On the Level of Center of Gravity Modeling Error in Neural Network Based Adaptive Controller Design

Onur TARIMCI *, Deniz YILMAZ † and Ilkay YAVRUCUK ‡

Middle East Technical University, Department of Aerospace Engineering, Ankara, Turkey

Abstract

This paper covers a parametric research study on the use of simulation models with center of gravity location errors in the design of adaptive neural network controllers. In this study a helicopter simulation model is developed along with an inner and outer loop cascade architecture flight control system, commonly found in literature. The center of gravity location is varied to severe values to observe adaptation characteristics to investigate the requirement on the knowledge of the center of gravity location during such adaptive controller design.

*Research Student, Department of Aerospace Engineering

†Research Student, Department of Aerospace Engineering

‡Faculty Member, Department of Aerospace Engineering

Nomenclature

A Total acceleration
Alt Altitude
A_1, A_2 Stability matrices
b_v, b_w Bias terms
B Control matrix
e Error vector
F Force
g Gravitational acceleration
I Inertia
$kron(a,b)$ Kronecker product function
$L_{VB}(\phi,\theta,\psi)$ Transformation matrix from body frame to navigation frame
L Roll moment in body-x axis
m Mass
M Pitch moment in body-y axis
$n1$ Number of Neural Network inputs
$n2$ Number of Neural Network neurons
$n3$ Number of Neural Network outputs
N Yaw moment in body-z axis
p Rotational body-x velocity
q Rotational body-y velocity
r Rotational body-z velocity
u Linear body-x velocity
U_1, U_2, U_3 Linear controller outputs
U_ϕ, U_θ, U_ψ Pseudo control signals
U_{NN} Neural Network output signal
v Linear body-y velocity
w Linear body-z velocity
X Position in North axis
V, W Weight matrices
Y Position in East axis
Z Position in Down axis

ϕ Roll attitude
θ Pitch attitude
ψ Yaw attitude
δ_a Lateral control deflection on trim
δ_e Longitudinal control deflection on trim
δ_p Pedal control deflection on trim
δ_c Collective control deflection on trim
γ_E e modification gain
σ Neuron activation potential function
σ' Derivative of the activation function
γ_V, γ_W Neural Network gains
$\bar{x}$ Inputs to the neural network including bias term

$\dot{()}$ Derivation w.r.t time
$()_B$ Model feedback
$()_x$ In body x-axis
$()_y$ In body y-axis
$()_z$ In body z-axis
$()_{xz}$ In body x-z plane
$()_D$ Desired values
$()_{NED}$ North-East-Down frame
$()_{trim}$ Trim values

1 Introduction

The conventional PID controller design involves the use of a linearized model of the aircraft. Commonly, a high fidelity model of the aircraft is build and linearized. However, creating a high fidelity mathematical model is a long, complex and an expensive process. Recent work focuses on the use of adaptive controllers using neural networks to reduce the requirement on the fidelity of the model. Many of the neural network based adaptive controllers however, have theoretical bounds on the adaptive error signals [3, 5].

In this study, we try to show the ability of neural network based adaptive controller by using the same controller on a helicopter models with large center of gravity location errors. The purpose is to demonstrate the effectiveness of the adaptive neural network controller on the center of gravity location uncertainty. The controller used in this study is a inner-outer loop cascade architecture with a single hidden layer neural network in the inner loop for modeling error adaptation. The network has a similar form and number of neurons commonly found in literature. It is expected that the neural network adaptation would be enough for the helicopter still to perform well under severe center of gravity location errors.

The controller architecture and the neural network placement is quite common in literature. In references [12], [13] and [14], a linear sigma-pi neural network architecture is used to handle the errors caused by linear model inversion in a controller designed for a tiltrotor application. In this study, attitudes of the tiltrotor are controlled by linear controllers augmented by an adaptive controller. A similar approach is also applied to unmanned helicopters [4, 6, 11] and single hidden layer neural network architectures are used for reusable launch systems such as the X-33 [9] and autonomous launch vehicles [8] for attitude control. All of these applications demonstrated the ability of the neural network on helicopter applications and other flying platforms. Yet no attempt is known to the authors to try how far the model uncertainty can practically be.

In this study an updated version of the well known minimum complexity mathematical model [17] is used to represent a base helicopter model [1] [2]. Variations are made to the center of gravity location of the aircraft in forward and aft positions in the longitudinal direction, while the aircraft controller was commanded pull-up and pushover maneuvers.

In this paper, first the description of the mathematical model is summarized. Then the designed controller is explained and the adaptive neural network architecture is described. This is followed with the variation of the center of gravity and the results obtained in this study. Finally whole work is summed up with the conclusion.

2 Mathematical Model

In reference [17], an example helicopter is modeled for pilot training using as simple components as possible. The idea is to include only the effects that a pilot in simulation would feel; hence the name minimum complexity. Here the main rotor flapping is calculated by a first order flapping dynamics assumption which includes coupling between lateral and longitudinal dynamics. The calculations of both induced velocities of the main rotor and the tail rotor are iterative methods based on classic momentum theory and Glauert theory [10], [15], [16], [17]. Quadratic lift coefficients are used to model the aerodynamic forces and moments on the various components of the helicopter, including horizontal tail, vertical tail, wing and fuselage. Main rotor inflow effects onto those components are also included. Finally, performance parameters such as the profile power of the main rotor, power required for climbing and parasite power are calculated separately. The model is updated using some modifications presented in references [1], [2] to obtain a higher fidelity math model. Some major modifications made to the minimum complexity model are as follows:

Tail rotor

The Tail rotor calculations of the minimum complexity simulation model consist of the same iterative method of the main rotor for the calculation of the thrust and induced velocity. Although the convergence of the iteration is fast, some maneuvers especially with high yaw rates causes the tail rotor iteration not to converge. Therefore, a more robust iterative method is added. Figure 1 is the schema of the tail rotor algorithm used. In these calculations the blockage effect of the vertical tail is also considered.

Wing, Horizontal and Vertical Tails

The minimum complexity simulation model uses a quadratic lift form based approach for the calculations of lifting surfaces. According to this approach, the stall of surfaces

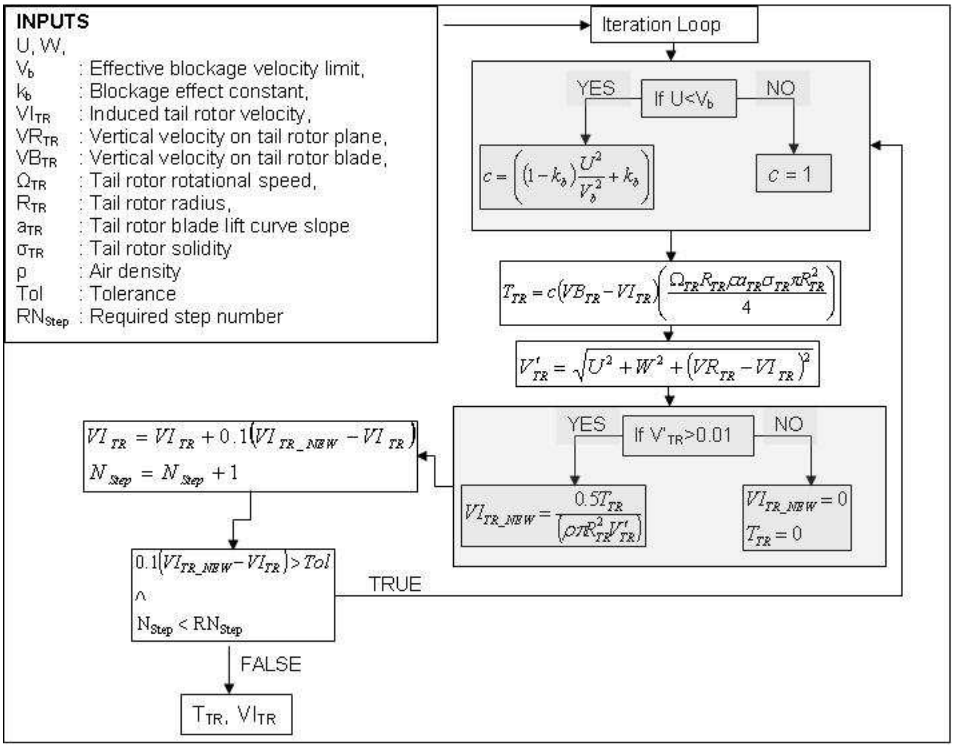


Figure 1: Tail rotor iteration schema, [1]

are checked and if stalled, a new set of constants are used for the calculations. This transition leads to a discontinuity throughout the calculations. Moreover, wing calculations of minimum complexity produces considerably high lifting force when compared to main rotor thrust values. This is thought to be unreasonable. Instead airfoil sectional lift and drag calculations are considered for varying Reynolds numbers and angle of attacks. Airfoil sections of wing, horizontal and vertical tail are assumed to be NACA0015. Airfoil test data of the horizontal tail are obtained from literature [18] for 180 degrees of angle of attack with various Reynold's number. Equations are generated from experimental data to calculate coefficients of lift and drag. Instantaneous Reynold's number and angle of attack values are used to calculate lift and drag forces.

Body Frame to Navigation Frame Transformations

The translational motion of the helicopter in the body-fixed coordinate system is modeled in the minimum complexity simulation model as reduced forms of the actual theoretical equations. Although this approach is reasonable for steady flight, effects of the maneuvers with high angular rates produce significant errors. Therefore, all the reduced terms are restored in the equations of motion to include the effects of transient dynamics during maneuvers. Similarly, the minimum complexity simulation model uses reduced order velocity transformation instead of the actual theoretical transformation matrices for the conversion from body frame to the navigation frame. The helicopter can be subjected to high Euler angles and simplified transformations tend to cause noticeable errors. Hence, full transformation matrices are used to be able to simulate the high Euler angle flight conditions.

Altitude and Thrust Coefficient Updates

The original Minimum Complexity Helicopter Model uses sea level assumptions and calculates thrust coefficients based on the weight of the helicopter. However, this assumption becomes cumbersome when the full flight envelope of the helicopter is considered. The calculations are updated to use the actual simulation altitude and instantaneous thrust coefficients are calculated.

Actuator Model

The minimum complexity model lacks an actuator model, which is an important aspect in our study. A second order lag is presented as an actuator model including angle and rate saturations of the swashplate. By examining a real helicopter, physical angle and rate limits are obtained. For this controller application, angle limits are assumed to be plus and minus deflections from the trim value in hover.

3 Controller Architecture

The flight control system consists of two loops, the inner loop and the outer loop and in the most outer layer there exists a trajectory generator. The inner loop is responsible for the control of the rotational states in the roll, pitch and yaw channels, whereas the outer loop is responsible for the control of north and east velocities along with the altitude. The command generator provides necessary inputs to the outer loop. The separation of the inner and outer loop is possible since the dynamics of the rotational states are much faster than the dynamics of translational states. Hence, the rotational states are controlled in the inner loop and the translational states in the outer loop. The block diagram of the control system developed in Simulink is depicted in Figure 2.

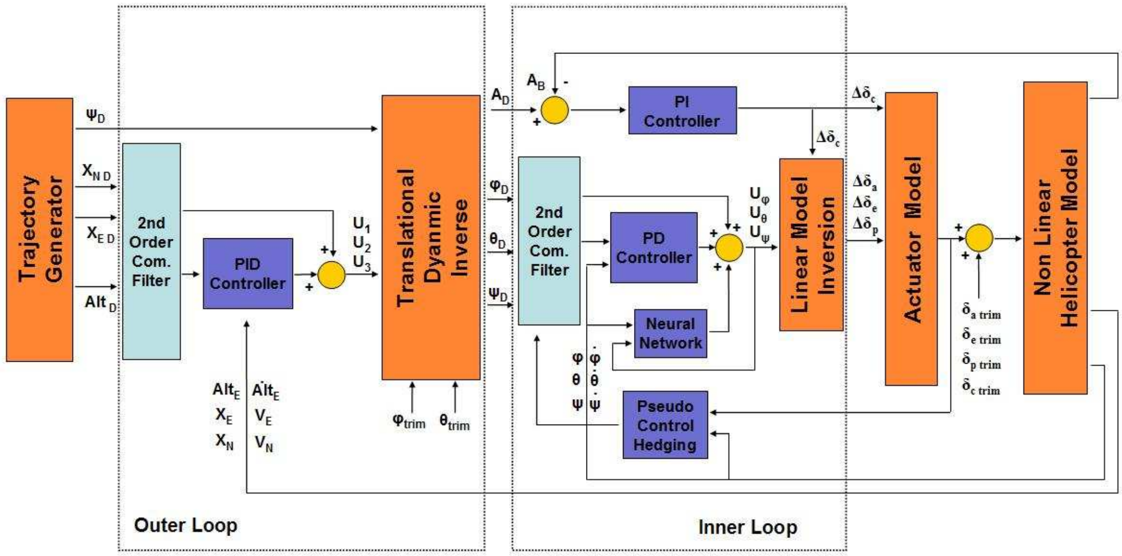


Figure 2: Block Diagram of the Control System

3.1 Inner Loop

The rotational states are controlled in the inner loop of the controller. The required commands for the inner loop are the Euler angles: ϕ, θ, and ψ. The control loop is also called the stabilization loop. The inner loop in this design generates the collective control deflections by a PI controller by using the total linear acceleration command generated in the outer loop and the linear acceleration feedback from the aircraft (Figure 2) [11]. Each attitude channel is controlled by using a PID controller. The Euler angle commands are passed through a second order command filter, a low pass filter, to match the performance of the aircraft to the commands. It represents the desired aircraft response. The command filter generates not only the command itself but also the first and second derivatives of the command and is used to obtain the derivative error. The second derivatives are used in a feed forward path as a command accelerator.

Longitudinal cyclic, lateral cyclic and pedal controls are generated through an inverted model [12] [13] [14] (Figure 2). In this work a linear model inversion is used and derived from the decoupled linear state space representation.

Decoupling is done between the fast rotational states, Euler angles, and fast translational states:

$$\left\{ \begin{array}{c} \delta_a \\ \delta_e \\ \delta_p \end{array} \right\} = B^{-1} \left(\left\{ \begin{array}{c} \dot{p_D} \\ \dot{q_D} \\ \dot{r_D} \end{array} \right\} - A_1 \left\{ \begin{array}{c} u \\ v \\ w \\ \delta_c \end{array} \right\} - A_2 \left\{ \begin{array}{c} p \\ q \\ r \end{array} \right\} \right) \tag{1}$$

Here it is assumed that the stability and control matrices are time invariant. Desired rotational accelerations are obtained by adding the PID controller outputs and command accelerators (Figure 2).

Note that the linear model inversion is only an approximation. Therefore, there will still be a modeling error. To cancel the modeling error an adaptive neural network controller is implemented to the inner loop. A single hidden layer neural network is used as an adaptive element.

3.1.1 Single Hidden Layer Neural Network

A single hidden layer network is a commonly used neural network architecture [3]. The inputs are chosen to be the errors with their derivatives in each channels, body velocities, u, v and w, pseudo controls, U_ϕ, U_θ and U_ψ and the Frobenius norm of the matrix Z, where Z is given in equation 2.

$$Z = \left\{ \begin{array}{cc} V & 0 \\ 0 & W \end{array} \right\} \tag{2}$$

where V and W are weight matrices depicted in figure 3 In this study, number of neurons are chosen to be 10. The neuron activation potential function, σ is shown in equation 3.

$$\sigma(z) = \frac{1}{1 + e^{-az}} \tag{3}$$

where z is the input to the neuron and a is the activation potential which are unique for each neuron.

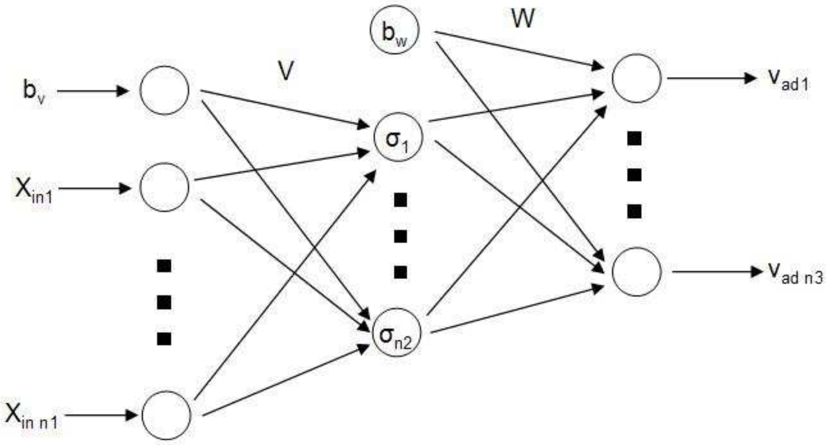


Figure 3: Block Diagram of the Single Hidden Layer Neural Network

The update law for the weight matrices (V and W) are given in equations 4 and 5. The derivations for these update laws can be found in [3].

$$\dot{W} = -\gamma_W(\sigma - \sigma^{'} V^T \bar{x}) r^T - \gamma_E ||e|| W \tag{4}$$

$$\dot{V} = -\gamma_V(\bar{x}(r^T W^T \sigma^{'}) r^T - \gamma_E ||e|| V \tag{5}$$

where r is given in equation 6.

$$r = (e^T P\, b)^T \tag{6}$$

3.1.2 Pseudo Control Hedging

When using adaptive neural networks in the controller, actuator saturation causes false learning. Pseudo control hedging is presented to deal with this problem [7]. This technique subtracts the extra input created by the controller caused by the saturation from the command to the system.

3.2 Outer Loop

In the outer loop, three positions are controlled in the north-east-down navigation frame for which it is also called the navigation loop. The outer loop receives position and heading commands to generate roll, pitch and acceleration commands to the inner loop [4]. The heading command ψ_D is passed to the inner loop. These commands are supplied to the outer loop by the trajectory generator. Similar to the one in the inner loop, a second order command filter is present in the outer loop. All channels are controlled by PID controllers. Similar to the inner loop, accelerations generated in the command filter are fed forward as command accelerators. Desired accelerations are obtained from the sum of the PID controller outputs and command accelerators (Figure 2). Desired accelerations of each channel are used to calculate the roll, pitch and total acceleration commands.

$$A_D = \sqrt{U_1^2 + U_2^2 + (U_3 - g)^2} \tag{7}$$

Roll and pitch commands are generated using the transformation between the navigation frame and body frame.

$$\begin{Bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{Bmatrix} = L_{VB}(\phi, \theta, \psi) \begin{Bmatrix} F_x/m \\ F_y/m \\ F_z/m \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \\ g \end{Bmatrix} \tag{8}$$

In equation 8, accelerations in the north-east-down navigation frame are replaced with desired accelerations in each channel U_1, U_2 and U_3:

$$\begin{Bmatrix} U_1 \\ U_2 \\ (U_3 - g) \end{Bmatrix} = L_{VB}(\phi, \theta, \psi) \begin{Bmatrix} F_x/m \\ F_y/m \\ F_z/m \end{Bmatrix} \tag{9}$$

Furthermore, by using equation 9 the desired roll and pitch angles can be calculated by the formulae in equations 10 and 11. Here, it is assumed that the cyclic and pedal control forces are much smaller compared to the collective control force. Moreover, forces in the x and y direction in the body fixed frame are assumed to be small compared to forces in z direction in body fixed frame [4].

$$\phi_D \approx \sin^{-1}\left(\frac{-U_1 \sin\psi_D + U_2 \cos\psi_D}{\sqrt{U_1^2 + U_2^2 + (U_3 - g)^2}}\right) + \phi_{trim} \tag{10}$$

$$\theta_D \approx \tan^{-1}\left(\frac{U_1 \cos\psi_D + U_2 \sin\psi_D}{(U_3 - g)}\right) + \theta_{trim} \tag{11}$$

3.3 Trajectory Generator

The trajectory generator generates the north and east position along with the altitude and the heading commands for the outer loop. It holds a library of maneuvers to be performed. Its role is to challenge the control system to carry out complex maneuvers similar to a human pilot. The trajectory generator holds a library of maneuvers such as a pull-up pushover maneuver, pirouette, slalom, etc.

In this work, particular focus has been paid to the pull-up pushover maneuver. The trajectory of the maneuver is chosen to be a sinusoid. During this maneuver the total velocity is always tangent to the trajectory. It should be noted that due to the architecture of the controller, the attitude of the helicopter is determined by the inner loop commands. Therefore, the aircraft x-body axis is not necessarily tangent to the trajectory.

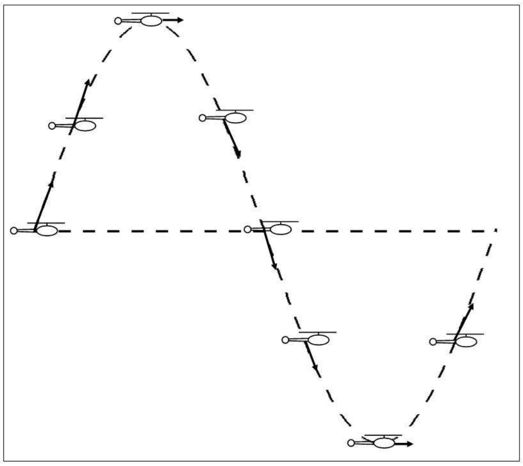


Figure 4: Pull-up Pushover Maneuver

4 Center of Gravity Variation

The center of gravity is one of the most influential parameters that effects the accuracy of a model. Therefore, classic linear controller designs are particularly sensitive to the wrong modeling of the center of gravity location of a helicopter. The center of gravity is not only hard to find, but also will change on a given helicopter based on fuel and other payload variations. Therefore, the adapting to different center of gravity configurations is important for a controller. Moreover, the model on which the controller design will be based on can be wrong. Now the question we would like to answer is how big can the error in the center of gravity location be?

Among the three axes the center of gravity location along the body x axis is the most crucial one in modeling and controller design since it is the one hardest to find and most likely to deviate. Here, the primary effect will be on the longitudinal dynamics of the helicopter. Therefore a pull-up pushover maneuver is chosen to investigate the effect. For this maneuver, the forward speed is 60 knots and the amplitude of the sinusoidal motion of the helicopter in the x-z plane is 50 ft.

The chosen baseline position of the center of gravity is 196 inch from the nose of the helicopter (stationline 196). All controller design is based on this model. Therefore it is expected that the controller will work the best for this configuration. Yet, the inverted model in the inner loop was designed for a hovering condition. Therefore even for the baseline model in forward flight it is expected that the neural network will do some work. The center of gravity location is moved forward and backwards up to 184 and 208 at the beginning. These values did not cause any control saturation for this maneuver.

It is observed that the changes in center of gravity do not affect the outer loop characteristics of this maneuver. Even without the neural network the controller is able to correct the error in the outer loop. In figures 5 and 6, the results are given for the most rearward and forward positions of the center of gravity, while avoiding any actuator saturations. The control deflections are seen in figures 7 and 8.

A strong variation in command following is observed in the inner loop errors. Here, the neural network is able to reduce the error in each channel for all center of gravity configuration, whereas without the neural network the controller is unable to overcome the error. The errors for each center of gravity configuration are shown in figures 9, 10 and 11.

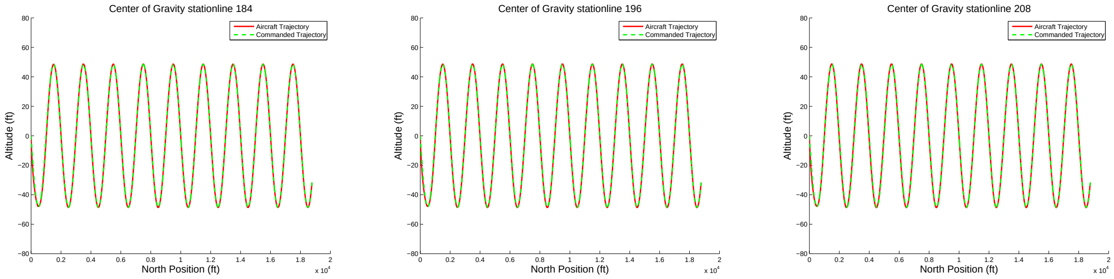


Figure 5: Trajectory of the vehicle for different center of gravity configurations without neural network

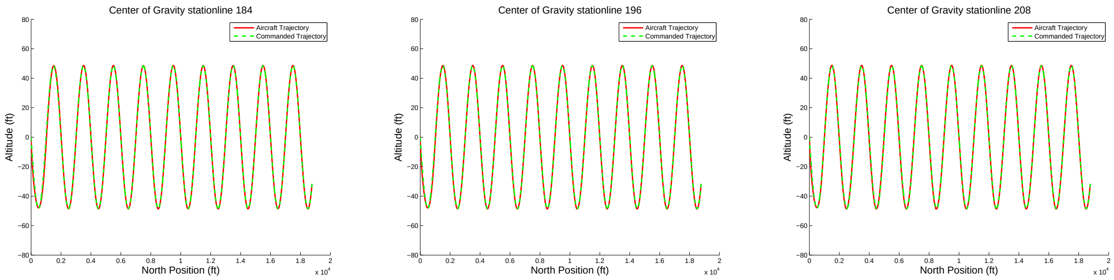


Figure 6: Trajectory of the vehicle for different center of gravity configurations with neural network

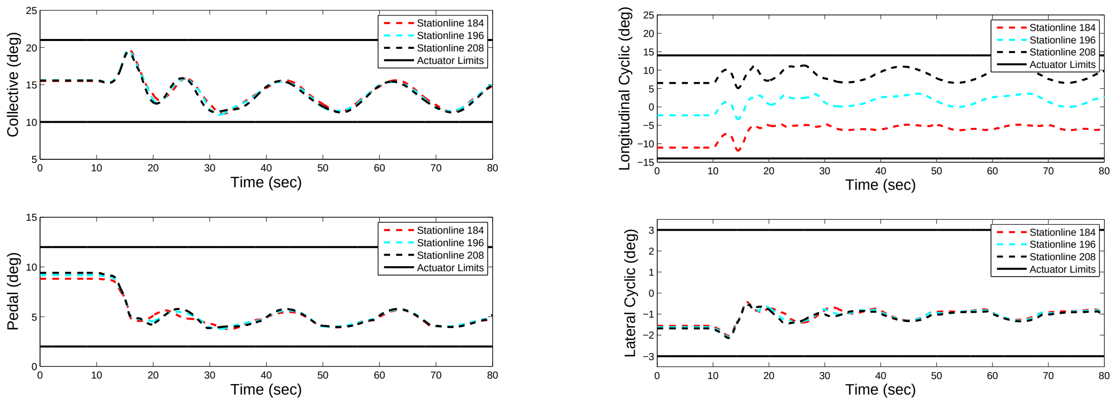


Figure 7: Control deflections of the helicopter without neural network

Note that, the network does less work when the center of gravity location is moved to

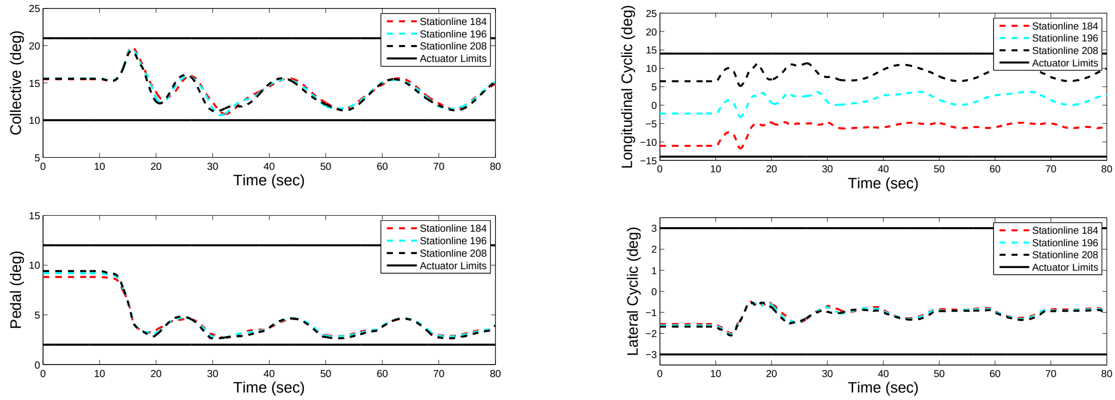
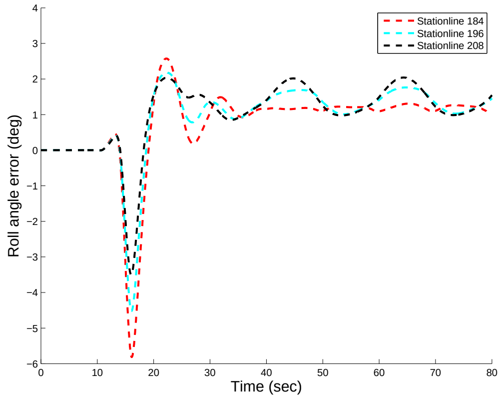


Figure 8: Control deflections of the helicopter with neural network

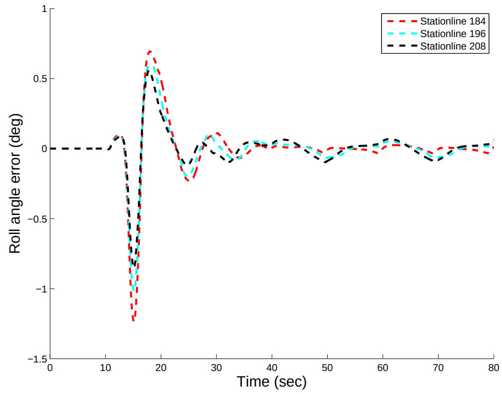
the back of the aircraft. This may be due to the fact that the inverted model - designed for one flight condition only - may be closer to the configuration with the most rearward center of gravity configuration for the given flight conditions. This may also be the reason why the errors are smallest in the transient region for the most rearward center of gravity configuration. When the helicopter reaches to a steady state, the errors are about the same.

In order to find how much center of gravity distortion the controller can handle, the center of gravity is moved backward and forward further by assuming the actuators do not saturate. Even with this severe modeling errors the controller was able to maintain the pull-up pushover maneuver with acceptable errors (figures 12, 13 and 14).

Figure 15 shows the amount of work done by the neural network in the pitch channel for different center of gravity configurations. As the center of gravity is moved forward from the original position, the amount of work done by the neural network increases (figure 15a), which means the neural network does more work to correct the error caused by the center of gravity distortion. However, when the center of gravity is moved backwards from the original position, the amount of work done by the neural network decreases (figure 15b). One reason for this result may be due to the fact that constant network gains are used for all simulations. Theoretically, the error signals are bounded for some learning rates. Therefore, different gains might cause different adaptation characteristics for different maneuvers and configurations. This can also explain why the most rearward center of gravity configuration has the largest pitch error than

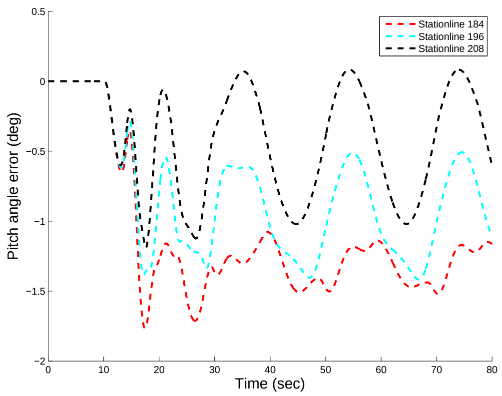


(a) Without neural network

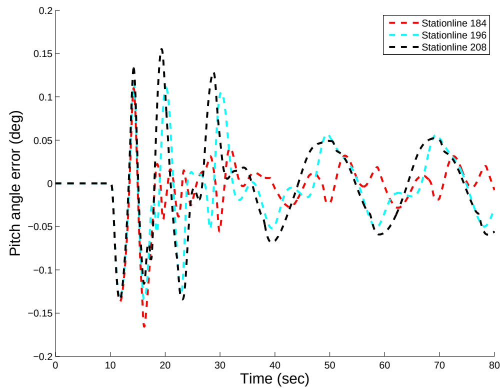


(b) With neural network

Figure 9: Variation of roll angle error with time for different center of gravity configurations

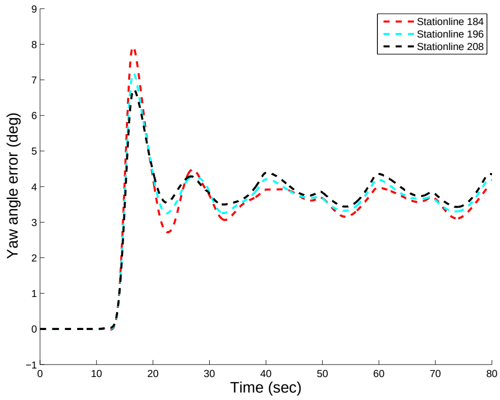


(a) Without neural network

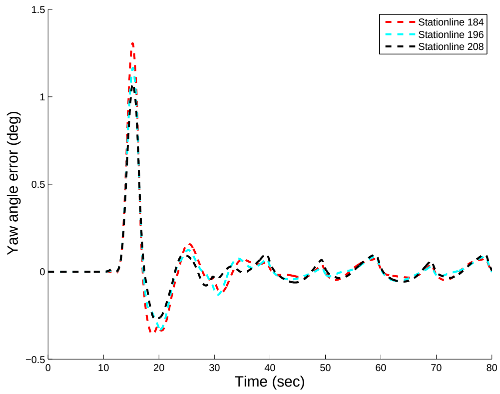


(b) With neural network

Figure 10: Variation of pitch angle error with time for different center of gravity configurations

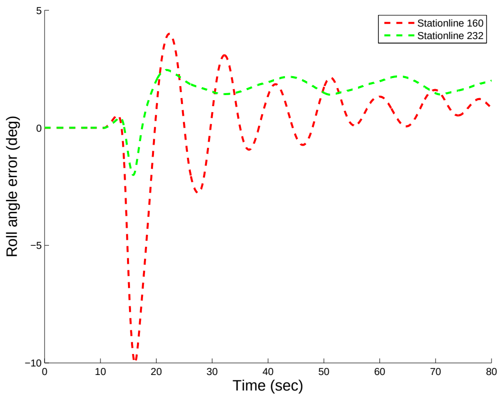


(a) Without neural network

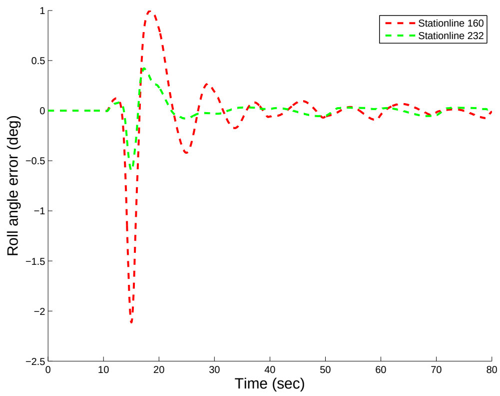


(b) With neural network

Figure 11: Variation of yaw angle error with time for different center of gravity configurations

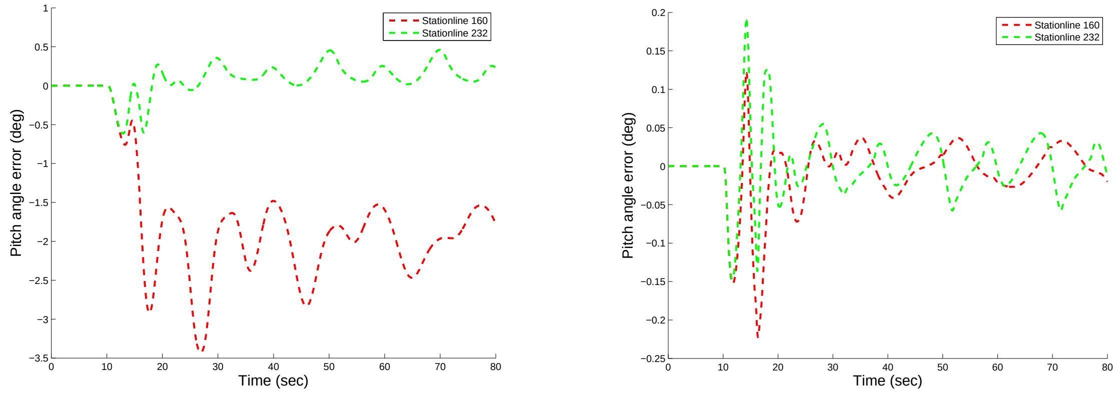


(a) Without neural network



(b) With neural network

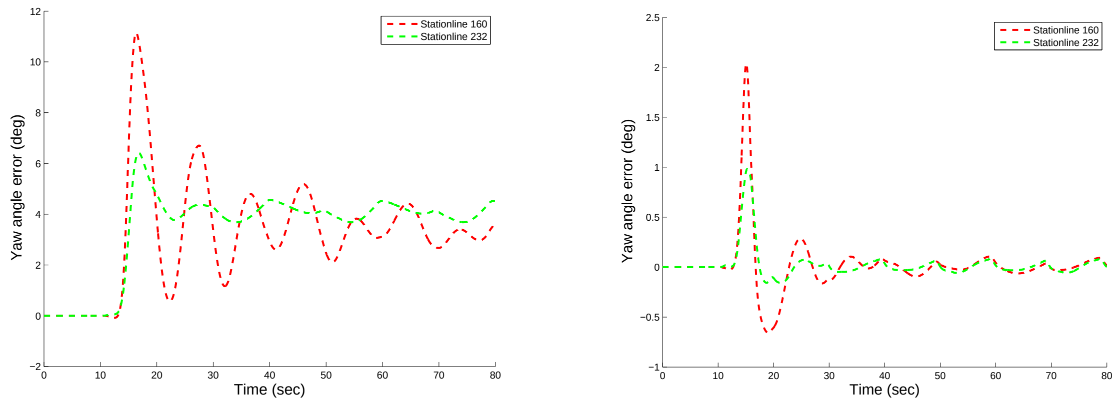
Figure 12: Variation of roll angle error with time for different center of gravity configurations



(a) Without neural network

(b) With neural network

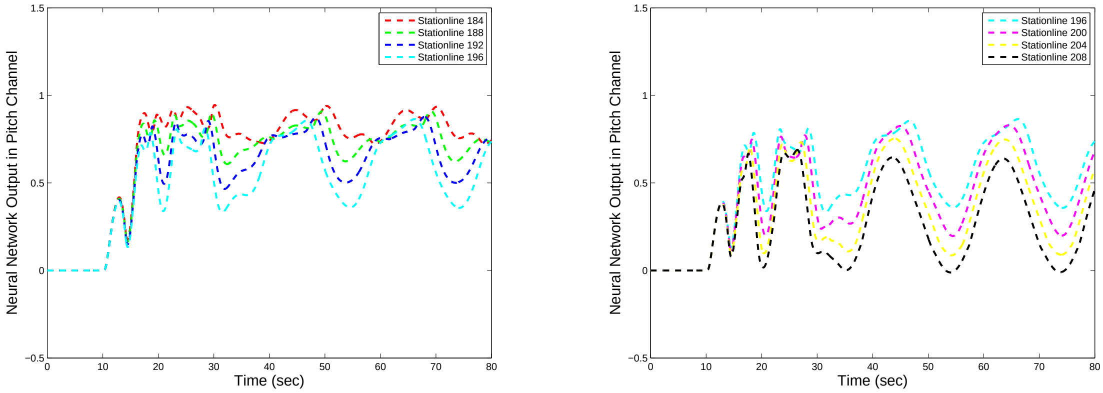
Figure 13: Variation of pitch angle error with time for different center of gravity configurations



(a) Without neural network

(b) With neural network

Figure 14: Variation of yaw angle error with time for different center of gravity configurations



(a) Forward center of gravity movement

(b) Backward center of gravity movement

Figure 15: Variation of the output of the neural network with time for different center of gravity configurations

most forward center of gravity configurations, while the neural network also does less work. In order to investigate this possibility, the neural network gains are modified in an effort to decrease the error in the pitch channel for the most forward center of gravity configuration. Figure 16 and 17 show that with a modified gain, the error in the pitch channel is decreased and the neural network does more work than with the original gains. However, even with these gain selections, the amount of work done by the neural network is less than the other configurations. Better results may be obtained when gains are further optimized. The corresponding neural network weights are shown in figure 18.

From these results, it can be said that with suitable learning rates in the presence of neural networks, the center of gravity information in the body x axis is less crucial for the modeling of the helicopter when used for controller design, as long as the actuators are not saturated. Neural network is able to handle the error caused by the incorrect center of gravity information in the body x axis direction. However a good actuator model is needed.

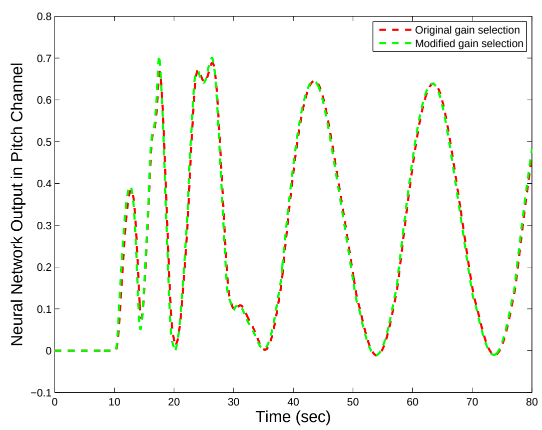


Figure 16: Variation of the output of the neural network with time for different neural network gains

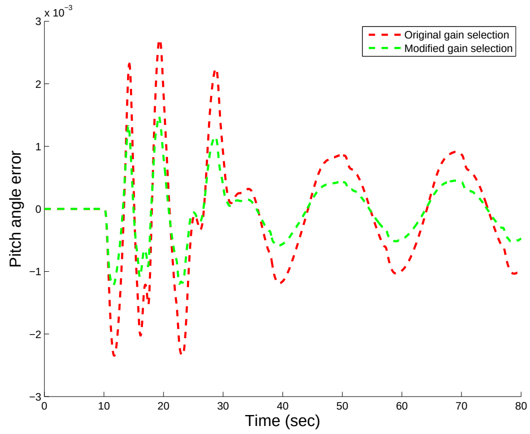


Figure 17: Variation of pitch angle error with time for different neural network gains

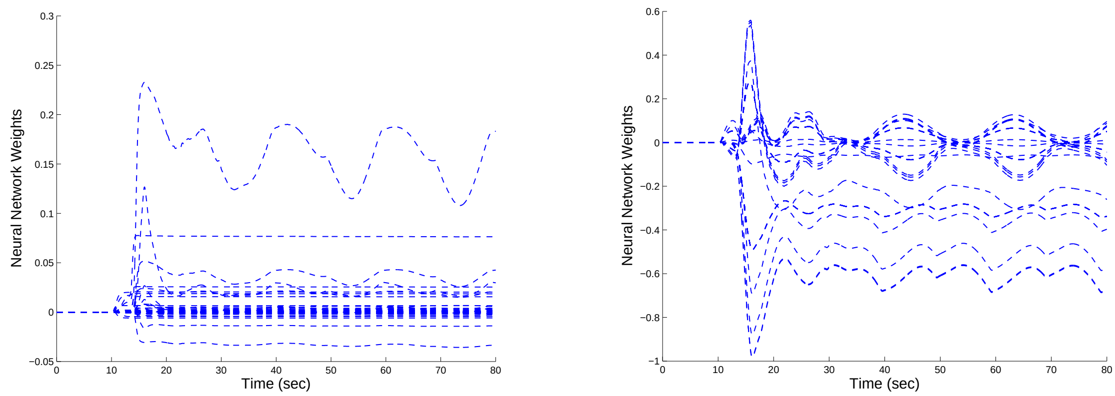


Figure 18: Variation of neural network gains with time

5 Conclusion

In this research a modified minimum complexity helicopter model is used to demonstrate the use of adaptive neural network controllers and the allowable center of gravity travel in the simulation model.

The controller is composed of an inner loop, an outer loop and a trajectory generator. In the inner loop, an adaptive element is included to carry out the research. A single hidden layer neural network architecture is used as an adaptive element. Along with the neural network, a pseudo control hedging technique is used to prevent neural network from improper learning in the presence of actuator saturations.

It is observed that the adaptive controller worked almost perfectly under any longitudinal center of gravity location as long as there was no actuator saturation. Even in presence of actuator saturation the neural network continued to learn through the use of pseudo control hedging, but was not able to follow the trajectory. Finally, the actuator saturation is artificially removed in the model to observe much severe center of gravity travel abilities. Even for center of gravity travels for which the pull-up pushover maneuver is not possible to carry out without actuator saturation, the adaptive controller was able to perform well.

References

[1] Yilmaz, D. "Evaluation and Comparison of Helicopter Simulation Models with Different Fidelities", MSc. Thesis Aerospace Engineering Department, Middle East Technical University, 2008.

[2] D. Yilmaz, I. Yavrucuk "Development of a Flight Dynamics Model for a UH-1H Helicopter Simulator", 4. Ankara International Aerospace Conference, 10-12 September, 2007 - METU, Ankara Paper No: AIAC-2007-076

[3] Johnson, E., and Kannan, S., "Adaptive Trajectory Control for Autonomous Helicopters", AIAA Journal of Guidance, Control, and Dynamics, Vol. 28, No. 3, 2005

[4] Corban, J. E., Calise, A. J., Prasad, J. V. R., Heynen, G., Koenig, B., Hur, J. "Flight Evaluation of an Adaptive Velocity Command System for Unmanned Helicopters", AIAA Guidance, Navigation and Control Conference and Exhibit, 11-14 August 2003, Austin, Texas

[5] Hovakimyan, N., Nardi, F., Kim, N., Calise, A. J. "Adaptive Output Feedback Control of Uncertain Systems using Single Hidden Layer Neural Networks", IEEE Transactions on Neural Networks, Vol. 13, No. 6, November 2002, pp. 1420-1431.

[6] Johnson, E. N., Kannan, S. K. "Adaptive Flight Control for an Autonomous Unmanned Helicopter", AIAA Guidance, Navigation, and Control Conference and Exhibit 5-8 August 2002, Monterey, California Paper No: 2002-4439

[7] Johnson, E. N., Calise, A. J., "Pseudo-Control Hedging: A New Method for Adaptive Control", Workshop on Advances in Guidance and Control Technology, Redstone Arsenal, Alabama, 2000.

[8] Johnson, E. N., Calise, A. J., Corban, J. E. "Adaptive Guidance and Control For Autonomous Launch Vehicles", Georgia Institute of Technology and Guided Systems Technologies, Inc. 2000.

[9] Calise, A. J., Johnson, E. N., Rysdyk, R., El-Shirbiny, H. A. "Feedback Linearization with Neural Network Augmentation Applied to X-33 Attitude Control", AIAA-2000-4157, AIAA Guidance, Navigation and Control Conference and Exhibit, 2000, Denver, CO.

[10] Leishman, J. G. "*Principles of Helicopter Aerodynamics*", Cambridge Aerospace Series, 2000

[11] Prasad, J. V. R., Calise, A. J., Pei, Y., Corban, J. E. "Adaptive Control Synthesis and Flight Test Evaluation on an Unmanned Helicopter", 1999 IEEE International Conference on Control Applications

[12] Calise, A. J., Rysdyk R. T. "Nonlinear Adaptive Flight Control Using Neural Networks", IEEE Controls Systems Magazine, 1998

[13] Calise, A. J., Rysdyk R. T. "Nonlinear Adaptive Control of Tiltrotor Aircraft Using Neural Networks", 1997 SAE/AIAA World Aviation Congress, 14-16 October, 1997, Anaheim, CA.

[14] Calise, A. J., Rysdyk R. T. "Adaptive Model Inversion Flight Control For Tiltrotor Aircraft", AIAA Guidance, Navigation and Control Conference, August 1997, Paper No: 97-3758

[15] Padfield, G. D. "*Helicopter Flight Dynamics: The Theory and Application of Flying Qualities and Simulation Modeling*", AIAA Education Series, 1996

[16] Prouty, R. W. "*Helicopter Performance, Stability, and Control*", Krieger Publishing Company, 1995

[17] Hefley, R. K., Mnich, M. A. "Minimum-Complexity Helicopter Simulation Math Model", NASA Contractor Report 177476 USAAVSCOM Technical Report 87-A-7 1988

[18] Sheldahl, R. E. , Klimas, P. C. "*Characteristics of Seven Airfoil Sections Through 180 Degrees Angle of Attack for Use in Aerodynamic Analysis of Vertical Axis Wind Turbines*", SAND80-2114,Sandia National Laboratories, Albuquerque, New Mexico, March 1981

Appendices

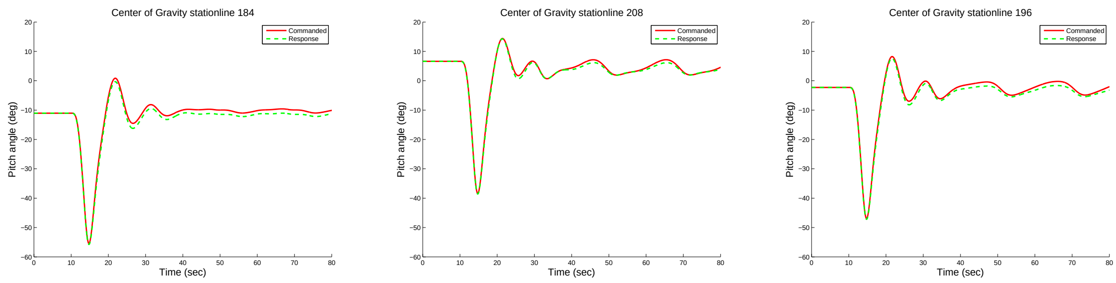


Figure 19: Variation of pitch angle with time for different center of gravity configurations without neural network

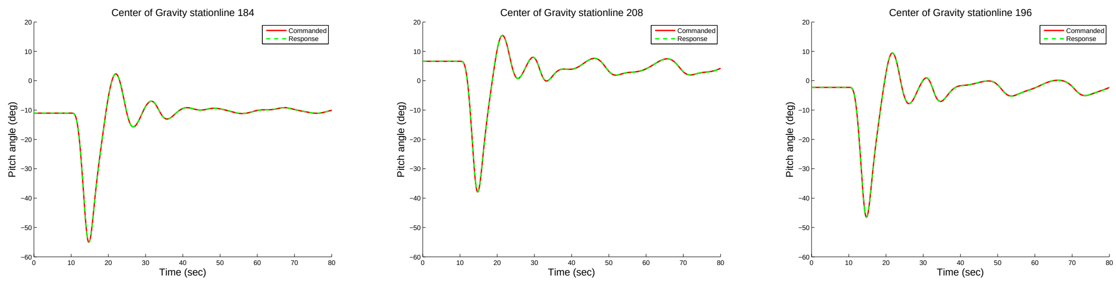


Figure 20: Variation of pitch angle with time for different center of gravity configurations without neural network