2010-03-12

Aerial Recovery of Micro Air Vehicles: Orbit Estimation and Tracking

Daniel Clarke Carlson
Brigham Young University - Provo

Aerial Recovery of Micro Air Vehicles: Orbit Estimation and Tracking

Daniel C. Carlson

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

Master of Science

Mark B. Colton, Chair
Tim W. McLain
Randal W. Beard

Department of Mechanical Engineering

Brigham Young University

April 2010

ABSTRACT

Aerial Recovery of Micro Air Vehicles: Orbit Estimation and Tracking

Daniel C. Carlson

Department of Mechanical Engineering

Master of Science

Aerial recovery of autonomous micro air vehicles (MAVs) presents many unique challenges due to the difference in size and speed of the recovery vehicle and MAV. This thesis presents algorithms to enable an autonomous MAV to estimate the orbit of a recovery vehicle and track the orbit until the final docking phase.

Methods for estimating ellipses that are rotated out of the $x-y$ plane are developed and demonstrated through simulation. These algorithms are shown to be robust to noise and stable numerically. Parameter update methods that are computationally inexpensive, such as recursive least squares and Kalman filtering, are discussed and simulated. A discussion is given of orbit tracking algorithms for circular orbits, and these methods are expanded to include elliptical orbits. These algorithms enable the MAV to track the recovery vehicle's orbit, based on a vector field approach. The tracking algorithms are divided into lateral and longitudinal controllers that allow for tracking of inclined orbits. Finally, the hardware and software setup for live flight tests is discussed. Flight test results are given that validate the functionality of the orbit estimation and orbit tracking algorithms.

Keywords: aerial recovery, unmanned air vehicles, micro air vehicles, towed cable systems, orbit estimation, orbit tracking

ACKNOWLEDGMENTS

Many people have been extremely influential in assisting me in the completion of my thesis work. Among them are family, friends, professors, and coworkers whose encouragement and assistance have been invaluable during the past two years.

First, I must thank my wife Katey and daughter Kylee who have supported and encouraged me in the pursuit of this degree. Their love and encouragement have helped continually motivate me through this process.

Next, I would like to thank Dr. Colton, who spent countless hours helping me over the course of my research. The time he devoted to helping brainstorm ideas, edit papers, and solidify research objectives made this thesis work possible. I would also like to thank Dr. McLain and Dr. Beard who gave valuable input as members of my graduate committee.

I also must acknowledge my peers and coworkers in the MAGICC lab. Being surrounded by such intelligent and diligent people definitely inspired me to be a better student and researcher. The wealth of knowledge that exists in the lab greatly aided this research, especially their expertise in the hardware aspects of this work. Individually, I must thank Jesse Williams and Scott Condie who built, maintained, and repaired the planes we used for our flight tests. Joe Nichols and Solomon Sun, two other members of the aerial recovery team, were also of great assistance during flight tests, lending an extra hand when needed. Additionally, I would like to thank the haptics researchers who sat in my row, namely Dan Ricks, Adam Brandt, Levi Leishman, and Nicole Giullian who continually listened to my successes, obstacles, and quandaries and voiced their support on practically a daily basis. Jeff Saunders also helped me greatly by writing the Mex Functions that create the interface between Virtual Cockpit and MATLAB. Finally, I must thank Bryce Ready and Jacob Bishop for teaching me all about LaTex and LyX and helping me with all my Subversion questions.

This research was supported by the Air Force Office of Scientific Research under STTR contract FA 9550- 09-C-0102 to Procerus Technologies and Brigham Young University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

1.1 Glossary

- AR - Aerial Recovery
- GPS - Global Positioning System
- IMU - Inertial Measurement Unit
- IR - Infrared
- MAV - Micro Air Vehicle
- PID - Proportional, Integral, Derivative
- PN - Proportional Navigation
- RC - Radio Controlled
- RLS - Recursive Least Squares
- SITL - Software in the Loop
- SVD - Singular Value Decomposition
- UAS - Unmanned Aircraft System
- UAV - Unmanned Air Vehicle
- VC - Virtual Cockpit

1.2 Problem Statement

In recent years unmanned aircraft systems (UAS) have become increasingly important in modern warfare. Large UAS are useful for large scale surveillance, while micro air vehicles (MAVs) can be carried and launched by a common soldier. Retrieving MAVs, however, has proved to be a difficult task in military applications. One possible method of recovery would be for MAVs to land near soldiers who would retrieve them. This is potentially dangerous because the MAV could give away the location of the soldier. Another solution could be for the MAV to fly to a safe location, but MAVs are often deployed deep in enemy territory and lack sufficient fuel to return to base. Another potential solution is for the MAVs to be recovered by a larger aircraft, either manned or unmanned. A major challenge with this solution is the fact that MAVs are often deployed by large, fast moving aircraft; thus, aerial recovery methods similar to mid-air fueling have additional complications due to the difference in size and speed of the vehicles. Additionally, large UAS have a large wash from the propeller and air flowing over the aircraft. This wash can cause major disturbances to a MAV trying to dock with a significantly larger vehicle. Due to these difficulties, MAV recovery has not been implemented in military applications. The objective of this thesis work is to develop innovative methods for aerial recovery that would enable MAVs to be recovered by a large, fast-moving aircraft.

1.3 Concept

The method of aerial recovery that is being developed at BYU is inspired by research done on tow planes dragging a cable with a drogue connected at the end [1–7]. Skop and Choo show that for a sufficiently long cable and with sufficient drag, the radius of the orbit of the drogue will approach zero. In our new method of aerial recovery, the drogue would be used as a docking station for the MAV. This concept can be expanded to control the drogue so that its speed is slightly slower than that of the MAV. In our implementation the radius of the drogue would not approach zero; rather, the drogue would maintain a stable orbit with a fixed radius that would give it a desired airspeed. Figure 1.1 shows the baseline concept.

There are three critical components in this research. First, the dynamics of the mothership-drogue system must be modeled. Specifically, the forward and inverse dynamics of the mothership-

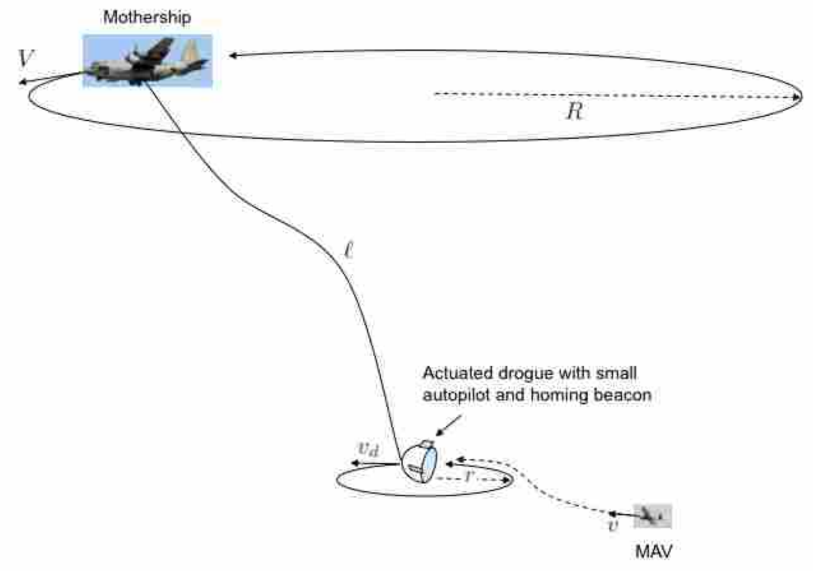


Figure 1.1: Baseline Concept

drogue system are needed for simulation and control, respectively [3–5]. This work was expanded recently to explore the problem of lifting payloads with multiple aircraft by Williams and Ockels [8]. The analysis of the mothership-drogue system has been considered in [6]. Second, estimation and control algorithms must be developed that enable autonomous cooperative docking between the MAV and the drogue. Third, the drogue and towing system must be designed to achieve the desired aerodynamic properties and to enable consistent and robust retrieval of the MAV.

This thesis focuses on the algorithms that enable a MAV to estimate and enter the orbit of the drogue in preparation for final rendezvous and docking. By estimating and tracking the orbit of the drogue, the attitudes and velocities of the vehicles will be similar as the MAV approaches the drogue, thus facilitating docking. For this work, it is assumed that there is communication between the drogue and MAV, allowing the drogue to transmit its position and heading (from an on-board GPS receiver) to the MAV. The drogue orbit is estimated from the transmitted data and a coordinated rendezvous approach is taken, rather than treating the drogue as a target during the initial stages of rendezvous.

During the final stages of rendezvous, when the distance between the MAV and drogue is small compared to the error of the orbit estimate, it is necessary to use some other rendezvous algorithm (e.g., vision-based proportional navigation) [9].

1.4 Contributions

This research has helped further aerial recovery research for MAVs. The primary contributions to the work include the following:

- A new method of estimating an arbitrarily inclined elliptical orbit based on x, y, and z position data.
- A novel tracking algorithm that allows the MAV to track an arbitrarily inclined elliptical drogue orbit.
- Flight-tested software that implements the tracking algorithms.
- A 6-DOF simulation in MATLAB that demonstrates each phase of the rendezvous process.
- Software-in-the-loop tests that demonstrate each algorithm's function and ability to work in real-time.
- Flight test results that demonstrate each algorithm's function and ability to work in real-time.

1.5 Thesis Overview

Chapter 2 discusses basic UAV kinematics and dynamics. In Chapter 3 several methods of orbit estimation are discussed; particularly the new method of estimating ellipses rotated out of the horizontal plane is emphasized. This method makes use of several previously published algorithms used for in-plane ellipse estimation and expands it to the more general case of an ellipse rotated arbitrarily out of the plane. Chapter 4 discusses an algorithm for elliptical orbit tracking. Chapter 5 discusses the results of the flight tests that were performed and demonstrates the functionality of the algorithms discussed in Chapters 3 and 4 in a real-time setting. Finally, Chapter 6 addresses some conclusions and plans for future work in the project.

CHAPTER 2. UAV KINEMATIC AND DYNAMIC MODEL

The following sections will discuss the general dynamic model and some reduced order models that simplify UAV control. The lower order 6-state model discussed in this chapter was used in the preliminary MATLAB simulation development, and the higher order 12-state model was used in the high-fidelity software-in-the-loop tests. The kinematic and dynamic models presented in this chapter are based on the models developed in an unpublished UAV book by Beard and McLain [10] and Johnson's masters thesis [11].

2.1 State Variables

The state variables that can be used to describe the attitude and position of the MAV are the following:

- $p_n =$ inertial position of the UAV in the x direction (north)
- $p_e =$ inertial position of the UAV in the y direction (east)
- $h =$ altitude of the UAV in the $-z$ direction
- $u =$ velocity of the UAV in the body frame x direction
- $v =$ velocity of the UAV in the body frame y direction
- $w =$ velocity of the UAV in the body frame z direction
- $V = \begin{bmatrix} u & v & w \end{bmatrix}^T$
- $V_a =$ airspeed of the UAV
- $\phi =$ roll angle defined with respect to the vehicle-2 frame
- $\theta =$ pitch angle defined with respect to the vehicle-1 frame

- $\psi =$ yaw angle defined with respect to the vehicle frame
- $p =$ roll rate measured about the x-axis in the body frame
- $q =$ pitch rate measured about the y-axis in the body frame
- $r =$ yaw rate measured about the z-axis in the body frame
- $\chi =$ angle between the inertial x direction (north) and the velocity with respect to the ground
- $\gamma =$ angle between the horizontal plane and the velocity with respect to the ground
- $\alpha =$ angle of attack
- $\beta =$ side-slip angle

2.2 Coordinate Frames

We define the inertial coordinate frame to be a fixed frame with the x-axis pointing north and the y-axis pointing east. Through a series of Euler angle transformations we can rotate from this inertial axis to a body-fixed coordinate frame [12]. The vehicle frame has the same orientation as the inertial frame but it is located at the center of the MAV. The translation to get from the inertial frame to the vehicle frame is given by $\begin{bmatrix} p_n & p_e & h \end{bmatrix}^T$. A 3-2-1 Euler angle transformation is then used to get from the vehicle frame to the body-fixed frame. First, a right-handed rotation of ψ about the vehicle z-axis is made to get to the vehicle-1 frame. Then a right-handed rotation is made about the vehicle-1 y-axis by the angle θ to get to the vehicle-2 frame. Finally a right-handed rotation is made about the vehicle-2 x-axis by the angle ϕ to get to the body frame. The body frame is defined with the x-axis pointing out the nose of the aircraft and the y-axis pointing out the right wing. The transformations are given as

$$R_\phi R_\theta R_\psi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

It is also useful to rotate from the body frame to the wind frame. The wind frame is defined as the orientation of the aircraft with respect to the wind vector. In order to generate lift, the aircraft must fly at a positive angle with respect to the wind. Additionally, the aircraft must crab into the wind in order to maintain a desired course in the presence of a crosswind. To perform this transformation, first, a left-handed rotation of α about the body y-axis is made. Then a right-handed rotation is made about the new z-axis by β. This transformation is given as

$$R_\beta R_\alpha = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}. \tag{2.2}$$

2.3 General Kinematic and Dynamic Model

The model for UAV kinematics and dynamics is described by

$$\begin{bmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{h} \end{bmatrix} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ s\theta & -s\phi c\theta & -c\phi c\theta \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}, \tag{2.3}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} rv - qw \\ pw - ru \\ qu - pv \end{bmatrix} + \frac{1}{m} \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix}, \tag{2.4}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{2.5}$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \Gamma_1 pq - \Gamma_2 qr \\ \Gamma_5 pr - \Gamma_6 \left(p^2 - r^2\right) \\ \Gamma_7 pq - \Gamma_1 qr \end{bmatrix} + \begin{bmatrix} \Gamma_3 l + \Gamma_4 n \\ \frac{1}{J_y} m \\ \Gamma_4 l + \Gamma_8 n \end{bmatrix}, \tag{2.6}$$

where $c\theta \overset{\triangle}{=} \cos\theta$ and $s\theta \overset{\triangle}{=} \sin\theta$, m, n, and l represent the angular momentum of the UAV in the body frame about the x, y, and z-axes respectively, f_{x-z} are the sum of all external forces in the x,

y, and z directions respectively, and Γ_{1-8} and J_y are variables that describe the moments of inertia for the UAV, whose definitions can be found in [10, 11] and in Appendix A.

2.4 6-State Model

To simplify the dynamics and control used in simulation, the 12-state model can be reduced to lower-order models by making certain assumptions. First, the velocity of the UAV can be related to the inertial frame by two rotations, heading (ψ) and flight path angle(γ), instead of the three Euler angles used in the 12-state model. Also, assuming the UAV makes coordinated turns allows us to introduce a load factor n defined as

$$n = \frac{\cos\gamma}{\cos\phi}, \tag{2.7}$$

and variable ν defined as

$$\nu = \frac{\cos\phi}{\cos\gamma} n. \tag{2.8}$$

If we also assume the UAV travels at a constant velocity, then the 12-state navigation equations can be reduced to the following 6-state navigation equations:

$$\dot{p_n} = V_a \cos\psi \cos\gamma + w_n \tag{2.9}$$

$$\dot{p_e} = V_a \sin\psi \cos\gamma + w_e \tag{2.10}$$

$$\dot{h} = V_a \sin\gamma + w_h \tag{2.11}$$

$$\dot{\psi} = \frac{g}{V_a} \tan(\phi) \nu^c \tag{2.12}$$

$$\dot{\gamma} = \frac{1}{\tau_\gamma} (\gamma^c - \gamma) \tag{2.13}$$

$$\dot{\phi} = \frac{1}{\tau_\phi} (\phi^c - \phi), \tag{2.14}$$

where g is gravity, τ_V, τ_ϕ, and τ_n are autopilot time constants that are tuned so that the simulation models true aircraft dynamics, and w_n, w_e, and w_h are the components of wind in the north, east and upwards directions respectively. If we further assume that the roll and pitch dynamics are fast relative to the heading and altitude dynamics, then Equations (2.11) and (2.12) become

$$\dot{h} = V_a \sin \gamma^c + w_h \tag{2.15}$$

$$\dot{\psi} = \frac{g}{V_a} \tan \phi^c. \tag{2.16}$$

Now the only inputs to the system are γ^c and ϕ^c. This 6-state model was used in the UAV simulations developed in MATLAB. By making these assumptions, the navigation model still maintains a high level of fidelity, and the control development is simplified.

2.5 Chapter Summary

This chapter defines the state variables used in UAV dynamics and control. Additionally, this chapter discusses the general dynamic UAV model and some reduced-order models that simplify UAV control. The lower-order models were used in the initial simulation development, and the higher-order models were used in the high-fidelity simulations.

CHAPTER 3. ORBIT ESTIMATION

This chapter describes one element of MAV-drogue rendezvous: the mathematical description of the drogue's orbit and the estimation of the orbit parameters from GPS data. For this work, it is assumed that there is communication between the drogue and MAV, allowing the drogue to transmit its position and heading (from an on-board GPS receiver) to the MAV. The drogue orbit is estimated in real-time as the MAV receives the transmitted drogue data, and a coordinated rendezvous approach is taken, rather than treating the drogue as a target during the initial stages of rendezvous. The MAV assumes the same orbit as the drogue, allowing a more natural initial rendezvous approach. During the final stages of rendezvous, when the relative position between the MAV and drogue is small compared to the error of the orbit estimate, it is necessary to use some other rendezvous algorithm (e.g., vision-based proportional navigation). This section focuses on the problem of estimating the drogue orbit from GPS data. Chapter 4 describes a method that enables the MAV to track the drogue orbit.

Ideally, the drogue will travel in a circular orbit in a plane parallel to the ground. In practice, the actual drogue orbit is not circular and in the horizontal plane due to (1) wind and other disturbances, and (2) the numerical approximations inherent in the inverse dynamics to find the required mothership orbit from the desired circular drogue orbit. Consequently, for estimation purposes, the drogue orbit will be treated as an arbitrarily rotated ellipse. Flight tests results shown in Chapter 5 further validate the use of an ellipse in the orbit characterization.

3.1 History of Orbit Tracking

Orbit tracking has been performed for many years, with foundations in the estimation of the orbits of planets from astronomical observations [13]. Newton first proposed a method of circular orbit estimation using three points in the Principia. Shortly thereafter, Halley published results of orbit calculations of heavenly bodies that had been sited using Newton's method. Euler and Lambert furthered the work in later years to expand to ellipses and hyperbolas. Lagrange solidified the work of Euler and Lambert into an elegant mathematical framework. Laplace also developed methods of orbit estimation. Gauss's research is of particular importance, as he developed the method of least squares which play a fundamental roll in the advanced methods that will be discussed later in this paper.

3.2 Circular Orbit Estimation

Initially, the equation of a circle was used instead of an ellipse to represent the drogue orbit. The purpose for this was to determine the feasibility of the algorithms on a simpler model. A circle has only three parameters to estimate, while an arbitrarily rotated ellipse has eight.

3.2.1 Gradient Descent Optimization

It is possible to estimate a circular orbit from GPS data using a gradient descent algorithm described by Moon and Sterling [14]. The three circle parameters to estimate can be assembled into a vector

$$\mathbf{a} = \begin{bmatrix} x_0 \\ y_0 \\ r \end{bmatrix}, \tag{3.1}$$

where x_0 represents the center of the circle in the north direction, y_0 represents the center of the circle in the east direction, and r represents the radius. The error in the circle parameter estimates is defined by the equation for a circle,

$$\text{error} = (x - x_0)^2 + (y - y_0)^2 - r^2, \tag{3.2}$$

where x and y are the most recent data point received. The objective function J is defined as the square of the error,

$$J = \text{error}^2. \tag{3.3}$$

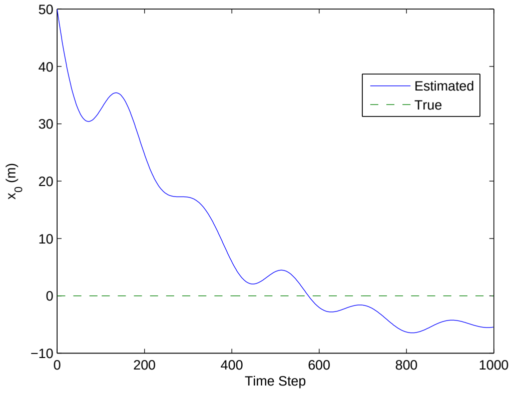


Figure 3.1: Estimate of x_0 using gradient descent optimization.

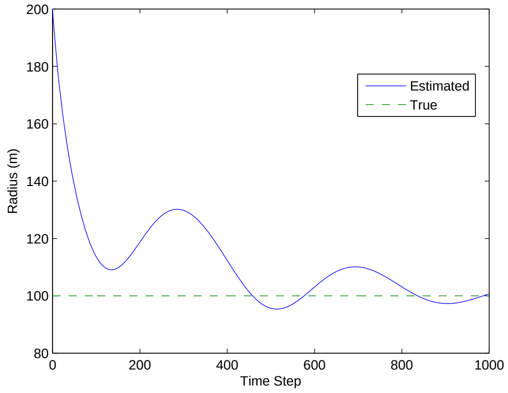


Figure 3.2: Estimate of radius using gradient descent optimization.

The objective of the gradient descent algorithm is to find the values of the circle parameters **a** that minimize J. This is accomplished by updating the estimates of **a** using the gradient of J:

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \mu \frac{\partial J}{\partial \mathbf{a}}, \tag{3.4}$$

where μ is the step size, which can be tuned to optimize the convergence rate.

3.2.2 Simulation Results

This algorithm was tested with noiseless simulated GPS data points. Results are shown in Figures 3.1 and 3.2. In the figures, each time step represents a new data point being sent to the estimator. Figure 3.1 shows the evolution of the estimate of x_0 from the reception of the first data point through the reception of 1000 data points. Figure 3.2 shows the radius estimate for the same data.

The results show that gradient descent is not an efficient way to estimate the parameters describing a circle because of the slow convergence (over 1000 iterations), even with noiseless data.

3.3 Boundary-Based Ellipse Fitting with Least Squares

A traditional method for boundary-based ellipse fitting, as described by Bookstein [15], will now be described. The parametric equations for an elliptical orbit in the horizontal (x-y) plane are given by

$$\begin{array}{l} x(\sigma) - x_0 = a\cos(\sigma)\cos(\psi) - b\sin(\sigma)\sin(\psi) \\ y(\sigma) - y_0 = a\cos(\sigma)\sin(\psi) + b\sin(\sigma)\cos(\psi) \end{array}, \tag{3.5}$$

where σ is the free parameter that can vary from 0 to 2π. As σ varies, the corresponding values of x and y describe the conic. The parameters x_0 and y_0 are the coordinates of the center of the ellipse, a and b are the major and minor axes, respectively, and ψ is the angle of rotation of the ellipse in the horizontal plane, measured from the x-axis. An ellipse may also be represented using the general equation of a conic,

$$F(x,y) = a_1x^2 + a_2xy + a_3y^2 + a_4x + a_5y + a_6 = 0, \tag{3.6}$$

where a_i, $i = 1...6$ are the conic parameters. The objective is to estimate these parameters from N measurements of the drogue's x and y GPS coordinates, which can be accomplished using least squares regression.

To determine the least squares solution, the N orbit data points are assembled in a matrix, and a matrix equation is formed that represents (3.6) for each known point on the ellipse:

$$\begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_Ny_N & y_N^2 & x_N & y_N & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}. \tag{3.7}$$

Singular value decomposition (SVD) can be used to determine the parameters that are optimal in the least squares sense. The resulting parameters may be converted to the parameters that describe an ellipse in parametric form, as defined by (3.5).

This method has the weakness that the solution parameters are not guaranteed to describe an ellipse, but may represent the parameters of another form of conic, such as a parabola or hyperbola. This is possible when relatively few data points (6-10) have been received, but becomes much less probable as more and more data is received, so the optimal solution over time is most likely to result in ellipse parameters. One way to guarantee ellipse parameters is to ignore solutions that give non-elliptical solutions and wait for more data to be received, but in a real-time aerial recovery situation, it may not be ideal to wait. Another solution is to iterate until the solution is forced to become an ellipse as described by Sampson [16], but that solution would be more computationally expensive.

3.3.1 Fitzgibbon's Method

Fitzgibbon proposed a method in which an ellipse-specific constraint is used to guarantee that the solution represents an ellipse [17]. For (3.6) to represent an ellipse, the parameters are constrained by

$$4a_1a_3 - a_2^2 > 0. \tag{3.8}$$

It has been shown that recasting this as an equality constraint, in which (3.8) is still satisfied, results in valid parameter estimates whose values are not dependent on the particular constant value used on the right side of (3.8) [17]. Essentially, since (3.8) can be multiplied by a constant and the result is the same ellipse, equation (3.8) can be set to an arbitrary constant without loss of generality. Thus the constraint can be written as

$$4a_1a_3 - a_2^2 = 1. \tag{3.9}$$

Equation (3.6) can be written in matrix form as

$$F(\mathbf{x}) = \mathbf{x} \cdot \mathbf{a}, \tag{3.10}$$

where

$$\mathbf{a} = \left[\begin{array}{cccccc} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{array} \right]^T \tag{3.11}$$

and

$$\mathbf{x} = \left[\begin{array}{cccccc} x^2 & xy & y^2 & x & y & 1 \end{array} \right]. \tag{3.12}$$

Determining the conic parameters is equivalent to minimizing the distance from all of the points to the conic,

$$\min_{\mathbf{a}} \sum_{i=1}^{N} (\mathbf{x_i} \cdot \mathbf{a})^2 \text{ subject to } 4a_1a_3 - a_2^2 = 1, \tag{3.13}$$

or, in matrix form, as

$$\min_{\mathbf{a}} ||\mathbf{Da}||^2 \text{ subject to } \mathbf{a^T C a} = 1, \tag{3.14}$$

where

$$\mathbf{D} = \begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N^2 & x_Ny_N & y_N^2 & x_N & y_N & 1 \end{bmatrix}, \tag{3.15}$$

and

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{3.16}$$

D therefore contains the N measured drogue locations and **C** describes the ellipse constraint. Equation (3.14) describes a quadratically constrained least squares minimization problem. By applying Lagrange multipliers, this equation can be solved using a method proposed by Gander [18], which recasts (3.14) as

$$\mathbf{Sa} = \lambda \mathbf{Ca} \text{ subject to } \mathbf{a^T C a} = 1, \tag{3.17}$$

where $\mathbf{S} = \mathbf{D^T D}$. Solving (3.17) using Gander's method gives the solution vector **a**.

3.3.2 Numerically Stable Improvement

Fitzgibbon's method has the drawback that the computation of the eigenvalues is sometimes unstable and can yield infinite or complex results. This arises from the fact that $\mathbf{D}^T\mathbf{D}$ is often nearly singular. Halir and Flusser [19] proposed a method for improving the accuracy and speed of the algorithm. This method makes use of the special structure of the matrices to eliminate the singularities. The result is unstable only if all of the points lie on the same line, in which case there is no suitable approximation for an ellipse. In this method $\mathbf{D}$ is restructured as $\mathbf{D} = \begin{pmatrix} \mathbf{D_1} & \mathbf{D_2} \end{pmatrix}$, where

$$\mathbf{D_1} = \begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 \\ \vdots & \vdots & \vdots \\ x_i^2 & x_iy_i & y_i^2 \\ \vdots & \vdots & \vdots \\ x_N^2 & x_Ny_N & y_N^2 \end{bmatrix}, \mathbf{D}_2 = \begin{bmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_i & y_i & 1 \\ \vdots & \vdots & \vdots \\ x_N & y_N & 1 \end{bmatrix}. \qquad (3.18)$$

S, C, and **a** are also restructured:

$$\mathbf{S} = \begin{bmatrix} S_1 & S_2 \\ S_2^T & S_3 \end{bmatrix} \text{ where } \begin{array}{c} \mathbf{S_1} = \mathbf{D_1^T D_1} \\ \mathbf{S_2} = \mathbf{D_1^T D_2} \\ \mathbf{S_3} = \mathbf{D_2^T D_2} \end{array} \qquad (3.19)$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{C_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \text{ where } \mathbf{C}_1 = \begin{bmatrix} 0 & 0 & 2 \\ 0 & -1 & 0 \\ 2 & 0 & 0 \end{bmatrix} \qquad (3.20)$$

$$\mathbf{a} = \begin{bmatrix} \mathbf{a_1} \\ \mathbf{a_2} \end{bmatrix} \text{ where } \mathbf{a_1} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ and } \mathbf{a_2} = \begin{bmatrix} d \\ e \\ f \end{bmatrix}. \qquad (3.21)$$

Equation (3.17) then becomes

$$\begin{bmatrix} \mathbf{S_1} & \mathbf{S_2} \\ \mathbf{S_2^T} & \mathbf{S_3} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_1} \\ \mathbf{a_2} \end{bmatrix} = \lambda \cdot \begin{bmatrix} \mathbf{C_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{a_1} \\ \mathbf{a_2} \end{bmatrix}, \qquad (3.22)$$

which is used to solve for the parameters in vectors $\mathbf{a}_1$ and $\mathbf{a}_2$.

3.4 Parameter Update Methods

The method introduced by Halir and Flusser is used to obtain an estimate of orbit parameters from the first N GPS data points provided by the drogue. As new GPS data become available during the course of the aerial recovery process, it is computationally expensive to continue to use the method just described to update the orbit estimates because the size of $\mathbf{D}$ continues to increase with each new data point. Instead, to continue to refine the estimate, a parameter update method is used.

3.4.1 Recursive Least Squares

Recursive least squares (RLS) [14] can be used to update the estimate of the ellipse parameters for each new GPS data point received from the drogue. The parameters to estimate are again represented by the vector $\mathbf{a}$ defined in (3.11). The parameter estimates are updated using

$$\gamma_{n+1} = \frac{1}{\lambda + \mathbf{x}_{n+1}^T \mathbf{P}_n \mathbf{x}_{n+1}} \tag{3.23}$$

$$\mathbf{a}_{n+1} = \mathbf{a}_n - \gamma_{n+1} \mathbf{P}_n \mathbf{x}_{n+1} \mathbf{x}_{n+1}^T \mathbf{a}_n \tag{3.24}$$

$$\mathbf{P}_{n+1} = \frac{1}{\lambda} \left(\mathbf{P}_n - \gamma_{n+1} \mathbf{P}_n \mathbf{x}_{n+1} \mathbf{x}_{n+1}^T \mathbf{P}_n \right). \tag{3.25}$$

The matrix $\mathbf{P}$ is proportional to the covariance of the parameter estimates [20] and is initialized to the identity matrix. The vector $\mathbf{x}$ represents a vector of the most recent GPS data received from the drogue, and λ is the forgetting factor, which is a tunable parameter that controls the responsiveness of the estimates and the level of filtering, with higher λ values resulting in smoother but less responsive parameter estimates.

3.4.2 Kalman Filtering

The RLS algorithm has the same weakness as the Bookstein method [15]: the solution is not guaranteed to be an ellipse. The initial estimate is constrained to be an ellipse using Halir's method, but once the RLS algorithm takes over, the solution could theoretically diverge and become some other conic such as a hyperbola. By using a Kalman filter with nonlinear state constraints, the

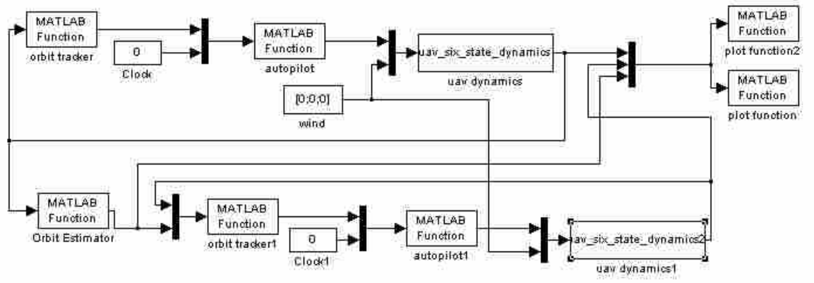


Figure 3.3: Simulink block diagram used to simulated UAV dynamics, orbit estimation and orbit tracking.

solution can be projected onto the space where the ellipse-specific constraint is satisfied [21]. This algorithm was not used in simulation, but its details can be found in Appendix B.

3.4.3 Simulation Results

Figure 3.3 shows the block diagram for the MATLAB simulation where the algorithms were tested.

The Halir-Flusser method followed by RLS was applied to simulated GPS data from the drogue, with additive Gaussian noise with a standard deviation of 5 m and no bias to simulate uncertainty in the GPS measurement. The RLS algorithm was used here instead of the Kalman filtering algorithm with nonlinear state constraints because the constraint matrix M cannot be decomposed as required in Equation B.15, since the matrix is not invertible. Additionally, the RLS algorithm is simpler computationally, and is unlikely to diverge, since the GPS data points do describe an ellipse. This algorithm was also tested on actual drogue GPS data as shown in Chapter 5, and in no case did the algorithm diverge, further validating the use of the RLS algorithm. Figure 3.4 shows the resulting estimated elliptical orbit. Figure 3.5 shows the evolution of the estimate of x_0 (the north position of the center of the ellipse) as new GPS data points are included in the recursion. It is clear that the estimate converges to the true value of 50 m. Figure 3.6 also shows the evolution of the estimate of the ellipse major axis. Again, the estimate converges to the true value. Estimates for the other ellipse parameters (minor axis, rotation angle, and east location of the center) follow similar trends.

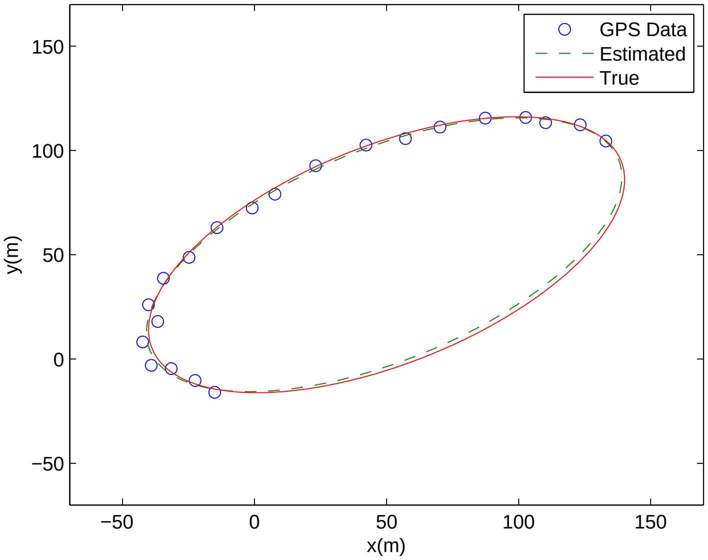


Figure 3.4: Estimated drogue orbit from simulated noisy data.. The solid line represents the true orbit. The circles represent simulated noisy GPS data points. The dashed line represents the estimated drogue orbit.

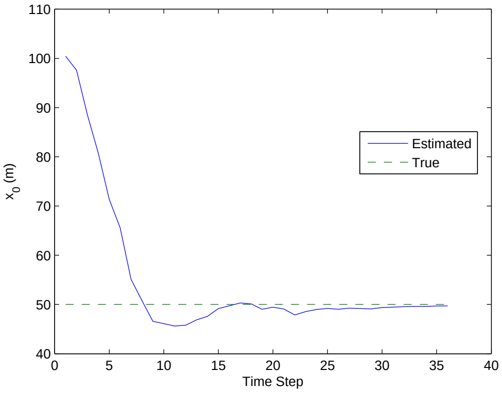


Figure 3.5: Estimate of x_0 using Halir-Flusser method and RLS.

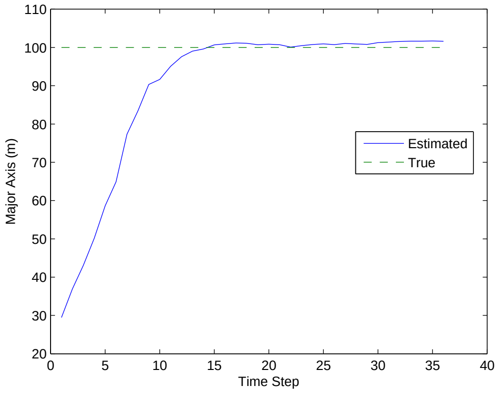


Figure 3.6: Estimate of major axis using Halir-Flusser method and RLS.

This method produces significantly improved estimates over the gradient descent algorithm, even with the introduction of noise. The algorithm converges after only 10-20 iterations. A good estimate is found after GPS data corresponding to approximately one half of the complete orbit is received and incorporated into the estimate.

3.5 Out-of-Plane Orbit Estimation

Ideally, the drogue will travel in a circular orbit in a plane parallel to the ground. In practice, the actual drogue orbit is not circular and is not restricted to the horizontal plane due to 1) wind and other disturbances, 2) unmodeled mothership-drogue dynamics, and 3) the numerical approximations inherent in the inverse dynamics to find the required mothership orbit from the desired circular drogue orbit. This, in fact, has been observed in initial experiments to validate the mothership-drogue dynamic model [6, 7]. The purpose of this section is to extend the methods of Sections 3.3 and 3.4, by which elliptical orbits in the horizontal plane may be estimated, to the more general problem of estimating arbitrarily rotated, out-of-plane elliptical orbits.

3.5.1 Method

There are five parameters that describe an ellipse in the horizontal $(x-y)$ plane,

$$\{(x,y) \mid f(x_0,y_0,a,b,\psi)=0\}, \tag{3.26}$$

where x_0 and y_0 describe the center of the ellipse, a and b are the major and minor axes, respectively, and ψ is the rotation of the ellipse from the x-axis. An ellipse in 3-space is described by eight parameters

$$\{(x,y,z) \mid f(x_0,y_0,z_0,a,b,\psi_1,\theta,\psi_2)=0\}, \tag{3.27}$$

where x_0, y_0, and z_0 describe the center of the ellipse, a and b are the major and minor axes, respectively, and ψ_1, θ, and ψ_2 are 3-2-3 Euler angles describing the rotation of the ellipse [12]. With the added degrees of freedom, estimating the ellipse becomes significantly more difficult. The approach taken in this research is to estimate the plane on which the points describing the ellipse lie. The normal vector describing that plane gives information about θ and ψ_1. The measured ellipse points are transformed into the coordinate frame describing the plane (the "ellipse frame"), and the ellipse parameters are estimated in the manner that was developed in Sections 3.3 and 3.4.

The transformation from the inertial frame to the ellipse frame is given by $R = R_{\psi_2} R_\theta R_{\psi_1}$, where $R_{\psi_1}$is a rotation about the inertial z-axis, R_θ is a rotation about the y-axis of the first intermediate frame, and R_{ψ_2} is a rotation about the z-axis of the second intermediate frame. The final orientation is given by

$$R_{\psi_2} R_\theta R_{\psi_1} = \begin{bmatrix} \cos\psi_2 & \sin\psi_2 & 0 \\ -\sin\psi_2 & \cos\psi_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi_1 & \sin\psi_1 & 0 \\ -\sin\psi_1 & \cos\psi_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{3.28}$$

The parametric form for an ellipse described in the inertial frame is given by

$$\begin{bmatrix} x - x_0 \\ y - y_0 \\ z - z_0 \end{bmatrix} = R_{\psi_1}^T R_\theta^T R_{\psi_2}^T \begin{bmatrix} a\cos(\sigma) \\ b\sin(\sigma) \\ 0 \end{bmatrix} \tag{3.29}$$

or

$$\begin{aligned} x - x_0 = {} & a\cos(\sigma)\cos(\psi_2)\cos(\theta)\cos(\psi_1) - b\sin(\sigma)\sin(\psi_2)\cos(\theta)\cos(\psi_1) - \\ & a\cos(\sigma)\sin(\psi_2)\sin(\psi_1) - b\sin(\sigma)\cos(\psi_2)\sin(\psi_1) \end{aligned} \tag{3.30}$$

$$\begin{aligned} y - y_0 = {} & a\cos(\sigma)\cos(\psi_2)\cos(\theta)\sin(\psi_1) - b\sin(\sigma)\sin(\psi_2)\cos(\theta)\sin(\psi_1) + \\ & a\cos(\sigma)\sin(\psi_2)\cos(\psi_1) + b\sin(\sigma)\cos(\psi_2)\cos(\psi_1) \end{aligned} \tag{3.31}$$

$$z - z_0 = -a\cos(\sigma)\cos(\psi_2)\sin(\theta) + b\sin(\sigma)\sin(\psi_2)\sin(\theta) \ . \tag{3.32}$$

The plane on which the ellipse points lie can be described in vector form as $\mathbf{db} = \mathbf{0}$, where $\mathbf{d} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix}^T$. For N measurements of the drogue position, the equation describing the orbit plane becomes $\mathbf{Db} = \mathbf{0}$, where

$$\mathbf{D} = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & z_N & 1 \end{bmatrix}. \tag{3.33}$$

Using Singular Value Decomposition (SVD) [14], the equation can be written as $\mathbf{U}\Sigma\mathbf{V}^T\mathbf{b} = \mathbf{0}$. The column of $\mathbf{V}$ that corresponds to the smallest singular value is the solution vector containing the plane parameters $\mathbf{b}$. The plane's normal vector, $\mathbf{n} = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}$, can then be calculated from

$$\begin{aligned} n_x &= \frac{b_1}{\sqrt{b_1^2+b_2^2+b_3^2}} \\ n_y &= \frac{b_2}{\sqrt{b_1^2+b_2^2+b_3^2}} \\ n_z &= \frac{b_3}{\sqrt{b_1^2+b_2^2+b_3^2}} \end{aligned} . \tag{3.34}$$

The normal vector can be written in the ellipse frame as $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, and related to inertial frame using

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = R_{\psi_2} R_\theta R_{\psi_1} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}. \tag{3.35}$$

The angle ψ_1 can then be calculated from

$$\tan\psi_1 = \frac{n_y}{n_x}, \tag{3.36}$$

and the normal vector is expressed in the first intermediate frame as $\mathbf{n}' = R_{\psi_1}\mathbf{n}$. The angle θ is calculated from

$$\tan\theta = \frac{n'_x}{n'_z}. \tag{3.37}$$

All data points can now be converted into the ellipse frame using

$$\begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = R_\theta R_{\psi_1} \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \tag{3.38}$$

and the remaining parameters a, b, x_0'', y_0'', ψ_2 are estimated using Halir's method as described in Section 3.3.2, where x_0'' and y_0'' are the center of the ellipse in the ellipse frame. Finally, x_0'' and y_0'' can be transformed into the inertial frame using

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = R_{\psi_1}^T R_\theta^T \begin{bmatrix} x_0'' \\ y_0'' \\ z_0'' \end{bmatrix}, \tag{3.39}$$

where z_0'' is 0.

After using these methods to obtain initial estimates of the normal vector and ellipse parameters, RLS is again employed to update the estimates as new drogue measurements become available. The RLS algorithm is applied twice. First, it is used to update the estimate of the normal vector that describes the plane on which all the measurements lie. Second, the most recent measurement is rotated into that frame, and RLS is used to update the ellipse parameter estimates.

3.5.2 Simulation Results

The results from the out-of-plane orbit estimation are shown in Figures 3.7-3.10. Again, simulated GPS data with added Gaussian noise is used as input to the estimator. The initial estimates are computed using the out-of-plane method combined with Halir's method. After obtaining a good initial estimate, the RLS algorithm is used to fine tune the parameter estimates. Figure 3.7 shows a 3D plot of the out-of-plane ellipse estimation.

Figure 3.8 shows the evolution of the estimate of x_0, and Figure 3.9 shows the evolution of the estimate for the major axis. Figure 3.10 shows the evolution of the estimate of the Euler angle θ. Results for the other parameters show similar trends. In these plots, the initial estimate method

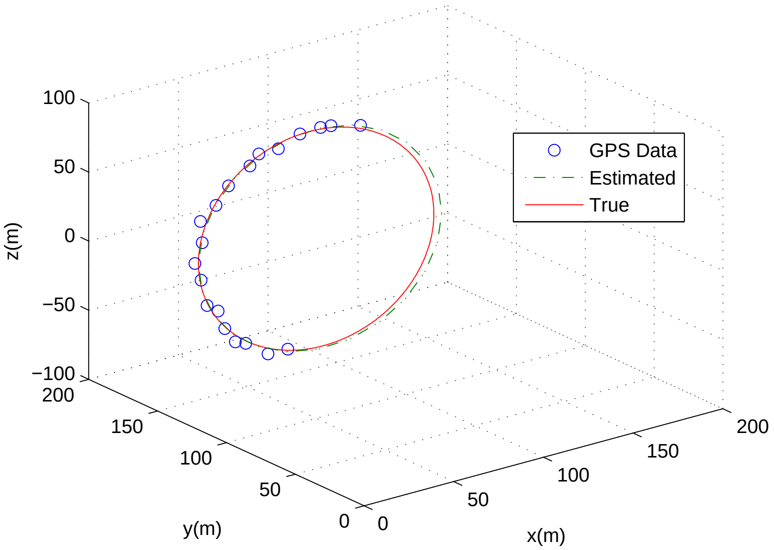


Figure 3.7: Estimated out-of-plane drogue orbit from simulated noisy data.

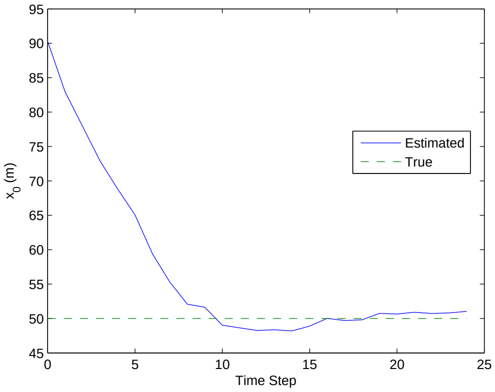


Figure 3.8: Out-of-plane estimation of x_0

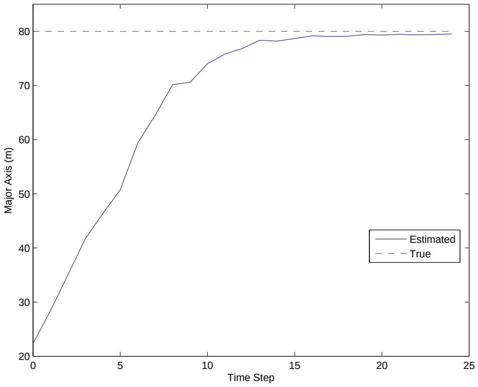


Figure 3.9: Out-of-plane estimation of the major axis

is used for the first 15 time steps (this corresponds to data for about half of the orbit), and then RLS is used from that point on.

3.6 Chapter Summary

This chapter describes methods for orbit estimation that allow the MAV to determine the drogue orbit. Numerically stable methods for making an initial estimate are described, and a new

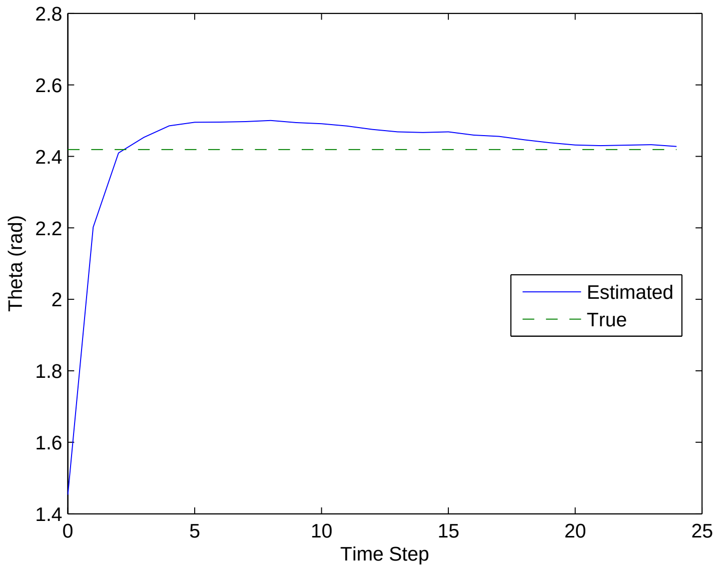


Figure 3.10: Out-of-plane estimation of θ

method of estimating an arbitrarily rotated ellipse is also presented. Computationally inexpensive parameter update methods that use the latest GPS measurement to fine tune the current estimates are also described. These algorithms were demonstrated in MATLAB using simulated GPS data with added Gaussian noise.

CHAPTER 4. ORBIT TRACKING

Chapter 3 described methods that allow an approaching MAV to estimate the elliptical orbit of the drogue. This chapter describes methods that enable the MAV to enter and track the drogue orbit.

To match the attitudes and velocities of the drogue and MAV, the MAV inserts itself onto the drogue orbit at a point behind the drogue and then tracks the orbit at a speed slightly higher than that of the drogue until rendezvous occurs. The control approach taken in this work is to decouple lateral control (for horizontal tracking) from longitudinal control (for altitude tracking). The horizontal orbit is the elliptical projection of the drogue orbit onto the horizontal plane, which is estimated using the methods described in Chapter 3.

4.1 Lateral Control

The lateral control to achieve elliptical orbit tracking of the MAV is accomplished using a vector field method based on the work done by Barber for circular orbits, but is expanded in this thesis to apply to the tracking of elliptical orbits [22, 23]. First, the coordinate frame is rotated so that there is no rotation of the ellipse from the x-axis (i.e., the major axis is aligned with the x-axis). This transformation is done using the rotation matrix

$$R_{\psi_2} = \begin{bmatrix} \cos\psi_2 & \sin\psi_2 & 0 \\ -\sin\psi_2 & \cos\psi_2 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.1}$$

where ψ_2 is the rotation of the ellipse from the x-axis as calculated in Section 3.5. By applying the transformation R_{ψ_2} to Equation (3.5), the resulting parametric equation for the ellipse becomes

$$\begin{array}{c} x - x_0 = a\cos(\sigma) \\ y - y_0 = b\sin(\sigma) \end{array}. \tag{4.2}$$

The partial derivatives of these equations with respect to the free parameter σ are

$$\begin{array}{c} \frac{\partial x}{\partial \sigma} = -a\sin(\sigma) \\ \frac{\partial y}{\partial \sigma} = b\cos(\sigma) \end{array}. \tag{4.3}$$

The desired course for the MAV, if it lies on the ellipse, is tangent to the curve. Thus the commanded course for the MAV, χ^c, is given by

$$\tan(\chi^c) = \frac{\partial y}{\partial x} = \frac{\frac{\partial y}{\partial \sigma}}{\frac{\partial x}{\partial \sigma}} = \frac{b\cos(\sigma)}{-a\sin(\sigma)}. \tag{4.4}$$

To eliminate the parameter σ from (4.4) we can solve (4.2) for $\cos(\sigma)$ and $\sin(\sigma)$, yielding

$$\tan(\chi^c) = \frac{b^2(x - x_0)}{-a^2(y - y_0)}. \tag{4.5}$$

If the MAV is not on the desired ellipse, then the commanded course must be modified. The error can be written as

$$1 - \frac{(x - x_0)^2}{a^2} - \frac{(y - y_0)^2}{b^2} = \varepsilon. \tag{4.6}$$

Whether the MAV is on or off of the elliptical orbit, its desired course is calculated using

$$\begin{array}{c} \Delta y = -b^2(x - x_0) + ka(y - y_0)\varepsilon \\ \Delta x = a^2(y - y_0) + ka(x - x_0)\varepsilon \end{array} \tag{4.7}$$

and

$$\chi^c = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) + \psi_2, \tag{4.8}$$

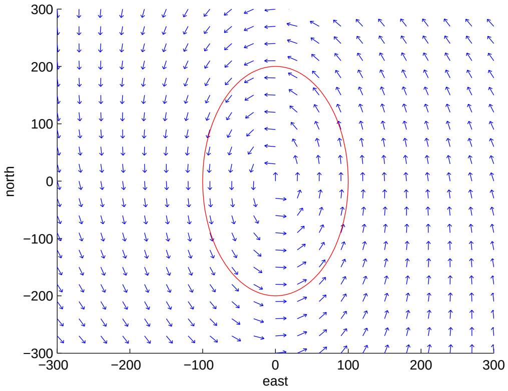


Figure 4.1: Vector field plot for an elliptical orbit with a small gain k.

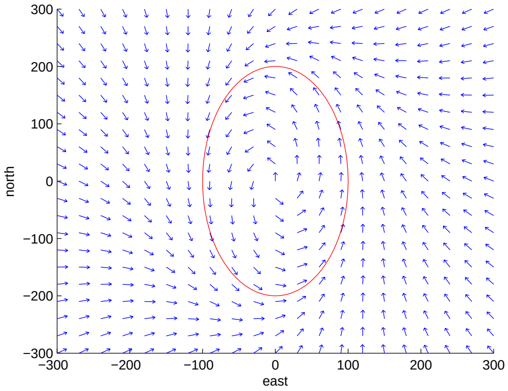


Figure 4.2: Vector field plot for an elliptical orbit with a large gain k.

where ψ_2 is the rotation of the ellipse from the x-axis, χ^c is the commanded course for the MAV, and k is a gain used to tune the resulting vector field. The addition of the error term allows the desired heading to point towards the center of the ellipse if the MAV is very far away from this desired orbit. For example, if the MAV is very far away in the x-direction (north), then the term $x - x_0$ will be large compared to $y - y_0$. Subsequently, the Δx term will be large compared to the Δy term. Because the error term is negative in this case, both Δx and Δy will be negative. Taking the inverse tangent will then result in a solution of nearly 180°, which would point the UAV directly towards the center of the ellipse. Similarly, if the MAV is very far away in the y-direction (east), then the term $y - y_0$ will be large compared to $x - x_0$. Subsequently, the Δy term will be large compared to the Δx term. Because the error term is negative in this case, both Δx and Δy will be negative. Taking the inverse tangent will then result in a solution of nearly $-90°$, which would point the UAV directly towards the center of the ellipse.

It is important to note that course (χ) is used instead of heading (ψ) because in the presence of a crosswind, the UAV needs to crab into the wind to maintain the desired course. Essentially, using course provides a way to reject the wind disturbance. Figures 4.1 and 4.2 show the effect of the gain k on the vector field as described by (4.8). Higher values for k cause the MAV to be driven more quickly towards the ellipse, but if k is too large, the system will become unstable. Too small of a gain k will lead to very slow convergence and large steady state error.

4.1.1 Feed-Forward Roll Angle

Equation (4.8) will yield a steady state error because the course control system is type I, and the input is of higher order than a step input. Barber addressed two methods of minimizing the steady-state error [22]. The first method is an alternate orbit size command. Based on the error in the system, the UAV can be commanded to track an ellipse with smaller major and minor axes so that the UAV tracks the orbit originally desired. A lookup table can be generated so that for any desired orbit size, the needed commanded orbit size can be determined. This method is less robust to disturbances such as wind gusts.

The other method described by Barber is to add a feed-forward term to the roll command that takes into account the change in course. If the UAV lies exactly on the orbit being tracked, its commanded roll angle would be 0, but that is not nominal, since the UAV is tracking an ellipse. Assuming coordinated turns, the nominal roll angle for a UAV on the orbit can be calculated by

$$\phi_{nom} = \frac{\dot{\chi}V}{g}, \tag{4.9}$$

where $\dot{\chi}$ is the course rate, V is the groundspeed of the UAV, and g is gravity. The roll angle commanded to the UAV can then be changed to

$$\phi^c = \phi_{nom} + k\left(\chi_d - \chi\right), \tag{4.10}$$

where k is the course to roll gain, and $\chi_d - \chi$ is the difference between the desired course and the actual course of the UAV. This method will reject disturbances such as wind gusts, and is generally more robust than the alternate orbit size method.

4.1.2 MATLAB Simulation Results

Figure 4.3 shows a simulation of the MAV entering and tracking an elliptical orbit using the vector field approach. In this simulation, the MAV starts in the center of the ellipse, enters the elliptical orbit, and executes a complete orbit. The MAV tracks the orbit to within one meter, as shown in Figure 4.4.

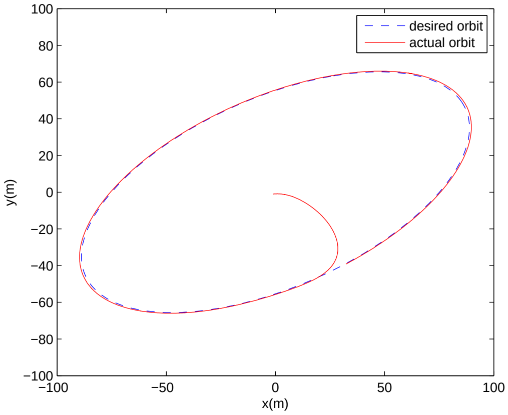


Figure 4.3: Elliptical orbit tracking. The dashed line represents the drogue orbit, and the solid line represents the MAV trajectory as it tracks the desired orbit, starting from an initial position at the center of the orbit.

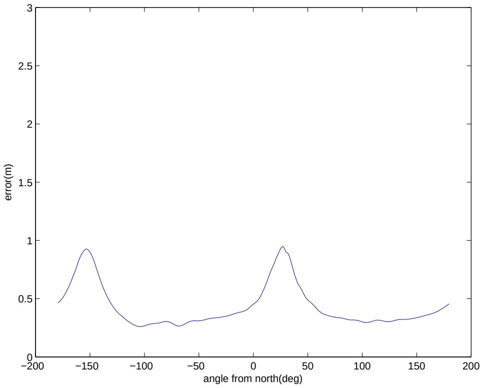


Figure 4.4: Error in tracking an elliptical orbit for a single orbit. The error is consistently less than 1 m for all locations on the orbit.

These plots show the vector field approach with no error correction. In this lower fidelity simulation, the steady state error was small enough that error correction techniques were not needed. In the higher fidelity simulations and flight tests shown in Chapter 5, the steady state error becomes significant, and the error removal techniques are employed.

The only gain that needed to be tuned in these simulations was the gain k that controls the intensity of the vector field as described in Equation (4.7). A low gain results in a large steady state error. A high gain results in instability and an inability for the UAV to converge due to overshoot. This high gain causes oscillations of the UAV around the desired orbit.

4.2 Longitudinal Control

The first step in the longitudinal control is to calculate the desired altitude of the MAV based on its current East-West location and the general ellipse Equations (3.30)-(3.32). Determining the desired altitude can be done by manipulating Equations (3.30) and (3.31), which can be rewritten as

$$\left.\begin{aligned} 0 &= h_{11}\cos(\sigma)+h_{12}\sin(\sigma)+h_{13} \\ 0 &= h_{21}\cos(\sigma)+h_{22}\sin(\sigma)+h_{23} \end{aligned}\right., \qquad (4.11)$$

where

$$
\begin{array}{rcc}
h_{11} & = & a\cos(\psi_2)\cos(\theta)\cos(\psi_1) - a\sin(\psi_2)\sin(\psi_1) \\
h_{12} & = & -b\sin(\psi_2)\cos(\theta)\cos(\psi_1) - b\cos(\psi_2)\sin(\psi_1) \\
h_{13} & = & x - x_0 \\
h_{21} & = & a\cos(\psi_2)\cos(\theta)\sin(\psi_1) + a\sin(\psi_2)\cos(\psi_1) \\
h_{22} & = & -b\sin(\psi_2)\cos(\theta)\sin(\psi_1) + b\cos(\psi_2)\cos(\psi_1) \\
h_{23} & = & y - y_0
\end{array}
. \qquad (4.12)
$$

Equation 4.11 can then be written as

$$\mathbf{Hs} = \mathbf{0}, \qquad (4.13)$$

where **0** is a 3 by 1 zero vector, and **H** and **s** are defined as

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \end{bmatrix}, \mathbf{s} = \begin{bmatrix} \cos(\sigma) \\ \sin(\sigma) \\ 1 \end{bmatrix}. \qquad (4.14)$$

Equation 4.14 can be decomposed using SVD as $\mathbf{U}\Sigma\mathbf{V}^T\mathbf{s} = \mathbf{0}$, and the solution vector **s** is the column of the **V** matrix that corresponds to the smallest singular value. The parameter σ can then be determined by

$$\sigma = \tan^{-1}\left(\frac{v_2}{v_1}\right). \qquad (4.15)$$

Once σ is found, it can then be inserted into Equation 3.32, and the desired altitude is then sent to the MAV.

In summary, the desired altitude of the MAV is calculated by rewriting Equations (3.30) and (3.31) and then solving for the free parameter σ. That free parameter is then inserted into Equation (3.32) to calculate the desired altitude z.

4.2.1 Feed-Forward Flight Path Angle

The longitudinal controller faces the same problem that the lateral controller does. Since the flight path angle control system is type I, there will be a steady-state error. Similar solutions can be applied to the altitude controller as were used in the lateral controller. An alternate altitude method can be used, where an additive altitude term can be calculated using the equation $z_a = k_1 \dot{z}$, where $\dot{z}$ can be calculated by taking the time derivative of Equation (3.32), and k_1 is a tunable gain. The commanded altitude then becomes

$$z^c = z_d + z_a, \tag{4.16}$$

where z_d is the desired altitude of the MAV as computed in the previous section.

Alternatively, a feed-forward flight path angle term can be added to the controller. A nominal flight path angle can be calculate from the altitude rate as $\gamma_{nom} = k_1 \dot{z}$, where k_1 is a tunable gain. The commanded flight path angle can then be written as

$$\gamma^c = \gamma_{nom} + k_2 \left(z_d - z \right). \tag{4.17}$$

Using Equation (4.17) and tuning the gain k_2 properly will minimize the error in the longitudinal tracking.

4.2.2 MATLAB Simulation Results

Figure 4.5 shows a 3D view of the tracking results for an ellipse rotated out of the plane. In simulation, the altitude of the MAV matches the altitude of the drogue to within 1 m for an entire orbit, as shown in Figure 4.6.

As in the lateral control, the steady state error in these lower fidelity MATLAB simulations was not significant enough to merit the use of the error removal techniques. In the high fidelity simulations and flight tests shown in Chapter 5, the steady state error becomes significant, and the error removal methods are demonstrated. In these plots the command input was the desired altitude, so no control parameters needed to be tuned.

4.3 Chapter Summary

This chapter explains the control algorithms that allow a MAV to track an arbitrarily rotated ellipse. The approach used is to decouple the lateral and longitudinal controllers. For the lateral controller, a vector field method is used. Previous techniques used in circle tracking were expanded

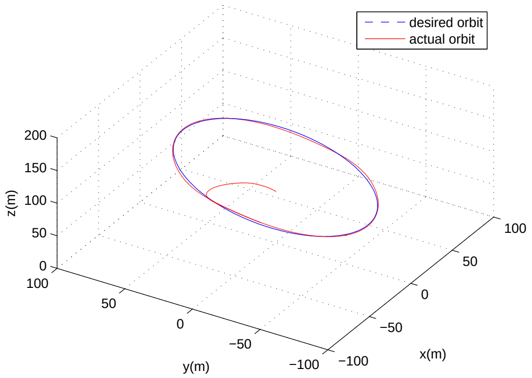


Figure 4.5: 3D view of elliptical orbit tracking. The dashed line represents the drogue orbit, and the solid line represents the MAV trajectory as it tracks the desired orbit, starting from an initial position at the center of the orbit. The orbit being tracked has a major axis of 100 m, a minor axis of 50 m, an in-plane rotation of 30°, and an out-of-plane rotation of 10°.

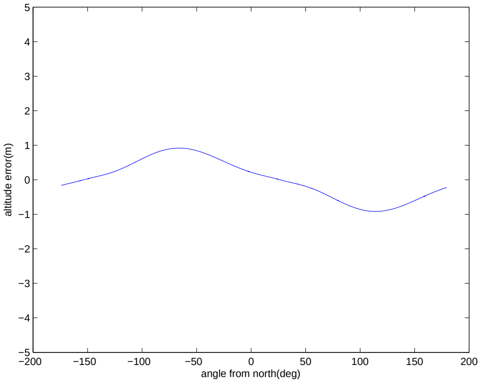


Figure 4.6: Altitude error for MAV tracking the drogue during an elliptical orbit

to apply to tracking of ellipses. For the longitudinal controller, a method for calculating the desired altitude based on the position on the MAV is described. Also, two methods of eliminating steady state error are presented. Results from MATLAB simulations demonstrating the control algorithms are also shown and demonstrate the functionality of the algorithms.

CHAPTER 5. FLIGHT TEST RESULTS

This Chapter describes the results from all the flight tests performed during this thesis work. Section 5.1 describes the hardware and software setup used in the flight tests. Section 5.2 shows the results from the mothership-drogue flight tests. Sections 5.3 and 5.4 show the results from the orbit estimation and orbit tracking algorithms respectively.

5.1 Sensors, Hardware and Software

To implement the estimation and control described in Chapters 3 and 4, accurate position and attitude measurements must be made. This section describes the hardware and software used in the flight tests performed for this thesis.

5.1.1 Kestrel Autopilot

The Kestrel autopilot shown in Figure 5.1 is the central piece in a flight test scenario. It is the controller for the aircraft, controlling the low level loops such as roll, pitch, and yaw. It also can control higher level loops such as heading and altitude and can control the UAV at the highest level for tracking of waypoints and circular orbits. The Kestrel performs the pose calculations, communicates with the ground station, and interfaces with the servos that control the aircraft. It is equipped with rate gyros and accelerometers, allowing it to to serve as an onboard IMU. Pressure sensors and a GPS antenna also provide input for position and attitude calculations. Further information can be found in reference [24].

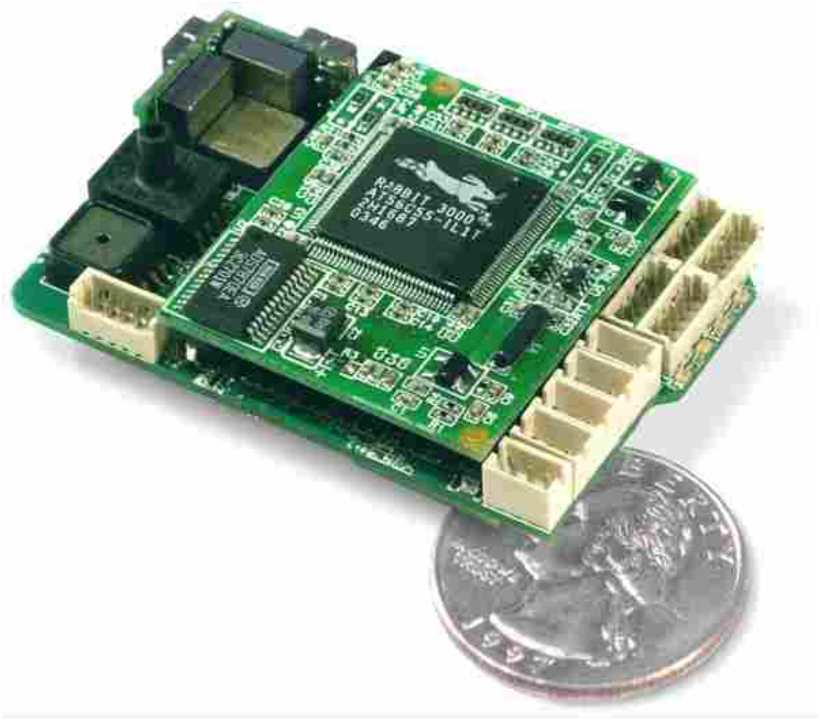


Figure 5.1: Kestrel autopilot

5.1.2 Commbox

The ground station commbox houses another Kestrel autopilot loaded with a different firmware and serves as a connection between the UAV and the ground station. The commbox is shown in Figure 5.2.

5.1.3 Virtual Cockpit

Virtual Cockpit (Figure 5.3) serves as the user interface that shows what the UAV is doing while in flight. The GUI shows the attitude of the aircraft and an overhead map view of the flight path. Waypoints and rally locations can be set by the user and uploaded to the onboard autopilot from Virtual Cockpit. This user interface also allows the user to send heading, roll, and altitude commands to manually control the aircraft. More information can be found in reference [25].

A group of MATLAB Mex functions were created that allow Virtual Cockpit to communicate with MATLAB. The state estimates calculated on the onboard autopilot are sent from Virtual Cockpit to MATLAB, where the orbit tracking algorithms are implemented. The commanded

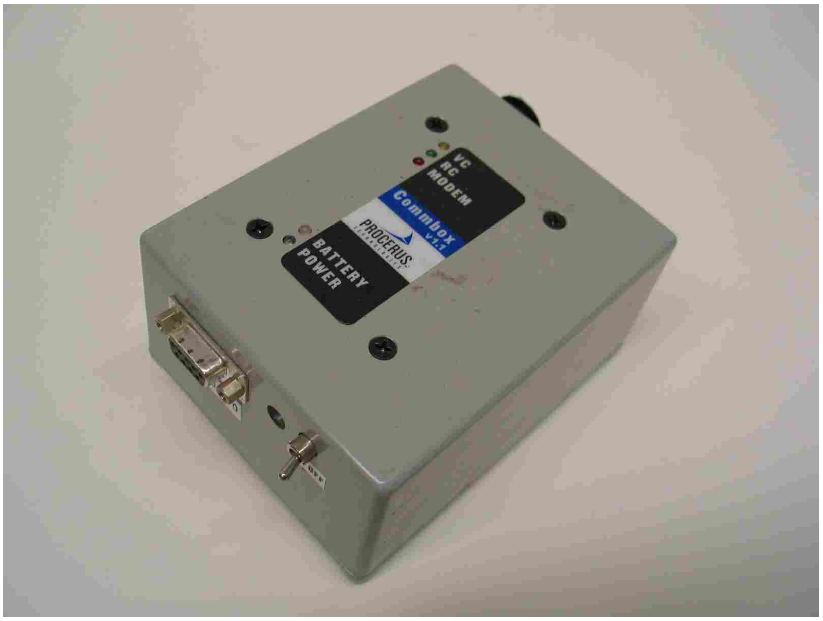
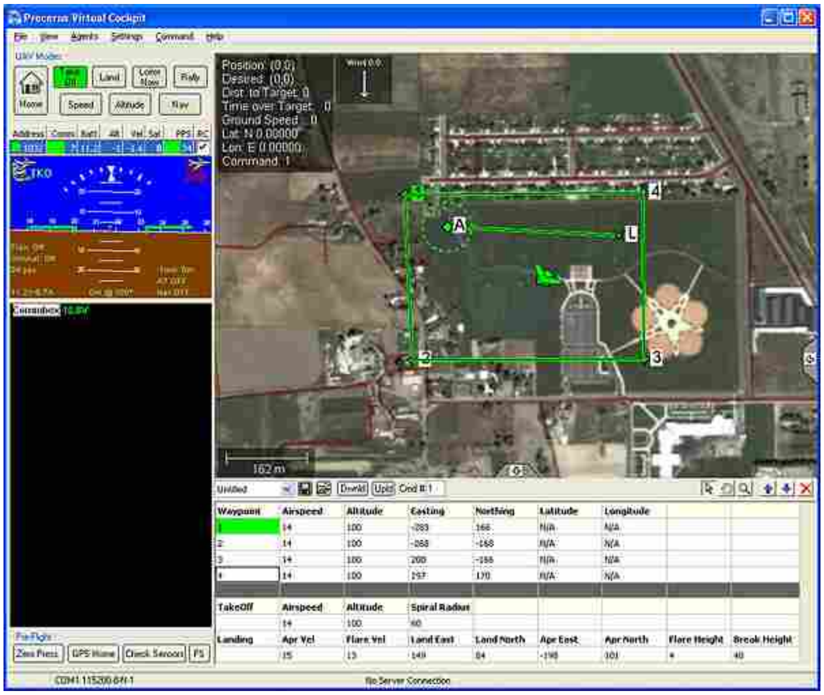


Figure 5.2: Commbox



Position: (0,0)
Desired: (0,0)
Dist. to Target: 0
Time over Target: 0
Ground Speed: 0
Lat: N 0.00000
Lon: E 0.00000
Command: 1

Cmd #: 1

Waypoint	Airspeed	Altitude	Easting	Northing	Latitude	Longitude		
1	14	100	-283	166	N/A	N/A		
2	14	100	-268	-168	N/A	N/A		
3	14	100	200	-156	N/A	N/A		
4	14	100	197	170	N/A	N/A		
TakeOff	Airspeed	Altitude	Spiral Radius					
	14	100	60					
Landing	Apr Vel	Flare Vel	Land East	Land North	Apr East	Apr North	Flare Height	Break Height
	15	13	149	84	-198	101	4	40

COM1 115200-8-N-1

No Server Connection

Figure 5.3: Virtual Cockpit

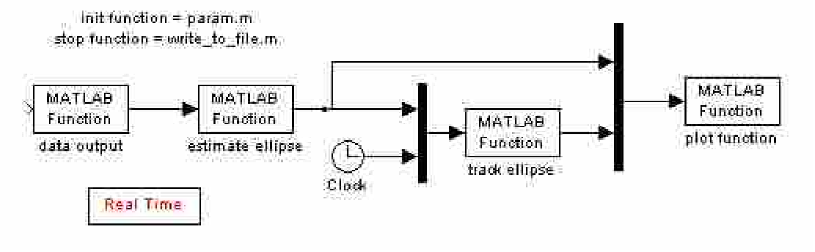


Figure 5.4: Simulink block diagram for flight tests and software in the loop simulations

headings and altitudes are then sent back through Virtual Cockpit to the onboard autopilot, where new commands are sent to the servos that drive the control surfaces of the aircraft. The Simulink block diagram used in the flight tests and in the software-in-the-loop simulations is shown in Figure 5.4.

5.1.4 Software-in-the-Loop (SITL) Testing

To refine the algorithms and tune gains in a setting that more accurately simulates the conditions of live flight tests, SITL simulations were developed. These tests closely resemble actual flight tests, but instead of communicating to the onboard autopilot, Virtual Cockpit communicates with a flight simulation program called Aviones. Aviones simulates the flight dynamics and pose estimation that would occur on the real aircraft. Figure 5.5 shows a comparison of the SITL simulations to a flight test. Figure 5.6 shows a screen shot of the SITL simulations.

5.2 Flight Tests for Mothership-Drogue Interaction

Flight tests were conducted to demonstrate mothership-drogue interactions. In these tests, the mothership, a twin prop, 55-inch wingspan, battery-powered, autonomous aircraft with a Kestrel autopilot (Figure 5.7), was hand launched while towing a hemispherical drogue (Figure 5.8).

The mothership was commanded to track circular orbits of various radii, and the resulting GPS data for the mothership and drogue orbits were recorded. A southwesterly wind was present during the tests of a magnitude of $1-3$ m/s. Figures 5.9-5.20 show the mothership and drogue orbits from several angles for multiple orbit sizes. The mothership tracked orbits with radii of 75,

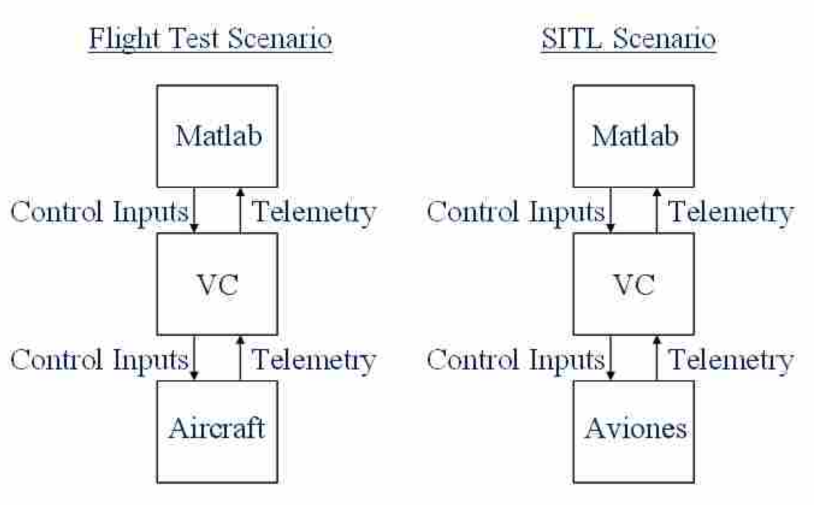
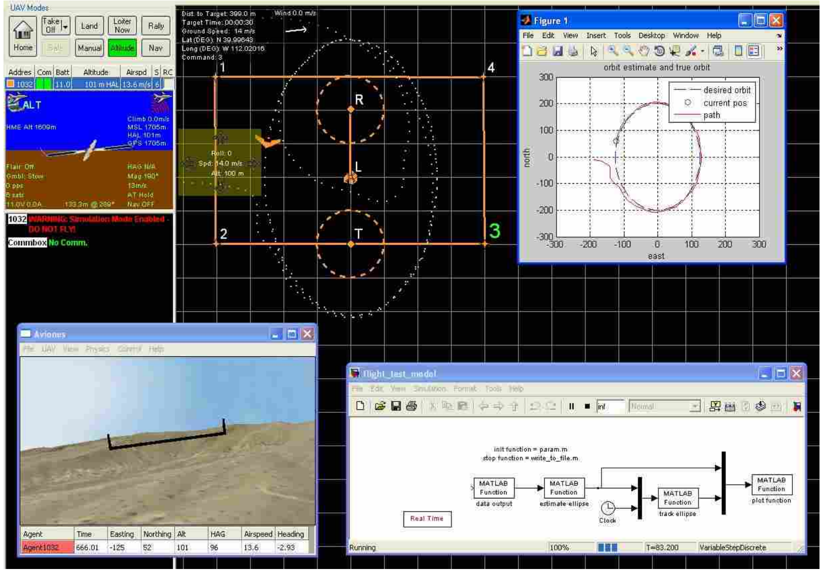


Figure 5.5: Diagram showing a comparison of SITL simulations and flight tests



Dist. to Target: 399.0 m
Target Time: 00:00:30
Ground Speed: 14 m/s
Lat (DEG): N 39.99643
Long (DEG): W 112.02016
Command: 3
Wind 0.0 m/s

Addres	Com	Batt	Altitude	Airspd	S	RC
1032		11.0	101 m HAL	13.6 m/s	6	

HME Alt 1609m
Climb 0.0m/s
MSL 1705m
HAL 101m
GPS 1705m
Flair Off
Gmbl: Stow
0 pps
6 sats
11.0V 0.0A
133.3m @ 289°
HAG N/A
Mag 190°
13m/s
AT Hold
Nav OFF

Roll: 0
Spd: 14.0 m/s
Alt: 100 m

1032 WARNING: Simulation Mode Enabled - DO NOT FLY!
Commbox No Comm.

Agent	Time	Easting	Northing	Alt	HAG	Airspeed	Heading
Agent1032	666.01	-125	52	101	96	13.6	-2.93

init function = param.m
stop function = write_to_file.m

Running 100% T=83.200 VariableStepDiscrete

Figure 5.6: Software-in-the-loop simulation

85, 100, and 150 m, and maintained an airspeed of 14 m/s for all flights. The cable length was fixed at a length of 125 m.

Figures 5.9-5.12 show the flight test results from a 3D view. The plots show flight paths for several revolutions of the mothership and drogue around their respective orbits, demonstrating that the drogue orbit is stable and repeatable, even in the presence of a constant wind. The results

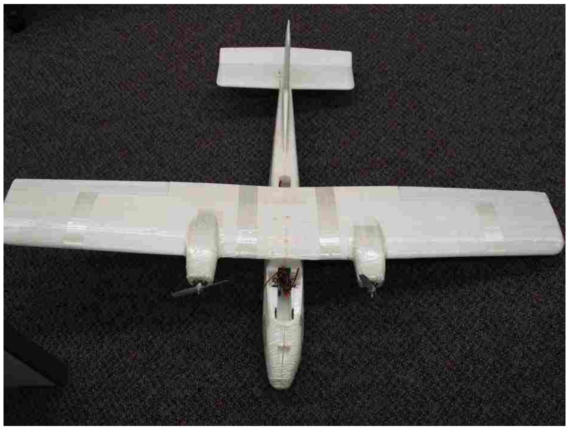


Figure 5.7: Mothership used in preliminary flight tests

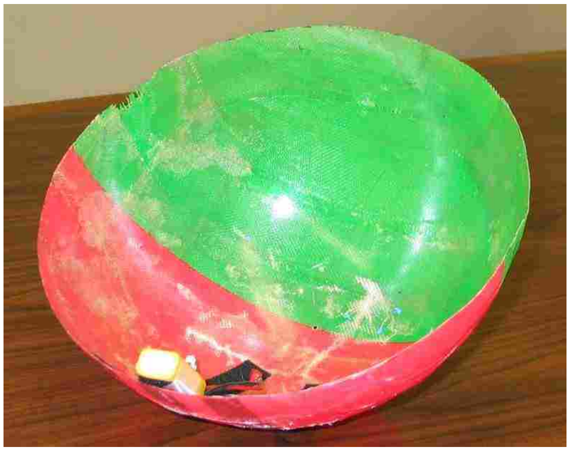


Figure 5.8: Drogue used in preliminary flight tests

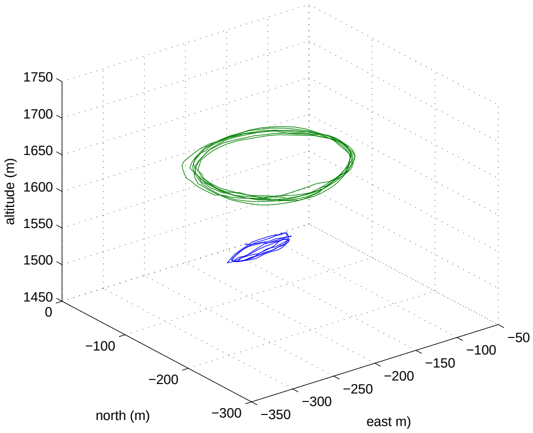


Figure 5.9: 3D plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 75 m.

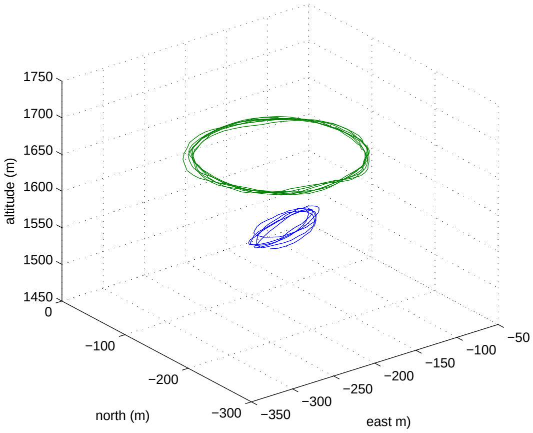


Figure 5.10: 3D plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 85 m.

illustrate the feasibility of achieving stable drogue orbits at a smaller radius and decreased speed, which is a necessity for aerial MAV recovery.

Figures 5.13-5.16 show the test results from an overhead view. It is interesting to note that as the mothership orbit gets smaller, the drogue orbit also decreases in size, but at an even greater rate. The ratio of the mothership to drogue orbits for the largest orbit size is 1.5 to 1, while the ratio for the smallest orbit size is 3 to 1.

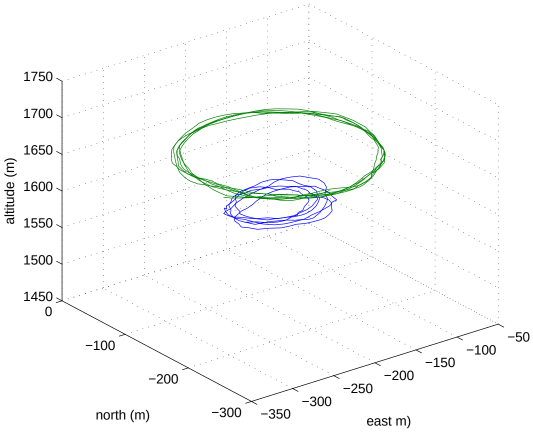


Figure 5.11: 3D plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 100 m.

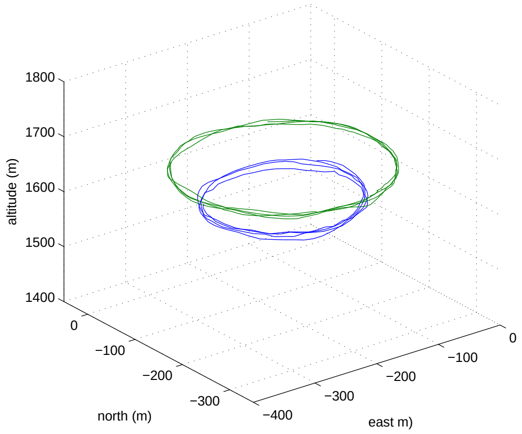


Figure 5.12: 3D plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 150 m.

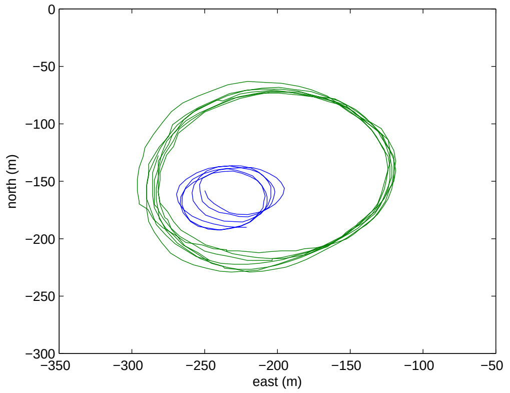


Figure 5.13: Overhead plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 75 m.

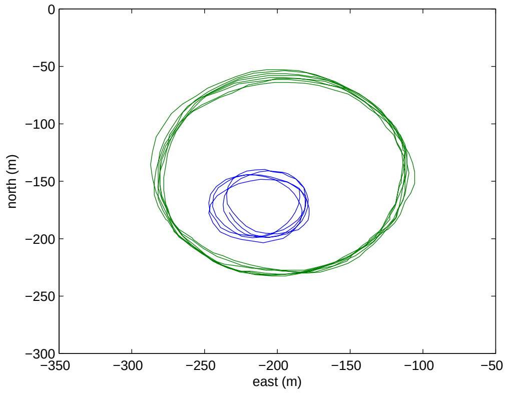


Figure 5.14: Overhead plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 85 m.

Figures 5.17-5.20 show the orbits of the mothership and drogue in the vertical (east-vertical) plane. Due to wind, the drogue orbit is tilted out of the horizontal plane in certain cases, despite the horizontal mothership orbit. It is also interesting to note that wind has a greater affect on the drogue when the mothership and drogue orbits are smaller. This is apparent because the incline of the drogue orbit is increasingly significant for smaller and smaller orbits even though the wind was fairly constant during the flight tests. One other point of interest is the fact that the difference

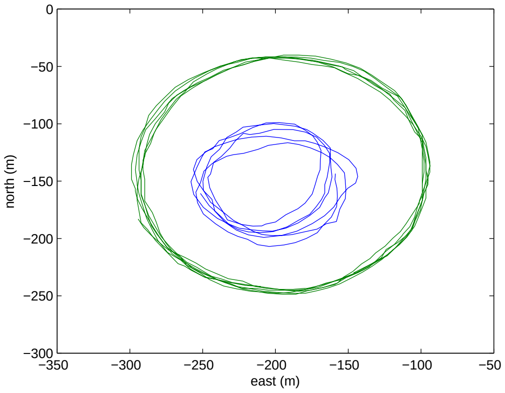


Figure 5.15: Overhead plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 100 m.

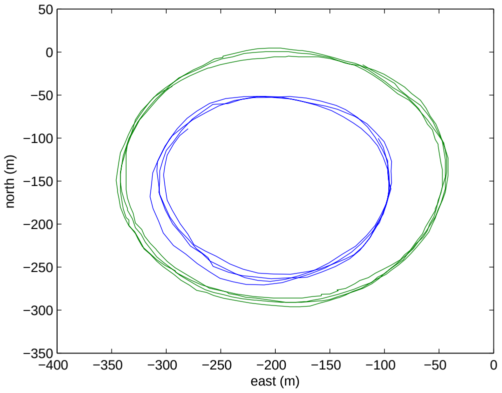


Figure 5.16: Overhead plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 150 m.

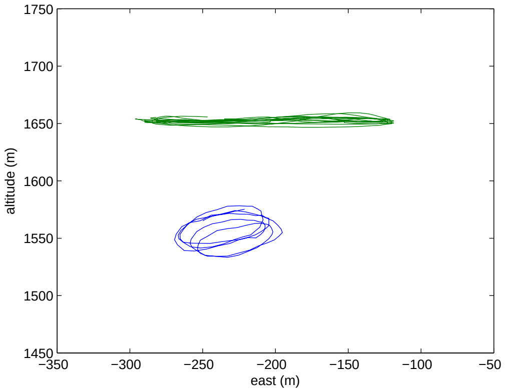


Figure 5.17: Side plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 75 m.

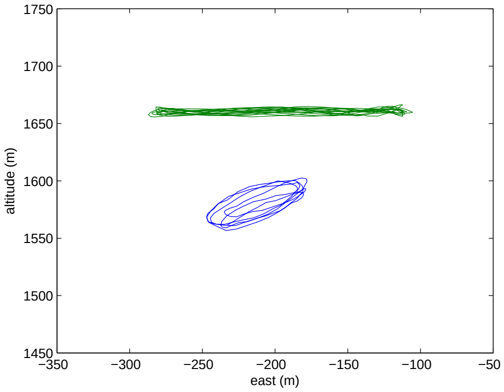


Figure 5.18: Side plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 85 m.

in altitude between the drogue and mothership is greater when the orbit sizes are smaller. For the mothership orbit size of 150 m, the difference in altitude between the mothership and drogue is about 60 m. For the mothership orbit size of 75 m, the difference in altitude is about 110 m.

5.3 Flight Tests for Orbit Estimation

The out-of-plane orbit estimation algorithm described in Section 3.5 was performed on the actual GPS data from the drogue shown in Section 5.2. Figures 5.21 and 5.22 demonstrate that the

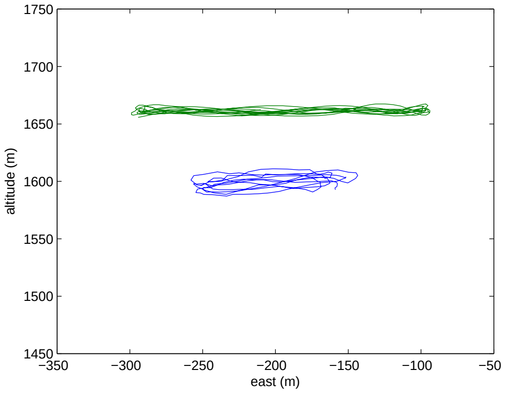


Figure 5.19: Side plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 100 m.

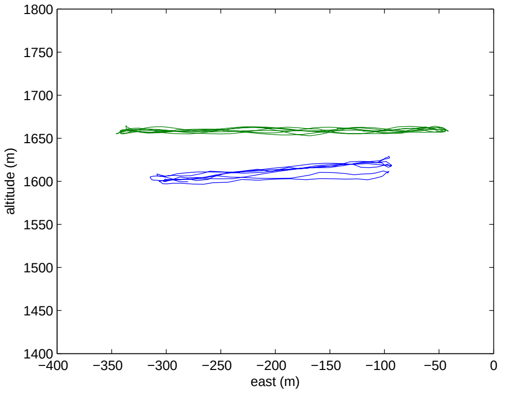


Figure 5.20: Side plot with 125 m cable, mothership airspeed of 14 m/s, and mothership radius of 150 m.

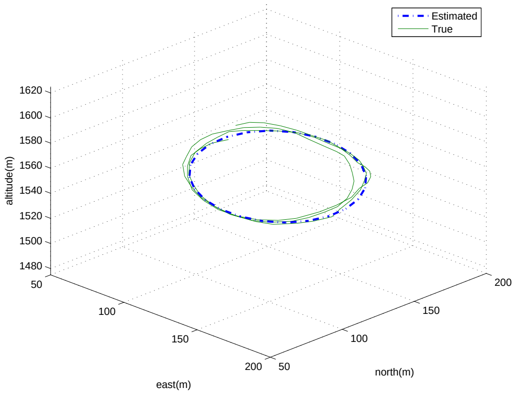


Figure 5.21: Drogue orbit estimation plot

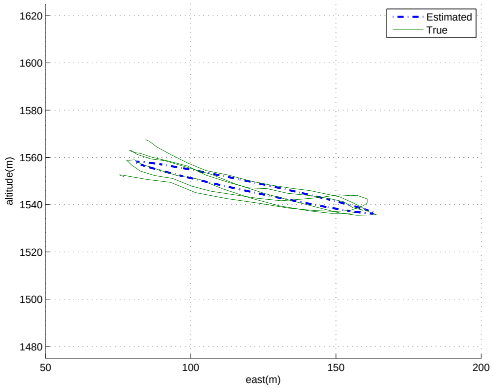


Figure 5.22: Drogue orbit estimation plot

estimation methods provide a good approximation of the flight path of the drogue. Note that the orbits shown in Figures 5.21 and 5.22 represent the estimation after all the data was received. The estimates are therefore essentially an average of all the orbits that the drogue traveled. An estimate of a single drogue orbit would match much more closely because the orbit parameters change less over the course of one orbit.

Figures 5.23-5.25 show the evolutions of the estimates of x_0, θ, and the major axis (a) respectively. The estimation converges within about 15 iterations. Since the drogue orbit is not perfect, the estimates do vary slightly over time, but signal is much stronger than the noise. Additionally, the estimate of x_0 begins to increase after about 40 iterations. This is due to the fact that

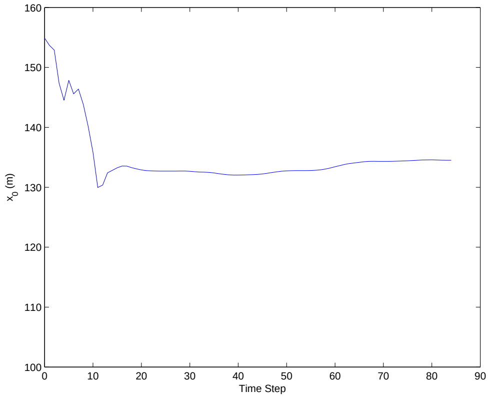


Figure 5.23: Estimate of x_0 on actual drogue data

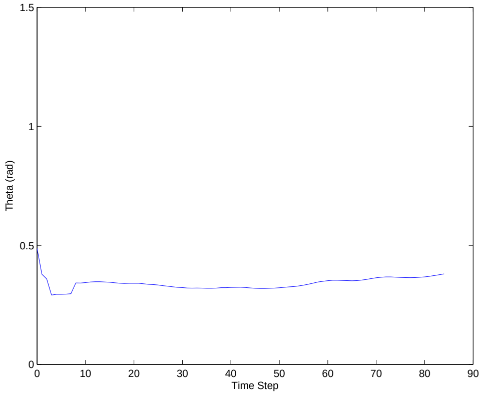


Figure 5.24: Estimate of θ on actual drogue data

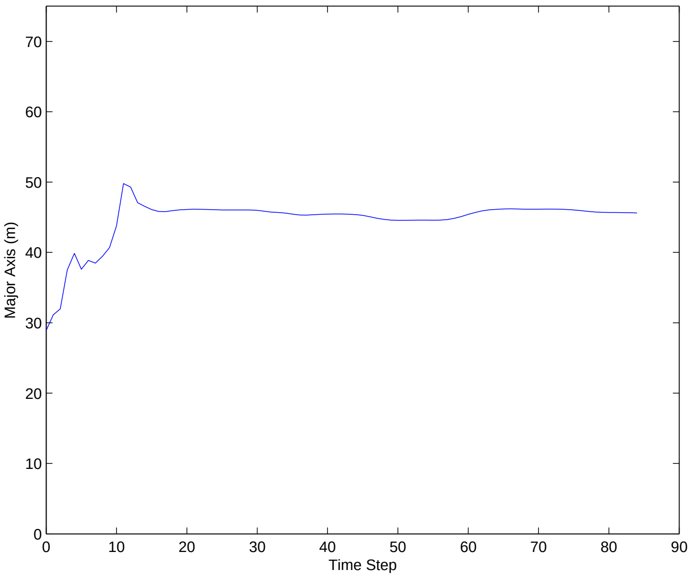


Figure 5.25: Estimate of major axis on actual drogue data

the drogue orbits are in fact drifting in the positive x direction over time, which indicates that the RLS algorithm is using the most recent data more heavily than the old data, which is desired.

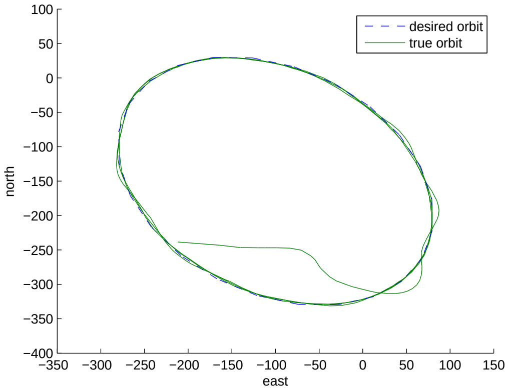


Figure 5.26: SITL elliptical orbit tracking with major axis of 200 m, minor axis of 150 m, and in-plane rotation of 45°.

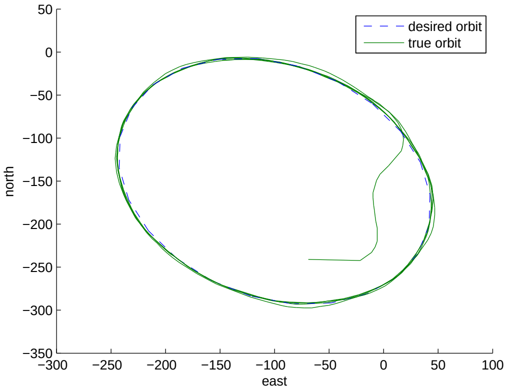


Figure 5.27: SITL elliptical orbit tracking with major axis of 150 m, minor axis of 125 m, and in-plane rotation of 45°.

5.4 Flight Tests for Orbit Tracking

SITL tests and flight tests were performed to test the orbit tracking algorithms to demonstrate their effectiveness in the presence of unmodeled disturbances such as wind gusts, inaccuracies due to reduced-order models, and other nonlinear effects that were not incorporated into the MATLAB simulation. Steady-state error removal methods were also used in these tests.

5.4.1 Lateral Control Software-in-the-Loop Simulation Results

In Section 4.1 two methods of minimizing steady-state error were discussed. For the SITL simulations, both methods were tested. Figures 5.26-5.29 show plots from the SITL simulations using the alternate orbit size method. The plots show tracking of ellipses rotated at 45° for increasingly smaller orbit sizes. The largest orbit shown has a major axis of 200 m and a minor axis of 150 m. The smallest orbit shown has a major axis of 100 m and a minor axis of 75 m.

These plots show that the alternate orbit size method does effectively remove the steady-state error. One point to note is that the error is greater for the tracking of smaller orbits. The standard deviation of error for the largest orbit size is less than 1 m, while the standard deviation of the error for the smallest orbit size is roughly 2.5 m. This is likely due to the fact that the heading rate is greater for smaller orbit sizes; thus, the UAV must fly at greater roll angles to fly

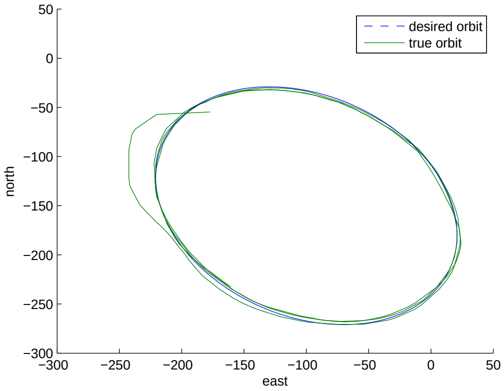


Figure 5.28: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and in-plane rotation of 45°.

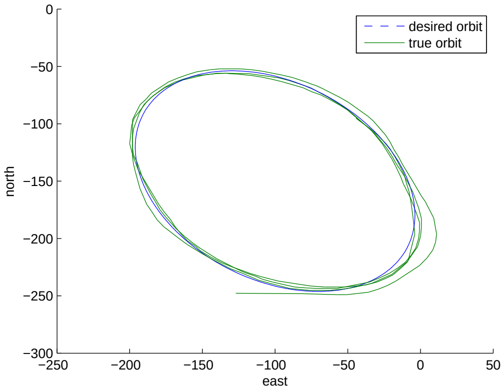


Figure 5.29: SITL elliptical orbit tracking with major axis of 100 m, minor axis of 75 m, and in-plane rotation of 45°.

the desired path. With a greater range in commanded roll angles, there is greater room for error in the controller.

Using this method, the only gain that needed to be tuned in these simulations was the gain k that controls the intensity of the vector field as described in Equation (4.7). A low gain resulted in a large steady state error. A high gain resulted in instability and an inability for the UAV to converge due to overshoot. This high gain caused oscillations of the UAV around the desired orbit as was the case in the low fidelity MATLAB simulation described in Chapter 4.

The error removal method of feed-forward roll angle was also tested in the SITL simulations. Figures 5.30 and 5.31 show the results. The left figure shows tracking of an orbit with a major axis of 200 m and a minor axis of 150 m. The right plot shows tracking of an orbit with a major axis of 100 m and a minor axis of 75 m.

These plots show that adding a feed-forward roll term effectively removes the steady-state error. As with the alternate orbit size method, the error is greater for the tracking of smaller orbits. The standard deviation of error for the largest orbit size is less than 1 m, while the standard deviation of the error for the smallest orbit size is roughly 2 m. Although the results were similar in the SITL tests, the feed-forward method of error removal will perform better in the presence of wind. Wind would add a bias which would cause the orbit to shift in the case of the alternate orbit size method, while a feed-forward term would reject that wind disturbance.

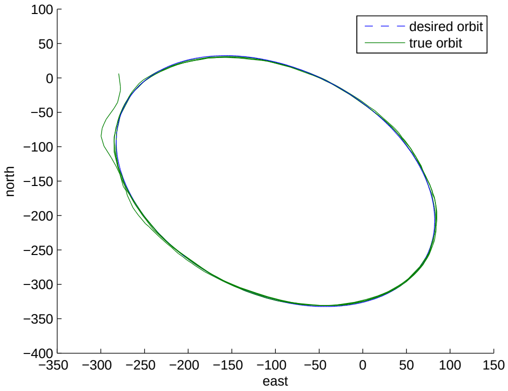


Figure 5.30: SITL elliptical orbit tracking with major axis of 200 m, minor axis of 150 m, and in-plane rotation of 45°. A feed-forward roll angle was used to remove steady state error.

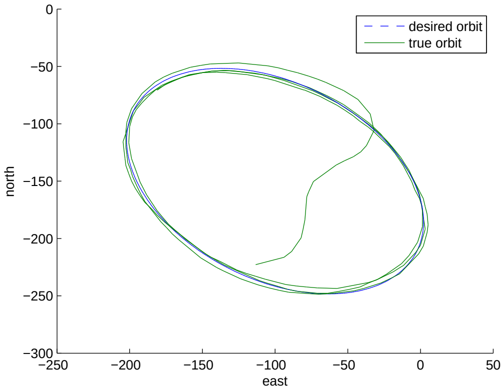


Figure 5.31: SITL elliptical orbit tracking with major axis of 100 m, minor axis of 75 m, and in-plane rotation of 45°. A feed-forward roll angle was used to remove steady state error.

Table 5.1: Table of error for altitude tracking

	$\theta = 0$	$\theta = 5$	$\theta = 10$	$\theta = 15$	$\theta = 20$
std dev of error (m)	1.1	1.5	2.4	5.3	12.8

5.4.2 Longitudinal Control Software-in-the-Loop Simulation Results

Figures 5.32-5.36 show the results from the out-of-plane ellipse tracking algorithms. Table 5.1 shows the standard deviation of the altitude tracking error for each out-of-plane rotation. The method used to remove the steady state error is the alternate altitude method. The feed-forward flight path angle method could not be used because Virtual Cockpit does not accept flight path angle commands; rather, it only accepts pitch and altitude commands.

These plots demonstrate the functionality of the orbit tracking algorithms and also its limitations. At rotations of 15° or more, the control loops start to saturate in pitch, so the altitude tracking increases its error significantly. Depending on the capabilities of the aircraft, the autopilot saturation limits could be increased so that higher pitch angles are allowed.

The only parameter that needed to be tuned is the gain k from Equation (4.16) that controls the amount that the altitude input is changed based on the altitude rate. A low gain caused the

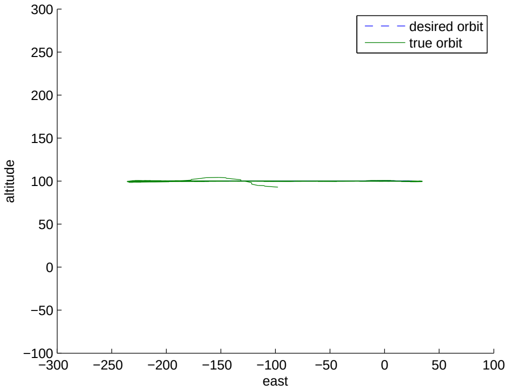


Figure 5.32: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 0°.

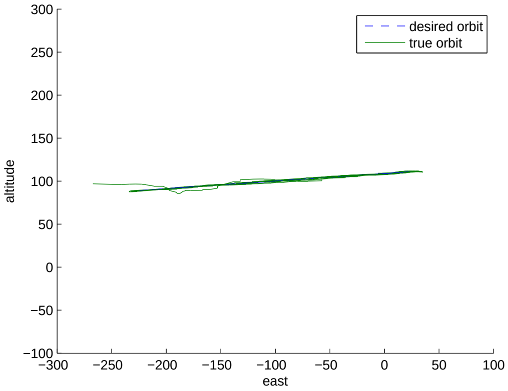


Figure 5.33: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 5°.

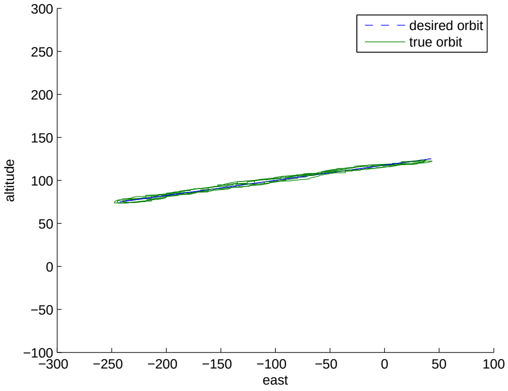


Figure 5.34: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 10°.

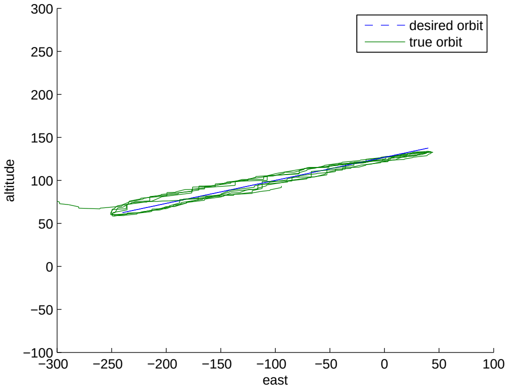


Figure 5.35: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 15°.

MAV to trail behind the desired altitude. Essentially, when the MAV was on a part of the orbit that was inclined, its altitude would be smaller than desired. A high gain caused the MAV to exceed the desired altitude. When the MAV was on a part of the orbit that was inclined, its altitude would be greater than desired.

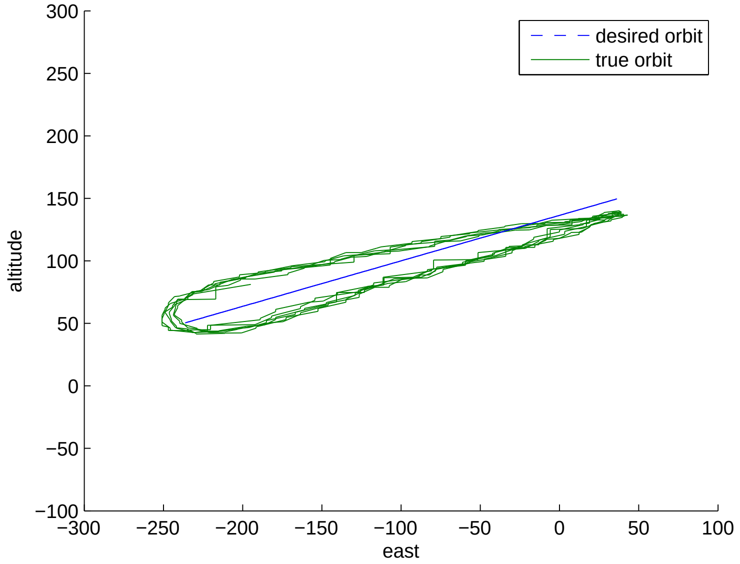


Figure 5.36: SITL elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 20°.

5.4.3 Lateral Control Flight Test Results

The aircraft used for the flight tests was a single prop zagi with a 60 inch wingspan shown in Figure 5.37. The UAV was equipped with a Kestrel autopilot described previously in this chapter. The flight tests took place in the presence of 1-2 m/s winds in the southwesterly direction.

Figures 5.38-5.41 show plots from the flight tests. The method of error removal used in these flight tests was the alternate orbit size methods. Although this method is not as robust as using a feed-forward term, it was necessary for several reasons. Initially it was attempted to send roll commands from the ground, which would allow for the use of the feed-forward term. The roll commands were computed on the ground station and then sent to the MAV, but attempting to control the low-level loops from the ground with the addition of the transmission delay caused the MAV to be unstable. The aircraft would oscillate over the desired path no matter how low the intensity of the vector field. For this reason, heading commands were given from the ground station instead of roll commands. This allowed the roll loop to happen onboard the aircraft, which made the low-level loop much faster. Since heading commands were given from the ground, adding a feed-forward roll term was not possible. In future work, firmware can be uploaded to the autopilot

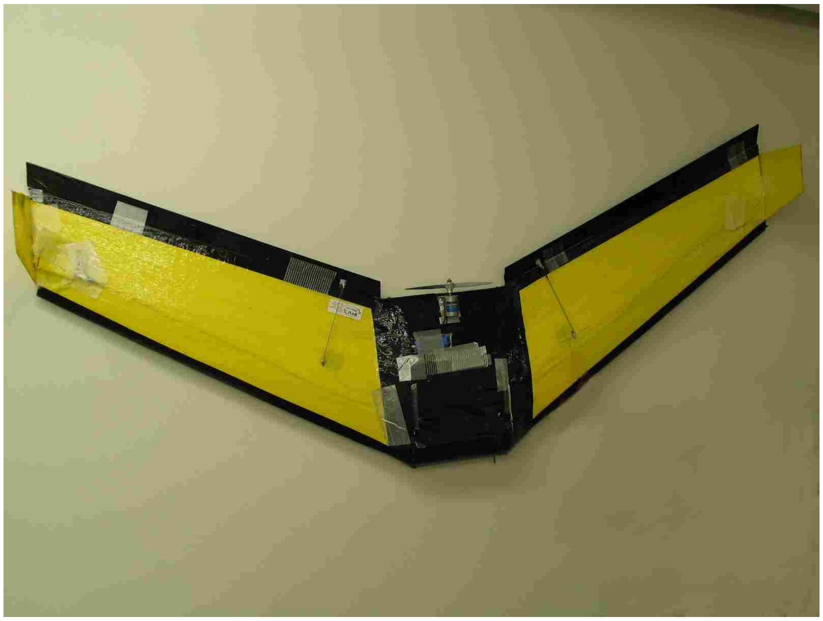


Figure 5.37: Zagi used in orbit tracking flight tests.

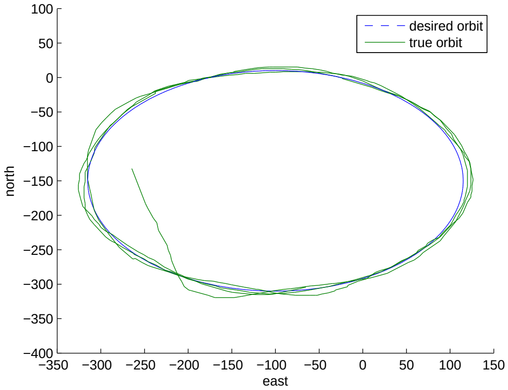


Figure 5.38: Flight test for elliptical orbit tracking with major axis of 215 m and minor axis of 160 m.

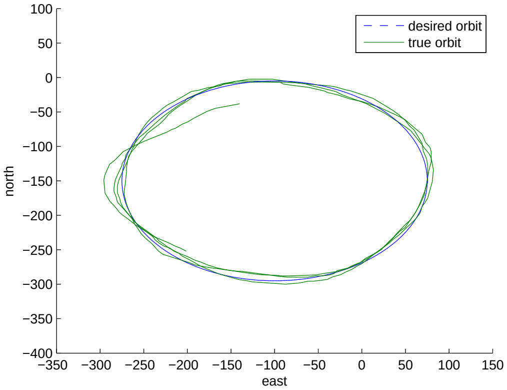


Figure 5.39: Flight test for elliptical orbit tracking with major axis of 175 m and minor axis of 145 m.

that would allow the orbit tracking algorithms to happen onboard. Doing this would permit the usage of the feed-forward roll term.

These plots show that the alternate orbit size method does effectively remove the steady state error in a live flight test scenario. As with the SITL simulations, the error is greater for the tracking of smaller orbits. The standard deviation of error for the largest orbit size is about 2 m, while the standard deviation of the error for the smallest orbit size is roughly 7 m. Although there

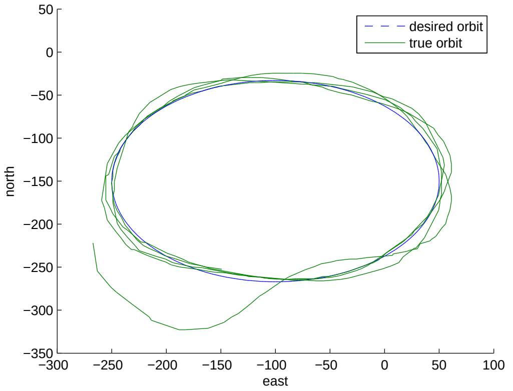


Figure 5.40: Flight test for elliptical orbit tracking with major axis of 150 m and minor axis of 115 m.

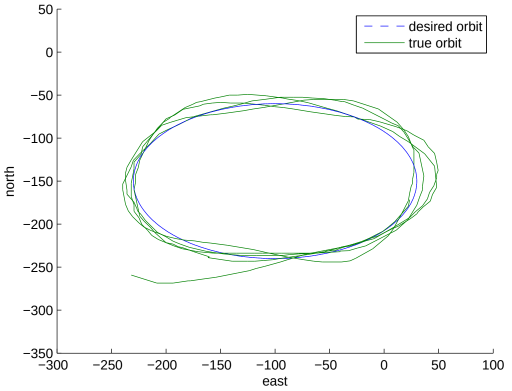


Figure 5.41: Flight test for elliptical orbit tracking with major axis of 130 m and minor axis of 90 m.

is error in the tracking, it is sufficient to get the MAV in the vicinity of the drogue so that a more precise vision-based algorithm can finalize the docking process. Additionally, the errors in the flight tests are larger than that of the SITL simulations, but that is expected since during a flight test, disturbances will affect the performance of the controller that were unmodeled in simulation.

Using this method, the only gain that needed to be tuned was the gain k that controls the intensity of the vector field as described in Equation (4.7). The value used was equivalent to that used in the SITL simulations, which demonstrates the high level of fidelity of that type of simulation. As with the SITL simulations, having a small gain caused a large steady state error, and having a large gain caused instability in the tracking.

5.4.4 Longitudinal Control Flight Test Results

The same aircraft was used for the longitudinal flight tests as for the lateral flight tests. The flight tests took place in the presence of 1-3 m/s winds in the northwesterly direction. Figures 5.42-5.45 show plots from the flight tests. Table 5.2 shows the standard deviation of the altitude tracking error for each out-of-plane rotation. The method used to remove the steady state error is the alternate altitude method. The feed forward flight path angle method again could not be used because the Virtual Cockpit does not accept flight path angle commands; rather, it only accepts pitch and altitude commands. Also, due to the transmission delay from the ground station to the

Table 5.2: Table of error for altitude tracking

	$\theta = 0°$	$\theta = 5°$	$\theta = 10°$	$\theta = 15°$
std dev of error (m)	.9	1.8	2.1	3.13

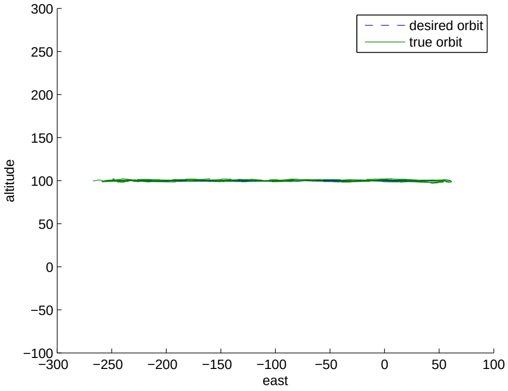


Figure 5.42: Flight test for elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 0°.

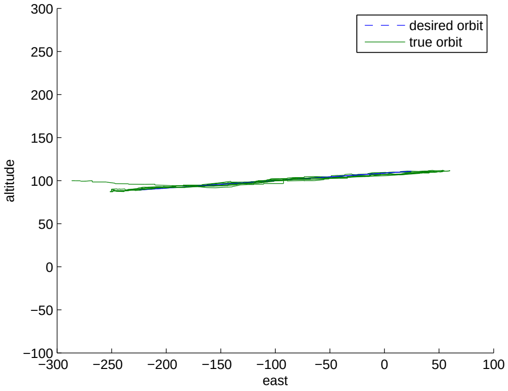


Figure 5.43: Flight test for elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 5°.

aircraft, trying to send a low-level command from the ground would likely result in instability as happened in the lateral control flight test. Giving altitude commands provided more stability since the low-level pitch loop was able to happen on board.

As with the SITL simulations, the functionality of the altitude tracking algorithms are demonstrated. Again, at rotations of 15° or more, the control loops start to saturate in pitch, so the altitude tracking increases its error. It was attempted to track an orbit with an incline of 20°, but the aircraft was unable to climb that steeply and stalled.

The only parameter that needed to be tuned is the gain k from Equation (4.16) that controls the amount that the altitude input is changed based on the altitude rate. The same gain was used in the flight tests as was used in the SITL simulation, further validating the fidelity of SITL simulations.

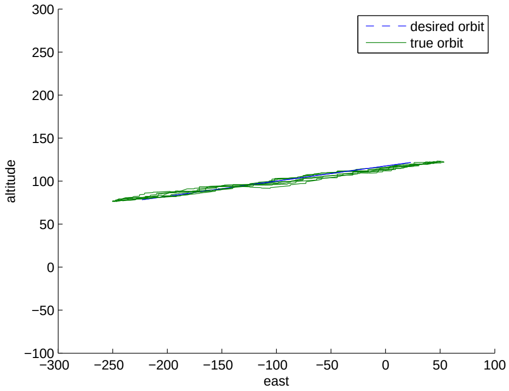


Figure 5.44: Flight test for elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 10°.

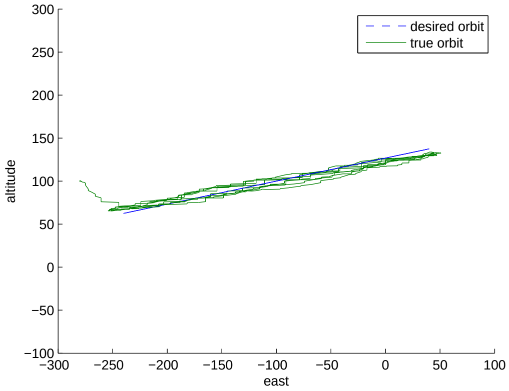


Figure 5.45: Flight test for elliptical orbit tracking with major axis of 125 m, minor axis of 100 m, and out-of-plane rotation of 15°.

5.5 Chapter Summary

This chapter describes the hardware and software used in flight tests and in software-in-the-loop simulations. The MATLAB simulink block diagram used in the flight tests and software-in-the-loop tests is also shown. Results are presented from the software-in-the-loop test for in-plane and out-of-plane orbit tracking. These simulations are shown to give high-fidelity results and provide a good place to tune gains in preparation for flight tests. Additionally, results are shown from the flight tests for in-plane and out-of-plane orbit tracking. These results are evaluated and demonstrate that the algorithms are effective in tracking arbitrarily rotated out-of-plane ellipses.

CHAPTER 6. CONCLUSION AND FUTURE WORK

6.1 Conclusions

In this thesis, essential methods to enable aerial recovery of micro air vehicles have been presented. The methods allow an autonomous MAV to estimate the elliptical orbit of a recovery drogue, and then track the drogue's orbit up until the point in which final docking approach occurs. In summary, contributions from this research to the area of aerial recovery are as follows:

- Methods were developed for estimating an arbitrarily rotated elliptical orbit. Previous methods for ellipse estimation were investigated and implemented into the aerial recovery scenario. These algorithms were expanded to be able to estimate an arbitrarily rotated ellipse. The methods for ellipse estimation have been demonstrated through simulation with added noise, and through testing on flight data to robustly and accurately estimate drogue trajectories.

- Parameter update methods were implemented that allow for fine tuning of the ellipse estimates with minimal computational expense as new GPS data points are received.

  - A recursive least squares (RLS) algorithm was applied to the aerial recovery scenario and tested on simulated data and actual flight data. Simulations and tests demonstrate convergence of the algorithm.

  - Kalman filter update methods were also explored and presented including Kalman filters with linear and nonlinear state constraints.

- Previous methods based on vector fields for circular orbit tracking were investigated and expanded to include tracking of elliptical orbits. These methods for orbit tracking were initially simulated in MATLAB. Software-in-the-loop (SITL) simulations were also developed and tested. The SITL simulations demonstrate that the control algorithms do allow for elliptical orbit tracking. Finally, these algorithms were validated through flight tests, which indicated that the algorithms can successfully enable a UAV to track an elliptical orbit.

- A method for longitudinal control was also developed that ueses a commanded altitude calculation with the addition of a feed-forward flight path angle term. Its capabilities were demonstrated in SITL simulations and flight tests.
- A journal article covering portions of this work was accepted for publication in the International Journal of Vehicle Autonomous Systems.
- A paper covering other portions of this work has been accepted for publication at the 2010 IEEE International Conference on Robotics and Automation.

6.2 Future Work

The aerial recovery project lends itself to many future research topics. It is a multifaceted area of research with many new concepts to discover and develop. The orbit estimation and orbit tracking techniques described in this thesis can be explored in greater detail, and research in the mothership-drogue dynamics can also be developed. Areas of research that may be of interest in the near future are listed as follows:

- As mentioned previously, the drogue orbit may be inclined due to wind and other disturbances. Future research can examine the possibility of altering the orbit of the mothership to make the drogue orbit more flat. An inclined mothership orbit could yield a flat drogue orbit in the presence of wind. Another method of making the drogue orbit more flat could be the addition of a winch that connects the cable to the mothership. This winch could control the length of the cable in order to flatten the drogue orbit.
- Suitable drogue platforms must be designed and constructed. Specific designs can be wind tunnel tested to measure drag and lift coefficients, and latching mechanisms for the capture of the MAV can be developed.
- Future research in orbit estimation can attempt to resolve the problem with the constraint matrix in the Kalman filter with nonlinear state constraints. Dividing the equations up in

a method similar to that posed by Halir [19] might be used. This addition would force the parameter updates to satisfy the ellipse-specific constraint.

- A sensitivity analysis could be performed to ascertain the robustness of the orbit estimation algorithms to noise.

- The research in orbit tracking also could be further refined. Optimal insertion of the MAV into the drogue orbit to avoid collision following the methods described by McLain and Beard [26] and Kingston and Beard [27] could be applied to the aerial recovery scenario. This addition would introduce a phase into the control law that describes the position of the drogue in the orbit. This work also extends to involve multiple MAVs, which may be applicable in future work involving multiple MAVs docking with the same drogue.

- Presently, the control laws demonstrate stability in the simulations and flight tests, but this work could be expanded by the development of Lyapunov based control laws. A control law using Lyapunov techniques could prove convergence and stability as was similarly demonstrated in [23] with circular orbits and waypoint following.

- To take advantage of the out-of-plane orbit estimation, a 3D guidance law could also be developed. Instead of decoupling the control into separate lateral and longitudinal algorithms, there would be a unified control law that uses the full 3D ellipse estimation instead of a projection of the ellipse onto the x-y plane.

- Firmware could be written that performs the calculations needed for the orbit tracking and loaded onto the autopilot for flight tests, instead of all the calculations happening on the ground station and being sent to the UAV. This would allow more processing to happen onboard and allow for better control of the UAV with less steady-state error.

- Vision-based proportional navigation [9] can also be tested in applications directly related to the aerial recovery situation.

- Larger scale flight tests can be performed in order to demonstrate all the aspects of an aerial MAV recovery in real-time.

- Flight tests with two UAVs can be performed. One UAV would follow an orbit as if it were the drogue, and the other UAV would estimate and track its orbit in real-time to demonstrate the approach phase of rendezvous.
- Another two-UAV test could be performed that demonstrates the proportional navigation algorithms. The leading UAV would again act as the drogue, and the trailing UAV would be equipped with a camera, which would track and finally bump into the leading UAV.
- A three-UAV test can be performed that demonstrates a scaled down full aerial recovery scenario. A large gas-powered UAV could drag a drogue large enough for a MAV to dock with. A MAV would estimate the drogue orbit, track the orbit, and finally use vision to finalize the docking process.

REFERENCES

[1] R. A. Skop and Y. Choo, “The configuration of a cable towed in a circular path,” *Journal of Aircraft*, vol. 8, pp. 856–862, 1971. 2

[2] J. J. Russel and W. J. Anderson, “Equilibrium and stability of a circularly towed cable subject to aerodynamic drag,” *Journal of Aircraft*, vol. 14, no. 7, pp. 680–686, 1977. 2

[3] R. M. Murray, “Trajectory generation for a towed cable flight control system,” in *13th Triennial World Congress of the International Federation of Automatic Control*, 1996, pp. 395–400. 2, 3

[4] P. Williams and P. Trivailo, “Dynamics of circularly towed cable systems, part 1: Optimal configurations and their stability,” *AIAA Journal of Guidance, Control, and Dynamics*, vol. 30, pp. 753–765, 2007. 2, 3

[5] ——, “Dynamics of a circularly towed cable systems, part 2: Transitional flight and deployment control,” *AIAA Journal of Guidance, Control, and Dynamics*, vol. 30, pp. 766–779, 2007. 2, 3

[6] L. Sun, R. W. Beard, M. B. Colton, and T. W. McLain, “Dynamics and control of cable-drogue system in aerial recovery of mirco air vehicles based on gauss’s principle,” in *Proceedings of the 2009 American Control Conference*, June 2009, pp. 4729–4734. 2, 3, 21

[7] M. Colton, S. Sun, D. Carlson, and R. Beard, “Multi-vehicle dynamics and control for aerial recovery of mcro air vehicles,” 2010, to Be Published in International Journal of Vehicle Autonomous Systems. 2, 21

[8] P. Williams and W. Ockels, “Dynamics of towed payload system using multiple fixed-wing aircraft,” *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 6, pp. 1766–1780, November-December 2009. 3

[9] R. W. Beard, J. W. Curtis, M. Eilders, J. Evers, and J. R. Cloutier, “Vision aided proportional navigation for micro air vehicles,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, August 2007. 3, 57

[10] R. W. Beard and T. W. McLain, “Guidance and control of autonomous fixed wing air vehicles,” unpublished. 5, 8

[11] N. Johnson, “Vision-assisted control of a hovering air vehicle in an indoor setting,” Master’s thesis, Brigham Young University, August 2008. 5, 8

[12] H. Baruh, *Analytical Dynamics*. WCB/McGraw-Hill, 1999. 6, 21

[13] R. R. Bate, D. D. Mueller, and J. E. White, *Fundamentals of Astrodynamics*. Dover Publications, Inc., 1971. 11

[14] T. K. Moon and W. C. Sterling, *Mathematical Methods and Algorithms*. Prentice Hall, 2000. 12, 18, 23, 64, 66

[15] F. L. Bookstein, "Fitting conic sections to scattered data," *Computer Graphics and Image Processing*, vol. 9, pp. 56–71, 1979. 14, 18, 63

[16] P. D. Sampson, "Fitting conic sections to very scattered data: An iterative refinement of the bookstein algorithm," *Computer Graphics and Image Processing*, vol. 18, pp. 97–108, 1982. 15

[17] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions: Pattern Analysis and Machine Intelligence*, vol. 21, pp. 476–480, 1999. 15

[18] W. Gander, "Least squares with a quadratic constraint," *Numerische Mathematik*, vol. 36, pp. 291–307, 1981. 16

[19] R. Halir and J. Flusser, "Numerically stable least squares fitting of ellipses," in *Proceedings of the 6th International Conference in Central Europe on Computer Graphics and Visualization*, 1998. 17, 57

[20] O. Nelles, *Nonlinear System Identification*. Springer Berlin, 2001. 18

[21] C. Yang and E. Blasck, "Kalman filtering with nonlinear state constraints," in *2006 9th International Conference on Information Fusion*, July 2006. 19, 63, 65

[22] D. B. Barber, S. R. Griffiths, T. W. McLain, and R. W. Beard, "Autonomous landing of miniature aerial vehicles," *AIAA Journal of Aerospace Computing, Information, and Communication*, vol. 4, pp. 70–84, May 2007. 27, 30

[23] D. R. Nelson, D. B. Barber, and T. W. McLain, "Vector field path following for miniature air vehicles," *IEEE Transactions on Robotics*, vol. 23, pp. 519–529, June 2007. 27, 57

[24] T. Titensor, *Kestrel User Guide*, Procerus, 2008. 35

[25] ——, *Virtual Cockpit 2.5*, Procerus, 2008. 36

[26] T. W. McLain and R. W. Beard, "Coordination variables, coordination functions, and cooperative-timing missions," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 27, no. 5, pp. 150–161, July 2004. 57

[27] D. B. Kingston and R. W. Beard, *Advances in Cooperative Control and Optomization*. Springer Berlin, 2007, ch. UAV Splay State Configuration for Moving Targets in Wind. 57

[28] D. Simon and T. L. Chia, "Kalman filtering with state equality constraints," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 1, pp. 128–136, January 2002. 64

APPENDIX A. MOMENT OF INERTIA TERMS

The inertia terms in Equations (2.3)-(2.6) are defined as follows:

$$\begin{aligned}
\mathbf{J} &= \begin{bmatrix} \int \left(y^2+z^2\right) dm & -\int xy\, dm & -\int xz\, dm \\ -\int xy\, dm & \int \left(x^2+z^2\right) dm & -\int yz\, dm \\ -\int xz\, dm & -\int yz\, dm & \int \left(x^2+y\right) dm \end{bmatrix} \\
&\triangleq \begin{bmatrix} J_x & -J_{xy} & -J_{xz} \\ -J_{xy} & J_y & -J_{yz} \\ -J_{xz} & -J_{yz} & J_z \end{bmatrix}
\end{aligned}$$

$$\begin{aligned}
\Gamma &= J_x J_z - J_{xz}^2 \\
\Gamma_1 &= \tfrac{J_{xz}\left(J_x - J_y + J_z\right)}{\Gamma} \\
\Gamma_2 &= \tfrac{J_z\left(J_z - J_y\right) + J_{xz}^2}{\Gamma} \\
\Gamma_3 &= \tfrac{J_z}{\Gamma} \\
\Gamma_4 &= \tfrac{J_{xz}}{\Gamma} \qquad . \\
\Gamma_5 &= \tfrac{J_z - J_x}{J_y} \\
\Gamma_6 &= \tfrac{J_{xz}}{J_y} \\
\Gamma_7 &= \tfrac{\left(J_x - J_y\right)J_x + J_{xz}^2}{\Gamma} \\
\Gamma_8 &= \tfrac{J_x}{\Gamma}
\end{aligned}$$

APPENDIX B. KALMAN FILTER PARAMETER UPDATE METHOD

The RLS algorithm has the same weakness as the Bookstein method [15]: the solution is not guaranteed to be an ellipse. The initial estimate is constrained to be an ellipse using Halir's method, but once the RLS algorithm takes over, the solution could theoretically diverge and become some other conic such as a hyperbola. By using a Kalman filter with nonlinear state constraints, the solution can be projected onto the space where the ellipse-specific constraint is satisfied [21]. The ellipse can be considered a system with its parameters being the states as described by

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + w_k \\ y_k &= Cx_k + v_k \end{aligned} \quad , \tag{B.1}$$

where x is the states (the ellipse parameters), y is the output of the system, u is the input to the system (the GPS data), A, B, and C are matrices that describe the system, and w and v simulate noise and other unknown disturbances.

The state and covariance matrices are updated using the the following Kalman filter equations:

$$\begin{aligned} \hat{x}_{k+1} &= A\hat{x}_k + Bu_k + K_k\left(y_k - C\hat{x}_k\right) \\ K_k &= AP_kC^T\left(CP_kC^T + R\right)^{-1} \\ P_{k+1} &= \left(AP_k - K_kCP_k\right)A^T + Q \end{aligned} \quad , \tag{B.2}$$

where K is a gain that help minimize the covariance error (P), R is the noise covariance of the measurement, and Q is the noise covariance of the model of the system. Equations (B.1) and (B.2) describe an unconstrained Kalman filter.

B.1 Kalman Filtering with Linear State Constraints

Unconstrained Kalman filtering in the case of ellipse estimation essentially reduces to the RLS algorithm since there is no time update ($A = I$), and there is no input ($u = [0]$). Subsequently, using an unconstrained Kalman filter could also diverge from an elliptical solution. Simon and Chia describe a method for projecting an unconstrained Kalman filter solution onto a linear state constraint [28].

First, the linear constraint D is added. D is an $s \times n$ matrix, where n is the number of states, and s is the number of constraints and $s \leq n$. This constraint is given by

$$Dx_k = d. \tag{B.3}$$

An unconstrained estimate is calculated using Equation B.2. That solution is projected onto the space that satisfies the constraint, which is accomplished using

$$\begin{array}{c} \breve{x} = \arg\min_{x \varepsilon S} (x - \hat{x})^T W (x - \hat{x}) \\ S = \{x : Dx = d\} \end{array} . \tag{B.4}$$

This optimization problem is again solved using the method of Lagrange Multipliers [14]. The Lagrangian is first constructed according to

$$J(x, \lambda) = (x - \hat{x})^T W (x - \hat{x}) + 2\lambda^T (Dx - d). \tag{B.5}$$

Taking partial derivatives gives the necessary conditions

$$\begin{array}{c} \frac{\partial J}{\partial x} = 0 \Rightarrow W(x - \hat{x}) + D^T \lambda = 0 \\ \frac{\partial J}{\partial \lambda} = 0 \Rightarrow Dx - d = 0 \end{array} , \tag{B.6}$$

which are needed for satisfying (B.4).

The solution that satisfies the conditions are found from

$$\begin{array}{c} \lambda = \left(DW^{-1}D^T\right)^{-1} (D\hat{x} - d) \\ \breve{x} = \hat{x} - W^{-1}D^T \left(DW^{-1}D^T\right)^{-1} (D\hat{x} - d) \end{array} , \tag{B.7}$$

where $\breve{x}$ contains the parameters that now satisfy the ellipse constraint.

B.2 Kalman Filtering with Nonlinear State Constraints

The ellipse-specific constraint is quadratic, so using a linear approximation could still yield non-ellipse results. Yang and Blasch expand the linear method described by Simon and Chia so that it incorporates quadratic state constraints [21].

First, a quadratic state constraint is given by

$$f(x) = \begin{bmatrix} x^T & 1 \end{bmatrix} \begin{bmatrix} M & m \\ m^T & m_0 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = x^T M x + m^T x + x^T m + m_0 = 0, \tag{B.8}$$

which in the aerial recovery case, can be stated as $\mathbf{a}^T \mathbf{C} \mathbf{a} - 1 = 0$, as shown in Section 3.3.1.

As in Section B.1, the solution that satisfies the constraint can be posed as a constrained least-squares optimization problem of the form

$$\begin{array}{c} \hat{x} = \arg\min\limits_{x} (z - Hx)^T (z - Hx) \\ \text{subject to } f(x) = 0 \end{array}, \tag{B.9}$$

where $W = H^T H$ and $z = H\hat{x}$. This substitution of variables allows for a more general formulation of the problem. This problem can again be solved using the method of Lagrange multipliers. The Lagrangian is formed as

$$J(x, \lambda) = (z - Hx)^T (z - Hx) + \lambda f(x). \tag{B.10}$$

Taking partial derivatives gives the necessary conditions

$$\frac{\partial J}{\partial x} = -H^T z + \lambda m + \left(H^T H + \lambda M\right) x = 0 \tag{B.11}$$

$$\frac{\partial J}{\partial \lambda} = x^T M x + m^T x + x^T m + m_0 = 0. \tag{B.12}$$

Equation (B.11) can be solved for x as

$$x = \left(H^T H + \lambda M\right)^{-1} \left(H^T z - \lambda m\right). \tag{B.13}$$

At this point, the solution can be simplified in the case of ellipse estimation because $m = 0$ in the constraint equation. Equation B.13 can subsequently be simplified as

$$\check{x} = (W + \lambda M)^{-1} (W\hat{x}), \tag{B.14}$$

where $\hat{x}$ is the updated parameters of the conic after running the input through a standard Kalman filter, and $\check{x}$ is the parameters of the conic projected onto the ellipse-specific constraint.

We then decompose W and M as

$$W = G^T G \text{ and } M = L^T L. \tag{B.15}$$

SVD [14] can then be performed on the matrix LG^{-1}as

$$LG^{-1} = U\Sigma V^T. \tag{B.16}$$

A new vector e is introduced which is defined as

$$e = V^T \left(G^T\right)^{-1} H^T z. \tag{B.17}$$

The second necessary condition can then be rewritten as a function of λ as

$$f(\lambda) = \sum_i \frac{e_i \sigma_i^2}{\left(1 + \lambda \sigma_i^2\right)^2} + m_0 \tag{B.18}$$

$$\dot{f}(\lambda) - 2\sum_i \frac{e_i^2 \sigma_i^4}{\left(1 + \lambda \sigma_i^2\right)^3}. \tag{B.19}$$

These equations can be solved for λ iteratively using Newton's method. Then λ can be substituted into Equation (B.14).

In summary, new GPS data is received and used as inputs to a standard Kalman filter to get $\hat{x}$. Then the Lagrange multipliers (λ) are solved for using Equations (B.18) and (B.19). Once λ and $\hat{x}$ are calculated, Equation (B.14) can then be computed to give $\check{x}$, which contains the updated conic parameters that satisfy the quadratic, ellipse-specific constraint.