TESTING THE EFFECTS OF VIOLATING COMPONENT AXIOMS IN VALIDATION OF COMPLEX AIRCRAFT SYSTEMS

A Thesis
Presented to
The Academic Faculty

By

Aparna Kansal

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Aerospace Engineering

Georgia Institute of Technology

December 2014

TESTING THE EFFECTS OF VIOLATING COMPONENT AXIOMS IN VALIDATION OF COMPLEX AIRCRAFT SYSTEMS

Approved by:

Dr. Amy R. Pritchett, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Brian J. German
School of Aerospace Engineering
Georgia Institute of Technology

Mr. Curtis E. Hanson
Flight Research Center
NASA Armstrong

Date Approved: November 26, 2014

To my parents and my husband, my greatest strengths.

ACKNOWLEDGEMENTS

I would like to use this opportunity to thank everyone who has supported me through the course of my Master's Thesis. First of all, my advisor, Dr. Amy Pritchett, thank you for taking me in as a part of your lab, for all the opportunities you gave me, and for all your constant support and encouragement. I am grateful for the opportunity I had to work with you and to get to know you personally. I would like to thank my committee members, Dr. Brian German and Mr. Curt Hanson for your helpful feedback on my work and for supporting the project.

Next, I wish to thank my CEC labmates without whom this experience would never have been this wonderful. Anil and Can, the first people I met when I joined the lab, who helped me get accustomed to the workings of the lab when I was new and gave me a hearty welcome into the CEC family. Alex and Scottie-Beth, being the oldest members of the lab, were always there when I needed someone to talk to and was looking for encouragement. Thanks for being there and for all the fun times we've had as a part of the CEC women's club. My labmates who were involved in making important updates to WMC which helped me complete my work successfully, including Anil, Can, Sebastien, Arvind and Raunak. Also, everyone in the lab who have been there to help me practice my presentations and given valuable feedback, been there at every presentation as pillars of support and helped me keep my nerves by giving me silent encouragement during my presentations. You all have been great friends and a huge source of support for me.

My wonderful roommates Bhavna and Suchitra, who have become my closest friends and have been my biggest support here. All those hours that we spent talking, tackling all the hardest times together, finding ways to cheer each other up and even the

silly fights we had sometimes; I am grateful for all that. All the wonderful friends that I have made here in Atlanta will always be a big part of my life and I am grateful for getting the chance to meet you and spend all this time with you. I will take back lovely memories of my time here at Tech, which has been this special because of you.

To my family and friends back home who have always encouraged me to pursue my dreams and although they missed my presence, they were always there for me throughout my journey. My parents and my now husband, Abhilash, who have called me every day over skype for the past 3 years while I've been here, patiently waiting for me to finish my degree, seeing to it that I am doing fine and keeping up my determination in the most challenging times. My brother, Siddharth and my best friend, Rima, for your messages from time to time which kept me in touch with the home I missed so much. To my Aunt, Charul bua and her family who are my closest family in the US, and who's place has been my regular vacation destination during my time here. I am grateful I had a home here, where I could visit anytime I felt homesick. My parents-in-law, my second set of parents, who have been my support when I was struggling to complete my thesis from India after my wedding, and because of whose support I came back and gave the final push to complete my degree.

Finally, I would like to thank Georgia Tech, which has not just given me wonderful memories to take back, but also made me face all kinds of challenges, thus making me a more mature and confident person, giving me even bigger dreams and challenging me to achieve greater heights in the years ahead.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This thesis focuses on estimating faults in complex large-scale integrated aircraft systems, especially where they interact with, and control, the aircraft dynamics. A general assumption considered in the reliability of such systems is that any component level fault will be monitored, detected and corrected by some fault management capability. However, a reliance on fault management assumes not only that it can detect and manage all faults, but also that it can do so in sufficient time to recover from any deviation in the aircraft dynamics and flight path.

Testing for system-level effects is important to ensure better reliability of aircraft systems. However, with existing methods for validation of complex aircraft systems, it is difficult and impractical to set up a finite test suite to enable testing and integration of all the components of a complex system. The difficulty lies in the cost of modelling every aspect of every component given the large number of test cases required for sufficient coverage. Just having a good simulator, or increasing the number of test cases is not sufficient; it is also important to know which simulation runs to conduct. For this purpose, the thesis proposes simulating faults in the system through the violation of "axiomatic conditions" of the system components, which are conditions on the functioning of these components introduced during their development. The thesis studies the effect, on the aircraft dynamics, of simulating such faults when reference models of the components representing their key functions are integrated.

1 INTRODUCTION

1.1 Problem Statement

This thesis proposes a simulation-based method to streamline the validation of complex large-scale integrated aircraft systems, particularly those involving adaptive controllers of the aircraft dynamics. Such systems can fail when the integration of their many components causes unexpected, undesired interactions between their individual behaviors. Such interactions may occur when a failure in one component cascades through the system; further, such a failure may even occur when all components are functioning correctly, but do not collectively provide the desired system performance. A general assumption considered in the reliability of aircraft systems with adaptive control is that any component level fault will be monitored, detected and corrected by some fault management capability. However, a reliance on fault management assumes not only that it can detect and manage all faults, but also that it can do so in a time span lesser than the time in which the aircraft dynamics and adaptive control could be affected by the fault. Thus, this thesis examines how faults can ripple through a system by simulating the key dynamics within the system before and after any fault management may resolve them.

Testing for system-level effects is important to ensure better reliability of aircraft systems. However, with existing methods for validation of complex aircraft systems, it is difficult and impractical to set up a finite test suite to enable testing and integration of all the components of a complex system. The difficulty lies in the cost of modelling every aspect of every component given the large number of test cases required for sufficient coverage. Thus, it is not sufficient to just have a good simulator, or to increase the detail

of the models of the components in the system being tested, or to increase the number of test cases; it is also important to know which simulation runs to conduct. This requires some insight into when a component will fail, or when a component may be technically working but might not meet the requirements of other components, causing the system as a whole to fail. Thus, this thesis provides a method to understand when the system will fail and identify which test cases should be run.

1.2 Some Key Definitions

For the purpose of this thesis we define the following relevant terms:

System: Integrated set of components integrated that accomplish a defined objective.

Components: Individual elements of a system which serve specific functions and work in collaboration with other components in the system to achieve an overall goal.

Complex Systems: Systems consisting of components without a centralized control, such that addition of another component into the system can increase the interactions between the components non-linearly.

Component Functions: Tasks to be performed by a component of a system to enable the system to achieve its overall objective.

Axiomatic Set of Conditions: Logical operations on the internal states and external conditions of a component of a complex aircraft system that identify whether the component is guaranteed to perform its intended functions as expected.

Fault: A condition in which a component is unable to perform its function as intended.

Fault Detection: Recognizing that a fault has occurred in a component of the system.

Fault Management: Action involving managing the fault after it has been detected by notifying relevant components about the occurrence of the fault, which in turn would

apply the necessary corrective actions to recover the aircraft behavior to the intended state.

Validation: (1) *INCOSE SE Handbook v. 3.2.2 – Appendix D:* Confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled [ISO 9000: 2000].

(2) *SAE ARP4754 (2.2 Definitions):* The determination that the requirements for a product are correct and complete. [Are we building the right aircraft/ system/ function/ item?]

Verification: (1) *INCOSE SE Handbook v. 3.2.2 – Appendix D:* Confirmation, through the provision of objective evidence, that specified requirements have been fulfilled [ISO 9000: 2000]

(2) *SAE ARP4754 (2.2 Definitions):* The evaluation of an implementation of requirements to determine that they have been met. [Did we build the aircraft/ system/ function/ item right?]

1.3 Motivation

Distributed architectures are becoming increasingly popular in aircraft systems due to their ease of installation, configuration and subsequent updating. Distributed systems consist of several components that cooperate to achieve a common goal. The asynchronous execution of their processes without a central top-down control, however, can cause the emergence of unexpected behavior at the system level.[1]

Each of the components in a complex distributed system individually has ranges of conditions under which it can operate, such as limits on bank angle, maximum speed, environmental conditions, etc. The limits on one component may be different from those

on another component in the system. They may also have certain assumptions made during their design, like known aircraft behavior in response to a given control input, or certain functions being assumed to be executed by another component. These so called "axiomatic set of conditions" can thus affect the behavior of the system as a whole when the integrated components are in operation. Hence, it is important to validate these components by validating the axiomatic conditions in the integrated environment: intensive testing and validation of just the individual components may not be sufficient to see these system-wide effects.

1.3.1 Study of Aircraft Accidents and Incidents

The following examples show how the violation of the axiomatic set of conditions of a component can affect the system in situations with missing or delayed inputs, a component fault, adverse environmental conditions, exceeding the operating range of a component, etc. These examples have been analyzed by understanding the system and components involved in the accident causation, their functions in regard to the task that was to be performed, the way in which the components were integrated and communicated with each other, the axiomatic set of conditions of the components; the violation of which led to the accidents, and the flight and environmental conditions under which this violation of axioms occurred.

1. **Bombardier Learjet 60 Runway overrun (September 19, 2008):** This accident developed from a failure of the main landing gear to the aircraft overrunning the runway during takeoff due to the failure of axiomatic conditions of the thrust reverser and Full Authority Digital Engine Control (FADEC) systems.[2,3]

Components: Relevant components include the thrust reversers, sensors for the thrust reverser system, electronic engine control computers, the Full Authority Digital Engine Control (FADEC) system and the pilot or flight crew.

Component Functions and Relationships: The thrust reversers, when deployed, provide deceleration force to assist with ground braking. The pilot can command forward or reverse thrust by operating the thrust levers. The electronic engine control (EEC) computers obtain information about the thrust lever angle, and provide corresponding electronic signals to each engine's full authority digital electronic control (FADEC) components. Based on the signals received from the EECs, the FADEC components, which perform functions such as thrust management and compressor surge control, regulate engine output to provide the level of engine power commanded by the pilot. The thrust reverser system applies logic control functions to prevent certain operations based on specific sensor inputs. These include squat switches located on the main landing gears which signal whether the airplane is on the ground, sensors to confirm that thrust reverser doors are fully open to release the thrust reverser levers, and microswitches to indicate that the thrust reverser levers are lifted before signaling the EEC and FADEC components to use reverse thrust engine power.

Axiomatic conditions: The thrust reverser system requires a positive signal from the squat switches on the main landing gear that the landing gear is on the ground. Hence, if the signal is missing the airplane is assumed to be in "air mode" and the thrust reversers cannot be deployed. The EECs, upon receiving the signal that thrust reversers are stowed and squat switches indicate air mode, gives a signal to

the FADEC components to change the engine thrust output to forward thrust, accelerating the aircraft.

Flight Conditions: Insufficient inflation in the right main landing gear caused its disintegration after the pilots initiated the takeoff roll and the aircraft passed the takeoff decision speed (V_1), that is, the maximum speed at which the airplane can be stopped within the accelerate-stop distance if brakes are applied. The signal from the squat switches on the main landing gear was missing due to the damage. The pilots attempted to abort takeoff after V_1.

System Behavior: The rupture of the landing gear hindered the pilots' ability to apply brakes. The thrust reverser system overrode the pilots' command to deploy thrust reversers, as the axiom requiring a positive signal from the squat switches on the main landing gear was violated. It also commanded the FADEC components to go into forward thrust mode, as no signal from squat switches was assumed to mean the aircraft was in air mode, causing the airplane to accelerate, overrun the runway and crash.

2. **In-flight Upset Flight 9M-MRG, Boeing 777-200 (August 1, 2005):** Erroneous vertical, lateral and longitudinal acceleration data from the aircraft's Air Data Inertial Reference Unit (ADIRU) caused a pitch upset event while the aircraft was climbing through 38,000 feet in a flight from Perth, Australia to Kuala Lumpur, Malaysia. A low speed and overspeed warning was reported simultaneously and the aircraft pitched up and climbed with a high vertical speed. In response, the crew had to return to Perth with the autopilot disengaged.[4]

Components: The components involved were the aircraft's ADIRU, the Engine Indicating and Crew Alerting System (EICAS), the autopilot and autothrottle.

Component Functions and Relationships: The function of the ADIRU is to provide air data and inertial reference data to several systems on the aircraft, including the primary flight control systems, the autopilot flight director system and the flight management system. The air data modules receive pressure inputs from the pitot probes and static ports and use it to calculate and supply air data information to the user systems. The ADIRU incorporates six gyros and six accelerometers to calculate inertial reference and navigation data for the other aircraft systems.

The engine indication and crew alerting system, or EICAS, displays all engine and aircraft system information required by the crew. The EICAS, in the operational mode, displays primary engine parameters and alert messages for monitoring by the crew in the upper display unit (DU), while the lower display unit shows secondary engine parameters. The alert messages on the upper DU are divided by priority into warnings, cautions, advisories, and data or communication and displayed in different colors.

The autopilot flight director system (AFDS) has a mode control panel (MCP) and three autopilot flight director computers. The MCP controls the autopilot, flight director, altitude alert, and autothrottle systems. It is used to select and activate the various flight modes, and establish altitudes, speeds, and climb/descent profiles. The flight director computers control the flight directors and autopilot. The flight director information is displayed on the primary flight displays (PFD).

The autothrottle mode is engaged using a push-button switch for various pitch modes or, if no pitch mode is selected, in the speed mode. It can be disconnected by moving the throttle arm switch to off or by pushing an autothrottle disconnect push-button type switches on the engine thrust levers. Upon disconnecting the autothrottle, the EICAS displays a message saying "AUTOTHROTTLE DISC" and the master caution lights come on. When the autothrottle is armed it automatically activates if the autopilot is not engaged and the airspeed is less than what was commanded, or the thrust is below that required for the selected mode of flight.

Axiomatic conditions: The ADIRU is a fault tolerant system with internal redundancies that automatically makes allowances for internal component faults to ensure the unit's overall functionality. It contains seven fault containment areas with fault containment modules which are physically and electrically separated from the other modules. The aircraft ADIRU is designed with system redundancy to prevent any malfunctions from occurring. With only one erroneous input, the system is designed to automatically stop accepting that input and divert to another input source for information. That event does not require any action by the flight crew, and intends to minimize the number of checklist items that a crew needs to maintain. With multiple erroneous sources of information or internal failures in the ADIRU, the EICAS message NAV AIR DATA SYS is displayed. That directs the crew to the appropriate checklist and the unreliable airspeed table.

Flight Conditions: On August 1, 2005 the Boeing 777-200 aircraft was climbing out through flight level (FL) 380 when the crew observed a LOW AIRSPEED

warning advisory on the aircraft's EICAS. One of the accelerometers of the ADIRU had failed in June 2001, but was still capable of producing high acceleration values that were erroneous. The ADIRU was excluding the failed accelerometer in its acceleration computations. Another accelerometer failed during flight, causing the ADIRU to use the previously failed accelerometer information with its high output values in its computations, resulting in erroneous acceleration outputs into the flight control system. Hence, a latent software anomaly allowed the ADIRU to again utilize the previously failed accelerometer's erroneous information into the flight control system.

System Behavior: The aircraft's EICAS showed a LOW AIRSPEED warning as the aircraft climbed through FL380. At the same time it was found that the aircraft's slip/skid indication deflected to the full right position on the PFD, indicating that the aircraft is out of trim in the yaw axis. The PFD speed tape then indicated that the aircraft was approaching the overspeed limit and stall speed limit simultaneously. The aircraft nose then pitched up and the aircraft climbed to FL410. The indicated airspeed then decreased from 270 to 158 knots and the stall warning and stick shaker devices activated. The pilot in command then disconnected the autopilot and lowered the nose of the aircraft. The aircraft autothrottle then commanded an increase in thrust, which was countered by moving the thrust levers to idle manually. The aircraft nose pitched up again and the aircraft climbed 2,000 feet. The pilots subsequently requested from air traffic control a return to Perth. The PFD indications appeared normal while descending through FL200 with the autopilot disengaged. However, when the LEFT autopilot

was turned ‘ON’, the aircraft banked to the right and the nose pitched down. Similar result was seen when the RIGHT autopilot was selected. Hence, the pilots disengaged the autopilot and flew the aircraft manually.

This case was a violation of the axiom of the ADIRU that several redundancies would ensure that the unit would function even if one of its components failed. Further, even though the EICAS displayed an ADIRU status message indicating a fault with the ADIRU, the flight crew was not provided with information that detailed the fault.

3. **Rudder Reversal USAir Flight 427, Boeing 737-300 (September 8, 1994):** In this accident, a B737 aircraft entered an uncontrolled descent and impacted terrain near Aliquippa, Pennsylvania. The accident investigation found that the rudder had reached its blowdown limit because of a power control unit servo valve jam which caused the rudder to deflect in the opposite direction to that commanded by the pilots.[2,5]

   Components: The components in the rudder control system which contributed to this scenario were the flight control system, auto-flight system, yaw damper system, main rudder power control unit (PCU) and PCU servo valve.

   Component Functions and Relationships: Flight Control about the vertical or directional (yaw) axis is provided by a single panel rudder, which is operated by moving either the right or left rudder pedal forward or aft. Flight control about the longitudinal (roll) axis is provided by an aileron on each wing assisted by two spoilers, which are operated by rotating the control wheel clockwise or

counterclockwise. Motion about the yaw and roll axes interact in flight. Hence, any yawing action (sideslip) causes the airplane to roll unless countered by ailerons. The auto-flight system provides control commands to the airplane's ailerons, flight spoilers, horizontal stabilizer and elevators to reduce pilot workload and provide smoother flight; however, it does not provide commands to the rudder system. The yaw damper system automatically stabilizes the airplane about its yaw axis by limiting yaw motions caused by atmospheric disturbances or the airplane. The rudder panel is actuated by a single hydraulic power control unit (PCU). A standby rudder actuator is available if the hydraulic system fails. The maximum amount of rudder travel available for an airplane at a given flight condition/ configuration is known as the "blowdown" limit. The main rudder PCU converts either a mechanical input from the rudder pedals or electrical signal from the yaw damper system into the motion of the rudder by means of mechanical linkages and has a servo valve that directs hydraulic fluid either to extend or retract the PCU actuator rod that moves the hinged rudder surface. The main PCU servo valve is a dual-concentric tandem valve composed of a primary slide that moves within a secondary slide that, in turn, moves within the servo valve housing. These slides translate inputs from the yaw damper and/or external input crank (from rudder pedals) into axial movement of the slides. The clearances between the slides and between the secondary slide and housing are very small.

Axiomatic Conditions: According to the design and working of the rudder power control unit, even in an abnormal condition such as a servo valve jam, a rudder pedal input resisting the jam is assumed to cause the rudder to move in a direction

opposite to the jam, towards neutral position, thus reducing the deflection of the rudder. Hence, a control reversal of the rudder was not considered as a possibility.

Flight Conditions: The aircraft was maneuvering to land at Pittsburgh International Airport with a left turn at an airspeed of 190 knots and altitude of 6000 feet MSL. There may have been wake turbulence from another aircraft. The pilots were trying to roll out of the turn to the intended heading.

System Behavior: The aircraft continued to roll left rapidly, although the pilots were trying to arrest the turn, and eventually crashed into terrain. Tests during the accident investigation revealed that, when the secondary slide was jammed to the servo valve housing at certain positions, the primary slide could travel beyond its intended stop position because of bending or twisting of the PCU's internal input linkages. This deflection allowed the primary slide to move to a position at which the PCU commanded the rudder in the direction opposite of the intended command (reversal). Specifically, the tests revealed that, when the secondary slide was jammed at positions greater than 50 percent off neutral toward the extend or retract position and a full-rate command was applied to the PCU, the rudder would move opposite to the commanded direction.

4. **Future Design of an Aircraft Adaptive Control System:** In an attempt to make aircraft control systems more intelligent and be quickly able to adapt to any sudden changes in aircraft dynamics, there is an ongoing research in the development of adaptive control systems. A discussion with researchers on such systems led to the realization that these systems work under the assumption that

the direction of aircraft motion is always known for a given control input. For example, it is assumed that a left rudder input will always cause the aircraft to yaw towards the left. However, there have been several known scenarios of control reversals, as noted in the above rudder reversal example. It is further assumed that a fault, if any, would be handled by a fault management system.

System Components: The components considered for a control reversal scenario in this case could be the aircraft adaptive control and the fault management.

Component Functions and Relationships: The adaptive control system is used to adapt to abrupt changes in the aircraft dynamics or changes in aerodynamic system parameters when subjected to system faults or structural damage, and to stabilize the aircraft. The fault management system is a system of sensors, detectors and actuators for fault detection and transmission of feedback signals to the aircraft adaptive control system.

Axiomatic Conditions: One of the key assumptions of an adaptive control system is that the adaptive controller knows the sign (positive or negative) of the relationship between a target state and the control actuator used to achieve it, and that, if there is any fault, it would be detected and addressed by some fault management system.

System Behavior Study: As has been seen in the previous example and many other such cases, control reversals need to be considered in the design and reliability testing of future systems. Further, the requirement that the fault management can not only detect the control reversal in theory, but also that it can detect the reversal before the adaptive control places the aircraft in an

unrecoverable state, needs to be evaluated. Possible system behavior in such scenarios can be observed and understood only when these components are integrated. In a later chapter, this thesis looks at a similar control reversal scenario as a case study reflecting a violation of the adaptive control system axiom.

In each of the above cases, there is a violation of a fundamental axiom of at least one of the system components that adversely impacts the behavior of other components and the aircraft's dynamics. It can be noted that, even when all the components are working precisely in accordance with their given requirements, the interactions between the different components under certain conditions can cause the system to behave in an adverse way.

1.3.2 Validation and Verification

Complex systems consist of a range of distributed components which are developed and validated independently before integration. There are methods available to test and validate the component software extensively to ensure there are no errors in the code. Of note, hardware-in-the-loop simulation is used in the development and test of complex real-time embedded systems. The complexity of the aircraft dynamics is included in the testing with a computational model. The simulation can also include electrical emulation of sensors and actuators which act as the interface between the plant simulation and the embedded system under test. Human-in-the-loop studies are also done to examine human factors, using simulators which include plant simulations of aerodynamics, engine thrust, environmental conditions, flight control dynamics, etc.

A Functional Hazard Assessment (FHA) is conducted at the beginning of the aircraft/system development cycle. FHA is recommended to be carried out at the system-level as well as aircraft level to identify failure conditions and their effects, classify failure conditions based on the identified effects, and assign safety objectives. The outcome of FHA is the categorization of circumstances and severity of each failure condition along with the rationale for its classification. In this process the safety practitioner must evaluate each function for the impact of functional failure. Failure types include a function failing to be performed, operating earlier or later than it should have, operating out of sequence, being unable to stop operation or a malfunction or degraded function. Further, methods such as Fault Tree Analysis (FTA), Failure Mode and Effect Analysis (FMEA), Common Cause Analysis (CCA), can each also identify specific safety concerns. [6]

The difficulty in testing complex systems is the ability to determine where to start the testing and how much testing is enough to be able to decide that all systems and system components meet the safety requirements. FHA may help get an idea of which conditions may be more hazardous than others. However, to evaluate possible catastrophic failure conditions which may emerge from the interaction of the components, key drivers of the dynamic relationships between components must be identified. For example, if an assumption underlying a component function is violated, the component may still perform its intended functions, but do so in conditions where these functions may cause other components to behave adversely.

Hence, the validation process for complex aircraft systems may be further streamlined by studying the effect on the aircraft dynamics of violating such assumptions,

which are termed as “axiomatic set of conditions”, considered in the development of the system components.

1.4 Approach

This thesis applies system-wide simulation to validate a complex aircraft system, including dynamic emergent behaviors which impact reliability when all the components of the system are integrated. Such a system-level failure can be due to the violation of underlying assumptions that serve as the axiomatic set for each component; these axioms may either disable or disallow the actions of another component or agent, cause the failure of a component at an unexpected time, or modify the system behavior in some unexpected, adverse way. Therefore, this research focuses on the simulation of such complex systems to systematically explore the impact of violating the underlying axioms of the components on the reliability of the system. It thus validates whether all the components of the system and the system as a whole are maintained if any axiom is violated and, further, it validates the requirements for fault detection and fault management applied to the system.

The simulation framework selected should enable the study of emergent behavior of any aircraft system, through an integration of its components, in the event of a violation of any axiomatic conditions. The generalized framework that has thus been employed for the case studies done in this thesis consists of the following key elements as shown in Figure 1.1:

- *Local models of components as simple dynamic representations of their target behavior and functions.* These models do not need to be the complete representation of the component, as it would be when integrated into the actual

aircraft, but instead only need to emulate its intended behavior. For example, a sensor to monitor a particular aircraft state may just be a variable showing the output of the state at certain intervals of time, based on how often the sensor is actually supposed to monitor the state, with any important property such as latency or noise added as appropriate. Thus, components can be simulated earlier in the design process, before their internal behaviors are completely specified.

- *Communication channels between the components, for their integration*, in the form of values or variables taken as input for a component, which may be the outputs from another component.
- *Axiomatic set of conditions for each of the components.* These determine the capabilities of the component, and whether the component functions as intended or not. They may be applied as conditions on the operation of the component, and on their input and output properties.
- *Model representing the aircraft dynamics for the selected flight maneuvers to be simulated for a specific flight path and environmental conditions.*
- *The ability to apply faults or unexpected events as required for the simulation*, particularly those that violate an axiomatic condition of a component.

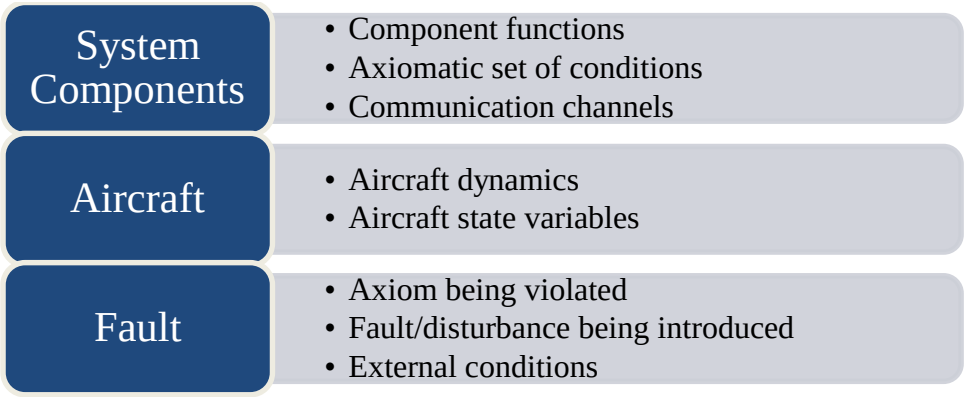

FIGURE 1.1: ELEMENTS OF A GENERIC SIMULATION FRAMEWORK

1.4.1 Identify Component Functions and Axiomatic Set of Conditions

The first step in the approach is to identify the key functions of the individual components of the aircraft system to be studied and create simple dynamic reference models to represent their intended behavior. For example, a fault management system may be represented as an action that detects a fault by checking if a condition is true or not without modeling all the sensors, avionics, etc.

In addition, it is required to model the axiomatic conditions of the components which define required conditions for the functioning of the components. This enables testing the system with violation of any fundamental axiom. The axiomatic set of conditions of a component include the operational limits or range of conditions over which the component would be able to perform its intended function, and the assumptions about its inputs and outputs when interacting with other system components. Thus, the axiomatic conditions may be identified by understanding:

1. The operational parameters known to affect the component state while performing its intended actions.
2. The limits or boundaries of these parameters, under which the component operates without deviation from its normal functioning.

3. External design conditions for the component, such as the severity of environmental conditions within which the component can maintain normal functioning.
4. The output that the component needs to provide to other components or agents, as well as input required from other components.

1.4.2 Identify Potential System-Level Failures

A system may fail due to a fault in any of the components of the system, as has been defined for the purpose of this thesis, under certain environmental conditions and circumstances. This thesis studies, through simulation of scenarios, the potential-system level failure when:

1. A component is placed outside of its allowable environmental condition and it does not respond properly for that condition;
2. All components act as desired, but the integration of the system as a whole fails in a given condition; or
3. One of the components fails to perform its intended function.

1.4.3 Identify Fault Detection and Fault Management Functions

The thesis also observes the effect of a fault recovery after a fault has been introduced. Hence, based on the fault scenario to be studied, the time of fault detection can be controlled in the simulation to emulate a range of available fault detection methods and methods of recovery from the fault.

To observe emergent behavior in the system when the axiomatic condition of any component is violated, the simulation model of the entire system, with all its components

integrated, is run along with the dynamic model of the aircraft. The flight conditions are set as required. At a certain point during the flight, the scripted fault or event may modify conditions, as required, to violate an axiom of a component, with the help of a "fault action". The aircraft state is then monitored. After a certain later time, a corrective "repair action" may be incorporated to mimic fault management functions, which would attempt to bring the aircraft back to its intended state.

To examine the system-wide implications of violating the axiomatic conditions of any of the components, specific scenarios have been selected to demonstrate the possibility of applying this simulation approach to the different ways in which emergent behavior could arise at the system-level when key axioms of the components are violated. This demonstrates how analysis of the axiomatic set of conditions can serve to identify test conditions warranting early attention.

1.5 Objectives and Scope

The objectives that this thesis satisfies through this approach are:

1. Defining a method to capture and describe the underlying assumptions or axiomatic set of conditions of the components within a distributed system.
2. Establishing a simulation framework for validation that can (a) incorporate models of relevant component functions, interactions between the components and a dynamic model of the aircraft, and (b) monitor the key axioms of the components. The simulation thus enables the identification of emergent behavior in a range of scenarios by introducing faults at varying times that violate key axioms, and by varying the duration after which the fault is detected.

3. Demonstrating the ability of the simulation to examine the system-wide implications of violating the axiomatic conditions of any of the components of the integrated system in the vehicle (aircraft) model, as well as actions seeking to repair any adverse conditions, if detected.

2 BACKGROUND

2.1 System Complexity and Integration

2.1.1 Systems and System Elements

A *system* is a combination of interacting elements organized to achieve one or more stated purposes. A member of a set of elements that constitutes the system is a *system element*. These elements may include hardware, software, people, information, techniques, processes, facilities, services, and other support elements. If the system elements are systems themselves, typically the case in large-scale, interdisciplinary problems with multiple, heterogeneous, distributed systems, the system of interest can be termed a *system-of-systems* (SoS). These interoperating collections of component systems usually produce results unachievable by the individual systems alone. The INCOSE Systems Engineering Handbook mentions the challenge of *complexity* associated with the development of SoS where the addition of new system elements increases the number of interactions between the system elements in a non-linear fashion. Complexity also makes it hard to define data exchanges across the interfaces between system elements.[7]

Overall, increased complexity can lead to unexpected and unpredictable system behavior. Among the many definitions of systems engineering, the one selected by the FAA Systems Engineering Manual is: "Systems engineering is a discipline that concentrates on the design and application of the whole (system) as distinct from the parts. It involves looking at a problem in its entirety, taking into account all the facets and all the variables and relating the social to the technical aspect." Thus a systems engineering perspective is needed to find the emergent properties of a complex system.[8]

2.1.2 Distributed Systems and Emergence

For the purpose of this thesis, a *distributed system* may be defined as a system of multiple processing elements or components, cooperating in a common purpose or to achieve a common goal. Distributed processing is defined in the IEEE Systems and Software Engineering Vocabulary as "information processing in which discrete components may be located in different places, and where communication between components may suffer delay or may fail." [9]

Distributed control architectures are becoming popular in aircraft systems due to ease of installation, configuration and updating. However, the asynchronous execution of these processes, without central top-down control, can cause the emergence of unexpected behavior at the system level in the form of error conditions or noise. These are not necessarily random events or caused by unit malfunctions, but can be generated by deterministic interactions among the control elements under certain conditions.[1,10]

2.1.3 Complex System Integration: Software, Hardware and Human Interface

Bloebaum and McGowan contrast in their paper "The design of large-scale complex engineered systems" the nature of complex systems versus complicated systems (such as a television or a computer). They say that a complicated system can be developed using the classical reductionist approaches, where the system may be reduced to its basic elements or parts and then studied as a summation of its parts. On the other hand, in complex systems the interdependent elements exhibit dynamic and reflective intra-system behavior, which nullify the possibility of using reductionist approaches.[11]

Many new hazards are related to the increase in complexity of the new systems being developed. In turn, this increased complexity makes identifying these hazards more difficult. S*ystem accidents,* as termed by Perrow[12], are caused by interactive complexity in the presence of tight coupling. Increased complexity and coupling make it difficult for the designer to consider all the hazards, or even many important ones, or for the operators to handle all normal and abnormal situations and disturbances safely.[13]

Today, computers are used in most systems to provide control functions, especially in safety-critical systems. They often replace traditional hardware safety interlocks and protection systems, or control existing hardware protection devices. New types of computer-related hazards appear in aircraft systems, primarily in flight control systems, navigation systems, and cockpit displays. They add a new dimension to the problem of human error. Some hazards are passive until just the right combination of circumstances arrives. Some result from the crew's multitude of choices in aircraft system management, often during task prioritizing. Computers can be used in safety-critical loops in many ways, with various levels of automation; for example, providing information to a human controller, interpreting data and displaying it to the controller, issuing commands directly but with human monitoring of the computer actions, providing varying levels of input, or eliminating the human from the control loop entirely. Computer-based systems are often meant to relieve pilot workload, which oftentimes perversely leads to complacency and/or lack of situational awareness.[13]

2.2 System Safety and Validation

2.2.1 System Safety and Hazard Analysis

The FAA Systems Safety Handbook defines system safety as a specialty within systems engineering that supports program risk management. The goal of systems safety is to optimize safety by the identification of safety related risks, eliminating or controlling them by design and/or procedures, based on the system safety order of precedence shown in Table 2.1.[14]

TABLE 2.1: SAFETY ORDER OF PRECEDENCE [14]

Description	Priority	Definition
Design for minimum risk	1	Design to eliminate risks. If the identified risk cannot be eliminated, reduce it to an acceptable level through design selection.
Incorporate safety devices	2	If identified risks cannot be eliminated through design selection, reduce the risk via the use of fixed, automatic, or other safety design features or devices. Provisions shall be made for periodic functional checks of safety devices.
Provide warning devices	3	When neither design nor safety devices can effectively eliminate identified risks or adequately reduce risk, devices shall be used to detect the condition and to produce an adequate warning signal. Warning signals and their application shall be designed to minimize the likelihood of inappropriate human reaction and response. Warning signs and placards shall be provided to alert operational and support personnel of such risks as exposure to high voltage and heavy objects.
Develop procedures and training	4	Where it is impractical to eliminate risks through design selection or specific safety and warning devices, procedures and training are used. However, concurrence of authority is usually required when procedures and training are applied to reduce risks of catastrophic, hazardous, major, or critical severity.

System safety is related to the potential unreliability of the system and associated adverse events. Adverse events may potentially contribute to system accidents. Reliability is the probability that a system will perform its intended function satisfactorily

for a prescribed period of time under stipulated environmental conditions. Since nothing is perfectly safe, the objective of system safety is to attain the "optimum degree of safety" by eliminating or controlling known system risk to an acceptable level.

While evaluating risk, contributory hazards are important. Contributory hazards are unsafe acts and conditions with a potential for harm. Unsafe acts are human errors that can occur at any time during the system lifecycle, while unsafe conditions can be failures, malfunctions, faults or system anomalies. An unreliable system may not be hazardous: systems can be designed to be fail-safe, with an assurance that no harm will result from contributory hazards.

Hazard analysis is the process of examining a system throughout its lifecycle to identify inherent safety related risks. To accomplish this, system risks are identified within potential system accident scenarios with associated contributory hazards. Controls are then designed to eliminate or mitigate risks to an acceptable level. Scenario descriptions can vary from general to specific, depending on the detail of knowledge available to the analysis. Specific integrated analyses like human interface analysis, abnormal energy exchange, software hazard analysis and fault hazard analysis are used to evaluate interactions between system elements. These interactions between system elements include interrelations between human, machine, environment and procedures. These methods, along with hazard control analysis, are used to analyze the possibility of insufficient control of the system. The purpose of this analysis is to identify possible adverse deviations which will affect system safety.[15]

2.2.2 Current Standards and Guidelines for Validating Aircraft Components

The process outlined in the SAE standard ARP4754A "Guidelines for Development of Civil Aircraft and Systems" presents the guidelines for developing aircraft-level, system-level, and item-level (or component-level) requirements to establish confidence for aircraft systems as a whole. The process includes validating requirements, verifying that requirements are met, and configuration management and process assurance activities. The safety analysis process is used in conjunction with a development assurance process to identify failure conditions and severity classifications which are used to derive the level of rigor required for development.

According to ARP4754A, complex systems and integrated aircraft level functions present greater risk of development error (requirements determination and design errors) and undesirable, unintended effects. It mentions the impracticality of developing a finite test suite for highly-integrated and complex systems which would conclusively demonstrate that there are no residual development errors. Since these errors are generally not deterministic and suitable numerical methods for characterizing them are not available, other qualitative means have been proposed to establish that the system can satisfy safety objectives. In this context, ARP4754A/ED-79A regards the activities of the RTCA documents DO-178C/ED-12C "Software Considerations in Airborne Systems and Equipment Certification" and DO-254/ED-80 "Design Assurance Guidance for Airborne Electronic Hardware" as a means to implement the development assurance rigor for the software and electronic hardware aspects of the design. However, these software and electronic hardware related processes are no longer considered to be adequate to mitigate aircraft/system errors without a development assurance process from aircraft-level down

to the item-level. Here an item is hardware or software component having bounded and well-defined interfaces, a system is a combination of interrelated items arranged to perform a specific function, and multiple distributed systems combine at the aircraft-level.

Thus, regulatory authorities have highlighted concerns about the efficiency and coverage of the techniques used for assessing safety aspects of highly integrated systems that perform complex and interrelated functions, particularly through the use of electronic technology and software based techniques. The concern is that design and analysis techniques traditionally applied to deterministic risks or to conventional, non-complex systems may not provide adequate safety coverage for more complex systems. Thus, other analysis techniques for assurance, validation and verification may need to be applied at the aircraft-level, or at least across integrated or interacting systems to increase confidence that errors in requirements or design, and integration or interaction effects have been adequately identified and corrected. Revisions have been made to Title 14 Code of Federal Regulations (14CFR) for Airworthiness Standards, the Advisory Circular for System Design and Analysis (AC 25.1309), the European Aviation Safety Agency (EASA) Certification Specification (CS-25), etc. addressing these concerns to include new approaches, both qualitative and quantitative, which may be used to assist in determining and establishing compliance with safety requirements considering the whole aircraft and its systems. It also provides guidance for determining when, or if, particular analyses or development assurance actions should be conducted in the frame of the development and safety assessment processes. Figure 2.1 outlines the relationships between the various development documents, which provide guidelines for safety

assessment, electronic hardware and software life-cycle processes and the system development process described in the ARP4754A document.[16]

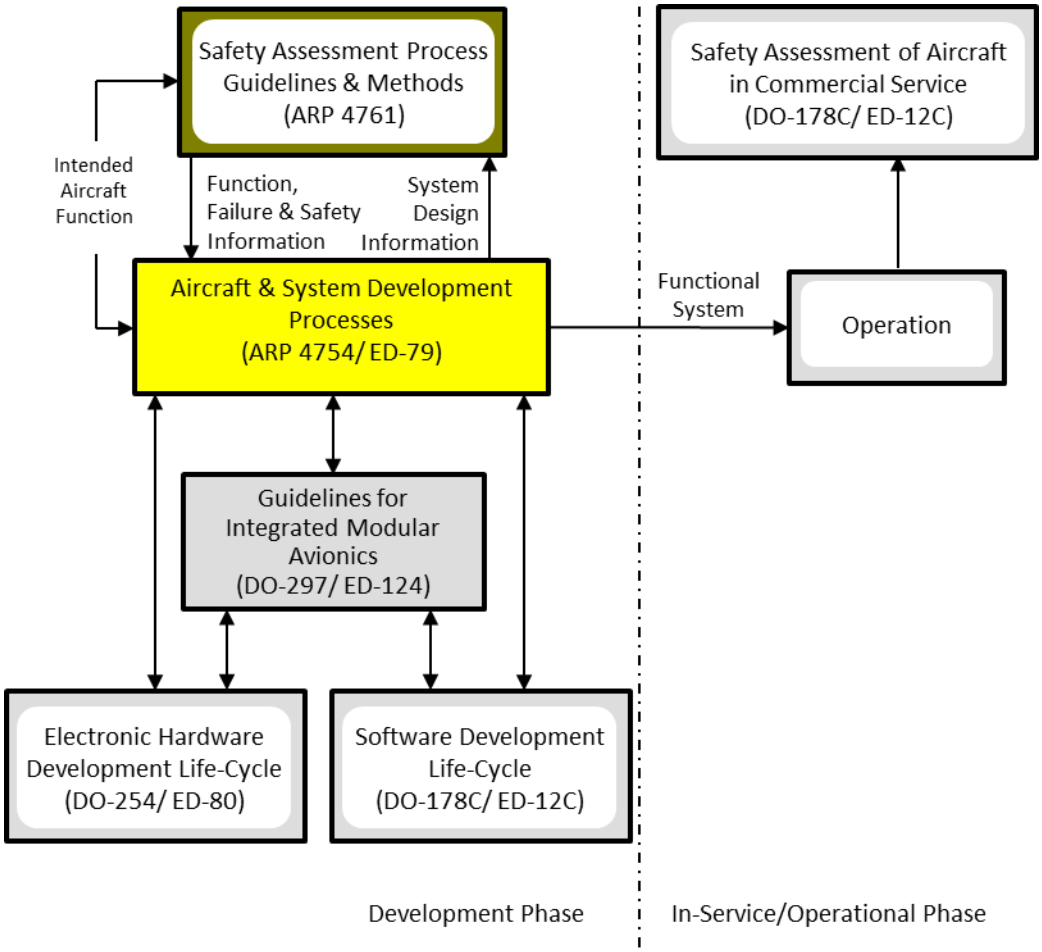

FIGURE 2.1: RELATIONSHIP BETWEEN GUIDANCE DOCUMENTS [16]

2.3 Summary

Although the current guidelines and processes talk about the importance of considering system integration for the validation of components in systems of increasing complexity, the methods available at this time may not be adequate to identify emergent system behavior. ARP4754A discusses the difficulty of developing a finite test suite for highly-integrated and complex systems which would conclusively demonstrate that there are no residual development errors, stating that it may be impractical to do so. They mention that, in such cases, a qualitative analysis may be more practical in order to determine which failure modes must be studied.

This thesis proposes that validation methods for such complex systems can be streamlined by directing their testing towards axioms of the system components, that is, the effect of assumptions and limiting design considerations based on which the components have been designed, and the system-level interactions due to the violation of these axioms under certain conditions. Further, it studies the importance of considering time based impact of faults and fault recovery, which could impact whether the aircraft can recover after the time lag between a fault recovery action and the aircraft recovery.

3 APPLIED SIMULATION ENVIRONMENT

3.1 Work Models that Compute

For the purpose of performing case studies to show the application of the proposed simulation framework, this thesis used the simulation environment "Work Models that Compute" or WMC. WMC is used, primarily, to model complex interplay of activities in complex work environments. The key aspect, work, is the pattern(s) of activity required to meet some goal(s) within a complex environment. This activity has to mirror the demands of the environment: for example, flying an aircraft requires sensing the aircraft state and moving the control surfaces in a manner consistent with the aircraft's dynamics, and picking a route of flight must consider both physical aspects of the environment (e.g., the aircraft performance, winds), and the "intentional" aspects of the environment created by established procedures and regulations (e.g., interacting with air traffic control according to published regulations). Models of work, or *work models*, should represent not only what this activity is, but also when it happens and how it impacts the work environment. [17,18]

Thus, the fundamental aspect of WMC, the *work model*, defines *actions* required to complete the required task, and *resources* to capture the aspects of the environment. *Resources* are the computational structures that capture aspects of the environment. For example, the work model of flying an aircraft would include resources that capture the aircraft's states. A*ctions* respond to the environment by "getting" resource values and then act upon the environment by "setting" resources; a 'flyaircraft' action, for example, gets the resources representing aircraft state and simulates what their next value should

be at the end of a simulation timestep, setting the new, updated values into the resources. In this thesis, the construct of actions is used to emulate component functions.

Work models describe all the actions that need to be done by some agent. Correspondingly, *agent models* are invoked at run-time to handle or execute whatever actions are required of them. A BasicAgentModel is created for each work model that does nothing but pass-through actions that are required (and can log its actions as it goes). More complex agent models can be invoked and assigned specific actions to examine how the human handles all the actions assigned to them (e.g., how many do they have, do they need to delay some) and may examine on-going situation awareness of the resources that these actions get and set.

Once the relevant work model(s) and agent model(s) are ready, then the simulation runs are conducted in the conduct of a *scenario*. A scenario invokes the relevant work models through *script*s. As many work models can be called as desired, including multiple instances of the same work model. Further, several scripts can call upon the same work model for different scenarios. The scenario tells the script the termination criteria for the simulation. Through a script, the scenario creates actions and resources for the work models it invokes, assigns actions to a BasicAgentModel or any additional agent model, schedules the next update times for the actions as desired, changes the initial values of resources, etc.

One of the main benefits of WMC is its ability to strictly control the timing of actions. Thus, the aircraft dynamic model can apply a continuous non-linear 6DOF dynamic model with timesteps of its own choosing, and other components can apply whatever model form (discrete event, expert system, script, etc) that best models their

desired component functions. Some actions can be triggered at pre-scripted times. For example, a scenario can examine the time based impact of faults on the system and the aircraft dynamics by setting when the fault is introduced into the system and when the recovery action is initiated.

Further, rather than passing data between actions, all actions act upon "resource" values stored within the simulation framework; this structure mirrors a common databus as appropriate for some data values; further, it enforces a modular structure on the component models that reduces the re-programming of one component model when another component is changed.

If there is a desire to model any component more accurately than as performance of its nominal function, that would just require adding more actions or adding detail to its established actions. Hence, at a later point in the design stage, more details can be included in the component models and a more detailed study will be possible.

3.1 Aircraft Model in WMC

The methodology applied in this thesis requires a dynamic model of an aircraft with its control system, including all autopilot control loops and a 6 degree-of-motion dynamic model to observe the effect of violating axiomatic conditions involving flight control. WMC provides a built in model of a flight control system of a large transport aircraft with a generic adaptive controller. It uses an adaptive control architecture for inner-loop attitude control to create a stable control behavior that follows a given reference model. This model allows the specification of the closed-loop behavior of the aircraft and the controller (pilot or autopilot). Further, the outer-loop guidance generates

the attitude commands to the inner-loop element to achieve a specified velocity, heading and altitude.

3.2 Applying Component Models in WMC

Application of the approach proposed in this thesis requires integration of several components to simulate their functions and their interactions. These components are further integrated with the model of the aircraft dynamics to study the aircraft behavior during faults and fault recovery. These components are represented by their component functions within WMC. Some of the component functions modeled within WMC are:

Aircraft Adaptive Control: The function of an adaptive control system is to adapt to any change in dynamics to stabilize the aircraft. It is normally difficult to implement closed-loop invariant behavior associated with manual aircraft control as well as models of automated flight control system behavior into simulations without inverting the aircraft dynamics. With recent advances in adaptive control, the desired closed-loop inner-loop dynamics of an aircraft and its control system can be obtained by specifying them as reference models. This has been implemented in WMC by using an adaptive control architecture that allows pilot or flight control system closed-loop inner-loop dynamics to be specified in the control of detailed, non-linear aircraft dynamic models. It eliminates the adaptive elements of the control architecture of the aircraft in the simulation and provides a simple, generic implementation suitable for many situations. [19]

Fault Management: The fault management component includes fault detection, communication of the fault to the relevant components and aircraft recovery from the fault. The component function itself does not necessarily require any detailed modeling of specific electronic or physical components. Instead, the fault detection function can be

simulated through actions scheduled at fixed times which indicate when the fault has been detected and schedule the recovery action. Hence, the effect of the time required for fault detection can be studied. Likewise, depending on the scenario being studied, the fault recovery action can be defined as acting on the components which need to change their behavior to try to recover the aircraft state. The fault recovery action may be performed by the adaptive control system, or by autopilot or autothrottle functions which may disregard certain erroneous values, or by a pilot who may disable the autopilot or change autoflight modes or change values in the mode control panel, etc.

Autoflight Functions: The autopilot and autothrottle flight modes have been modeled in WMC as control loops which can be enabled or disabled to engage or disengage different flight modes. They also allow setting of values of the aircraft state that can be commanded to be maintained for the different flight modes. The aircraft model then changes the aircraft dynamics and flight path based on the selected modes and set values. The flight modes in WMC are based on a generic Boeing 747-like aircraft. The outer loops include the altitude, airspeed, vertical speed, flight level change, flight path angle and heading loops, while the inner loops include the phi (roll angle), beta (side slip), theta (pitch angle) and thrust loops. The altitude loop sets the pitch attitude and vertical speed to reach the commanded altitude. The airspeed loop sets the commanded thrust to maintain a commanded indicated airspeed. The flight level change (flch) loop varies the pitch to maintain a commanded indicated airspeed, with a constant thrust commanded using the thrust loop. The vertical speed (vspeed) loop varies the pitch to maintain a specified rate of climb or descent. The flight path angle loop maintains a specified flight path angle for climb or descend by adjusting the pitch. The heading loop

varies the roll angle to reach a commanded heading. Selection and change of these control loops to simulate the selection of autopilot and autothrottle modes and setting commanded values as done via a mode control panel (MCP), by using the changeControlLoop function in a script or work model, or through an action to schedule the time at which a flight mode is changed during the simulation.

Pilot Operations: The case studies performed for this thesis do not model pilot behavior in the aggregate. Instead, specific tasks to be done by the pilot are scheduled through actions. For example, the action of a pilot changing flight modes or entering values into the MCP is modeled through an action to change control loops at scheduled times in the simulation. Error in performing the tasks can also be scheduled, as can actions to recover from a fault by enabling or disabling flight modes or changing commanded values, etc.

3.3 Configuring WMC

Any scenario to be simulated is configured using scripts which invoke the necessary work models and schedule actions representing component functions. In the fault scenarios simulated for this thesis, the script specifies the start and end time of the simulation and provides termination criteria for the simulation. It then invokes the work model to fly an aircraft. This work model contains a flyaircraft action which enables all the control loops and simulates the aircraft dynamics. The script which runs the fault scenario or case study specifies the initial values for the aircraft state.

Axiomatic conditions of the components are then violated by invoking an action introducing a fault. An action is created to take corrective action when the fault is detected to simulate fault management functions. The fault detection is scheduled, and as

desired, any fault recovery. The aircraft state is monitored by getting the values of the resources which define the values of the aircraft state variables at each timestep.

4 CASE STUDY: ELEVATOR CONTROL REVERSAL

Control reversals have been known to occur in certain flight conditions, like the rudder control reversal of the USAir flight 427 noted in Chapter 1, or some combination of structural malfunction/degradation and aeroelastic effects. This case study was motivated by contrasting such control reversal incidents with the underlying axiom of the most proposed adaptive control systems that assume the sign of the relationship between the elevator deflection and pitch would always be known. This axiom thus also requires fault detection and fault recovery functions.

This hypothetical case study looked at an elevator control reversal scenario, and thus, demonstrated the effect of the aircraft's response to violation of a component's (adaptive control) axiomatic condition due to being placed outside its allowable environmental condition. Applying the proposed simulation framework, the study looked at the effects of an elevator reversal with increasing fault duration before a fault detection and recovery action was invoked.

4.1 System Components

The components of the aircraft system, considered for this scenario, were the aircraft adaptive control system and the fault management system, as described earlier in Chapter 3.

4.1.1 Component Functions

The aircraft adaptive control system emulates all aircraft control surfaces, connecting linkages, and automated control through the Flight Control Unit. It functions to control the aircraft dynamics and adapt to any change in the dynamics to maintain the aircraft's stability. The fault management function monitors and detects unexpected behavior in the aircraft dynamics, and notifies the adaptive control system once it detects a fault so that the control system can correct its internal assumption of the sign of the relationship between control input and resulting motion.

4.1.2 Axiomatic Conditions

The key axiom of the adaptive control system is that the direction of motion of the aircraft in response to a given control actuator input is always known. It also assumes that any control reversal, which would be termed as an abnormal condition, would be handled by the fault management system through fault detection and reporting to the adaptive control system.

4.1.3 Component Interactions

Given a control input to the elevator, the direction of the resulting pitching moment is known by the aircraft control. The fault management checks whether the direction (sign) is appropriate or reversed. In case of a fault in the direction of the aircraft pitching moment, a resource variable which takes a boolean value of TRUE/FALSE is used for the fault management to convey the fault to the adaptive control for corrective action.

4.2 Simulation Execution

4.2.1 Scenario

The aircraft is in a continuous descent from an altitude of 10,000 feet with airspeed of 250 knots and a vertical speed of 700 fpm. An indicated airspeed of 230 knots is then commanded at the start of the flight. The aircraft's normal flight profile without any fault is as seen in Figure 4.1.

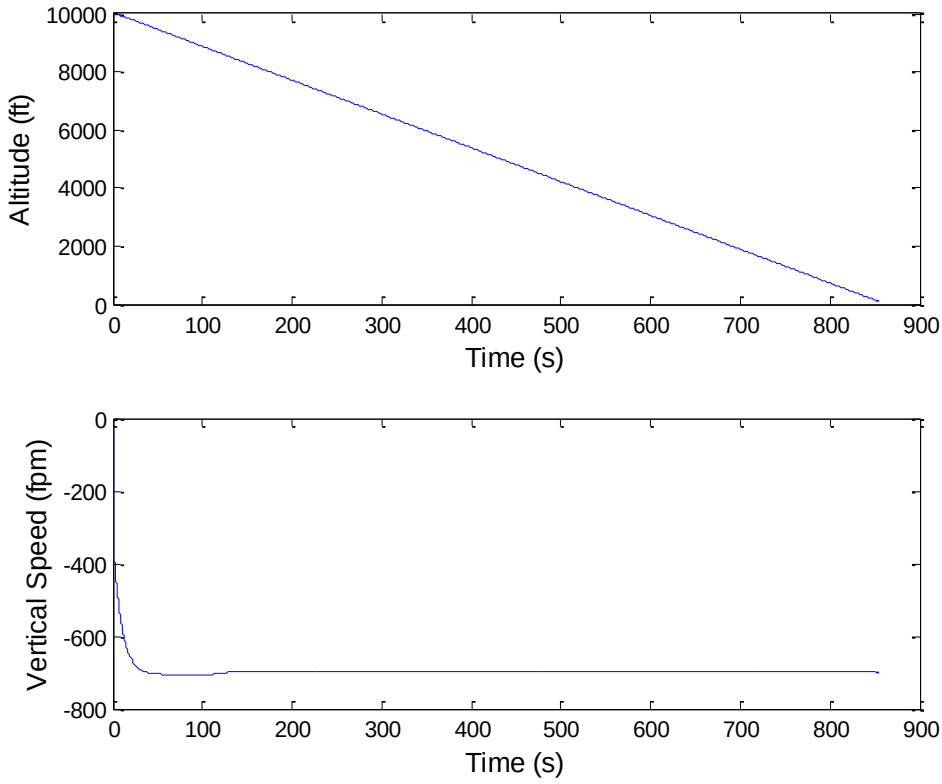

FIGURE 4.1: FLIGHT PROFILE FOR AIRCRAFT IN DESCENT

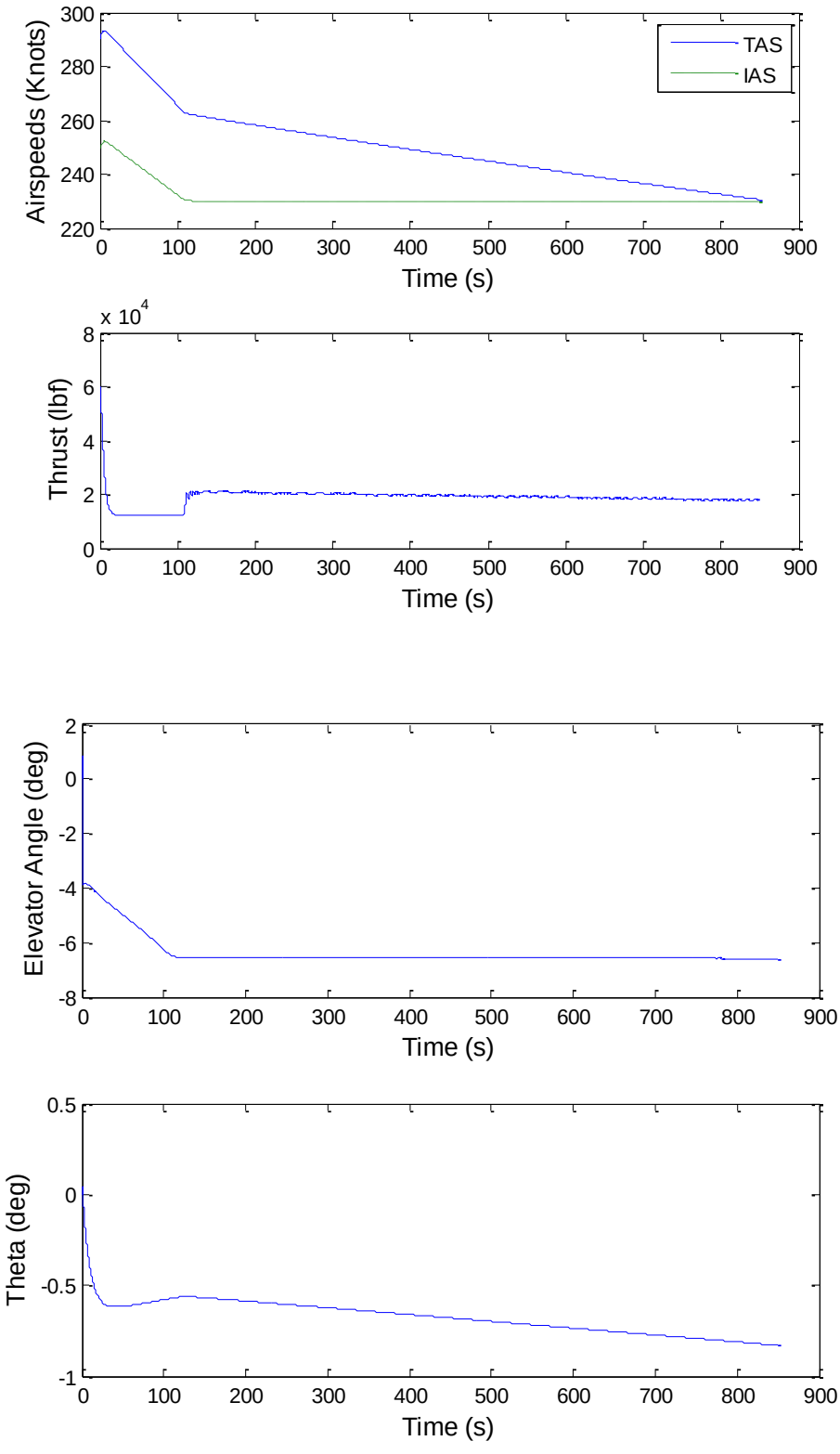

FIGURE 4.1 CONTINUED

4.2.2 Fault Introduction and Recovery

One hundred seconds after the start of the simulation, as the aircraft was descending through an altitude of 10,000 feet MSL, slowing to an airspeed of 230 knots, the fault was introduced: the direction of aircraft pitching moment became opposite of what was expected for the given elevator input. This is a violation of one of the axiomatic conditions of the aircraft adaptive control, that is, the assumption that the direction of motion of any control surface is always known for a given control input.

The *fault duration*, that is the time taken for the fault to be detected by fault management and for corrective action to be taken by the adaptive control, was varied. The fault management function alerts the adaptive control system. The adaptive controller then effectively repairs the control reversal.

4.2.3 System Behavior

Figure 4.2 shows the effect of an undetected elevator control reversal on the aircraft. Figure 4.3 shows the aircraft behavior when the fault is detected, conveyed to the adaptive control system and corrected after a given fault duration.

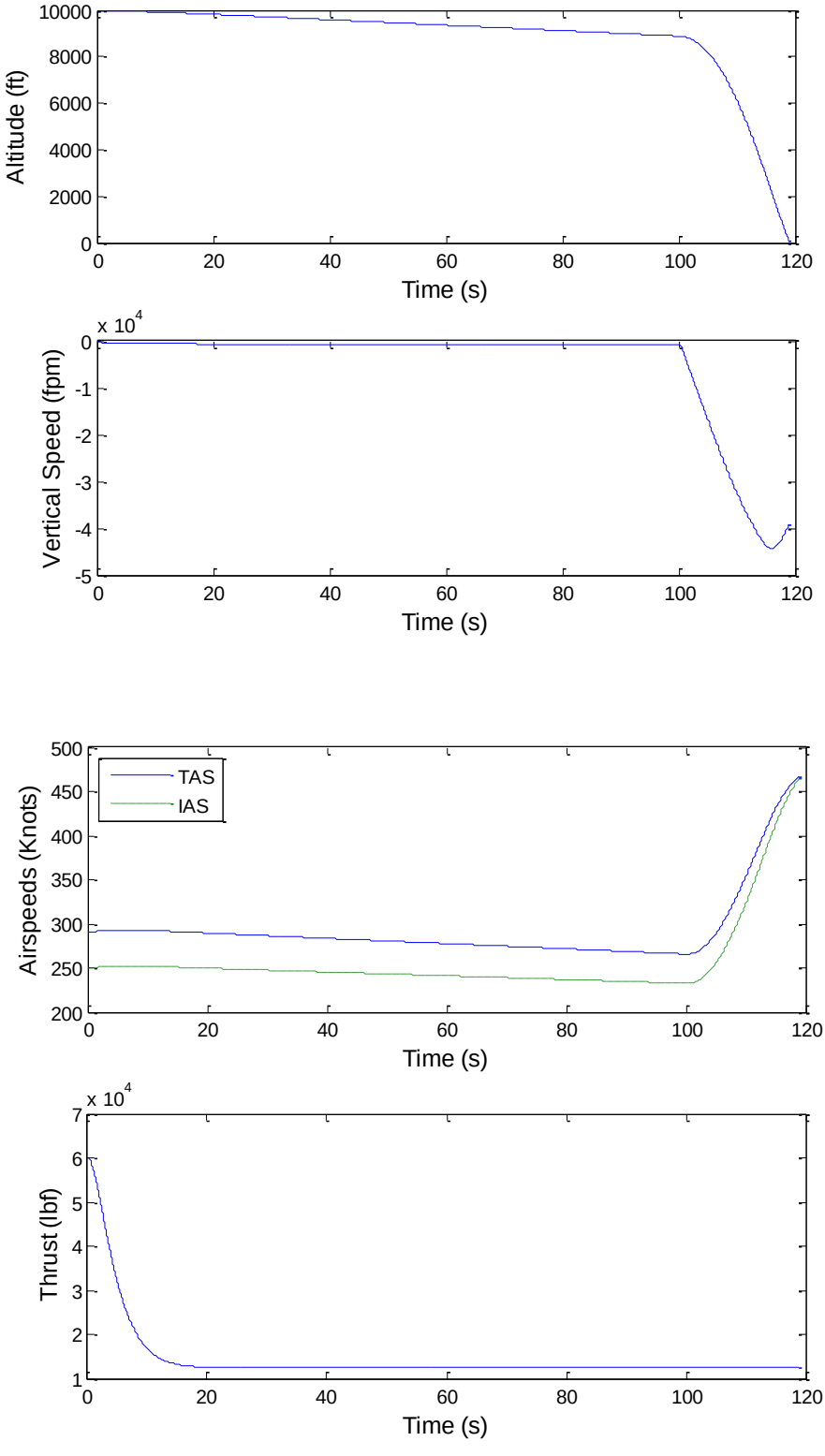

FIGURE 4.2: EFFECT OF ELEVATOR REVERSAL FAULT WHEN FAULT IS INTRODUCED AT 100 SEC

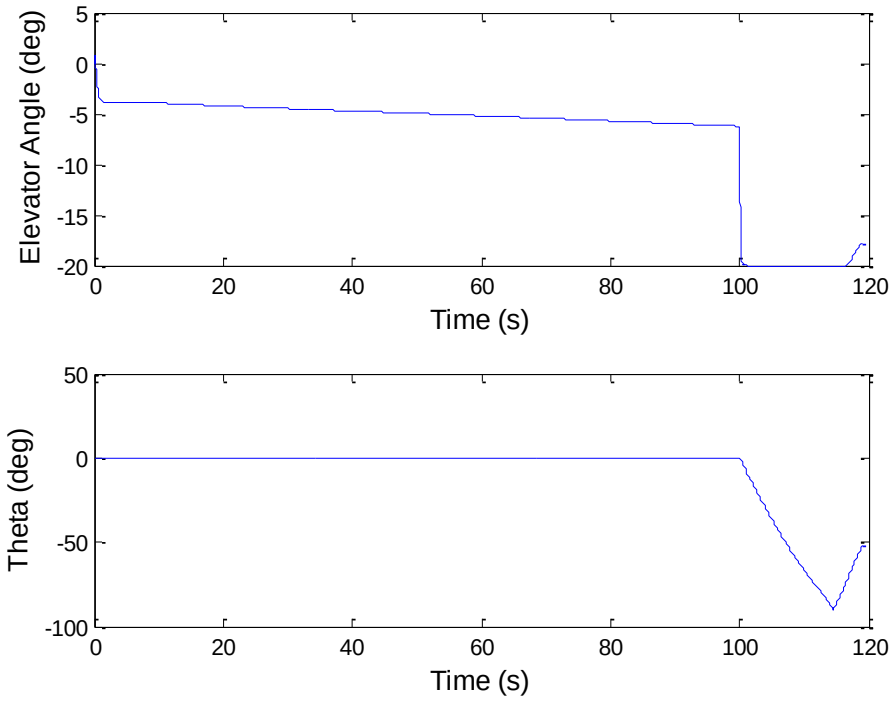

FIGURE 4.2 CONTINUED

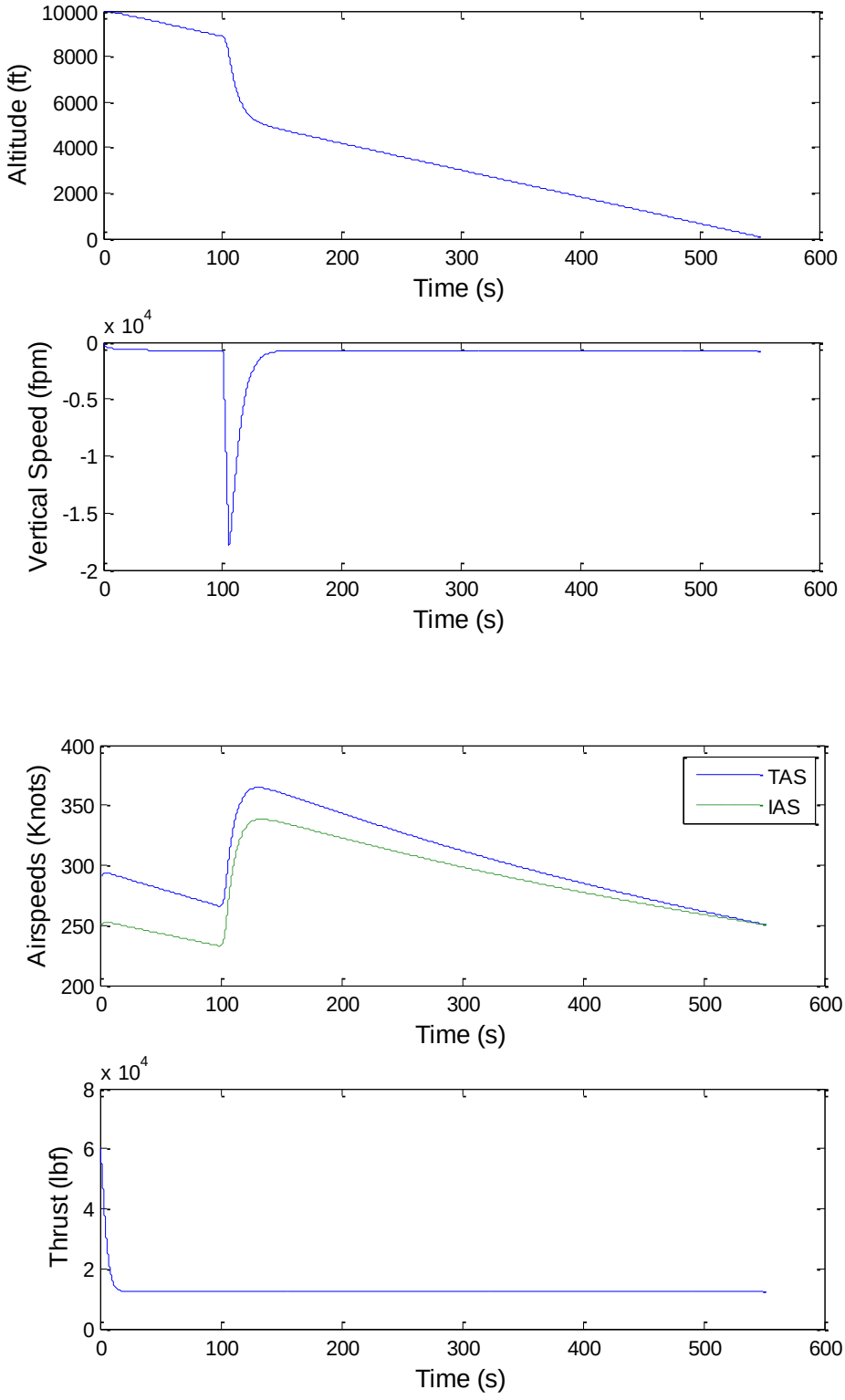

FIGURE 4.3: ELEVATOR REVERSAL FAULT REPAIRED AFTER 5 SEC OF FAULT INTRODUCTION

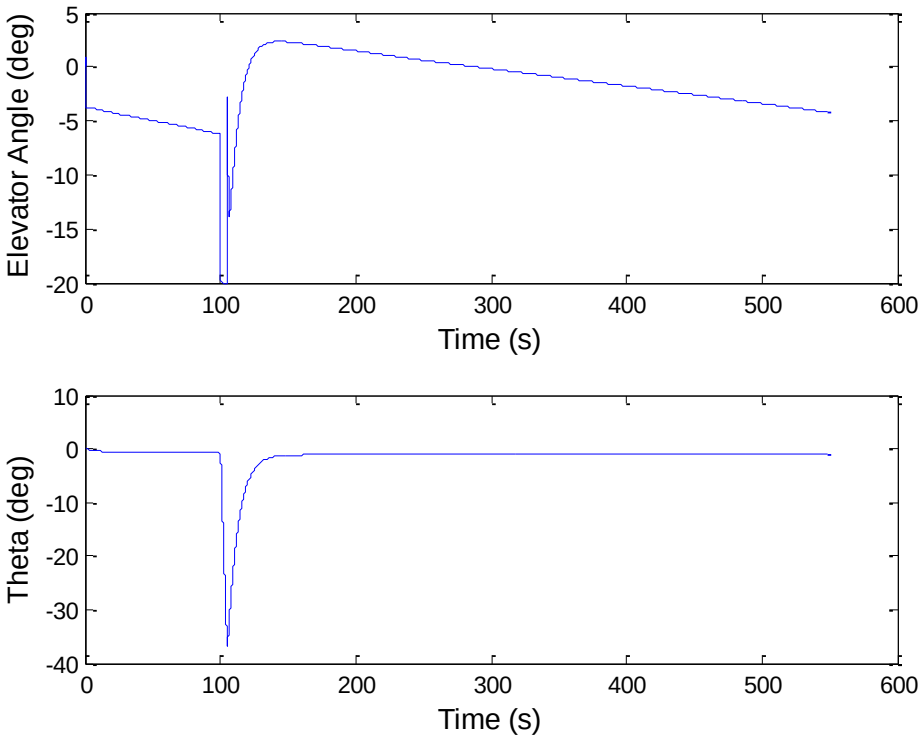

FIGURE 4.3 CONTINUED

4.3 Impact of Fault Recovery Functions

4.3.1 Comparison of Results

The recovery was affected by the fault duration as shown in Figure 4.4. For example, when the fault was not detected for 11 seconds, the aircraft continued to pitch downward with increasing pitch attitude even after the fault recovery action was initiated till it reached a downward vertical dive. The rate of descent increased to greater than 30,000 fpm and the aircraft crashed in approximately 30 seconds from the time the fault was initiated. In contrast, when the fault was detected within 5 seconds, the aircraft was able to regain its intended state even after a sudden drop in altitude due to the fault.

After the fault was detected and the repair action was initiated, the aircraft required a certain duration of time before it started regaining its intended state. Hence, in this case study it was observed that aircraft recovery would not be possible beyond a fault duration of 10 seconds.

The results of this case study thus demonstrate how key assumptions in one component can be tested in the context of the entire system. It also showed how the component functions may have a time attached to them, such as the allowable delay in fault detection.

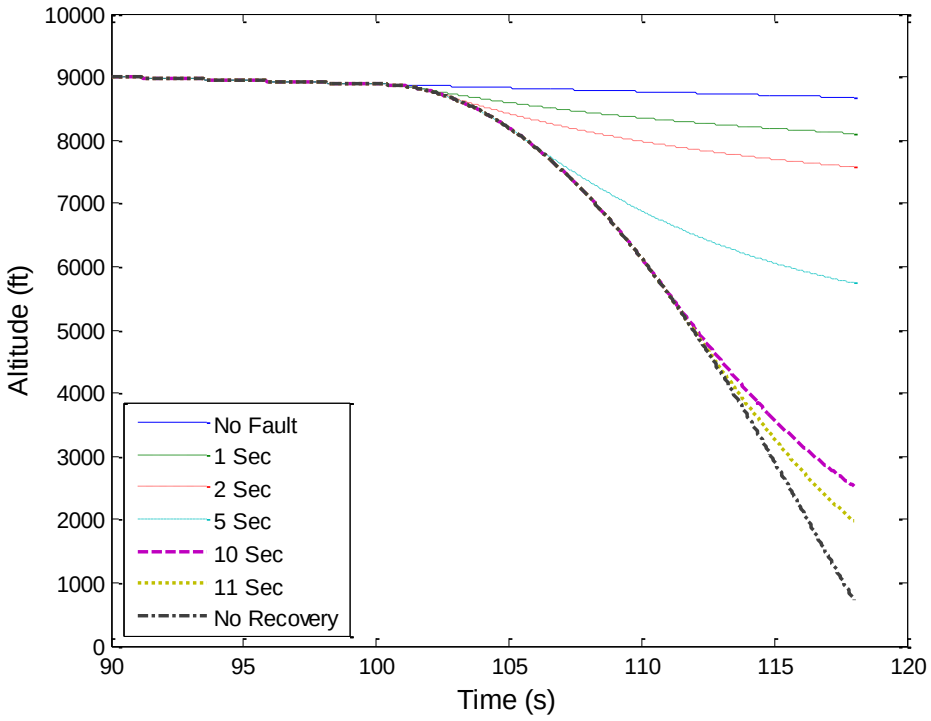

FIGURE 4.4: COMPARISON OF AIRCRAFT BEHAVIOR WITH VARYING FAULT DURATION

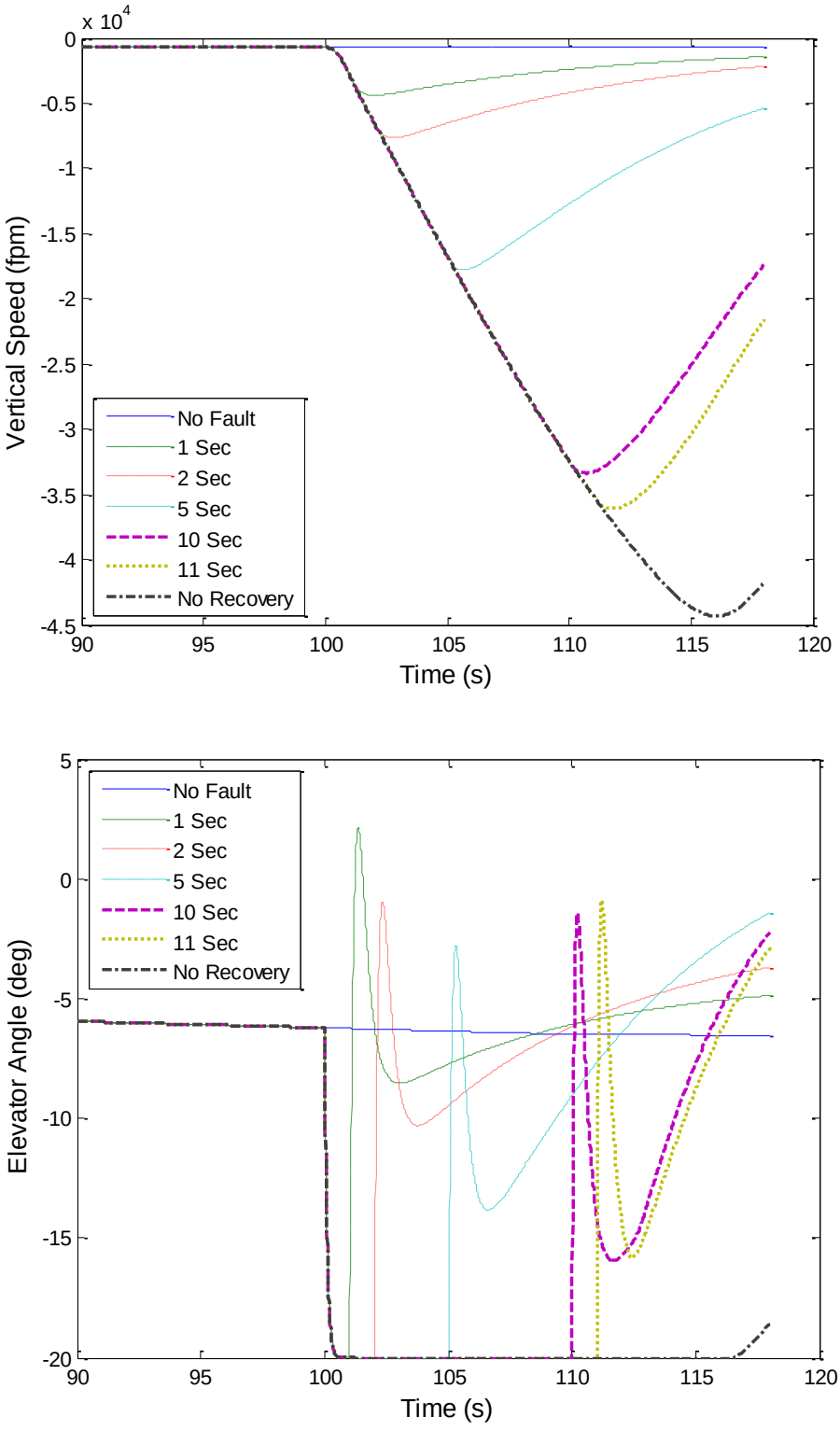

FIGURE 4.4 CONTINUED

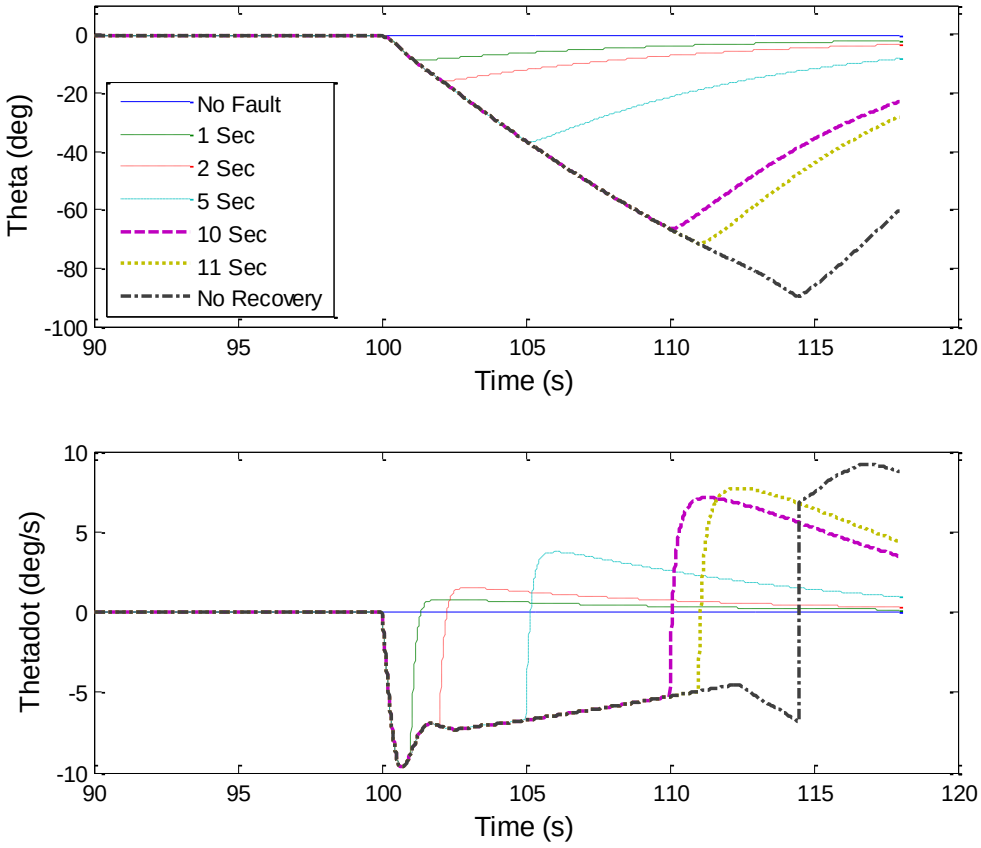

FIGURE 4.4 CONTINUED

4.3.2 Outcome of Study

This study enabled identifying a key requirement to be considered in the development of a fault management function – the fault duration. It was found that the fault duration affected not only the aircraft recovery but also the duration of safe flight after recovery. This would help determine the time available for the fault management to ensure the safety requirements are maintained. The following Figure 4.5 shows the time required for the aircraft to recover to a stable vertical speed descent after the elevator

reversal fault is introduced, based on the time of fault detection. After 10 seconds of the fault, the aircraft is unable to recover, hence this has been depicted as zero.

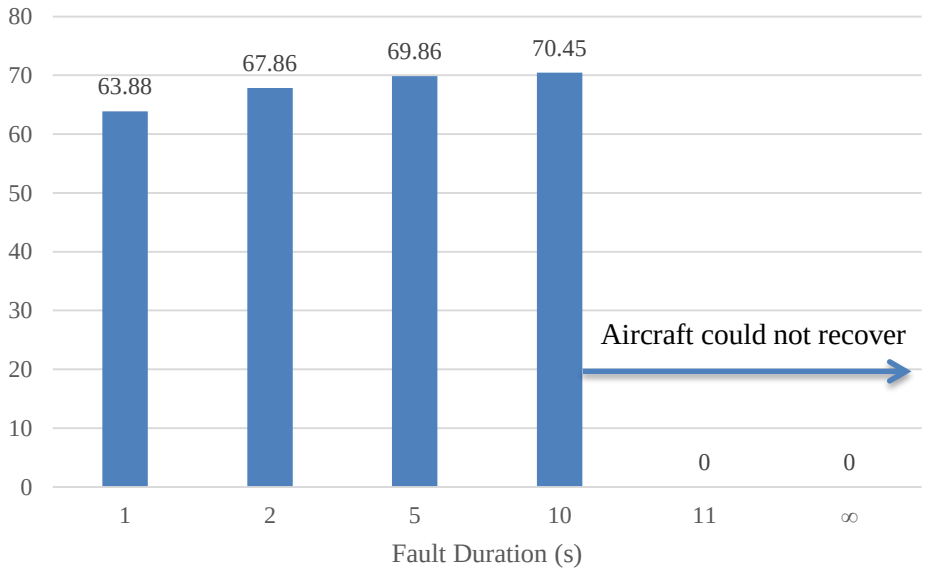

FIGURE 4.5: TIME REQUIRED TO RECOVER AIRCRAFT DESCENT RATE AFTER FAULT INTRODUCTION BASED ON FAULT DURATION

5 CASE STUDY: ERRONEOUS AIRSPEED DATA

This case study examined possible adverse emergent behaviors at the system-level when all components were performing their set functions as desired but received bad input from the environment, in this case with plugged pitot tubes.

This scenario was inspired by past aircraft incidents in which the Air Data Inertial Reference Unit (ADIRU) provided erroneous airspeed data. The ADIRU uses multiple redundant sensors to help compute the aircraft attitude. It receives total pressure and static pressure inputs from three pitot and static pressure sources, and compares them to get a reliable estimate of airspeed, altitude and vertical speed that is both given to the autopilot and portrayed to the pilots. The indicated airspeed is a function of dynamic pressure, that is, the difference between total pressure obtained from the pitot probe and the static pressure.

$$P_t = P_s + \frac{\rho V^2}{2}$$

P_t: Total Pressure (psf) (ram pressure), from pitot probes

P_s: Static Pressure (psf), from static pressure sources

ρ: Air Density (slug/ft^3)

V: Airspeed (knots)

Indicated airspeed is calculated by calibrating the air density to that at sea level at standard conditions i.e., 0.0023769 slug/ft^3.

Hence, indicated airspeed:

$$V_{ind} = \sqrt{\frac{2\,(P_t - P_s)}{\rho_0}}$$

The airspeed indicator becomes unreliable if the pitot probes are blocked. This resulting erroneous indicated airspeed is then fed to other systems including overspeed warnings, windshear warnings, the autopilot, autothrottle and flight directors[20]. For example, the autothrottle system may erroneously sense an overspeed and command a thrust reduction which might result in a stall, or the autopilot may command an abrupt pitch up.

Several accidents and incidents related to plugged pitot tubes were noted in 2009, including the crash of the Air France Flight 447 into the Atlantic Ocean on June 1, 2009 in which the pilots reacted adversely to wrong airspeed data, which caused a disconnection of the autopilot and autothrottle[21], an incorrect airspeed and altitude information incident on a TAM Airlines Airbus A330-200 flight 8091 in cruise flight on May 21, 2009[22], and another airspeed anomaly incident in cruise flight on an Airbus A330-323 Northwest Airlines flight on June 23, 2009[23]. In reference to these incidents, this case study observed the effect of completely plugged pitot probes leading to erroneous airspeed input to the aircraft control system, and the impact of fault management with varying fault detection times.

5.1 System Components

The components which have been considered to simulate this scenario are the Air Data Inertial Reference Unit (ADIRU), autopilot, autothrottle, and fault management.

5.1.1 Component Functions

The pitot probes provide pitot pressure or ram pressure to the ADIRU, which then computes the airspeed based on the inputs from the pitot and static pressure sources and

provides a single airspeed as output to the autopilot and autothrottle systems. The autopilot sets the pitch attitude to attain a specified vertical speed for the descent or climb, given the pitch mode selected. The autothrottle provides the required power settings to maintain the airspeed set in the Mode Control Panel (MCP) in speed (SPD) mode, or a commanded thrust in thrust (THR) mode. The fault management detects any discrepancy in the airspeed data and reports it to the autopilot and autothrottle systems. These systems will respond by disregarding the airspeed data from the ADR system, and adjusting to a safe power and pitch setting to return to the intended flight path.

5.1.2 Axiomatic Conditions

The design of the ADIRU assumes that the several redundancies in the form of multiple pitot tubes ensures that the airspeed calculation within the ADIRU is immune to any isolated failures of the sensors and pressure probes, but this axiom can be violated when all the pitot tubes ice over. Any error in the pressure data obtained from the pitot probes is assumed to trigger the fault management to alert the autopilot/autothrottle.

5.1.3 Component Interactions

The total pressure data from the pitot tubes is used by the ADIRU to calculate the indicated airspeed, which is relayed to the autopilot and autothrottle. These autoflight systems use this airspeed data and vary the pitch and power settings to match it with the commanded airspeed as entered in the MCP. In case of a discrepancy between commanded and true values of airspeed, the fault management notifies the autopilot and autothrottle systems, which then switch the autoflight modes to theta and THR modes respectively.

5.2 Simulation Execution

5.2.1 Scenario

The plugged pitot tubes cause an error in the indicated airspeed calculated by the ADIRU system such that, the calculated indicated airspeed does not reflect changes in true airspeed; but instead only varies with the different static atmospheric pressures at different altitudes. Since an airspeed has been commanded to be maintained in all the studies, the aircraft tries to maintain a constant indicated airspeed. Since the indicated airspeed is erroneous, the constant indicated airspeed leads to an error in the true airspeed. The fault management detects the problem in each simulation run and notifies the autopilot and autothrottle systems. The autopilot then sets the pitch of the aircraft to a stable attitude, while the autothrottle maintains the power to descend or climb as necessary, disregarding the indicated airspeed. This requires disabling the autopilot and autothrottle modes. Four different flight profiles have been studied:

1. V/S mode descent
2. V/S mode climb
3. FLCH mode descent
4. FLCH mode climb

The Figure 5.1, Figure 5.2, Figure 5.3 and Figure 5.4 show the normal flight profiles for the four cases when no fault is introduced.

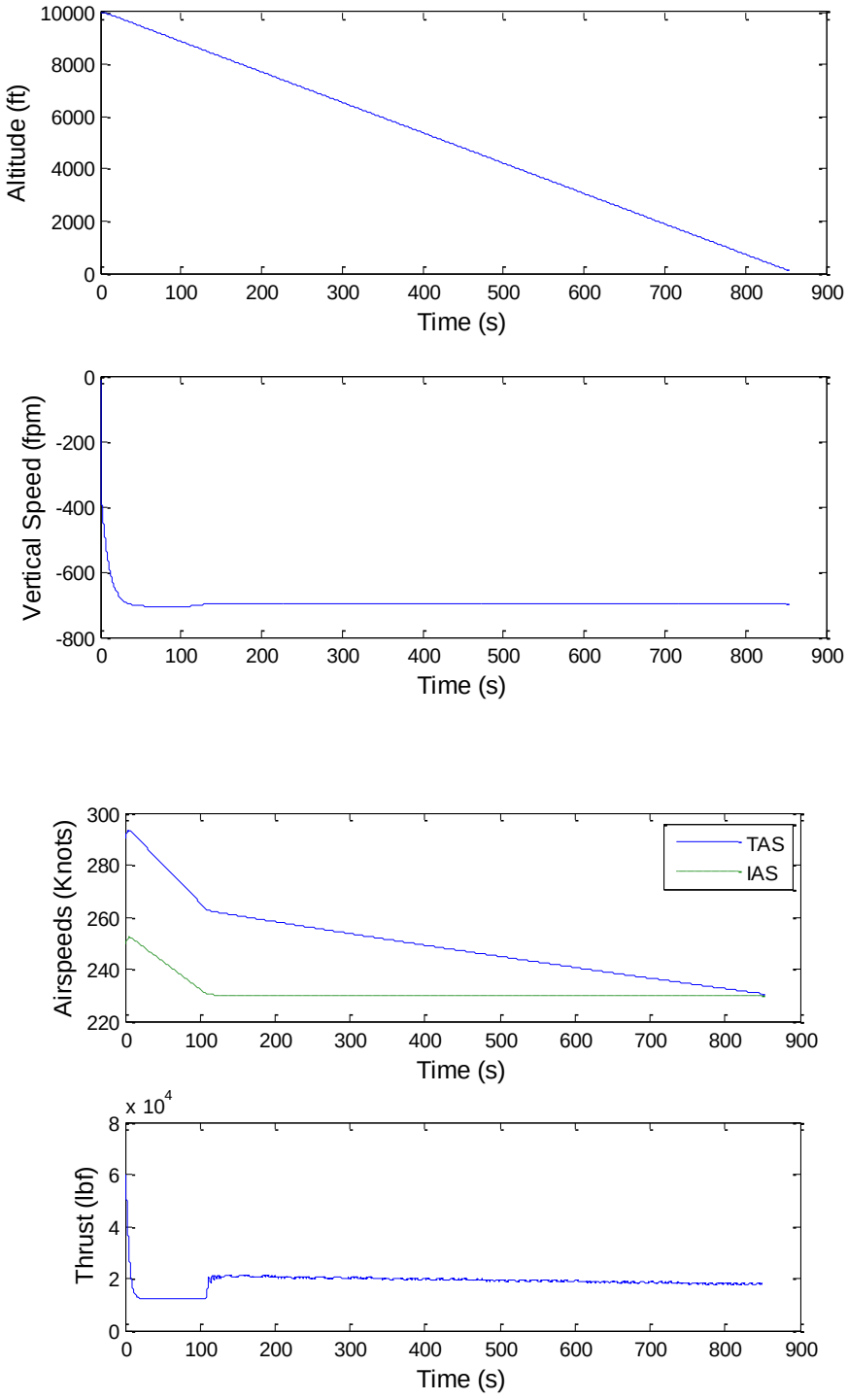

FIGURE 5.1: NORMAL FLIGHT PROFILE FOR V/S DESCENT MODE

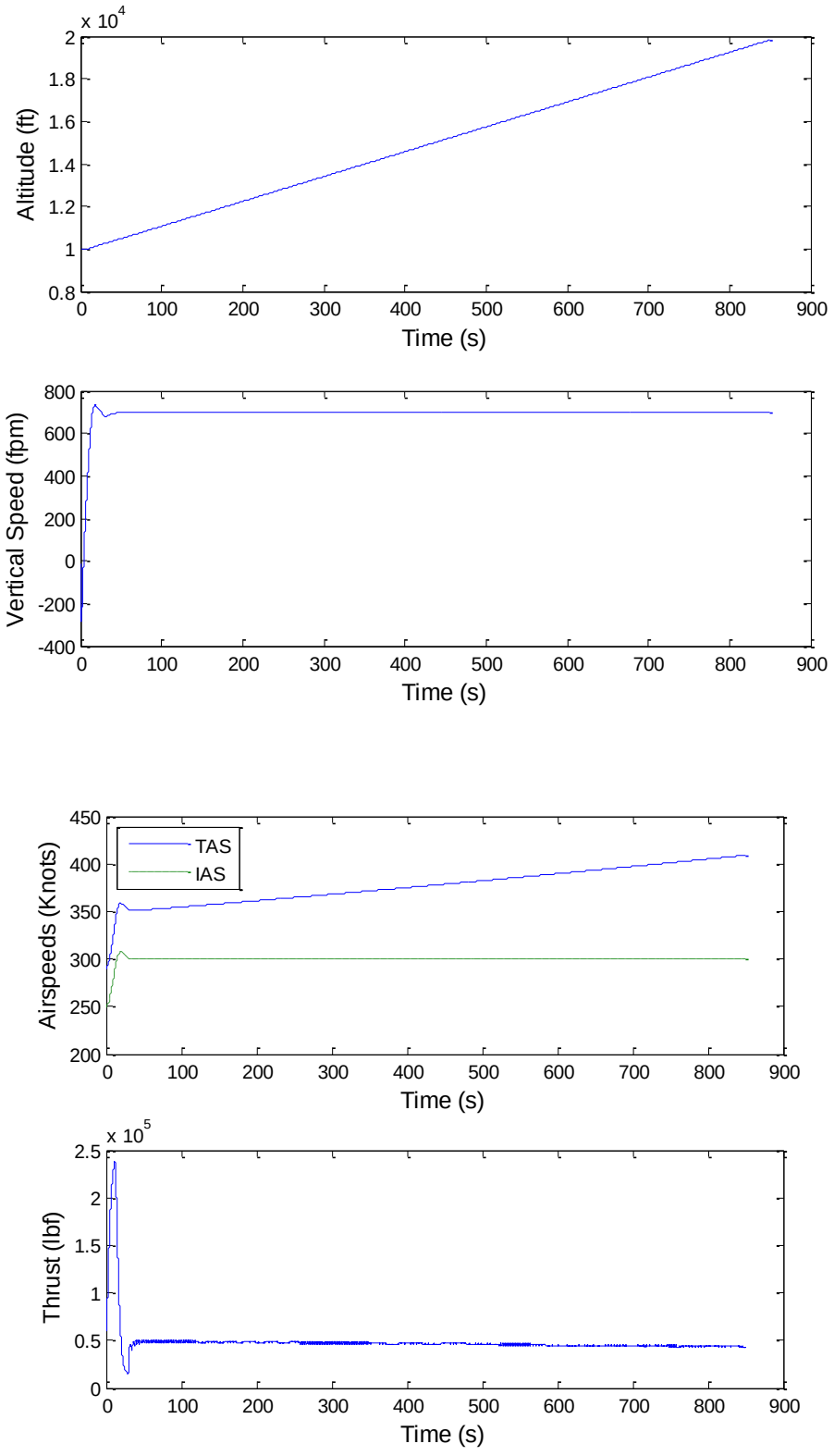

FIGURE 5.2: NORMAL FLIGHT PROFILE FOR V/S CLIMB MODE

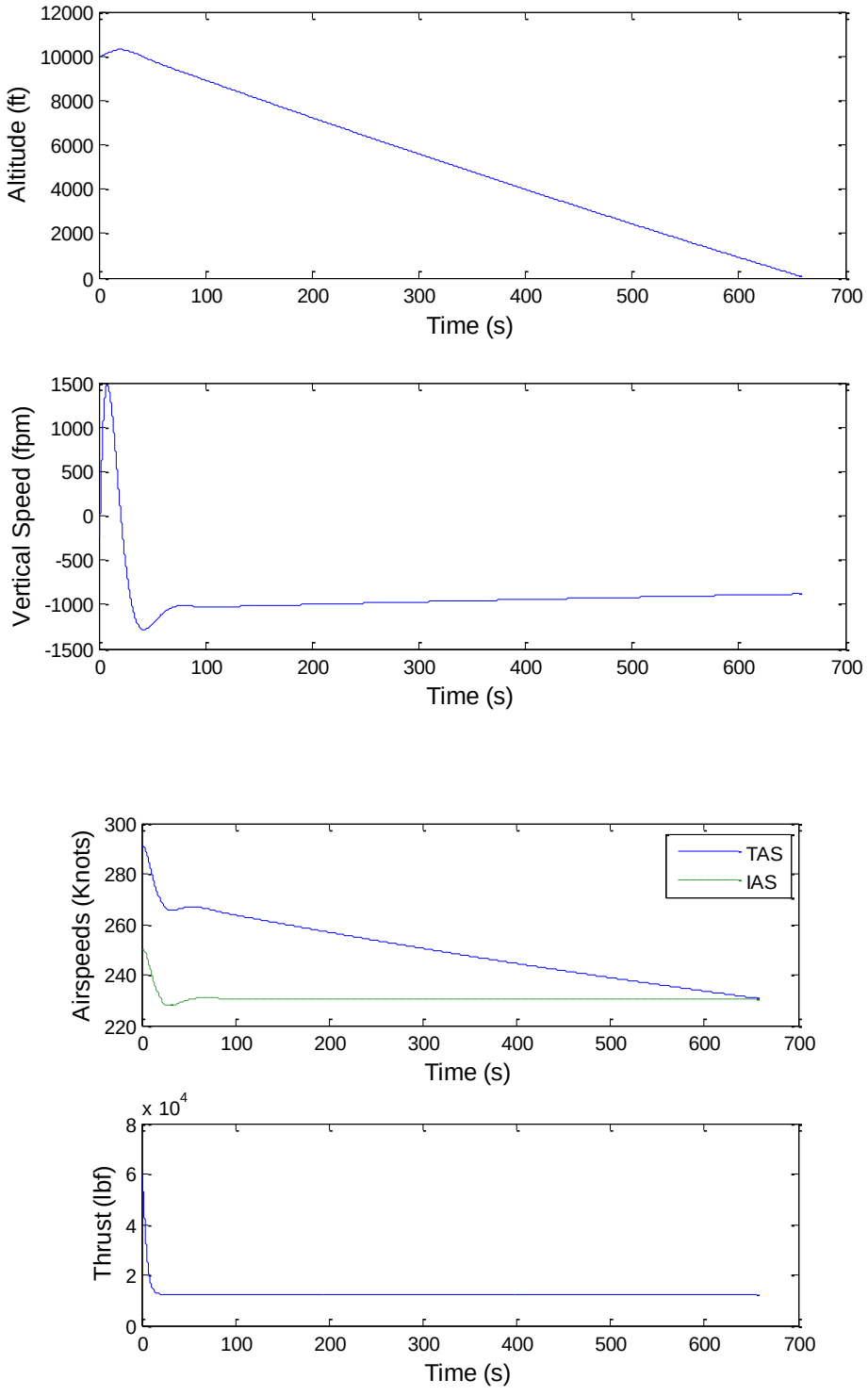

FIGURE 5.3: NORMAL FLIGHT PROFILE FOR FLCH DESCENT MODE

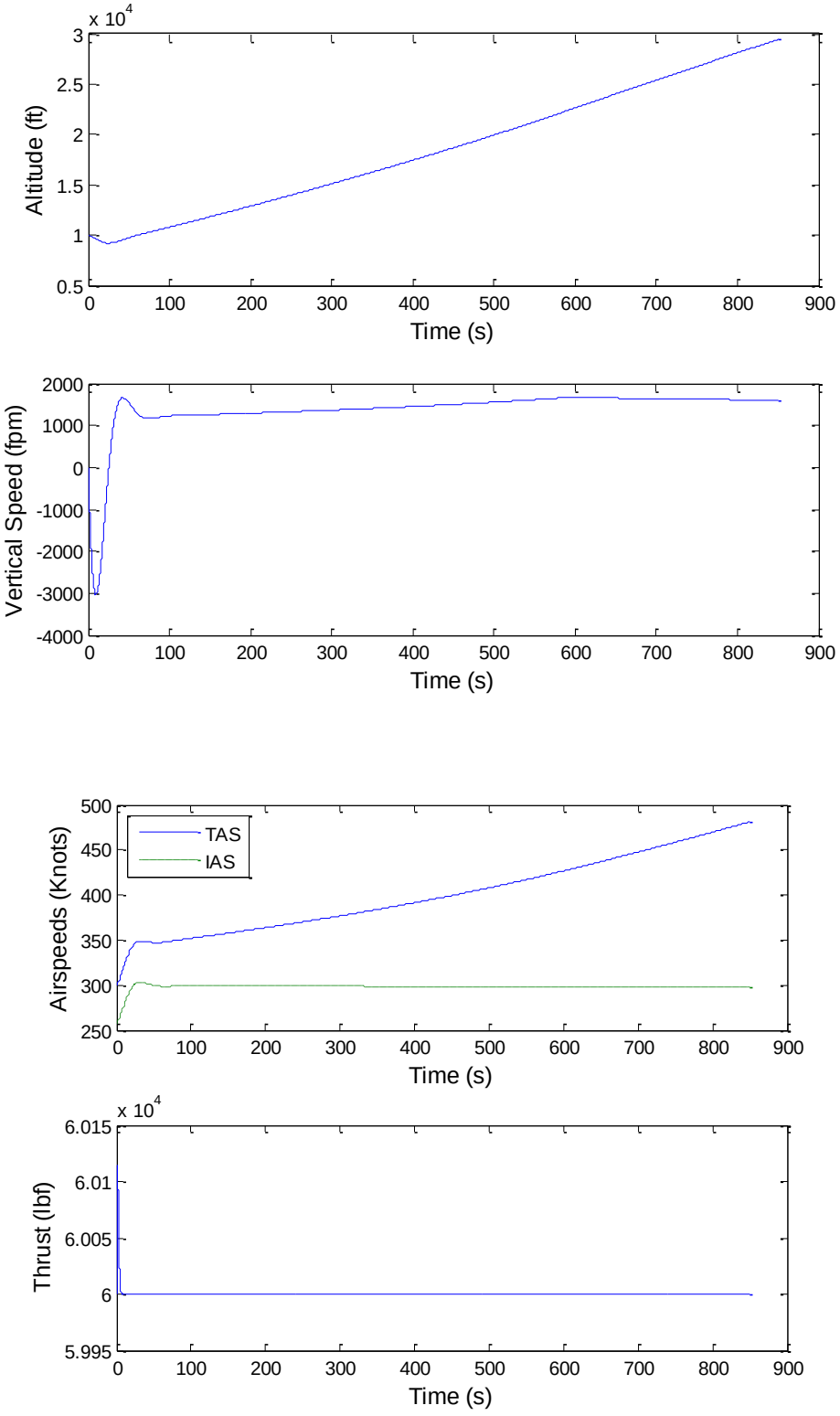

FIGURE 5.4: NORMAL FLIGHT PROFILE FOR FLCH CLIMB MODE

5.2.2 Fault Introduction and Recovery

All the pitot tubes get completely plugged at a scheduled time in the simulation due to some environmental factors, causing their inability to obtain the ram pressure input. The air pressure in the tubes before they get plugged becomes the constant pressure input to the ADIRU. Since the aircraft is in a climb or descent, the static pressure keeps changing as it increases with decrease in altitude. This causes an error in the calculated indicated airspeed which is a function of the difference in static and total pressure.

To recover from the adverse behavior of the aircraft due to this scenario, the reliance on airspeed data is disabled by disengaging the airspeed control loop or flight level change control loop, whichever is engaged, and setting a pitch angle and thrust to be maintained to perform the descent or climb as required.

5.2.3 System Behavior

When the fault is introduced, that is, the pitot tubes are plugged, it is a violation of the axiom that the several redundancies would prevent incorrect airspeed data being obtained from the ADIRU. When the autopilot tries to maintain the erroneous indicated airspeed according to the commanded value, the true airspeed of the aircraft is affected as the aircraft climbs or descends. This causes the flight path to change unexpectedly and may also cause the aircraft to slow or speed up to unsafe airspeeds.

5.3 Impact of Fault Introduction and Recovery

5.3.1 Comparison of Results

Impact of the fault involving a pitot tube block was studied for the four selected flight paths. The recovery action was studied in the vertical speed descent case for two possible parameters for fault detection which affected the time of fault detection.

5.3.1.1 V/S Descent

When the aircraft is in a V/S descent and the fault is introduced about 100 seconds from the start of the simulation, the true airspeed is found to increase while the autothrottle attempts to track the erroneous indicated airspeed. This effect builds as the difference in the total and static pressures increases with decrease in altitude. This thus causes the aircraft to start climbing.

5.3.1.1.1 No Fault Recovery

It is noticed that the effect on the altitude is seen more than 500 seconds after the fault is introduced. This occurs due to a sudden increase in vertical speed as the thrust goes to a maximum value to try to maintain the commanded indicated airspeed. The true airspeed reaches a high value which eventually stabilizes as no more power can be added. This could result in an overspeed warning. Figure 5.5 shows the results of this simulation.

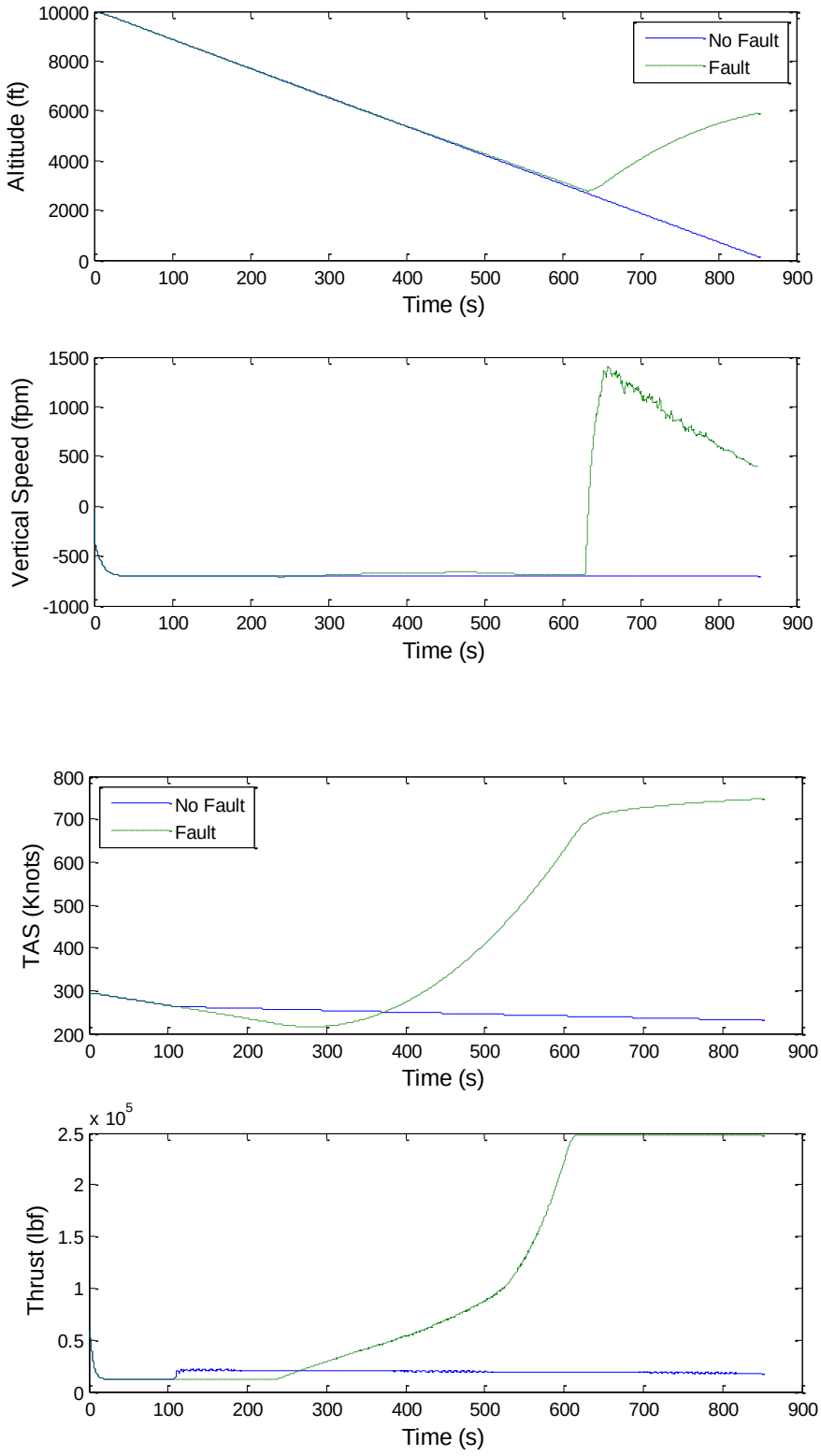

FIGURE 5.5: V/S DESCENT - FAULT NOT REPAIRED

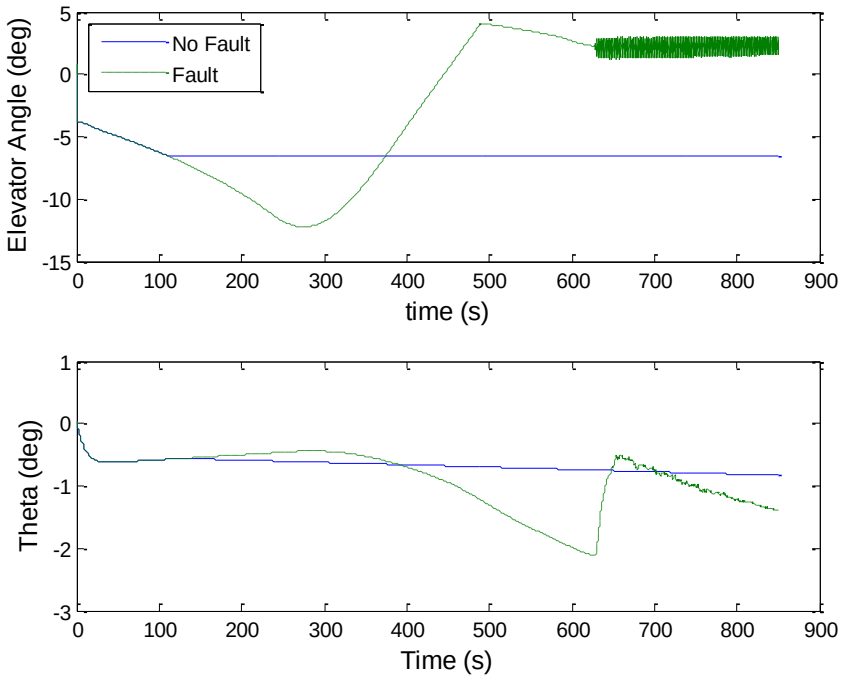

FIGURE 5.5 CONTINUED

5.3.1.1.2 Fault Detected 150 sec after Pitot Tubes Plugged

If the fault is detected after the true airspeed is found to become lower than the indicated airspeed and then increase, which confirms an error, the aircraft behavior is as seen in Figure 5.6. The fault is detected 150 seconds after the pitot tubes are plugged. To recover, the airspeed loop is disabled and fixed values of pitch and thrust are commanded to recover the intended flight path. It is found that a gradual recovery is possible in this case. Figure 5.6 shows the results of this simulation.

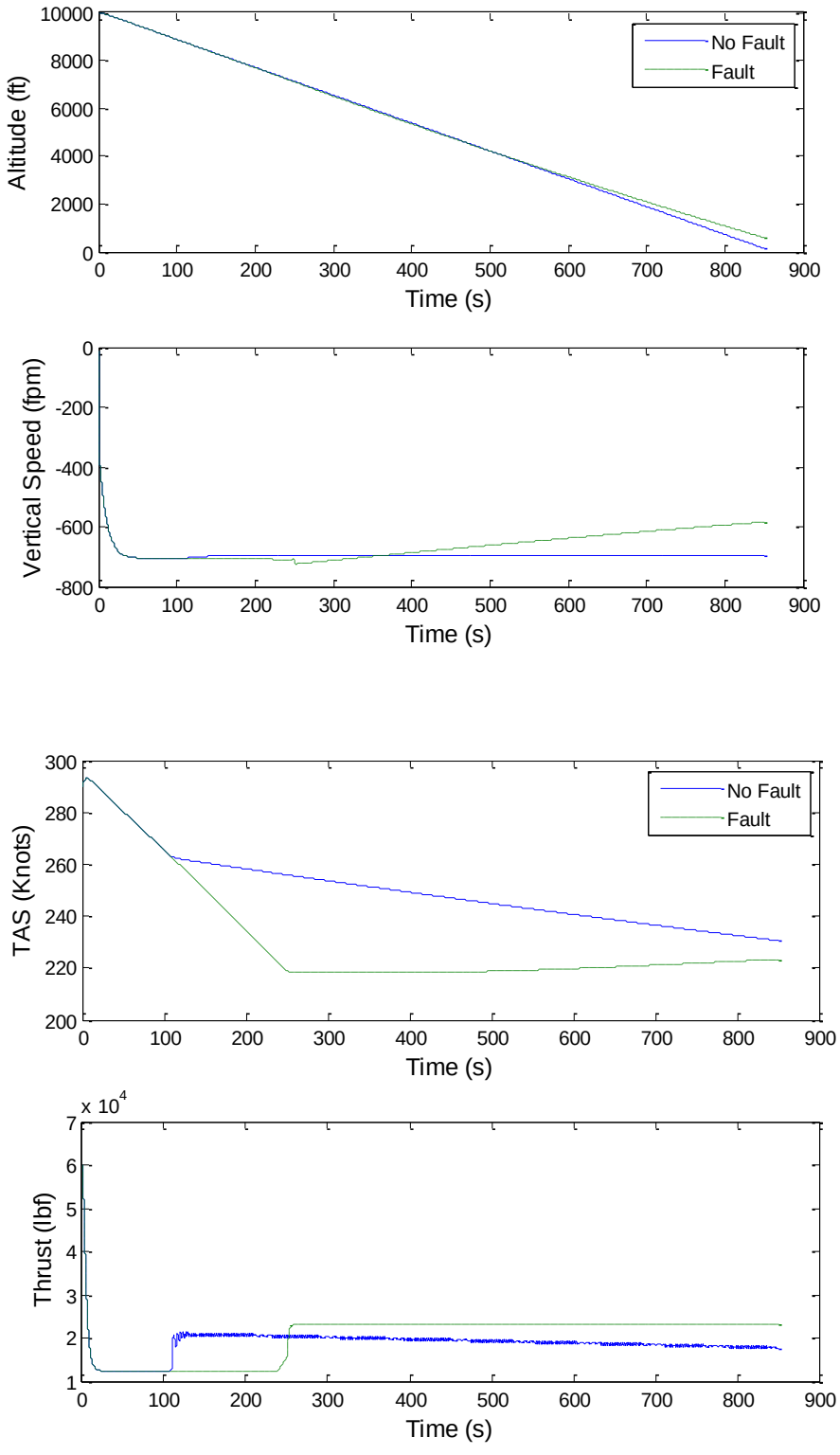

FIGURE 5.6: V/S DESCENT – FAULT DURATION 150 SEC

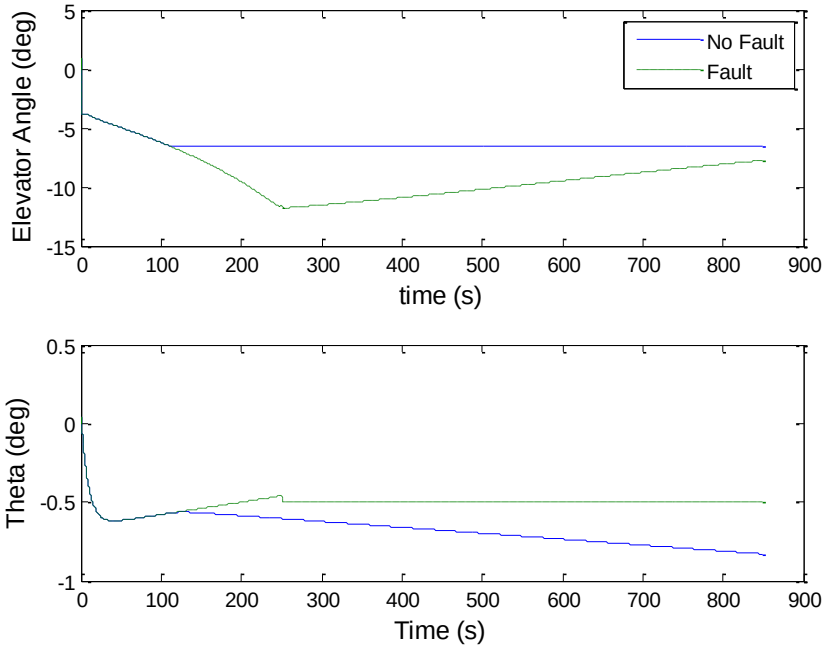

FIGURE 5.6 CONTINUED

5.3.1.1.3 Fault Detected 600 sec after Pitot Tubes Plugged

However, if it is not possible to monitor the true airspeed and the fault is detected only after the altitude starts increasing when the aircraft is supposed to descend, the recovery, though possible, is found to be more difficult. It was found from this case study that it is not just the time after which the fault is detected that would affect the aircraft recovery; it is also the commanded values of thrust and pitch required to achieve a stable recovery. It is also noted that the time at which the fault can be detected is dependent on various factors, like which parameters need to be monitored to enable earliest detection of the fault. Figure 5.7 shows the results of this simulation.

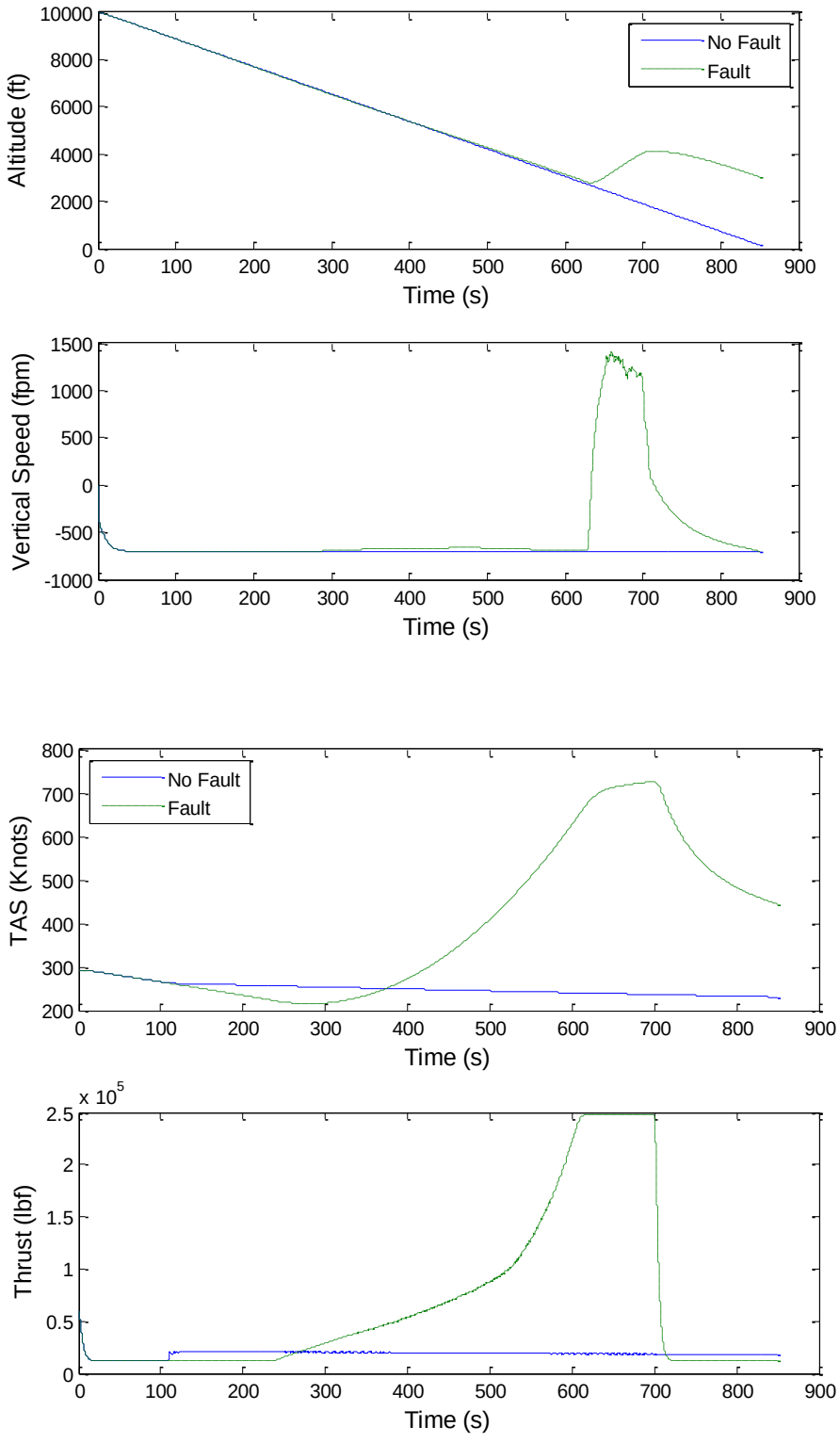

FIGURE 5.7: V/S DESCENT – FAULT DURATION 600 SEC

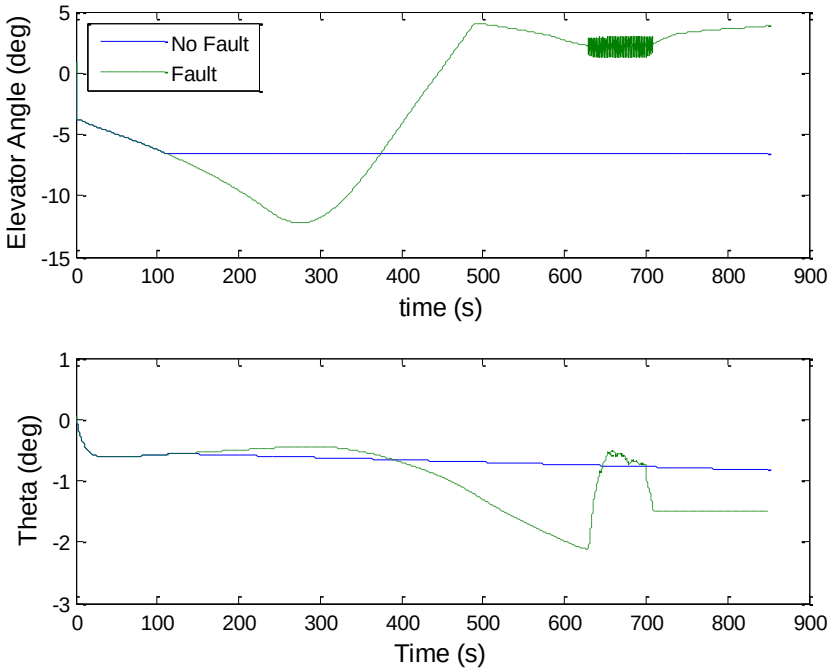

FIGURE 5.7 CONTINUED

5.3.1.2 V/S Climb

When the fault was introduced during a V/S climb, 100 seconds from the start of the simulation, the aircraft was found to stall about 250 seconds after the fault was introduced. Figure 5.8 shows the results of this simulation.

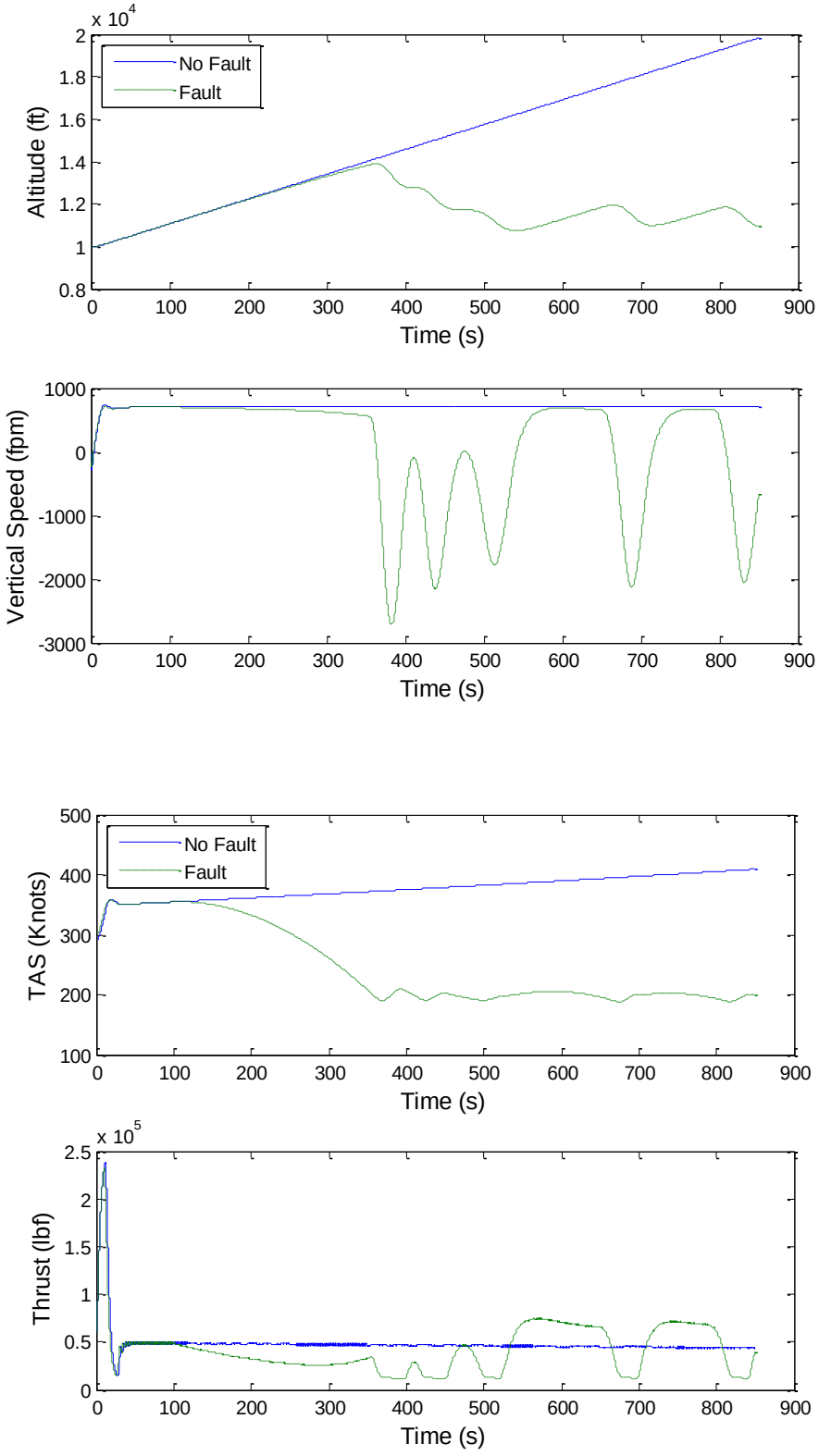

FIGURE 5.8: PITOT TUBES PLUGGED IN V/S CLIMB

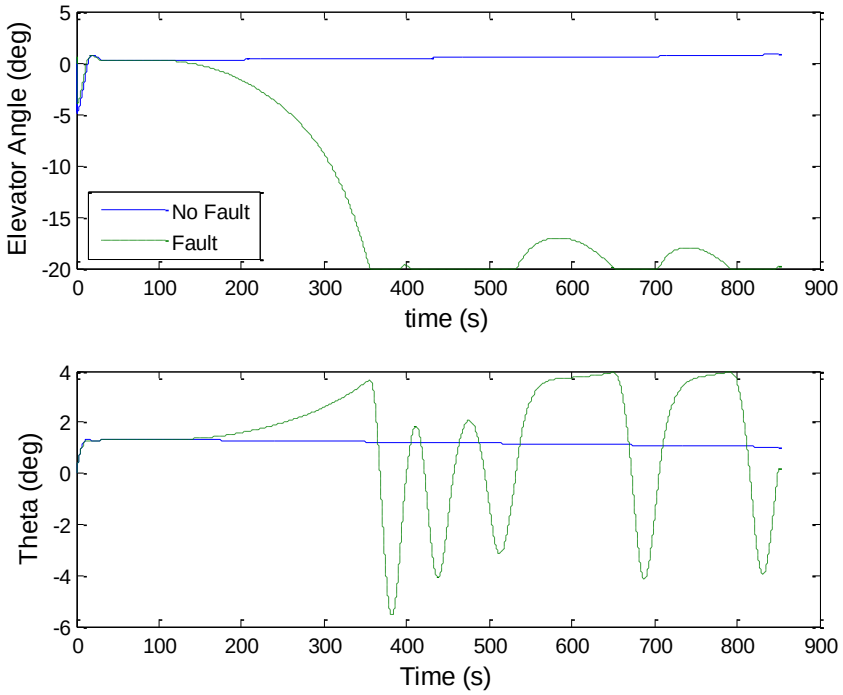

FIGURE 5.8 CONTINUED

5.3.1.3 FLCH Descent

During a flight level change descent in which the autopilot commanded elevator to pitch to maintain indicated airspeed, the aircraft was found to descend at a much faster rate than it normally would, with the true airspeed increasing throughout. This could cause the airplane to crash catastrophically if the fault is not repaired in time. The altitude was found to decrease rapidly because the power is set to idle for a flight level change descent. Figure 5.9 shows the results of this simulation.

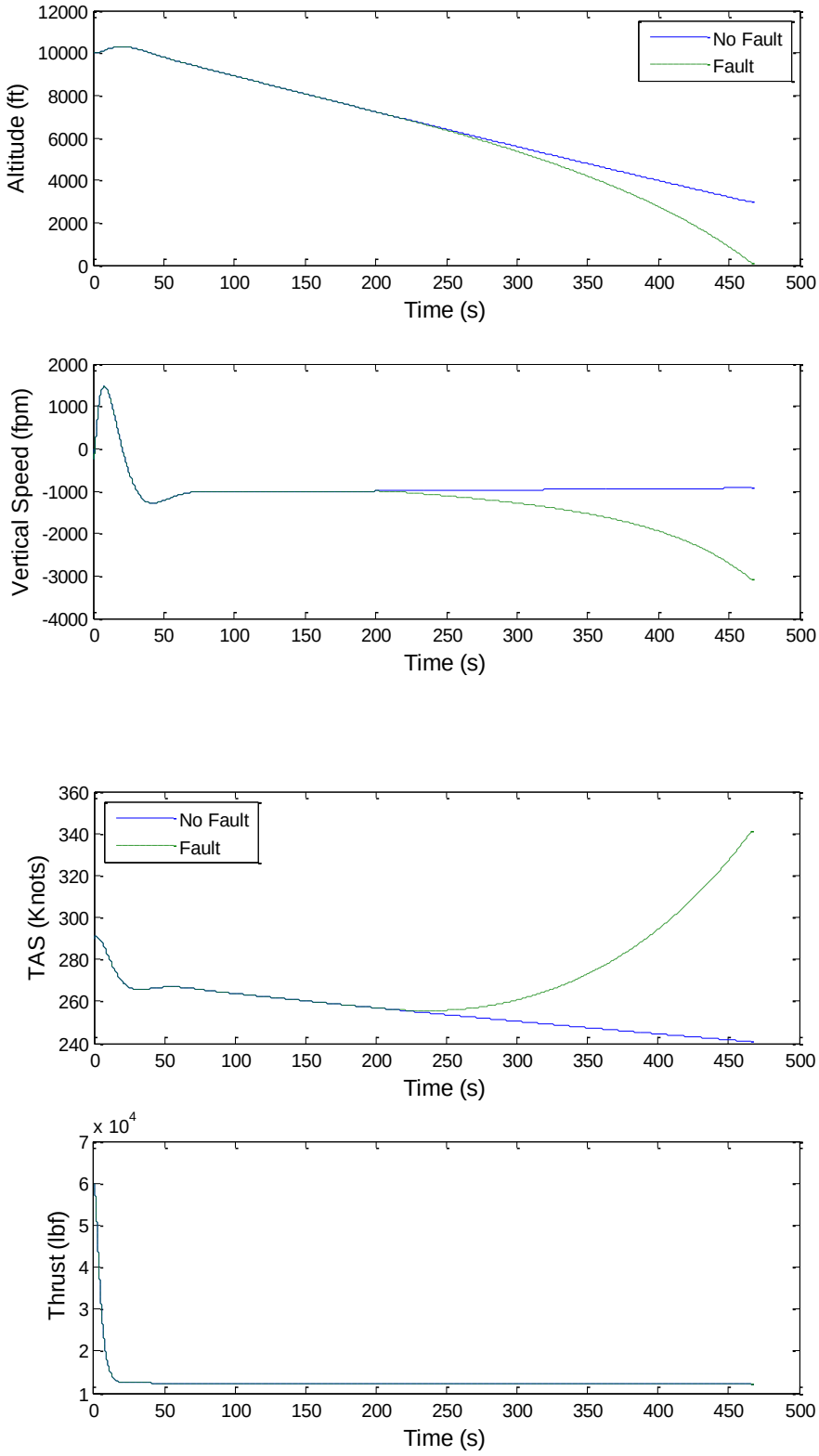

FIGURE 5.9: PITOT TUBE PLUGGED IN FLCH DESCENT

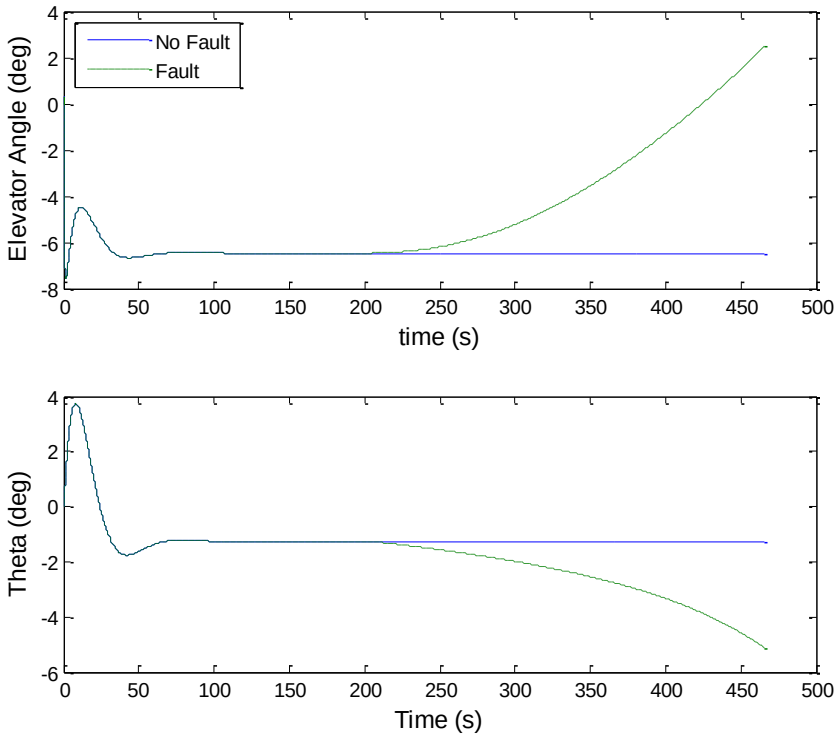

FIGURE 5.9 CONTINUED

5.3.1.4 FLCH Climb

In the case of a flight level change climb, the altitude stops increasing after a certain time after the fault is introduced as the thrust is at a maximum value for FLCH climb mode while the true airspeed is decreasing due to the fault. Eventually, it could cause the aircraft to stall and start falling rapidly. Figure 5.10 shows the results of this simulation.

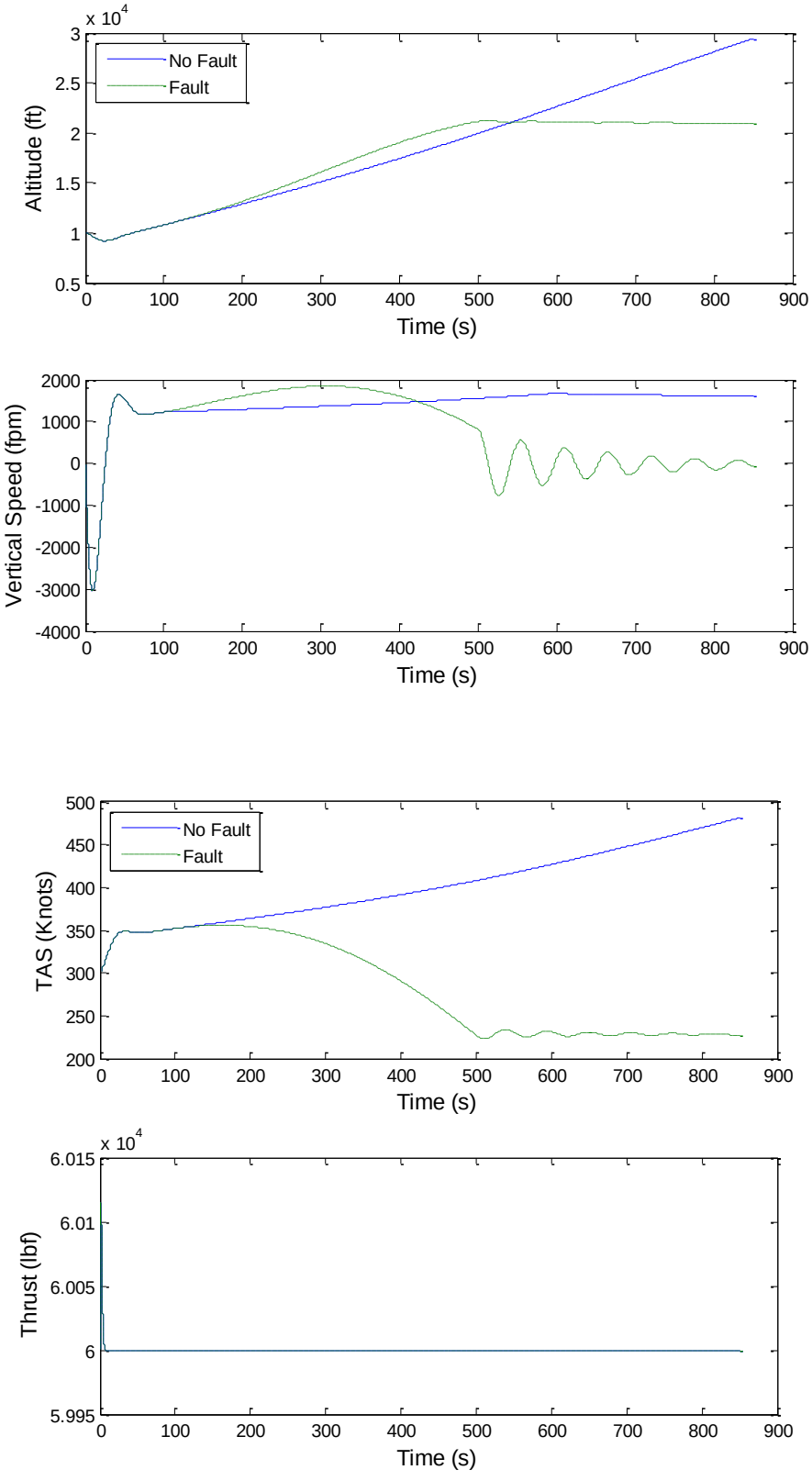

FIGURE 5.10: PITOT TUBES PLUGGED IN FLCH CLIMB

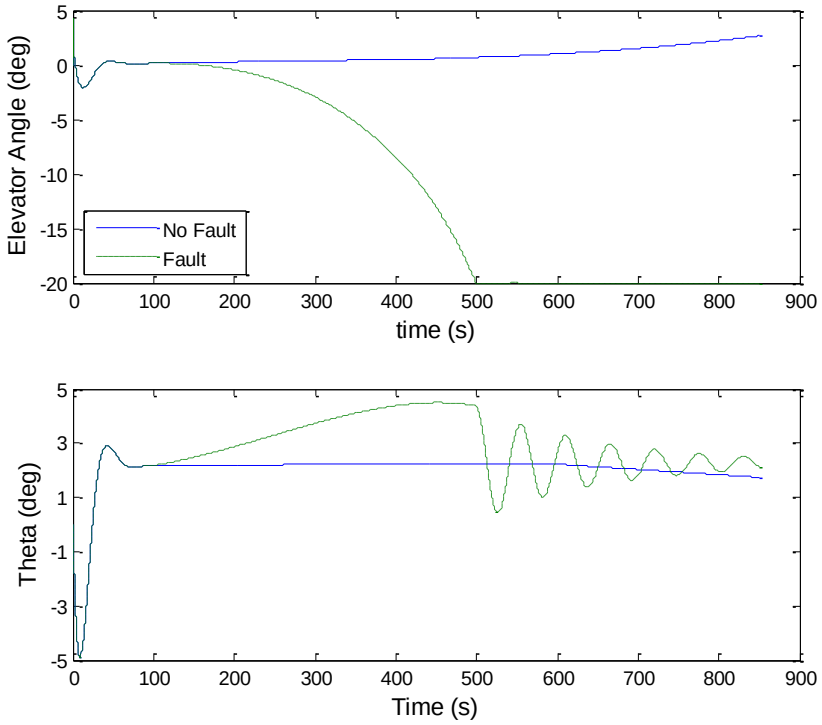

FIGURE 5.10 CONTINUED

5.3.2 Outcome of Study

This case study helped understand several requirements for the fault management which can affect the aircraft recovery:

Firstly, the phase of flight or flight path that the aircraft is following. It was found that the effect of the fault depended on the selected autoflight modes and whether the aircraft was in a climb or descent. Each of the studied flight paths had distinctly different results and hence, the aircraft recovery functions will vary based on the flight path.

Secondly, the parameter selected for fault detection. In this case it was observed that depending on whether the true airspeed or an abrupt change in altitude was

considered as the parameter to detect the fault, the time at which the fault was detected varied.

Finally, the fault duration. Depending the time after which the fault was detected, the recovery of the aircraft was affected. Hence, the longer the fault duration, the more difficult the recovery. This aspect can help determine the time available to ensure safe recovery of the aircraft and thus, the parameter to be selected for easy detection of the fault to ensure a timely recovery.

6 CASE STUDY: HUMAN-AUTOMATION INTERFACE FAILURE

The third case study observed changes in system behavior when a component failed to perform its required function. In this case, the pilot was considered as the component that failed to perform the required actions for landing an aircraft in normal flight conditions, causing the emergence of adverse system behavior.

This case was inspired by an aircraft accident in June 2013 where an Asiana Airlines flight, a Boeing 777-200, stalled and collided with the sea wall while landing at San Francisco International Airport. The pilots appear to have failed to comprehend how their selected autopilot modes, commanded altitude and airspeed, and an autothrottle setting commanding idle power caused the aircraft to slow to unsafe airspeeds.[24]

The Mode Control Panel (MCP) of a Boeing 777 aircraft has autopilot (A/P) engage and disengage switches, as well as the various autopilot flight director system (AFDS) modes like ALT, HDG HOLD, V/S, TO/GA, LNAV, VNAV and FLCH. Through these modes, the AFDS provides autopilot pitch, roll and yaw commands to the Primary Flight Control System (PFCS) when the autopilot is engaged, and the pitch and roll commands on the Primary Flight Display (PFD) when the flight directors are engaged. The autothrottle system controls include left and right arm switches to control the left and right engine autothrottle respectively. The autothrottle is coupled with some AFDS pitch modes (VNAV, FLCH or TO/GA). For example, if the FLCH, or flight level change mode, is operational during descent, the autothrottle operates in thrust mode, generally commanding idle thrust for descent and full power for climb. With other pitch

modes (e.g. ALT, V/S), the autothrottle can be put into SPD mode which varies throttle to control speed.

In the Asiana Airlines accident, the autopilot was in the vertical speed (V/S) pitch mode with a descent rate of 1000 ft/min, and the autothrottle was in SPD mode to maintain a set airspeed. The altitude set in the mode control panel (MCP) was the missed approach altitude, i.e. 3,000 feet. At about 1,200 feet altitude, a pilot selected FLCH mode to descend quickly as they were high relative to the approach path. The autothrottle mode changed to thrust mode and the aircraft started climbing towards the MCP altitude of 3000 feet. Seeing this, a pilot disengaged the autopilot and manually brought the throttle lever to idle to descend. This caused the autothrottle to go to HOLD mode, with the autothrottle holding idle power. When the pilots entered an airspeed in the MCP, it was not automatically maintained by the autothrottle as it was not in SPD mode, and the idle power caused the airspeed to reduce until the aircraft stalled. This accident appears to have occurred due to violation of the assumption that the pilots know all the autopilot and autothrottle modes, and the behavior of the autopilot flight director system and the primary flight control system in response to the modes they select. Thus, it was seen that commanding inappropriate autoflight modes, and their interactions with the control actions executed by the pilots, may cause the emergence of an adverse flight condition.

6.1 System Components

The components used to simulate this scenario are the pilot, autopilot and autothrottle.

6.1.1 Component Functions

The pilot's functions for this case study include enabling or disabling autopilot and autothrottle modes, commanding values like airspeed, vertical speed, altitude, thrust, etc. depending on the modes selected, and monitoring the active flight modes, which are available on the primary flight display, and the aircraft state, by observing the flight instruments.

The autopilot functions are to command the aircraft to maintain the values commanded by the pilot and to update the aircraft state based on the active autopilot mode.

The autothrottle functions set the power based on whether it is in SPD mode or THR mode. In SPD mode, it adjusts power to maintain a commanded indicated airspeed, whereas in THR mode it maintains the commanded power setting. Manual operation of the autothrottle causes it to switch to HOLD mode, where it no longer controls the airspeed and maintains the current power setting.

6.1.2 Axiomatic Conditions

It is assumed that the pilot is aware of the outcomes of selecting any autopilot and autothrottle modes, and can predict the aircraft state that will result in response to any input commanded by him/her. Further the pilot is assumed, during final approach, to continuously monitor the aircraft state and to take necessary actions to recover from any abnormal behavior of the aircraft.

6.1.3 Component Interactions

The pilot provides inputs to the autopilot by selecting specific modes from the mode control panel. Depending on the flight modes selected, the autopilot will command the primary flight control system to take appropriate control action on the aircraft, while the autothrottle will provide the thrust control or speed control based on the autothrottle mode selected.

6.2 Simulation Execution

6.2.1 Scenario

The aircraft is in V/S mode in a descent for landing at 4,000 feet with a vertical speed of 1,000 fpm and maintaining an indicated airspeed of 180 knots. An indicated airspeed of 160 knots is commanded at the start of the simulation. About 150 seconds from the start of the simulation, when the altitude is just under 2,000 feet, the pilot changes the control loop to FLCH attempting to descend at a faster rate; however, the autothrottle increases the power to maximum takeoff thrust and the aircraft starts climbing. This is due to a wrongly entered higher altitude in the MCP. After 90 seconds the pilots notice the rise in altitude and disable the autopilot and manually command an idle thrust and lower the pitch. The autothrottle, still being engaged, goes into HOLD mode, holding idle power. After descending for a minute with a high descent rate, the pilots command an indicated airspeed of 150 knots by entering it in the MCP and command the aircraft to maintain a fixed flight path angle of 3 degrees to maintain the glide slope. The flight profile for this scenario when no fault is introduced is shown in Figure 6.1.

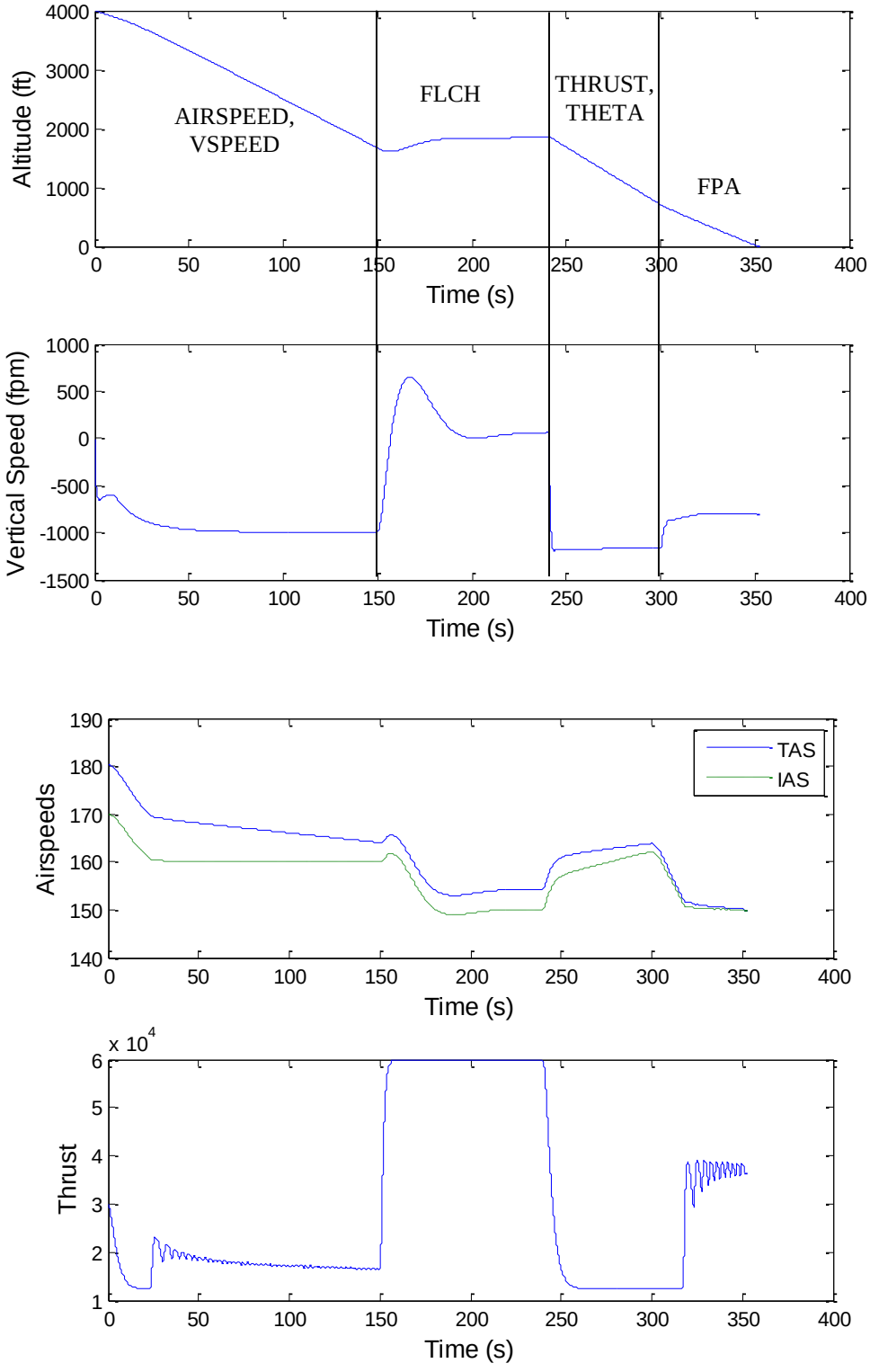

FIGURE 6.1: NORMAL FLIGHT PROFILE FOR PILOT DESCENDING WHILE CHANGING MULTIPLE FLIGHT MODES

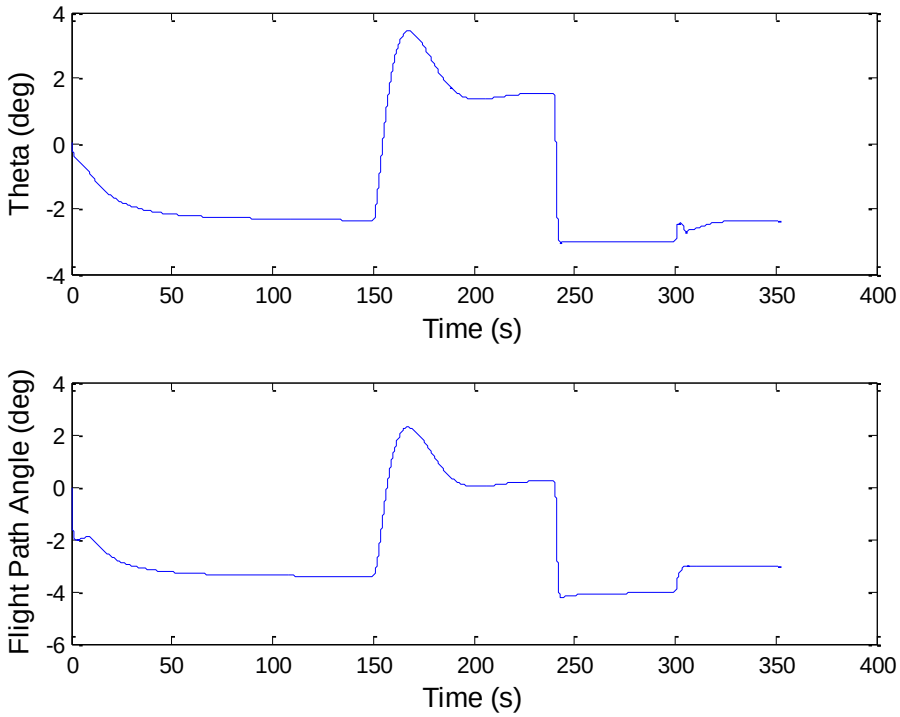

FIGURE 6.1 CONTINUED

6.2.2 Fault Introduction and Recovery

When the autothrottle goes into HOLD mode at idle power the airspeed is not being controlled by the autothrottle. Thus, the pilot commanding an airspeed of 150 knots on the mode control panel does not have any effect on the airspeed. However, in this case the pilot assumes that the autothrottle is engaged in SPD mode and also fails to perform his required monitoring functions completely. This is a violation of the axiom that the pilot knows all autopilot modes or autothrottle settings and monitors the aircraft state.

At a later time in the simulation, the pilot may realize that the airspeed is too low or the aircraft is unstable and descending at too high a rate and unable to maintain the glide slope and would try to recover by applying full throttle and attempting a go-around. This recovery action must be done early enough to be able to perform a go-around.

6.2.3 System Behavior

With the autothrottle in HOLD mode, it did not control airspeed. Since the power was set to idle, this caused the airspeed to reduce. The airspeed continued to reduce below the commanded value until the pilot realized this fault and set the autothrottle to maximum thrust to recover. The fault duration is varied to observe the aircraft's possible recovery. Figure 6.2 shows the effect of the undetected fault while Figure 6.3 shows the effect of a timely initiated recovery action after the fault was introduced.

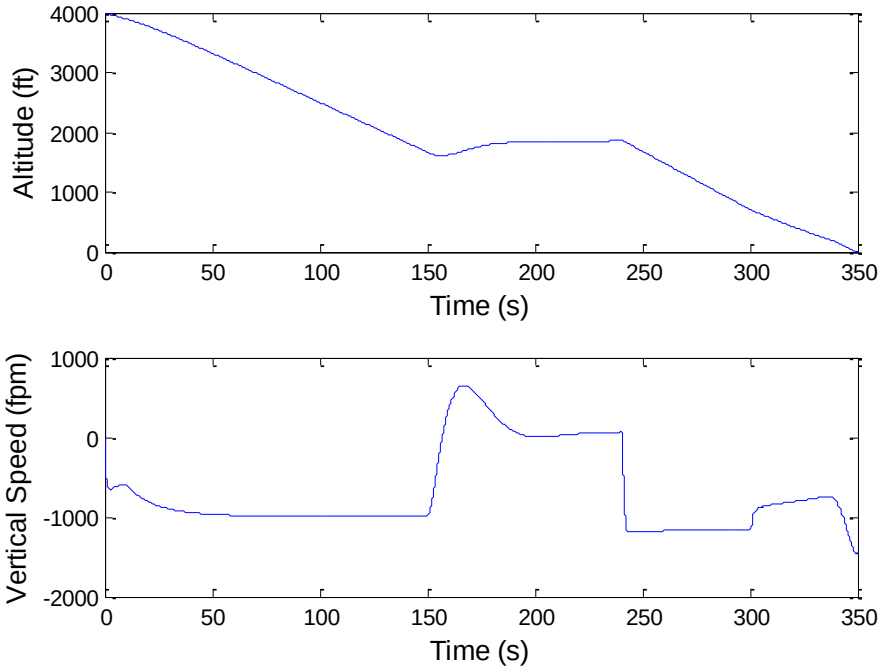

FIGURE 6.2: PILOT ERROR INTRODUCED AT 300 SEC (THRUST REMAINS ON IDLE)

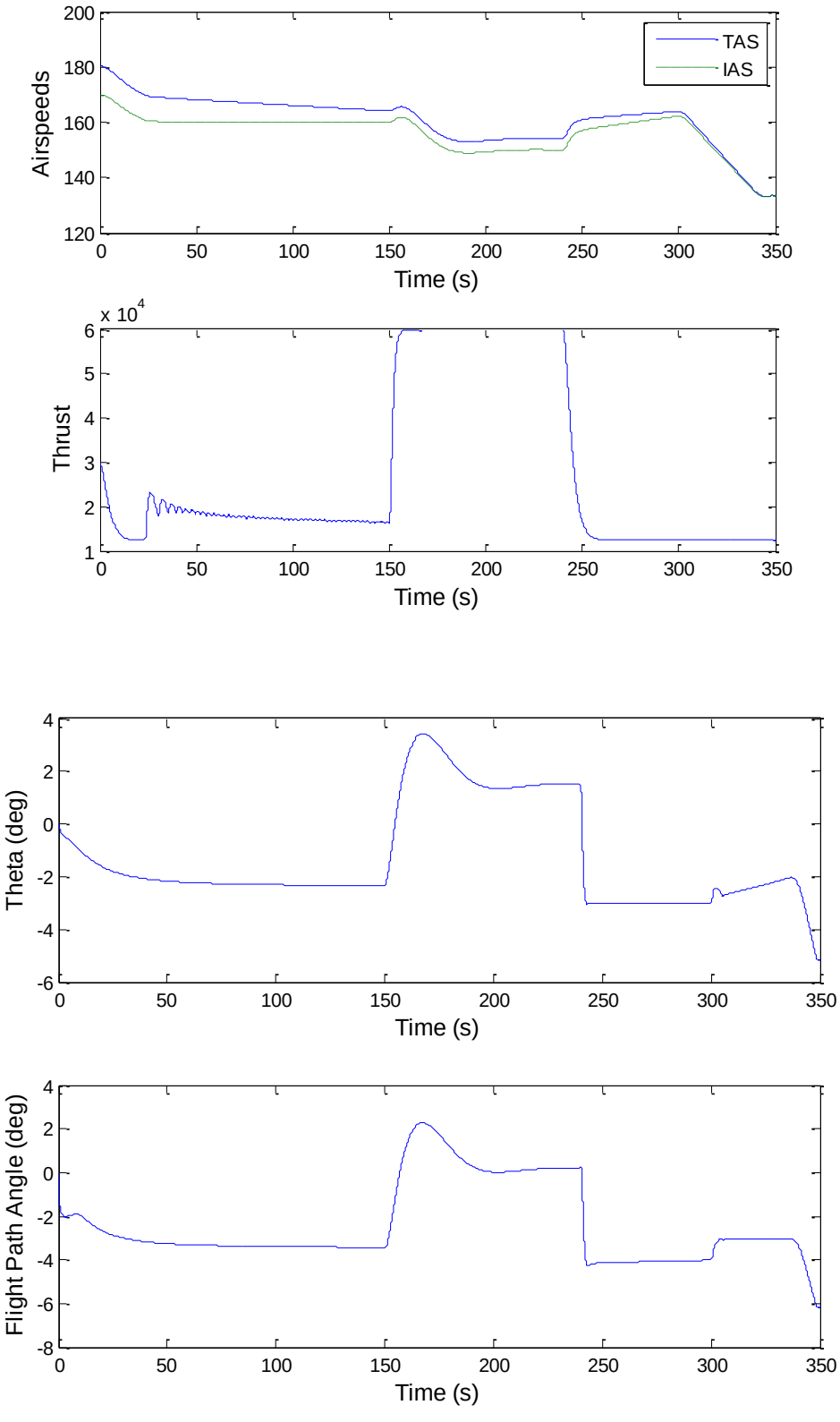

FIGURE 6.2 CONTINUED

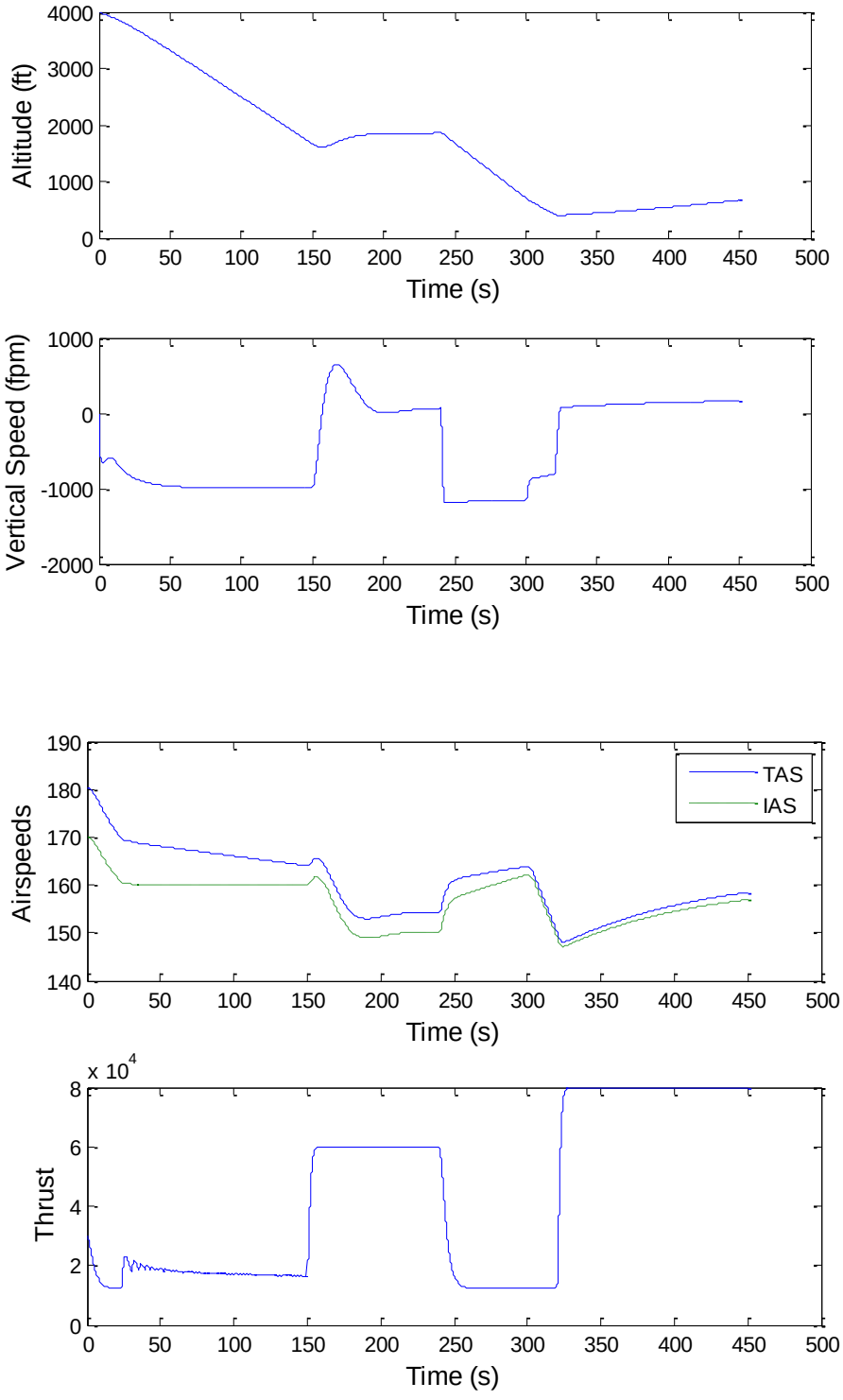

FIGURE 6.3: PILOT FAULT DETECTED AND GO-AROUND INITIATED 20 SECONDS AFTER FAULT INTRODUCTION

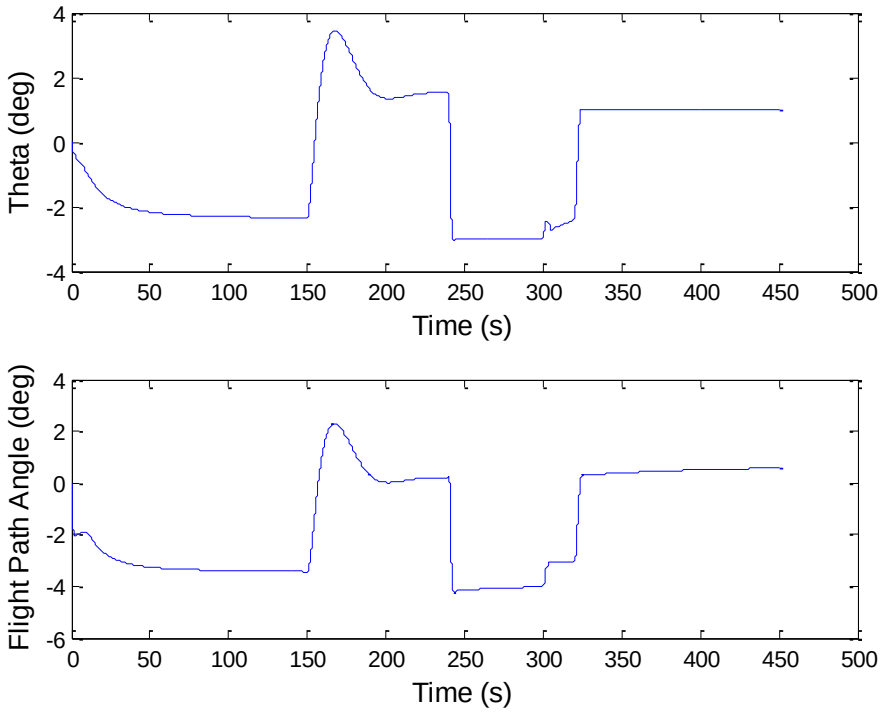

FIGURE 6.3 CONTINUED

6.3 Impact of Fault Recovery Functions

6.3.1 Comparison of Results

As in the previous two case studies, it was found that the fault duration affected the aircraft recovery. It was found that, after the recovery action was initiated, the aircraft continued to descend at a fast rate for a certain time before it recovered. Hence, beyond 40 seconds from the fault introduction, the aircraft was too close to the ground and unable to recover. It was found to hit the ground with a high vertical speed. Figure 6.4 shows the impact of fault recovery on the aircraft with varying fault duration.

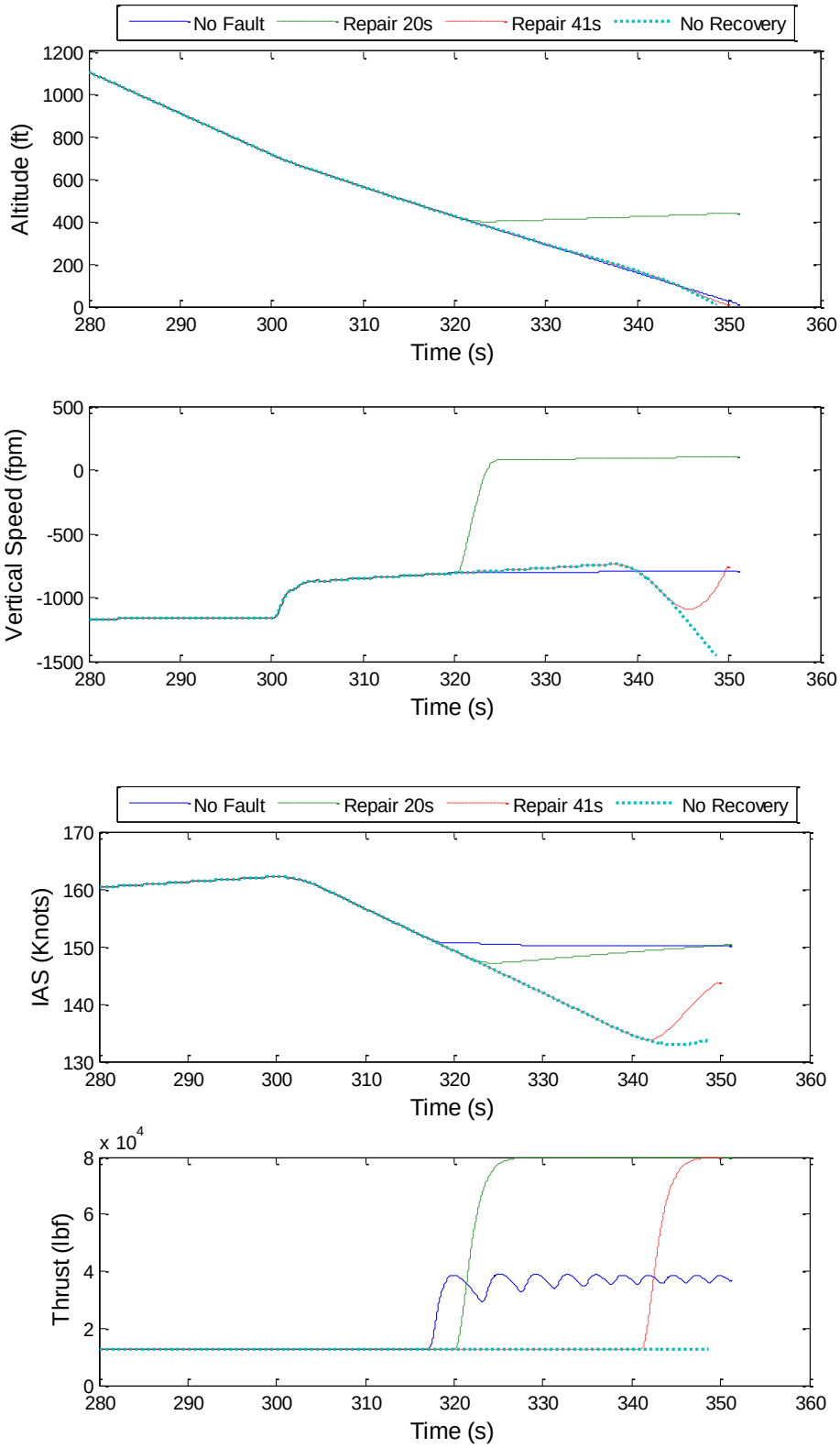

FIGURE 6.4: COMPARISON OF RESULTS FOR PILOT ERROR WHILE CHANGING FLIGHT MODES

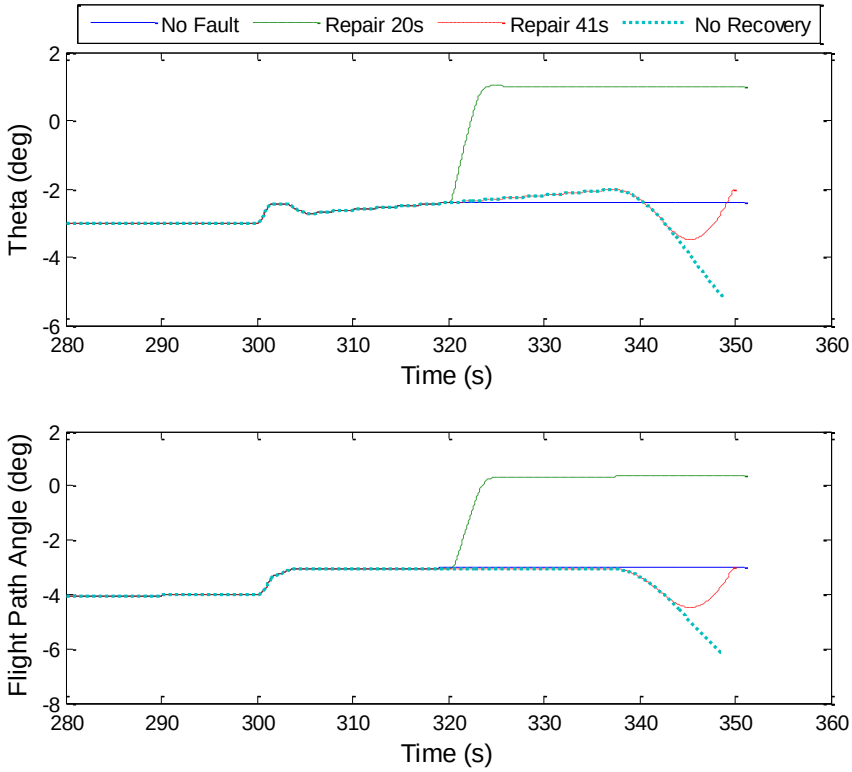

FIGURE 6.4 CONTINUED

6.3.2 Outcome of Study

In this study too it was found that the parameter selected for fault detection affected the time at which the fault could be detected, hence the possibility of aircraft recovery. It was noticed that if the pilot detected the fault when the indicated airspeed reduced below the commanded value, the aircraft could be recovered; however, if the fault was detected only after the aircraft abruptly pitched downwards, the aircraft was too close to the ground to be able to recover.

7 CONCLUSIONS AND FUTURE WORK

7.1 Summary

This thesis proposed a simulation-based method to validate large-scale complex aircraft systems. Examining several aircraft accidents, it was noted that the aircraft behaved in an unexpected manner when conditions on the functioning of a system component were violated. These conditions were defined in the thesis as axiomatic conditions of the system components. The thesis thus proposed that, by identifying the axiomatic set of conditions of the system components and studying the system behavior when these axioms are violated, it is possible to identify and observe potential emergent unexpected behaviors in the system. Thus, the thesis proposes that testing and validation of large scale systems can be fostered by early identification of potential adverse behavior that when:

1. A component is placed outside of its allowable environmental condition and it does not respond properly for that condition;
2. One of the components fails; or
3. All components act as desired, but the system as a whole fails in a given condition.

This thesis showed that, by simulating the violation of component axioms in the integrated system, where the components are modeled as dynamic reference models of their intended key functions, it is possible to quickly determine the parameters to be considered in the development and validation of such complex aircraft systems. It was demonstrated, by simulating three case studies, how emergent behavior can be observed

in the system and the aircraft state using this approach. The method also identified the requirements of any fault management function.

7.2 Contributions

This thesis, first, contributed a method to capture and describe the underlying axiomatic set of conditions of the components of a distributed system. This method describes the general set of parameters to be considered to enable identifying the component axioms. Those parameters capture the requirements and design considerations of the component for its intended function. By identifying the axiomatic conditions of the components of the complex aircraft system to be studied and observing the emergent system behavior when these axioms are violated, possible adverse system behavior can be identified early, focusing test and validation efforts on likely problems.

Second, it contributed a simulation framework for validation which (a) incorporated models of relevant component functions, interactions between the components and a dynamic model of the aircraft, and (b) monitored the key axioms of the components. Through this, it enabled the identification of emergent behavior in a range of scenarios by introducing faults at varying times that violated key axioms and varying the duration after which the fault was detected. Hence, it is possible to understand through simulation of the integrated complex system, all the important considerations to be taken to avoid or recover from a fault which involves the violation of a component axiom. This includes identifying requirements for fault management that recognize key aspects of the dynamics that might limit how long a fault duration can be tolerated, or a relationship between fault detection time and the type of fault recovery that will be required. Further, simulation of component functions enables evaluation of important

failure modes and emergent effects of such faults earlier in the design process than current component-in-the-loop testing allows.

Finally, this thesis demonstrated, using three case studies, the ability to examine the system wide implications of violating the axiomatic conditions of any of the components of the integrated system in the aircraft model as well as subsequent fault management functions. This method may be implemented later in the development process as well, by including more detailed functions or more detailed models of the components. Implementation of this method at every stage in the development process can help determine many fault scenarios which could lead to unfavorable and dangerous behavior in the aircraft and which otherwise go unnoticed. In this way, this method has the ability to scale to a full complex aircraft system, as it models the component functions rather than detailed full scale models of the components. It can also help to ease the process of finding fault management methods for such cases and help prevent those faults by taking all important considerations into account. This method can be especially advantageous when developing and implementing new and novel technologies for safety critical aircraft systems, to ensure that they will satisfy all the safety objectives. At present, it takes several years to implement new ideas for the fear that there is no method to determine with confidence that such systems will function safely.

7.3 Future Work

The work in this thesis looked at some broad areas to consider based on ways in which emergent behavior can be identified in testing based on component axioms. This gave a broad perspective on the application of violating component axioms for system validation. As a future work, it may be required to focus the testing on more specific

areas; for example, specific to the type of system being tested, the number of components or detail of component functions included, or the stage of development the system is in.

Further, detailed advancement may be required in the area of identifying the axiomatic set of conditions for all system components. Through the thesis the definition of axiomatic conditions and the process of identifying them has been understood, however the implementation of this process to systematically determine the entire axiomatic set of conditions of any component may be required.

More specifically, the validation process through this approach would need to be carried out by a system validation group, which would need information from the developers/designers of the components about the requirements and design considerations incorporated into the component design to identify the axiomatic set of conditions of the components based on their functions towards the overall goal of the system being tested. Since this approach can help identify requirements for fault management, it can also potentially be applied by the developers of a fault management system for validation of selected fault management functions, and to test various fault detection and recovery options.

Finally, this approach would need to be applied at every stage in the development process to ensure adequate coverage of all component functions. At the start of the design process, it could be applied to validate the high level functions of the components by violating specific axiomatic conditions for the preliminary design considerations. Further in the design process, more detailed models of the component functions would need to be included and the number of axiomatic conditions would increase.

It would also be interesting to perform a study considering the effects of cascading failures, where a violation of an axiom of one component may lead to the violation of several other axioms. This can be done either through sequential scheduling of axiom violations, or through incorporating the axioms of the component through component interactions such that simulating a violation of one axiom would automatically lead a violation of some other axioms.

REFERENCES

[1] Parunak, H. V. D., and Vanderbok, R. S., "Managing Emergent Behavior in Distributed Control Systems 1," pp. 1–8.

[2] Favarò, F. M., Jackson, D. W., Saleh, J. H., and Mavris, D. N., "Software contributions to aircraft adverse events: Case studies and analyses of recurrent accident patterns and failure mechanisms," Reliability Engineering & System Safety, vol. 113, May 2013, pp. 131–142.

[3] Aircraft Accident Report: Runway Overrun During Rejected Takeoff, Bombardier Learjet 60, N999LJ, Columbia, South Carolina, September 19, 2008, Washington, DC.: 2010.

[4] Transport, A., and Investigation, S., In-flight upset event 240 km north-west of Perth, WA Boeing Company 777-200, 2005.

[5] Aircraft Accident Report: Uncontrolled Descent and Collision with Terrain, Boeing 737-300 USAIR Flight 427, N513AU, Near Aliquippa, Pennsylvania, September 8, 1994, Aliquippa, Pennsylvania: .

[6] SAE International, "ARP4761: Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment," 1996.

[7] Haskins, C., Forsberg, K., Krueger, M., Walden, D., and Hamelin, R. D., eds., Incose Systems Engineering Handbook, 2011.

[8] NAS System Engineering Manual Vol1, 2006.

[9] "Systems and Software Engineering -- Vocabulary," vol. 2010, 2010.

[10] Black, J., and Koopman, P., "System safety as an emergent property in composite systems," 2009 IEEE/IFIP International Conference on Dependable Systems & Networks, 2009.

[11] Bloebaum, C. L., and McGowan, A.-M. R., "The Design of Large-Scale Complex Engineered Systems: Present Challenges and Future Promise," 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Indianapolis, Indiana: 2012, pp. 1–19.

[12] Perrow, C., Normal Accidents: Living with High Risk Technologies, New York: Basic Books, Inc., 1984.

[13] Leveson, N., SafeWare: System Safety and Computers, Addison Wesley Publishing Company Incorporated, 1995.

[14] "Principles of System Safety," FAA System Safety Handbook, 2000.

[15] "Integrated System Hazard Analysis," FAA System Safety Handbook, 2000.

[16] International, S., "Aerospace Recommended Practice 4754 Rev. A: Guidelines for Development of Civil Aircraft and Systems," 2010.

[17] Pritchett, A. R., and Feigh, K. M., "Simulating first-principles models of situated human performance," 2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), IEEE, 2011, pp. 144–151.

[18] Pritchett, A. R., Christmann, H. C., and Bigelow, M. S., "A simulation engine to predict multi-agent work in complex, dynamic, heterogeneous systems," 2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), Feb. 2011, pp. 136–143.

[19] Johnson, E. N., and Pritchett, A. R., "Generic Pilot and Flight Control Model for use in Simulation Studies," AIAA Modeling and Simulation Technologies Conference and Exhibit, 2002.

[20] "Erroneous Flight Instrument Information," AERO Magazine.

[21] Interim Report: Crash into Atlantic Ocean, Airbus A330-203, Air France Flight 447, Paris: 2009.

[22] Bauer, M., Specialist's Factual Report: Airspeed Indication Anomaly, A330-200 TAM Airlines Flight 8091, 2011.

[23] Bauer, M., Specialist's Factual Report: Airspeed Indication Anomaly, A330-323 NorthWest Airlines Flight 008, 2011.

[24] Accident Docket: Impact with Sea Wall during Final Approach, Boeing 777-200ER, Asiana Airlines Flight 214, San Fransisco.