldsymbol{\omega}}$ is a skew-symmetric matrix of body-frame angular rates,

$$\hat{\boldsymbol{\omega}} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}. \tag{4.12}$$

where p, q, and r are defined as the angular rates about the $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$-axis, respectively.The third column of the rotation matrix is found from Equation (4.10), which can be written as

$$\begin{aligned} \ddot{\mathbf{p}} - \mathbf{g} &= \mathbf{R}\mathbf{a}_b \\ &= \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 \end{bmatrix} \begin{bmatrix} a_{bx} \\ 0 \\ a_{bz} \end{bmatrix} \end{aligned} \tag{4.13}$$

Recalling that the columns of the rotation matrix are orthogonal unit vectors, the following two properties for inner products are true:

$$\mathbf{c}_i^T \mathbf{c}_j = 0, \text{ if } \ i \neq j$$

$$\mathbf{c}_i^T \mathbf{c}_j = 1, \text{ if } \quad i = j.$$

Thus, we can solve Equation (4.13) by multiplying both sides by the vector $\mathbf{c}_1^T$ calculated in Equation (4.9) to yield

$$\begin{aligned} \mathbf{c}_1^T (\ddot{\mathbf{p}} - \mathbf{g}) &= a_{bx} \mathbf{c}_1^T \mathbf{c}_1 + a_{bz} \mathbf{c}_1^T \mathbf{c}_3 \\ \mathbf{c}_1^T (\ddot{\mathbf{p}} - \mathbf{g}) &= a_{bx} + 0, \end{aligned}$$

which can be substituted back into Equation (4.13) to yield

$$\begin{aligned} \ddot{\mathbf{p}} - \mathbf{g} &= \mathbf{c}_1^T (\ddot{\mathbf{p}} - \mathbf{g}) \mathbf{c}_1 + a_{bz} \mathbf{c}_3 \\ a_{bz} \mathbf{c}_3 &= \ddot{\mathbf{p}} - \mathbf{g} - \mathbf{c}_1 \mathbf{c}_1^T (\ddot{\mathbf{p}} - \mathbf{g}) \\ a_{bz} \mathbf{c}_3 &= \left(\mathbf{I} - \mathbf{c}_1 \mathbf{c}_1^T \right) (\ddot{\mathbf{p}} - \mathbf{g}) . \end{aligned}$$

Physically, it can be seen that the matrix $\left(\mathbf{I} - \mathbf{c}_1 \mathbf{c}_1^T \right)$ projects the acceleration vector onto the unit vector perpendicular to $\mathbf{c}_1$. The magnitude of the acceleration is found from the norm of $\left(\mathbf{I} - \mathbf{c}_1 \mathbf{c}_1^T \right) (\ddot{\mathbf{p}} - \mathbf{g})$ and its direction is negated to account for the aircraft definition of positive $\mathbf{z}$-axis pointing out the aircraft belly. Thus, the third column of the rotation matrix can be written

$$\mathbf{c}_3 = \frac{\left(\mathbf{I} - \mathbf{c}_1 \mathbf{c}_1^T \right) (\ddot{\mathbf{p}}_d - \mathbf{g})}{-\| (\mathbf{I} - \mathbf{c}_1 \mathbf{c}_1^T) (\ddot{\mathbf{p}}_d - \mathbf{g}) \|}, \tag{4.14}$$

where $\mathbf{I}$ represents a 3×3 identity matrix.

Finally, the second column of the rotation matrix is computed using the cross product

$$\mathbf{c}_2 = \mathbf{c}_3 \times \mathbf{c}_1. \tag{4.15}$$

Using this procedure, the rotation matrix can be computed at each instant of time.

Step 4 - Desired Euler Angles Once the rotation matrix is computed, Equation (3.2) is used to compute the desired pitch angle, θ, and Equation (3.3) is used to compute the desired roll angle, ϕ.

Step 5 - Desired Angular Rates We derive expressions for the body-frame angular rates, p and q, starting with the derivative of the coordinated flight condition shown in Equation (4.8), which can be written as

$$\ddot{\mathbf{p}} = \dot{V}\mathbf{R}\hat{\mathbf{e}}_x + V\dot{\mathbf{R}}\hat{\mathbf{e}}_x. \tag{4.16}$$

Setting this relationship equal to the right side of Equation (4.10) yields

$$\mathbf{g} + \mathbf{R}\mathbf{a}_b = \dot{V}\mathbf{R}\hat{\mathbf{e}}_x + V\dot{\mathbf{R}}\hat{\mathbf{e}}_x,$$

which can be expanded according to

$$\begin{aligned}
\mathbf{g} + \mathbf{R}\mathbf{a}_b &= \dot{V}\mathbf{R}\hat{\mathbf{e}}_x + V\dot{\mathbf{R}}\hat{\mathbf{e}}_x \\
\mathbf{g} + \mathbf{R}\left(a_{bx}\hat{\mathbf{e}}_x + a_{bz}\hat{\mathbf{e}}_z\right) &= \dot{V}\mathbf{R}\hat{\mathbf{e}}_x + V\mathbf{R}\hat{\boldsymbol{\omega}} \times \hat{\mathbf{e}}_x \\
\mathbf{g} + \mathbf{R}\left(a_{bx}\hat{\mathbf{e}}_x + a_{bz}\hat{\mathbf{e}}_z\right) &= \dot{V}\mathbf{R}\hat{\mathbf{e}}_x + V\mathbf{R}\left(r\hat{\mathbf{e}}_y - q\hat{\mathbf{e}}_z\right) \\
\mathbf{R}^T\mathbf{g} + a_{bx}\hat{\mathbf{e}}_x + a_{bz}\hat{\mathbf{e}}_z &= \dot{V}\hat{\mathbf{e}}_x + Vr\hat{\mathbf{e}}_y - Vq\hat{\mathbf{e}}_z \\
\mathbf{R}^T\mathbf{g} + a_{bx}\hat{\mathbf{e}}_x + a_{bz}\hat{\mathbf{e}}_z - \dot{V}\hat{\mathbf{e}}_x - Vr\hat{\mathbf{e}}_y + Vq\hat{\mathbf{e}}_z &= 0.
\end{aligned}$$

This expression can be isolated into $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ components, where the $\mathbf{z}$-axis component yields the desired angular rate about the body-frame $\mathbf{y}$-axis. This desired angular rate can be written as

$$q_d = \frac{-\left(g_{bz} + a_{bz}\right)}{V}, \tag{4.17}$$

where g_{bz} is the component of gravity in the body-frame $\mathbf{z}$-axis.

The body-frame **x**-axis desired angular rate, p, is derived by differentiating Equation (4.10), yielding the third derivative of position,

$$\begin{aligned} \mathbf{p}^{(3)} &= \dot{\mathbf{g}} + \dot{\mathbf{R}}\mathbf{a}_b + \mathbf{R}\dot{\mathbf{a}_b} \\ &= 0 + \mathbf{R}\left(\boldsymbol{\omega} \times \mathbf{a}_b + \dot{\mathbf{a}_b}\right) \\ &= \mathbf{R}\left(\begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}\begin{bmatrix} a_{bx} \\ 0 \\ a_{bz} \end{bmatrix} + \begin{bmatrix} \dot{a_{bx}} \\ 0 \\ \dot{a}_{bz} \end{bmatrix}\right) \\ &= \mathbf{R}\left(\begin{bmatrix} q a_{bz} \\ r a_{bx} - p a_{bz} \\ -q a_{bx} \end{bmatrix} + \begin{bmatrix} \dot{a_{bx}} \\ 0 \\ \dot{a}_{bz} \end{bmatrix}\right). \end{aligned} \tag{4.18}$$

Equation (4.18) can be rearranged and reduced to vector components (using the unit vector $\hat{\mathbf{e}}_y = \begin{bmatrix} 0, & 1, & 0 \end{bmatrix}^T$) to yield

$$\begin{aligned} p_d &= \frac{1}{a_{bz}}\left(r a_{bx} + \hat{\mathbf{e}}_y^T \mathbf{R}^T \mathbf{p}^{(3)}\right) \\ &= \frac{1}{a_{bz}}\left(r a_{bx} + \mathbf{p}_{by}^{(3)}\right), \end{aligned} \tag{4.19}$$

the desired body-frame angular rate about the **x**-axis.

Examination of the equations for the trigonometric approach to trajectory generation makes it evident why this approach cannot be used for the aileron roll maneuver; the roll angle is varied without changing the direction of the velocity vector as needed in Equation (4.9) and without any body-frame accelerations as needed in Equation (4.14). This is because in the differentially flat aircraft flight model, roll rate is independent of the body-frame accelerations as long as the velocity vector alignment assumption holds. A complete library of maneuvers should incorporate the ability to command aileron rolls as well as three-dimensional maneuvers, thus both the polynomial and trigonometric approaches are important.

Given the desired pitch and roll angles and the body-frame angular rates, we can build a feedback control law to track the attitude trajectories to autonomously fly aerobatic maneuvers.

4.2.3 Additional Maneuvers

As described previously, the functions shown in Equations (4.5), (4.6), and (4.7) are the trajectory and derivatives for a Close-Q maneuver. Similarly, the functions for a loop maneuver can be written as

$$\begin{aligned} \mathbf{p}_d &= \begin{bmatrix} r\sin(\omega t) \\ 0 \\ -r + r\cos(\omega t) \end{bmatrix} \\ \dot{\mathbf{p}}_d &= \begin{bmatrix} V\cos(\omega t) \\ 0 \\ -V\sin(\omega t) \end{bmatrix} \\ \ddot{\mathbf{p}}_d &= \begin{bmatrix} -\omega V\sin(\omega t) \\ 0 \\ -\omega V\cos(\omega t) \end{bmatrix}, \end{aligned} \tag{4.20}$$

where the aggressiveness of the maneuver is determined by the radius, r, and the duration is $2\pi/\omega$. The first portion of an Immelmann maneuver uses the exact same equations as the loop, however, the duration is cut in half, π/ω, and the controller commands a half roll based on a polynomial trajectory in roll angle.

4.3 Attitude Tracking Feedback Control Law

Defining a maneuver as the transition between two steady-state flight conditions leads naturally to a simple and effective proportional-derivative (PD) maneuver control law (see Figure 4.3) that can be tuned for a specific aircraft and maneuver aggressiveness.

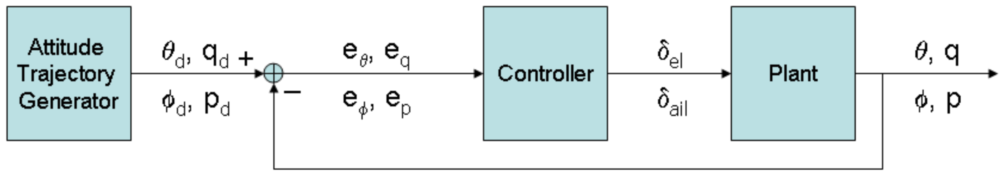


Figure 4.3: Trajectory Tracking Controller Block Diagram. The trajectory generation and tracking control law for autonomous trajectory tracking aerobatic maneuvers.

The control law can be written as

$$\delta_{el} = k_\theta (\theta_d - \theta) + k_q (q_d - q) \quad (4.21)$$

$$\delta_{ail} = k_\phi (\phi_d - \phi) + k_p (p_d - p) \quad (4.22)$$

where the actual values of θ, ϕ, p and q are aircraft state variables, and k_θ, k_q, k_ϕ and k_p are the feedback gains. These four gains must be tuned for the specified maneuver aggressiveness and airframe turn rate and thrust characteristics. For loop, aileron roll, and Immelmann maneuvers, the δ_{el} and δ_{ail} are entirely independent of each other so only two gains are tuned for each of these maneuvers. However, for the Close-Q maneuver, all four gains need to be adjusted simultaneously to achieve good tracking of the desired aircraft orientation. In addition to tuning the controller gains, the maneuver aggressiveness variables (maneuver duration or maneuver radius) need to be adjusted to ensure the aircraft has sufficient thrust and control authority to complete the maneuver. For example, the MAV used in this research cannot turn a loop faster than 2 seconds for lack of control authority and has insufficient thrust available to fly a 5-second loop.

4.4 Results

The trajectory generation algorithms and maneuver tracking controller were developed in simulation where the gains were tuned prior to being flight tested.

4.4.1 Simulation Results

Flight simulations of various maneuvers commanded by the attitude trajectory generation algorithms and tracking controller were conducted using the Aviones flight simulator. The aileron roll maneuver was executed using the polynomial functions to generate the attitude trajectories, and the loop, Immelmann and Close-Q maneuvers were performed using the combined attitude trajectory based on trigonometric functions. The maneuver aggressiveness variables are annotated with each figure. For each maneuver the starting altitude was 100 meters, the flight airspeed was 13 meters per second, and the initial heading was north.

The simulation results for the aileron roll maneuver show that this idealized model is not completely accurate; because the velocity vector is not perfectly aligned with the body-frame **x**-axis, it produces a loss of altitude when the aircraft is inverted. Additionally, there is a loss of lift when the roll angle approaches 90 and 270 degrees, contributing to the loss of altitude shown in Figure 4.4.

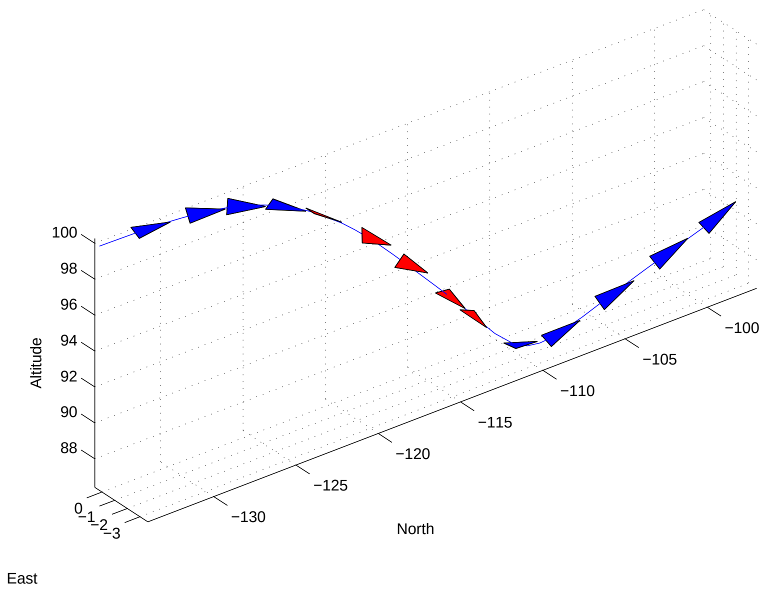


Figure 4.4: Trajectory Tracking Aileron Roll Simulation. An aileron roll maneuver ($t_f = 2$) was flown autonomously in the flight simulator. The bi-colored triangles indicate the aircraft orientation at various positions.

As mentioned, the aileron roll maneuver was accomplished using the fifth-order polynomial trajectories, however, the loop, Immelmann and Close-Q maneuvers shown in Figures 4.5- 4.7 were generated using the trigonometric functions for desired attitude and angular rates. The results for a loop maneuver are shown in Figure 4.5. Note that although a loop maneuver was flown, the shape of the path followed by the aircraft is not a perfect circle and the final altitude is below the initial level. This reflects the fact that the controller has worked to match the attitude specified by the trajectory generation algorithm, not necessarily to follow a circular path.

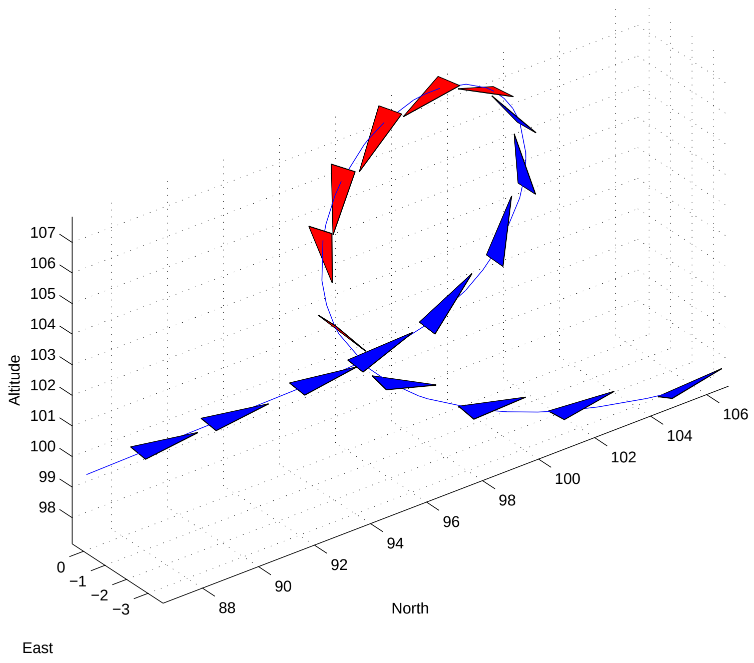


Figure 4.5: Trajectory Tracking Loop Simulation. An autonomous loop maneuver ($t_f = 2$) is shown using bi-colored triangles to indicate the aircraft orientation at various locations.

By combining the half loop with the half aileron roll in series, two new maneuvers can be performed, the Split-S and the Immelmann. Flying half of a loop followed by an half roll results in an Immelmann maneuver as shown in Figure 4.6. Notice that the aircraft was in an inverted, slightly nose-down attitude when the aileron roll portion of the maneuver was executed, causing a rapid decrease in altitude back to the nominal flight level.

The tuning of the gains to achieve accurate attitude tracking of both roll and pitch angles was extremely sensitive to errors in roll angle; a small error in roll angle resulted in greatly degraded maneuver performance. Therefore, the controller gains were tuned iteratively to ensure accurate tracking of the desired aircraft attitude during all four maneuvers with a single set of gains.

4.4.2 Flight Results

The polynomial trajectory generation algorithm was used to generate roll angle and rate commands to fly an aileron roll maneuver. The aircraft was flying a constant altitude, waypoint navigation, flight plan when the aileron roll maneuver was commanded. Referring to Figure 4.8, the pitch and roll angles (θ and ϕ) are plotted as a function of time. The roll maneuver can be seen in the angle ϕ rapidly increasing to 180 degrees, wrapping around to -180 degrees and proceeding back to 0. It is important to note that the same gains found through simulations were used for the flight test. The greater control authority of the actual MAV as compared to the simulator model resulted in turning the maneuver in less than two seconds and the large overshoot, as can be seen in Figure 4.8.

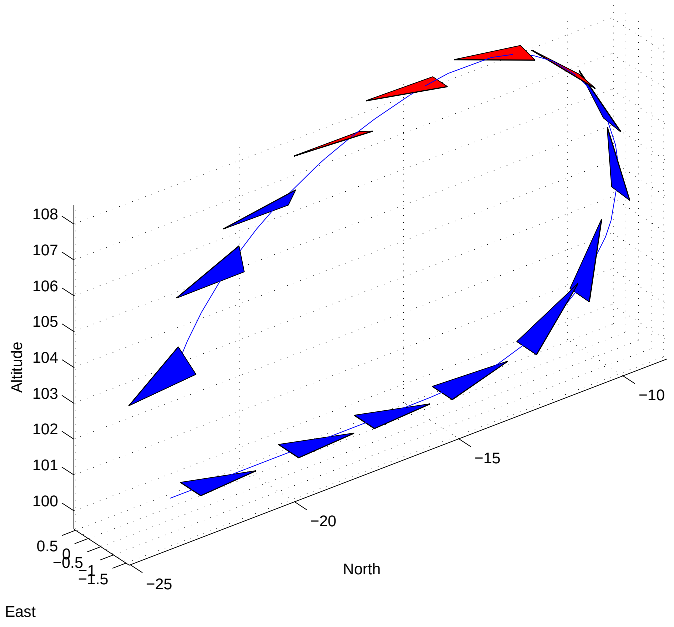


Figure 4.6: Trajectory Tracking Immelmann Simulation. An Immelmann maneuver with combined half loop maneuver duration ($t_f = 2$) and half roll maneuver duration ($t_f = 2$).

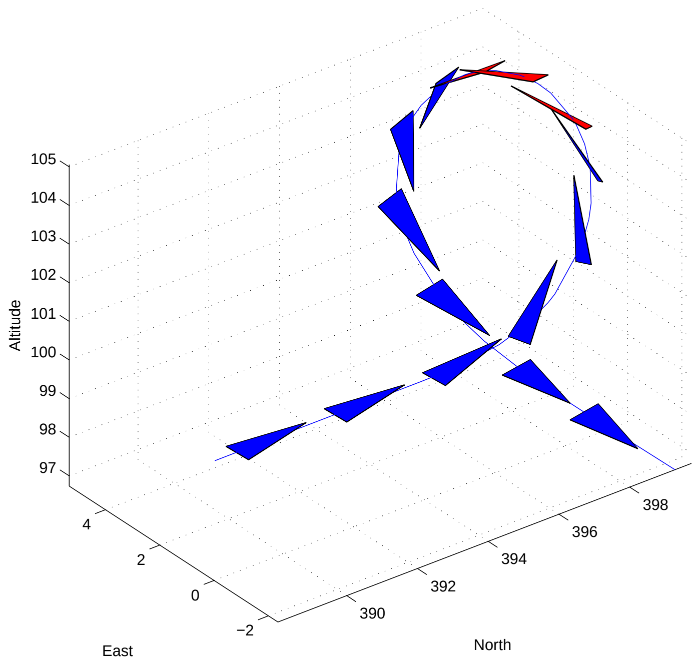


Figure 4.7: Trajectory Tracking Close-Q Simulation. The autonomous Close-Q maneuver ($r = 5$) is demonstrated making a right-hand turn. The bi-colored triangles shown the aircraft orientation throughout the maneuver.

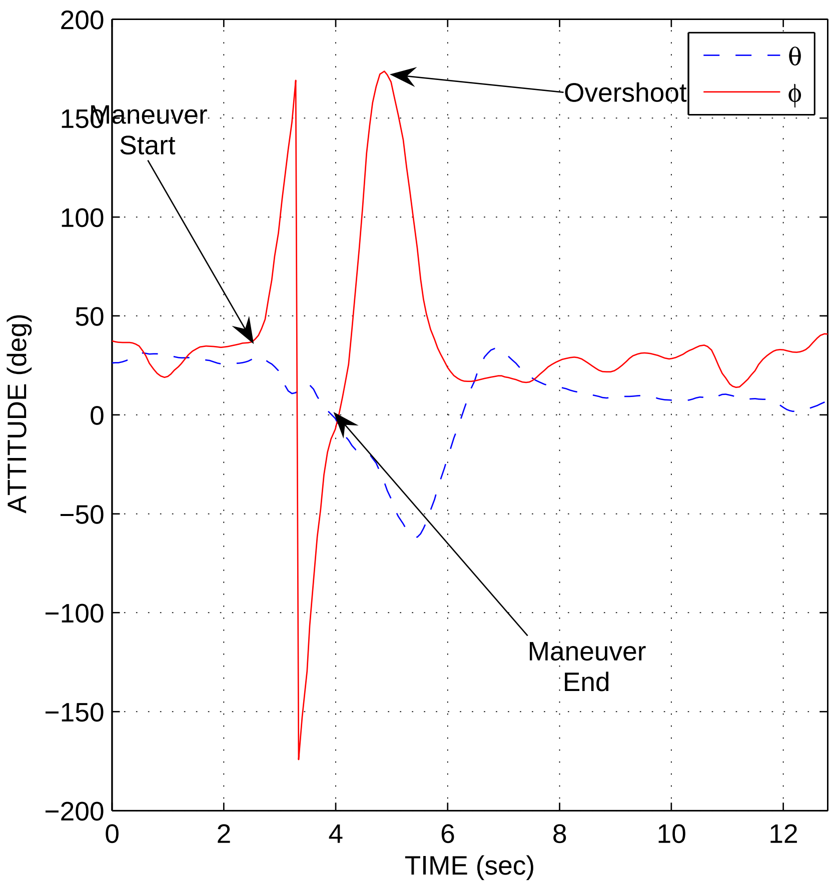


Figure 4.8: Trajectory Tracking Aileron Roll Flight Data. An autonomous aileron roll maneuver ($t_f = 2$) was flown by the MAV. The plot shows ϕ increasing up to 180 degrees, wrapping to -180 degrees and continuing back to 0, indicating a complete revolution about the **x**-axis.

The MAV experienced large control surface deflections immediately after the maneuver, see Figure 4.9, to compensate for the large overshoot in roll angle and the loss of altitude. By aggressively actuating the ailerons and elevator, the MAV was able to quickly recover to the desired altitude and course after completing the maneuver.

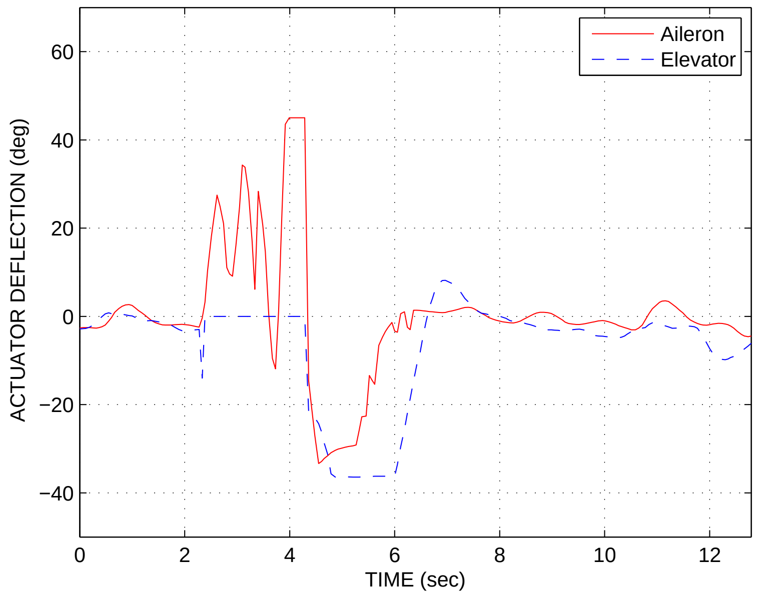


Figure 4.9: Actuator Deflections for the Trajectory Tracking Aileron Roll. The aileron and elevator deflections during the flight test of the aileron roll maneuver show the large deflection necessary to track the desired roll angle. The control surface deflections immediately after maneuver completion were caused by the aircraft working to rejoin the waypoint path and altitude.

4.5 Summary

The simulation and flight results for the various maneuvers demonstrated the ability of the two trajectory generation algorithms, polynomial and trigonometric, to command maneuvers using time-parameterized trajectories of attitude. The results also validate the performance of the attitude tracking control law in commanding the aircraft through the maneuvers. However, it should be noted that because position error is not included in the control law, this approach is inappropriate for obstacle avoidance maneuvering, where disturbances such as wind would cause the aircraft to track a different path than the desired one.

Chapter 5

Path Following

The goal of aerobatic maneuvering for MAV obstacle avoidance is to minimize the footprint of the maneuver. The maneuver footprint is defined as the projection of the maneuver onto the horizontal plane. The desired minimization must be accomplished while still adhering to the kinematic and dynamic constraints of the aircraft. Anderson [5] defines the turning performance of an aircraft in terms of the load factor, $n = L/W$, where L is the lift produced by the aircraft and W is the aircraft weight. If an aircraft has a maximum n that is much greater than unity, the equation that models an aircraft turn radius can be written as

$$R = \frac{V_\infty^2}{gn}, \tag{5.1}$$

where V_∞ is defined as the magnitude of the aircraft velocity. Given that a specific airframe will have a maximum load factor, n, there is a minimum turn radius that is independent of the plane in which the turn is made. From simple intuition one can think of the shadow cast by a coin onto a table; if the coin is flat the shadow is large but if turned on its edge, the shadow is much smaller. Similarly, a minimum-radius turn in the horizontal plane will have a larger footprint than one in which the vertical direction is exploited.

5.1 Background

Several different approaches have been investigated to solve the problem of following a specified path in space. Both straight and curved horizontal paths were followed using a sliding mode control law in [47, 11], which has been demonstrated extensively in autonomous flights of MAVs. A proportional-navigation control law based on visual tracking of roads was demonstrated in [48]. The pure-pursuit method demonstrated in [4] uses a nonlinear proportional navigation control law. Nonlinear backstepping was utilized in [49] to create a controller for maneuver regulation along sinusoidal paths. In [38], the path was bisected by control planes that tracked the aircraft progress along the path. For the innovative work in [12], the desired path was a single hover point and the control law was a PID controller built in a unit-norm quaternion formulation. The common link for each of these approaches is the control law was based on the error between the aircraft position and the desired path. We develop a methodology to compute the position error and its first derivative in the inertial reference frame. These error vectors are transformed into the body-frame and used to create a PID control law. The three control elements, proportional, integral and derivative, are mapped into a unit-norm quaternion and composed together using quaternion algebra to form the MAV controller commands.

5.2 Path Position and Derivative

5.2.1 Minimum Distance Position Parameter

The keystone of the path following control law is determining the location of the path that is nearest to the aircraft's actual position. To do this, the path is parameterized by

$$s(t) = (0 \rightarrow 1),$$

which can be thought of as a spatial, rather than temporal, variable. The challenge is to compute the value of the path parameterizing variable s at the point on the path that is the minimum distance, $\mathbf{D}_{\text{min}}$, from the aircraft position. Our approach to solving this problem is to use the value of s and the aircraft location, $\mathbf{p}_a$, from the previous time step combined with the rate of change of the path with respect to s and the aircraft velocity to predict the change in the path parameter, Δs. A graphical representation of the predictive-corrective path parameter calculation problem set-up is shown in Figure 5.1.

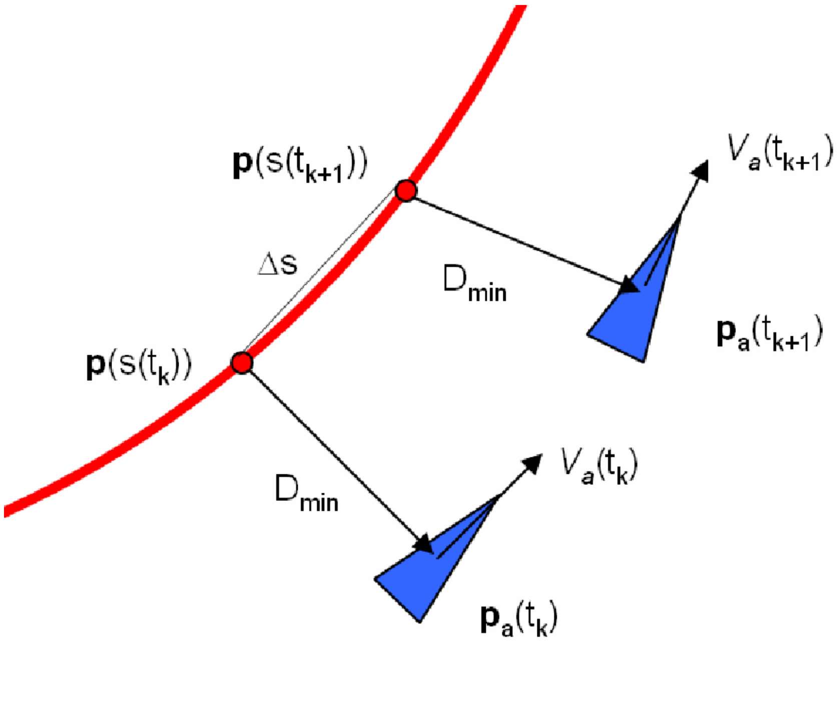


Figure 5.1: Path Parameter Calculation. The path parameter is calculated using a predictive-corrective algorithm that uses the minimum distance to determine the path location with respect to the aircraft.

Given the position of the aircraft at some instant in time, $\mathbf{p}_a(t_k)$, the change in position during the interval of time, $\Delta t = t_{k+1} - t_k$, can be written as $\Delta \mathbf{p}_a = (\mathbf{p}_a(t_{k+1}) - \mathbf{p}_a(t_k))$. For the next instant in time, the minimum distance can be found from the squared difference between the previous aircraft position plus its change in position and the corresponding previous path location, $\mathbf{p}(s(t_k))$, plus the corresponding change in desired path location, $\frac{\partial \mathbf{p}}{\partial s}\Delta s$. The equation for the minimum squared distance is given by

$$\begin{aligned}\mathbf{D}_{\min}(s) &= \min_{\Delta s} \| (\mathbf{p}_a(t_k) + \Delta \mathbf{p}_a) - \left(\mathbf{p}(s(t_k)) + \frac{\partial \mathbf{p}}{\partial s}\Delta s\right) \|^2 \\ &= \min_{\Delta s} \| -\frac{\partial \mathbf{p}}{\partial s}\Delta s + (-\mathbf{p}(s(t_k)) + \mathbf{p}_a(t_k) + \Delta \mathbf{p}_a) \|^2\end{aligned}$$

$$\begin{aligned} &= \min_{\Delta s} \left[\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right) \Delta s^2 - 2 \left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right) \Delta s \right] + \\ &\quad \left[\left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right)\right]. \end{aligned} \tag{5.2}$$

The value of Δs that yields the minimum distance can be found by setting the derivative of Equation (5.2) with respect to Δs equal to zero, yielding

$$2\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right) \Delta s - 2\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right) = 0. \tag{5.3}$$

Simplifying and solving Equation (5.3) for Δs yields

$$\Delta s = \frac{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right)}{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right)}. \tag{5.4}$$

We can find the time rate of change of the path parameter, $\dot{s}$, by dividing by Δt and finding the limit as $\Delta t \rightarrow 0$, which yields

$$\begin{aligned} \dot{s} &= \lim_{\Delta t \rightarrow 0} \left(\frac{\Delta s}{\Delta t}\right) \\ &= \lim_{\Delta t \rightarrow 0} \left(\frac{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right) + \Delta \mathbf{p}_a\right)}{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right) \Delta t} \right) \\ &= \frac{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \dot{\mathbf{p}}_a}{\|\frac{\partial \mathbf{p}}{\partial s}\|^2} + \Gamma \left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(-\mathbf{p}\left(s\left(t_k\right)\right) + \mathbf{p}_a\left(t_k\right)\right), \end{aligned} \tag{5.5}$$

where Γ is defined as

$$\begin{aligned} \Gamma &= \lim_{\Delta t \rightarrow 0} \left(\frac{1}{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right) \Delta t} \right) \\ &= \left(\frac{K_\Gamma}{\left(\frac{\partial \mathbf{p}}{\partial s}\right)^T \left(\frac{\partial \mathbf{p}}{\partial s}\right) \Delta t} \right), \end{aligned} \tag{5.6}$$

and the gain $K_\Gamma = \lim_{\Delta t \rightarrow 0}$ allows for tuning. The first part of Equation (5.5) predicts where the path should be at the next time step and the second term corrects for errors in the prediction and compensates for inevitable numerical errors. Finally, we ensure that the path only moves in a forward direction by setting

$$\dot{s} \;=\; \max\{0,\, \dot{s}\}\,. \tag{5.7}$$

The actual value of $s(t_{k+1})$ is found by using numerical integration to propagate Equation (5.5) subject to the constraint in Equation (5.7). This value of s is then substituted for t in the maneuver function to obtain a value for the instantaneous desired path location.

5.2.2 Desired Path Derivatives

The objective of the path following control law is to drive the error between the aircraft position and the desired path measured in the inertial reference frame to zero. However, due to the nature of the quaternion control law, the derivative of desired position needs to be determined so that a derivative error can be computed. We define the error in the position as $\tilde{\mathbf{p}} = \mathbf{p} - \mathbf{p}^d$ and the error in the derivative of position as $\dot{\tilde{\mathbf{p}}} = \dot{\mathbf{p}} - \dot{\mathbf{p}}^d$. However, for the spatially-parameterized path, the path function should be written as $\mathbf{p}(s(t))$, although were drop the $(s(t))$ for succinctness. The first two derivatives of the path can be written as

$$\dot{\mathbf{p}}^d \;=\; \dot{s}\frac{d\mathbf{p}^d}{ds} \tag{5.8}$$

$$\ddot{\mathbf{p}}^d \;=\; \dot{s}^2\frac{d^2\mathbf{p}^d}{ds^2} + \ddot{s}\frac{d\mathbf{p}^d}{ds}. \tag{5.9}$$

It is important to note that in these equations the derivatives of the path-parameterizing variable are those related to the desired path. However, if the aircraft is off the path, we must calculate the true value of the path variable. Using the subscript t to denote true, the algorithm then recomputes Equations (5.8) and (5.9) with s_t and $\dot{s}_t$. The values of $\dot{\mathbf{p}}^d$ and $\ddot{\mathbf{p}}^d$ thus calculated provide compensation for the aircraft being away from the path. The true path parameters can be found using the state-space formulation

$$\begin{bmatrix} \ddot{s}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} -K_{s_t} & -K_{s_t} \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{s}_t \\ s_t \end{bmatrix} + \begin{bmatrix} K_{s_t} & K_{s_t} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{s} \\ s \end{bmatrix}, \tag{5.10}$$

where K_{s_t} is selected to ensure fast convergence of the s_t parameter. Given the values of the path position and its derivatives, we can now formulate a proportional-derivative (PD) control law as a unit-norm quaternion to create body-frame control commands given position error in the inertial reference frame.

5.3 Path Following Maneuver Control Law

As mentioned, the objective of the path following control law is to drive the error in the inertial-frame to zero. Based on the method used for determining position relative to the path, the aircraft should lie on a point orthogonal to the path, meaning the path reference frame $\mathbf{y}-\mathbf{z}$ plane. The path reference frame is the orthogonal coordinate frame defined by $\mathbf{c}_1$, $\mathbf{c}_2$, and $\mathbf{c}_3$ from Section 4.2.2, where the vectors $\mathbf{c}_2$ and $\mathbf{c}_3$ lie in the $\mathbf{y}-\mathbf{z}$ plane. Thus, three-dimensional path following is an instantaneous, two-dimensional problem in the path reference frame. Knoebel [12] presents a quaternion control law for regulating an autonomous hovering vehicle to a given inertial north-east position. By transforming the error vectors from the inertial reference frame to the path reference frame, the quaternion control approach can be generalized for three-dimensional paths. There are two components that comprise the path following control law: 1) building the quaternion PD control law and 2) transforming and scaling these controller outputs into changes in the throttle setting and control surface deflections.

5.3.1 Quaternion Control Law

To construct the quaternion PID control law, the error in position and the first derivative of position need to be calculated. From the path function presented in Section 5.2.2, we have the desired position and first derivative, $\mathbf{p}_i^d$ and $\dot{\mathbf{p}}_i^d$, in the inertial reference frame. From the aircraft position estimation, we have values for the vector of aircraft position in inertial, world coordinates, $\mathbf{p}_i = \begin{bmatrix} p_n, & p_e, & p_d \end{bmatrix}^T$. To obtain $\dot{\mathbf{p}}_i$, we again impose the assumption that the aircraft velocity vector is aligned with the body-frame $\mathbf{x}$-axis.[1] This can be written as $\dot{\mathbf{p}}_b = \begin{bmatrix} V & 0 & 0 \end{bmatrix}^T$, where V is the airspeed of the aircraft. Using the aircraft attitude quaternion, this vector can be transformed into the inertial reference frame using the quaternion algebraic vector transformation, which can be written as

$$\dot{\mathbf{p}}_i = \mathbf{e} \otimes \dot{\mathbf{p}}_b \otimes \mathbf{e}^*, \tag{5.11}$$

where $\mathbf{e}$ and $\mathbf{e}^*$ are the attitude quaternion and inverse described in Chapter 3. Given these four vectors, the error and derivative error vectors can be computed as,

$$\tilde{\mathbf{p}}_i = \mathbf{p}_i - \mathbf{p}_i^d \tag{5.12}$$

$$\dot{\tilde{\mathbf{p}}}_i = \dot{\mathbf{p}}_i - \dot{\mathbf{p}}_i^d. \tag{5.13}$$

For path following, we measure the position error in the inertial reference frame, and the controller commands are effected in the body-frame of the aircraft. Therefore, the next step in the control law is to transform the two error vectors into the path reference frame and then into the body-frame. However, the two rotations can be effected as a single rotation of the error vectors directly from the inertial-frame to the body-frame, as was discussed in Section 3.2.2. This is accomplished by applying the aircraft attitude quaternion for both error vectors in the following form:

$$\tilde{\mathbf{p}}_b = \mathbf{e}^* \otimes \tilde{\mathbf{p}}_i \otimes \mathbf{e} \tag{5.14}$$

$$\dot{\tilde{\mathbf{p}}}_b = \mathbf{e}^* \otimes \dot{\tilde{\mathbf{p}}}_i \otimes \mathbf{e}. \tag{5.15}$$

The build up of the control quaternion follows this concept; the path has an orientation at each point in time that can be expressed as a quaternion, $\mathbf{e}_p$. At

[1] The velocity vector alignment assumption is valid as long as the aircraft side-slip and angle of attack is small (less than 5 degrees).

the same instant in time, there exists an unit-norm correction quaternion, $\mathbf{e}_c$, that reflects the difference between the path and the aircraft attitude quaternion, $\mathbf{e}_b$; this relationship can be written as $\mathbf{e}_p = \mathbf{e}_c \otimes \mathbf{e}_b$. The quaternion path following control law computes the correction quaternion that drives the body to the path orientation at the instantaneous location.

In general, the typical PD control law is the summation of proportional and derivative errors, each scaled by a separate gain. For the quaternion control law, one constructs a unit-norm quaternion for each PD element and then composes the three together according to the formula,

$$\mathbf{e}_c = \mathbf{e}_{c(P)} \otimes \mathbf{e}_{c(D)}. \tag{5.16}$$

The unit-norm quaternion for each PID element is constructed using the Euler formulation,

$$\mathbf{e} \equiv \begin{bmatrix} \cos\left(\frac{\Theta}{2}\right) \\ E_x \sin\left(\frac{\Theta}{2}\right) \\ E_y \sin\left(\frac{\Theta}{2}\right) \\ E_z \sin\left(\frac{\Theta}{2}\right) \end{bmatrix}, \tag{5.17}$$

where the control law is defined by calculating a rotation angle, Θ, and the components of a unit-length rotation vector, E_x, E_y, and E_z. For the proportional element of the path following control law, Θ is computed from the scaled magnitude of the body-frame position error as

$$\Theta_{(P)} = K_{(P)}\sqrt{\tilde{\mathbf{p}}_{yp}^2 + \tilde{\mathbf{p}}_{zp}^2}. \tag{5.18}$$

The unit-length vector, $\mathbf{E}$, for the proportional element of the control law is made up of the components of the body-frame position error divided by the magnitude of the error, which yields

$$\mathbf{E}_{(P)} = \begin{bmatrix} \frac{-\tilde{\mathbf{p}}_{yb}}{\sqrt{\tilde{\mathbf{p}}_{yp}^2+\tilde{\mathbf{p}}_{zp}^2}} & \frac{-\tilde{\mathbf{p}}_{zb}}{\sqrt{\tilde{\mathbf{p}}_{yp}^2+\tilde{\mathbf{p}}_{zp}^2}} & 0 \end{bmatrix}^T, \tag{5.19}$$

where each component of the $\mathbf{E}_{(P)}$ vector corresponds to a body-frame rotation. Error in the body-frame $\mathbf{y}$-axis produces rotation about the body-frame $\mathbf{x}$-axis (roll angle) and error in the body-frame $\mathbf{z}$-axis produces rotation about the body-frame $\mathbf{y}$-axis (pitch). The typical MAGICC lab MAVs do not have a rudder, meaning that rolling about the body-frame $\mathbf{x}$-axis is used for turning rather than using skid-to-turn, which causes the last component of $\mathbf{E}_{(P)}$ to equal zero. The rotation angle and vector for the proportional error can be mapped into a unit-norm correction quaternion to yield

$$\mathbf{e}_{c(P)} = \begin{bmatrix} \cos\left(\frac{\Theta_{(P)}}{2}\right) \\ E_{x(P)}\sin\left(\frac{\Theta_{(P)}}{2}\right) \\ E_{y(P)}\sin\left(\frac{\Theta_{(P)}}{2}\right) \\ E_{z(P)}\sin\left(\frac{\Theta_{(P)}}{2}\right) \end{bmatrix} \tag{5.20}$$

$$= \begin{bmatrix} e_{c0(P)} \\ e_{cx(P)} \\ e_{cy(P)} \\ e_{cz(P)} \end{bmatrix}. \tag{5.21}$$

In the proportional element of the correction quaternion, the $e_{cx(P)}$ term is the normalized error transformed into the body-frame and scaled by the sine of half the rotation angle. This component corresponds to a body-frame $\mathbf{x}$-axis rotation, which is produced by aileron deflections. Thus, $e_{cx(P)}$ will be part of the command to the ailerons and in a similar manner, $e_{cy(P)}$ will be part of the command to the elevator once the entire correction quaternion is composed.

The derivative element of the quaternion PID control law is built in a manner similar to that used for the proportional element. The rotation angle for the derivative element of the control law is computed as

$$\Theta_{(D)} = K_{(D)}\sqrt{\dot{\tilde{\mathbf{p}}}_{yp}^2 + \dot{\tilde{\mathbf{p}}}_{zp}^2}, \tag{5.22}$$

and the unit-norm rotation axis can be found from the expression

$$\mathbf{E}_{(D)} = \begin{bmatrix} \frac{-\dot{\tilde{\mathbf{p}}}_{yb}}{\sqrt{\dot{\tilde{\mathbf{p}}}_{yp}^2+\dot{\tilde{\mathbf{p}}}_{zp}^2}} & \frac{-\dot{\tilde{\mathbf{p}}}_{zb}}{\sqrt{\dot{\tilde{\mathbf{p}}}_{yp}^2+\dot{\tilde{\mathbf{p}}}_{zp}^2}} & 0 \end{bmatrix}^T, \tag{5.23}$$

with the derivative element correction quaternion constructed as in Equation (5.20). As with the vector components of the proportional element of the correction quaternion, the unit-norm quaternion built from the derivative error has vector components that relate directly to the commands for aileron and elevator.

It is important to remark that given this formulation, the control effort diminishes if the magnitude of the rotation angles (Θ) for any of the PID elements exceeds π and the control effort reverses sign if the magnitude exceeds 2π. Additionally, if the aircraft is too far away from the maneuver path, it would be natural to abandon the maneuver. Therefore, the gains for each element were constructed as

$$K_{(P)} = K_{d(P)}\frac{\pi}{\tilde{\mathbf{p}}\text{max}} \tag{5.24}$$

$$K_{(D)} = K_{d(D)}\frac{\pi}{\dot{\tilde{\mathbf{p}}}_{\max}}, \tag{5.25}$$

where the value of each design gain, $K_{d(P)}$ and $K_{d(D)}$, must be on the range $0 \rightarrow 1$, and the values of $\tilde{\mathbf{p}}\text{max}$ and $\dot{\tilde{\mathbf{p}}}_{\max}$ are selected according the flight envelope of the aircraft. For example, $\tilde{\mathbf{p}}\text{max}$ was selected to be twice the maneuver radius and $\dot{\tilde{\mathbf{p}}}_{\max}$ was selected as the nominal maximum airspeed of the MAVs. Once the correction quaternion for the individual elements–proportional, and derivative–are constructed, they are composed together according to Equation (5.16).

5.3.2 Lyapunov Stability of the Quaternion Control Law

We define the subscript p to denote to the path reference frame, subscript b to denote to the body-frame, and subscript c to denote to the rotation that corrects the error between the two. We will make three simplifying assumptions for this proof: 1) the desired path is straight, level, and heading north $\left(\mathbf{e}_p = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}\right]\right)$; 2) the desired body reference frame orientation is equal to the actual aircraft body-frame $\left(\mathbf{e}_b = \mathbf{e}_b^d\right)$; and 3) the control law will be comprised of only the proportional control quaternion $\left(\mathbf{e}_c = \mathbf{e}_{c(P)}\right)$. We define the correction quaternion as the rotation between the path-frame and the desired body-frame, which can be written as

$$\mathbf{e}_b = \mathbf{e}_c \otimes \mathbf{e}_p.$$

For the path error dynamics, we define the position error as

$$\tilde{\mathbf{p}} = \mathbf{p}_p - \mathbf{p}_b,$$

and the derivative of the position error can be written as

$$\dot{\tilde{\mathbf{p}}} = \dot{\mathbf{p}}_p - \dot{\mathbf{p}}_b.$$

Applying the first assumption results in $\dot{\mathbf{p}}_p = \left[\begin{array}{ccc} \dot{p}_{px} & 0 & 0 \end{array}\right]$. The expression for $\dot{\mathbf{p}}_b$ can be written as the aircraft velocity vector rotated into the inertial reference frame, yielding

$$\begin{aligned} \dot{\mathbf{p}}_b &= \mathbf{R}_{i\leftarrow b} \left[\begin{array}{c} V_a \\ 0 \\ 0 \end{array}\right] \\ &= \left[\begin{array}{c} e_{b0}^2 + e_{bx}^2 - e_{by}^2 - e_{bz}^2 \\ 2\left(e_{bx}e_{by} + e_{bz}e_{b0}\right) \\ 2\left(e_{bx}e_{bz} - e_{by}e_{b0}\right) \end{array}\right]. \end{aligned}$$

Having defined these two expressions, the derivative of the position error can be written as

$$\dot{\tilde{\mathbf{p}}} = \begin{bmatrix} \dot{p}_{px} - V_a\left(e_{b0}^2 + e_{bx}^2 - e_{by}^2 - e_{bz}^2\right) \\ -2V_a\left(e_{bx}e_{by} + e_{bz}e_{b0}\right) \\ -2V_a\left(e_{bx}e_{bz} - e_{by}e_{b0}\right) \end{bmatrix}.$$

The Lyapunov candidate function can be written as

$$V = \frac{1}{2}\tilde{\mathbf{p}}^T\tilde{\mathbf{p}},$$

whose derivative can be written as

$$\begin{aligned} \dot{V} &= \tilde{\mathbf{p}}^T\dot{\tilde{\mathbf{p}}} \\ &= \tilde{\mathbf{p}}^T \begin{bmatrix} \dot{p}_{px} - V_a\left(e_{b0}^2 + e_{bx}^2 - e_{by}^2 - e_{bz}^2\right) \\ -2V_a\left(e_{bx}e_{by} + e_{bz}e_{b0}\right) \\ -2V_a\left(e_{bx}e_{bz} - e_{by}e_{b0}\right) \end{bmatrix}. \end{aligned}$$

Given the correction quaternion of

$$\mathbf{e}_c = \begin{bmatrix} \cos\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) \\ \frac{-\tilde{p}_{yp}}{\|\tilde{\mathbf{p}}_p\|\tilde{p}_{zp}}\sin\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) \\ \frac{-\tilde{p}_{zp}}{\|\tilde{\mathbf{p}}_p\|}\sin\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) \\ 0 \end{bmatrix} \tag{5.26}$$

and applying assumptions 1) and 2) allows us to compute the second and third terms of $\dot{\tilde{\mathbf{p}}}$ as

$$\begin{aligned} \dot{\tilde{p}}_2 &= -2V_a\frac{\tilde{p}_{yp}}{\|\tilde{\mathbf{p}}_p\|}\sin^2\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) \\ \dot{\tilde{p}}_3 &= -2V_a\frac{\tilde{p}_{zp}}{\|\tilde{\mathbf{p}}_p\|}\sin\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right)\cos\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right), \end{aligned}$$

which can be multiplied by $\tilde{\mathbf{p}}^T$ to yield

$$\dot{V} = -2V_a \frac{\tilde{p}_{yp}^2}{\|\tilde{\mathbf{p}}_p\|} \sin^2\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) - 2V_a \frac{\tilde{p}_{zp}^2}{\|\tilde{\mathbf{p}}_p\|} \sin\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right) \cos\left(\frac{K_{(P)}\|\tilde{\mathbf{p}}_p\|}{2}\right).$$

For the quaternion control law, the sine and cosine function are both restricted to quadrant one, implying that both are guaranteed positive, and thus the expression for $\dot{V}$ is negative definite and shows that $\tilde{\mathbf{p}} \to 0$.

We can validate the second assumption by treating the desired quaternion as an input to an inner loop controller. If we define an error quaternion as

$$\mathbf{e}_b = \mathbf{e}_e \otimes \mathbf{e}_b^d,$$

then to show that $\mathbf{e}_b \to \mathbf{e}_b^d$ we need $\mathbf{e}_e \to \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$. If we assume that the ailerons and elevators can have a first-order response to the control input $\left(\mathbf{e}_e = \exp^{-Kt}\right)$, then the error dynamics can be written as $\dot{\mathbf{e}}_e = -K\mathbf{e}_e$. We define the Lyapunov function for the inner loop control law as

$$V = \frac{1}{2}\mathbf{e}_e^T \mathbf{e}_e,$$

where the derivative is

$$\begin{aligned} \dot{V} &= \mathbf{e}_e^T \dot{\mathbf{e}}_e \\ &= -K\mathbf{e}_e^T \mathbf{e}_e \end{aligned}$$

which is negative definite, and thus we assert that $\mathbf{e}_b \to \mathbf{e}_b^d$.

5.3.3 MAV Control Actuation

The typical BYU MAGICC lab MAV has only two control surfaces, called elevons, which combine the functions of ailerons and elevators. Deflecting the elevons in opposite directions creates a torque about the body-frame **x**-axis while deflecting the elevons in the same direction produces torque about the body-frame **z**-axis. The elegance of the control commands generated by the correction quaternion is that they are decomposed into the individual axes. Thus, the command for the aileron deflection is not a function of roll angle but lateral error and the elevator command is not a function of pitch angle but vertical error. Finally, the throttle command is decoupled from the path following control law; the sole function of the throttle is to prevent aircraft stall by maintaining the desired airspeed.

Thus, the position error that has been defined in the path reference frame must be transformed into the body-frame. These error terms are used to convert the control law commands into actual control surface deflections by merely scaling the correction quaternion relevant to that particular body-frame axis. This scaling is a function of the control authority of the separate actuators, which is determined by the size and placement of the control surfaces and the time constant of the servos and in the case of the throttle, the speed control system. Thus, for the commands to the three typical MAV actuators, we can write

$$\delta_{th} = K_{th}\left(V^c - V\right) \tag{5.27}$$

$$\delta_{ail} = K_{ail}\left(e_{cx}\right) \tag{5.28}$$

$$\delta_{el} = K_{el}\left(e_{cy}\right), \tag{5.29}$$

where e_{cx} and e_{cy} are the components of the complete correction quaternion related to rotations about the body-frame **x**- and **y**-axes, respectively. The three gains (K_{th}, K_{ail}, and K_{el}) are each tuned separately for the available thrust and control authority of the specific aircraft.

5.4 Results

The path following maneuver control law was refined and demonstrated using a sequential process of increasing fidelity. The first step used pure flight simulation in Aviones, which was followed by hardware-in-the-loop simulations, and finally flight tests.

5.4.1 Aviones Simulation Results

The quaternion PID control law was used to control the MAV through several maneuvers that could be employed for avoiding obstacles. As with the attitude tracking controller, the control law was verified in the Matlab Simulink tool and the Aviones flight simulation software.

For the simulated path following maneuvers, the initial flight velocity was north at 13 meters per second, at a starting altitude of 100 meters, and the nominal maneuver radius was 4 meters. As noted in Section 4.2, many different maneuvers can be defined and flown, however, we have limited our testing to three maneuvers, specifically the loop, the Immelmann and the Close-Q. Although loops are not a useful maneuver for avoiding obstacles, they can be employed as a simple means for testing the control law.

Figure 5.2 shows the position and orientation of the MAV, represented as the blue line and bi-colored triangles, and the desired maneuver path, the red line. As can be seen in the plot, the quaternion path following control law was able to regulate the aircraft position along the desired path throughout the maneuver. The tracking of the loop was made more difficult due to the changing airspeed; as airspeed decreased, the turning radius causing the aircraft to more rapidly correct the path error.

Figure 5.3 shows the position error for the MAV flying a simulated loop maneuver. As can be seen in the plot, the control law was able to maintain the aircraft to within 4 meters of the desired position throughout the maneuver.

The path following Immelmann maneuver is accomplished by first executing half of the path following loop maneuver and then half of the trajectory tracking aileron roll maneuver. The simulated MAV follows the half-loop portion of the path quite well. However, the half-roll portion of the Immelmann maneuver is commanded as an attitude trajectory (see Section 4.2), not a path following maneuver; this results in the MAV having to recover altitude and heading after the aileron roll portion of the maneuver, as can be seen in Figure 5.4. The desired path for the Immelmann maneuver shows a discontinuity at the end of the maneuver where the desired path

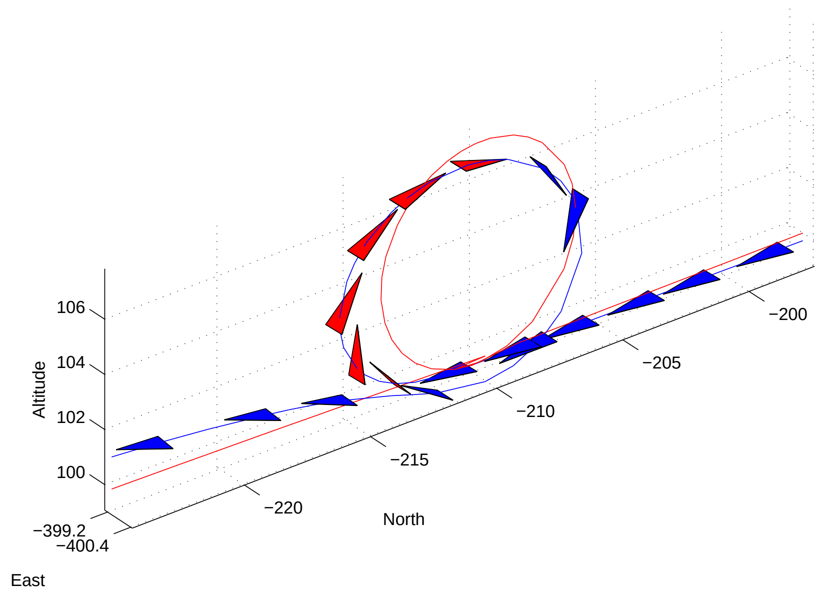


Figure 5.2: Path Following Loop Simulation. The bi-colored triangles provide an indication of aircraft attitude throughout a path following, 4-meter radius, loop maneuver.

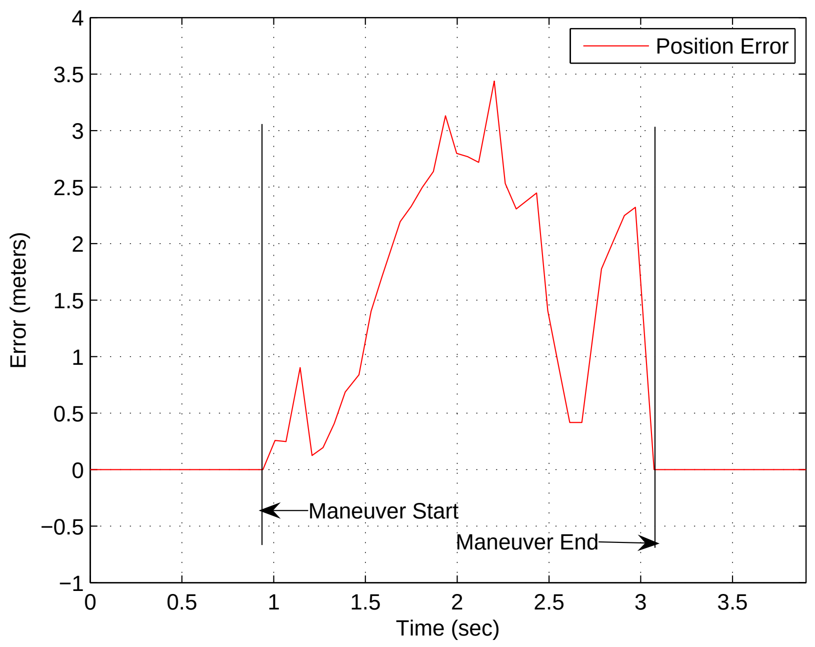


Figure 5.3: Loop Maneuver Position Error. The position error for the loop maneuver using the quaternion path following control law.

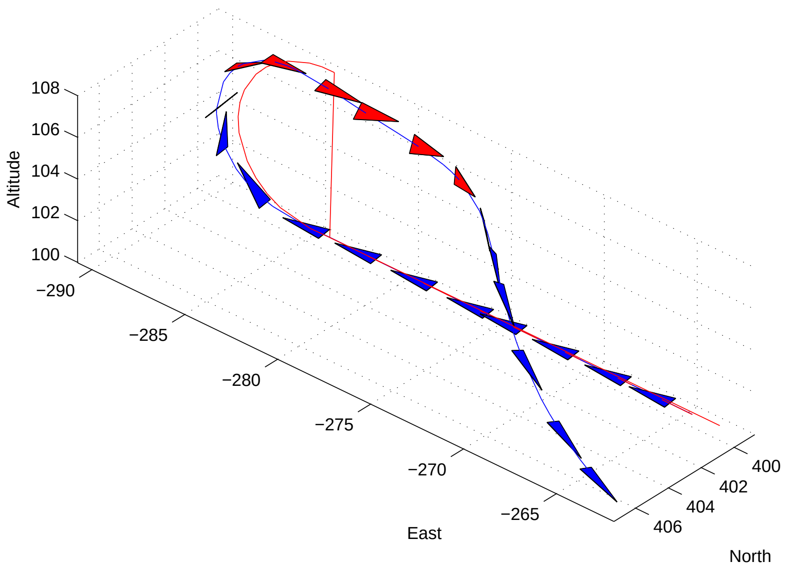


Figure 5.4: Path Following Immelmann Simulation. The bi-colored triangles provide an indication of aircraft attitude throughout a path following Immelmann maneuver. The aircraft recovers to the pre-maneuver waypoint for altitude and heading.

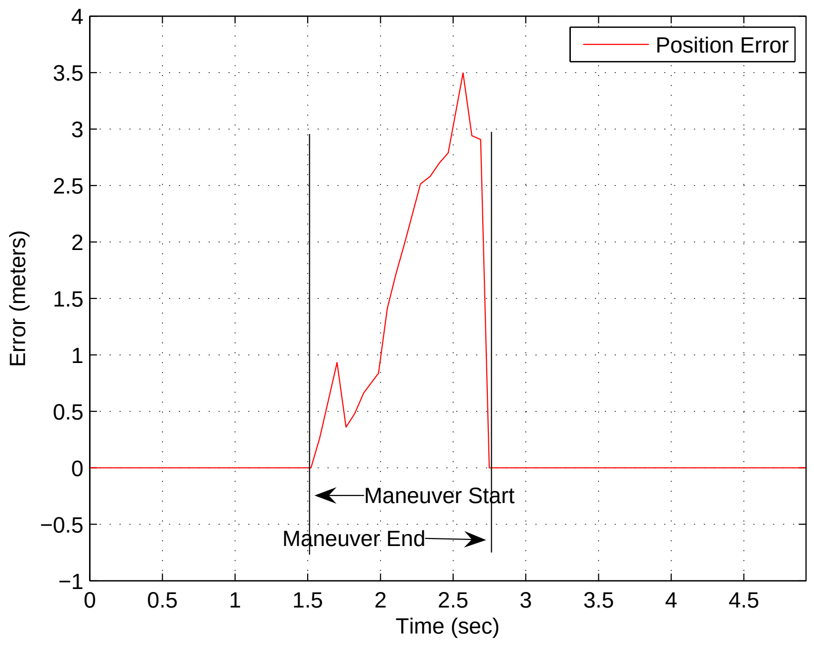


Figure 5.5: Immelmann Maneuver Position Error. The position error during the path following portion of the Immelmann maneuver is plotted as a function of time.

transitions from the half-loop portion of the maneuver back to the previously tracked waypoint direction and altitude.

The performance of the path following control law during the half-loop portion of the Immelmann maneuver is shown in Figure 5.5. The deviations from the subsequent desired heading and altitude seen in Figure 5.4 is due to the controller switching from path following control to attitude trajectory tracking during the aileron roll portion of the Immelmann maneuver.

One of the strengths of the quaternion PD path following control law compared to the trajectory tracking controller was in flying the three-dimensional Close-Q maneuver. The balancing of the elevator and the aileron actuation was much more straight forward. The elevator control gains that would allow the aircraft to fly the loop and Immelmann maneuver were found first and then the aileron gain was tuned upwards until the Close-Q path tracking was acceptable. Figure 5.6 shows a Close-Q maneuver executing a 90-degree right turn by turning upwards to the left. A slight overshoot in altitude is evident as the aircraft recovers after the maneuver and the control mode switches from quaternion PD path following to the standard waypoint navigation and altitude hold.

The aircraft deflection angles for the ailerons and elevator during the Close-Q maneuver are presented in Figure 5.7. The proper balancing of the aileron and elevator gains produces a smooth, well-controlled maneuver. Note that the throttle is plotted as a percent of maximum and saturates through the maneuver as the throttle works to maintain desired airspeed. Also, the elevator deflection saturates temporarily as the control law works bring the aircraft onto the path.

One measure of the performance of the quaternion control law is the distance from the path as a function of time. The simulation results shown in Figure 5.8 show that the controller maintains the position of the aircraft within less than a total of 4 meters of the desired path.

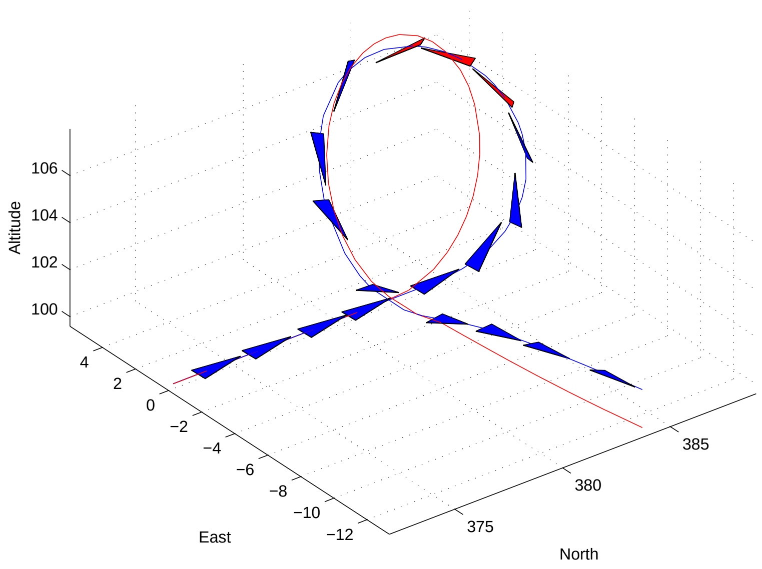


Figure 5.6: Path Following Close-Q Simulation. The Close-Q maneuver is plotted against the desired path. The maneuver was commanded in conjunction with the right angle turn from a northerly heading to an easterly heading. The bi-colored triangles on the blue line indicate the aircraft orientation and path and the red line is the desired path.

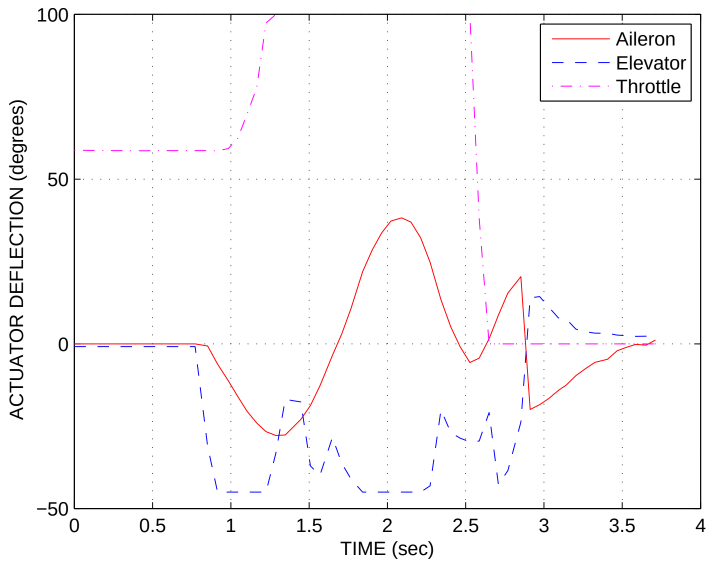


Figure 5.7: Close-Q Actuator Deflections. The throttle setting and control surface deflections during the simulated Close-Q maneuver are plotted as a function of time. Note that the throttle is plotted as a percentage of maximum throttle and the actuator deflections are measured in degrees.

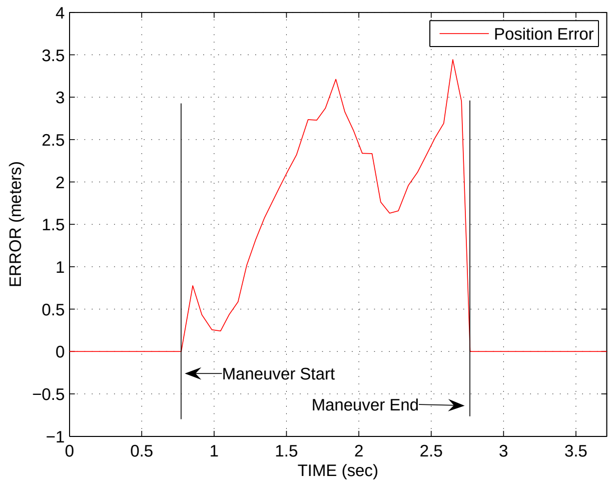


Figure 5.8: Close-Q Position Error. The magnitude of the inertial position error is plotted as a function of time during the Close-Q maneuver.

5.4.2 Hardware-in-the-Loop Simulation Results

As an intermediate step between the Aviones simulations and the actual flight tests, the quaternion PD control law was tested using the aircraft hardware as a simulation environment. These hardware-in-the-loop simulations were achieved by having the autopilot simulate sensor inputs based on evaluating the six-degree-of-freedom aircraft equations of motion. Throughout the hardware-in-the-loop simulations, the MAV autopilot is directly connected to the ground station computer through two cables, one for the Virtual Cockpit interface and the other for receiving output variables from the autopilot. The hardware-in-the-loop simulations demonstrated that the autopilot was calculating and sending the proper responses to the actuators and throttle during the various path following maneuvers.

The system of equations for the motion of the aircraft include three for linear velocities, three for angular velocities, three for the navigation variables (north, east, and altitude) and four equations for the kinematics of the attitude quaternion. Thus, the full equations of motion consist of thirteen nonlinear differential equations. Whereas Aviones uses a fourth-order Runge-Kutta method to numerically propagate the thirteen state variables from equations (2.1 - 2.4), this approach is too computationally costly to implement on the autopilot. However, a simple Newton integration was deemed insufficiently accurate for tracking aerobatic maneuvers. As a compromise between accuracy and computational cost, a first-order Heun's method was implemented, which in generalized form can be written as

$$x_{k+1} = x_k + \frac{1}{2}\left(\dot{x}_k + \dot{x}_{k+1}\right). \tag{5.30}$$

Using the autopilot to propagate the aircraft position and attitude sensor information allowed the various path following maneuvers to be simulated. Throughout the maneuvers, the MAV control surfaces deflected in accordance to the control commands from the quaternion PD control law.

In Figure 5.9, the hardware-in-the-loop simulation was used to fly a loop maneuver with a four meter radius. One benefit of the hardware-in-the-loop simulation was that it allowed control law gains to be tuned without endangering the aircraft. It was found that the gains found using Aviones were too aggressive for the actual aircraft actuators and caused instability in the various maneuvers.

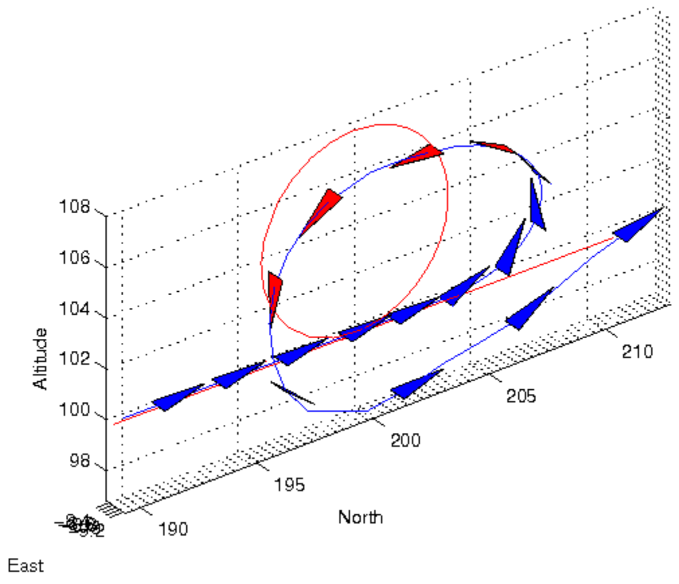


Figure 5.9: Path Following Loop Hardware-in-the-loop Simulation. The bi-colored triangles indicate aircraft attitude throughout a 4-meter radius loop.

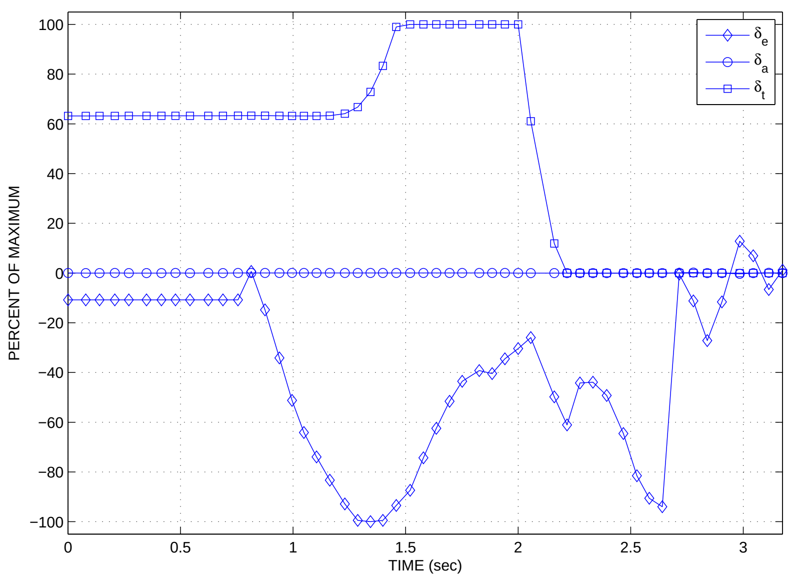


Figure 5.10: Loop Actuator Commands. For the Hardware-in-the-loop simulation of the loop maneuver, the elevator, aileron and throttle deflections are plotted in response to control law commands during a 4-meter radius loop maneuver.

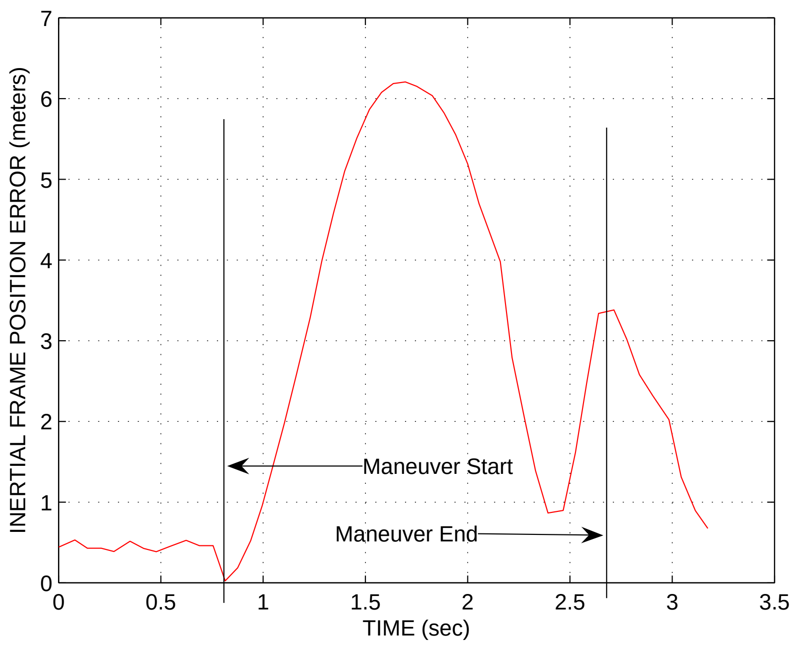


Figure 5.11: Loop Maneuver Position Error. The total position error during a Hardware-in-the-loop simulation of a path-following, 4-meter radius, loop maneuver.

One of the critical pieces of data gathered during the hardware-in-the-loop simulations was the movement of the aircraft control surfaces during the maneuvers. These deflections for the loop maneuver are shown graphically in Figure 5.10. Note that according to convention, positive deflection of the elevator is trailing edge down, which produces a negative pitching moment. Thus, the negative deflection shown in Figure 5.10 is what one should expect for a standard loop maneuver. The throttle response also follows the expected trend; increasing to 100% during the pull-up portion of the loop and then decreasing to zero during the second half of the maneuver as velocity of the aircraft exceeds the fixed preset value. The position error, measured in the inertial reference frame was plotted as a function of time during the loop maneuver. This total position error can be seen in Figure 5.11.

The hardware-in-the-loop simulation was also used to demonstrate the Immelmann and Close-Q maneuvers, as can be seen in Figures 5.12 and 5.13.

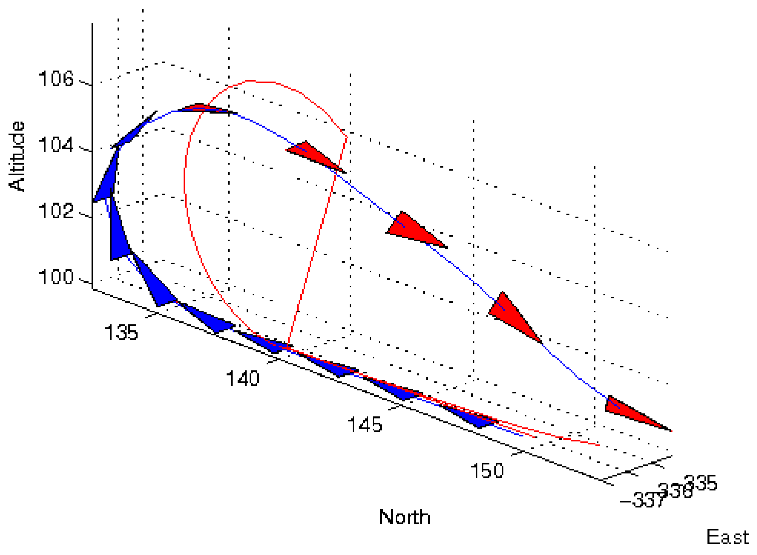


Figure 5.12: Path Following Immelmann Hardware-in-the-loop Simulation. The bi-colored triangles indicate the aircraft attitude throughout a 4-meter radius Immelmann.

The figures for the Immelmann and the Close-Q maneuvers both show the discontinuity as the desired path (smooth red line) switches from the maneuver path

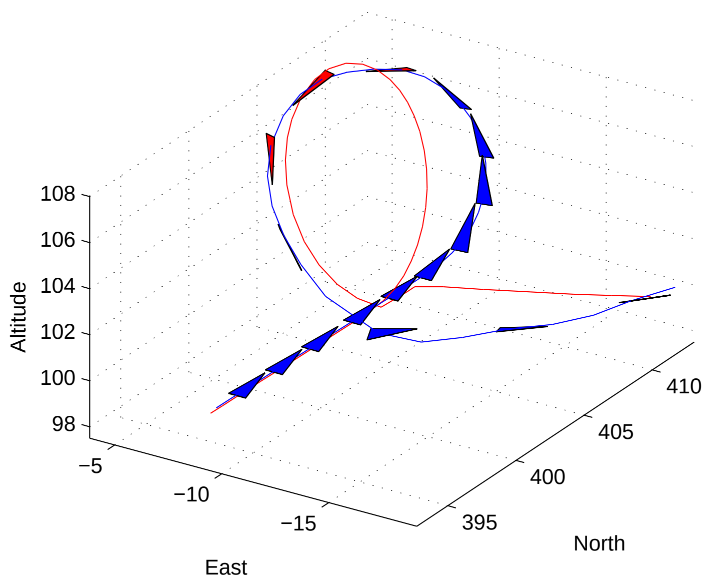


Figure 5.13: Path Following Close-Q Hardware-in-the-loop Simulation. The bi-colored triangles indicate the simulated aircraft attitude throughout a 4-meter radius Close-Q maneuver.

back to the waypoint-following path with its associated altitude. Throughout both of these maneuvers, the flight control surfaces on the MAV were moved about according to the commands from the autopilot.

5.4.3 Flight Test Results

Subsequent to the development and refinement work completed in using Aviones and Hardware-in-the-loop simulations, the maneuver algorithm was flight tested on the MAV. The quaternion path following control law was successful in controlling the MAV through a loop, an Immelmann, and a Close-Q maneuver. Since the loop maneuver is the simplest representative of class of path following maneuvers, only the flight results for the loop will be presented and discussed. The results of the flight testing demonstrate the validity of the quaternion path following approach to controlling a MAV though three-dimensional paths. The results also bring to light some shortcomings that need to be addressed prior to implementation on an operational aircraft.

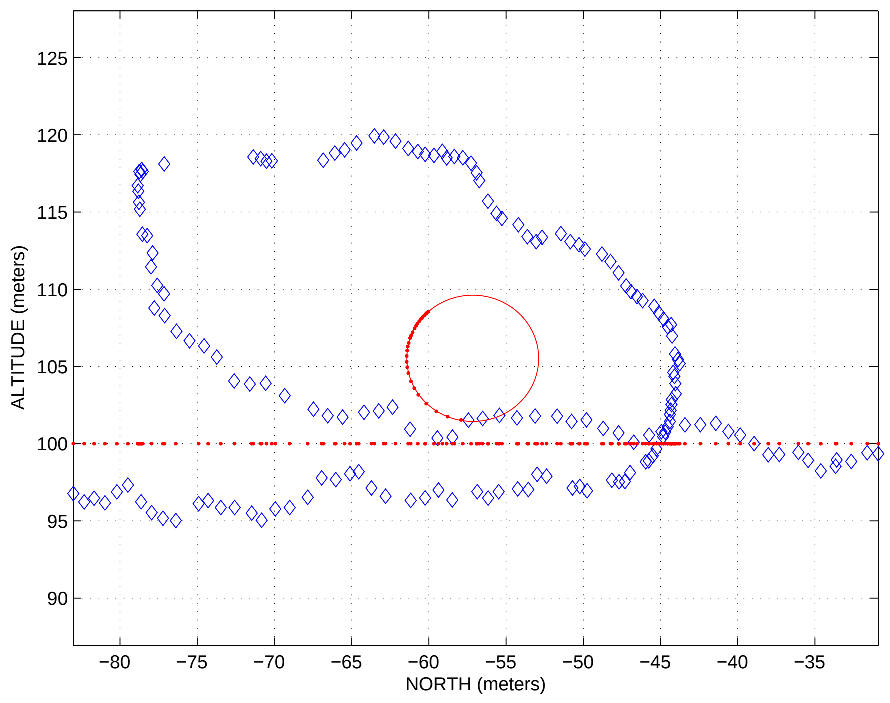


Figure 5.14: Path Following Loop Flight Data. The large, hollow diamonds indicate the aircraft location and the small, solid diamonds track the desired position of the aircraft before, during, and after a 4-meter radius loop maneuver.

Figure 5.14 shows the position and orientation of the MAV as it flew a loop maneuver. The desired path (shown in small, solid diamonds) is at 100 meters prior to the start of the maneuver, but at the instant the maneuver is triggered, the desired position is set to the actual aircraft location to allow for smooth transition to the maneuver. As can be seen in the large, hollow diamonds in Figure 5.14, the quaternion path following control law successfully guided the MAV through a complete loop maneuver. However, the tracking performance could be improved by increasing the gain on position error to a higher value. In addition to the path tracking performance, several key features of Figure 5.14 bear further discussion, namely: 1) the spacing of the data points in the desired maneuver path, 2) the truncation of the desired path, and 3) the uncertainty in the MAV position.

The algorithm we developed for generating the desired path is dependent on the actual location of the aircraft. Thus, the data points for the desired path become closer together as the MAV moved further away from the path without increasing following the perfectly circular path. The key point is that each data point on the desired path reflects the shortest distance from the actual MAV location to the path. Hence, the path generation algorithm is functioning exactly as designed.

Additionally, data points for the loop path do not form a complete circle, stopping at approximately the eleven o'clock position. This is due to the safety feature in the maneuver algorithm, which computes an time limit for each maneuver based on path length and maneuver velocity. If the time limit is exceeded, the maneuver is aborted and the aircraft control reverts back to waypoint following. Because the MAV was not tracking the maneuver very closely, the time limit for the loop was exceeded and the maneuver terminated exactly as designed.

Finally, the high frequency movement in the indicated position of the MAV reflects a small amount of uncertainty in the static pressure sensor and GPS readings which provide the basis for the MAV position estimation algorithms. Certainly one of the dangers in creating a path following maneuver control law is that the position estimation system has inherent uncertainties. For waypoint following flight regimes characterized by steady-level flight, these uncertainties are small enough to be insignificant. However, the errors in position are of greater importance when attempting to track a tightly turning three-dimensional path. The results indicate that the position estimation accuracy is sufficient for maneuvers with a radius on the order of 8 meters as shown here. However, if the aircraft were capable of turns with a radius less than a meter, the position uncertainty would be a limiting factor.

The elevator portion of the actuator deflection graph shown in Figure 5.15 gives evidence to indicate that more control authority is available for executing loop maneuvers. This confirms the assertion that increasing the position error gain would allow for tighter tracking of the desired path and improve maneuver performance. As expected, the commanded aileron deflections during the loop maneuver were practically zero, and the throttle cut back during the second half of the loop as the MAV increased velocity while returning to the waypoint altitude of 100 meters.

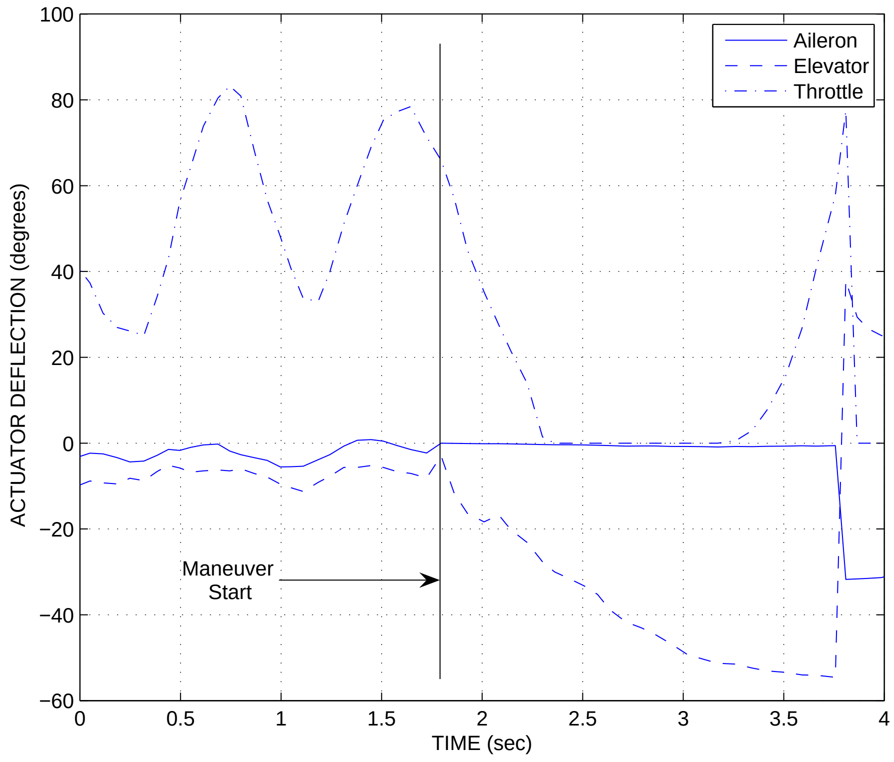


Figure 5.15: Actuator Deflections for the Path Following Loop. The flight data for the aileron and elevator deflections measured in degrees are plotted with the throttle setting measured in percent of maximum are shown for the path following loop maneuver.

5.5 Summary

The quaternion PD control law performs the task of regulating the aircraft to eliminate the error between the aircraft position and the desired path for various maneuvers. The key to success was the innovative approach to determining the location on the path along with the associated path parameter that corresponded to the actual aircraft position. The transformation of the inertial frame errors to the body-frame allowed the actuation to be commanded directly without regard for the aircraft attitude, which eliminated issues related to pitch angle range definition for aerobatic maneuvers. The error of less than four meters in path following performance indicates that this approach is appropriate for applying to the obstacle avoidance maneuvering problem.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

For the estimation of aircraft attitude, we developed a quaternion-based MEKF procedure to fuse data from the avionics suite used by the MAGICC lab MAVs, i.e. low-cost gyroscopes, accelerometers, and GPS. The direct calculation of the attitude error quaternion is what allowed the use of the quaternion attitude representation in the EKF framework. From the simulation, ground, and flight tests, we conclude that the parallel propagation of the attitude quaternion and the attitude error in the MEKF provided an accurate and functional approach to attitude estimation that allows for three-dimensional maneuvering.

Using the quaternion MEKF attitude estimation, we next developed two structured methodologies for specifying maneuvers: fifth-order polynomials in a single angle and angular rate and the trigonometric functions for simultaneous pitch and roll specification as functions of time. The key to both of these approaches was enforcing the maneuver boundary conditions to ensure smooth transition from steady-level flight as part of the maneuver definition. This structured methodology was used to create aileron roll, loop, and Immelmann maneuvers, as well as to define a completely new maneuver, the Close-Q. These maneuvers were flown in simulation and the trajectory tracking aileron roll was demonstrated in flight test by the MAV.

Using the idealized trajectories generated using the trigonometric boundary condition approach, we developed an innovative prediction-correction methodology for determining the path location closest to the current aircraft location at each instant during a maneuver. This knowledge of path location and the path parameter was essential to developing the quaternion PID control law for following three-dimensional paths. Computer simulations and hardware flight tests demonstrated the ability to track the specified paths, validating this approach for the obstacle avoidance maneuvering problem.

6.2 Future Work

Three areas of potential future work include: 1) attitude estimation refinement, 2) altitude hold during the aileron roll maneuver, and 3) obstacle avoidance maneuvering integration into the MAV navigation algorithms. In addition to the control law research, the most tedious hardware problems experienced during flight testing were caused by the lack of robustness in the GPS receiver and antenna combination; effort should be made to develop and implement a more robust solution.

Before replacing the MAGICC lab default attitude estimation scheme (VGO) with the MEKF, it would be valuable to develop a truth metric for the flight test data. Specifically, the attitude angles could possibly be extracted from the rotation matrix built from the knowledge of position and altitude. This would utilize the same alignment of the body-frame **x**-axis and the velocity vector assumption discussed in Section 4.2.2. Additionally, the test fixture used for ground testing the MEKF could be used to test and verify a static gyroscope bias estimation routine to populate the "Reset Gyro" button on the airplane page in Virtual Cockpit. Currently, pressing this button sets the attitude estimate to zero without running any calculations. The other refinement in the MEKF attitude estimation would be to research the normalization of the attitude quaternion subsequent to the time update. It might be possible to convert the second term in Equation 3.27 into a unit-norm quaternion and compose it with the previous quaternion to maintain the unit-norm constraint. Beyond aerobatic maneuvering, the improved attitude estimation accuracy enables better geo-location from the MAV platform and improves altitude hold performance during turns, therefore I would like to demonstrate that the MEKF works for any airframe used by the MAGICC lab.

As was remarked in relationship with the aileron roll maneuver and the second portion of the Immelmann maneuver, these maneuvers do not hold a desired altitude and heading. This is due to the velocity vector alignment assumption and the lack of feedback of desired altitude and heading. Further research time could be spent to develop control laws that would hold altitude and heading during these maneuvers.

Naturally, the next step would be to integrate the attitude estimation and path following maneuver control into a complete obstacle avoidance system. This integration would necessarily include sensors to locate obstacles in the flight path, as well as maneuver selection logic based on the locations of the objects in the immediate vicinity. Additionally, algorithms for path planning after avoiding the obstacle need to be developed so as to ensure mission completion.

Although it seems whimsical to consider aerobatic maneuvering in the context of public flight demonstrations, the US Air Force and Navy spend millions of dollars each year for public relations, especially air shows. The motivation and rationale behind the Air Force Thunderbirds and Navy Blue Angels is to increase the public awareness and pride in the military aircraft technology. This is an important source of recruits and funding. Therefore, it would be natural to extend my research to develop a methodology for controlling synchronized, cooperative path following maneuvers for MAVs to perform at air shows.

References

[1] D. Shim, H. Chung, H. Kim, and S. Sastry, "Autonomous Exploration in Unknown Urban Environments for Unmanned Aerial Vehicles," *2005 AIAA Guidance, Navigation, and Control Conference and Exhibit*, pp. 1–8, 2005. 1

[2] L. Singh and J. Fuller, "Trajectory Generation for a UAV in Urban Terrain, using Nonlinear MPC," *Proceedings of the American Control Conference*, vol. 3, 2001. 1

[3] S. Griffiths, J. Saunders, A. Curtis, T. McLain, and R. Beard, "Obstacle and Terrain Avoidance for Miniature Aerial Vehicles." 1

[4] S. Park, J. Deyst, and J. How, "Performance and Lyapunov Stability of a Nonlinear Path-Following Guidance Method," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 6, pp. 1718–1728, 2007. 1, 72

[5] J. Anderson, *Introduction to Flight*. McGraw-Hill Science/Engineering/Math, 2004. 2, 71

[6] M. D. Shuster, "A Survey of Attitude Representations," *The Journal for the Astronautical Sciences*, vol. 41, no. 4, pp. 439–517, October - December 1993. 2, 21, 22

[7] R. S. Christiansen, "Design of an Autopilot for Small Unmanned Aerial Vehicles," Master's thesis, Brigham Young University, 2004. 3

[8] D. Kingston, R. Beard, T. McLain, M. Larsen, and W. Ren, "Autonomous Vehicle Technologies for Small Fixed Wing UAVs," *AIAA 2nd Unmanned Unlimited Systems, Technologies, and Operations–Aerospace, Land, and Sea Conference and Workshop & Exhibit*, pp. 2003–6559, 2003. 3

[9] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman Filtering for Spacecraft Attitude Estimation," *20th AIAA Aerospace Sciences Meeting, Orlando, FL*, 1982. 4, 21

[10] J. Hauser and R. Hindman, "Aggressive Flight Maneuvers," *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, 1997. 5, 50

[11] S. R. Griffiths, "Vector Field Approach for Curved Path Following For Miniature Aerial Vehicles," *AIAA Journal of Guidance, Navigation, and Control*, not submitted yet. 5, 72

[12] N. B. Knoebel, "Adaptive Quaternion Control of a Miniature Tailsitter UAV," Master's thesis, Brigham Young University, Provo, UT 84602, July 2007. 5, 72, 76

[13] B. L. Stevens and F. L. Lewis, *Aircraft Control and Simulation.* Hoboken, NJ: John Wiley & Sons, Inc, 2003. 9

[14] J. N. Ostler, "Flight Testing Small Electric Powered Unmanned Aerial Vehicles," Master's thesis, Brigham Young University, March 2006. 9

[15] M. Akella, J. Halbert, and G. Kotamraju, "Rigid Body Attitude Control with Inclinometer and Low-Cost Gyro Measurements," *Systems and Control Letters*, vol. 49, no. 2, pp. 151–159, 2003. 21

[16] T. Hamel and R. Mahony, "Attitude Estimation on SO (3) Based on Direct Inertial Measurements," *IEEE International Conference on Robotics and Automation, Orlando, Florida, May*, pp. 15–19. 21

[17] D. B. Kingston and R. W. Beard, "Real-Time Attitude and Position Estimation for Small UAVs Using Low-Cost Sensors," 2004. 21

[18] A. M. Eldredge, "Attitude Estimation For Miniature Air Vehicles," Master's thesis, Brigham Young University, Provo, UT 84602, July 2006. 21, 22

[19] E. R. Bachmann, "Inertial and Magnetic Tracking of Limb Segment Orientation for Inserting Humans into Synthetic Environments," *Computer science, California, United states navy, Naval postgraduate school, PhD*, 2000. 21

[20] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME-Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960. 21

[21] P. S. Maybeck, *Stochastic Models, Estimation and Control.* New York, San Francisco, London: Academic Press, a Subsidiary of Harcourt Brace Jovanovich. 21

[22] S. Julier and J. Uhlmann, "A New Extension of the Kalman Filter to Nonlinear Systems," *Proc. of the 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls*, 1997. 21

[23] I. Bar-Itzhack and Y. Oshman, "Attitude Determination from Vector Observations: Quaternion Estimation," *IEEE Transactions on Aerospace and Electronic Systems* , pp. 128–136, 1985. 21

[24] I. Y. Bar-itzhack, J. Deutschmann, and F. Markley, "Quaternion Normalization in Additive EKF for Spacecraft Attitude Determination," *Flight Mechanics/Estimation Theory Symposium, 1991 p 403-421(SEE N 92-14070 05-13)*, 1991. 21

[25] D. Choukroun, I. Bar-Itzhack, and Y. Oshman, "Novel Quaternion Kalman Filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 174–190, 2006. 21, 22, 33

[26] W. Phillips, C. Hailey, and G. Gebert, "A Review of Attitude Representations Used for Aircraft Kinematics," *AIAA Journal of Aircraft*, vol. 38, no. 4, pp. 718–737, 2001. 21

[27] M. L. Psiaki, "Attitude-Determination Filtering via Extended Quaternion Estimation," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 2, pp. 206–214, March-April 2000. 21

[28] F. L. Markley, "Attitude Error Representations for Kalman Filtering," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 26, p. 2003, 2003. 21, 22, 29

[29] G. Wahba, "A Least Squares Estimate of Spacecraft Attitude," *SIAM Review*, 1965. 22

[30] G. Creamer, "Spacecraft Attitude Determination Using Gyros and Quaternion Measurements," *Journal of the Astronautical Sciences*, vol. 44, no. 3, pp. 357–371, 1996. 22

[31] D. Gebre-Egziabher, R. Hayward, and J. Powell, "Design of Multi-Sensor Attitude Determination Systems," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 40, no. 2, pp. 627–649, 2004. 22, 33, 36

[32] G. Welch and G. Bishop, "SCAAT: Incremental Tracking with Incomplete Information," 1997. 22, 33

[33] F. L. Lewis, *Optimal Estimation: With an Introduction to Stochastic Control Theory.* John Wiley & sons, 1986. 29

[34] D. Barber, J. Redding, T. McLain, R. Beard, and C. Taylor, "Vision-based Target Geo-location using a Fixed-wing Miniature Air Vehicle," *Journal of Intelligent and Robotic Systems*, vol. 47, no. 4, pp. 361–382, 2006. 33, 36

[35] S. Lindemann and S. LaValle, "Current Issues in Sampling-Based Motion Planning," *Proceedings of the International Symposium of Robotics Research*, 2003. 49

[36] E. Frazzoli, M. Dahleh, and E. Feron, "A Hybrid Control Architecture for Aggressive Maneuvering of Autonomous Helicopters," *Proceedings of the 38th IEEE Conference on Decision and Control*, vol. 3, 1999. 49

[37] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-Time Motion Planning for Agile Autonomous Vehicles," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002. 49

[38] M. Mockli, "Guidance and Control for Aerobatic Maneuvers of an Unmanned Airplane," dissertation, Swiss Federal Institute of Technology, Zurich, ETH Zentrum, ML; CH-8092 Zurich; Switzerland, 2006. 49, 72

[39] J. Craig, *Introduction to Robotics: Mechanics and Control.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989. 49, 50

[40] O. Yakimenko, "Direct Method for Rapid Prototyping of Near-Optimal Aircraft Trajectories," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 865–875, 2000. 49

[41] R. Cunha, C. Silvestre, A. Pascoal, and I. Kaminer, "Path Following Control for an Autonomous Helicopter," *Proceedings of the 10th Mediterranean Conference on Control and Automation*, 2002. 49

[42] S. Waydo and R. Murray, "Vehicle Motion Planning Using Stream Functions," *Proceedings, International Conference on Robotics and Automation*, vol. 2, 2003. 50

[43] M. Fliess, J. Levine, P. Martin, and P. Rouchon, "Flatness and Defect of Nonlinear Systems: Introductory Theory and Examples," CAS Internal Report, Tech. Rep. A-284, January 1994. 50

[44] J. Hauser and R. Hindman, "Maneuver Regulation from Trajectory Tracking: Feedback Linearizable Systems," *Proceedings, IFAC Symposium on Nonlinear Control Systems Design*, pp. 595–600, 1995. 50

[45] M. Van Nieuwstadt and R. Murray, "Real-Time Trajectory Generation for Differentially Flat Systems," *International Journal of Robust and Nonlinear Control*, vol. 8, no. 11, pp. 995–1020, 1998. 50

[46] C. Nielsen and M. Maggiore, "Maneuver Regulation Via Transverse Feedback Linearization: Theory and Examples," *Proceedings of the IFAC Symposium on Nonlinear Control Systems (NOLCOS).* 50

[47] D. Nelson, T. McLain, and R. Beard, "Vector Field Path Following for Small Unmanned Air Vehicles," *IEEE Transactions on Control Systems Technology*, 2005. 72

[48] R. Holt, "Three Enabling Technologies for Vison-Based, Forest-Fire Perimeter Surveillance Using Multiple Unmanned Aerial Systems," Master's thesis, Brigham Young University, 2007. 72

[49] G. Notarstefano and R. Frezza, "Aircraft Maneuver Regulation: a Receding Horizon Backstepping Approach," *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 687–692, 2005. 72