DEMONSTRATION OF THE OPTIMAL CONTROL MODIFICATION FOR GENERAL
AVIATION: DESIGN AND SIMULATION

A Thesis by

Scott Reed

Bachelor of Science, Wichita State University, 2009

Submitted to the Department of Aerospace Engineering
and the faculty of the Graduate School of
Wichita State University
in partial fulfillment of
the requirements for the degree of
Master of Science

December 2010

© Copyright 2010 by Scott Reed

All Rights Reserved

DEMONSTRATION OF THE OPTIMAL CONTROL MODIFICATION FOR GENERAL AVIATION: DESIGN AND SIMULATION

The following faculty members have examined the final copy of this thesis for form and content, and recommend that it be accepted in partial fulfillment of the requirement for the degree of Master of Science with a major in Aerospace Engineering.

James Steck, Committee Chair

Kamran Rokhsaz, Committee Member

John Watkins, Committee Member

DEDICATION

For the many who have mentored and taught me

ACKNOWLEDGEMENTS

This material is based upon work supported by NASA under award number NNXO9AP20A. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and does not necessarily reflect the views of the National Aeronautics and Space Administration.

A special thank you to Nhan Nguyen for allowing me to adapt his Optimal Control Modification to general aviation and for his help in my understanding of his method.

ABSTRACT

This work presents the design and simulation of a model reference adaptive flight control system for general aviation. The controller is based on previous adaptive control research conducted at Wichita State University (WSU) and the National Aeronautics and Space Administration (NASA) Ames Research Center. The control system is designed for longitudinal control of a Beech Bonanza given the commands of pitch rate and airspeed.

The structure of the controller includes a first-order model follower, proportional-integral (PI) controller, inverse controller, and adaptation element. Two adaptation methods were considered, the WSU-developed Adaptive Bias Corrector (ABC) and the Optimal Control Modification (OCM). The ABC is used with two error schemes, adapting to the modeling-error and the tracking-error. Three variations of the OCM are presented, which differ in the parameterization of the adaptive signal. The first is called OCM-Linear (OCM-L), where the adaptive signal is linearly related to the states. The second variation is OCM-Bias (OCM-B), which only includes a bias term. The third is the OCM-Linear and Bias (OCM-LB), a combination of the previous two variations.

To design the controllers, varied values of the PI gains and adaptive gains were evaluated based on time response tracking of a pitch doublet and time delay margin. The time delay margin is based on error metrics developed at NASA Ames.

Of the five controllers presented, the OCM-L and ABC with tracking-error adaptation performed the best. The ABC with modeling-error adaptation did not track the pitch doublet. The OCM-B and OCM-LB are good controllers but had worse performance than OCM-Linear in tracking and time delay margin, respectively.

TABLE OF CONTENTS

Chapter	Page
1. INTRODUCTION.....	1
1.1 Literature Review.....	2
1.1.1 Model Reference Adaptive Control	2
1.1.2 Adaptive Control Research at Wichita State University	4
1.2 Motivation and Thesis Structure	6
2. CONTROL ARCHITECTURE.....	9
2.1 Commands.....	9
2.2 Model Follower	11
2.3 Proportional-Integral Controller.....	13
2.4 Inverse Controller.....	14
2.5 Adaptation	14
2.5.1 Adaptive Bias Corrector.....	15
2.5.2 Optimal Control Modification.....	18
2.6 Architecture Differences	22
3. METHOD OF ANALYSIS AND DESIGN.....	23
3.1 Aircraft Model.....	23
3.2 Time Response Tracking.....	24
3.3 Error Metric Study	26
3.4 Gain Tuning Process	28
4. RESULTS.....	30
4.1 Adaptive Bias Corrector Controllers.....	31
4.2 Optimal Control Modification Controllers.....	42
4.3 Comparisons of ABC and OCM Controllers	58
5. CONCLUSION	62
5.1 Adaptive Bias Corrector Adaptation.....	62
5.2 Optimal Control Modification Adaptation.....	63
5.3 Future Work and Recommendations.....	63
REFERENCES.....	64

TABLE OF CONTENTS (continued)

Chapter		Page
APPENDICES.....		68
A.	Simulation Routine for Inverse Controller.....	69
B.	Simulation Routine for ABC with Modeling-Error Adaptation.....	72
C.	Simulation Routine for ABC with Tracking-Error Adaptation.....	74
D.	Simulation Routine for OCM Adaptation	76
E.	Simulation Routine for Velocity-Loop Adaptation.....	79

LIST OF FIGURES

Figure	Page
1. NASA pilot evaluations of adaptive control in motion based simulator.....	7
2. Flight controller overview.....	10
3. Model follower in Simulink programming	12
4. ABC with modeling-error adaptation linear block diagram.....	17
5. ABC with tracking-error adaptation linear block diagram.....	17
6. Three view drawing of Beech Bonanza	24
7. Time response tracking pitch rate doublet command.....	25
8. Time response tracking pitch rate doublet model	26
9. Example of error metric study.....	27
10. Time response for ABC with modeling-error adaptation at 65 knots.....	32
11. Time response for ABC with modeling-error adaptation at 100 knots.....	33
12. Time response for ABC with modeling-error adaptation at 165 knots.....	34
13. Error metric study for ABC with modeling-error adaptation.....	35
14. Time response for ABC with tracking-error adaptation at 65 knots.....	36
15. Time response for ABC with tracking-error adaptation at 100 knots.....	37
16. Time response for ABC with tracking-error adaptation at 165 knots.....	38
17. Error metric study for ABC with tracking-error adaptation.....	39
18. ABC comparison at 65 knots	40
19. ABC comparison at 100 knots	41
20. ABC comparison at 165 knots	42
21. Time response for OCM-L at 65 knots	44

LIST OF FIGURES (continued)

Figure	Page
22. Time response for OCM-L at 100 knots	45
23. Time response for OCM-L at 165 knots	46
24. Error metric study for OCM-L	47
25. Time response for OCM-B at 65 knots	48
26. Time response for OCM-B at 100 knots	49
27. Time response for OCM-B at 165 knots	50
28. Error metric study for OCM-B	51
29. Time response for OCM-LB at 65 knots	52
30. Time response for OCM-LB at 100 knots	53
31. Time response for OCM-LB at 165 knots	54
32. Error metric study for OCM-LB	55
33. OCM comparison at 65 knots	56
34. OCM comparison at 100 knots	57
35. OCM comparison at 165 knots	58
36. ABC and OCM comparison at 65 knots	59
37. ABC and OCM comparison at 100 knots	60
38. ABC and OCM comparison at 165 knots	61

LIST OF TABLES

Table	Page
1. ABC Controller with Modeling-Error Adaptation: Final Gains	31
2. ABC Controller with Tracking-Error Adaptation: Final Gains	35
3. Optimal Control Modification Controllers: Final Gains	43

LIST OF ABBREVIATIONS

ABC	Adaptive Bias Corrector(s)
AGATE	Advanced General Aviation Transportation Experiment
AIAA	American Institute of Aeronautics and Astronautics
ANN	Artificial Neural Network
DOF	Degree(s) of Freedom
FAR	Federal Aviation Regulations
HBC	Hawker Beechcraft Corporation
IRAC	Integrated Resilient Aircraft Control
MRAC	Model Reference Adaptive Control
NASA	National Aeronautics and Space Administration
OCM	Optimal Control Modification
OCM-L	Optimal Control Modification-Linear
OCM-B	Optimal Control Modification-Bias
OCM-LB	Optimal Control Modification-Linear and Bias
PD	Proportional-Derivative
PID	Proportional-Integral-Derivative
PI	Proportional-Integral

LIST OF SYMBOLS

C_{D_K}	Drag coefficient due to lift
$C_{D_{\delta e}}$	Drag coefficient due to elevator deflection
$(C_D)_{\delta e=0}$	Drag coefficient at zero elevator deflection
C_{D_0}	Drag coefficient due to parasite drag
C_{L_i}	Lift coefficient due to tail incidence
C_{L_α}	Lift coefficient due to angle of attack
C_{L_0}	Lift coefficient at zero angle of attack
C_{M_q}	Moment coefficient due to pitch rate
C_{M_α}	Moment coefficient due to angle of attack
$C_{M_{\delta e}}$	Moment coefficient due to elevator deflection
$(C_M)_{\delta e=0}$	Moment coefficient due to zero elevator deflection
C_{M_0}	Moment coefficient at zero angle of attack
$\bar{c}$	Mean geometric chord
d_T	Thrust moment
e	Error
g	Gravitational acceleration, 32.2 ft/sec
K_d	PID derivative gain
K_i	PID integral gain
K_p	PID proportional gain

LIST OF SYMBOLS (continued)

I_{yy}	Moment of inertia of the pitch axis
J	Optimal control cost function
M	Error metric
m	Mass
p	Roll rate
Q	Optimal control cost function weighting matrix
q	Pitch rate
q_{com}	Pitch rate command
q_e	Pitch rate error
q_m	Pitch rate from model
$\dot{q}$	Pitch acceleration
$\dot{q}_{add}$	Pitch acceleration correction
$\dot{q}_{com}$	Pitch acceleration command
$\dot{q}_{des}$	Pitch acceleration desired
$\dot{q}_m$	Pitch acceleration from model
$\bar{q}$	Dynamic pressure
$\hat{q}$	Non-dimensional pitch rate
r	Yaw rate
S	Wing reference area
T	Thrust

LIST OF SYMBOLS (continued)

T_r	Rise time, pitch loop
T_v	Rise time, velocity loop
W	ABC weight
$\dot{W}$	ABC weight update law
V	Forward airspeed
V_e	Forward airspeed error
$\dot{V}$	Forward acceleration
$\dot{V}_{add}$	Forward acceleration correction
$\dot{V}_{com}$	Forward acceleration command
$\dot{V}_{des}$	Forward acceleration desired
ν	OCM damping gain
α	Angle of attack
Γ	OCM adaptive gain matrix
γ	Flight path angle
γ_q	OCM adaptive gain on pitch rate
γ_α	OCM adaptive gain on angle of attack
γ_θ	OCM adaptive gain on theta
δ_e	Elevator deflection
Θ	OCM adaptation weight matrix
Θ_B	OCM-Bias adaptation weight

LIST OF SYMBOLS (continued)

$\dot{\Theta}$	OCM-Linear weight update law
$\dot{\Theta}_B$	OCM-Bias weight update law
θ	Pitch angle
η	ABC adaptive gain or learning rate
Φ	OCM adaptation regression vector
ϕ_T	Thrust line angle relative to fuselage axis
ω_n	Natural frequency of pitch axis
ζ	Damping ratio

CHAPTER 1

INTRODUCTION

Since aircraft began flying more than 100 years ago, they have required control. An aircraft is not useful unless it can be directed to a particular location. Controls in early aircraft and in general aviation today are direct mechanical linkages to the control surfaces. With smaller and faster computers, the use of more complex flight control systems can make flying easier and safer for pilots. Today's commercial airliners use flight control computers and fly-by-wire control actuation to increase the safety of commercial flight with the addition of safety features that cannot be implemented with mechanical control linkages.

The use of adaptive control has the potential to allow advanced flight control systems to be used in general aviation. But many of these advantageous systems have yet to reach general aviation because of their cost and the need for redundancy associated with such systems. Commercial airliners are certified to use computer-based flight control systems, in part because they have multiple redundant control and actuator systems. Adaptive control is an alternative to redundancy and has the capacity to surpass the utility of redundant systems. Adaptive systems adjust to flight conditions online without the need to know potential failures beforehand. In this way, these systems provide the safety required without the need for the expense and weight of redundant flight controls.

Another development in aircraft controls is the use of decoupled flight control systems. Decoupled flight controls could expand the pool of pilots, since these systems reduce pilot workload and are more intuitive to fly. In decoupled flight control systems, pilot commands are designed to only affect one aircraft state. This control approach has the potential to reduce the

amount of training required to be licensed to pilot aircraft, and therefore would open opportunities for more individuals to become pilots.

1.1 Literature Review

1.1.1 Model Reference Adaptive Control

Model Reference Adaptive Control (MRAC) is an adaptive control methodology that allows for a nonlinear plant to follow the dynamics of a desired linear system. In order to drive the nonlinear plant to the desired model dynamics, an adaptation signal is added to the control. This adaptive signal can be generated by a neural network or some other adaptive element. The adaptation element can also be used to correct for unmodeled system dynamics and changes in the plant due to failures on the aircraft. A good overview of the general MRAC used in aerospace research today was written by Campbell et al. [1].

Early MRAC methods were introduced in the mid-1980s to be used with inverse controllers for robots. Neural networks were used to adjust the robot controllers to account for unmodeled system dynamics [2] [3]. Over the following decade, MRAC was implemented for aircraft by Calise and Rysdyk [4] and Rysdyk and Calise [5]. They combined the use of adaptive control techniques with a linear inverse aircraft controller and formed the basis for most MRAC variations used in aircraft control theory today .

MRAC has been augmented to include methods to increase the robustness of adaptive controllers. Standard MRAC without any modification is very sensitive to time delay and gross changes in the plant, thus producing instability. A variety of modifications have been developed to alleviate this issue. Volyanskyy et al. [6] modified the MRAC method of Calise and Rysdyk to include an extra term in the weight update rule, which also improved the rate of convergence. This method, named the Q modification, was simulated with good results on a model of the

Boeing X-45 by Kutay et al. [7]. More recently Yucelen et al. [8] researched σ and e modifications, which are extensions of the Q modification and add terms to the adaptive law in order to increase the adaptive controller's robustness and adaptive speed. Another method to increase robustness in adaptive control is to recover the stability margins of the original plant, combining the techniques of loop recovery with adaptive control to form Adaptive Loop Recovery as developed by Calise et al. [9].

Researchers at the NASA Ames Research Center have been developing methods of model reference adaptive control. One method researched at Ames is Hybrid Adaptive Control, which includes two adaptive signals: one direct and the other indirect. The direct method is similar to other MRAC adaptations, but the indirect adaptation computes online estimates of the dynamics of the plant due to damage. The indirect adaptation has been demonstrated with both a neural network and by a recursive least-squares method [10] [11]. Elements of these previous controller designs were used in the development of this thesis, which will be explained in greater detail in Chapter 2. Nguyen et al. [12] and Nguyen [13] developed an adaptation method called the Optimal Control Modification (OCM). The weight update rule of this adaptive law was formulated using optimal control techniques. The OCM adds a damping modification term to the standard MRAC adaptive law to increase robustness. This research is the basis of the adaptation method implemented in this thesis for general aviation and will be described in greater detail in Chapter 2, Section 2.5.2.

Researchers at NASA Ames examined many of these MRAC techniques, plus others, using the same simulation aircraft model. Campbell et al. [1] tested the performance of each controller with added delay between the controller and aircraft in cases of actuator and aircraft damage and presented general observations for each MRAC technique.

1.1.2 Adaptive Control Research at Wichita State University

Research in adaptive control of general aviation aircraft at Wichita State University (WSU) began in the mid-1990s. The city of Wichita is home to many general aviation aircraft producing companies, including Hawker Beechcraft, Cessna, and Bombardier-Learjet; therefore, the introduction of advanced flight control systems in general aviation is relevant to this community.

An early control method designed for general aviation was researched by Duerksen [14] for his doctoral dissertation. In his research funded by NASA's Advanced General Aviation Transportation Experiment (AGATE) program, he developed a longitudinal decoupled flight controller with fuzzy logic that commands flight path angle and airspeed. In 2002, Beringer [15] used Duerksen's controller to conduct simulated pilot testing. He tested the fuzzy logic decoupled controller with pilots of varying degrees of training and found that the controller reduced pilot workload and was preferred, in terms of pilot effort, over conventional control systems.

Steck et al. [16] researched an adaptive control system with decoupled pilot commands. These efforts resulted in a flight control system that included a linear compensator and an Artificial Neural Network (ANN) that was trained offline. This technique was simulated with a general longitudinal delta wing model and included the pitch attitude and airspeed commands.

In the early part of this decade, the research turned to adaptive inverse control design. This began with Pesonen et al. [17] simulating a one Degree of Freedom (DOF), elevator deflection, inverse controller with an artificial neural network using the ANN toolbox in Matlab. The aircraft model and inverse controller were then expanded to a 3-DOF longitudinal simulation of a Beech Bonanza, a general aviation aircraft [17].

In the next year, the 3-DOF decoupled inverse adaptive controller was given the addition of a Proportional-Derivative (PD) linear controller and flown in a piloted simulation [18] [19]. During simulated flight testing over a variety of test conditions and simulated errors, pilots rated the adaptive inverse controller as being an average of 7 on the modified Cooper-Harper rating scale. This rating indicates fair to minor, but annoying, deficiencies in controlling the aircraft.

The goal of developing new control systems is that they will be certifiable for aircraft. As a follow on to the previous controller, the impacts on FAR-23 were examined by Steck et al. [20]. The regulations of FAR-23 were written for more classical methods of aircraft control design. With the introduction of adaptive control systems, some of these regulations would no longer be applicable or need to be changed to adjust to the new control architecture.

The Wichita State University developed MRAC was tested by Steck et al. [21] using turbulence in the aircraft simulation to determine the controller's performance in the presence of atmospheric disturbances. The adaptive dynamic inverse flight controller was able to track flight path angle and airspeed well in the presence of turbulence with the introduction of a filter on the commanded elevator.

In 2005, a new method of adaptation called the Adaptive Bias Corrector (ABC) was developed at WSU. This method is a simple adaptation scheme that will be discussed in more detail in Chapter 2. This adaptation method, as well as a controller that included envelope prevention schemes, was presented to the FAA in a report by Steck et al. [22].

Over the past year, the dynamic inverse adaptive flight controller has been revisited with the help of a grant from NASA and the Integrated Resilient Aircraft Control (IRAC) project. The research by Lemon et al. [23] reexamined the gains in the PD controller, with the use of the

newly developed stability metrics for adaptive control [24] [25]. A new dynamic inverse controller was developed using all three of the longitudinal equations of motion [23].

1.2 Motivation and Thesis Structure

The above-mentioned work at Wichita State University was conducted using a decoupled flight controller that commands the flight path angle, γ , or pitch angle, θ . This method was designed with an emphasis on non-pilots and new pilots, as the commanding of γ is more intuitive to a public that has experience driving motor vehicles. Experienced pilots are more familiar with commanding pitch rate, q , having generally only used mechanical controls. The task of the author was to develop a flight control system to track a pilot's given pitch rate command and to add an implementation of the OCM. In the research completed at NASA Ames, the aircraft rotational rates were used as their commands, and so elements of their control architecture is the foundation of this thesis.

In the previously mentioned computer simulation of various MRAC designs completed by NASA, the OCM performed in the middle for most tests, based on the error metrics used. It had the least error in the roll axis for the case of performing the standardized longitudinal doublet maneuver with artificial cross-coupling. Campbell et al. [1] noted that the OCM added little complexity to the traditional MRAC controller.

In a motion based flight simulator study completed at NASA Ames, the OCM was rated well by pilots, a summary of which is shown in **Error! Reference source not found.** [26]. As can be seen, the OCM improved the Cooper-Harper rating during all failures. To this end, the Optimal Control Modification was chosen for the author to explore in this thesis.

My contributions to the research of MRAC control systems and adaptive control are as follows:

- Development of a longitudinal flight controller for commanding pitch rate and airspeed based on controllers researched by Nguyen et al. and previous research at WSU.
- Implementation of the OCM developed by Nguyen et al. in the developed flight controller.
- Modification of the WSU-developed ABC for pitch rate, adapting both the tracking-error and the modeling-error.
- Development of an OCM simplification as a bias neural network element, and comparison of this to the implemented OCM and the use of both together.
- Comparison of the ABC and the OCM adaptation methods.

Cooper-Harper Rating Differentials

Figure 1. NASA pilot evaluations of adaptive control in motion based simulator [26].

This thesis is organized as follows: Chapter 2 describes the control architecture of the pitch rate commanded, longitudinal adaptive flight control system, including explanations of the model follower, inverse controller, and adaptation techniques. Chapter 3 is a description of the method of analysis and design, including a description of the aircraft model, time domain response analysis, error metric studies used to evaluate the controller's performance, and design procedure. Chapter 4 contains results from simulation in the Matlab/Simulink environment. Results are presented first for those controllers with the ABC adaptation, followed by those with the OCM adaptation, and then a comparison between the two methods. The thesis concludes with Chapter 5. The appendices show some of the routines used in the Matlab/Simulink simulation.

CHAPTER 2

CONTROL ARCHITECTURE

The overall structure of the controller is shown in Figure 2. The pilot commands the pitch rate, q_{com} , and desired airspeed, V_{com} . The pitch rate command is passed through a first-order model follower to generate q_m , the model pitch rate, and its derivative, $\dot{q}_m$. The tracking-error, $q_m - q$, is then passed through a Proportional-Integral (PI) controller, giving $\dot{q}_{des}$. This signal is added to $\dot{q}_m$ and $\dot{q}_{add}$ to generate $\dot{q}_{com}$, an input to the inverse controller. The $\dot{V}_{com}$ input to the inverse controller is the error in the airspeed divided by the velocity loop time constant with the addition of the adaptive signal, $\dot{V}_{add}$. The inverse controller solves the longitudinal aircraft equations of motion, given the two acceleration commands and the states as inputs, for the required thrust and elevator deflection for the aircraft. These are then passed to the aircraft model. The aircraft model outputs the current aircraft states that are used in the inverse controller and in the adaptation. The adaptation requires $\dot{V}_{des}$ and $\dot{q}_{des}$ or q_m from the controller, states from the aircraft, and the actual engine thrust and elevator deflection. With this input, the signals, $\dot{V}_{add}$ and $\dot{q}_{add}$, are given as output. Subsequent sections of this chapter will provide more detail on the model follower, the PI controller, the inverse controller, and the adaptation methods used. The aircraft model is described in Chapter 3.

2.1 Commands

The commands to the controller are the pitch rate and airspeed. With this architecture, these two commands are decoupled in the sense that during most flight conditions, a command in one will not affect the other. For example, without this decoupling, pitching the nose up causes the aircraft to lose airspeed, and pitching the nose down causes an increase in aircraft speed.

Figure 2. Flight controller overview

With a pilot in the loop, the airspeed command is given by the position of a speed lever, where the position indicates a particular airspeed. The pitch rate command is given by a pilot pulling back or pushing forward on a joystick or yoke. The position of the stick will determine a particular pitch rate command. When the aircraft has achieved the desired pitch attitude, returning the stick to neutral will leave the aircraft at that pitch attitude until another pitch rate command is given.

The command method differs from previous research conducted at WSU, which uses the commands of flight path angle and airspeed, called E-Z fly. The E-Z fly system commands are better suited to less-experienced pilots since they are based on a position command rather than a rate command, the former being more intuitive.

2.2 Model Follower

The first element of the controller in the pitch loop is the model follower. This model represents the desired dynamics of the system. Due to the extensive work by Nguyen et al. [10] in commanding rotational rates in adaptive control, this element of the controller is adapted from that work in Section V, Part B.

The reference model is a first-order system calculating ω_m for the input of the vector of commanded aircraft rotational rates, ω_c , and is given by Nguyen et al. [10] in their equation (40) as

$$\dot{\omega}_m + \omega_n \omega_m = \omega_n \omega_c \quad (2.1)$$

where ω_n is the matrix of natural frequencies of each axis along the diagonal.

Equation (2.1) has been applied to only the longitudinal direction for the pitch rate command, q_{com} , to yield

$$\dot{q}_m + \omega_n q_m = \omega_n q_{com} \quad (2.2)$$

where ω_n is the natural frequency of the pitch axis.

In the frequency domain, the model follower is represented as

$$q_m = \frac{\omega_n}{s + \omega_n} q_{com} \quad (2.3)$$

This is a first-order transfer function with a time constant of $\frac{1}{\omega_n}$.

For the use of this controller's design and evaluation, ω_n was calculated based on the desired rise time of the system, T_r , as

$$\omega_n = \frac{2.2}{T_r} \quad (2.4)$$

In order to gain the full benefits of the use of a Proportional-Integral-Derivative (PID) controller without taking the derivative of the potentially noisy error signal, the derivative portion is taken directly from the model as $\dot{q}_m$. This is then added to the output of the PI controller and the adaptive signal to form $\dot{q}_{com}$. In the Simulink environment, the model follower is programmed as shown in Figure 3. This bypasses the need to take a direct derivative of q_m to generate $\dot{q}_m$.

Figure 3. Model follower in Simulink programming

This method is shown to be the equivalent of equation (2.2) by rearranging that equation as

$$\dot{q}_m = \omega_n (q_{com} - q_m) \quad (2.5)$$

It can be seen from Figure 3 that $\dot{q}_m$ is equal to the gain block of ω_n multiplied by the difference of q_{com} and q_m . Then q_m is $\dot{q}_m$ integrated, which in the frequency domain is represented as $\frac{1}{s}$ so that

$$q_m = \frac{1}{s} \dot{q}_m \quad (2.6)$$

This can be seen in Figure 3 using the Integrator block in Simulink.

2.3 Proportional-Integral Controller

The second element of the pitch rate loop of the controller is a PI controller. The PI was also taken from the work of Nguyen et al. [10]. This might be considered a PID, with the derivative term coming from the model and with a derivative gain of unity. In a linear analysis of the controller in the case where the inverse controller and the aircraft exactly cancel to an integrator, using a gain other than unity for K_d will result in q not exactly following q_m . The PI controller in the frequency domain is represented as

$$\dot{q}_{des} = \left(\frac{K_i}{s} + K_p \right) q_e \quad (2.7)$$

where $q_e = q_m - q$.

Nguyen et al. [10] recommend relating the gains to the desired dynamics of the system for a given damping ratio, ζ , and natural frequency, ω_n , in the case of the longitudinal controller

$$K_p = 2\zeta\omega_n \quad (2.8)$$

$$K_i = \omega_n^2 \quad (2.9)$$

These recommended gains did not produce the desired time response tracking for the inverse controller and aircraft model used, and so were modified by the method described in Section 3.4.

2.4 Inverse Controller

The inverse controller used in this control system was previously developed at Wichita State University. Inputs to the inverse controller are the commanded pitch acceleration and forward acceleration, $\dot{q}_{com}$ and $\dot{V}_{com}$. The equations of motion for the aircraft are then solved to generate the required control values of elevator deflection, δ_e , and thrust, T , to achieve the desired accelerations [18] [23]. An approximate solution to the equations of motion can be found assuming $C_{D_{\delta_e}} \approx 0$ so that

$$T = \frac{1}{\cos(\alpha + \phi_T)} \left[m\dot{V}_{command} + mg \sin \gamma + \bar{q}S(C_D)_{\delta_e=0} \right] \quad (2.10)$$

$$\delta_e = \frac{I_{yy}}{\bar{q}S\bar{c}C_{M_{\delta_e}}} \dot{q}_{command} - \frac{(C_M)_{\delta_e=0}}{C_{M_{\delta_e}}} + \frac{d_T}{\bar{q}S\bar{c}C_{M_{\delta_e}}}(T) \quad (2.11)$$

where $(C_D)_{\delta_e=0} = C_{D_0} + C_{D_K}(C_{L_0} + C_{L_\alpha}\alpha - C_{L_i})^2$ and $(C_M)_{\delta_e=0} = C_{M_0} + C_{M_\alpha}\alpha + C_{M_q}\hat{q}$.

The values for the aerodynamic coefficients, and geometric and mass data were provided by Hawker Beechcraft for a Beech Bonanza single-engine general aviation aircraft. The inverse controller uses the aerodynamic coefficients of a model of the aircraft at 100 knots.

2.5 Adaptation

The key to model reference adaptive control is the adaptive element. The adaptive element adjusts the inputs to the inverse controller to account for the unmodeled dynamics between the aircraft and the inverse controller. This adjustment allows the handling of nonlinear

effects in the plant as well as changes in the plant due to failures and damage to the aircraft. Two methods of adaptation are presented. The first is the WSU-developed Adaptive Bias Corrector, which is used with two different error schemes. The second adaptation method is the Optimal Control Modification. Three variants of this adaptive element are discussed.

2.5.1 Adaptive Bias Corrector

The ABC was developed at Wichita State University as a simple adaptive element. Earlier-researched controllers used larger artificial neural network architectures and computationally expensive learning rules. The ABC adaptation uses a simple bias neural network element that parameterizes the modeling or tracking-error to form the corrective signals of $\dot{q}_{add}$ and $\dot{V}_{add}$ such that

$$U_{add} = W \quad (2.12)$$

where W is the ANN weight that is updated online by the update law:

$$\dot{W}(t) = \eta e(t) \quad (2.13)$$

where η is the adaptive gain or learning rate, and e is the modeling or tracking-error.

The modeling-error has been used historically at WSU. The modeling-error is the difference in acceleration such that

$$e(t) = \dot{x}_{des} - \dot{x} \quad (2.14)$$

where $\dot{x}_{des}$ is either $\dot{q}_{des}$ or $\dot{V}_{des}$. The desired pitch acceleration, $\dot{q}_{des}$, is the output of the PI controller, and the desired forward acceleration, $\dot{V}_{des}$, is the output of the velocity loop gain or

$$\dot{V}_{des} = \frac{1}{T_v} V_e \quad (2.15)$$

where T_v is the design rise time of the velocity loop. *All versions of the controller presented in this thesis use ABC adaptation with modeling-error for the velocity loop.* These are the same adaptations used in the work by Lemon et al. [23]. For all controllers, $T_v = 15$ and $\eta = 0.05$ in the velocity loop.

The other error method is the tracking-error. This was investigated with the ABC and the OCM, which is standard for the OCM. Instead of error in equation (2.14), the tracking-error is

$$e(t) = x_m - x \quad (2.16)$$

Since this is only used for adaptation in the pitch loop

$$q_e = q_m - q \quad (2.17)$$

This represents the difference of the aircraft in tracking the reference model as given in the output of the model follower.

Both versions of the ABC can be represented as a linear system based on the assumption that the inverse controller and the aircraft perfectly cancel to form an integrator, $\frac{1}{s}$. Figure 4 shows the block diagram for the ABC with modeling-error adaptation, and Figure 5 shows the block diagram for the ABC with tracking-error adaptation. These include the term u , which represents a disturbance or unmodeled dynamics.

Figure 4. ABC with modeling-error adaptation linear block diagram

Figure 5. ABC with tracking-error adaptation linear block diagram

From these block diagrams the transfer function for the pitch rate can be determined. For the ABC with modeling-error adaptation, the pitch rate is

$$q = \frac{u}{(s + \eta) \left(1 + \frac{K_p}{s} + \frac{K_i}{s^2} \right)} + \frac{\omega_n}{s + \omega_n} \cdot q_c \quad (2.18)$$

Simplifying the denominator of the u term yields

$$q = \frac{s^2}{s^3 + (K_p + \eta)s^2 + (K_i + \eta K_p)s + \eta K_i} u + \frac{\omega_n}{s + \omega_n} q_{com} \quad (2.19)$$

For the ABC with tracking-error adaptation the pitch rate is

$$q = \frac{u}{s + \frac{\eta}{s} + K_p + \frac{K_i}{s}} + \frac{\omega_n}{s + \omega_n} q_{com} \quad (2.20)$$

And simplified it becomes

$$q = \frac{s}{s^2 + k_p s + k_i + \eta} u + \frac{\omega_n}{s + \omega_n} q_c \quad (2.21)$$

In both cases should u be zero, the pitch rate will track the model exactly. Further linear analysis is not presented as part of this thesis.

2.5.2 Optimal Control Modification

The Optimal Control Modification is an augmentation of the original MRAC to increase the robustness of adaptive control and allow for high adaptive gains without oscillations. The standard MRAC adaptive law is given the addition of a damping term that requires persistent excitation. The OCM is termed as such because the weight update law is derived using optimal control techniques.

The implementation of the OCM for general aviation is taken from Nguyen et al. [12], Section III, with the full derivation given in Section II. The adaptive signal of the OCM, u_{add} , is parameterized as

$$u_{add} = \Theta^T \Phi(x) \quad (2.22)$$

where Θ is the adaptive weight, and Φ a chosen regression vector that is a function of the states. Specific forms of the adaptive signal will be discussed later.

The OCM method pertains especially to the weight update law, which is given by Nguyen et al. [12] in their equation (109) as

$$\dot{\Theta} = -\Gamma \Phi (e^T P b + \nu \Phi^T \Theta K_i^{-2}) \quad (2.23)$$

where Γ is the matrix of adaptive gains, e is the tracking-error as in equation (2.24), and P and b are related to the error dynamics of the system and are given in equations (2.25) and (2.29), respectively. The damping term is the second term in parentheses of equation (2.23). The damping gain, ν , is selected to improve the robustness of the adaptive law. A method of analytical selection of ν is given by Nguyen [13].

This weight update law is expanded using definitions from Nguyen et al. [12]:

$$e = \begin{bmatrix} \int_0^t x_e d\tau \\ x_e \end{bmatrix} \quad (2.24)$$

$$P = \begin{bmatrix} \frac{K_p}{K_i} + \frac{K_i + 1}{K_p} & \frac{1}{K_i} \\ \frac{1}{K_i} & 1 + \frac{1}{K_i} \end{bmatrix} \quad (2.25)$$

where P is the solution to the Lyapunov equation

$$PA_c + A_c^T P = -Q \quad (2.26)$$

and $Q = 2I$, and is the weight of the cost function, J , given by Nguyen et al. [12] as

$$J = \frac{1}{2} \int_0^{t_f} (e - \Delta)^T Q (e - \Delta) dt \quad (2.27)$$

where Δ is the tracking-error at $t = t_f$.

A_c and b are defined by Nguyen et al. [12] by the error dynamics of the system so that

$$A_c = \begin{bmatrix} 0 & I \\ -K_i & -K_p \end{bmatrix} \quad (2.28)$$

$$b = \begin{bmatrix} 0 \\ I \end{bmatrix} \quad (2.29)$$

Expanding the first term in the parentheses of equation (2.23) for only the longitudinal axis, q , yields

$$e^T Pb = \begin{bmatrix} \int_0^t q_e d\tau & q_e \end{bmatrix} \cdot \begin{bmatrix} \frac{K_p}{K_i} + \frac{K_i+1}{K_p} & \frac{1}{K_i} \\ \frac{1}{K_i} & \frac{K_i+1}{K_p K_i} \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (2.30)$$

$$e^T Pb = \begin{bmatrix} \int_0^t q_e d\tau & q_e \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{K_i} \\ \frac{K_i+1}{K_p K_i} \end{bmatrix} \quad (2.31)$$

$$e^T Pb = \frac{\int_0^t q_e d\tau}{K_i} + q_e \left(\frac{K_i+1}{K_p K_i} \right) \quad (2.32)$$

Combining equation (2.32) with equation (2.23) yields

$$\dot{\Theta} = -\Gamma \Phi \left(\frac{\int_0^t q_e d\tau}{K_i} + q_e \left(\frac{K_i+1}{K_p K_i} \right) + \frac{\nu \Phi^T \Theta}{K_i^2} \right) \quad (2.33)$$

where Γ is the matrix of adaptive gains (learning rates). For the choice of Φ in equation (2.36) the adaptive gain matrix is

$$\Gamma = \begin{bmatrix} \gamma_q & 0 & 0 \\ 0 & \gamma_\theta & 0 \\ 0 & 0 & \gamma_\alpha \end{bmatrix} \quad (2.34)$$

The parameterization for u_{add} used in the pitch controller then is

$$\dot{q}_{add} = \Theta^T \Phi \quad (2.35)$$

Three versions of the OCM are presented. The first is a linear parameterization of the adaptive signal. This method is called Optimal Control Modification-Linear (OCM-L). This

version is considered to be linear due to the multiplication of the weights and the states. For the longitudinal case and $\Phi(x) = x$, the author chose $x = [q \ \theta \ \alpha]^T$, so

$$\Phi = [q \ \theta \ \alpha]^T \quad (2.36)$$

which are the standard longitudinal states. With this choice of Φ , there is a linear relationship between the states and the adaptive signal. For the OCM-L controller the adaptive signal is

$$\dot{q}_{add} = \Theta^T \Phi = \Theta^T [q \ \theta \ \alpha]^T \quad (2.37)$$

The second method is a hybrid of the OCM and the ABC. The simple bias element of the ABC is used with the weight update law for the OCM, with $\Phi = 1$. For this reason it is called the Optimal Control Modification-Bias (OCM-B). Instead of multiplying by the states to form the adaptive signal, the adaptation is equal to the weights. This method uses $\Phi = 1$ so that

$$\dot{q}_{add} = \Theta^T \Phi = \Theta_B \quad (2.38)$$

$$\dot{\Theta}_B = -\Gamma_B \left(\frac{\int_0^t q_e d\tau}{K_i} + q_e \left(\frac{K_i + 1}{K_p K_i} \right) + \frac{v \Theta_B}{K_i^2} \right) \quad (2.39)$$

where Γ_B is the bias adaptive gain.

The third version of the OCM is the combination of the previous two methods. Using the weight update laws of equations (2.33) and (2.39) the weights of the linear and bias terms are computed separately and then combined to form the adaptive signal

$$\dot{q}_{add} = \Theta^T \Phi + \Theta_B \quad (2.40)$$

This method is called Optimal Control Modification-Linear and Bias (OCM-LB) since the two methods are added together.

The three methods will be compared in Chapter 4.

2.6 Architecture Differences

There are slight differences in the overall control architecture depending on the adaptation method used. In Figure 2, there is no sign indicated for the signal of $\dot{q}_{add}$ in the summation block that calculates $\dot{q}_{com}$. The omission is intentional because this sign is different for the ABC and OCM adaptation methods. For the ABC, this is a plus sign, in order to be consistent with the previous development of this adaptation method [23]. Also, to be consistent with previous development, OCM adaptations include a negative sign at this point [12] [13].

Another difference between the controllers is the states passed to the adaptation from the aircraft. In Figure 2, the signal that differs is labeled as “qdot or q or phi.” For the ABC with modeling-error adaptation, this signal contains only $\dot{q}$. For the ABC with tracking-error adaptation and OCM-B, the signal contains only q . The “phi” in the signal name represents the Φ term in the adaptive signal calculation of the OCM-L and OCM-LB.

CHAPTER 3

METHOD OF ANALYSIS AND DESIGN

The controller design was analyzed and designed for an aircraft model of a Beech Bonanza. Iterative techniques were used to vary the PI controller gains and the adaptive gains of the controller to achieve the best mix of command tracking and time delay margin. The parameters that were adjusted to design each controller were the PI gains of K_p and K_i . Controllers with the ABC adaptation included the design of the adaptive gain η . The examined controllers with the OCM had four adaptive gains that could be adjusted, γ_q , γ_θ , γ_α , and ν . The gains were evaluated based on the time response tracking of a pitch doublet and error metric studies to determine robustness.

3.1 Aircraft Model

The test aircraft was a Beech Bonanza, a single engine, four-passenger, general aviation aircraft, with a standard tail. Specifically, the aircraft is a model F33C, designated as CJ-144. A three-view drawing of a Beech Bonanza is shown in Figure 6 [27]. The Bonanza is a cantilevered low-wing monoplane with a wingspan of 33 feet 5.5 inches, and a length of 25 feet 6 inches. The maximum takeoff and landing weight of the Bonanza is 3,400 pounds, with an empty weight of 2,000 pounds [28]. The aircraft model was based on the nonlinear equations of motion as given by Roskam [29] in body coordinates for longitudinal wings level flight:

$$\dot{U} + g \sin \theta + WQ = \frac{1}{m} F_{A_x} + \frac{1}{m} T \cos \phi_T + \frac{1}{m} F_{X_{\delta_e}} \delta_e \quad (3.1)$$

$$\dot{W} - g \cos \theta - UQ = \frac{1}{m} F_{A_z} - \frac{1}{m} T \sin \phi_T + \frac{1}{m} F_{Z_{\delta_e}} \delta_e \quad (3.2)$$

$$\ddot{\theta} = \frac{1}{I_{yy}} M_A - \frac{1}{I_{yy}} T d_T + \frac{1}{I_{yy}} M_{\delta_e} \delta_e \quad (3.3)$$

The force and moment terms in equations (3.1) to (3.3) are built up based on aerodynamic and thrust coefficients provided by Hawker Beechcraft Corporation (HBC). The coefficients are functions of five flight variables: angle of attack, thrust coefficient, product of thrust and angle of attack, gear retracted or deployed, and flap setting. The aerodynamic model is representative of HBC's Fly-By-Wire Testbed that has been used for testing the previous WSU adaptive control systems. See Lemon et al. [23]. This model is proprietary to HBC and was not given in this thesis.

Figure 6. Three view drawing of Beech Bonanza

3.2 Time response Tracking

To test the time response tracking of the developed controllers, a simulation in Matlab/Simulink was used. The simulation lasted 200 seconds with a commanded pitch doublet. The doublet began at 95 seconds into the simulation with a commanded pitch rate of +0.5 degrees/second for ten seconds and then -0.5 degrees/second for ten seconds, returning to a commanded q of zero at 105 seconds. This doublet was chosen to increase the pitch angle by

five degrees, which would be a common maneuver in a general aviation aircraft. The commanded pitch doublet is shown in Figure 7.

Figure 7. Time response tracking pitch rate doublet command

The commanded airspeed was held constant during the doublet at one of three airspeeds: 65 knots, 100 knots, or 165 knots. These were chosen to follow previous research at Wichita State University, by Lemon et al. [23], and represent three areas of the flight envelope of the Bonanza. The low-speed case of 65 knots was just above the stall speed without flaps of 63 KIAS [30]. The mid-speed case of 100 knots was a nominal flight condition and the flight speed at which the inverse controller was set. The high-speed case of 165 knots was just below the upper limit of the normal operating range at 167 KIAS [30]. Changes in commanded airspeed were not tested, since the velocity loop was not changed from the previously researched controller of Lemon et al. [23].

Each iteration of a controller in the design phase was evaluated on how closely the pitch rate would follow the modeled commanded pitch rate, q_m . For the pitch rate command doublet shown in Figure 7, the model pitch rate is shown in Figure 8 for the pitch rate and the pitch

angle. This q_m is a first-order model of the commanded pitch rate generated by the model follower detailed in Section 2.2.

The rise time chosen to calculate ω_n was one second. This rise time was chosen in order to have the same fast response that a pilot would expect from commanding the pitch rate if s/he were directly controlling the elevator. The use of this rise time gave a $\omega_n = 2.2$, which falls into the range $0.8 \leq \omega_{n_{sp}} \leq 5.5$, the short period natural frequency required for Level I handling qualities given in the military specifications MIL-F-8785 C. The specifications were determined from information presented by Roskam [29] for a Class I aircraft under Category B flight phase, with $n/\alpha = 8.5$ for the Beech Bonanza at 100 knots.

Figure 8. Time response tracking pitch rate doublet model

3.3 Error Metric Study

A measure of the robustness of a controller, given certain design parameters, can be determined using the method developed at NASA Ames for error metric studies [24] [25]. This method involves artificially delaying the elevator control signal between the inverse controller

and the aircraft model. The amount of delay was varied from 0 to 1.0 seconds in 0.02 second increments. The metric is the integration of the second norm of the tracking-error to the pitch doublet of Figure 8:

$$M = \frac{\|q_m - q\|_2}{\|q_m\|_2} \quad (3.1)$$

The error metric is calculated from the beginning of the maneuver at 95 seconds until the end of the simulation at 200 seconds so that

$$M = \int_{95}^{200} \frac{\sqrt{(q_m - q)^2}}{\sqrt{q_m^2}} dt \quad (3.2)$$

An example of an error metric study is shown in Figure 9. The abscissa shows the amount of time delay, and the ordinate shows the error metric, M . The metric study shown is for a given set of gains examined at each of the test airspeeds. The metric study can also be used to evaluate different gains at a single speed by a similar analysis. It can be seen that the value of the error metric increases with increasing delay, as is expected.

Figure 9. Example of error metric study

Of note is the value of the delay, where the system becomes either unstable or reaches a certain large amount of error. The point where this occurs is an estimate of how much delay the controller can tolerate and is a measure of robustness. The metric study shown in Figure 9 indicates that the low-speed case of 65 knots can only handle about 0.08 seconds of delay before it becomes unstable. This point is called the time delay margin and is measured in seconds. In the 100 knot case, the maximum delay is more difficult to assign and is taken to be where there is a significant increase in the error, in this case around 0.74 seconds. The high-speed case never has a significant turn within the one second of delay tested.

With all tested controllers, the 65 knot case represents the critical case with respect to the time delay margin.

3.4 Gain Tuning Process

The gain tuning process is described for the OCM controller gain selection, a similar process that was used for the controllers with ABC adaptation. The PI controller gains were selected first, followed by the adaptation gains. The first gain chosen was K_p . The other gains, K_i , γ_q , γ_θ , γ_α , v , and Γ_B , were held at constant nominal values. Then K_p was varied systematically to determine the gain values for the best tracking performance to the pitch rate doublet. These were examined at the three test speeds. Once three values of K_p that resulted in good tracking were found, an error metric study was completed to see which would produce the greatest time delay margin. The gain with both good tracking and time delay margin at all three airspeeds was selected. The proportional gains tested were in the range of one to twenty-five.

With the proportional gain fixed at the value found by the above procedure, the PI integral gain, K_i , was selected. The other gains were kept at the same nominal values as before. In a similar way as described above, K_i was varied to determine a range of gains that produced a

good time response tracking in pitch rate to the commanded doublet. After the K_i gain value was narrowed down to three, an error metric study was completed to determine which of the gains had the best time delay margin. The integral gain was varied in the range of zero to fifty.

With the PI controller gains selected, the adaptive gains of γ_q , γ_θ , γ_α , and v were each, in turn, varied. A range of values was determined so that the system would not be unstable and the tracking performance would not be reduced. These three gains, the components of Γ , were selected based on the OCM-Linear controller. Error metric studies were completed for sets of possible values for each gain. The γ adaptive gains were varied in the range of zero to 100,000, with v varied from zero to 0.5.

CHAPTER 4

RESULTS

The results presented are the time response tracking and error metric studies of the final design gains for each controller. A description of the gain selection process is given in Section 3.4.

The time response of each controller to the pitch rate doublet described in Section 3.2 will be shown for the states of pitch rate, pitch angle, angle of attack, and airspeed and controls of elevator deflection and thrust. A separate time response is shown for each test airspeed: 65 knots, 100 knots, and 165 knots. In the figures that follow, the black line in the pitch rate plot is the pitch rate model to the doublet command, and the black line in the airspeed plot is the commanded airspeed.

An error metric study for the three airspeeds will be shown for each controller. From this information, time delay margins were determined. In all figures showing error metrics results, the blue line with circles represents the low-speed case, the green line with diamonds represents the mid-speed case, and the red line with squares represents the high-speed case.

This chapter has three sections. The first section presents results of the two controllers using the ABC adaptation methods. The first of these controllers was adapted on the modeling-error, and the second was adapted on the tracking-error. Each of these is presented separately, and then the time responses are compared to each other. The following section presents the three controllers with the OCM adaptation, first separately and then with direct time response comparisons among the three. The three OCM variations are the OCM-Linear, OCM-Bias, and OCM-Linear and Bias. Lastly, the time responses of controllers with ABC adaptations and the OCM adaptations are directly compared.

4.1 Adaptive Bias Corrector Controllers

Two controllers using the ABC adaption method were examined: one adapting on the modeling-error, e , and the other adapting on the tracking-error, q_e . The modeling-error controller is presented first. Table 1 gives the final gains for the modeling-error controller.

Table 1

ABC Controller with Modeling-Error Adaptation: Final Gains

K_p	1.06
K_i	0.125
η	0.15

The time response to the pitch rate doublet for this controller is shown in Figure 10 for the low-speed, 65 knot case; in Figure 11 for the mid-speed, 100 knot case; and in Figure 12 for the high-speed, 165 knot case.

This ABC controller with modeling-error did not provide good tracking of the model pitch rate. There was significant overshoot in the pitch rate for all three of the tested airspeeds. In the high-speed case, the aircraft did not track the airspeed command due to saturation of thrust during the pitch-up maneuver at this high-speed. This was seen in all of the tested controllers. The error metric for the three speeds is shown in Figure 13, which indicates relatively high error throughout the delays, but the mid-speed and high-speed never have a significant jump in error, so they are tolerant to delay but do not track the command signal well. The time delay margin for the 65 knot case is 0.12 seconds.

Figure 10. Time response for ABC with modeling-error adaptation at 65 knots

Figure 11. Time response for ABC with modeling-error adaptation at 100 knots

Figure 12. Time response for ABC with modeling-error adaptation at 165 knots

Figure 13. Error metric study for ABC with modeling-error adaptation

The second ABC controller adapted on the tracking-error, q_e . Table 2 gives the final gains for the tracking-error ABC controller.

Table 2

ABC Controller with Tracking-Error Adaptation: Final Gains

K_p	6
K_i	16
η	0.15

The time responses for the above gains in the controller at the three tested airspeeds are shown in Figures 14 to 16. All three speeds track the pitch rate doublet with an initial overshoot and varying degrees of steady-state error. The least steady-state error occurs in the 100 knot case. The low-speed case has slight oscillations. Error metrics for the three speed cases with the ABC controller with tracking-error adaptation are given in Figure 17. This controller is less tolerant to delay compared to the ABC with modeling-error adaptation, but the initial error metric values

are significantly lower. The low-speed has the least time delay margin at 0.08 seconds, increasing to 0.26 seconds at 100 knots, and 0.4 seconds at 165 knots.

Figure 14. Time response for ABC with tracking-error adaptation at 65 knots

Figure 15. Time response for ABC with tracking-error adaptation at 100 knots

Figure 16. Time response for ABC with tracking-error adaptation at 165 knots

Figure 17. Error metric study for ABC with tracking-error adaptation

Of the two controllers using the ABC adaptation, the one that adapted on the tracking-error is the better controller. It had better pitch rate tracking with similar airspeed tracking. This is especially evident in Figures 18 to 20, where a comparison of the two ABC adaptation methods are presented for adapting on the modeling-error and tracking-error, shown in blue and green, respectively. From the time delay margins given, the ABC controller that adapted on tracking-error was less tolerant to delay at all speeds but had significantly better pitch rate tracking.

Figure 18. ABC comparison at 65 knots

Figure 19. ABC comparison at 100 knots

Figure 20. ABC comparison at 165 knots

4.2 Optimal Control Modification Controllers

Three implementations of the optimal control modification are presented here. All three versions used the same PI gains of K_i and K_p , and adapted on the tracking-error. Depending on the method of OCM, some or all of the adaptive gains, γ_q , γ_θ , γ_α , v , or Γ_B , were used, as

described in Section 2.5.2. The adaptive gains of γ_q , γ_θ , and γ_α form the diagonal of Γ , the adaptive gain matrix. These final gains are presented in Table 3.

Table 3
Optimal Control Modification Controllers: Final Gains

K_p	8
K_i	8
γ_q	1000
γ_θ	1000
λ_α	1000
ν	0.3
Γ_B	20

These gains for the PI controller vary from those recommended by Nguyen et al. [11] [12], since the final gains were found to provide better tracking of the command signal. Based on the parameters selected for these controllers, the recommended PI controller gains would be $K_p = 3.11$ and $K_i = 4.84$.

The time responses of the OCM-Linear controller are shown in Figures 21 to 23. The OCM-L used the linear adaptive element only, as described in Chapter 2, so that Γ had the values given in Table 1, and Γ_B was effectively zero. The 65 knot case shown in Figure 21 indicates good tracking, with some overshoot and oscillation in the pitch rate response. The mid-speed case, shown in Figure 22, has some overshoot in the pitch rate tracking and good airspeed tracking. The final case for this controller is shown in Figure 23 for 165 knots, which indicates undershoot in the pitch rate response. It can be seen that the controller is not able to maintain the 165 knot commanded airspeed. This can be seen in all high-speed cases and is due to saturation of thrust for the pitch-up maneuver at this high-speed. The error metric of the OCM-L is shown

in Figure 24. The time delay margins for the OCM-L are 0.08 seconds at 65 knots, 0.74 seconds at 100 knots, and 1 second at 165 knots.

Figure 21. Time response for OCM-L at 65 knots

Figure 22. Time response for OCM-L at 100 knots

Figure 23. Time response for OCM-L at 165 knots

Figure 24. Error metric study for OCM-L

The second variation on the OCM adaption was the OCM-Bias, which included the bias element only with the OCM weight update rule. This is similar to the ABC adaptation, but using the OCM weight update rule. In order to compare the two methods, an effective learning rate from the OCM-B was calculated. The middle term, the term for the tracking-error, was taken from equation (2.34) and multiplied by the simulation rate to become

$$\eta_{ocm} = \Gamma_B \cdot \left(\frac{K_i + 1}{K_p K_i} \right) \cdot \Delta t \quad (4.1)$$

Substituting the values into equation (4.1), $\eta_{ocm} = 0.056$. This is a little more than a third of the ABC learning rate of 0.15. The OCM also included terms for the integration of the error and the weights in the weight update law.

The time responses, shown in Figures 25 to 27, show that this controller gave an overshoot in the pitch rate for each airspeed case. The 65 and 165 knot cases included some oscillations, with the low-speed at higher frequency. The error metric study for the three commanded airspeeds of the OCM-B is shown in Figure 28. The time delay margin at low-speed

is 0.24 seconds, at mid-speed 0.48 seconds, and at high-speed 0.34 seconds. The lowering of the mid- and high-speed time delay margins can also be seen in the ABC controller with tracking-error. Both of these controllers were based on tracking-error and only included a bias term. They show the trend of squeezing the time delay margins together.

Figure 25. Time response for OCM-B at 65 knots

Figure 26. Time response for OCM-B at 100 knots

Figure 27. Time response for OCM-B at 165 knots

Figure 28. Error metric study for OCM-B

The final variant of the controllers that use the optimal control modification is the combination of the two previous methods, OCM-Linear and Bias. The bias and linear elements were computed and summed. The time responses in the pitch rate, as shown in Figures 29 to 31, had the best tracking of the three OCM variants. The amount of overshoot and error was less than the other two OCM methods. The error metric for the OCM-LB is shown in Figure 32. The low-speed case had a short time delay margin at 0.06 seconds, increasing to 0.34 seconds at 100 knots, and then to 0.66 seconds at 165 knots.

Figure 29. Time response for OCM-LB at 65 knots

Figure 30. Time response for OCM-LB at 100 knots

Figure 31. Time response for OCM-LB at 165 knots

Figure 32. Error metric study for OCM-LB

To better see how the three variations of the OCM compare with each other, all are plotted on the same axes for each of the tested airspeeds in Figures 33 to 35. The three OCMs have similar time responses. The OCM-L is shown in blue, the OCM-B is shown in green, and the OCM-LB is shown in red.

Figure 33. OCM comparison at 65 knots

Figure 34. OCM comparison at 100 knots

Figure 35. OCM comparison at 165 knots

4.3 Comparisons of ABC and OCM Controllers

To evaluate the differences in the two methods of adaptation, time responses are given for the ABC controller with tracking-error, OCM-L and OCM-LB, representing the better controllers of each adaptation method. These are shown in Figures 36 to 38 for the test airspeeds.

There is less steady-state error when using the OCM controller, and generally more overshoot than the ABC with tracking-error adaptation.

Figure 36. ABC and OCM comparison at 65 knots

Figure 37. ABC and OCM comparison at 100 knots

Figure 38. ABC and OCM comparison at 165 knots

CHAPTER 5

CONCLUSION

This thesis presented five model reference pitch rate commanded longitudinal flight controllers using different adaptive techniques. All of the controllers had the same basic architecture, beginning with a pitch rate feedback PI controller in the pitch loop, a dynamic inverse controller that received desired pitch and velocity accelerations, and an adaptive element. These combined to calculate the necessary elevator and throttle commands. These commands were the inputs to a nonlinear aircraft simulation of a Beech Bonanza.

Each design involved the tuning of the PI controller gains, K_p and K_i . Then the adaptive gains were tuned to produce the best performance. The adaptive gain in the ABC controllers was η . The OCM controllers had adaptive gains of Γ , Γ_B , and v .

Each controller with the final gains was presented with the time response tracking to a commanded pitch rate doublet. Time responses for pitch rate, pitch angle, angle of attack, airspeed, thrust, and elevator deflection were given for each of three test speeds. The time delay margin for each controller at each airspeed was calculated and presented using a 2-norm error metric. The following are the conclusions for the two adaptation methods based on the results in Chapter 4.

5.1 Adaptive Bias Corrector Adaptation

Two controllers with ABC adaptation were presented. The controller that adapted on the modeling-error, e , was not able to track the commanded pitch rate. This lack of tracking was primarily due to the inherent delay introduced by the elevator model.

Using the ABC to adapt on the tracking-error, q_e , was shown to have adequate tracking of the commanded pitch rate and sufficient time delay margins. The adaptive bias corrector

adaptation method was computationally simple and easy to program. The tracking-error ABC is a good option for further testing and expansion.

5.2 Optimal Control Modification Adaptation

The OCM was modified for use on a general aviation aircraft. Three versions of this adaptation were tested. Each showed good tracking of the pitch rate command. The amount of time delay with which each of the controllers could still perform varied. The OCM-LB showed the best tracking performance but had the worst time delay performance at low-speed. The OCM-L had slightly poorer tracking than the OCM-LB but a better time delay margin at 65 knots. The OCM-B had the worst tracking performance but was still relatively good, with the best time delay margin at low-speed. All three controllers provide to be sufficient for further study and expansion, preferably OCM-Linear.

3.3 Future Work and Recommendations

The work presented in this thesis was for the longitudinal case only. It is recommended that the best methods, the ABC with tracking-error, and all three OCM methods be expanded to a full six degrees of freedom simulation. The six DOF controllers would add the commands of roll rate, p , and yaw rate, r , to fit with the rate controller architecture.

It is also recommended that the controllers be adapted for use in a piloted simulation environment by adding a mechanism to interface with the flight simulator X-plane. This would provide more feedback about the best version of adaption of the OCM and how it compares in flight simulation with the ABC method.

After the piloted simulation has been satisfactorily completed, running the best controller on CJ-144, HBC's fly-by-wire test, would validate the design of the controllers.

REFERENCES

REFERENCES

- [1] Campbell, S., Kaneshige, J., Nguyen, N., and Krishnakumar, K., "Implementation and Evaluation of Multiple Adaptive Control Technologies for a General Transport Aircraft Simulation," *AIAA Infotech@Aerospace 2010*, Atlanta, Georgia, 20–22 April, 2010, AIAA 2010-3322.
- [2] Narendra, K., and Annaswamy, A., "A New Adaptive Law for Robust Adaptation without Persistent Excitation," *IEEE Transactions on Automatic Control*, Vol. AC-32, No. 2, February 1987, pp. 134–145.
- [3] Lewis, F., Yeşildirek, A., and Liu, K., "Multilayer Neural-Net Robot Controller with Guranteed Tracking Performance," *IEEE Transactions on Neural Networks*, Vol. 7, No. 2, March 1996, pp. 388–399.
- [4] Calise, A., and Rysdyk, R., "Adaptive Model Inversion Flight Control for Tiltrotor Aircraft," *AIAA Guidance, Navigation, and Control Conference*, New Orleans, Louisiana, 11–13 August, 1997, AIAA 1997-3758.
- [5] Rysdyk, R., and Calise, A., "Fault Tolerant Control via Adaptive Neural Network Augmentation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Boston, Massachusetts, 10–12 August, 1998, AIAA 1998-4483.
- [6] Volyanskyy, K., Calise, A., Yang B.-J., and Lavretsky, E., "An Error Minimization Method in Adaptive Control," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, 21–24 August, 2006, AIAA 2006-6343.
- [7] Kutay, A., Chowdhary, G., Calise, A., and Johnson, E., "A Comparison of Two Novel Direct Adaptive Control Methods under Actuator Failure Accommodation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Honolulu, Hawaii, 18–21 August, 2008, AIAA 2008-7286.
- [8] Yucelen, T., Calise, A., Haddad, W., and Volyanskyy, K., "A Comparison of a New Neuroadaptive Controller Architecture with the σ - and e- Modification Architectures," *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, 10–13 August, 2009, AIAA 2009-6060.
- [9] Calise, A., Yucelen, T., and Muse, J., "A Loop Recovery Method for Adaptive Control," *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, 10–13 August, 2009, AIAA 2009-5967.
- [10] Nguyen, N., Krishnakumar, K., Kaneshige, J., and Nespeca, P.I., "Flight Dynamics and Hybrid Adaptive Control of Damaged Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, May-June 2008, pp. 751–764.

REFERENCES (continued)

- [11] Nguyen, N., Krishnakumar, K., Kaneshige, J., and Nespeca, P., “Dynamics and Adaptive Control for Stability Recovery of Damaged Asymmetric Aircraft,” AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado, 21–24 August, 2006, AIAA 2006-6049.
- [12] Nguyen, N., Krishnakumar, K., and Boskovic, J., “An Optimal Control Modification to Model-Reference Adaptive Control for Fast Adaptation,” AIAA Guidance, Navigation, and Control Conference and Exhibit, Honolulu, Hawaii, 18–21 August, 2008, AIAA 2008-7283.
- [13] Nguyen, N., “Asymptotic Linearity of Optimal Control Modification Adaptive Law with Analytical Stability Margins,” AIAA Infotech@Aerospace 2010, Atlanta, Georgia, 20–22 April, 2010, AIAA 2010-3301.
- [14] Duerksen, N., “Fuzzy Logic Adaptive Decoupled Flight Controls for General Aviation Airplanes,” Ph.D. Dissertation, Department of Aerospace Engineering, Wichita State University, Wichita, KS, 1996.
- [15] Beringer, D., “Applying Performance-Controlled Systems, Fuzzy Logic, and Fly-By-Wire Controls to General Aviation,” FAA Final Report, Civil Aerospace Medical Institute, Oklahoma City, Oklahoma, May 2002, DOT/FAA/AM-02/7.
- [16] Steck, J., Rokhsaz, K., and Shue, S.-P., “Linear and Neural Network Feedback for Flight Control Decoupling,” *IEEE Control Systems Magazine*, Vol. 16, No. 4, August 1996, pp. 22–30.
- [17] Pesonen, U., Steck, J., Rokhsaz, K., Bruner, S., and Duerksen, N., “Adaptive Neural Network Inverse Controller for General Aviation Safety,” 41st Aerospace Sciences Meeting and Exhibit, Reno, Nevada, 6–9 January, 2003, AIAA 2003-578.
- [18] Pesonen, U., Steck, J., Rokhsaz, K., Bruner, S., and Duerksen, N., “Adaptive Neural Network Inverse Controller for General Aviation Safety,” *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 3, May-June 2004, pp. 434-443.
- [19] Steck, J., Rokhsaz, K., Pesonen, U., Mochrie, S., and Maxfield, M., “Pilot Evaluation of an Adaptive Controller on a General Aviation SATS Testbed Aircraft,” AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence Rhode Island, 16–19 August 2004, AIAA 2004-5239.
- [20] Steck, J., Rokhsaz, K., Pesonen, U., and Duerksen, N., “An Advanced Flight Control System for General Aviation Application,” General Aviation Technology Conference and Exhibition, Wichita, Kansas, April 2004, SAE 2004-01-1807.

REFERENCES (continued)

- [21] Steck, J., Rokhsaz, K., Pesonen, U.J., Singh, B., and Chandramohan, R., “Effect of Turbulance on an Adaptive Dynamic Inverse Flight Controller,” Infotech@Aerospace, Arlington, Virginia, 26–29 September, 2005, AIAA 2005-7038.
- [22] Steck, J., Rokhsaz, K., Namuduri, K., and Bruner, S., “Exploring Critical Flight Conditions, Controller Modes, and Parameter Estimation for Adaptive Flight Controls in General Aviation Aircraft,” FAA Final Report, Wichita State University, Wichita, Kansas, May 2006, DOT/FAA/AR-01-C-AW-WISU-50.
- [23] Lemon, K., Steck, J., Hinson, B., Nguyen, N., and Kimball, D., “Model Reference Adaptive Flight Control Adapted for General Aviation: Controller Gain Simulation and Preliminary Flight Testing on a Bonanza Fly-By-Wire Testbed,” AIAA Guidance, Navigation, and Control Conference, Toronto, Ontario, Canada, 2–5 August, 2010, AIAA 2010-8278.
- [24] Nguyen, N., Ishihara, A., Krishnakumar, K., and Bakhtiari-Nejad, M., “Bounded Linear Stability Analysis—A Time Delay Margin Estimation Approach for Adaptive Control,” AIAA Guidance, Navigation, and Control Conference, Chicago, Illinois, 10–13 August, 2009, AIAA 2009-5968.
- [25] Stepanyan, V., Krishnakumar, K., Nguyen, N., and Van Eykeren, L., “Stability and Performance Metrics for Adaptive Control”, AIAA Guidance, Navigation, and Control Conference, Chicago, Illinois, 10–13 August, 2009, AIAA 2009-5965.
- [26] NASA Ames Research Center, “Pilot Evaluations of Adaptive Control in a Motion Based Flight Simulator-Poster,” *NASA Aviation Safety Program Technical Conference*, McLean, Virginia, 17–19 November 2009.
- [27] Jackson, P., ed., “Beech Bonanza A36,” *Jane’s All the World’s Aircraft 2004–2005*, Coulsdon, Surrey, UK, 2004, pp. 565–566.
- [28] Taylor, J. W. R., ed., “Beech Bonanza A36 B,” *Jane’s All the World’s Aircraft 1974–1975*, Macdonald and Janes, London, 1974, p. 255.
- [29] Roskam, J., *Airplane Flight Dynamics and Automatic Controls Part I*, DAR Corporation, Lawrence, Kansas, 2007.
- [30] Raytheon Aircraft Company, *Beechcraft Bonanza F33A and F33C Acrobatic: Pilot’s Operating Handbook and FAA Approved Airplane Flight Manual*, 1996.

APPENDICES

APPENDIX A

SIMULATION ROUTINE FOR INVERSE CONTROLLER

```
function [sys,x0,str,ts] = inverse_linear(t,x,u,flag)
% Aircraft constants initialized in bona.m
%
% Inverse model for longitudinal flight control
% from inverse equations derived by Dr. Steck
%
% UJ Pesonen, 05/15/02
% Converted to matlab M-file J. Steck 9/2/09
% Updated drag model to match HBC model B. Hinson 10/30/09
%
% See sfuntmpl.m for a general S-function template.
%
% Dispatch the flag. The switch function controls the calls to
% S-function routines at each simulation stage of the S-function.
%
switch flag,
    %%%%%%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%%%%%
    % Initialize the states, sample times, and state ordering strings.
    case 0
        [sys,x0,str,ts]=mdlInitializeSizes;

    %%%%%%%%%%%%%%%
    % Outputs %
    %%%%%%%%%%%%%%%
    % Return the outputs of the S-function block.
    case 3
        sys=mdlOutputs(t,x,u);

    %%%%%%%%%%%%%%%
    % Unhandled flags %
    %%%%%%%%%%%%%%%
    % There are no termination tasks (flag=9) to be handled.
    % Also, there are no continuous or discrete states,
    % so flags 1,2, and 4 are not used, so return an emptyu
    % matrix
    case { 1, 2, 4, 9 }
        sys=[];

    %%%%%%%%%%%%%%%
    % Unexpected flags (error handling)%
    %%%%%%%%%%%%%%%
    % Return an error message for unhandled flag values.
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

end

%
```

APPENDIX A (continued)

```

%=====
%
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
%
function [sys,x0,str,ts] = mdlInitializeSizes()
sizes = simsizes;
sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
sizes.NumOutputs = 2; % dynamically sized with -1
sizes.NumInputs = 8; % dynamically sized with -1
sizes.DirFeedthrough = 1; % has direct feedthrough
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0 = [];
ts = [-1 0]; % inherited sample time

%=====
%
% mdlOutputs
% Return the output vector for the S-function
%=====
%
function [sys] = mdlOutputs(t,x,u)

% Inverse model for longitudinal flight control
% from inverse equations derived by Dr. Steck
%
% UJ Pesonen, 05/15/02
% Converted to matlab M-file J. Steck 9/2/09
% Updated drag model to match HBC model B. Hinson 10/30/09
% Aircraft constants initialized in bona.m
% Longitudinal
global g Sref Iyyb cbar masscontrol phiT dT
global CD0 CDK CLI CLA CL0 CM0 CMAcontrol CMQHAT CMADOTTHAT CMDE CMDETAB
CLDETAB detabtrim
%
alpha=u(1);
alphadhat=u(2);
alphadd=u(3);
qbar=u(4);
qhat=u(5);
gamma=u(6);
vdot=u(7);
gammadd=u(8);

```

APPENDIX A (continued)

```
cd = CD0+CDK*(CL0+CLDETAB*detabtrim+alpha*CLA-  
CLI)*(CL0+CLDETAB*detabtrim+alpha*CLA-CLI);  
cmde0=CM0+CMAcontrol*alpha+CMADOTTHAT*alphadhat+CMQHAT*qhat+CMDETAB*detabtrim;  
t=1/cos(alpha+phiT)*(masscontrol*vdot+masscontrol*g*sin(gamma)+qbar*Sref*cd);  
cm=Iyyb/(qbar*Sref*cbar)*(gammadd+alphadd*0)-cmde0+dT/(qbar*Sref*cbar)*t; %  
+alphadd can be removed to command thetadd */  
% output the control deflections  
out(1) = t; %thrust  
out(2) = cm/CMDE; %elevator  
sys = out;
```

APPENDIX B

SIMULATION ROUTINE FOR ABC WITH MODELING-ERROR ADAPTATION

```
function [sys,x0,str,ts] = abcq(t,x,u,flag)
% ABC with modeling error adaptation in pitch loop
%
switch flag,
    %%%%%%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%%%%%
    % Initialize the states, sample times, and state ordering strings.
    case 0
        [sys,x0,str,ts]=mdlInitializeSizes;

        %%%%%%%%%%%%%%%
        % Outputs %
        %%%%%%%%%%%%%%%
        % Return the outputs of the S-function block.
    case 3
        sys=mdlOutputs(t,x,u);

        %%%%%%%%%%%%%%%
        % Unhandled flags %
        %%%%%%%%%%%%%%%
        % There are no termination tasks (flag=9) to be handled.
        % Also, there are no continuous or discrete states,
        % so flags 1,2, and 4 are not used, so return an emptyu
        % matrix
    case { 1, 2, 4, 9 }
        sys=[];

        %%%%%%%%%%%%%%%
        % Unexpected flags (error handling)%
        %%%%%%%%%%%%%%%
        % Return an error message for unhandled flag values.
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

end

%
%=====
%
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
%
%
function [sys,x0,str,ts] = mdlInitializeSizes()
sizes = simsizes;
sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
sizes.NumOutputs = 1; % dynamically sized with -1
```


APPENDIX B (continued)

```
sizes.NumInputs      = 4; % dynamically sized with -1
sizes.DirFeedthrough = 1; % has direct feedthrough
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [-1 0]; % inherited sample time

% end mdlInitializeSizes

%
%=====
=
% mdlOutputs
% Return the output vector for the S-function
%=====
=
%
function [sys] = mdlOutputs(t,x,u)
global maxelev minelev

qdot_d=u(1);
qdothat_d=u(2);
cmd=u(3);
de=u(4);

error=cmd-qdot_d;
qdothat=qdothat_d;

% check for control saturation
if((de <= maxelev && de >= minelev) || (de > maxelev && error > 0) || (de <
minelev && error < 0))
    qdothat=qdothat_d+0.15*error; % abc
end
out=qdothat;

sys = out;
```

APPENDIX C

SIMULATION ROUTINE FOR ABC WITH TRACKING-ERROR ADAPTATION

```
function [sys,x0,str,ts] = abcq_qerr(t,x,u,flag)
% ABC with tracking error adaptation in pitch loop

switch flag,
    %%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%
    % Initialize the states, sample times, and state ordering strings.
    case 0
        [sys,x0,str,ts]=mdlInitializeSizes;

    %%%%%%%%%%%
    % Outputs %
    %%%%%%%%%%%
    % Return the outputs of the S-function block.
    case 3
        sys=mdlOutputs(t,x,u);

    %%%%%%%%%%%
    % Unhandled flags %
    %%%%%%%%%%%
    % There are no termination tasks (flag=9) to be handled.
    % Also, there are no continuous or discrete states,
    % so flags 1,2, and 4 are not used, so return an emptyu
    % matrix
    case { 1, 2, 4, 9 }
        sys=[];

    %%%%%%%%%%%
    % Unexpected flags (error handling)%
    %%%%%%%%%%%
    % Return an error message for unhandled flag values.
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

end

%
%=====
%
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
%
%
function [sys,x0,str,ts] = mdlInitializeSizes()
sizes = simsizes;
sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
```

APPENDIX C (continued)

```
sizes.NumOutputs      = 1; % dynamically sized with -1
sizes.NumInputs       = 4; % dynamically sized with -1
sizes.DirFeedthrough  = 1; % has direct feedthrough
sizes.NumSampleTimes  = 1;

sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [-1 0]; % inherited sample time

% end mdlInitializeSizes

%
%=====
%
% mdlOutputs
% Return the output vector for the S-function
%=====
%
function [sys] = mdlOutputs(t,x,u)
global maxelev minelev

q_d=u(1);
qdothat_d=u(2);
qm=u(3);
de=u(4);

error=qm-q_d;
qdothat=qdothat_d;

%check for control saturation
if((de <= maxelev && de >= minelev) || (de > maxelev && error > 0) || (de <
minelev && error < 0))
    qdothat=qdothat_d+0.15*error; % abc
end
out=qdothat;

sys = out;
```

APPENDIX D

SIMULATION ROUTINE FOR OCM ADAPTATION

```
function [sys,x0,str,ts] = ocmqlb_new2(t,x,u,flag)
% OCM adaptation
% OCM version is selected with OCMchoose
% 1 == OCM-L
% 2 == OCM-B
% 3 == OCM-LB

switch flag,
    %%%%%%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%%%%%
    % Initialize the states, sample times, and state ordering strings.
    case 0
        [sys,x0,str,ts]=mdlInitializeSizes;

        %%%%%%%%%%%%%%%
        % Outputs %
        %%%%%%%%%%%%%%%
        % Return the outputs of the S-function block.
    case 3
        sys=mdlOutputs(t,x,u);

        %%%%%%%%%%%%%%%
        % Unhandled flags %
        %%%%%%%%%%%%%%%
        % There are no termination tasks (flag=9) to be handled.
        % Also, there are no continuous or discrete states,
        % so flags 1,2, and 4 are not used, so return an emptyu
        % matrix
    case { 1, 2, 4, 9 }
        sys=[];

        %%%%%%%%%%%%%%%
        % Unexpected flags (error handling)%
        %%%%%%%%%%%%%%%
        % Return an error message for unhandled flag values.
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

end

%
%=====
%
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
%
%
```

APPENDIX D (continued)

```
function [sys,x0,str,ts] = mdlInitializeSizes()
sizes = simsizes;
sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
sizes.NumOutputs = 5; % dynamically sized with -1
sizes.NumInputs = 10; % dynamically sized with -1
sizes.DirFeedthrough = 1; % has direct feedthrough
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0 = [];
ts = [-1 0]; % inherited sample time

% end mdlInitializeSizes

%
%=====
%
% mdlOutputs
% Return the output vector for the S-function
%=====
%
function [sys] = mdlOutputs(t,x,u)
global maxelev minelev sum_error Kp_q Ki_q OCMchoose Gamma nu bias

q_d=u(1);
ltheta1_d=u(2);
ltheta2_d=u(3);
ltheta3_d=u(4);
btheta_d=u(5);
prate=u(6);
alpha=u(7);
de=u(8);
qm=u(9);
theta=u(10);

error=qm-q_d;
ltheta_d=[ltheta1_d ltheta2_d ltheta3_d]';
ltheta=ltheta_d;
btheta=btheta_d;
phi=[prate theta alpha]';
sum_error=sum_error+(error*0.02);

%check control saturation
if((de <= maxelev && de >= minelev) || (de > maxelev && error > 0) || (de <
minelev && error < 0))

    lthetadot = -
Gamma*phi*((sum_error/Ki_q)+(error*(Ki_q+1)/(Kp_q*Ki_q))+(nu*phi'*ltheta_d/Ki
_q^2));
```

APPENDIX D (continued)

```
ltheta=ltheta_d+(lthetadot*0.02);

bthetadot = -
bias*((sum_error/Ki_q)+(error*(Ki_q+1)/(Kp_q*Ki_q))+(nu*btheta_d/Ki_q^2));
btheta=btheta_d+(bthetadot*0.02);

if OCMchoose ==1                %linear only
    uadd = ltheta'*phi;
elseif OCMchoose ==2            %bias only
    uadd = btheta;
elseif OCMchoose ==3            %linear and bias
    uadd = (ltheta'*phi)+btheta;
end
end

out(1) = uadd;
out(2) = ltheta(1);
out(3) = ltheta(2);
out(4) = ltheta(3);
out(5) = btheta;

sys = out;
```

APPENDIX E

SIMULATION ROUTINE FOR VELOCITY-LOOP ADAPTATION

```
function [sys,x0,str,ts] = abcv(t,x,u,flag)
% ABC with modeling error in velocity loop
% Updated control saturation check to check PLA B. Hinson 4-16-10

switch flag,
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Initialization %
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Initialize the states, sample times, and state ordering strings.
    case 0
        [sys,x0,str,ts]=mdlInitializeSizes;

    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Outputs %
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Return the outputs of the S-function block.
    case 3
        sys=mdlOutputs(t,x,u);

    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Unhandled flags %
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % There are no termination tasks (flag=9) to be handled.
    % Also, there are no continuous or discrete states,
    % so flags 1,2, and 4 are not used, so return an emptyu
    % matrix
    case { 1, 2, 4, 9 }
        sys=[];

    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Unexpected flags (error handling)%
    %%%%%%%%%%%%%%%%%%%%%%%%%%
    % Return an error message for unhandled flag values.
    otherwise
        error(['Unhandled flag = ',num2str(flag)]);

end

%
%=====
%
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for the S-function.
%=====
%
%
function [sys,x0,str,ts] = mdlInitializeSizes()

sizes = simsizes;
```

APPENDIX E (continued)

```

sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
sizes.NumOutputs    = 1; % dynamically sized with -1
sizes.NumInputs     = 4; % dynamically sized with -1
sizes.DirFeedthrough = 1; % has direct feedthrough
sizes.NumSampleTimes = 1;

sys = simsizes(sizes);
str = [];
x0  = [];
ts  = [-1 0];

% end mdlInitializeSizes

%
%=====
%
% mdlOutputs
% Return the output vector for the S-function
%=====
%
function [sys] = mdlOutputs(t,x,u)
global maxPLA minPLA

vdot_d=u(1);
vdothat_d=u(2);
vcmd=u(3);
pla=u(4);

error=(vcmd-vdot_d);
vdothat=vdothat_d;

% check control saturation
if((pla <= maxPLA && pla > minPLA) || (pla < minPLA && error > 0) || (pla >
maxPLA && error < 0))
    vdothat=vdothat_d+0.01*error+0.0*error*abs(error);
end

out=vdothat;

sys = out;

```