Fault Detection, Isolation, and Recovery for Autonomous Parafoils

by

Matthew Robert Stoeckle

B.S., University of Maryland, College Park (2012)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

Author .....
Department of Aeronautics and Astronautics
May 22, 2014

Certified by .....
Jonathan P. How
Richard C. Maclaurin Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .....
Louis S. Breger
Member of the Technical Staff, Draper Laboratory
Thesis Supervisor

Accepted by .....
Paulo C. Lozano
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Fault Detection, Isolation, and Recovery for Autonomous Parafoils

by

Matthew Robert Stoeckle

Submitted to the Department of Aeronautics and Astronautics
on May 22, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Autonomous precision airdrop systems are widely used to deliver supplies to remote locations. This aerial delivery method provides a safety and logistical advantage over traditional ground- or helicopter-based payload transportation methods. The occurrence of a fault during a flight can severely degrade vehicle performance, effectively nullifying the value of the guided system, or worse. Quickly detecting and identifying faults enables the choice of an appropriate recovery strategy, potentially mitigating the consequences of an out-of-control vehicle and recovering performance. This thesis presents a fault detection, isolation, and recovery (FDIR) method for an autonomous parafoil system. The detection and isolation processes use residual signals generated from observers and other system models. Statistical methods are applied to evaluate these residuals and determine whether a fault has occurred, given a priori knowledge of how the system behaves in the presence of faults. This work develops fault recovery strategies that are designed to mitigate the effects of several common faults and allow for a successful mission even with severe loss of control authority. An extensive, high-fidelity, Monte Carlo simulation study is used to assess the effectiveness of FDIR, including the probability of correctly isolating a fault as well as the target miss distance improvement resulting from the implementation of fault recovery strategies. The integrated FDIR method demonstrates a very high percentage of successful isolation as well as a substantial decrease in miss distance for cases in which a common fault occurs. Flight test results consistent with simulations show successful detection and isolation of faults as well as implementation of recovery strategies that result in miss distances comparable to those from healthy flights.

Thesis Supervisor: Jonathan P. How
Title: Richard C. Maclaurin Professor of Aeronautics and Astronautics

Thesis Supervisor: Louis S. Breger
Title: Member of the Technical Staff, Draper Laboratory

Acknowledgments

I would not have been able to complete this thesis without the contributions and joint effort of a large group of people who have supported me during the last two years. First, I would like to thank my advisors, Professor Jonathan How from MIT and Dr. Louis Breger from Draper Laboratory. Their guidance and technical input kept me focused and helped me to develop my research. I am grateful to both MIT and Draper Laboratory for funding my graduate education.

My interest in engineering in general and controls in particular was nurtured by many great teachers I've had over the years, notably Jim Hessler from Lenape High School in New Jersey, and Dr. Robert Sanner from the University of Maryland. Their passion for physics and controls, respectively, and the skill with which they taught these subjects, made for important milestones in my educational career.

I would like to thank my colleagues at Draper, especially Amer Fejzic, for helping to acclimate me to the Airdrop program and for allowing me to work on such an interesting and challenging task. I was lucky enough to share an office with two other graduate students, Jared Rize and Celena Dopart, who helped to keep things light at Draper and collaborated with me on work at MIT.

I also owe a great deal of thanks to Ellie Farr, whose constant love and support, even when she was busy working and applying to medical school, helped make this process much easier. Finally, and most importantly, I would like to thank my parents, without whose guidance and encouragement I would not be the person I am today.

Contents

1 Introduction **17**

1.1 Problem and Objectives 17

1.2 Literature Review 19

1.3 Content Overview 22

2 System Overview **25**

2.1 Parafoil System 25

2.2 Parafoil Dynamics 28

2.2.1 Full Dynamics and Nonlinear Simulator 28

2.2.2 Linearized Lateral Dynamics 28

2.3 Nominal Guidance Strategy 31

2.3.1 Homing 31

2.3.2 Loiter 31

2.3.3 Final Approach 32

2.4 System Faults 33

2.4.1 Criss-Crossed Lines 33

2.4.2 Broken Line 35

2.4.3 Stuck Motor 36

2.4.4 Other Faults 36

2.5 Effect of Faults on System Performance 37

3 Detection **39**

3.1 Residual Generation 40

3.1.1 Heading Rate Residual . . . 40
3.1.2 Motor Residual . . . 42
3.2 Residual Evaluation . . . 44
3.2.1 Heading Rate Residual . . . 45
3.2.2 Motor Residual . . . 48
3.3 Detection Results . . . 50

4 Isolation **55**
4.1 Evaluation of Motor Residuals . . . 56
4.2 Fault-Specific Observers . . . 56
4.2.1 Residual Generation . . . 56
4.2.2 Residual Evaluation . . . 59
4.3 Preventing False Isolation . . . 65
4.4 Full FDI Implementation . . . 66
4.4.1 Overall Logic . . . 66
4.4.2 Results . . . 69

5 Recovery **75**
5.1 Criss-Crossed Lines . . . 75
5.2 Broken Line . . . 76
5.2.1 Track and Loop Guidance . . . 77
5.3 Stuck Motor . . . 82
5.4 Preventing False Isolation . . . 86
5.4.1 Effect on Performance . . . 86
5.4.2 Post-Isolation Analysis . . . 88
5.5 Results . . . 93
5.5.1 Criss-Crossed Lines . . . 93
5.5.2 Broken Line . . . 94
5.5.3 Stuck Motor . . . 94
5.5.4 Overall Simulation Results . . . 96

6 Flight Test Results **99**

6.1 Simulated Broken Left Line 100

6.2 Simulated Criss-Crossed Lines 101

6.3 Stuck Left Motor Small 101

6.4 Stuck Left Motor Large 103

6.5 Physical Criss-Crossed Lines 104

6.6 Conclusions 105

7 Conclusions and Future Work **107**

7.1 Conclusions 107

7.2 Future Work 108

A Determination of Observer Gain **109**

B Exponential Low-Pass Filtering **113**

C Track and Loop Guidance Parameter Selection **117**

List of Figures

2-1 Parafoil and Payload System 26
2-2 Parafoil AGU and Control Lines 27
2-3 East-North Parafoil Coordinate Frame 27
2-4 Typical System Groundtrack: Nominal Guidance Strategy 32
2-5 Parafoil Fault Hierarchy 33
2-6 East-North Groundtrack: Criss-Crossed Lines Fault 34
2-7 East-North Groundtrack: Broken Line Fault 35
2-8 East-North Groundtrack: Stuck Motor Fault 36
2-9 Miss Distance CDFs with 95% Confidence Bounds: Nominal Guidance Strategy 37

3-1 Heading Rate Residual Generation Block Diagram 41
3-2 Motor Residual Generation Block Diagram 43
3-3 CDFs for Healthy and Fault Data: Heading Rate Residual 46
3-4 CDFs for Healthy and Fault Data: Motor Residual 49
3-5 Missed Detection and False Alarm Rates from Simulation 51
3-6 Fault Detection Example: Broken Right Line 52
3-7 Fault Detection Example: Stuck Left Motor 52

4-1 Fault-Specific Observer Residual Generation Block Diagram 57
4-2 PDFs of Fault-Specific Observer Residuals: Criss-Crossed Lines Fault Present 60
4-3 PDFs of Fault-Specific Observer Residuals: Broken Line Fault Present 61

4-4 PDFs of Fault-Specific Observer Residuals: Criss-Crossed Lines Fault Present with Scaling Factor Implemented 62
4-5 PDFs of Fault-Specific Observer Residuals: Broken Line Fault Present with Scaling Factor Implemented 63
4-6 CDFs of Fault-Specific Observer Residuals 66
4-7 Fault Detection Logic 68
4-8 Fault Isolation Logic 68
4-9 Isolation Lag Time Statistics for Various Flight Modes 70
4-10 Altitude Remaining in Flight After Successful Isolation 71
4-11 Heading Rate and Motor Residuals: Stuck Right Motor FDI 72
4-12 Heading Rate and Motor Residuals: Broken Left Line FDI 73
4-13 Fault Specific Observer Residuals: Broken Left Line FDI 73

5-1 Track and Loop Parameters in Parafoil Coordinate Frame 79
5-2 Track and Loop Guidance Logic for Broken Left Line Recovery 81
5-3 Normalized Miss Distances for Different Stuck Motor Positions: Control Modification Only 84
5-4 Mean Normalized Miss Distances for Stuck Motor Value Bins 85
5-5 Normalized Miss Distances for Different Stuck Motor Positions: Full Recovery Implemented 86
5-6 Miss Distance CDFs with 95% Confidence Interval: Broken Line False Isolation 87
5-7 Miss Distance CDFs with 95% Confidence Interval: Criss-Crossed Lines False Isolation 87
5-8 Post-Isolation Broken Line Residual Data: Broken Line Isolated . . . 89
5-9 Post-Isolation Criss-Crossed Lines Residual Data: Criss-Crossed Lines Fault Isolated 90
5-10 Fault Specific Observer Residuals: Broken Left Line with Criss-Crossed Lines False Isolation 91

5-11 Fault Specific Observer Residuals: Criss-Crossed Lines with Broken Left Line False Isolation 92
5-12 East-North Groundtrack of Simulation with Criss-Crossed Lines Fault: Recovery Implemented 94
5-13 East-North Groundtrack of Simulation with Broken Left Line: Recovery Implemented 95
5-14 East-North Groundtrack of Simulation with Stuck Right Motor at Full Deflection: Recovery Implemented 96
5-15 Miss Distance CDFs with 95% Confidence Bounds: Recovery Strategies Implemented 97
5-16 Miss Distance CDFs with 95% Confidence Bounds: Overall Results . 98

6-1 East-North Groundtrack of Flight with Simulated Broken Left Line Fault 100
6-2 East-North Groundtrack of Flight with Simulated Criss-Crossed Lines Fault 101
6-3 East-North Groundtrack of Flight with Stuck Left Motor at Small Value 102
6-4 East-North Groundtrack of Flight with Stuck Left Motor at Large Value 103
6-5 East-North Groundtrack of Flight with Criss-Crossed Lines Fault . . 104

B-1 Bode Plot of Parafoil Lateral Dynamics Model 114
B-2 Bode Plot of Motor Model 114

List of Tables

3.1 Optimized Thresholds for Various FOMs: Heading Rate Residual . . 47

3.2 Optimized Thresholds for Various FOMs: Motor Residual 49

3.3 Fault Detection Test with Various Threshold Choices 50

4.1 Optimized Scaling Values for Various FOMs 64

4.2 FDI Parameters 67

4.3 FDI Results 69

5.1 Track and Loop Guidance Parameters 81

5.2 Mean Normalized Miss Distances for Various Fault Cases with and without Recovery Strategy 98

C.1 Track and Loop Parameter Iteration 119

Chapter 1

Introduction

Autonomous precision airdrop is used to deliver payloads to areas that would be dangerous or difficult to reach through more conventional means. Missions for guided parafoils include military resupply of troops and humanitarian efforts [19] [20]. The goal of each mission is to land the payload as close as possible to the target while minimizing ground speed at impact [5] [21]. Flight testing has shown that a variety of faults can occur [11] [44]. These faults increase target miss distance and landing speed, potentially rendering payloads unusable. In addition, the possibility of an in-flight fault and the resulting behavior could preclude delivering supplies to more densely populated areas where an out-of-control vehicle could pose a danger to persons or property. Detecting, isolating, and responding to faults can improve performance and expand the space of missions available for guided parafoils. This work designs and implements a fault detection, isolation, and recovery (FDIR) strategy that is effective in the unique conditions under which the parafoil operates.

1.1 Problem and Objectives

To my knowledge, no FDIR method exists for the parafoil and payload system. Consequently, all faults that occur in-flight go unchecked, generally resulting in large miss distances and mission failures. Flight testing of the guided airdrop system has revealed that there are several discrete faults that occur more commonly than others.

These faults cause either large changes in system characteristics or severely modify or limit the actuation ability of the system. Without an effective fault detection and isolation (FDI) method, any fault occurring will not be identified online. Therefore, implementing a recovery strategy to mitigate the effects of the fault is difficult. Even with an effective FDI method, the nominal guidance strategy used on many airdrop systems is not well-suited to control the parafoil in the presence of faults. The dramatic changes resulting from these faults affect the performance characteristics of the system, and the assumptions made in the design of the nominal guidance strategy are no longer valid in the presence of many faults.

The objective of this work is to develop and test, both in simulation and in flight, an FDIR method that can identify several common parafoil system faults, and implements new guidance strategies designed to recover from the effects of these faults and result in a mission success. This method must not only be effective, but it should minimize the impact on healthy flights. The implementation of the FDIR algorithm into flight software should have a negligible impact on overall software performance. Moreover, the rate of false fault alarms during a healthy flight, which may result in the implementation of a guidance strategy that actually makes system performance worse, should be minimized as much as possible. A successful FDIR method will detect and isolate a fault correctly and quickly, such that an appropriate recovery strategy can be implemented, which allows the system to perform the mission as desired even in the presence of a fault.

Many existing FDI methods, described in Section 1.2, require diagnostic actions or other disturbances to the nominal flight scenario. Others require a high computational load. This work utilizes existing passive, observer-based methods in a unique application of FDI to the parafoil and payload system, and develops new fault recovery strategies designed to mitigate the effects of several faults that occur on guided airdrop systems.

1.2 Literature Review

Online systems for FDI fall into two categories: those that exploit hardware redundancy and those that rely on analytical redundancy [22]. Systems with a large number of sensors, actuators, and measurements employ hardware redundancy for FDI or system health management [13] [14]. The parafoil has a small number of sensors, and so analytical redundancy methods are used.

Isermann and Ballé [25] define FDI terminology. A fault is defined as an unpermitted deviation of at least one characteristic property or parameter of the system from the acceptable/usual/standard condition. Fault detection is the determination of the faults present in a system and the time of detection. Fault isolation is the determination of the kind, location, and time of detection of a fault. The process of isolation follows that of detection.

For FDI to be effective, 1) the effects of faults must be distinguishable from the effects of unknown inputs including modeling errors, disturbances, and measurement uncertainty, and 2) faults must be distinguishable from each other [16]. This is typically accomplished by considering a residual signal [22]. The residual signal chosen has approximately zero mean when no fault is present and a nonzero mean when a fault has occurred. In this context, the residual signal is the difference between a measurable system output and the corresponding expected output. After the residual has been generated, it is evaluated. The goal of the evaluation process is to determine whether a fault alarm should be raised based on the properties of the residual signal.

A large group of FDI methods are classified as observer-based [15]. These methods use an observer of the nominal system to generate the expected system output. This output is compared to measurements from the actual system to generate the residual signal. Though a simulation of the system with no feedback can also be used to generate the residual signal, an observer is chosen to make the residual generation process robust to differences in initial conditions. The initial states of the system are not precisely known at the beginning of a flight, so the use of an observer allows for the convergence of the observer states to the true states of the system when there is

no fault present.

A common method of residual generation that uses observers is called the fault detection filter (FDF). This method generates a residual signal that is projected onto subspaces associated with various faults, so that detection and isolation are both possible [4] [27]. See Douglas and Speyer [12] for a robust implementation of the FDF. For isolation, the FDF requires that each fault under consideration acts on the system in a known, unique way. This is not the case for the parafoil system; many faults act on the actuators of the system and are not distinguishable from each other. Another observer-based FDI approach uses a technique called eigenstructure assignment. This approach is used to decouple effects of disturbances from those of faults by nulling the transfer function from the disturbances to the residual signal [36]. A weighting matrix that is used to assign eigenvectors to the closed-loop observer of the system accomplishes this task. In order to construct this weighting matrix, however, there must be more independent outputs of the system than independent disturbances [36]. The parafoil system is single-output, so eigenstructure assignment is not possible.

Other analytical methods for FDI include parity relations approaches, Kalman filter-based approaches, and system identification methods. Parity relations methods generate and transform residual signals in one process so that desired properties are met [17] [35]. This is similar to the FDF in that the transformation is designed to increase observability of individual faults. As with the FDF, parity relations are not suited to the FDI problem for the parafoil system because of the similar nature of each of the faults considered in this work. Kalman filter-based approaches perform tests for whiteness and covariance on the residual signals to detect anomalies [18] [31] [32]. This process can be computationally intensive. The parafoil system on which FDIR is performed in this work requires that the algorithm have a minimal computational load on the system, which is already running several expensive algorithms on the flight software. System identification procedures for fault detection use online parameter estimation to detect faults as changes in system parameters [24] [41]. Ward et al. [46] developed a system identification method for the parafoil system. While these

methods may work for the parafoil FDI problem, the objective of this thesis is to develop an FDIR method that does not require the use of diagnostic actions. The algorithm should be able to run without affecting the flight of the system until a fault has been successfully detected and isolated.

The FDI method presented in this thesis is observer-based, but takes a different approach than the FDF or parity relations method. Many existing observer-based methods incorporate isolation into the detection process by exploiting the system property that each fault under consideration is distinguishable from all other faults [16]. However, this is not the case for many faults that occur on the parafoil system. As a result, the detection and isolation processes are separate for this work.

For detection, a residual signal is generated using observer-based methods. This residual is evaluated using hypothesis testing. Several residual evaluation techniques, such as the sequential probability ratio test (SPRT) [45] and cumulative sum (CUSUM) [34] use a moving average as well as conditional probabilities to determine if a change to the system has occurred. Since no explicit knowledge of the conditional probabilities of the residual signals exists, a simpler hypothesis testing approach is used to evaluate residuals in this work. If the magnitude of the residual signal crosses above a predetermined detection threshold, the null hypothesis is rejected and a fault is declared [16]. This threshold is designed based on empirical results to limit the number of false alarms and missed detections resulting from this process. Sargent et al. [40] use hypothesis testing with thresholds for FDI on the Orbital Cygnus vehicle. Rossi [37] uses hypothesis testing for health management of spacecraft.

If a fault is declared, isolation is performed. In this thesis, two different isolation methods are used concurrently. The first method uses residual signals from actuator data. If evaluation of these signals indicate that an actuator fault has occurred, isolation is complete. Non-actuator faults are isolated using a bank of fault-specific observers. The purpose of these observers is to determine when the system exhibits characteristics of a particular fault [48]. Evaluating residual signals from these observers indicates if a specific non-actuator fault is present. Successful isolation will result in the declaration of a fault on one of the actuators or the declaration of a

particular non-actuator fault.

A recovery strategy that uses the results of FDI to allow for nominal, or close to nominal, performance can be beneficial, and allows for a higher mission success rate as well as mitigation of potentially dangerous aspects of an out-of-control system that are often the consequence of a severe fault. The nominal guidance strategy used by precision airdrop systems is unable to deal with the effects of many common faults that occur. An adaptive control method designed to be robust to faults that cause changes in system characteristics, such as canopy tears, has been developed and tested successfully in simulation [11]. The faults considered in this work fundamentally change the control authority of the system rather than change the system properties. The adaptive control method presented in Culpepper et al. [11] cannot recover from substantial loss of control authority. A multiple-model approach in which guidance and control modifications are implemented to mitigate the specific effects of each of the faults considered in this work is used for fault recovery for the parafoil system.

1.3 Content Overview

Chapter 2 provides an overview of the parafoil and payload system, including dynamics and the nominal guidance strategy, as well as a description of the faults considered in this work. Chapter 3 describes the detection method, which involves residual generation and evaluation. Results are provided that analyze the success of this method. Chapter 4 describes the isolation method, including the use of fault-specific observers as well as the integration of this method with the detection method developed in Chapter 3. Chapter 5 describes modifications to the nominal guidance strategy used to recover from the faults considered in this work. These methods are integrated with the FDI method developed in Chapters 3 and 4 and tested and evaluated extensively in simulation. Chapter 6 presents results from five flight tests during which FDIR is tested in the presence of a fault. Chapter 7 concludes the thesis and provides suggestions for future work.

This thesis presents the first FDIR method for the parafoil and payload system.

This method is implemented into the flight software of the system, and does not affect a flight unless a successful isolation occurs. The implementation of this method allows for a high rate of detection and isolation of several common system faults, as well as recovery from those faults resulting in miss distances comparable to those observed during healthy flights.

Chapter 2

System Overview

2.1 Parafoil System

The parafoil and payload system consists of three main components: canopy, airborne guidance unit (AGU), and payload. The system is shown in Figure 2-1. The canopy has an airfoil cross-section, and generates lift for the system. There are two control lines, each of which attach to either the left or right trailing edge of the canopy and wrap around one of the two motors on the AGU. Figure 2-2 shows a close-up view of the AGU and control line attachments. Actuating a motor pulls the control line, which deflects the trailing edge of the canopy, inducing a turn in the corresponding direction (e.g., deflecting left motor will yield a left turn). The deflection of each motor and corresponding control line will be indicated in this thesis as a percentage of the maximum deflection (e.g., deflection of 0.5 indicates that the line is pulled in to half of the maximum amount). This motor deflection is measured by an encoder on each motor of the AGU. These measurements will be used to detect the presence of actuator faults. The payload is attached under the AGU and contributes to most of the system weight. The goal of the system is to precisely deliver this payload to a desired location.

In flight, performance is measured by how well the parafoil tracks the heading rate command given by guidance. Heading rate is defined as the rate at which the airspeed vector of the parafoil rotates with respect to the inertial North axis. Figure 2-3 shows

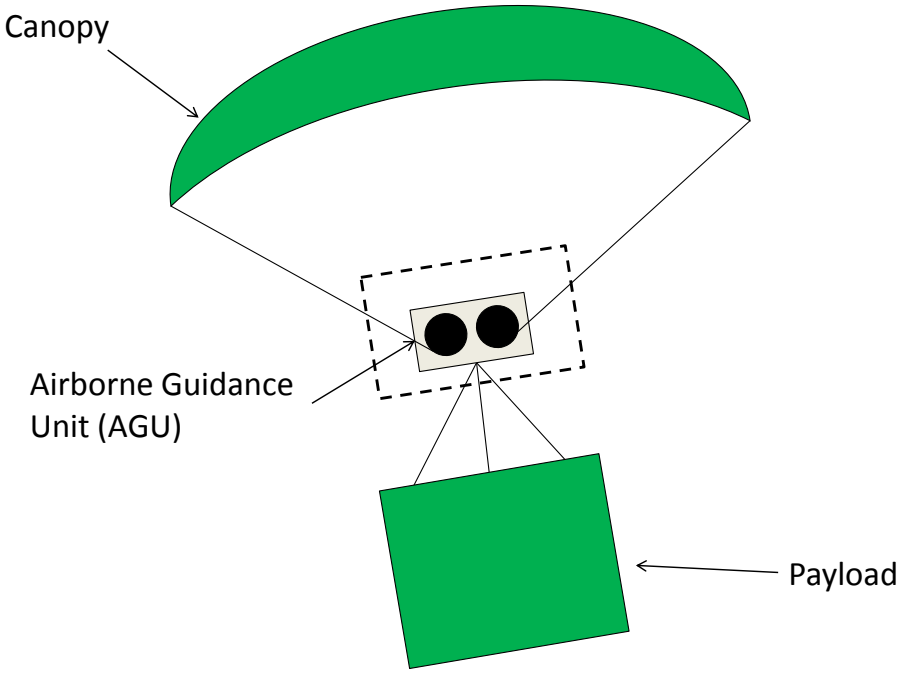

Figure 2-1: Parafoil and Payload System

how the heading angle of the parafoil is defined in the East-North plane. Nominal mission performance is expected if the system can track heading rate commands well. For this work, the heading rate will be indicated as a percentage of the maximum heading rate command (e.g., heading rate of 0.8 means that system is turning at 80% of the maximum command) that guidance is allowed to give. A positive heading rate indicates a right turn, while a negative heading rate indicates a left turn. When a fault occurs, it is likely that the heading rate command tracking performance will be adversely affected due to loss of control authority.

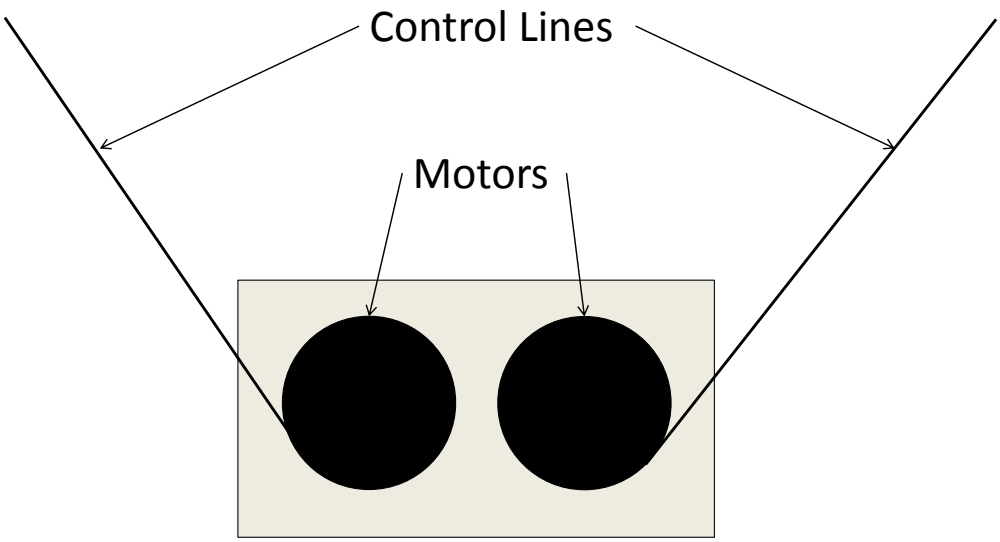

Figure 2-2: Parafoil AGU and Control Lines

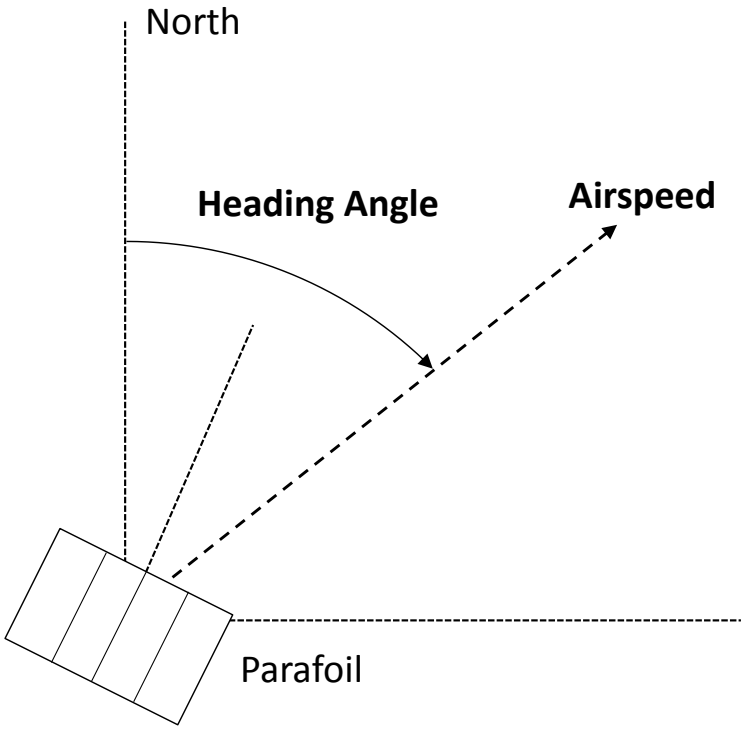

Figure 2-3: East-North Parafoil Coordinate Frame

2.2 Parafoil Dynamics

2.2.1 Full Dynamics and Nonlinear Simulator

Analysis and development of FDIR is performed using Draper Laboratory's high-fidelity, nonlinear parafoil simulation. This simulation utilizes a full nonlinear model of the parafoil, using dynamics similar to those described in Barrows [3], Crimi [10], Mortaloni et al. [33], and Ward et al. [47]. Information about parafoil aerodynamics can be found in Barrows [2] and Slegers and Costello [42]. The simulator provides the capability to run a set of Monte Carlo trials, in which conditions are randomly varied during each trial to provide a reasonable envelope of the flight conditions a parafoil may experience. Parameters varied during each Monte Carlo trial include: initial position, altitude, orientation, and velocity with respect to the target, 3-D wind profile, payload weight, canopy lift-to-drag ratio, and turn rate bias (i.e., nonzero heading rate in the presence of zero control input) [7]. This simulator is used to test new flight software modifications in preparation for a flight test. Large sets of trials, typically 1000 at a time, are run during the design and implementation of the FDIR methods developed in this work. The realism of the simulation provides a strong framework in which FDIR can be validated.

2.2.2 Linearized Lateral Dynamics

A linear model of the parafoil system is used to generate the expected system output used in the generation of the residual signal. This model is chosen to be linear to reduce the computational cost of generating the expected output, but must still generate an output that accurately reflects the parafoil behavior. For this work, a linear model of the lateral system dynamics is used. This model takes as an input the *differential toggle* command and outputs the heading rate. Differential toggle is the difference between the deflection of the right and left control lines,

$$\delta_t = \delta_R - \delta_L \tag{2.1}$$

where $\delta_t \in \mathbb{R}$ is the differential toggle, $\delta_R \in \mathbb{R}$ is the deflection of the right control line, and $\delta_L \in \mathbb{R}$ is the deflection of the left control line. The input to the system is the differential toggle command, or the difference between the commands applied to the right and left motors.

The single-input, single-output (SISO) parafoil model used in this work was developed using a system identification procedure. Input-output data was obtained from flight tests of the parafoil and payload system. The results from this system identification provide information about the lateral dynamics of the parafoil, including two complex poles that capture the dutch roll mode and a third real pole that captures effects from the other lateral modes as well as first-order lag in the system response. The results of the system identification indicate that the linear model of the lateral dynamics used in this thesis provides an accurate model of the heading rate response of the system. The model used in this thesis is described by the discrete-time, linear, time-invariant (LTI) system,

$$x[k+1] = (A + \Delta A)x[k] + (B + \Delta B)\delta_{t,cmd}[k] + w[k] \tag{2.2}$$

$$y[k] = Cx[k] + v[k] \tag{2.3}$$

where $A, \Delta A \in \mathbb{R}^{3\text{x}3}$, $B, \Delta B \in \mathbb{R}^3$, and $C \in \mathbb{R}^{1\text{x}3}$. A, B, and C are known dynamics, control, and output matrices, respectively. ΔA and ΔB represent perturbations to the nominal LTI model derived using system identification, which occur due to modeling errors, differences between the characteristics of each parafoil and payload system, or certain faults. All time-varying inputs to the system are indexed by k, the current time-step of the discrete-time system. The sample rate of the system is 1 Hz. Due to the way the system identification was performed, the states $x[k] \in \mathbb{R}^3$ do not correspond to any physical states. We are concerned instead with the output $y[k] \in \mathbb{R}$, which is the heading rate. The input to the system $\delta_{t,cmd}[k] \in \mathbb{R}$ is the differential toggle command, and can be written equivalently as $\delta_{R,cmd}[k] - \delta_{L,cmd}[k]$, where $\delta_{R,cmd}[k]$ is the right motor command and $\delta_{L,cmd}[k]$ is the left motor command. The noise terms, $w[k] \in \mathbb{R}^3$ and $v[k] \in \mathbb{R}$, characterize the process noise and measurement

noise, respectively, of the system.

The effects of the uncertainty and noise terms, ΔA, ΔB, $w[k]$, and $v[k]$, on the FDI process cause the residual signals to be nonzero even when no fault has occurred (see Section 3.1.1). However, the size of the residual during a healthy flight is small compared to the size of the signal when a fault has occurred. In other words, faults are still observable even if the noise and uncertainty terms are neglected. This is demonstrated by the results in Chapter 3. Therefore, these terms are neglected and the FDI problem is solved using an observer-based approach [16] rather than a Kalman filter-based approach [31].

After ignoring the noise terms, the lateral system dynamics model used in the fault detection observer reduces to,

$$x[k+1] = Ax[k] + B\delta_{t,cmd}[k] \tag{2.4}$$

$$y[k] = Cx[k] \tag{2.5}$$

where A, B, and C are obtained from a system identification on flight data.

The parafoil system has no sensors for measuring heading rate directly; instead, this quantity is estimated using an Extended Kalman Filter (EKF). The only state information that the parafoil software has access to is position and translational velocity data from the onboard GPS. The GPS measures the ground speed as well as the sink rate of the parafoil. An EKF is used to estimate the wind velocity, and from this information the airspeed velocity and the heading rate are estimated, similarly to work done in Ward et al. [47]. This EKF-estimated heading rate is treated as a heading rate measurement, and is used to generate the residual signals used for detection and isolation. The system states x used in Eqs. (2.2) - (2.5) are not available from the EKF and are unknown.

2.3 Nominal Guidance Strategy

The nominal guidance strategy used in this thesis works very well for healthy flights [8] [9]. Despite the success of this strategy for healthy cases, some modifications need to be made when faults are present. These modifications are presented in Chapter 5. The typical guidance strategy for the parafoil system is broken into five modes: preflight, homing, loiter, final approach, and flare [8]. A review of the guidance, navigation, and control system used for many large parafoils is found in Carter et al. [9]. A typical groundtrack trajectory plot of the system is shown in Figure 2-4 with all of the guidance modes labeled. The preflight mode ends once the system has exited the carrier aircraft and canopy deployment is successful. GPS data is not valid until preflight has ended; consequently, FDIR does not begin until after the preflight mode. During the flare mode, both control lines are pulled to the maximum deflection, slowing the parafoil before impact. Since flare occurs so close to the ground, recovery from a fault occurring during this mode would be difficult. As a result, FDIR is not active during flare. This thesis focuses on the effects of a fault on the FDIR process during the homing, loiter, and final approach modes.

2.3.1 Homing

The goal of homing is to fly directly towards the landing target. This requires the system to turn towards the target and maintain zero heading rate once the system is oriented as desired [8]. Under normal conditions, each control line maintains zero deflection. Consequently, a fault that manifests itself as zero line deflection (e.g., broken line) may not be observable during the homing mode.

2.3.2 Loiter

The loiter strategy implemented on nominal parafoil flights is called energy management. During energy management, the parafoil executes figure-eight trajectories in order to decrease altitude while remaining close to the desired landing location. These figure-eight maneuvers consist of homing towards one of two locations (i.e., each end

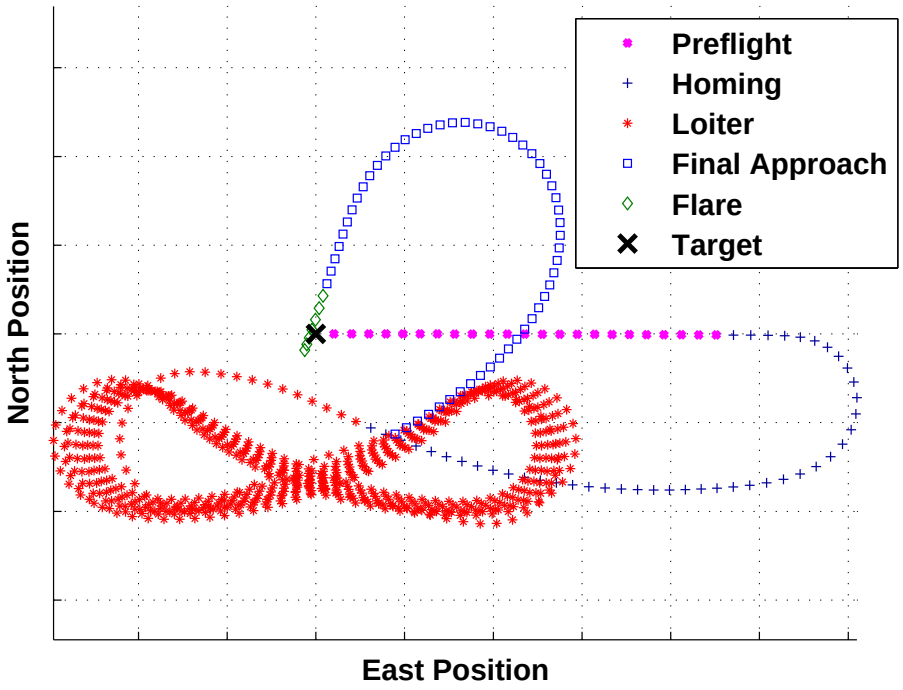

Figure 2-4: Typical System Groundtrack: Nominal Guidance Strategy

of the figure-eight shape), executing a turn once this location is reached, and homing towards the other location. This process repeats until an altitude threshold is reached, at which point the system enters the final approach mode [8].

2.3.3 Final Approach

Many parafoil systems employ Band-Limited Guidance (BLG) as the final approach strategy [8]. The goal of BLG is to guide the parafoil to land into the wind while limiting the commanded heading rate profiles to be much less than the system bandwidth. BLG optimally minimizes a weighted sum of the horizontal distance to the target at impact and the difference between the heading angle at impact and the desired terminal heading angle [8]. This optimization is subject to the constraint,

$$\dot{\psi}_{cmd} \in \left[-\dot{\psi}_{lim}, \quad \dot{\psi}_{lim}\right] \tag{2.6}$$

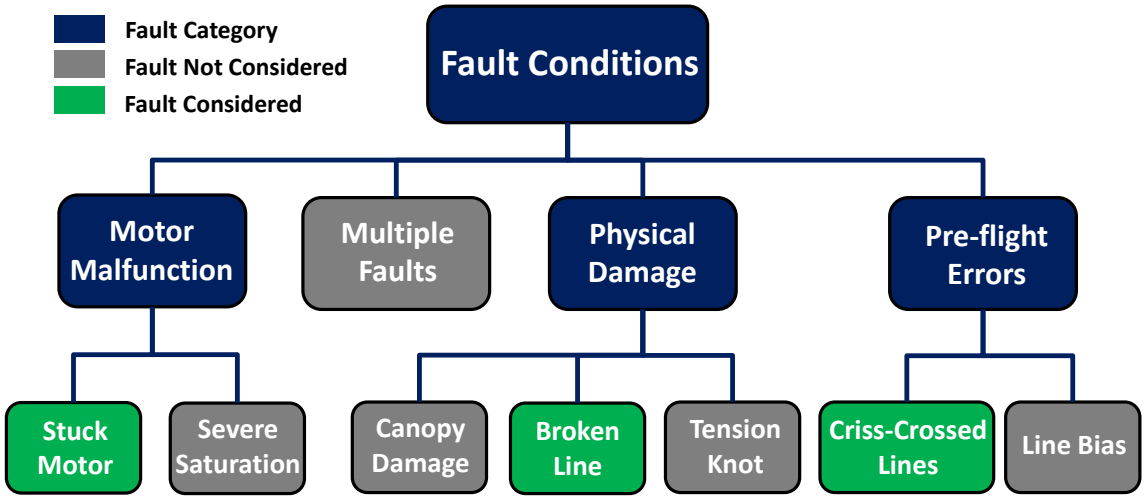

Figure 2-5: Parafoil Fault Hierarchy

where $\dot{\psi}_{cmd}$ is the heading rate command sent to control, and $\dot{\psi}_{lim}$ is the heading rate limit, chosen as a design parameter [8]. If a fault occurs during this mode, there are two issues: 1) the BLG solution is not likely to be optimal, and 2) there is a limited window of time for detection and isolation. Even if a fault is successfully isolated during final approach, the recovery strategy may not have sufficient time to recover performance for a safe landing.

2.4 System Faults

Flight tests have shown that there are common faults that occur that greatly reduce system performance, increasing miss distance and often resulting in a mission failure [11]. These common faults are shown in the hierarchy in Figure 2-5. This work considers three of the most common faults: broken line, stuck motor, and criss-crossed lines. These faults are chosen for analysis because they have effects that are well-understood and they occur relatively commonly in flight tests.

2.4.1 Criss-Crossed Lines

It is possible, while rigging the control lines to the AGU, that the line attached to the left trailing edge of the parafoil is spooled around the right motor, and vice

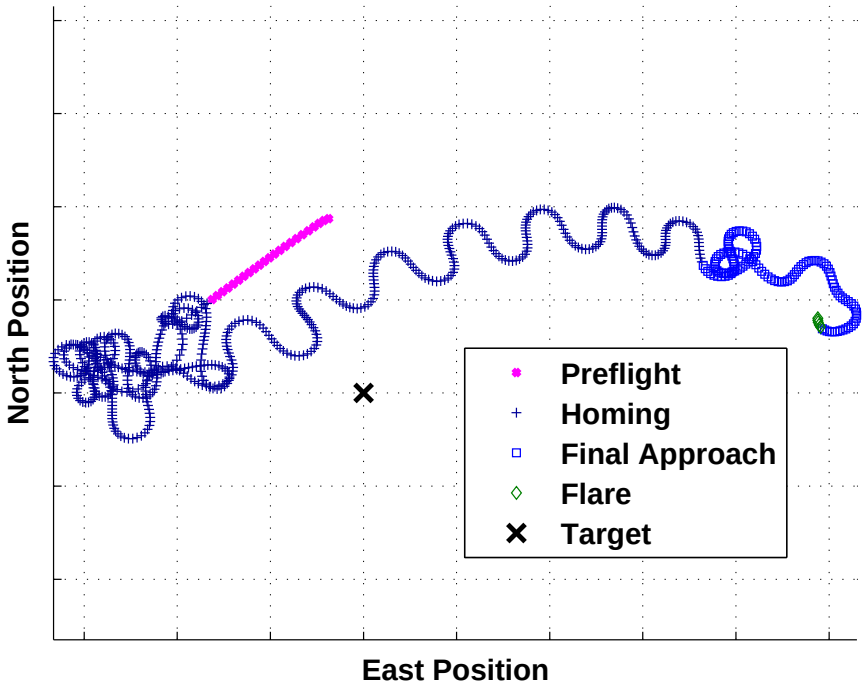

Figure 2-6: East-North Groundtrack: Criss-Crossed Lines Fault

versa. In this case, a command to the right motor will yield a deflection of the left control line, and a command to the left motor will yield a deflection of the right control line. This fault is called criss-crossed lines. This is an example of a fault that has a straightforward recovery strategy. No change to the existing guidance strategy is necessary; the controller need only reverse the commands given to each motor to achieve the desired performance. However, this recovery approach cannot be implemented unless FDI successfully detects and isolates the fault. This will be addressed in Chapters 3 and 4. Figure 2-6 shows an example groundtrack of a simulation in which a system is incorrectly rigged, resulting in a criss-crossed lines fault. After the preflight mode ends, homing begins and the system attempts to turn towards the target. However, the criss-crossed lines fault reverses the effect of each control input. Consequently, the system goes away from the target for the duration of the flight. This results in a very large miss distance and a mission failure.

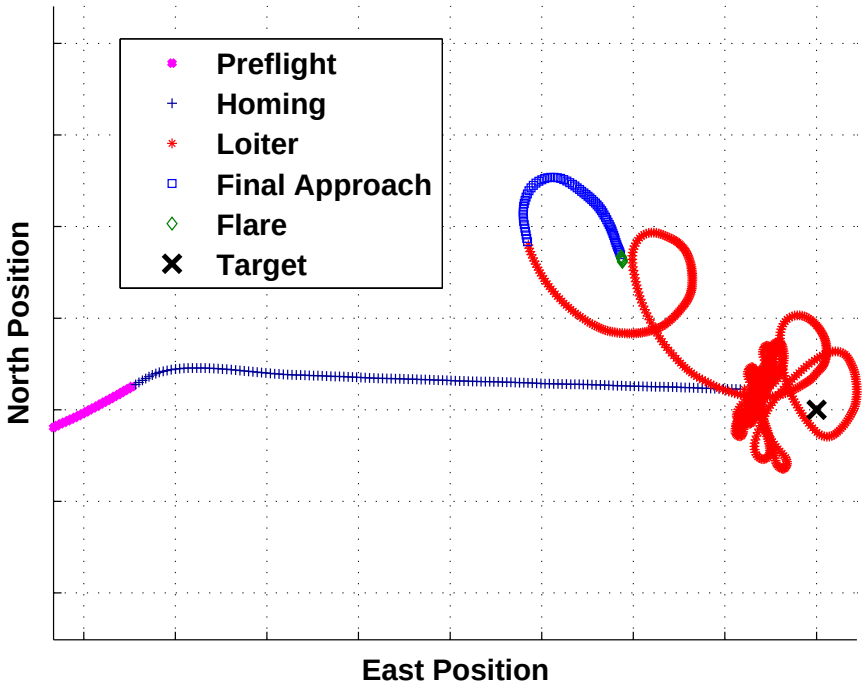

Figure 2-7: East-North Groundtrack: Broken Line Fault

2.4.2 Broken Line

A broken line fault occurs when one of the control lines attached to the motors on the AGU breaks. In this case the motor is still free to turn, but there is no corresponding response in line deflection. This prevents the parafoil from turning in the direction of the side on which the line is broken (e.g., system with a broken left line can only turn to the right). This fault often occurs upon canopy deployment [11]. Figure 2-7 shows an example of the groundtrack from a simulation in which a broken left line occurs during the loiter phase. The system progresses through the flight normally until the fault occurs. At that point, the system begins to spiral away from the target while making right turns in an attempt to track the desired figure-eight trajectory. The system then executes terminal guidance and lands far from the target.

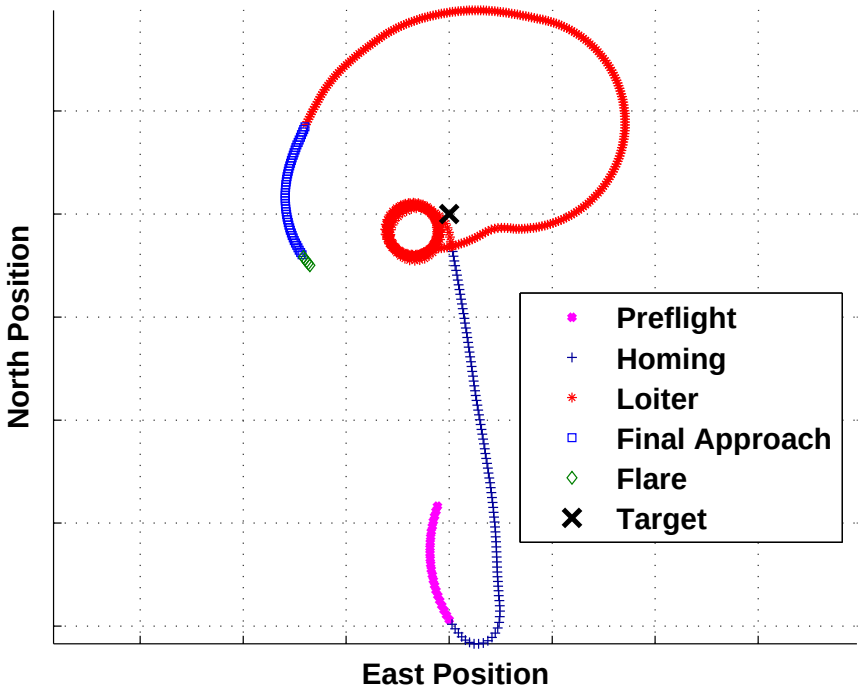

Figure 2-8: East-North Groundtrack: Stuck Motor Fault

2.4.3 Stuck Motor

A stuck motor fault occurs when one of the motors on the AGU freezes at a particular location for the remainder of the flight. This forces the control line attached to the faulty motor to be stuck at a fixed deflection. The other motor must be used to attempt to track heading rate commands. Figure 2-8 shows an example of the groundtrack from a simulation in which the right motor becomes stuck at a small value (i.e., right turn cannot be made) occurs just before the loiter phase. The system cannot track the desired figure-eight trajectory during loiter, and lands far from the target. Again, lack of knowledge of the type of fault that has occurred prevents an adjustment of the guidance strategy and the mission is a failure.

2.4.4 Other Faults

The other faults listed in the fault hierarchy in Figure 2-5 are not considered for FDIR in this work. First, the canopy damage fault has a broad set of manifestations. The

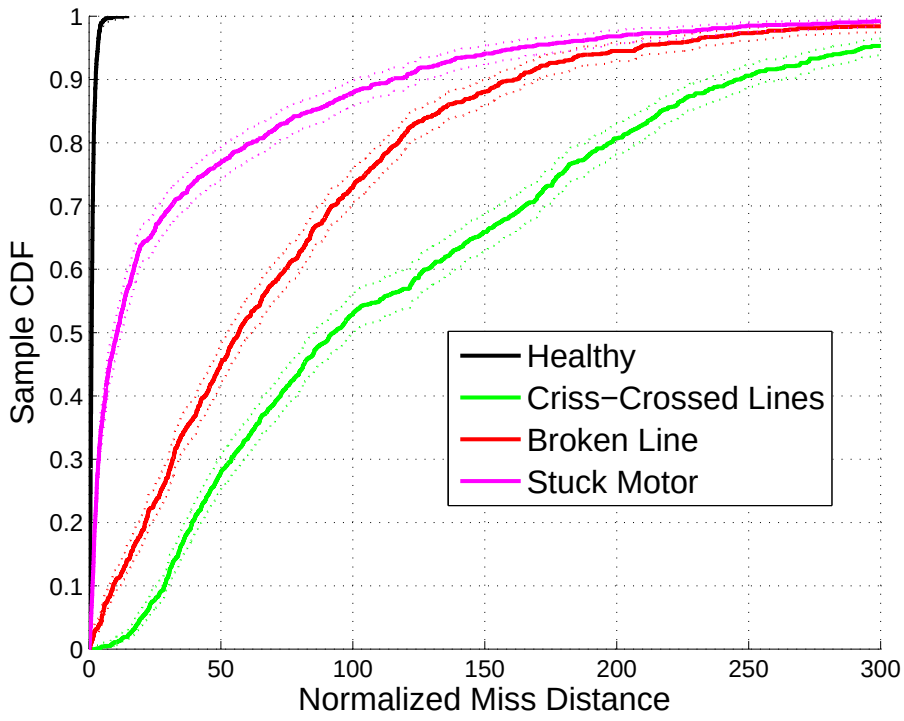

Figure 2-9: Miss Distance CDFs with 95% Confidence Bounds: Nominal Guidance Strategy

FDI method presented in this work is not well-equipped to deal with this type of fault. Adaptive control work may be more suitable to this class of faults [11]. The effects of other faults such as tension knot and line bias, which both cause turn rate bias, are mitigated by the integrator on the system's PID controller. Multiple faults and severe motor saturation occur rarely, and the similarity of severe motor saturation to a stuck motor fault makes classifying severe saturation separately impractical.

2.5 Effect of Faults on System Performance

The faults described in Section 2.4 severely impact system performance, resulting in large miss distances. Figure 2-9 shows cumulative density functions (CDFs) of miss distance results from 1000 Monte Carlo simulations of healthy flights as well as 1000 simulations each of flights in which one of the three faults considered in this thesis (i.e., stuck motor, broken line, or criss-crossed lines) occur. The miss distances

in Figure 2-9 are normalized so that a miss of 1 corresponds to the median miss distance recorded from the healthy flight simulations. This quantity will be used to normalize all miss distances in this thesis. The median miss distances for the criss-crossed lines and broken line cases are well over 50 times the median miss for healthy flights, and a large percentage of flights with stuck motor faults have miss distances at unacceptable levels. For this work, a normalized miss distance greater than 10 is considered unacceptably large. This is a design parameter that is chosen based on mission requirements. The performance degradation caused by the presence of faults motivates the need for an effective FDIR method that can drive miss distances down to acceptable values, even in the presence of the faults considered in this work. Chapter 3 describes the first step of this process, fault detection.

Chapter 3

Detection

The detection method used in this work belongs to a class of methods referred to as observer-based [16]. Observer-based methods utilize an observer of the system that generates an expected system output. The output of the observer, which models the system in a healthy state, is compared to the corresponding output as measured by the sensors on the system. The difference between these two quantities is called the residual signal. This signal should be chosen such that it is large when a fault has occurred and small in the healthy case [16].

Detection is broken into two phases: residual generation and residual evaluation [22]. Residual generation is the process of constructing the residual signal. Residual evaluation is the process of taking this signal and using it to either validate or reject a null hypothesis. The null hypothesis is that the system is healthy; a rejection of this hypothesis indicates a fault. Residual evaluation is performed using a threshold, which is designed so that if the residual signal rises above this threshold there is a reasonable probability that a fault is present [16],

$$\text{If } r[k] \leq \lambda_{th}, \text{ null hypothesis confirmed} \tag{3.1}$$

$$\text{If } r[k] > \lambda_{th}, \text{ null hypothesis rejected; fault} \tag{3.2}$$

where $r[k] \in \mathbb{R}$ is a time-varying residual signal and $\lambda_{th} \in \mathbb{R}$ is a mission-specific, constant threshold value. The FDI method in this work uses the parafoil heading rate

output for residual generation. The parafoil guidance system commands the parafoil by specifying a desired heading rate. If the parafoil is not tracking the heading rate as expected, the system is likely in a faulty condition.

3.1 Residual Generation

3.1.1 Heading Rate Residual

The linear model described in Eqs. (2.4) and (2.5) is used to construct the observer used for the *heading rate residual* generation process of fault detection. The heading rate residual is the primary residual used for fault detection, and is based on how well the measured heading rate of the system matches the expected heading rate. Figure 3-1 shows a block diagram of the observer-based residual generation logic. The differential toggle command $\delta_{t,cmd}$ given by control is passed into the motor on the AGU, as well as an observer of the lateral dynamics of the parafoil. The motors on the AGU deflect to the desired position, yielding the motor differential toggle δ_t. This process is subject to actuator faults; a stuck motor, for example, will likely cause δ_t to be different from $\delta_{t,cmd}$. The actuation of the motors deflects the control lines that act on the parafoil system itself, which is subject to non-actuator faults (e.g., broken line or criss-crossed lines). The resulting state of the system is measured by the GPS. The heading rate y is then estimated using an EKF. This measured heading rate is compared to the heading rate output $\hat{y}$ from the observer model. This signal error is fed back into the observer and scaled by the observer gain L.

The observer used to generate the residual signal is based on the LTI parafoil model in Eqs. (2.4) and (2.5) and is given as follows,

$$\hat{x}[k+1] = A\hat{x}[k] + B\delta_{t,cmd}[k] + L(y[k] - \hat{y}[k]) \tag{3.3}$$

$$\hat{y}[k] = C\hat{x}[k] \tag{3.4}$$

where $\hat{x}[k] \in \mathbb{R}^3$ is an estimate of the system states $x[k]$, and $\hat{y}[k] \in \mathbb{R}$ is the observer estimate of the heading rate. The feedback gain $L \in \mathbb{R}^3$ is designed to make $A - LC$

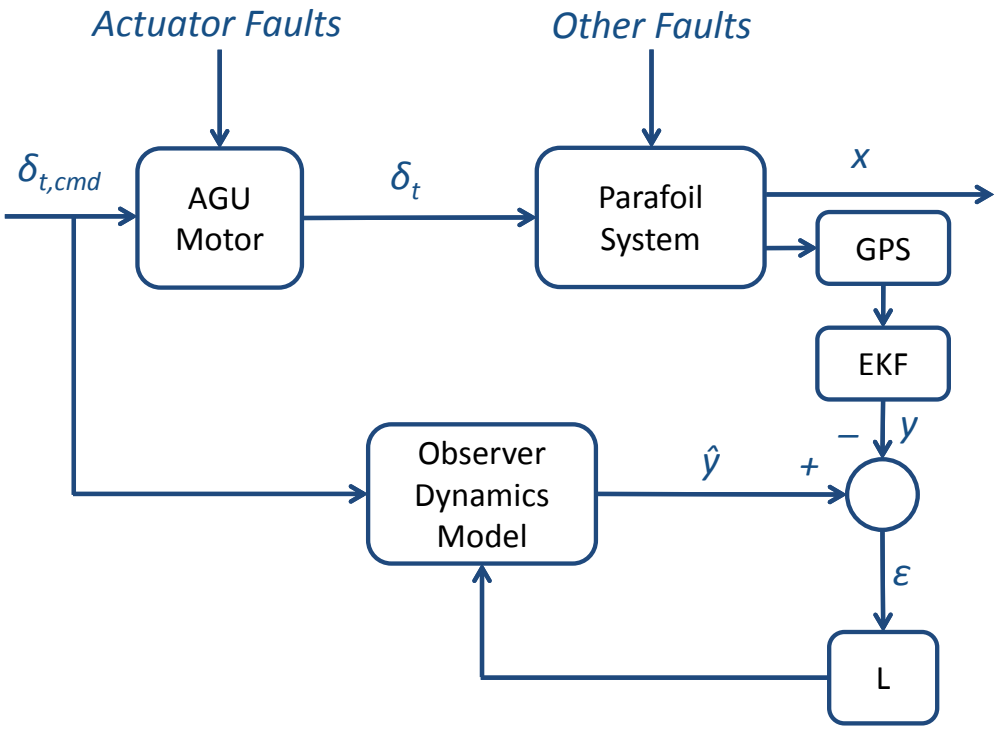

Figure 3-1: Heading Rate Residual Generation Block Diagram

stable and well-behaved by using the Linear Quadratic Estimator (LQE) formulation [28], described in Appendix A. Error terms are defined as,

$$e[k] = \hat{x}[k] - x[k] \tag{3.5}$$

$$\varepsilon[k] = \hat{y}[k] - y[k] \tag{3.6}$$

and the residual used for fault detection is chosen as,

$$r[k] = \varepsilon^2[k] \tag{3.7}$$

which is evaluated at each time step to determine whether a fault alarm should be raised. By substituting Eqs. (3.4) and (2.3) into Eq. (3.6), $\varepsilon[k]$ is written as,

$$\varepsilon[k] = Ce[k] - v[k] \tag{3.8}$$

which indicates that the residual signal will almost always be nonzero due to measurement noise. Furthermore, substituting Eqs. (3.3) and (2.2) into Eq. (3.5) and simplifying yields,

$$e[k+1] = (A - LC)e[k] + Lv[k] - w[k] - (\Delta A x[k] + \Delta B \delta_{t,cmd}[k]) \tag{3.9}$$

where designing L such that $A-LC$ is stable drives the first term to zero in the steady-state [1]. The other three terms in Eq. (3.9) cause the error signal to grow, and the residual signal to be nonzero. The process noise term $w[k]$ and measurement noise term $Lv[k]$ will always cause fluctuations in the residual. The dynamics uncertainty term ΔA causes the error signal to change when modeling errors or canopy damage faults have occurred. The control uncertainty term ΔB causes the error signal to change when modeling errors or faults that affect control authority (e.g., criss-crossed lines, broken line, stuck motor) occur. In order for fault detection to be successful, the effects of the faults on the error signal must be much more observable than the effects of the noise and modeling errors [16]. The residual data in Section 3.2 show that this is indeed the case.

3.1.2 Motor Residual

The heading rate residual signal designed in Eq. (3.7) is useful for detecting the presence of generic severe faults. However, detecting mild actuator faults is difficult because of the effects of these faults on system performance. For example, consider a stuck left motor fault at 0.5 (i.e., left motor is stuck at half of its maximum deflection). The parafoil still has approximately half of its maximum turning authority in each direction. Therefore, many heading rate commands are achievable even in the presence of this fault. The heading rate residual signal will not exhibit off-nominal behavior in this case. Despite the mild effects of these types of faults, it is still beneficial to be able to detect when stuck motor faults occur. As a result, a new residual is introduced, called the *motor residual signal*.

Generation of this residual signal is accomplished using a model of the motor

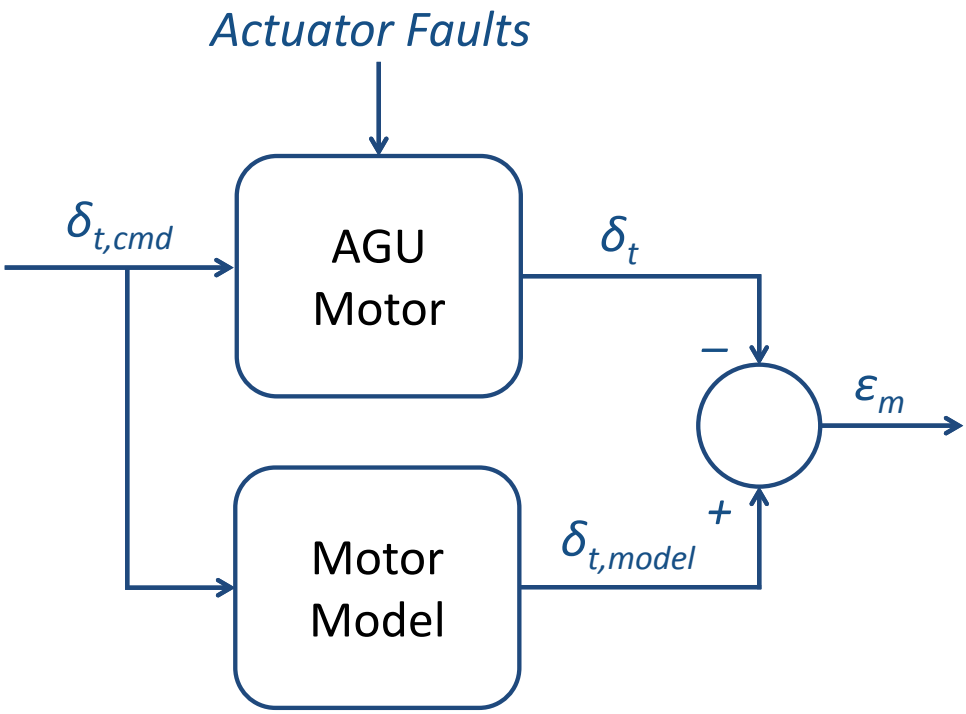

Figure 3-2: Motor Residual Generation Block Diagram

used on the parafoil. This model is based on the one used in the Draper simulation described in Section 2.2.1, and forces a first-order lag between the differential toggle command and the resulting differential toggle of the motors. In the healthy case, the output from the motor model should be approximately equal to the motor deflections as measured by the encoders on each motor. Each motor residual signal is generated by computing the difference between the model output and measurement, as shown in Figure 3-2. The motor residual signals are chosen as,

$$\varepsilon_{m,L}[k] = \delta_{L,model}[k] - \delta_L[k] \tag{3.10}$$

$$\varepsilon_{m,R}[k] = \delta_{R,model}[k] - \delta_R[k] \tag{3.11}$$

$$r_{m,L}[k] = |\varepsilon_{m,L}[k]| \tag{3.12}$$

$$r_{m,R}[k] = |\varepsilon_{m,R}[k]| \tag{3.13}$$

where $\delta_{L,model}[k]$ and $\delta_{R,model}[k]$ are the left and motor deflections, respectively, that are the outputs from the model, $r_{m,L}[k]$ is the residual for the left motor, and $r_{m,R}[k]$ is the residual for the right motor. If one of the residual signals is large, it is likely that a fault has occurred in the corresponding motor. Analyzing the heading rate and motor residual signals concurrently provides a means for detecting the presence of the three fault cases considered in this thesis. If one of these signals crosses its corresponding threshold, a fault alarm is raised.

3.2 Residual Evaluation

A threshold is the main tool used in this work for residual evaluation. This threshold is chosen such that there is a high probability that a fault is present when the residual is above the threshold and a low probability of a fault when the residual signal is below the threshold. Statistical methods are used for threshold determination. To improve detection statistics, the heading rate and the motor residual signals at each time step are smoothed using a discrete-time, low-pass filter, described in Appendix B. The filter cutoff frequency can be varied according to design needs. A lower cutoff frequency captures the trend of the signal while filtering out noise, but will cause a lag between the occurrence of a fault and the response of the signal. This parameter was tuned using frequency response information from the parafoil model and motor model to achieve desired detection characteristics. Since the smoothed residual signal calculated using the filter in Eq. (B.1) is what is evaluated for fault detection, the term *residual* will be used for the remainder of this thesis to refer to a smoothed residual signal.

It is important to note that FDI does not begin immediately after the parafoil is released from the aircraft and the canopy is deployed. FDI is off for the entire preflight mode, the period of time after canopy deployment before navigation data is reliable. After preflight ends, FDI begins computing residual values at each time step. However, the residuals are not evaluated until 15 seconds after residual data collection begins. This is to allow time for the observer to converge so the residuals

settle to reasonable values. Often, there is a spike in the signal during the first few time steps that is not indicative of true system performance. All residual evaluation occurs after preflight ends and the 15-second settling period concludes. All analysis and data collection relating to threshold design only considers residual values after this point.

When designing a detection threshold, the goal is to minimize two quantities: probability of missed detection $P(MD)$, and probability of false alarm $P(FA)$ [39] [43]. The probability of missed detection is the probability that a fault has occurred and no fault alarm is raised; the probability of false alarm is the probability that an alarm is raised when no fault has occurred. These quantities are predicted by collecting data from simulations of healthy flights and flights in which a fault occurs. CDFs of data from simulated healthy flights and flights in which a fault occurs are useful for visualizing how a chosen threshold affects $P(FA)$ and $P(MD)$.

3.2.1 Heading Rate Residual

A threshold is chosen for the heading rate residual signal, the primary detection residual, by collecting data from a set of Monte Carlo simulations. To generate CDFs that will accurately show $P(FA)$ and $P(MD)$ for various thresholds, large data sets are collected that reflect the range of conditions a parafoil system experiences during a healthy flight as well as flights in which faults of varying type and severity occur. For the healthy data set, 1000 Monte Carlo simulations are performed; characteristics of each simulation are varied as described in Section 2.2.1. This set of 1000 simulations reflects an envelope of operating conditions that could be expected in flight. No fault occurs during any of these simulations. The data of interest from each Monte Carlo trial is the maximum value that the residual signal reaches during the simulation. This maximum bounds the residual signal expected from healthy flights.

Data collection from the fault cases is treated differently than data collection from healthy flight simulations. Again, 1000 Monte Carlo simulations are performed, but for these simulations random non-actuator faults are inserted. The nature of each fault is chosen with uniform probability to be either a broken line or criss-crossed lines.

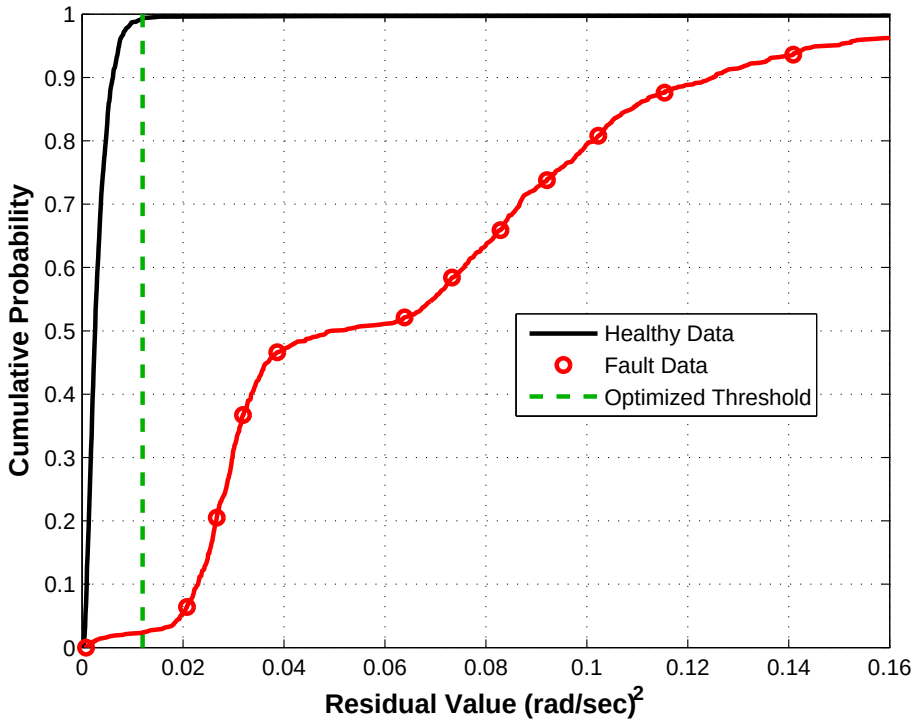

Figure 3-3: CDFs for Healthy and Fault Data: Heading Rate Residual

Stuck motor faults are not simulated because the motor residual signal is designed to detect the presence of that particular fault. Detection of stuck motor faults is covered in Section 3.2.2. The altitude at which each simulated broken line fault occurs is chosen with uniform probability from 100 meters AGL up to the altitude at which the parafoil is released from the aircraft. The side on which a broken line occurs is also chosen randomly. In this thesis, randomization of broken line faults during Monte Carlo simulations are always performed as described here, unless otherwise specified. The criss-crossed lines fault, a rigging error, always occurs at the beginning of the flight. Multiple faults are not considered. The data chosen from these runs in which a non-actuator fault is present is the maximum value that the residual signal reaches after the fault occurs. This bounds the size of the heading rate residual during a flight in the presence of a particular fault.

Figure 3-3 shows CDF plots of data from the Monte Carlo simulations. As expected, the plots indicate that the heading rate residual tends to reach a large max-

imum when a fault is present, and remains small in the healthy case. The selection of an appropriate threshold given the data in Figure 3-3 depends on the emphasis placed on minimizing $P(FA)$ versus $P(MD)$. The following figure of merit (FOM) is used to penalize $P(FA)$ and $P(MD)$ as desired [38]:

$$FOM_{det} = \frac{c_{FA}P(FA) + c_{MD}P(MD)}{c_{FA} + c_{MD}} \tag{3.14}$$

where the weightings c_{FA} and c_{MD} can be varied according to design needs, where a higher weighting on either $P(FA)$ or $P(MD)$ indicates that it is more important to minimize that particular probability. Then, a threshold is determined that minimizes the chosen FOM. An optimized threshold is chosen by using an unconstrained minimization algorithm in MATLAB [30]. When equal weighting is placed on minimizing $P(FA)$ and $P(MD)$ (i.e., $c_{MD} = c_{FA}$), the optimized threshold is 0.0120 $(\text{rad/sec})^2$. This threshold is shown as a green, dotted, vertical line in Figure 3-3. The cumulative probability at the intersection of this vertical line with the fault data CDF gives the expected $P(MD)$ that results from this threshold choice. The intersection of the vertical line with the healthy data CDF gives $1 - P(FA)$ expected with this threshold choice. A threshold of 0.0120 $(\text{rad/sec})^2$ yields an expected $P(MD)$ of 2.37% and an expected $P(FA)$ of 0.68%. Table 3.1 shows optimized thresholds and resulting detection probabilities for several different weightings on $P(FA)$ and $P(MD)$.

Table 3.1: Optimized Thresholds for Various FOMs: Heading Rate Residual

$\mathbf{c_{MD}}$	$\mathbf{c_{FA}}$	**Threshold** $(\frac{\text{rad}}{\text{sec}})^2$	**P(MD)**	**P(FA)**	**FOM**
0.01	0.99	0.0170	0.0325	0.0039	0.0042
0.20	0.80	0.0160	0.0298	0.0040	0.0091
0.30	0.70	0.0150	0.0285	0.0043	0.0116
0.50	0.50	0.0120	0.0237	0.0068	0.0152
0.90	0.10	0.0100	0.0222	0.0129	0.0212
0.95	0.05	0.0080	0.0199	0.0340	0.0234
0.99	0.01	0.0010	0.0020	0.8740	0.0107

For the parafoil mission, minimizing false alarms is crucial. Since the frequency of drops is high, even a small false alarm rate has a significant negative impact on healthy flights. The heading rate residual threshold, T_{det}, chosen for this system is 0.0160 (rad/sec)^2 to limit the expected $P(FA)$ to 0.4%. This threshold will be used for the remainder of this thesis for fault detection using the heading rate residual signal, unless otherwise specified.

3.2.2 Motor Residual

A similar process is used to determine the threshold for the motor residual signals. These residuals are used for detecting motor faults, and will be used during isolation as well (see Section 4.1). Data collection is performed by running 1000 Monte Carlo simulations of healthy flights, and 1000 simulations of flights in which a stuck motor fault occurs. The altitude at which the fault occurs is chosen with uniform probability from 100 meters AGL up to the altitude at which the parafoil is released from the aircraft. The side on which the motor is stuck, as well as its value (e.g., 0.2) are also chosen randomly from a uniform distribution. In this thesis, randomization of stuck motor faults during Monte Carlo simulations are always performed as described here unless otherwise specified. Similarly to data collection for the heading rate residual described in Section 3.2.1, the CDFs in Figure 3-4 indicate the maximum value of the motor residual; for the healthy cases the signal is evaluated for the entire flight, for the fault cases the residual is evaluated only after the fault has occurred. The data indicated by the CDFs in Figure 3-4 are collected from the motor residual corresponding to the fault that is present (e.g., left motor residual data collected if left stuck motor fault occurs). For the healthy case, the data is the maximum of both motor residual signals.

As with the heading rate residual in Section 3.2.1, weightings on the FOM in Eq. (3.14) must be chosen for $P(FA)$ and $P(MD)$ to determine an optimized threshold. For equal weightings on each probability, the optimized threshold is 0.1390 meters. This threshold is shown as a vertical dotted line in Figure 3-4. Table 3.2 shows optimized thresholds for several different FOMs. Since the threshold chosen

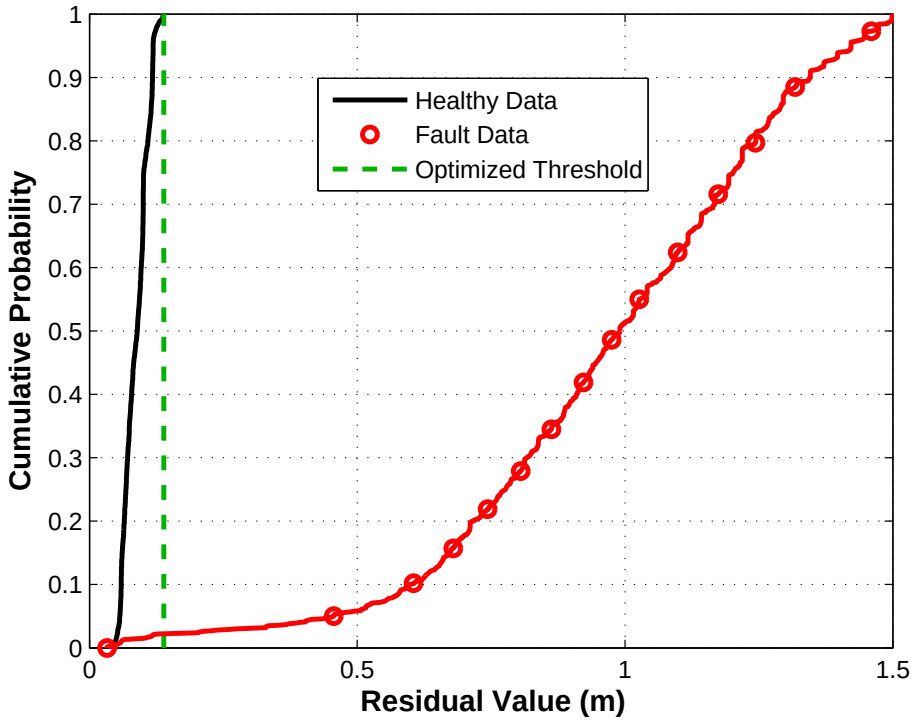

Figure 3-4: CDFs for Healthy and Fault Data: Motor Residual

with equal weighting on minimizing false alarm and missed detection has zero instances of false alarm during 1000 simulations, a motor residual threshold, T_m, of 0.1390 meters will be used for the remainder of this thesis unless otherwise specified. Since the heading rate residual is the primary detection residual, it is important that the motor residual have a negligible impact on the false alarm rate.

Table 3.2: Optimized Thresholds for Various FOMs: Motor Residual

c_{MD}	c_{FA}	Threshold $(\frac{rad}{sec})^2$	P(MD)	P(FA)	FOM
0.01	0.99	0.1390	0.0224	0.0000	0.0002
0.50	0.50	0.1390	0.0224	0.0000	0.0112
0.97	0.03	0.1200	0.0212	0.0360	0.0216
0.99	0.01	0.0350	0.0020	0.9989	0.0120

3.3 Detection Results

The integration of the heading rate residual evaluation and motor residual evaluation into the complete detection method is simple; if any one of these signals crosses its corresponding threshold, a fault alarm is raised. Performance evaluation of the detection method consists of using the thresholds chosen in Sections 3.2.1 and 3.2.2 during simulated flights and evaluating $P(FA)$ and $P(MD)$. These values should be comparable to the expected values given each threshold choice, shown in Tables 3.1 and 3.2. Results for the full detection method are given along with results from isolation in Section 4.4.2.

Table 3.3: Fault Detection Test with Various Threshold Choices

Non-Actuator Fault			**Actuator Fault**		
T_{det}	$P(MD)$	$P(FA)$	T_m	$P(MD)$	$P(FA)$
0.0010	0.0000	0.8740	0.0600	0.0020	0.8720
0.0030	0.0060	0.4150	0.1000	0.0110	0.3270
0.0120	0.0230	0.0070	0.1200	0.0210	0.0360
0.0400	0.4690	0.0040	0.1390	0.0224	0.0000
0.1600	0.9620	0.0030	0.8490	0.3330	0.0000

To analyze the effects of different threshold choices on detection performance, the fault detection method presented in this chapter is tested on three sets of 1000 Monte Carlo simulations of flights with randomized conditions, as described in Section 2.2.1. One set consists of all healthy flights, the second set consists of flights in which a random non-actuator fault occurs, and the third consists of flights in which a stuck motor fault occurs. The severity of each fault and altitude at which the fault occurs is randomized where applicable. $P(FA)$ is evaluated using the healthy flight simulations, and $P(MD)$ is evaluated using the simulations in which a fault is present. The heading rate and motor residual thresholds are evaluated separately. While evaluating the heading rate residual threshold, the motor residual threshold is not used. The converse is true when evaluating the performance of the motor residual threshold.

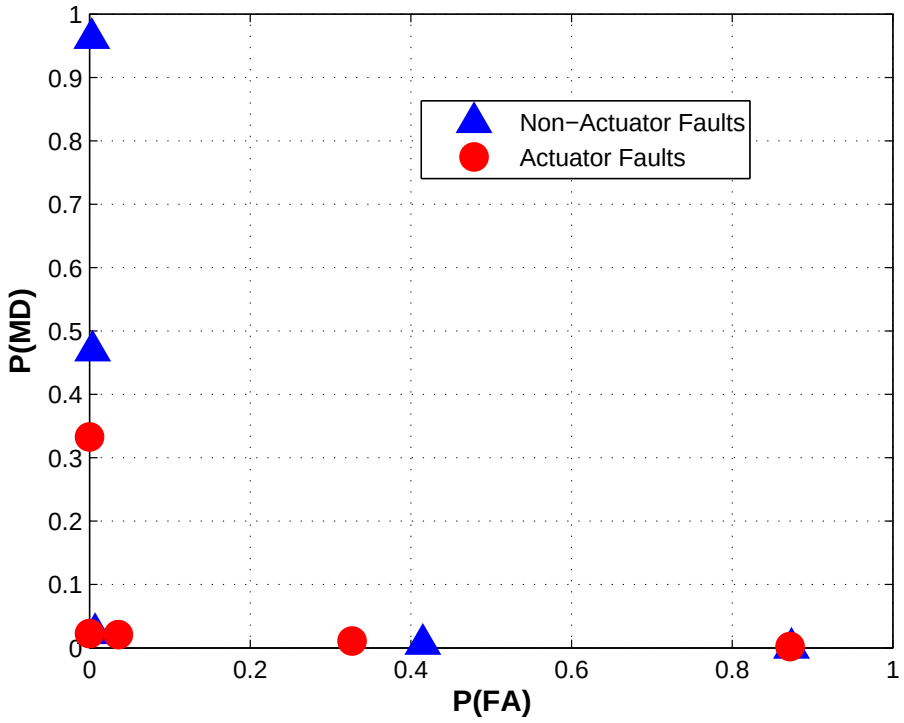

Figure 3-5: Missed Detection and False Alarm Rates from Simulation

Results from these simulations show how varying thresholds effects the rates of false alarm and missed detection.

Figure 3-5 plots $P(MD)$ versus $P(FA)$ for each of a set of threshold choices. The closer the data are to the origin, the better the overall performance [38]. These plots also indicate an important point about fault detection: there is always a tradeoff between false alarm and missed detection [37]. One can choose a threshold according to design needs by varying the figure of merit in Eq. (3.14). Table 3.3 tabulates the results from Figure 3-5.

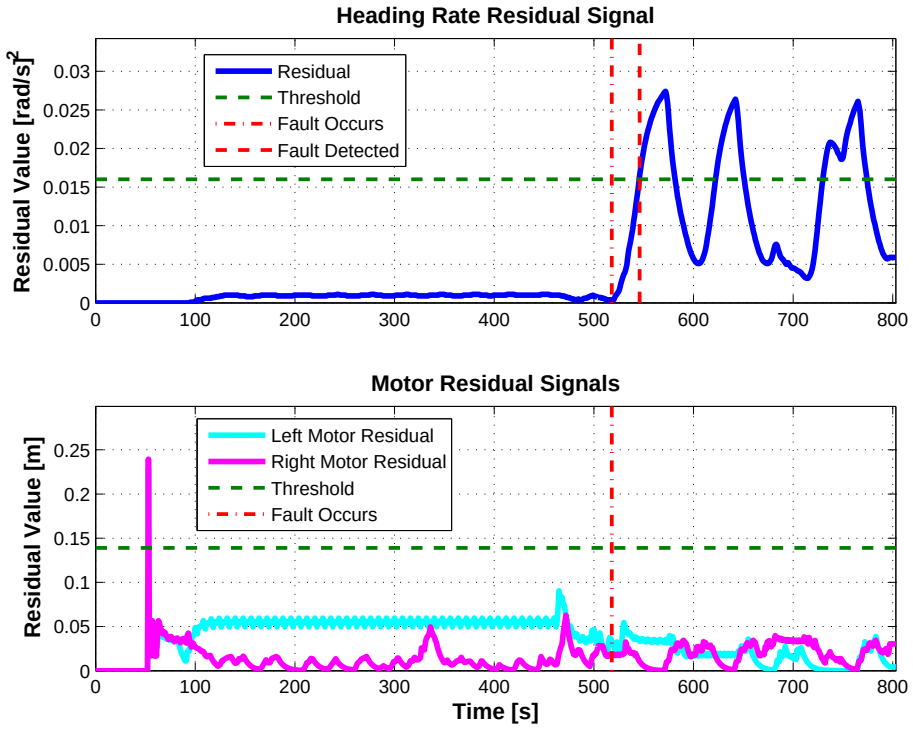

Figure 3-6: Fault Detection Example: Broken Right Line

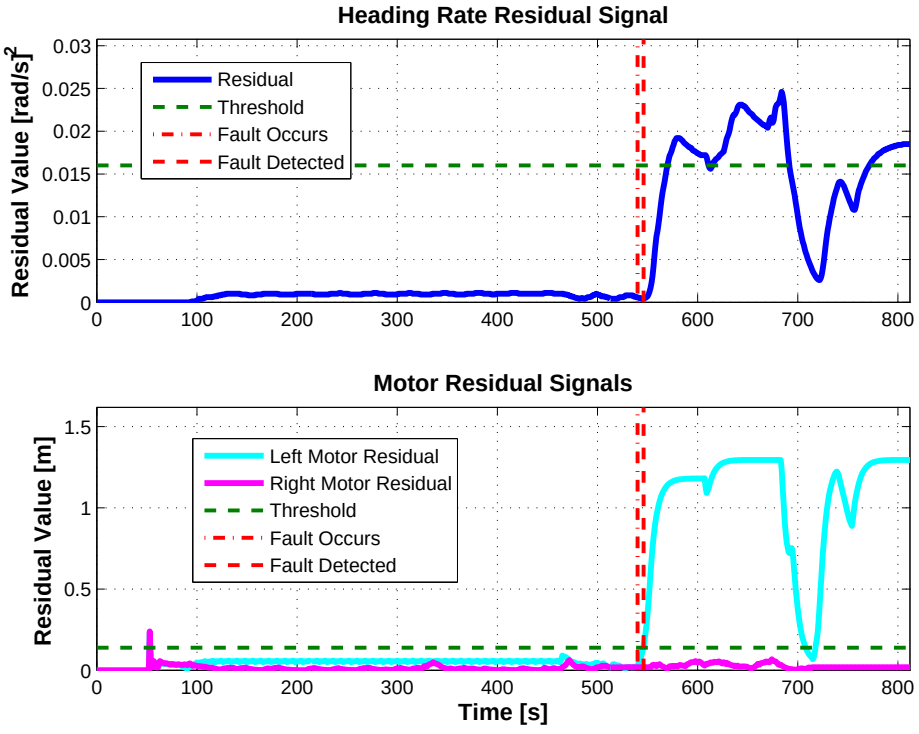

Figure 3-7: Fault Detection Example: Stuck Left Motor

Figure 3-6 shows an example of a simulation during which a broken right line occurs 518 seconds into the flight. There is a spike in the motor residuals early in the flight, however this occurs immediately after preflight, before the 15-second settling time described in Section 3.2 has expired. This is common for the residual signals at this point in the flight. For the remainder of the simulation, both the left and right motor residuals remain under the threshold, as expected from a non-actuator fault. The heading rate residual, however, responds quickly to the presence of the fault, and crosses the threshold 28 seconds after the fault occurs.

Figure 3-7 shows an example of a simulation in which there is a stuck left motor at a deflection of 0.15 that occurs 540 seconds into the flight. The heading rate residual remains under the threshold for several seconds after the fault occurs and is slow to respond. However, the left motor residual rises very quickly and raises a detection alarm only 6 seconds after the fault occurs. These two examples demonstrate how both the heading rate residual and motor residuals can be used for fast, effective fault detection. Chapter 4 describes how, once a detection alarm is raised using the methods described in this chapter, the isolation process determines which particular fault is present.

Chapter 4

Isolation

Once the detection algorithm has determined that a fault has occurred, the isolation process attempts to determine which particular fault is present. The data from the AGU motors can be used to determine if the fault is actuator-related. Isolation evaluates the motor residual signals developed in Section 3.1.2. Each signal, one for each motor, should be small when the motor is behaving well and large when an actuator fault has occurred. The thresholds determined in Section 3.2.2 are used for motor fault isolation as well as detection.

A bank of *fault-specific observers* [48] is used to attempt to declare that a particular non-actuator fault has occurred. Some faults, such as a stuck motor, are difficult to model a priori because the fault is parameter-dependent (e.g., a stuck left motor at 0.75) and can manifest itself in many different ways. Other faults, such as criss-crossed lines and broken line, can be modeled in a straightforward manner as the effects of the faults are well-known. Each fault-specific observer models the parafoil system in the presence of a particular fault. When a residual signal generated from a fault-specific observer is small, it is likely that the system has experienced the fault associated with that particular observer. Successful isolation will result in the declaration of a stuck left motor, stuck right motor, broken left line, broken right line, or a criss-crossed lines fault.

4.1 Evaluation of Motor Residuals

After a fault detection alarm is raised, the motor residual signals described in Section 3.1.2 are analyzed to determine whether an actuator fault is present. If, at any time after the detection alarm is raised and before isolation has declared a fault, both the left and right motor residual signals are greater than the chosen threshold, isolation is inconclusive. At worst, this represents a complete loss of actuator control, otherwise it may be a mistake on the part of FDI. If this occurs, FDI is restarted and another attempt at successful detection and isolation is made. When FDI is restarted, the algorithm assumes that no faults are present and resets all of the residual signals, starting residual generation and smoothing over again without retaining old data.

If only one of the motor residual signals crosses above the threshold, the fault corresponding to that residual is declared and isolation concludes. For example, if the left motor residual crosses the threshold and the right motor signal remains below the threshold, a stuck left motor is declared and the recovery process begins (see Chapter 5). This motor residual evaluation process occurs concurrently with the evaluation of fault-specific observers, which are described in Section 4.2. Once any fault is declared, isolation ends. For example, if a stuck left motor fault is declared evaluation of all fault-specific observer residuals immediately ceases. Intelligently choosing thresholds limits the number of times that a fault is isolated incorrectly.

4.2 Fault-Specific Observers

4.2.1 Residual Generation

Evaluation of fault-specific observers comprises the second part of the isolation process, which occurs concurrently with evaluation of the motor residuals. As with the heading rate residual, the residuals for fault-specific observers are constructed by differencing the EKF-estimated heading rate from the actual system and the expected heading rate output from the system observer. However, the expected heading rate signal comes from an observer that uses the dynamics of a system with a specific

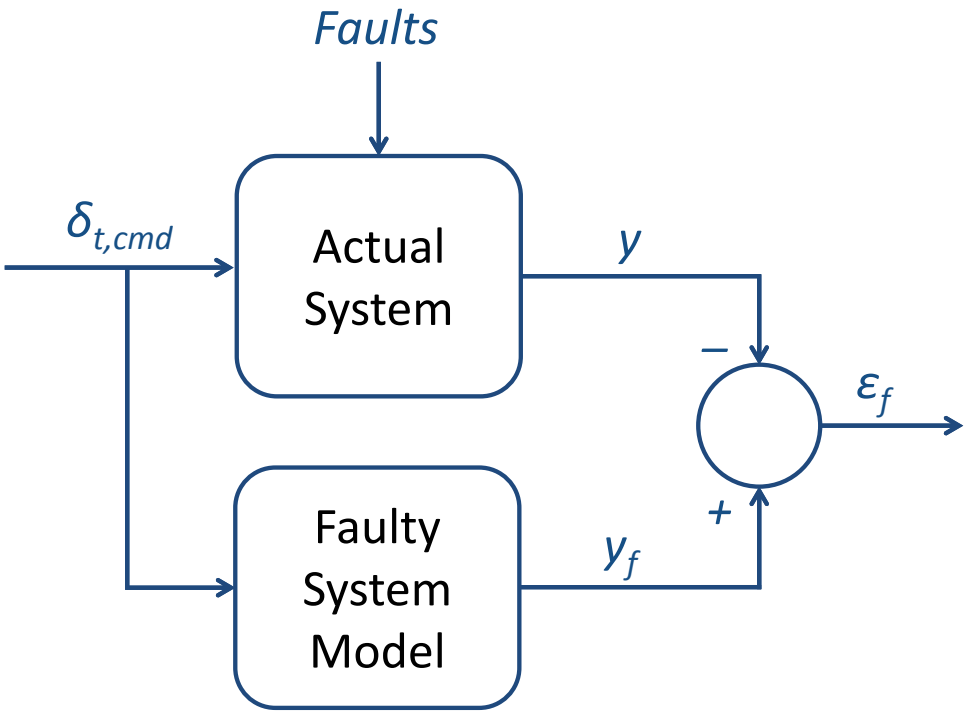

Figure 4-1: Fault-Specific Observer Residual Generation Block Diagram

fault implemented. A block diagram of the residual generation process is shown in Figure 4-1. As with the heading rate and motor residuals, each fault-specific observer residual is smoothed. The residuals are smoothed to ensure that they respond quickly to faults in order to minimize total isolation time. The cutoff frequency used by the smoothing filter is the same as the one used for the heading rate residual, since the fault-specific observers also use the LTI system model from Eqs. (2.4) and (2.5). See Appendix B for details on the filtering procedure.

Fault-specific observers are constructed for the broken left line, broken right line, and criss-crossed lines faults. First, consider the broken left line fault. A simple modification to the nominal observer in Eq. (3.3) is required to construct this observer. The broken left line fault-specific observer is,

$$\hat{x}_{bl}[k+1] = A\hat{x}_{bl}[k] + B\delta_{R,cmd}[k] + L(y[k] - \hat{y}_{bl}[k]) \tag{4.1}$$

$$\hat{y}_{bl}[k] = C\hat{x}_{bl}[k] \tag{4.2}$$

and the observer for the broken right line is similar,

$$\hat{x}_{br}[k+1] = A\hat{x}_{br}[k] - B\delta_{L,cmd}[k] + L(y[k] - \hat{y}_{br}[k]) \tag{4.3}$$

$$\hat{y}_{br}[k] = C\hat{x}_{br}[k] \tag{4.4}$$

where $\hat{x}_{bl}[k]$, $\hat{x}_{br}[k] \in \mathbb{R}^3$ are the state estimates for the broken left and right line observers, respectively, and $\hat{y}_{bl}[k]$, $\hat{y}_{br}[k] \in \mathbb{R}$ are the outputs from each observer. The only modification to Eq. (3.3) needed to build these fault-specific observers is to replace the differential toggle command $\delta_{t,cmd}$ with the individual commands to either the right or left motor, depending on the type of fault. The corresponding residual signals for these fault-specific observers are,

$$r_{bl}[k] = (\hat{y}_{bl}[k] - y[k])^2 \tag{4.5}$$

$$r_{br}[k] = (\hat{y}_{br}[k] - y[k])^2 \tag{4.6}$$

where $r_{bl}[k]$, $r_{br}[k] \in \mathbb{R}$ and $y[k]$ is the EKF-estimated heading rate of the system.

The observer for the criss-crossed lines fault is,

$$\hat{x}_{cc}[k+1] = A\hat{x}_{cc}[k] - B\delta_{t,cmd}[k] + L(y[k] - \hat{y}_{cc}[k]) \tag{4.7}$$

$$\hat{y}_{cc}[k] = C\hat{x}_{cc}[k] \tag{4.8}$$

where $\hat{x}_{cc}[k] \in \mathbb{R}^3$ is the state estimate for the criss-crossed lines observer, and $\hat{y}_{cc}[k] \in \mathbb{R}$ is the output from the observer. The only modification to Eq. (3.3) needed to create the observer is to reverse the effect of the input. The residual signal for the criss-crossed lines observer is,

$$r_{cc}[k] = (\hat{y}_{cc}[k] - y[k])^2 \tag{4.9}$$

where $r_{cc} \in \mathbb{R}$.

These residuals represent how well each observer models the current condition of the system, and should be small only when the fault that is modeled by the

observer is present in the actual system. These signals should be compared such that distinguishing between the three non-actuator faults is possible.

4.2.2 Residual Evaluation

The goal while evaluating fault-specific observers is to determine which of the faults under consideration (i.e., broken left line, broken right line, or criss-crossed lines) is the most likely to be present. This is accomplished by comparing the *relative size* of each residual signal. The relative size of a residual is its value after it has been scaled by a chosen factor. This scaling is performed to allow for comparable rates of successful isolation for broken line and criss-crossed lines faults. The residuals for the broken line faults and criss-crossed lines fault behave differently. For example, the criss-crossed lines residual tends to be smaller in the presence of a broken line fault than the broken line residuals are in the presence of a criss-crossed lines fault. The criss-crossed lines residual should be scaled up in order to be compared to the broken line residuals in such a way that both broken line and criss-crossed lines faults can be isolated effectively. The fault-specific observer residual with the smallest relative size, provided that the residual is not too large itself, informs the presence of the fault associated with that residual.

To improve isolation statistics and reduce the probability that a fault is isolated incorrectly, each residual signal is evaluated over a period of time, not instantaneously as is the case for the detection residual. For this work, a fault isolation is declared if the relative size of one residual is smaller than the relative sizes of the other two residuals for 10 consecutive seconds. This 10-second period is a design parameter that can be varied based on mission requirements. A shorter period allows for faster isolation, but results in a higher incidence of incorrect fault isolation.

The first step in evaluating the fault-specific observer residuals is to determine the scaling factor that should be applied to each residual such that a fair size comparison can be made. To do this, 1000 Monte Carlo simulations each are run for cases in which a criss-crossed lines fault or a broken line occur. All three fault-specific observer residuals are measured during each simulation. Since this isolation method

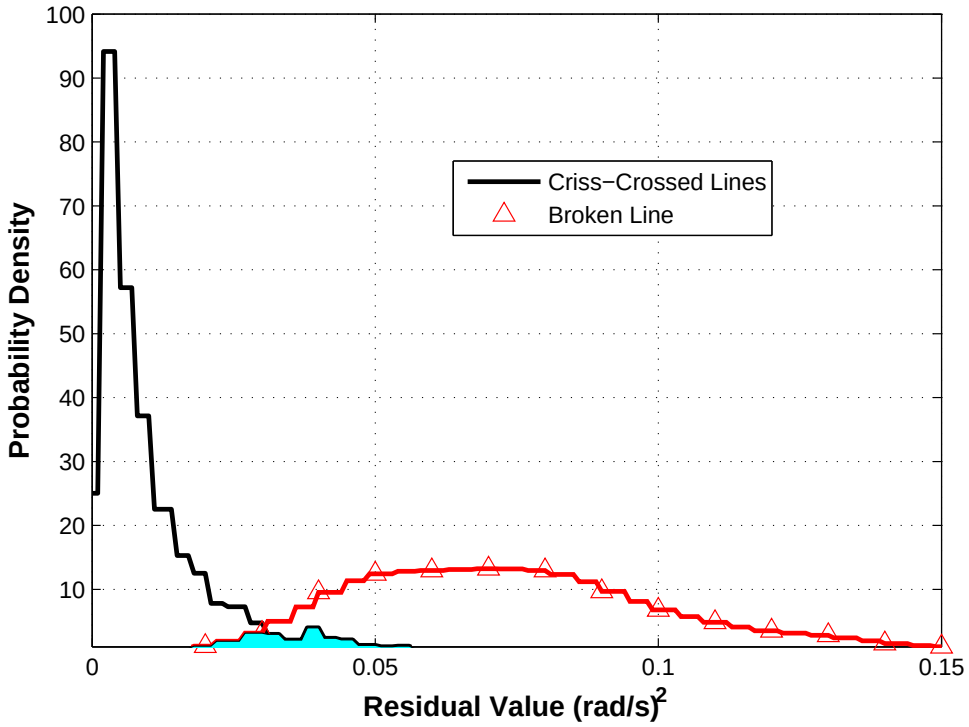

Figure 4-2: PDFs of Fault-Specific Observer Residuals: Criss-Crossed Lines Fault Present

evaluates the signal over a 10-second period, the data collected from each simulation is the largest value of each residual that is contained within a 10-second time window of residual values greater than or equal to the selection. This bounds the largest 10-second block of residual values that is expected during a flight.

Figure 4-2 shows empirical probability density functions (PDFs) of each residual signal in the presence of a criss-crossed lines fault. As expected, the criss-crossed lines residual is small for a large percentage of simulations. The left and right broken line residual signals have approximately the same statistics when a criss-crossed lines fault is present, so data from these residuals is indicated as one signal, the *broken line residual*. The broken line residual is large for a high percentage of flights in which a criss-crossed lines fault is present. Since there are only two signals to consider in this case, only the criss-crossed lines residual needs to be scaled to allow for relative size comparisons. Scaling this signal will stretch or compress the corresponding PDF and change the size of the criss-crossed lines signal relative to that of the broken line. The

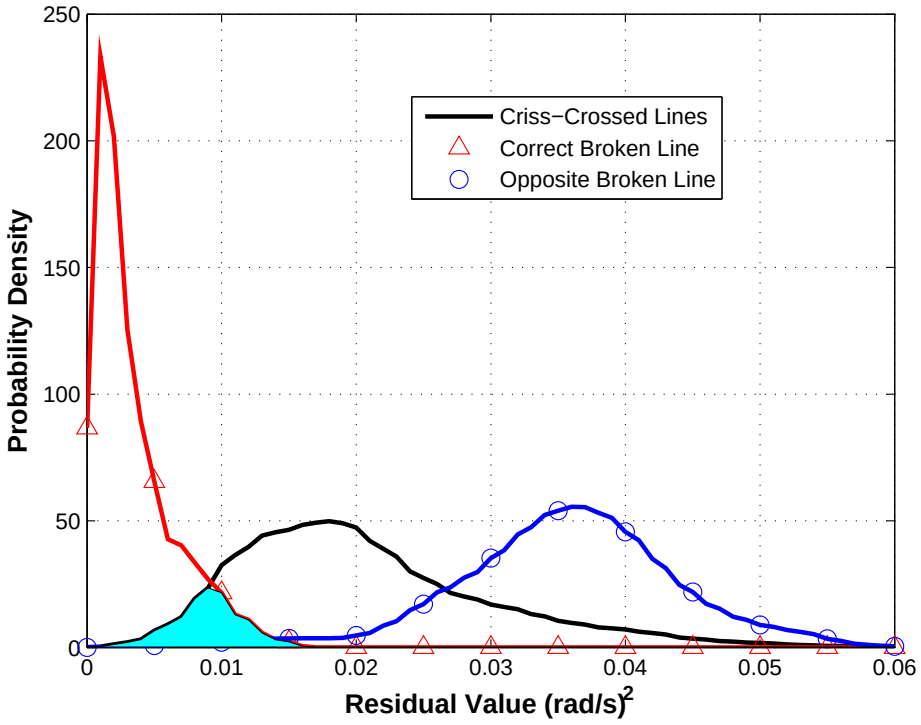

Figure 4-3: PDFs of Fault-Specific Observer Residuals: Broken Line Fault Present

area between these two PDFs, indicated as a light blue shaded area in Figure 4-2, is a measurement of how likely it is that the broken line residual is less than the criss-crossed lines residual when a criss-crossed lines fault is present. The smaller that area is, the lower the chance of a false isolation when a criss-crossed lines fault has occurred.

Figure 4-3 shows empirical PDFs of each residual signal in the presence of a broken line fault. Three PDFs are shown, one for the criss-crossed lines residual, one for the *correct broken line residual* (e.g., broken left line residual when a broken left line is present), and one for the *opposite broken line residual* (e.g., broken right line residual when a broken left line is present). Again, scaling the criss-crossed lines residual would stretch or compress the corresponding PDF, affecting the comparison of this signal and the correct broken line signal. Since both broken line residuals have the same behavior, only the criss-crossed lines residual must be scaled. The area between the correct broken line and criss-crossed lines PDFs is a measure of how likely it is

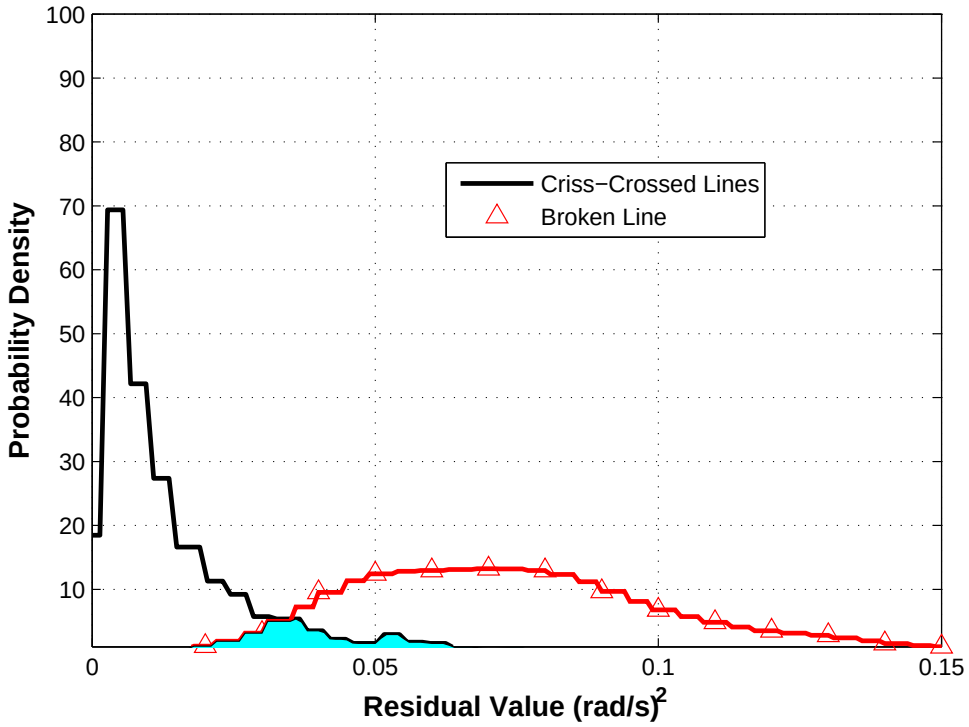

Figure 4-4: PDFs of Fault-Specific Observer Residuals: Criss-Crossed Lines Fault Present with Scaling Factor Implemented

that a criss-crossed lines fault is isolated even through a broken line has occurred. When this area is small, there is a lower chance of a false isolation when a broken line is present.

The optimized scaling factor for the criss-crossed lines residual should result in high rates of successful isolation when each of the three faults under consideration is present. Intuitively, scaling the criss-crossed lines residual down improves isolation of a criss-crossed lines fault, and scaling the criss-crossed lines residual up improves isolation of broken line cases. Finding a balance between these two extremes ensures high rates of successful isolation for all cases.

To choose an optimized scaling factor, minimize the area of intersection between PDFs of the residual signals, shown as a blue shaded areas in Figures 4-2 and 4-3. The smaller the area of intersection, the higher the likelihood that the residual corresponding to the fault that has occurred will be less than the other two residual signals. For optimization, minimize the weighted sum of the area between the criss-

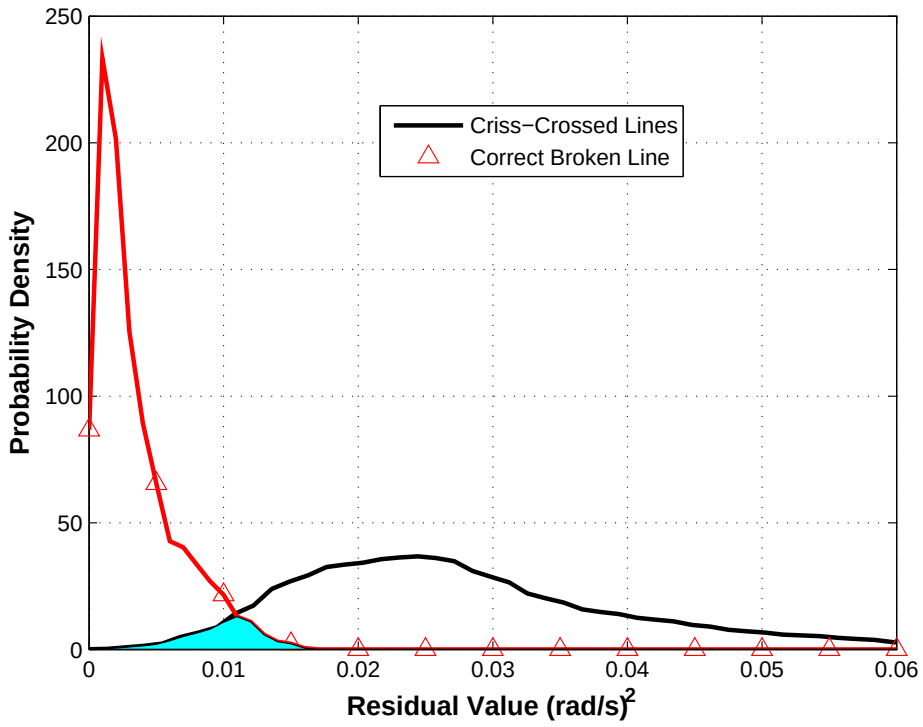

Figure 4-5: PDFs of Fault-Specific Observer Residuals: Broken Line Fault Present with Scaling Factor Implemented

crossed lines residual and broken line residual PDFs, given that a criss-crossed lines fault has occurred, and the area between the correct broken line residual PDF and the criss-crossed lines residual PDF, given that a broken line has occurred. Since neither broken line residual is scaled, the area between the correct broken line and opposite broken line PDFs will remain constant.

The scaling factor, T_s, that will be applied to the criss-crossed lines residual during the evaluation process, is chosen using an unconstrained minimization solver [30] that minimizes the following figure of merit:

$$FOM_{iso} = \frac{c_{cc}A_{cc} + c_{bl}A_{bl}}{c_{cc} + c_{bl}} \tag{4.10}$$

where c_{cc} and c_{bl} are weighting factors that determine which fault it is more important to isolate. A high value of c_{cc} relative to c_{bl} indicates that it is desirable to have a higher rate of successful isolation of criss-crossed lines faults with respect to broken

line faults. A_{cc} is the area between the scaled criss-crossed lines and broken line PDFs, given that a criss-crossed line has occurred, and A_{bl} is the area between the scaled criss-crossed lines and correct broken line PDFs, given that a broken line fault has occurred. Table 4.1 shows the optimized scaling factor that should be applied to the criss-crossed lines residual for isolation, given different weightings on isolation of criss-crossed lines or broken line faults.

Table 4.1: Optimized Scaling Values for Various FOMs

$\mathbf{c_{cc}}$	$\mathbf{c_{bl}}$	**Scaling Factor**	$\mathbf{A_{cc}}$	$\mathbf{A_{bl}}$	**FOM**
0.1	0.9	2.8330	0.3294	0.0314	0.0612
0.2	0.8	2.2780	0.2700	0.0407	0.0866
0.4	0.6	1.6240	0.1909	0.0740	0.1208
0.5	0.5	1.3570	0.1592	0.1000	0.1296
0.6	0.4	1.0720	0.1192	0.1479	0.1307
0.8	0.2	0.5520	0.0510	0.3347	0.1077
0.9	0.1	0.3750	0.0323	0.4661	0.0756

For $c_{cc} = c_{bl}$, the optimized scaling factor $T_s = 1.3570$. This scaling factor will be used in the remainder of this thesis, unless otherwise specified, to scale the criss-crossed lines residual for isolation. Figures 4-4 and 4-5 show PDFs when this factor is applied to the criss-crossed lines PDFs from Figures 4-2 and 4-3. The opposite broken line residual is omitted from Figure 4-5. Notice that the shaded area in Figure 4-4 is larger than that in Figure 4-2, while the shaded area in Figure 4-5 is smaller than that in Figure 4-3. This is due to the positive scaling of the criss-crossed lines residual. Given this scaling, a criss-crossed lines fault will be isolated if $T_s r_{cc}[k] < r_{bl}[k]$ and $T_s r_{cc}[k] < r_{br}[k]$ for 10 consecutive time steps. A broken left line fault is isolated if $r_{bl}[k] < T_s r_{cc}[k]$ and $r_{bl}[k] < r_{br}[k]$ for 10 consecutive time steps. A broken right line fault is isolated analogously to the broken left line case.

4.3 Preventing False Isolation

The scaling method described in Section 4.2.2 works well when it is certain that either a criss-crossed lines or broken line fault has occurred. However, consider a scenario in which there is a false detection alarm, or another fault is present. Then, all three fault-specific observer residuals can be large, indicating that is unlikely one of those faults has occurred, but isolation may still declare one of those faults. All that is required for isolation of a non-actuator fault is for the relative size of its corresponding residual to be less than the sizes of the other two residuals for 10 consecutive seconds. There should be some notion of when a fault-specific observer residual is too large; large enough that it is unlikely that the corresponding fault has occurred, regardless of the relative sizes of the other residuals. Thresholds are used to bound the likely size of each fault-specific observer residual. In order to develop these *limit thresholds* that assist in preventing false isolation of a non-actuator fault, large-scale Monte Carlo simulations are again used. Now, we are concerned with the size of each fault-specific observer residual in the presence of its corresponding fault only.

Figure 4-6 shows CDFs of the maximum value of the fault-specific observer residuals observed during each flight in the presence of their corresponding faults. The broken left and right line residuals behave the same, and their results are aggregated into a single broken line residual shown on the CDF. As expected, each residual is generally small when the fault associated with that residual has occurred. If the limit threshold for each fault case is chosen such that there is a low chance that the residual crosses the threshold when the corresponding fault is present, false isolations can be minimized while preserving almost all successful isolations.

Figure 4-6 shows that the broken line residual curve rises quickly while the criss-crossed lines curve tapers off as the cumulative probability approaches 1. The broken line limit threshold is set to 0.0160 $(\text{rad/sec})^2$; 100% of the data on the broken line curve is under this threshold. The criss-crossed lines limit threshold is chosen to be 0.0808 $(\text{rad/sec})^2$, which leaves 0.5% of data greater than that value.

These chosen limit thresholds serve as a final check to ensure that a fault is not

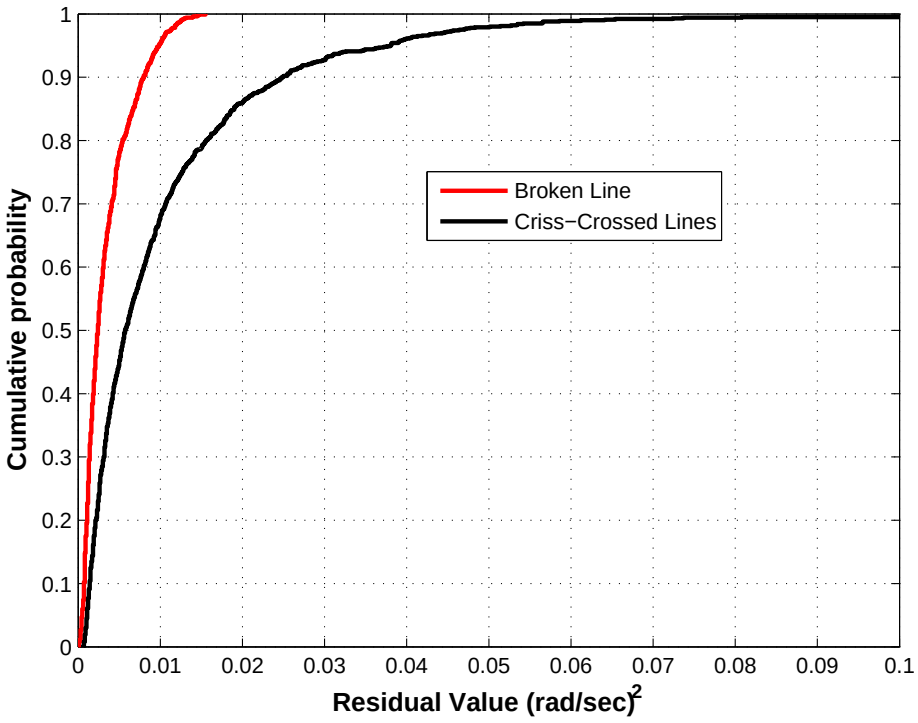

Figure 4-6: CDFs of Fault-Specific Observer Residuals

declared if the fault-specific observer residuals clearly indicate that the fault is not present. A non-actuator fault will be declared during isolation only if the relative size of the corresponding residual is smaller than the others for 10 seconds, and the residual has not crossed the limit threshold during that period. If the limit threshold is crossed, FDI is restarted. This will likely be the case if a fault that has not been modeled has occurred; a detection alarm will probably be raised but all of the fault-specific observers will likely be above the limit thresholds.

4.4 Full FDI Implementation

4.4.1 Overall Logic

Chapters 3 and 4 outline methods for both detection and isolation of faults for the parafoil and payload system. Figures 4-7 and 4-8 show an overview of the integrated process. Table 4.2 summarizes the thresholds and scaling factor chosen for detec-

tion and isolation that will be used for all future simulations and flight tests, unless otherwise indicated.

Table 4.2: FDI Parameters

Name	**Symbol**	**Value**
Detection Threshold	T_{det}	0.0160 (rad/sec)^2
Motor Residual Threshold	T_m	0.1390 m
Criss-Crossed Lines Residual Scaling Factor	T_s	1.357
Broken Line Limit Threshold	T_{bl}	0.0160 (rad/sec)^2
Criss-Crossed Lines Limit Threshold	T_{cc}	0.0808 (rad/sec)^2

The first step is fault detection. The heading rate residual signal is monitored throughout the entire flight. If, at any point, the smoothed residual signal rises above a predetermined threshold, a fault is declared. Two motor residual signals are also monitored during detection. If any one, or both, of these signals cross above the chosen motor threshold, a fault is declared. Once the alarm is raised, the algorithm progresses to the isolation method.

Isolation is performed by evaluating the motor residual signals used for detection as well as residuals generated from a bank of fault-specific observers. If the smoothed residual from either the left or right motor is above the predetermined threshold, a fault in the corresponding motor is declared. With the declaration of an actuator fault, the FDI process ends. Alternately, if the relative size of the smoothed residual signal associated with one of the fault-specific observers is smaller than the others for 10 consecutive seconds, and is smaller than its limit threshold, the algorithm declares that the fault corresponding to that particular observer has occurred. If a limit threshold is crossed or both motor residuals cross the threshold, FDI has failed and is restarted. In this case, it is likely that a fault that is not considered for FDI has occurred.

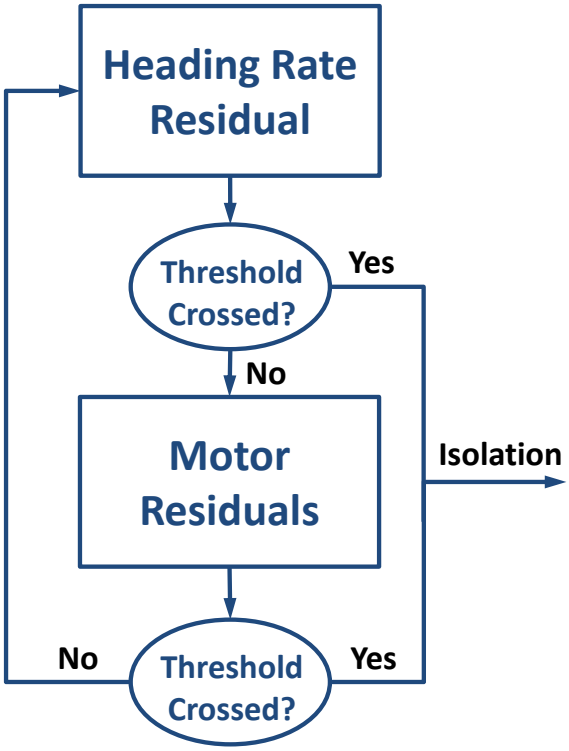

Figure 4-7: Fault Detection Logic

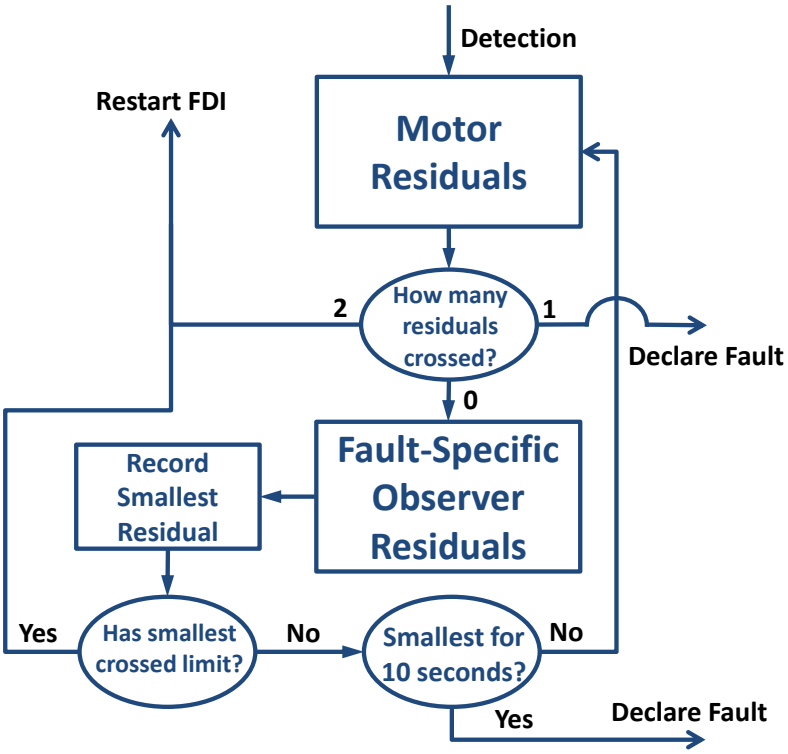

Figure 4-8: Fault Isolation Logic

4.4.2 Results

This section presents results of the full FDI implementation for three different faults: stuck motor, broken line, and criss-crossed lines. Using the thresholds and scaling factor specified in Table 4.2, the performance of the full FDI method is tested on 1000 Monte Carlo simulations each in which a fault in one of the three categories being considered occurs. 1000 Monte Carlo simulations are also performed in which a random fault occurs as well as an additional 1000 simulations for flights in which no fault occurs (i.e., healthy case). The performance of the FDI method is evaluated in terms of the probabilities of detecting and isolating faults successfully. Detection is evaluated by considering the probability of successful detection, $P(DET)$ as well as $P(FA)$ and $P(MD)$. Several probabilities are used to assess the effectiveness of the isolation method. The probability that the correct fault is isolated given successful detection is $P(ISO)$. The probabilities that, once detection occurs, the isolation phase is inconclusive or declares an incorrect fault are given by $P(INC)$ and $P(FI)$, respectively. Table 4.3 shows FDI results. The results show that both detection and isolation are highly successful for each fault case. Broken line faults are the most difficult to detect and isolate, but still have high detection and isolation rates of 95.8% and 98.12%, respectively. Rates of false alarm are very low for all cases.

Table 4.3: FDI Results

Case	**P(DET)**	**P(MD)**	**P(FA)**	**P(ISO)**	**P(INC)**	**P(FI)**
Healthy	N/A	N/A	0.0050	N/A	N/A	N/A
Random Fault	0.9730	0.0270	0.0040	0.9930	0.0050	0.0020
Criss-Crossed Lines	0.9880	0.0120	0.0000	0.9960	0.0040	0.0000
Broken Line	0.9580	0.0420	0.0030	0.9812	0.0115	0.0073
Stuck Motor	0.9780	0.0220	0.0050	1.0000	0.0000	0.0000

Detection and isolation statistics are useful for analyzing the success of FDI, but the time it takes to successfully isolate a fault must also be considered. If the time between the occurrence of a fault and the isolation of that fault is large, there may not be enough time to implement an effective recovery strategy. FDI should not only

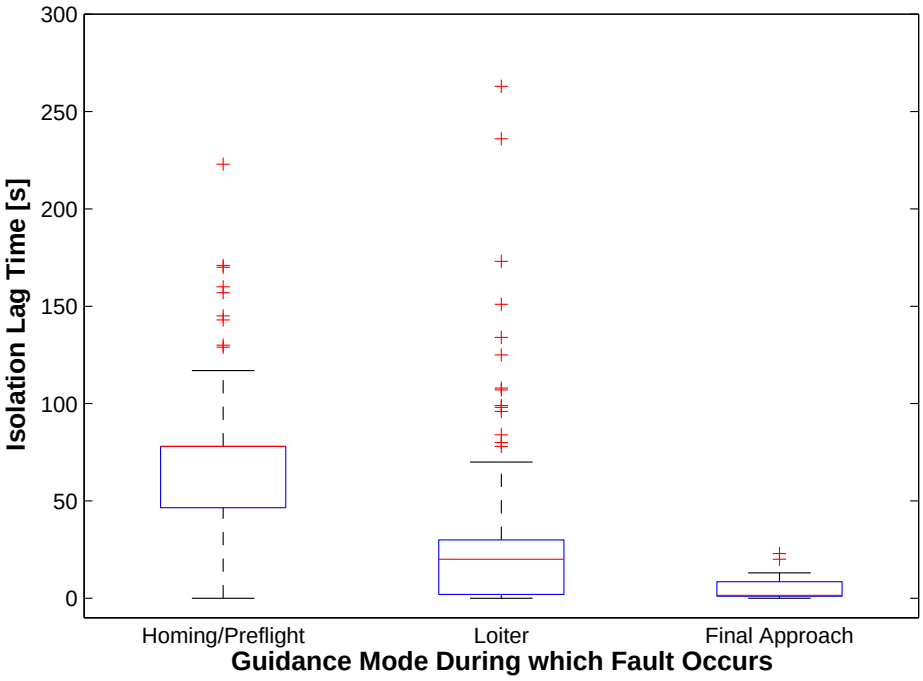

Figure 4-9: Isolation Lag Time Statistics for Various Flight Modes

successfully isolate faults in a high percentage of cases, isolation should occur quickly enough to allow for successful recovery.

The observability of a fault is often dependant upon the guidance mode during which the fault occurs. Consider, for example, a broken line fault that occurs during homing. If the system is already flying towards the target and no turn command is given, the effects of the broken line will not be apparent. If the fault occurs during energy management however, the changing heading rate commands will almost certainly make the presence of the broken line observable. Figure 4-9 shows a box and whisker plot of the *isolation lag time* statistics from 1000 Monte Carlo simulations during which a random fault occurs. Isolation lag time is defined as the amount of time between the occurrence of a fault and the successful isolation of that fault. The results are split up by mode; there are different data groups for faults that occur during homing or preflight, loiter, and final approach. One-third of the data come from flights in which a criss-crossed lines fault occurs during rigging, so most of the

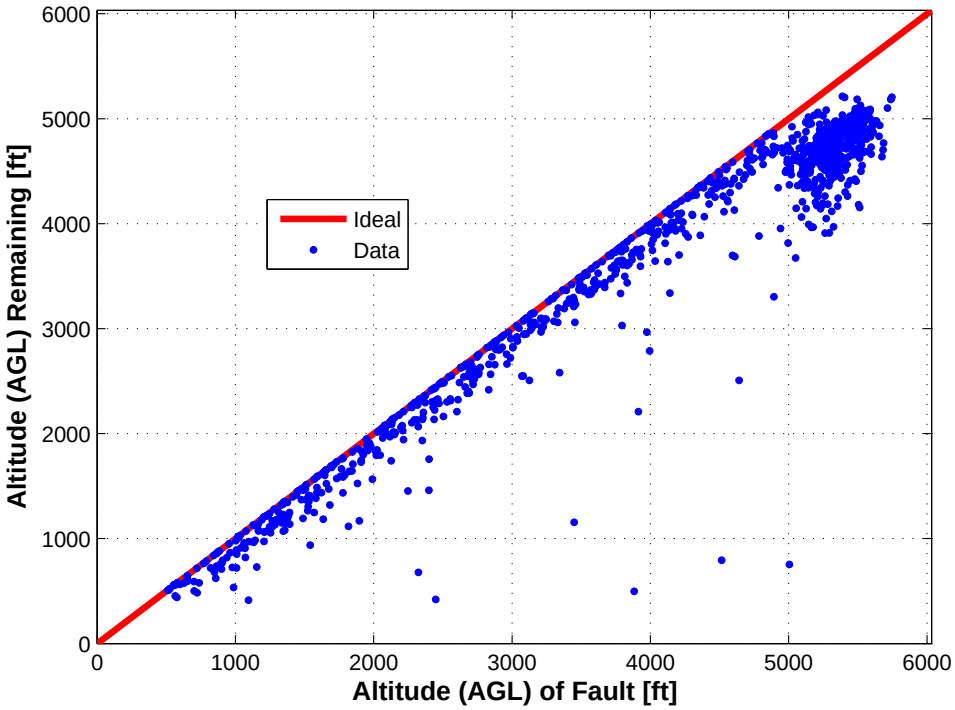

Figure 4-10: Altitude Remaining in Flight After Successful Isolation

data is in the Homing/Preflight section. The red horizontal line in each box indicates the median isolation lag time, while the boundaries of the blue box indicate the 25^{th} and 75^{th} percentiles. The black whiskers represent the remaining data not considered outliers, which are shown as red crosses. For clarity, some large outliers are left outside of the figure's axis range. Figure 4-9 shows that the median isolation lag time is 78 seconds if the fault occurs during homing as compared to 20 seconds if the fault occurs during loiter and 8.5 seconds if the fault occurs, and is isolated successfully, during final approach.

Longer isolation lag times are more acceptable if the fault occurs during homing because there is more time remaining in the flight to perform recovery as opposed to when faults occur during a later guidance mode. Figure 4-10 shows the altitude AGL with respect to the target after a fault has been successfully isolated versus the altitude at which the fault occurs. The best possible isolation would be instantaneous (i.e., data on the red line). Results in Figure 4-10 show that most flights result in

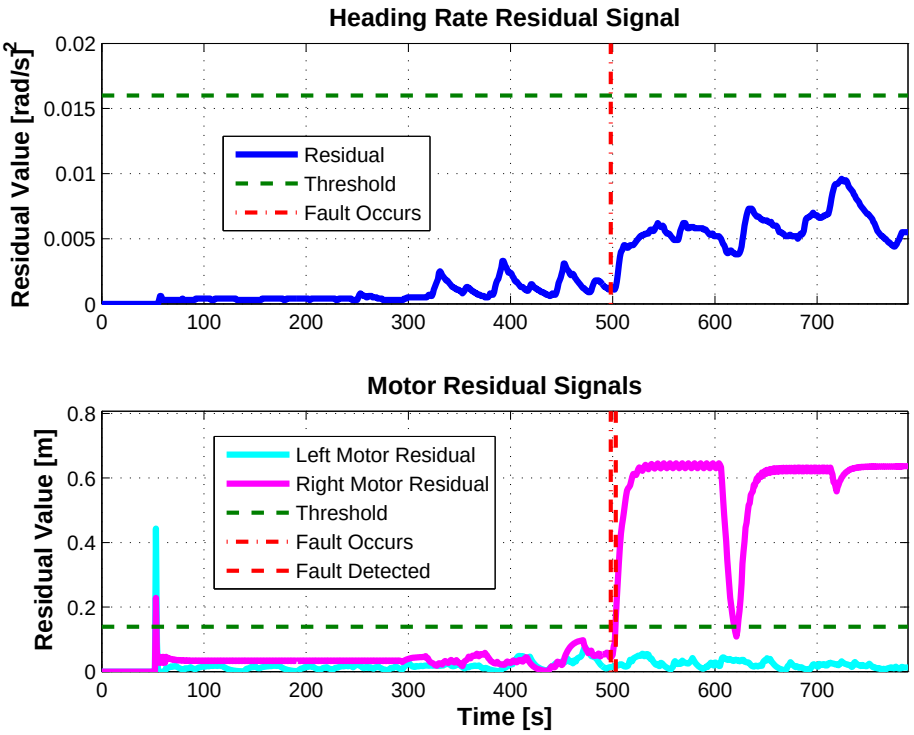

Figure 4-11: Heading Rate and Motor Residuals: Stuck Right Motor FDI

a quick isolation, and leave a large portion of the flight during which recovery can be implemented. There is a large collection of data for faults that occur at a high altitude; these correspond to criss-crossed lines faults which always occur pre-flight.

Figure 4-11 shows residual data from a full FDI implementation on a simulation in which a stuck right motor at a deflection of 0.57 occurs 498 seconds into the flight. This fault is both detected and isolated using the motor residual signal. Detection occurs 5 seconds after the fault occurs when the right motor residual signal crosses the threshold. Isolation occurs during the same time step, finishing the FDI process (see Figure 4-8). Evaluation of fault-specific observers is not necessary because one motor residual signal has already crossed its corresponding threshold. A stuck right motor is now declared. The stuck motor value is recorded as 0.57 by measuring the motor deflection at the time of isolation.

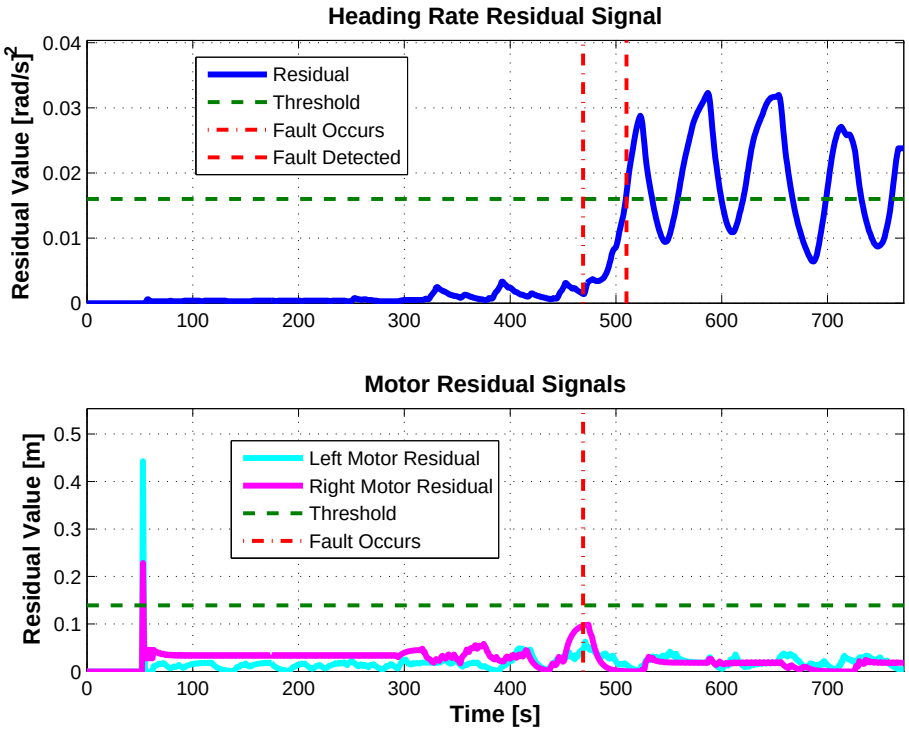

Figure 4-12: Heading Rate and Motor Residuals: Broken Left Line FDI

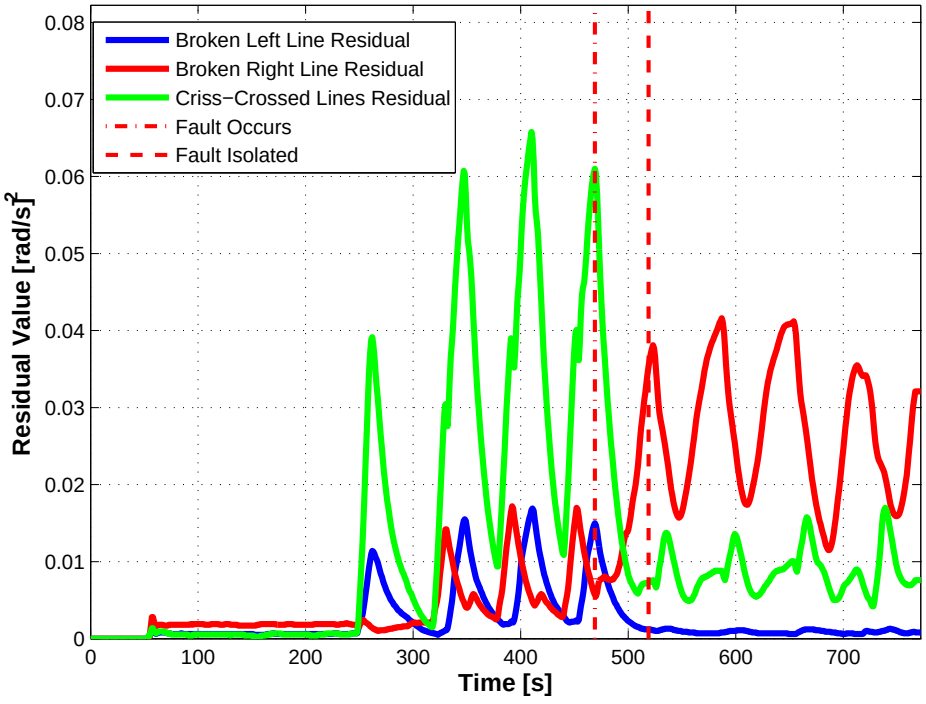

Figure 4-13: Fault Specific Observer Residuals: Broken Left Line FDI

The next example involves a non-actuator fault; a broken left line occurs 469 seconds into this simulation. Figure 4-12 shows the heading rate and motor residuals for this flight. Fault detection occurs when the heading rate residual signal crosses the threshold 41 seconds after the fault occurs. The motor residuals remain under the threshold for the entire flight, so isolation does not declare a motor fault. Figure 4-13 shows the relative size of each fault specific observer residual over the duration of the flight. The residual for the criss-crossed lines fault is scaled according to the scaling factor from Table 4.2. Once the fault occurs, the broken right line and criss-crossed lines residuals grow large, while the broken left line residual decays. This indicates that a broken left line fault is likely present. After the broken left line residual remains smaller than the other two signals for 10 time steps (i.e., 10 seconds), isolation concludes and a broken left line fault is declared. In this example, isolation occurs 50 seconds after the fault occurs. At this point, information about the fault is passed on to recovery, described in Chapter 5, during which an appropriate recovery strategy is chosen to mitigate the effects of the fault declared during isolation.

Chapter 5

Recovery

The guidance modes described in Section 2.3 perform well for healthy flights. The presence of a fault, however, can render these existing guidance strategies ineffective. Fault-dependent modifications to the nominal guidance strategy are made that are designed to deal with each fault case individually. This chapter describes new guidance strategies that allow the system to recover desired performance even in the presence of each of the faults considered in this work.

5.1 Criss-Crossed Lines

The effect of a criss-crossed lines fault is to generate the opposite response to what is expected for a given differential toggle or heading rate command. For example, a heading rate command of 0.5 will in theory yield a heading rate response of -0.5 when a criss-crossed lines fault has occurred. As such, only a simple modification to the control law is required to recover nominal performance: reverse the differential toggle command as shown in Eq. (5.1),

$$\delta_{t,cmd,cc} = -\delta_{t,cmd} \tag{5.1}$$

where $\delta_{t,cmd}$ is the nominal differential toggle command given by control and $\delta_{t,cmd,cc}$ is the actual differential toggle command sent to the motors when the criss-crossed

lines recovery strategy is implemented. No other modifications to guidance or control are necessary.

5.2 Broken Line

For the criss-crossed lines fault, both motors and control lines are still functional. This is not the case for a broken line fault, in which all turning ability in one direction is lost. This loss of control authority affects the performance of each nominal guidance mode, which assume that the system is able to turn in both directions.

A specific concern is the energy management mode, which requires frequent left and right turns. As a result, a new mode during loiter to replace energy management must be used when a broken line has occurred. Since the goal of loitering is to gradually lose altitude while remaining close to the desired landing target [8], a natural choice for a loiter mode that only requires turns in one direction is a circular horizontal trajectory [26]. Instead of tracking a figure-eight as dictated by energy management, this simple mode requires tracking a circle of a specified radius that is centered at the desired landing location. The parafoil travels clockwise around the circle when a broken left line has occurred, and counter-clockwise in the broken right line case. Though some of the benefits of the figure-eight trajectory are lost [21], such as mitigation of high-wind effects, tracking a circle provides a simple, yet effective solution to the problem presented when a broken line fault occurs.

For the final approach mode BLG is still used, but with an intuitive modification. The existing restriction of heading rate commands used in BLG, which ensures that the parafoil turns at a rate much slower than the system bandwidth, should be changed to reflect the loss of control caused by a broken line. For this fault case, heading rate commands are restricted to,

$$\dot{\psi}_{cmd} \in \left[0, \quad \dot{\psi}_{lim}\right] \tag{5.2}$$

$$\dot{\psi}_{cmd} \in \left[-\dot{\psi}_{lim}, \quad 0\right] \tag{5.3}$$

during BLG, where Eq. (5.2) is the limit imposed when a broken left line is present, and Eq. (5.3) is the limit when there is a broken right line.

5.2.1 Track and Loop Guidance

Although the nominal homing mode and a circular loiter mode trajectory should work well for a system with restricted turning capabilities, in practice a small deviation from the desired trajectory due to wind and other uncertainty can put the parafoil in a position from which it cannot recover using the existing guidance and control method. If, for example, during homing a broken right line occurs and the desired homing target is 30 degrees to the right, the system will normally command a right turn, an action it is unable to accomplish due to the line break. Instead, executing a left turn through 330 degrees until the parafoil approaches the desired heading is a solution to this issue that allows for heading command tracking even in the presence of a broken right line. This is an example of the logic behind a new modification to the nominal guidance strategy, track and loop guidance.

The track and loop guidance strategy is designed to alternate between two submodes that run in support of the current guidance mode (e.g., homing mode). The first submode, called tracking, guides the system to a desired heading. However, the system tracks the heading command with a bias to prevent an overshoot of the heading target. A smaller heading rate command is applied when the tracking submode is active, compared to the command that would have been given under the nominal guidance strategy. When the system approaches the desired heading, the heading rate command is set to zero to avoid overshooting the target.

Despite this bias, it is still possible that the system will deviate too far from its desired heading in the direction from which it cannot turn back (e.g., the parafoil is to the right of the target and has a broken left line). If this occurs, the system enters the looping submode, which commands the parafoil to execute a turn at the maximum allowable heading rate in order to bring the parafoil back to the desired heading. When the parafoil approaches its intended target again, the tracking submode is resumed.

This guidance method is formally implemented using four parameters relating to the heading angle error measured by the controller: heading error bias ψ_{bias}, small heading error ψ_{small}, loop start error ψ_{start}, and loop end error ψ_{end}. All of these parameters are greater than zero. These angles are shown in Figure 5-1 along with the heading angle command ψ_{cmd} from guidance as well as the heading angle of the parafoil ψ, estimated by the EKF. The figure shows the parameters for a broken left line case where the parafoil can only execute right turns. The parameters would be reflected about the North axis for a broken right line fault.

The heading error bias ψ_{bias} is a measure of how the heading rate of the system should be scaled down to prevent an overshoot past the heading command. The small heading error ψ_{small} determines the amount of heading error below which it is not necessary to continue to turn towards the commanded heading. The loop start and end errors ψ_{start} and ψ_{end} determine deviations from the commanded heading that trigger the beginning and end of the looping submode of the track and loop strategy.

The utilization of these four parameters is described in terms of the heading error,

$$\psi_e = \psi_{cmd} - \psi \tag{5.4}$$

that measures how far away the system is from the desired heading. This heading error is nominally used in the Proportional-Derivative (PD) controller,

$$\dot{\psi}_{cmd} = K_p \psi_e + K_d \dot{\psi}_e \tag{5.5}$$

to determine the heading rate command that is sent to control. In Eq. (5.5), $\dot{\psi}_{cmd}$ is the heading rate command, K_p and K_d are the proportional and derivative gains, respectively, of the PD controller, and $\dot{\psi}_e$ is the derivative of the heading error.

The tracking submode passes a new value, called the effective heading error ψ_{eff}, in place of ψ_e to the PD controller that scales down the heading rate command. The effective heading error is determined from the nominal heading error of the system as well as the error bias and small error values. This is essentially a way to change the gain K_p to reflect the change in conditions caused by the fault. The modified PD

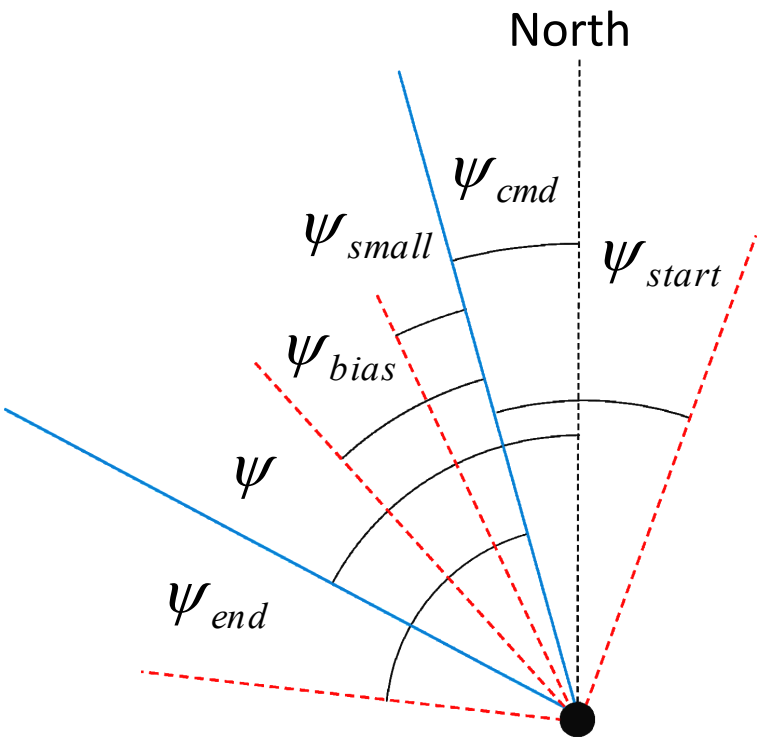

Figure 5-1: Track and Loop Parameters in Parafoil Coordinate Frame

controller is given as,

$$\dot{\psi}_{cmd} = K_p \psi_{eff} + K_d \dot{\psi}_e \tag{5.6}$$

where ψ_{eff} replaces ψ_e and reduces the resulting heading rate command. The value of the nominal heading error ψ_e determines the method used for calculating the effective heading error ψ_{eff}. Consider the case depicted in Figure 5-1 where a broken left line has occurred. As discussed at the beginning of this section, this error should be biased to a smaller value so the system does not turn past the target. There are three regions of the tracking submode in which the effective heading error is chosen differently. For a broken left line fault, the effective heading error is,

$$\psi_{eff} = \begin{cases} \psi_e - (\psi_{bias} - \psi_{small}), & \text{if } \psi_e \geq \psi_{bias} & \text{Region 1} \\ \psi_{small}, & \text{if } \psi_e < \psi_{bias} \bigcap \psi_e \geq \psi_{small} & \text{Region 2} \\ 0, & \text{if } \psi_e < \psi_{small} & \text{Region 3} \end{cases} \quad (5.7)$$

during the tracking submode. In Region 1, the heading error is reduced from the nominal value to slow the turn rate towards the target. Figure 5-1 depicts a system in Region 1 of the tracking submode. In Region 2, the effective heading error is set to a constant, small value so that the system can slowly approach the target. When the system is in Region 3, the effective heading error is set to zero to prevent overshoot. At this point, deviations from the target heading may be due to wind changes or other system perturbations. If $\psi_e < -\psi_{start}$, the system has overshot the desired heading target by a large amount and the system enters the looping submode. In this submode, the heading rate command is directly applied as,

$$\dot{\psi}_{cmd} = \dot{\psi}_{max} \quad (5.8)$$

where $\dot{\psi}_{max}$ is the maximum allowed heading rate command. Once the system approaches the target heading again (i.e., $\psi_e < \psi_{end}$) the tracking submode is resumed in Region 1. Figure 5-2 shows the implementation of the track and loop guidance logic for the broken left line case.

Alternately, the broken right line case requires a bias in the opposite direction and therefore, requires different computations. The effective heading error is given as,

$$\psi_{eff} = \begin{cases} \psi_e + (\psi_{bias} - \psi_{small}), & \text{if } \psi_e \leq -\psi_{bias} & \text{Region 1} \\ -\psi_{small}, & \text{if } \psi_e > -\psi_{bias} \bigcap \psi_e \leq -\psi_{small} & \text{Region 2} \\ 0, & \text{if } \psi_e > -\psi_{small} & \text{Region 3} \end{cases} \quad (5.9)$$

for each region of the tracking submode. If $\psi_e > \psi_{start}$, the system enters the looping

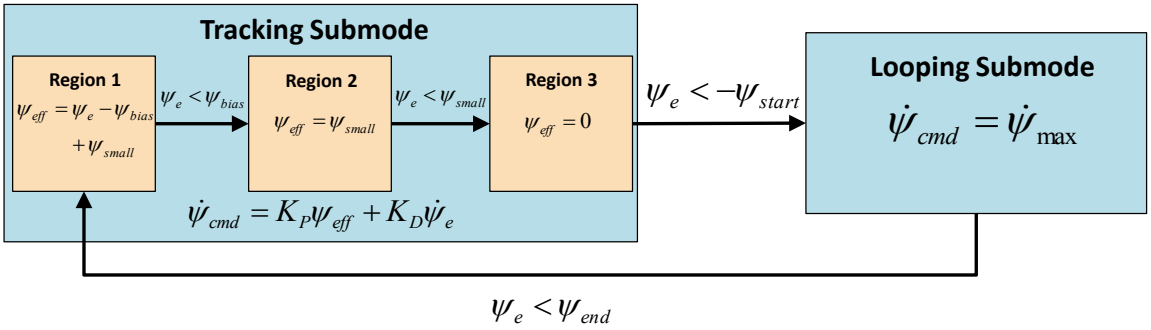

Figure 5-2: Track and Loop Guidance Logic for Broken Left Line Recovery

submode and the heading rate command is,

$$\dot{\psi}_{cmd} = -\dot{\psi}_{max} \tag{5.10}$$

which commands a left turn until the heading target is approached again (i.e., $\psi_e > -\psi_{end}$), at which point the tracking submode is resumed.

Table 5.1: Track and Loop Guidance Parameters

Parameter Name	**Symbol**	**Value (deg)**
Heading Error Bias	ψ_{bias}	30
Small Heading Error	ψ_{small}	5
Loop Start Error	ψ_{start}	30
Loop End Error	ψ_{end}	90

The four parameters used in track and loop guidance are selected using a numerical parametric study that utilize Monte Carlo simulations to maximize the performance (i.e., minimize miss distance). This process is described in Appendix C. Table 5.1 summarizes the four parameters chosen for track and loop guidance. These parameters are used in all simulations in this work that employ fault recovery strategies and use the track and loop guidance modification. By changing the loiter mode of the system from energy management to a circular trajectory and introducing the track and loop submode modification, the nominal guidance strategy is modified so that recovery from a broken line fault is possible.

5.3 Stuck Motor

A stuck motor fault occurs when one of the motors on the AGU freezes at a particular location, forcing the line to be stuck at a single deflection for the remainder of the flight. This limits the turning authority in each direction, depending on the position of the stuck motor. Consider a stuck left motor fault. Qualitatively, a stuck left motor at a small value severely limits the turning ability to the left, and a stuck left motor at a large value severely limits turning ability to the right. That is, a stuck left motor at a small value is similar to a broken left line fault, and a stuck left motor at a large value is similar to a broken right line fault. An analogous situation is true for the stuck right motor fault. The recovery strategy for stuck motor faults exploits this property. Values not in either one of these categories (i.e., stuck motor at a moderate value) should be able to use the nominal guidance strategy, as reasonable turning authority in both directions is still available, albeit with more limitations than the healthy case. Determining whether a particular stuck motor value should be considered small, moderate, or large is important so that the best recovery strategy is implemented.

For all stuck motor cases, one modification to control is made that exploits the knowledge of the stuck motor position. In the healthy case, the differential toggle command is usually distributed to one motor only. For example, a differential toggle command of -0.3 is normally set as a deflection of 0.3 on the left motor and 0.0 on the right motor. For the stuck motor case, only one motor can be actuated. The differential toggle command should be applied differently in this case. The command modifications are given as,

$$\delta_{R,cmd} = \delta_{L,stuck} + \delta_{t,cmd} \tag{5.11}$$

$$\delta_{L,cmd} = \delta_{R,stuck} - \delta_{t,cmd} \tag{5.12}$$

where $\delta_{t,cmd}$ is the differential toggle command, $\delta_{R,stuck}$ and $\delta_{L,stuck}$ are right and left stuck motor positions, respectively, and $\delta_{R,cmd}$ and $\delta_{L,cmd}$ are the resulting right and

left motor commands, respectively, after this modification is made. Equation (5.11) is used when a stuck left motor fault has occurred, and Eq. (5.12) is used for a stuck right motor. This modification ensures that the desired differential toggle is still achieved, when possible, in the presence of a stuck motor. However, Eqs. (5.11) and (5.12) are subject to the motor saturation limits, and some differential toggle commands will not be attainable for certain stuck motor cases.

The data in Figure 5-3, which is obtained from 1000 Monte Carlo simulations in which a randomly selected stuck motor fault occurs (e.g., stuck right motor at 0.35), shows normalized miss distance versus the stuck motor position for each simulation run. The only recovery strategy the system employs during these simulations is to change the motor commands as described in Eqs. (5.11) and (5.12) based on the knowledge of the stuck motor position, obtained from motor measurements, and the differential toggle command given by control. This data is used to set boundaries for what should be considered a small or large stuck motor position. It is clear that a high number of stuck motor positions in the middle of the actuation range (i.e., moderate stuck motor values) still perform well even with no modifications to guidance. However, results are much worse when the motor is stuck at a small or large value.

These stuck motor values should be grouped into three categories (i.e., small, moderate, or large), as indicated at the beginning of this section. This process is started by grouping stuck motor positions into 20 bins that collectively span the range of the motors. For example, bin 1 contains stuck motor values from 0 to 0.05, bin 10 contains stuck motor values from 0.45 to 0.5, and bin 20 contains stuck motor values from 0.95 to 1.0. The mean normalized miss distance for each bin is computed, based on simulation results that generated the data in Figure 5-3. These normalized miss distances are shown in Figure 5-4. If a bin has a mean miss distance greater than 10 times the median miss for healthy flights (i.e., normalized miss distance greater than 10), the stuck motor values within that bin are considered severe (i.e., small or large depending on the value). Figure 5-4 indicates that faults that are within bins 1-4 (i.e., stuck motor value of 0.0-0.2) should be considered small stuck motor faults,

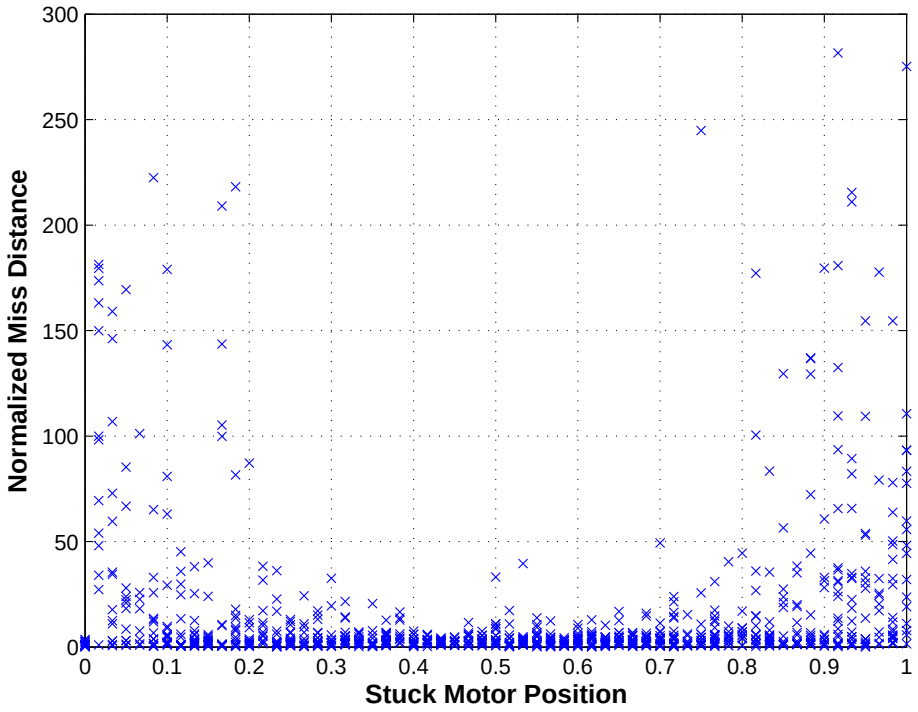

Figure 5-3: Normalized Miss Distances for Different Stuck Motor Positions: Control Modification Only

and faults within bins 17-20 (i.e., stuck motor value of 0.8-1.0) should be considered large. Faults within the remaining bins (i.e., stuck motor value of 0.2-0.8) should be considered moderate. The large cutoff does exclude bin 15, which lies slightly above the cutoff line. This bin is omitted from the large classification to retain continuity in the groupings. These three categories are used during recovery to determine which modified guidance strategy to use. A stuck motor at a small value uses the same guidance strategy as a broken line on the side on which the motor is stuck. A stuck motor at a large value is treated as a broken line on the side opposite of the one on which the motor is stuck. The nominal guidance strategy is used for a stuck motor at a moderate value. All stuck motor faults use the adjustment to the motor toggle command shown in Eqs. (5.11) and (5.12). Figure 5-5 shows normalized miss distance versus the stuck motor value for a set of Monte Carlo simulations, similarly to Figure 5-3. However, Figure 5-5 shows results when the full recovery strategy is implemented. The trend of the data is relatively flat, indicating that the severe values

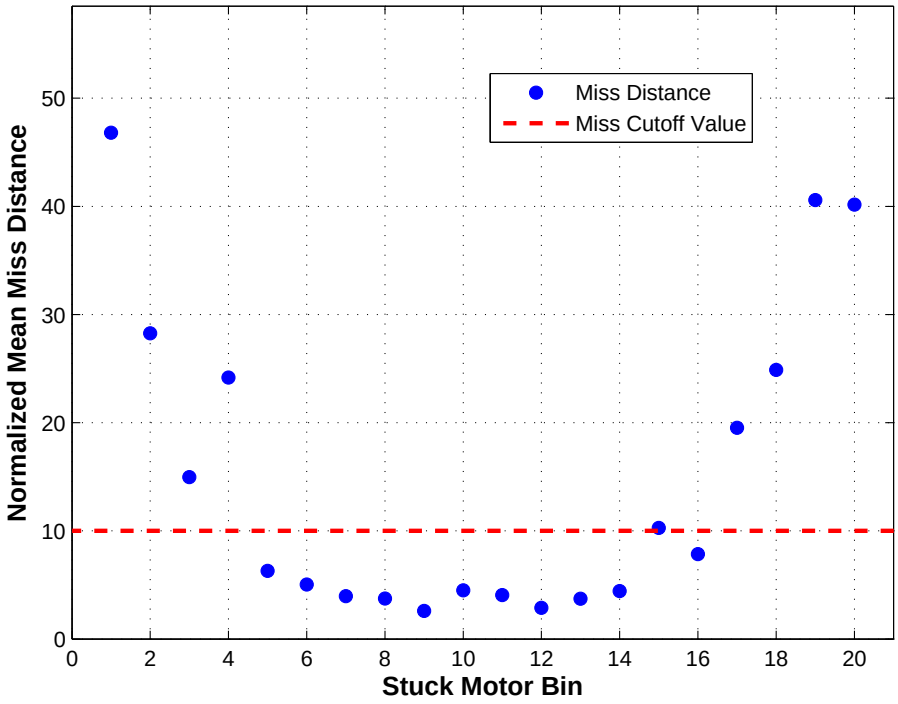

Figure 5-4: Mean Normalized Miss Distances for Stuck Motor Value Bins

are chosen appropriately.

BLG is used for final approach when there is a stuck motor fault. As with the broken line case, the restriction on turning ability must be added as a constraint to the BLG algorithm to find the optimal terminal trajectory given the fault. The restriction in this case is based on the amount of turning authority remaining and is given as,

$$\dot{\psi}_{cmd} \in \left[-\delta_{L,stuck}\dot{\psi}_{lim}, \quad (1-\delta_{L,stuck})\dot{\psi}_{lim}\right] \tag{5.13}$$

$$\dot{\psi}_{cmd} \in \left[-(1-\delta_{R,stuck})\dot{\psi}_{lim}, \quad \delta_{R,stuck}\dot{\psi}_{lim}\right] \tag{5.14}$$

where Eq. (5.13) is the limitation imposed when a stuck left motor fault is present, and Eq. (5.14) is the limit for the stuck right motor case.

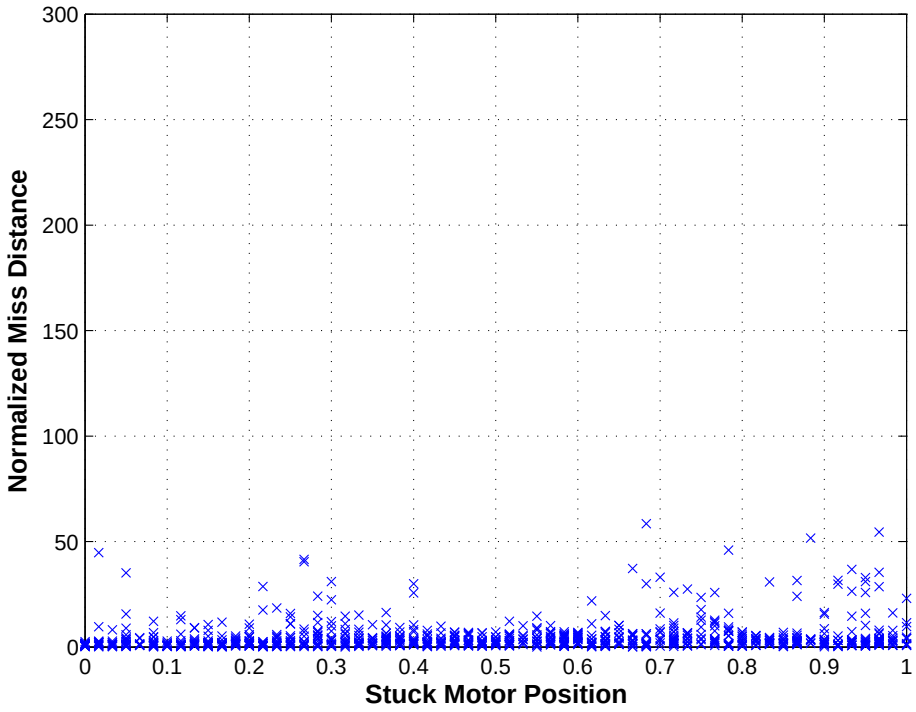

Figure 5-5: Normalized Miss Distances for Different Stuck Motor Positions: Full Recovery Implemented

5.4 Preventing False Isolation

5.4.1 Effect on Performance

The implications of a false isolation are clear now that recovery strategies have been developed for each fault. If a fault is isolated incorrectly, or a fault is isolated when the system is in a healthy state, a guidance strategy will be implemented that may not perform well given the true system state. A Monte Carlo simulation study is used to analyze the degradation in performance resulting from a false isolation. Sets of simulations with 1000 Monte Carlo trials each are run in which a false isolation occurs 100 seconds into the flight. In each of these cases there is either no fault, or a fault is present that is not correctly identified.

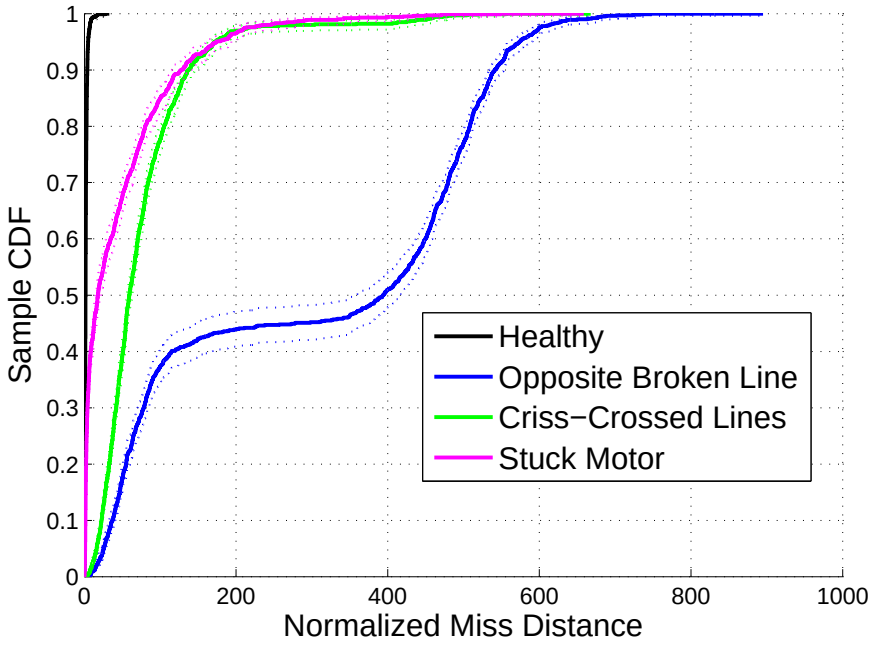

Figure 5-6: Miss Distance CDFs with 95% Confidence Interval: Broken Line False Isolation

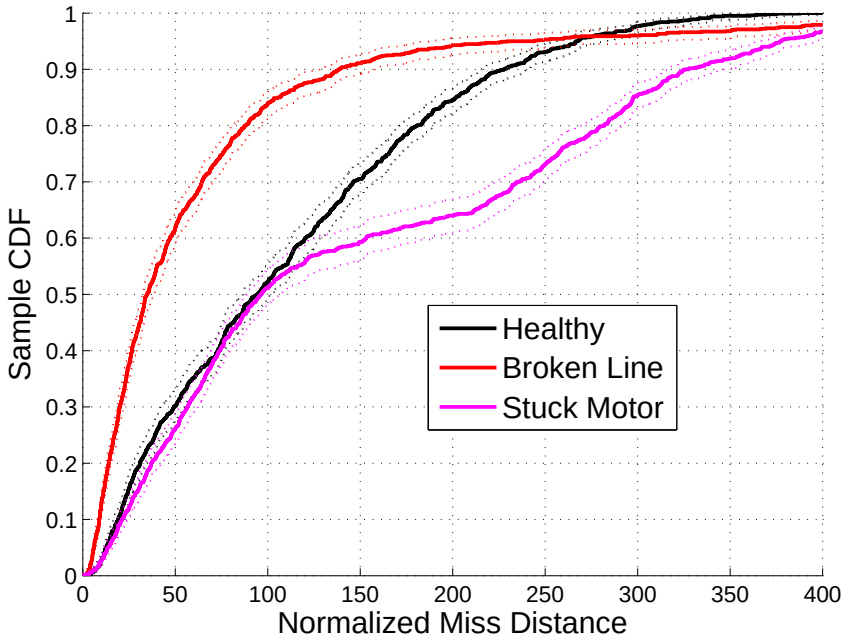

Figure 5-7: Miss Distance CDFs with 95% Confidence Interval: Criss-Crossed Lines False Isolation

Figure 5-6 shows miss distance CDFs of simulations in which a broken line fault is isolated 100 seconds into the flight. The legend indicates which fault is actually present. The *opposite broken line* curve indicates that a broken left line occurs but a broken right line is isolated, or vice versa. The actual fault occurs immediately upon canopy deployment in every case. Notice that, while flights in which a fault occurs perform poorly, the healthy flight still has good performance when a broken line fault is isolated. In fact, the median miss is only 1.5. The broken line recovery strategy assumes that one actuator is completely lost. Implementing this strategy on a healthy flight limits the system's capabilities unnecessarily, but does not negatively impact performance in any other way.

Figure 5-7 shows miss distance CDFs of flights in which a criss-crossed lines fault is isolated 100 seconds into the flight. The legend indicates which fault is actually present. The actual fault occurs immediately upon canopy deployment in every case. Since the recovery strategy for a criss-crossed lines fault is to reverse the differential toggle command, all false isolation cases perform poorly. Identifying when false isolations have occurred, especially in the healthy case, may prevent large miss distances and mission failure. The CDFs in Figures 5-6 and 5-7 indicate that false isolations have a large negative impact on system performance.

5.4.2 Post-Isolation Analysis

It is possible to limit the impact of a false isolation by detecting when one has occurred and restarting the FDIR process. Detection of this condition will focus on the behavior of the fault-specific observer residuals. If, for example, a broken left line is declared by isolation, but the broken left line residual is large, it is unlikely that a broken left line fault is actually present. *Post-isolation thresholds* are designed for each fault-specific observer to attempt to correctly declare when a false isolation has occurred. Using the set of Monte Carlo runs performed in Section 5.4.1, data from the fault-specific observers are collected. This data is the maximum value of the fault-specific observer residual that is measured from 15 seconds after isolation occurs to the end of the flight. The 15-second waiting period is implemented to en-

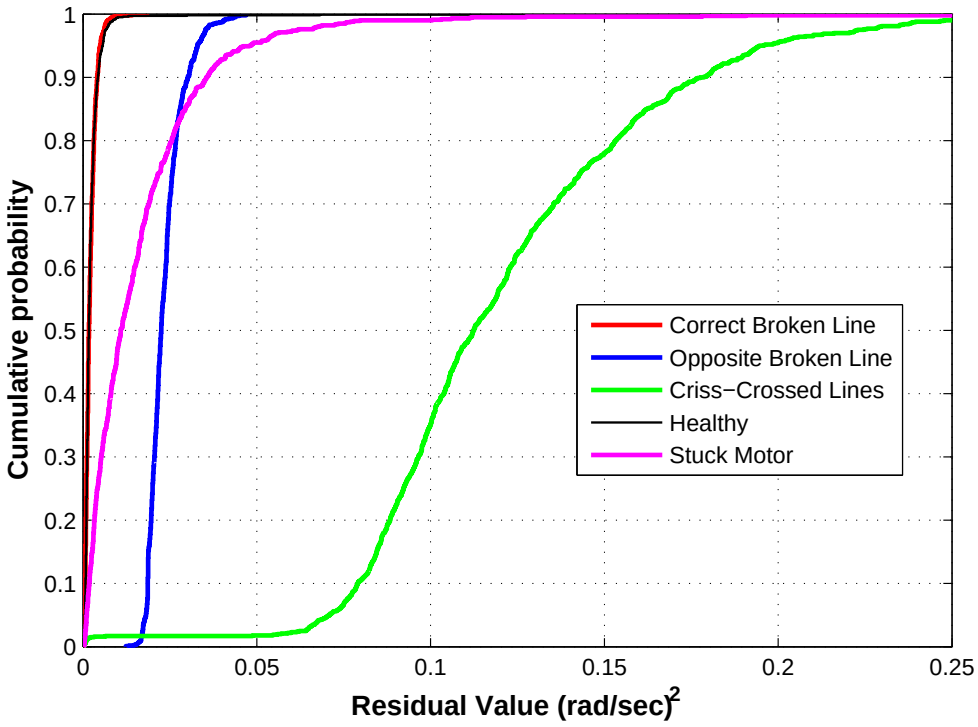

Figure 5-8: Post-Isolation Broken Line Residual Data: Broken Line Isolated

sure that the residuals have sufficient time to settle after the implementation of the new recovery strategy. If isolation is successful, the maximum value of the residual corresponding to the isolated fault should be small. Otherwise, the residual should be large, as the recovery strategy that is implemented is likely not suitable to mitigate the effects of the true fault that is present. Once the thresholds are chosen, they will be used to identify a false isolation. If the fault-specific observer residual corresponding to the fault that was isolated crosses above its post-isolation threshold, the nominal guidance strategy is again used and FDIR is restarted.

Figure 5-8 shows CDFs of data from simulations in which a broken line is isolated 100 seconds into the flight. However, some of the data are from flights in which there is no fault (i.e., healthy), a criss-crossed lines fault occurs, a stuck motor fault occurs, or an opposite-side broken line fault occurs. All data is taken from the correct broken line residual signal. The legend indicates which fault is actually present.

Notice that the data from healthy flights and flights with a broken line lie almost

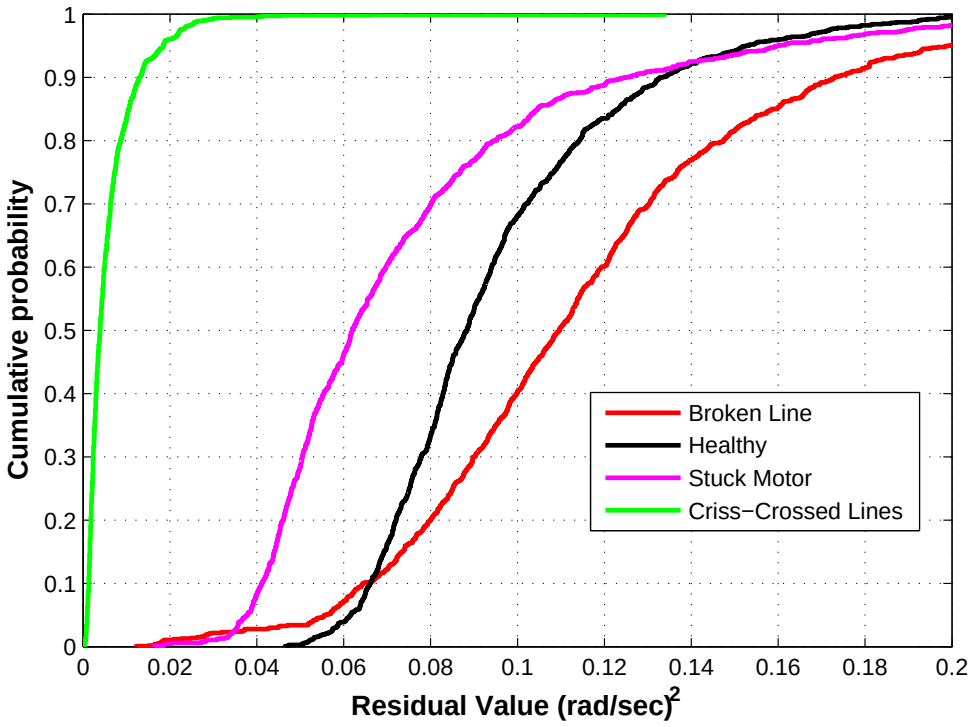

Figure 5-9: Post-Isolation Criss-Crossed Lines Residual Data: Criss-Crossed Lines Fault Isolated

on top of each other. This is because a healthy system still performs well if a broken line recovery strategy is implemented, as demonstrated by results in Section 5.4.1. The broken line residual is large, however, if the broken line recovery strategy is implemented while a criss-crossed lines fault or opposite broken line fault is present. Some flights with a stuck motor fault still perform well because the broken line recovery strategy is effective for many stuck motor cases. Consequently, data from the stuck motor curve is ignored. False isolation of a stuck motor fault is rare, given the low rate of false alarm for the motor residual signals (see Chapter 3). Choosing a post-isolation threshold of 0.016 $(\text{rad/sec})^2$ for the broken line residual ensures that about 99% of false isolations where a criss-crossed lines fault or opposite broken line fault is present will be detected, and about 99.9% of successful isolations will not be triggered as incorrect. False isolation will likely not be declared in the healthy case, but the miss distance should still be small.

Figure 5-9 shows CDFs of data from simulations in which a criss-crossed lines

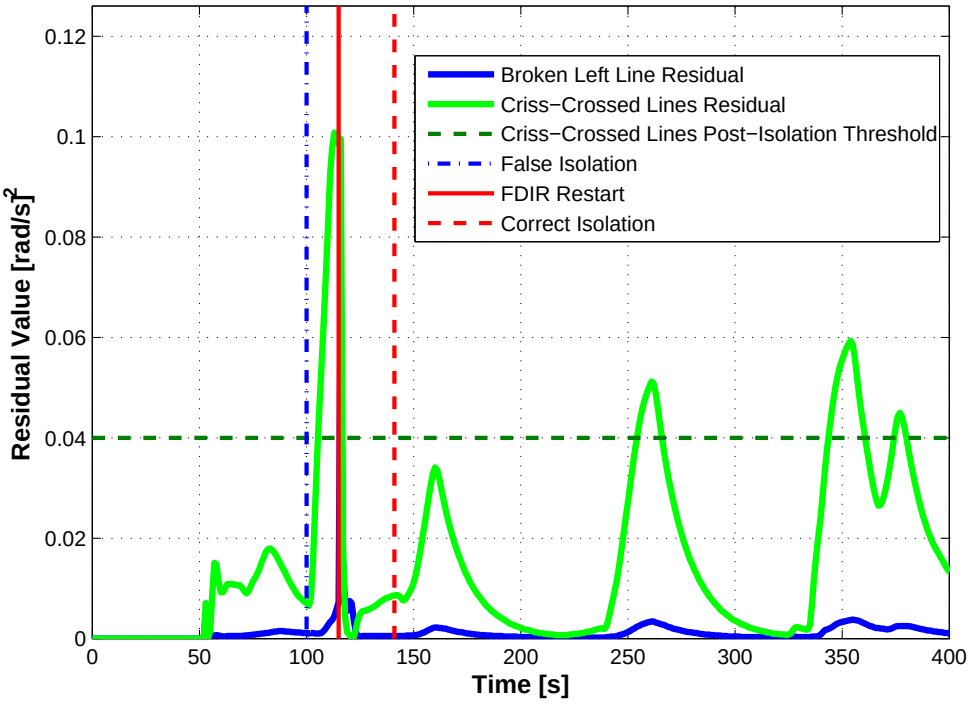

Figure 5-10: Fault Specific Observer Residuals: Broken Left Line with Criss-Crossed Lines False Isolation

fault is isolated 100 seconds into the flight. Data is included from flights in which a criss-crossed lines, broken line, stuck motor, or no fault has occurred. All data is taken from the criss-crossed lines residual signal. The legend indicates which fault is actually present. Unlike a false isolation of a broken line, healthy flights perform poorly when the criss-crossed lines fault is isolated. This is because implementing the criss-crossed lines recovery strategy on healthy flights is essentially simulating a criss-crossed lines fault by reversing the differential toggle command. The post-isolation threshold for criss-crossed lines is chosen to be 0.04 $(\text{rad/sec})^2$. This is chosen to ensure that only about 0.4% of flights in which a criss-crossed lines fault actually occurs is marked as a false isolation. Choosing this threshold, however, means that 2.8% of flights in which a broken line occurs but a criss-crossed lines fault is isolated will not be identified as a false alarm. Considering the fact that the probability of a false isolation is so small (see Table 4.3), this is acceptable.

Figure 5-10 shows an example of the fault-specific observer residual behavior dur-

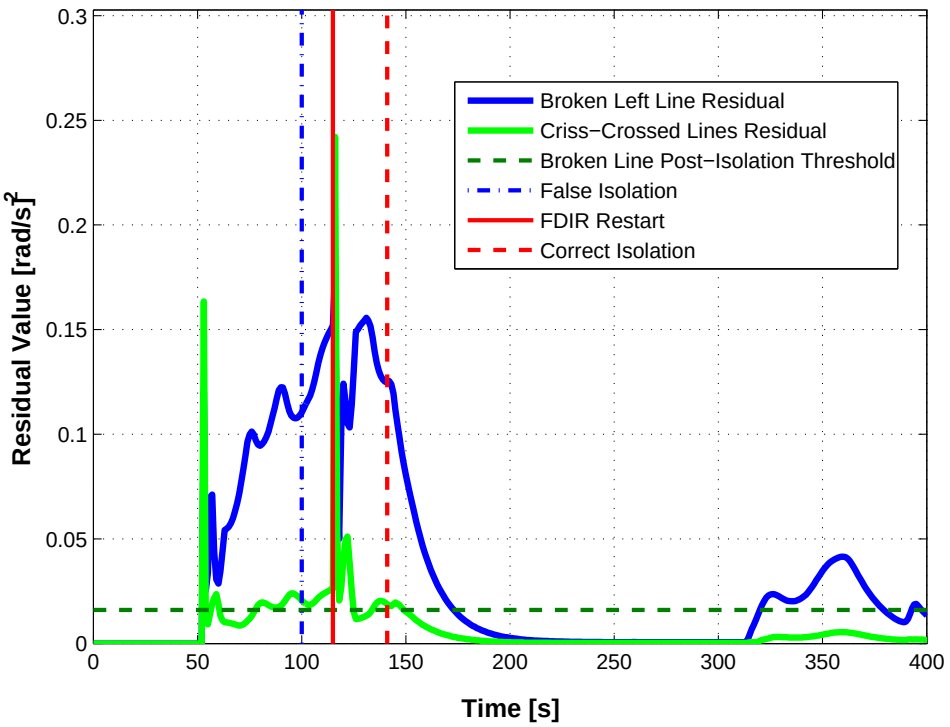

Figure 5-11: Fault Specific Observer Residuals: Criss-Crossed Lines with Broken Left Line False Isolation

ing a simulation in which a broken left line occurs at the beginning of the flight. However, a criss-crossed lines fault is incorrectly isolated 100 seconds into the flight. Note that this false isolation was forced in the simulation; the residuals in Figure 5-10 show that a broken left line would have been correctly isolated during a normal flight. After the 15-second settling period has passed after the false isolation, the criss-crossed lines residual is well above the post-isolation threshold. So, a restart of FDIR is initiated. Several seconds later, a correct isolation occurs in which a broken left line fault is declared.

Figure 5-11 shows a similar example, but in this case a criss-crossed lines fault is present while a broken left line is isolated incorrectly 100 seconds into the flight. Again, the false isolation was forced, because a correct isolation would normally have occurred in this particular case. The broken left line residual is above the threshold 15 seconds after the false isolation, and FDIR is restarted. Shortly after the restart, FDIR is successful in isolating the presence of the criss-crossed lines fault. Implement-

ing post-isolation thresholds provides a safeguard against a potentially devastating false isolation, and gives FDIR a second chance to successfully isolate a fault..

5.5 Results

The success of the fault recovery strategies described in the previous section is measured by comparing the resulting miss distance of a flight with a fault to the miss distance expected from a healthy flight. This section presents results from an extensive Monte Carlo simulation study. The FDI method described in Chapters 3 and 4 is used in each of these simulations. As a result, the recovery statistics are subject to the performance of the FDI algorithm. A lack of successful isolation, for example, makes it impossible to successfully recover from the fault. The nominal guidance strategy continues to run until isolation has declared that a particular fault has occurred. Integrating the recovery strategy presented in this chapter with the FDI method yields results that can reasonably be expected in flight.

5.5.1 Criss-Crossed Lines

To assess the success of the criss-crossed lines recovery strategy, 1000 Monte Carlo simulations in which a criss-crossed lines fault occurs at the beginning of each simulation are analyzed. Figure 5-12 shows an example of the East-North groundtrack from a simulation in which a criss-crossed lines fault is implemented. The system initially starts to loop as it attempts to go towards the target, but begins to execute homing correctly after isolation is successful. Detection and isolation occur soon after FDI is turned on (i.e., preflight mode has ended). At this point the recovery strategy for this fault, given in Eq. (5.1), is implemented, and the trajectory for the remainder of the simulation is similar to what would be expected from a healthy flight. The normalized miss distance for this simulation is 1.34, close to the median miss for healthy flights. CDFs of miss distances for all simulations of fault cases and the healthy case are shown in Figure 5-15. Table 5.2 shows the mean normalized miss distance of each case.

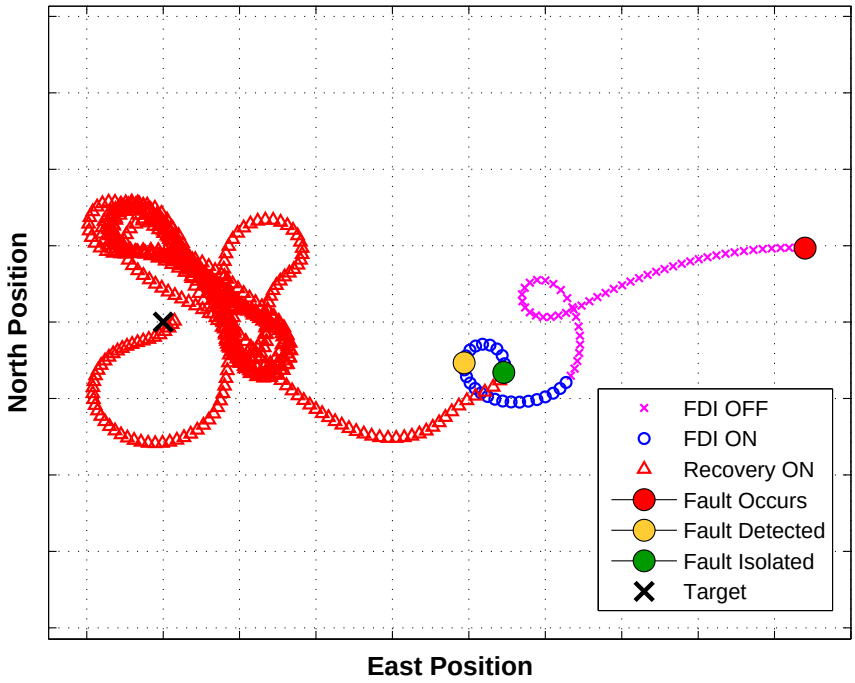

Figure 5-12: East-North Groundtrack of Simulation with Criss-Crossed Lines Fault: Recovery Implemented

5.5.2 Broken Line

Performance of the broken line recovery strategy is analyzed by running 1000 Monte Carlo simulations of flights in which a randomly chosen broken line fault occurs (e.g., broken left line). An example trajectory is shown in Figure 5-13, in which a broken left line occurs early in the simulation. During homing, the parafoil transitions to the looping submode three times when it strays too far from its desired heading. The loiter mode consists of tracking a circular trajectory, as expected, and the system lands close to the target despite the fault. The normalized miss distance for this simulation is 1.46.

5.5.3 Stuck Motor

As discussed in Section 5.3, there are three different categories of stuck motor faults for which there are three different recovery strategies. A stuck motor at a small

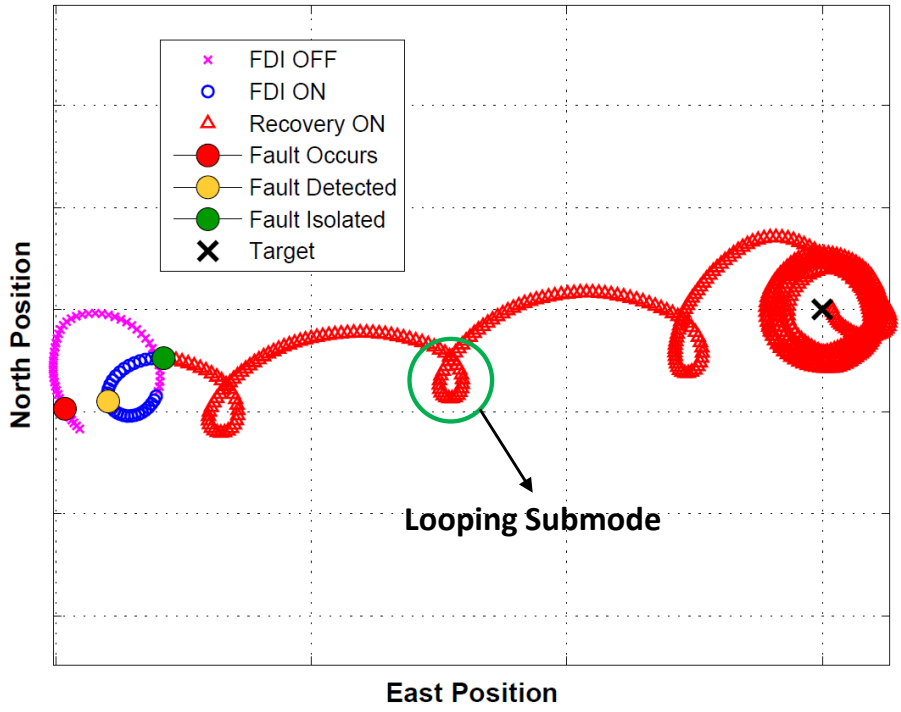

Figure 5-13: East-North Groundtrack of Simulation with Broken Left Line: Recovery Implemented

value is treated in the same way as a broken line on the side on which the motor is stuck. A stuck motor at a large value is treated as a broken line on the side opposite of the one on which the motor is stuck. A moderate stuck motor fault is allowed to progress through the nominal guidance plan, with the adjustment to the motor toggle command shown in Eqs. (5.11) and (5.12) employed to exploit the knowledge of the stuck motor position. Results from 1000 Monte Carlo simulations that vary the severity of the motor fault uniformly from stuck at zero deflection to maximum deflection are used to compile recovery statistics for this fault. The altitude at which each fault occurs is drawn from a uniform distribution, in a similar manner to the fault altitude selection for previous simulations.

Figure 5-14 shows the groundtrack for a simulated flight in which a stuck right motor occurs at 1.0 (i.e., the motor is stuck at its full deflection). This fault occurs during energy management. Note that detection and isolation occur almost simultaneously. As expected, the parafoil follows the recovery strategy for a broken left line

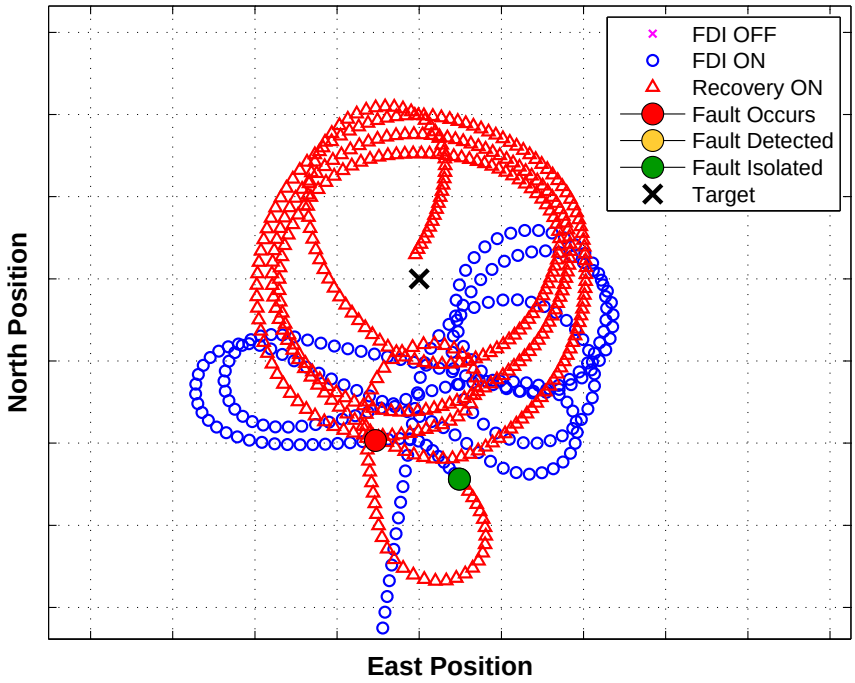

Figure 5-14: East-North Groundtrack of Simulation with Stuck Right Motor at Full Deflection: Recovery Implemented

and immediately transitions into a clockwise circular trajectory. The normalized miss distance for this simulation is 1.07. In each of the three simulation results presented, miss distances are comparable to those observed in the nominal case.

5.5.4 Overall Simulation Results

Figure 5-15 shows CDFs of normalized miss distances for the 1000 Monte Carlo simulations of the three fault cases as well as the healthy flight case with full FDIR implementation. Comparison to results in Figure 2-9 show a huge improvement over results without recovery. Moreover, a high percentage of simulated flights in which a fault occurs result in a miss distance of less than ten times the median miss for the healthy case. Figure 5-16 shows CDFs of normalized miss distances from 1000 Monte Carlo simulations of healthy flights as well as flights in which a randomly selected fault occurs, with and without recovery implemented. This figure shows tremendous

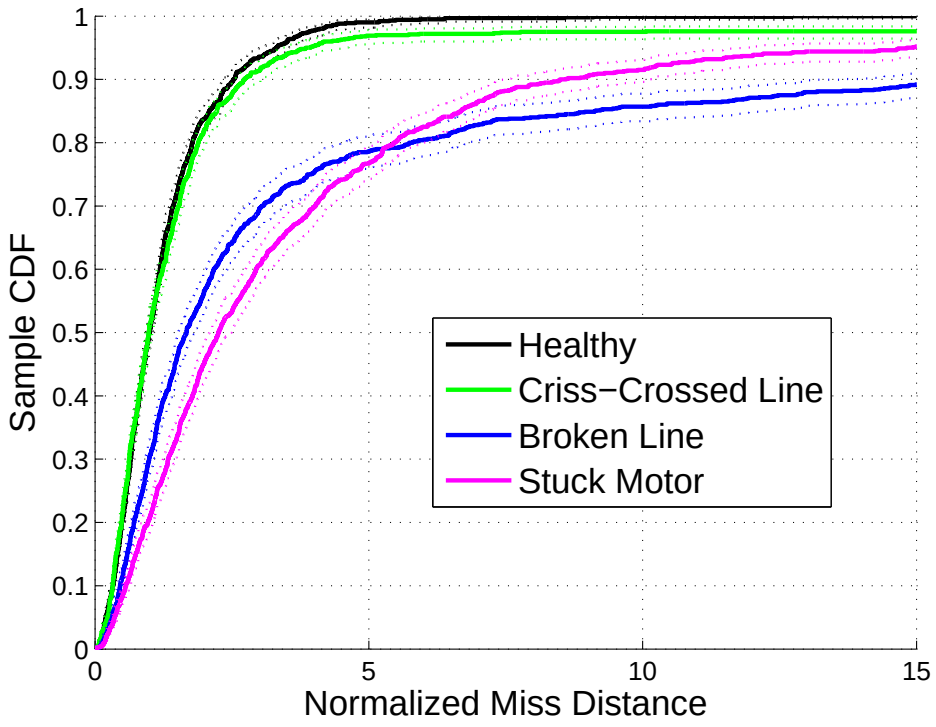

Figure 5-15: Miss Distance CDFs with 95% Confidence Bounds: Recovery Strategies Implemented

improvement in miss distance when the recovery strategies are implemented versus when the nominal guidance strategy is used. With recovery implemented, 90.9% of simulated flights result in a miss distance that meets the mission requirement (i.e., less than 10 times the median miss for healthy flights). Without recovery, only 17.5% of simulations meet this requirement. Table 5.2 summarizes the mean miss distance results from the simulations, and in particular shows the difference in results for stuck motor values of varying severity. Substantial improvement is seen for each case when recovery is implemented. The smallest improvement occurs for stuck motors at moderate values. Often, the remaining functioning motor is able to closely track differential toggle commands in this case.

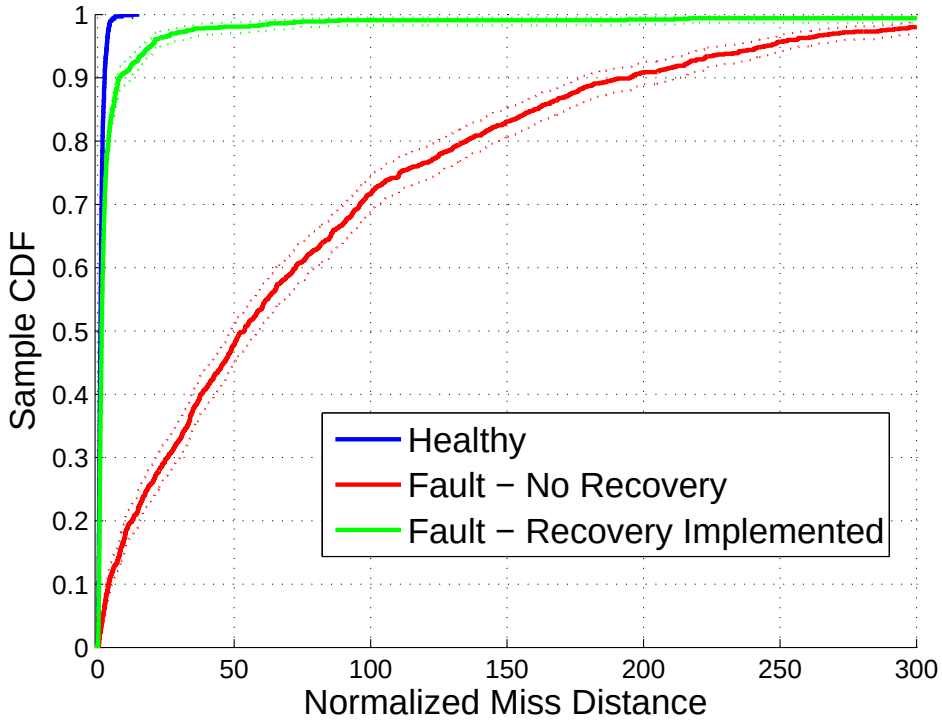

Figure 5-16: Miss Distance CDFs with 95% Confidence Bounds: Overall Results

Table 5.2: Mean Normalized Miss Distances for Various Fault Cases with and without Recovery Strategy

Case	**Mean Normalized Miss Distance**		
	No Recovery (i.e., Used Nominal Guidance)	Recovery Implemented	Improvement Factor
Healthy	1.3	1.3	1.0
Random Fault	79.3	7.1	11.2
Criss-Crossed Lines	121.8	10.7	11.4
Broken Line	76.2	7.0	10.9
Stuck Motor - Overall	37.0	4.3	8.6
Stuck Motor - Small Values	66.7	2.5	26.7
Stuck Motor - Moderate Values	12.2	4.5	2.7
Stuck Motor - Large Values	76.4	5.6	13.6

Chapter 6

Flight Test Results

The high-fidelity simulations used to demonstrate the success of the recovery strategies developed in this work provide insight into how the system is expected to perform during an actual flight. The demonstration of the success of the FDIR algorithm in-flight can be used to indicate the validity of some of the assumptions made when simulating. This section presents data from five actual flight tests in which faults were simulated or implemented physically and the FDIR algorithm attempted to isolate the fault and then recover from that fault using the recovery methods developed in this work.

The faults implemented on the flight tests are as follows: one flight had a broken left line, two had criss-crossed lines, and two had stuck left motors. For four of these flight tests, the faults were not physically implemented, but were simulated by commanding the motors. One criss-crossed lines fault was simulated by sending the opposite differential toggle command to control. The other criss-crossed lines fault was physically implemented; the lines were actually criss-crossed during rigging. The broken left line fault was simulated by freezing the left motor at zero deflection. The stuck left motor faults were simulated by commanding the left motor to remain at the stuck value for the entire flight. FDIR did not receive knowledge of the motor commands used to simulate each fault; the nominal motor signal is used so that each flight test reasonably approximates the occurrence of an actual fault.

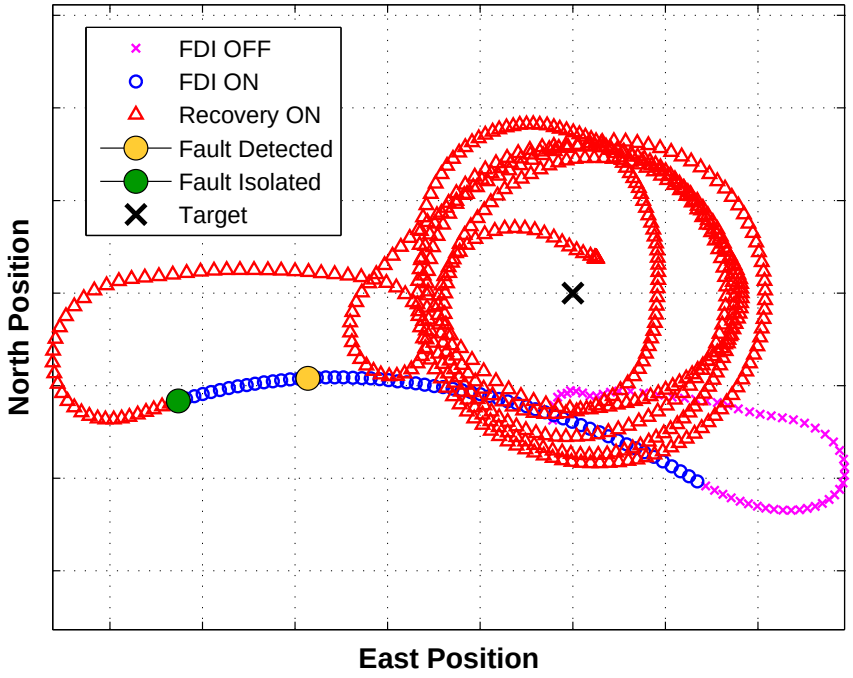

Figure 6-1: East-North Groundtrack of Flight with Simulated Broken Left Line Fault

6.1 Simulated Broken Left Line

See Figure 6-1 for the groundtrack results from the first flight test, in which a broken left line was simulated at the beginning of the flight. Broken line faults often occur during canopy deployment when the system exits the aircraft. Simulating a broken line at the beginning of the flight represents a likely scenario for an occurrence of this fault [11]. During the homing mode, detection and isolation occur and recovery begins. After this point, the parafoil behaves as expected based on the simulation results. The system follows a circular loitering trajectory before landing close to the target. The normalized miss distance for this flight is 1.68, over four times better than the mean normalized miss distance from broken line simulations (see Table 5.2).

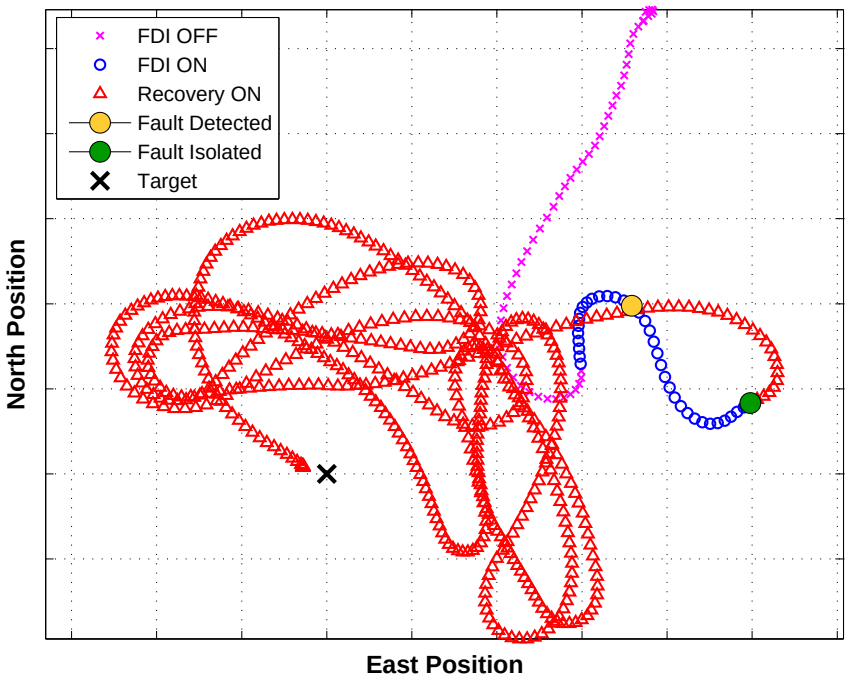

Figure 6-2: East-North Groundtrack of Flight with Simulated Criss-Crossed Lines Fault

6.2 Simulated Criss-Crossed Lines

Figure 6-2 shows a flight in which a criss-crossed lines fault was simulated immediately upon canopy deployment. This reflects a realistic scenario because this fault occurs during the pre-flight canopy rigging process. The system goes away from the target initially until isolation occurs and recovery begins. Then, the flight proceeds normally with homing, energy management, BLG, and flare modes all executed as expected. The normalized miss distance for this flight is 1.11, about ten times better than the mean normalized miss distance from criss-crossed lines simulations (see Table 5.2).

6.3 Stuck Left Motor Small

Figure 6-3 shows a flight in which a stuck left motor at a deflection of 0.08 was simulated immediately upon canopy deployment. Although this fault was simulated

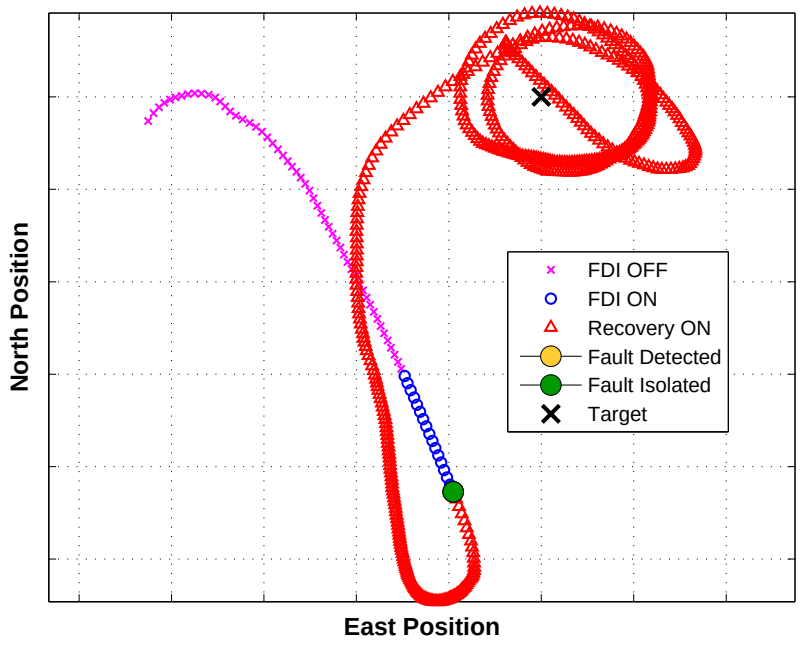

Figure 6-3: East-North Groundtrack of Flight with Stuck Left Motor at Small Value

by commanding the left motor to be 0.08 for the entire flight, the behavior should be similar to what is expected when an actual stuck motor occurs. This fault is detected and isolated simultaneously; the detection occurs via the left motor residual signal, and since this signal is large when isolation begins a stuck left motor fault is immediately declared. A stuck motor value of 0.08 lies in the small stuck motor region as described in Section 5.3, so FDIR implements the broken left line recovery strategy. The system follows a clockwise circular loiter trajectory and lands with a normalized miss distance of 5.19, which is over two times worse than the mean normalized miss distance for faults in which a motor is stuck at a small value. Although this miss is larger than expected, it is still only about half as large as the maximum acceptable miss of 10.

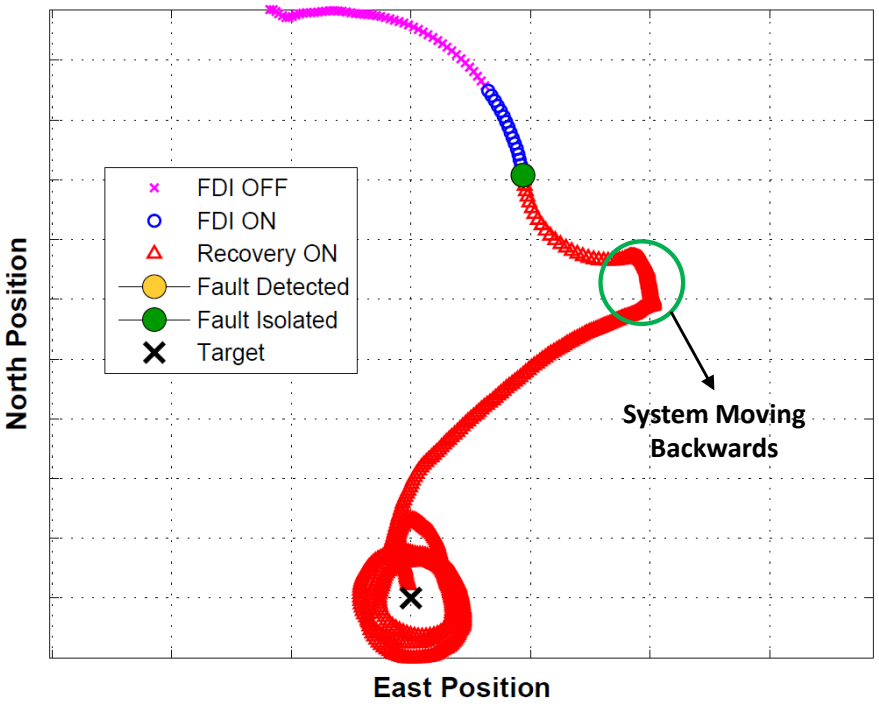

Figure 6-4: East-North Groundtrack of Flight with Stuck Left Motor at Large Value

6.4 Stuck Left Motor Large

Figure 6-4 shows a flight in which a stuck left motor at a deflection of 0.8 was simulated immediately upon canopy deployment. This fault falls within the large stuck motor category, specified in Section 5.3. Consequently, this fault is treated as a broken right line. Similarly to the stuck motor at a small value, detection and isolation occur simultaneously early on in the flight. There is an anomaly that occurs during homing, and is circled in Figure 6-4. When the parafoil attempts a left turn during the looping submode to face the homing target, high winds are encountered that blow the system backwards for a period of time. Once the wind subsides, the system resumes a normal flightpath, executing a counter-clockwise circular loiter trajectory and landing with a normalized miss distance of 1.68, which is about three times better than the mean normalized miss distance for faults in which a motor is stuck at a large value. This flight test provides an example of how FDIR can be successful even in difficult environmental conditions.

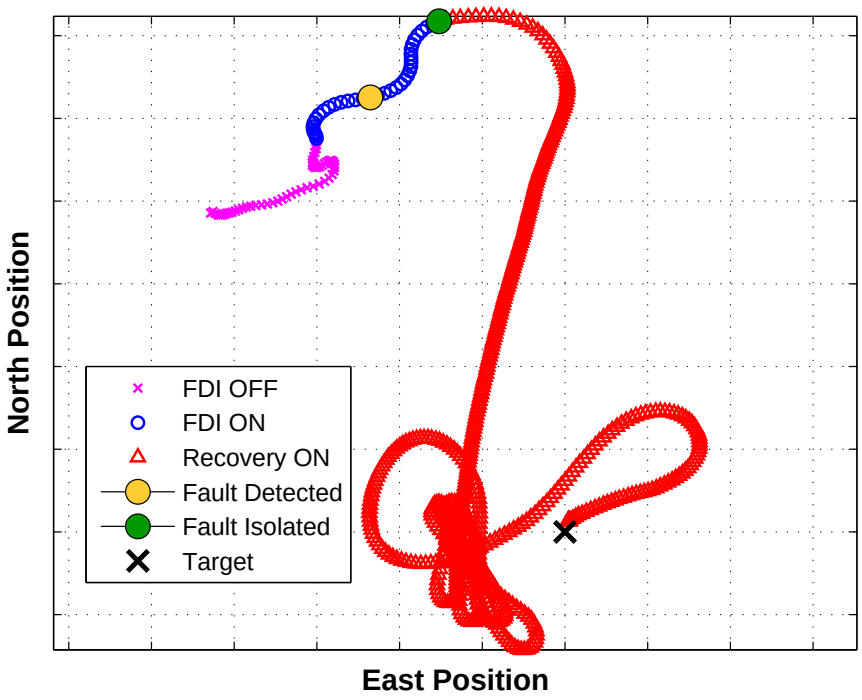

Figure 6-5: East-North Groundtrack of Flight with Criss-Crossed Lines Fault

6.5 Physical Criss-Crossed Lines

Figure 6-5 shows a flight in which the control lines were physically criss-crossed during the rigging process. This differs from the simulated case in Section 6.2, and is the first full test of FDIR on the parafoil system when a fault is actually present. As the results in Figure 6-5 demonstrate, the system performs similarly in this case as it does in simulation and during the previous flight test in which a criss-crossed lines fault was simulated. The fault is isolated early in the flight at which point recovery is implemented. The system then completes homing, energy management, and BLG successfully and lands with a normalized miss distance of 1.09, which is close to the median miss for healthy flights.

6.6 Conclusions

During these flight tests, FDI identified each fault correctly and sufficiently quickly that the recovery process had time to implement the appropriate strategy. For these five cases, results are consistent with the FDIR algorithm design assumptions. Specifically, results show that the linear model used for FDIR captures the heading rate behavior well for these fault cases, and the simulation results reasonably reflect expected in-flight performance. The successful diagnosis of the faults in both cases should be considered evidence that linearization, uncertainty, and noise assumptions used in developing FDIR were reasonable.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

This thesis presented a fault detection, isolation and recovery method for the autonomous parafoil system that is effective during flights in which one of three common faults occurs. The detection method evaluates a residual signal generated from the navigated heading rate and the output from an observer based on the nominal system model, as well as signals generated from motor telemetry. The isolation method uses the same motor signals from detection to determine if the fault is actuator-related. Other, non-actuator faults are isolated using a bank of fault-specific observers that model the system in the presence of a fault. Recovery strategies were developed that specifically mitigate the effects of these particular faults, using the results from the isolation phase. These strategies modify existing guidance methods and introduce a new method, called track and loop guidance, that enables the parafoil to track a heading command with a single functioning actuator. FDIR was evaluated using high-fidelity Monte Carlo simulations. Results from extensive simulation testing showed that choosing appropriate thresholds and scaling factors for FDI result in a 97.8% rate of successful detection and 99.59% rate of successful isolation for simulations in which a random fault occurs, and only a 0.5% rate of false alarms for healthy flight simulations. Implementing recovery results in a reduction of mean miss distance by a factor of 11.2 when recovery strategies were implemented on simulations with a

random fault present, versus cases without recovery. Results from five flight tests indicate that FDIR can be performed successfully in practice as well as in simulation. Miss distances for the five flight tests that executed FDIR in the presence of simulated faults were consistent with those typically seen under nominal conditions.

7.2 Future Work

Future work on this topic will involve expanding the set of faults considered during FDIR. This may require integrating the method presented in this work with another method (e.g., Culpepper et al. [11]), or expanding upon the FDIR method presented in this work. Extension of this method may require adding to the bank of fault-specific observers, or executing a diagnostic action that assists with the isolation of new faults.

Another way of validating results presented in this work would be to run FDI on previous flight tests. While recovery could not be implemented, it is possible that running the FDI algorithm on collected telemetry from old flight tests may provide insight as to whether or not a fault occurred, as well as what that particular fault was. Furthermore, running FDI on healthy flight tests will generate an empirical rate of false alarms for flight tests, and further evaluate the success of FDI.

We may also consider adding more measurements to the system to improve FDI. A major challenge of this work was using the limited available measurements to attempt to determine whether an anomaly has occurred. Redundant measurements can greatly increase detection rates as well as decrease rates of false alarm.

Appendix A

Determination of Observer Gain

When choosing a gain L for the state observer in Eqs. (3.3) and (3.4) used for fault detection, the goal is to make the closed-loop estimator stable and utilize the system model and measurements in a systematic and effective manner [6]. This is accomplished using the Linear Quadratic Estimator (LQE) formulation [28]. LQE solves an optimal control problem to minimize the following cost function,

$$J = \text{trace}\{Q[k]\} \tag{A.1}$$

$$\text{s.t. } Q[k] = E[(x[k] - \hat{x}[k])(x[k] - \hat{x}[k])^T] \tag{A.2}$$

where $x[k]$ is the true state vector at time step k and $\hat{x}[k]$ is the state vector estimate at time step k after the measurement update step has been performed. In the steady-state,

$$L_{ss} = Q_{ss}C^T(CQ_{ss}C^T + R_{vv})^{-1} \tag{A.3}$$

where the steady-state error covariance Q_{ss} is determined by solving the discrete-time algebraic Riccati equation [28],

$$Q_{ss} = A(Q_{ss} - Q_{ss}C^T(CQ_{ss}C^T + R_{vv})^{-1}CQ_{ss})A^T + R_{ww} \tag{A.4}$$

where $R_{vv} = E[vv^T] \in \mathbb{R}$ is the sensor noise covariance matrix, $R_{ww} = E[ww^T] \in \mathbb{R}^{3x3}$ is the process noise covariance matrix, and A and C are the dynamics and output matrices, respectively, from Eqs. (2.4) and (2.5). By determining the covariance matrices R_{ww} and R_{vv} for the parafoil system used in this work, the optimal observer gain $L = L_{ss}$ is chosen. The covariance matrices are not explicitly known, so simulation results are used to estimate the noise covariance. Specifically, the focus in designing L is based on trust in the model versus trust in the measurements [1]. A large value of R_{vv} relative to the entries in R_{ww} indicate that the measurements are more reliable than the system model. Conversely, if the entries in R_{ww} are larger than R_{vv} the model should be trusted more than the measurements [6]. Only the ratio of R_{vv} to R_{ww} is of interest, and R_{ww} is set to I_3 for simplicity. The value of R_{vv} is set to the ratio of the empirical covariance of the measurement error to the empirical covariance of the model output error. When computing L, R_{vv} contains all of the information about the trust in the model relative to the measurements.

The covariances of the measurement error and model output error are estimated empirically by analyzing results from 1000 Monte Carlo simulations of the parafoil system using the nonlinear simulator described in Section 2.2.1. For each of the 1000 simulations, the measurement error is computed as follows,

$$e_{meas} = y_{true} - y_{meas} \tag{A.5}$$

where y_{meas} is the vector of measured heading rate values over the entire simulation, estimated from the on-board Extended Kalman Filter (EKF), and y_{true} is the vector of actual heading rate values obtained from simulation data. The model error is computed as,

$$e_{model} = y_{true} - Cx \tag{A.6}$$

where x evolves over time according to Eq. (2.4). The average covariance ratio is determined as,

$$R_{vv,i} = \frac{\text{Cov}(e_{meas,i})}{\text{Cov}(e_{model,i})} \tag{A.7}$$

$$R_{vv} = \text{mean}(R_{vv,i}) \tag{A.8}$$

where $R_{vv,i}$ is the covariance ratio for simulation run i, $\text{Cov}(e_{meas,i})$ is the covariance of the measurement error during run i, and $\text{Cov}(e_{model,i})$ is the covariance of the model error during run i. By letting $R_{ww} = I_3$ and using Eq. (A.8) to define R_{vv}, L is obtained by solving Eqs. (A.3) and (A.4). This gain stabilizes the system observer and adjusts the state estimate optimally given the information we have about the process and measurement noise. The gain used for residual generation in this work is chosen using the methods described in this Appendix.

Appendix B

Exponential Low-Pass Filtering

Filtering, or smoothing, each residual signal used for fault detection and isolation (FDI) is a way to improve detection and isolation statistics by capturing the trend of the signal rather than potentially raising false alarms with each signal spike. The goal of the filtering process is to eliminate high-frequency effects as much as is necessary while minimizing the resulting lag of the residual signal. Each residual is smoothed using an exponential low-pass filter [23],

$$r_s[k] = r_s[k-1] + \alpha(r[k] - r_s[k-1]) \tag{B.1}$$

$$\alpha = \frac{1}{1 + \frac{1}{2\pi f}} \tag{B.2}$$

where $r_s[k]$ is the value of the *smoothed* residual at time step k, $r[k]$ is the value of the true residual from Eq. (3.7), Eq. (3.12), or Eq. (3.13) at time step k, $\alpha \in [0, 1]$ is the smoothing factor and f is the cutoff frequency of the filter. The smaller the cutoff frequency, the more smoothed the data becomes [23]. The cutoff frequency for the filters of the heading rate-based residuals (i.e., heading rate residual and fault-specific observer residuals) and the motor residuals are chosen based on the frequency response characteristics of the system model and of the motor model. The cutoff frequency for each system should be less than the slowest pole [29].

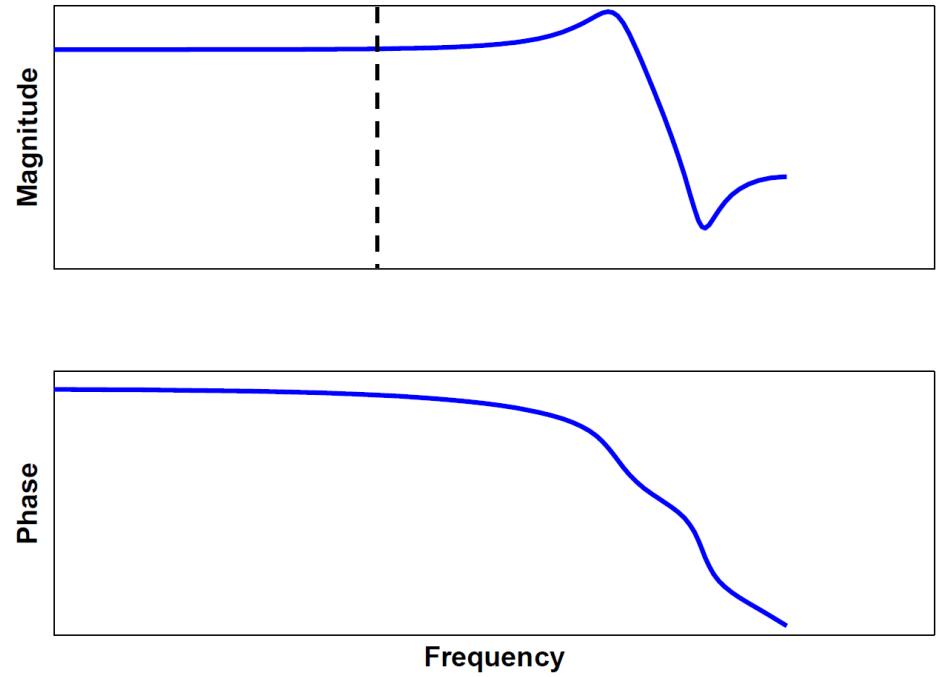

Figure B-1: Bode Plot of Parafoil Lateral Dynamics Model

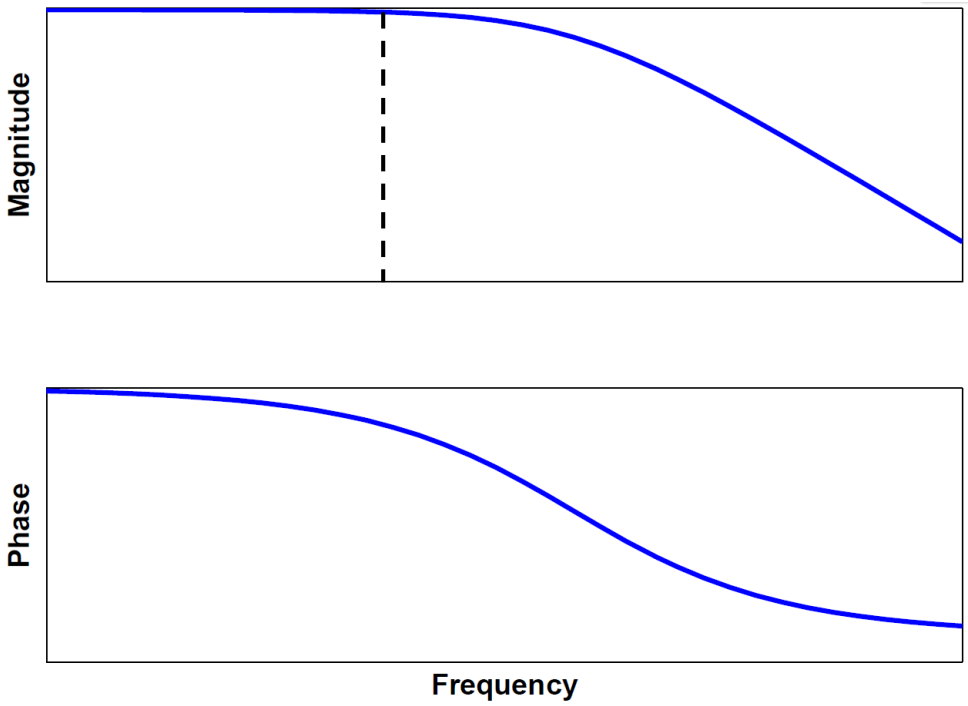

Figure B-2: Bode Plot of Motor Model

Figure B-1 shows the Bode plot for the parafoil system model given in Eqs. (2.4) and (2.5). Although, the axes have been removed, the dutch roll mode of the system is visible. The cutoff frequency is placed as shown by the vertical dotted line to eliminate the effects of the system modes. This cutoff frequency is used to filter the heading rate residual for detection in Section 3.2.1 as well as all of the fault-specific observer residuals in Section 4.2.2.

The motor model used to generate the motor residual signal can be approximated by a system with a single real pole. The Bode plot of the motor model is shown in Figure B-2, and the cutoff frequency is placed as shown by the vertical dotted line. This cutoff frequency is used to filter the motor residual signal in Section 3.2.2.

Choosing the cutoff frequencies as described in this appendix is used to improve detection and isolation statistics by filtering out high-frequency effects so that the overall behavior of each residual can be analyzed without responding to anomalous spikes in the signals.

Appendix C

Track and Loop Guidance Parameter Selection

The track and loop guidance modification described in Section 5.2.1 is used for recovery from faults in which turning authority in one direction is extremely limited. This modification is implemented by choosing four parameters. These parameters are related to the heading error ψ_e; the difference between the current heading and target heading of the parafoil. Two track and loop parameters, loop start error ψ_{start} and loop end error ψ_{end} are used to define when the system transitions from the tracking submode to the looping submode, and vice versa. These two parameters are chosen to limit the amount of times the system enters the looping submode. The other two parameters, heading error bias ψ_{bias} and small heading error ψ_{small}, determine how the heading rate command is adjusted during the tracking mode, based on the heading error of the system. These parameters are chosen in tandem to maximize the performance of this algorithm (i.e., minimize miss distance of flights in which this method is used).

The loop start error determines when the transition occurs from the tracking submode to the looping submode; this happens when the system has a large heading error and cannot turn back to the target (e.g., system is far to the right of the target and has a broken left line). The system should only enter looping mode, and execute a turn back to the target, when there is a high confidence that the system

has overshot the target. The measured heading error is subject to the accuracy of the Extended Kalman Filter (EKF) estimate of the system heading, so error statistics of the heading rate estimate are used to determine ψ_{start}. The loop start error is chosen to be somewhere between two and three times the standard deviation of the heading error. Consider a fictitious parafoil and payload system where the standard deviation of the heading error is 7 degrees. Then, the loop start error should be selected between 14 and 21 degrees. This can be tuned based on observed performance in simulation or flight test.

The loop end error determines when the transition between the looping submode and the tracking submode occurs. When the system exits the looping submode, it should be turning at the maximum allowable heading rate. The loop end error should be chosen such that the system has sufficient time to slow down such that it does not overshoot the heading target again. Consider a system with a maximum turn rate command of 10 deg/sec that takes 5 seconds for a heading rate command to take effect. So, assuming the worst-case, the parafoil will sweep through 50 degrees before fully slowing down from the maximum heading rate. Again allowing for some buffer for errors in the heading estimate, a loop end error between 64 and 71 degrees will provide enough of a heading range for the system to slow down and not overshoot the target again. EKF error information from the true system is used to determine ψ_{start} and ψ_{end} for this work.

The other two parameters, heading error bias and small heading error, are chosen iteratively to maximize the performance. For each selection of ψ_{bias} and ψ_{small}, 200 Monte Carlo simulations are run in which a broken line occurs immediately upon canopy deployment, and recovery is implemented 100 seconds into the flight. These simulations are run with the true system used in this work, as opposed to the fictitious system used to describe the process of selecting ψ_{start} and ψ_{end}. Fault detection and isolation (FDI) is not performed because this is only an analysis of the effectiveness of the fault recovery strategy. The mean normalized miss distance of each set of trials is recorded in Table C.1. The best performance results when $\psi_{bias} = 30$ deg and $\psi_{small} = 5$ deg, and these values are used for the parafoil system in this thesis.

Table C.1: Track and Loop Parameter Iteration

Heading Error Bias (deg)	**Small Heading Error (deg)**	**Mean Normalized Miss Distance**
0	0	8.70
10	7.5	8.56
10	5	7.84
10	2.5	7.41
10	0	5.25
20	7.5	3.41
20	5	3.57
20	2.5	3.60
20	0	3.58
30	7.5	3.21
30	**5**	**3.13**
30	2.5	3.29
30	0	3.74
40	7.5	3.16
40	5	3.20
40	2.5	3.51
40	0	4.12
50	7.5	3.34
50	5	3.24
50	2.5	3.75
50	0	5.87

Bibliography

[1] Michael Athans. The role and use of the stochastic linear-quadratic-gaussian problem in control system design. *IEEE Transactions on Automatic Control*, 16 (6):529–552, 1971.

[2] Timothy M. Barrows. Apparent mass of parafoils with spanwise camber. *Journal of Aircraft*, 39(3):445–451, 2002.

[3] Timothy M. Barrows. Multibody parafoil model. In *Proc. 20th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Seattle, WA, May 2009.

[4] Richard V. Beard. *Failure Accommodation in Linear Systems Through Self-Reorganization*. PhD dissertation, Massachusetts Institute of Technology, 1971.

[5] Keith Bergeron, Amer Fejzic, and Steve Tavan. AccuGlide 100: Precision airdrop guidance and control via glide slope control. In *Proc. 21st AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Dublin, Ireland, May 2011.

[6] Arthur E. Bryson, Jr. and Yu-Chi Ho. *Applied Optimal Control: Optimization, Estimation and Control*, chapter 12, pages 348–389. Taylor & Francis, Abingdon, UK, 1975.

[7] David Carter, Sean George, Philip Hattis, Leena Singh, and Steven Tavan. Autonomous guidance, navigation, and control of large parafoils. In *Proc. 18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Munich, Germany, May 2005.

[8] David Carter, Leena Singh, Leonard Wholey, Scott Rasmussen, Tim Barrows, Sean George, Marc McConley, Chris Gibson, Steve Tavan, and Brian Bagdonovich. Band-limited guidance and control of large parafoils. In *Proc. 20th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Seattle, WA, May 2009.

[9] David W. Carter, Sean George, Philip D. Hattis, Marc W. McConley, Scott A. Rasmussen, Leena Singh, and Steve Tavan. Autonomous large parafoil guidance, navigation, and control system design status. In *Proc. 19th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Williamsburg, VA, May 2007.

[10] Peter Crimi. Lateral stability of gliding parachutes. *Journal of Guidance, Control, and Dynamics*, 13(6):1060–1063, 1990.

[11] Sean Culpepper, Michael Ward, Mark Costello, and Keith Bergeron. Adaptive control of damaged parafoils. In *Proc. AIAA Aerodynamic Decelerator Systems Conference*, Daytona Beach, FL, 2013.

[12] Randal K. Douglas and Jason L. Speyer. Robust fault detection filter design. *Journal of Guidance, Control, and Dynamics*, 19(1):214–218, 1996.

[13] Fernando Figueroa, John Schmalzel, Mark Walker, Meera Venkatesh, Ravi Kapadia, Jon Morris, Mark Turowski, and Harvey Smith. Integrated system health management: Foundational concepts, approach, and implementation. In *Proc. AIAA Infotech@Aerospace Conference*, Seattle, WA, 2009.

[14] Fernando Figueroa, John Schmalzel, Jon Morris, Mark Turowski, and Richard Franzl. Integrated system health management: Pilot operational implementation in a rocket engine test stand. In *Proc. 18th AIAA Infotech@Aerospace*, Atlanta, GA, 2010.

[15] Paul M. Frank. Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy–a survey and some new results. *Automatica*, 26 (3):459–474, 1990.

[16] Paul M. Frank. Enhancement of robustness in observer-based fault detection. *International Journal of Control*, 59(4):955–981, 1994.

[17] Janos Gertler. Analytical redundancy methods in fault detection and isolation. In *Proc. IFAC/IAMCS Symposium on Safe Process*, 1991.

[18] Peter D. Hanlon and Peter S. Maybeck. Characterization of kalman filter residuals in the presence of mismodeling. *IEEE Transactions on Aerospace and Electronic Systems*, 36(1):114–131, 2000.

[19] Phil Hattis and Steve Tavan. Precision airdrop. *Aerospace America*, pages 38–42, 2007.

[20] Philip D. Hattis, Brent D. Appleby, Thomas J. Fill, and Richard Benney. Precision guided airdrop system flight test results. In *Proc. 14th Aerodynamic Decelerator Systems Technology Conference*, San Francisco, CA, 1997.

[21] Philip D. Hattis, Darryn P. Campbell, David W. Carter, Marc McConley, and Steve Tavan. Providing means for precision airdrop delivery from high altitude. In *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, 2006.

[22] Inseok Hwang, Sungwan Kim, Youdan Kim, and Chze Eng Seah. A survey of fault detection, isolation, and reconfiguration methods. *IEEE Transactions on Control Systems Technology*, 18(3):636–653, 2010.

[23] Rob J. Hyndman, Anne B. Koehler, J. Keith Ord, and Ralph D. Snyder. *Forecasting with Exponential Smoothing: The State Space Approach*, chapter 2, pages 348–389. Springer, New York, NY, 2008.

[24] Rolf Isermann. Process fault detection based on modeling and estimation methods: A survey. *Automatica*, 20(4):387–404, 1984.

[25] Rolf Isermann and Peter Ballé. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 5 (5):709–719, 1997.

[26] Thomas Jann. Advanced features for autonomous parafoil guidance, navigation and control. In *Proc. 18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Munich, Germany, May 2005.

[27] Harold L. Jones. *Failure Detection in Linear Systems*. PhD dissertation, Massachusetts Institute of Technology, 1973.

[28] Rudolph E. Kalman and Richard S. Bucy. New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 82(1):95–108, 1961.

[29] John H. Lumkes, Jr. *Control Strategies for Dynamic Systems: Design and Implementation*, chapter 6. Marcel Dekker, Inc., 2002.

[30] MATLAB. *version 7.14.0.739 (R2012a)*. The MathWorks Inc., Natick, Massachusetts, 2012.

[31] R.K. Mehra and J. Peschon. An innovations approach to fault detection and diagnosis in dynamic systems. *Automatica*, 7(5):637–640, 1971.

[32] Timothy E. Menke and Peter S. Maybeck. Multiple model adaptive estimation applied to the VISTA F-16 flight control system with actuator and sensor failures. In *Proc. IEEE National Aerospace Electron Conference*, Dayton, OH, May 1992.

[33] Paul A. Mortaloni, Oleg A. Yakimenko, Vladimir N. Dobrokhodov, and Richard M. Howard. On the development of a six-degree-of-freedom model of a low-aspect-ratio parafoil delivery system. In *Proc. 17th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Monterey, CA, May 2003.

[34] E.S. Page. Continuous inspection schemes. *Biometrika*, 41(1):100–115, 1954.

[35] Ron J. Patton and Jie Chen. Review of parity space approaches to fault diagnosis for aerospace systems. *Journal of Guidance, Control, and Dynamics*, 17(2):278–285, 1994.

[36] Ron J. Patton and Jie Chen. On eigenstructure assignment for robust fault diagnosis. *International Journal of Robust and Nonlinear Control*, 10(14):1193–1208, 2000.

[37] Christopher Rossi. Vehicle health monitoring using stochastic constraint suspension. Master's thesis, Massachusetts Institute of Technology, 2012.

[38] Christopher M. Rossi, David Benson, Russell Sargent, and Louis S. Breger. Model-based design for vehicle health monitoring. In *Proc. Infotech@Aerospace*, Garden Grove, CA, 2012.

[39] Christopher M. Rossi, Louis Breger, David Benson, Russell Sargent, and Lorraine M. Fesq. Vehicle health monitoring using stochastic constraint suspension. In *Proc. AIAA Guidance, Navigation, and Control Conference*, Minneapolis, MN, 2012.

[40] Russell Sargent, Ian Mitchell, Louis Breger, David Benson, Chris Bessette, Renato Zanetti, and Joseph E. Groszkiewicz. A fault management strategy for autonomous rendezvous and capture with the ISS. In *Proc. Infotech@Aerospace*, St. Louis, MO, 2011.

[41] Silvio Simani, Cesare Fantuzzi, and Ron J. Patton. Model-based fault diagnosis in dynamic systems using identification techniques. *International Journal of Robust and Nonlinear Control*, 15(1):509–512, 2005.

[42] Nathan Slegers and Mark Costello. Model predictive control of a parafoil and payload system. In *Proc. AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Providence, RI, 2004.

[43] Mark A. Sturza. Navigation system integrity monitoring using redundant measurements. *Journal of The Institute of Navigation*, 35(4):69–87, 1988.

[44] Steve Tavan. Status and context of high altitude precision aerial delivery systems. In *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, 2006.

[45] Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.

[46] Michael Ward, Mark Costello, and Nathan Slegers. Specialized system identification for autonomous parafoil and payload systems. In *Proc. AIAA Atmospheric Flight Mechanics Conference*, Toronto, ON, 2010.

[47] Michael Ward, Carlos Montalvo, and Mark Costello. Performance characteristics of an autonomous airdrop system in realistic wind environments. In *Proc. AIAA Atmospheric Flight Mechanics Conference*, Toronto, ON, 2010.

[48] Alan S. Willsky. A survey of design methods for failure detection in dynamic systems. *Automatica*, 12(6):601–611, 1976.