Nonlinear Stabilization And Control Of Medium Range Surface To Air Interceptor Missiles

2009

Mark Snyder
University of Central Florida

Find similar works at: https://stars.library.ucf.edu/etd

University of Central Florida Libraries http://library.ucf.edu

Part of the Electrical and Electronics Commons

STARS Citation

Snyder, Mark, "Nonlinear Stabilization And Control Of Medium Range Surface To Air Interceptor Missiles" (2009). Electronic Theses and Dissertations. 4180.
https://stars.library.ucf.edu/etd/4180

This Masters Thesis (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of STARS. For more information, please contact lee.dotson@ucf.edu.
NONLINEAR STABILIZATION AND CONTROL OF MEDIUM RANGE SURFACE TO AIR INTERCEPTOR MISSILES

by

MARK G. SNYDER
B.S. in Electrical Engineering
University of Central Florida, 2006

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering in the School of Electrical Engineering and Computer Science in the College of Engineering and Computer Science at the University of Central Florida Orlando, Florida

Spring Term
2009
SECURITY DISCLAIMER

The results of this research work were developed using a highly accurate aerodynamic missile model generated by the United States Air Forces MISSILE DATCOM which is a United States State Department International Traffic in Arms Regulations (ITAR) controlled data compendium and is not intended for use by persons of foreign nationality. Only the results of this research work have been released and in keeping with ITAR restrictions the aerodynamic model is to remain undisclosed to the general public. The usage of an accurate missile model is to effectively evaluate the performance of a nonlinear autopilot design which is the scope of this document. For more information on ITAR regulations please contact the United States State Department.
Nonlinear stabilization and control autopilots are capable of sustaining nominal performance throughout the entire flight envelope an interceptor missile may encounter during hostile engagements and require no gain scheduling to maintain autopilot stability. Due to non minimum phase conditions characteristic of tail controlled missile airframes, a separation of time scales within the dynamic equations of motion between rotational and translational differential equations was enforced to overcome unstable effects of non minimum phase. Dynamic inversion techniques are then applied to derive linearizing equations which, when injected forward into the plant result in a fully controllable linear system. Objectives of the two time scale control architecture are to stabilize vehicle rotational rates while at the same time controlling acceleration within the lateral plane of the vehicle under rapidly increasing dynamic pressure. Full 6 degree of freedom dynamic terms including all coriolis accelerations due to translational and rotational dynamic coupling have been taken into account in the inversion process. The result is a very stable, nonlinear autopilot with fixed control gains fully capable of stable nonlinear missile control. Several actuator systems were also designed to explore the destabilizing effects second order nonlinear actuator characteristics can have on nonlinear autopilot designs.
DEDICATED TO MY MOTHER AND FATHER JOHN AND DEE ANN WHO STOOD BEHIND ME THROUGHOUT ALL OF MY FAILURES AND ULTIMATELY GAVE ME THE STRENGTH, RESOURCES AND ENCOURAGEMENT TO SUCCEED AT ALL OF MY LIFE PURSUITS AND ENDEAVORS. YOUR LOVE AND SUPPORT WILL NEVER BE FORGOTTEN………..

THANK YOU
ACKNOWLEDGMENTS

Special thanks would like to be extended to Dr. Zhihua Qu, University of Central Florida for his guidance and support throughout the long development of this project. Also deserving of special thanks is Bill Galbraith for his furnishing of many vital resources towards this project.
# TABLE OF CONTENTS

LIST OF FIGURES ......................................................................................................................... viii  
LIST OF ACRONYMS/ABBREVIATIONS ................................................................................ xi  

CHAPTER 1  INTRODUCTION ................................................................................................... 1  
  Old Methods of Control ......................................................................................................... 2  
  The Concept of Dynamic Inversion ...................................................................................... 3  
  Vehicle Characteristics ......................................................................................................... 4  

CHAPTER 2  KINEMATICS OF TRANSLATION AND ROTATION ............................................. 7  
  Critical Support Data .......................................................................................................... 10  
  Determining Rate of Change of Gravity Bias ...................................................................... 13  

CHAPTER 3  NON MINIMUM PHASE FOR TAIL CONTROLLED MISSILES ....................... 16  
  Non Minimum Phase .......................................................................................................... 16  
  Separation of Time Scales ................................................................................................... 18  
  Time Scale Objectives ......................................................................................................... 19  

CHAPTER 4  VEHICLE AERODYNAMICS .............................................................................. 21  
  Stability Derivatives .......................................................................................................... 23  
  Setting the Stage for Dynamic Inversion ........................................................................... 23  

CHAPTER 5  DYNAMIC MODEL ............................................................................................ 26  

CHAPTER 6  ACTUATOR DYNAMICS ................................................................................... 31  
  Actuator Types ................................................................................................................... 31  
  Second Order Nonlinear Actuator with Dynamic Inversion Compensator ...................... 33  
  Creating the Nonlinear Actuator Model ............................................................................ 35  

CHAPTER 7  NONLINEAR AUTOPILOT ................................................................................ 41
Effects of Rapid Dynamic Pressure ................................................................. 42
Slow Time Scale Linearizing Equation and Controller- Controlling Normal Acceleration .... 44
Quadratic Slow Time Scale Form ...................................................................... 55
Fast Time Scale Linearizing Equation and Controller- Stabilizing Body Rate q .......... 60
Implementing the Controls .............................................................................. 63
Slow Time Scale Linearizing Equation and Controller- Controlling Side Acceleration .... 65
Fast Time Scale Linearizing Equation and Controller- Controlling Body Rate r .......... 70
CHAPTER 8 VEHICLE ROLL AXIS STABILIZATION .................................... 75
CHAPTER 9 ANALYSIS OF VEHICLE PERFORMANCE .............................. 81
Analysis of Test Results .................................................................................. 97
CONCLUSION .................................................................................................. 99
REFERENCES ............................................................................................... 101
LIST OF FIGURES

Figure 1: Fin Configuration for Zero Degree Roll Orientation ...................................................... 5
Figure 2: Coordinate Systems......................................................................................................... 5
Figure 3: Error Signal ................................................................................................................... 17
Figure 4: General Autopilot/Plant Form ....................................................................................... 19
Figure 5: Body/Aerodynamic Axes .............................................................................................. 24
Figure 6: Vehicle Dynamics Block ............................................................................................... 28
Figure 7: Vehicle Dynamics Block with Support Data Network .................................................. 29
Figure 8: Second Order Linear Actuator Structure ...................................................................... 34
Figure 9: Second Order Nonlinear Actuator Structure ................................................................ 36
Figure 10: Dynamic Inversion Controller with Actuator Plant ..................................................... 37
Figure 11: Linearizing Control Law ............................................................................................. 38
Figure 12: Proportional Plus Integral Plus Derivative Controller ............................................... 38
Figure 13: Actuator Network ........................................................................................................ 40
Figure 14: Contrast Between Body/Aerodynamic Axes ............................................................... 47
Figure 15: Complete Autopilot Structure (Pitch Channel) ............................................................ 64
Figure 16: Fast Time Scale Response ........................................................................................... 82
Figure 17: Slow Time Scale Response ......................................................................................... 82
Figure 18: Normal Acceleration ................................................................................................... 83
Figure 19: Angle of Attack ........................................................................................................... 83
Figure 20: Pitch Moment .............................................................................................................. 84
Figure 21: Airframe Normal Force ............................................................................................... 84
Figure 44: Slow Time Scale Response ......................................................................................... 96
# LIST OF ACRONYMS/ABBREVIATIONS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>ANGULAR ROLL RATE</td>
</tr>
<tr>
<td>$q$</td>
<td>ANGULAR PITCH RATE</td>
</tr>
<tr>
<td>$r$</td>
<td>ANGULAR YAW RATE</td>
</tr>
<tr>
<td>$u$</td>
<td>TRANSLATIONAL AXIAL VELOCITY</td>
</tr>
<tr>
<td>$v$</td>
<td>TRANSLATIONAL SIDE VELOCITY</td>
</tr>
<tr>
<td>$w$</td>
<td>TRANSLATIONAL NORMAL VELOCITY</td>
</tr>
<tr>
<td>$\bar{q}$</td>
<td>DYNAMIC PRESSURE</td>
</tr>
<tr>
<td>$F_{ap1}$</td>
<td>AIRFRAME FORCE IN AXIAL DIRECTION</td>
</tr>
<tr>
<td>$F_{ap2}$</td>
<td>AIRFRAME FORCE IN SIDE DIRECTION</td>
</tr>
<tr>
<td>$F_{ap3}$</td>
<td>AIRFRAME FORCE IN NORMAL DIRECTION</td>
</tr>
<tr>
<td>$m_{B1}$</td>
<td>ROLL MOMENT</td>
</tr>
<tr>
<td>$m_{B2}$</td>
<td>PITCH MOMENT</td>
</tr>
<tr>
<td>$m_{B3}$</td>
<td>YAW MOMENT</td>
</tr>
<tr>
<td>$t_{13}$</td>
<td>AXIAL COMPONENT OF GRAVITY BIAS</td>
</tr>
<tr>
<td>$t_{23}$</td>
<td>SIDE COMPONENT OF GRAVITY BIAS</td>
</tr>
<tr>
<td>$t_{33}$</td>
<td>NORMAL COMPONENT OF GRAVITY BIAS</td>
</tr>
<tr>
<td>$i_{13}$</td>
<td>TIME RATE OF CHANGE OF GRAVITY BIAS - AXIAL</td>
</tr>
<tr>
<td>$i_{23}$</td>
<td>TIME RATE OF CHANGE OF GRAVITY BIAS - SIDE</td>
</tr>
<tr>
<td>$i_{33}$</td>
<td>TIME RATE OF CHANGE OF GRAVITY BIAS - NORMAL</td>
</tr>
</tbody>
</table>

g \quad \text{RATE OF GRAVITY}

C_{Na} \quad \text{ANGLE OF ATTACK STABILITY DERIVATIVE – NORMAL FORCE}

C_{N\delta} \quad \text{CONTROL STABILITY DERIVATIVE – NORMAL FORCE}

C_{ma} \quad \text{ANGLE OF ATTACK STABILITY DERIVATIVE – PITCH MOMENT}

C_{m\delta} \quad \text{DEFLECTION CONTROL STABILITY DERIVATIVE – PITCH MOMENT}

C_{y\beta} \quad \text{SIDESLIP STABILITY DERIVATIVE – SIDE FORCE}

C_{y\delta} \quad \text{DEFLECTION CONTROL STABILITY DERIVATIVE – SIDE FORCE}

C_{Y\beta} \quad \text{SIDESLIP STABILITY DERIVATIVE – YAW MOMENT}

C_{Y\delta} \quad \text{DEFLECTION CONTROL STABILITY DERIVATIVE – YAW MOMENT}

\delta_C \quad \text{AUTOPILOT OUTPUT COMMAND}

\delta \quad \text{ACTUATOR OUTPUT COMMAND}

\dot{\delta} \quad \text{TIME RATE OF CHANGE OF ACTUATOR DEFLECTION}

q_C \quad \text{PITCH RATE COMMAND}

r_C \quad \text{yaw RATE COMMAND}

\alpha \quad \text{ANGLE OF ATTACK}

\beta \quad \text{SIDESLIP}

I_1 \quad \text{PRINCIPAL MOMENT OF INERTIA ABOUT LONGITUDINAL AXIS (1B)}

I_2 \quad \text{PRINCIPAL MOMENT OF INERTIA ABOUT SIDE AXIS (2B)}

I_3 \quad \text{PRINCIPAL MOMENT OF INERTIAL ABOUT NORMAL AXIS (3B)}
CHAPTER 1
INTRODUCTION

The stabilization and control of missile airframes is highly complex in nature; in this case, it is rocket science. The equations of motion are nonlinear, coupled differential equations; to make matters worse, highly nonlinear aerodynamic forces and moments are nested within the equations of motion making a pure mathematical nightmare for any controls engineer to tackle, especially if a nonlinear control system design is the target end result. The aerodynamic forces and moment’s incident on missile airframes are not clear cut in nature. Generic equations for modeling missile forces and moments exist but to capture exact characteristics of a specific design, exhaustive wind tunnel testing must occur in the final design stages. To get a general idea of how a design might perform in the early stages of design, flow modeling software such as the United States Air Forces MISSILE DATCOM can be used to generate the necessary aerodynamic data imperative in autopilot control design. While the scope of this research report is not missile aerodynamics, a generalized focus on the concept must be given because of the strong coupling between autopilot design and aerodynamic characteristics of the vehicle in question. The missile model used within this report is of the same physical size and weight as the Patriot Interceptor; however, the aerodynamic data used was generated by the U.S. Air Forces MISSILE DATCOM and may not be exact and specific in nature to the Patriot. This particular aerodynamic model will, however, capture the general performance characteristics of a large missile such as the Patriot.
**Old Methods of Control**

Many research documents make simplifications to some of the equations of motion or even reduce the degrees of freedom which, in turn, reduce the overall complexity of the control problem; however, the results of any advanced guidance and control study based on such simplified models are questionable at best. Since its birth as a field of technology, the stabilization and control of missile airframes has been achieved through a seemingly straightforward method of airframe linearization about certain points of interest along a reference flight and determining autopilot gains required for stable operation based on this linearized airframe model. The autopilot design is a linear-type controller which requires different gains under different operating conditions. In other words, the gains for the linear controller are dynamically switched or “scheduled” to meet the needs of the current flight conditions in order to maintain autopilot stability. This is a way to control a highly nonlinear dynamic system with linear type controllers. This method has been used by the aerospace industry since the beginning but the days are numbered for this venerable design technique because of increasing demand for highly maneuverable stealthy weapons. Two basic problems arise from this linearizing technique; 1) literally thousands of operating points must be linearized about thousands of potential reference flights in order to develop a table of autopilot gains that can be dynamically switched depending on the flight conditions encountered throughout the entire potential flight envelope. It is obvious that extensive amounts of gain data are required in order to cover all of the possible flight conditions that may be encountered especially by an interceptor missile engaging a hostile target. 2) If the missile encounters an extreme operating condition not previously addressed by the linearization process, the missile autopilot may potentially become unstable and saturate the
control actuators, resulting in a catastrophic failure of the vehicle. In other words, stability cannot always be guaranteed. The basis for this failure can be found in the very process of linearization itself. If a nonlinear system is linearized about a certain point of operation and a linear controller is designed, it must be guaranteed that the nonlinear dynamic system state trajectories do not stray too far away from this point of operation. If the system state trajectories venture outside of the region of attraction for this linearized operating point, the system states will begin to diverge and the autopilot becomes unstable.

**The Concept of Dynamic Inversion**

The previous discussion describes the process of gain scheduling which has its roots in Lyapunov linearization techniques. While this is a tried and true design method, current demands for highly agile missiles and kinetic strike interceptor technology is placing more and more burden on the controls engineer to develop nonlinear control designs robust enough to remain stable under the most extreme of operating conditions while at the same time eliminating the costly and time consuming process of gain scheduling. While a rich and fully developed history exists for linear control theory, the same cannot be said for nonlinear control. Interestingly enough, linear control theory can in fact be applied to the design of nonlinear autopilots. The design technique is called Dynamic Inversion (also known as Feedback Linearization) and if designed properly will robustly handle any extreme flight condition the missile may encounter within the expected flight envelope. Problems can arise, however, when designing a feedback linearization control system. Missile airframes exhibit a strong non minimum phase which results in a failure of direct feedback linearization. Several forms of feedback linearization have
been developed over the last few years to remove the non minimum phase which then allows successful application of Dynamic Inversion. One technique, called the Two-Time-Scale approach is a modified version of Dynamic Inversion and is the focus of this research report. The missile dynamic equations of motion are separated into slow and fast dynamics and are individually targeted for control to achieve body rate stabilization and lateral acceleration control, which are the two fundamental objectives of any autopilot design. Each chapter of this report deals with the intricacies of applying a two time scale design to missile dynamics which eventually leads to a full 6 degree of freedom autopilot design. We shall now begin the extensive process of introducing preliminaries, discussing design hurdles, building the final design and evaluating the overall design performance.

**Vehicle Characteristics**

- **Gravity** $9.81 \frac{m}{s^2}$
- **Mass** $1000 \ kg$
- **Principal Moments of Inertia**
  - Inertia about 1B axis $21.0125 \ kgm^2$
  - Inertia about 2B axis $2093.84 \ kgm^2$
  - Inertia about 3B axis $2093.84 \ kgm^2$

**Vehicle Roll Orientation/Fin Configuration**

Roll orientation $0^\circ$ with X fin configuration


**Physical Dimensions**

Length 5.2 m  
Width .4100 m

**Vehicle Steering Policy**

Skid-to-turn

**Coordinate Systems Used in Simulation**

Figure 2 shows the coordinate systems associated with this missile design. Body coordinates are fixed to the vehicle body hence the name. Most of the equations derived in this document are expressed in body coordinates unless otherwise specified. Aerodynamic coordinates are many times used to specify aerodynamic data and are closely related to body coordinates. Care must be
taken to ensure the proper transformation between body and aerodynamic equations is used to prevent simulation and design errors. Local coordinates are typically fixed to the ground normally at the point of launch.
We will begin this chapter by presenting Newton’s and Euler equations for translation and rotation respectively; but first, a brief description of the two sets of equations is in order. Six degree of freedom (dof) simulations obviously require six separate dynamic equations that describe a vehicle’s motion in three dimensional space. The first three degrees of freedom describe the translation of the center of mass of the vehicle and typically take the form of a displacement vector from the point of launch, as in this case, to the vehicle’s center of mass. That is, the movement of the vehicle’s center of mass with respect to the inertial coordinate system (the local-level axes). Translational equations, however, only describe the movement of the center of mass and give no indication as to the vehicle’s orientation about the center of mass in inertial space. This is where the last 3 d.o.f. come into play. Rotational equations (Euler’s equations) describe the vehicle’s orientation about the vehicle’s center of mass with respect to an inertial coordinate system, again the local-level coordinate system. Equations 2.1 through 2.3 represent the translational dynamic equations of the vehicle’s center of mass in the axial, side and normal directions respectively.

Newton’s Equations (Translation):

\[ \dot{u} = rv - qw + \frac{f_{ap1}}{m} + t_{13}g \]  
\[ \dot{v} = pw - ru + \frac{f_{ap2}}{m} + t_{23}g \]  
\[ \dot{w} = qu - pv + \frac{f_{ap3}}{m} + t_{33}g \]
All components, with the exception of gravity, of equations 2.1 – 2.3 are understood to be in body coordinates unless otherwise specified. Gravity is given in local coordinate axes and transformed by the direction cosine into components of gravity along each of the principle body axes. Throughout the remainder of this report, both rotational and translational differential equations will be given without the body axis notation as shown in equations 2.1a – 2.3a. It should be understood all quantities are in body axes. Care must be taken when modeling aerodynamic forces and moments because most of the time they are specified in aerodynamic axes which requires the proper transformations to be made before being injected into these equations.

\[
\begin{align*}
\dot{u}_B^B &= \left[\dot{r}_B^B \right]^B - \left[q_B^B w_B^B \right]^B + \left[\frac{f_{a,p1}}{m} \right]^B + \left[t_{13}\right]^B G^L \right]^L \quad (2.1a) \\
\dot{v}_B^B &= \left[p_B^B w_B^B \right]^B - \left[r_B^B u_B^B \right]^B + \left[\frac{f_{a,p2}}{m} \right]^B + \left[t_{23}\right]^B G^L \right]^L \quad (2.2a) \\
\dot{w}_B^B &= \left[q_B^B u_B^B \right]^B - \left[p_B^B v_B^B \right]^B + \left[\frac{f_{a,p3}}{m} \right]^B + \left[t_{33}\right]^B G^L \right]^L \quad (2.3a)
\end{align*}
\]

Where:  
\( u \) = Axial vehicle velocity in body coordinates.  
\( v \) = Side vehicle velocity in body coordinates.  
\( w \) = Normal vehicle velocity in body coordinates.  
\( p \) = Vehicle roll rate.  
\( q \) = Vehicle pitch rate.  
\( r \) = Vehicle yaw rate.  
\( \frac{f_{a,p1}}{m} \) = Vehicle axial force divided by vehicle mass.  
\( \frac{f_{a,p2}}{m} \) = Vehicle side force divided by vehicle mass.  
\( \frac{f_{a,p3}}{m} \) = Vehicle normal force divided by vehicle mass.
where:
- $t_{13}g$ = axial gravity bias.
- $t_{23}g$ = side gravity bias.
- $t_{33}g$ = normal gravity bias.

Euler’s Equations (Rotation):

\[
\dot{p} = I_1^{-1} m_{B1} \tag{2.4}
\]

\[
\dot{q} = I_2^{-1} ((I_3 - I_1)pr + m_{B2}) \tag{2.5}
\]

\[
\dot{q} = I_3^{-1} ((I_1 - I_2)pr + m_{B3}) \tag{2.6}
\]

Where:
- $p$ = vehicle roll rate.
- $q$ = vehicle pitch rate.
- $r$ = vehicle yaw rate.

$I_1^{-1}, I_2^{-1}, I_3^{-1} =$ inverse elements of the inertia tensor that appear as

\[
\begin{bmatrix}
I_1^{-1} & 0 & 0 \\
0 & I_2^{-1} & 0 \\
0 & 0 & I_3^{-1}
\end{bmatrix}
\]

$m_{B1} =$ roll moment (from aerodynamic force / moment equations

$m_{B2} =$ pitch moment (from aerodynamic force / moment equations

$m_{B3} =$ yaw moment (from aerodynamic force / moment equations

Notice that both sets of equations are first order coupled differential equations. Aerodynamic forces and moments comprise inputs to these equations along with gravity bias and complete the final form for the set of rotational equations. This set of rotational differential equations, along with the translational Newton’s equations is solved at each time step during execution of the simulation. The resulting state variables produced are $u, v, w, p, q$ and $r$. The vehicle state vector will be addressed in greater detail in chapter 5, Dynamic Model.
Critical Support Data

The Translational and rotational equations of motion form the heart of a missile system and provide key components in the form of state vector outputs to other subsystems of the missile simulation; however, while not part of the missile’s dynamic state vector, crucial support data must be computed and provided to the overall system function to round out and form a complete missile simulation. One such set of support data comprises vehicle attitude with respect to an inertial frame. Development of inertial attitude relies heavily on computation of the vehicle state vector but other computational issues can plague the proper calculation of these data sets. The issues involved in determining attitude center mainly around singularities in the Euler equations produced at vertical launch, steep climb or steep vehicle dives. One method of avoiding this is to use the quaternion methodology which contains no mathematical singularities. This process involves three steps:

1) Use solutions of the Newton and Euler equations (vehicle state vector) as inputs to the quaternion differential equations.
2) Solve the quaternion differential equation.
3) Use the state vector solution of the quaternion to solve the final expressions for vehicle attitude.

The quaternion differential equation (D.E.) and corresponding attitude equations are presented in equations 2.7 – 2.10. The use of the quaternion D.E. does not end there however. The most important tool used in missile guidance is produced from the quaternion method. It is
known as the direction cosine matrix. It is nothing more than a coordinate transformation matrix that mathematically relates body axes to local-level axes (the inertial frame in this case). For instance, vehicle velocity expressed in body coordinates can be expressed in inertial coordinates directly through use of the direction cosine matrix.

\[
\begin{bmatrix}
q_0 \\
q_1 \\
q_2 \\
q_3
\end{bmatrix}
= \frac{1}{2}
\begin{bmatrix}
0 & -p & -q & -r \\
p & 0 & r & -q \\
q & -r & 0 & p \\
r & q & -p & 0
\end{bmatrix}
\begin{bmatrix}
q_0 \\
q_1 \\
q_2 \\
q_3
\end{bmatrix}
\]  
(2.7)

Notice the quaternion D.E. uses body rates calculated from the solution of the Euler D.E.’s. Do not mistake \( q_n \) in this representation for either \( q \) pitch rate or \( \overline{q} \) for dynamic pressure. Once the quaternion D.E. solution is found, attitude angles can be determined.

They are as follows:

\[
\tan \Psi = \frac{2(q_2q_3 + q_0q_1)}{q_0^2 + q_1^2 + q_2^2 + q_3^2}
\]  
(2.8)

\[
\sin \theta = -2(q_1q_3 - q_0q_2)
\]  
(2.9)

\[
\tan \phi = \frac{2(q_2q_3 + q_0q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}
\]  
(2.10)

Where \( \Psi \) = Yaw angle, \( \theta \) = Pitch angle, \( \phi \) = Roll angle

Using the quaternion D.E., attitude angles of the vehicle with respect to the inertial frame is found in equations 2.8 – 2.10.
The next tool we develop is the direction cosine matrix. As stated before, the direction cosine is imperative in transforming coordinates between body axes and local-level coordinates. The direction cosine is given in equation 2.11.

\[
T^B_L = \begin{bmatrix}
q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 + q_0q_3) & 2(q_1q_3 - q_0q_2) \\
2(q_1q_2 - q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 + q_0q_1) \\
2(q_1q_3 + q_0q_2) & 2(q_2q_3 - q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2
\end{bmatrix}
\] (2.11)

To summarize, the quaternion accepts body rates from the solution of the Euler D.E. and in turn is used to solve the quaternion D.E. From the quaternion state vector, attitude angles and direction cosine can be determined. It is obvious that the Euler differential equations provide the necessary information that all other missile subsystems such as guidance, for instance, needs in order to successfully intercept a target.

Two remaining sets of support data are still of great interest in this chapter and are calculated from Newton’s equations; i.e., the translational equations of motion. They are angle of attack and sideslip. Angle of attack, in the Cartesian sense is the angle formed when the missiles nose pitches up or down which misaligns the 1B axis from the axial component of resultant velocity. In Cartesian body coordinates, angle of attack is expressed in equation 2.12. Sideslip is the yaw channel equivalent to angle of attack and is expressed in equation 2.13. As the vehicle “skids” into a turn, the vehicle center of mass scribes an arc as the vehicle turns. The nose of the vehicle is “pitched” and “yawed” away from the axial velocity component resulting in an angle of attack and sideslip. Sideslip and angle of attack are the key components required as data inputs to the aerodynamic equations. Inputting these two values along with autopilot outputs result in airframe
forces and moments being generated which comprise part of the inputs to dynamic equations 2.1 – 2.6.

\[
[\alpha]^{B} = \tan^{-1}\left(\frac{[w]^{B}}{[u]^{B}}\right) \\
[\beta]^{B} = \tan^{-1}\left(\frac{[v]^{B}}{[u]^{B}}\right)
\] (2.12) (2.13)

Once again it is important to note that each of these critical values is calculated in body axes. The importance of specifying the coordinate system cannot be stressed enough. While the linearizing functions that drive autopilot operation utilize both angle of attack and sideslip in body coordinates, aerodynamic modeling uses the two values calculated in aerodynamic axes and it is extremely important to know when to use the proper forms and apply the corresponding transformations. This critical design point is illustrated in chapter 4, Aerodynamics and again in chapter 7, Nonlinear Autopilot Design.

**Determining Rate of Change of Gravity Bias**

During the development of nonlinear autopilots, a need for mathematical expressions representing the time rate of change of the three axis components of gravity is required and will be derived in this section. Inspection of column 3 of the direction cosine, equation 2.11, reveals the gravitational components incident on the airframe in the axial direction \( t_{13} \), side direction \( t_{23} \) and the normal direction \( t_{33} \).

\[
t_{13} = 2(q_{1}q_{3} - q_{0}q_{2})
\] (2.14)
Because of the design requirements of this particular type of nonlinear autopilot, we seek expressions for the time rate of change of equations 2.15 – 2.16 as the vehicle maneuvers throughout its flight envelope. Let us now find the expressions for $i_{13}, i_{23},$ and $i_{33}$.

\[
\begin{align*}
    i_{13} &= 2(q_1 \dot{q}_3 + \dot{q}_1 q_3 - (q_0 \dot{q}_2 + \dot{q}_0 q_2)) \\
    i_{23} &= 2(q_2 \dot{q}_3 + \dot{q}_2 q_3 + (q_0 \dot{q}_1 + \dot{q}_0 q_1)) \\
    i_{33} &= 2q_0 \dot{q}_0 - 2q_1 \dot{q}_1 - 2q_2 \dot{q}_2 + 2q_3 \dot{q}_3
\end{align*}
\]

Next, we must obtain expressions for $\dot{q}_0, \dot{q}_1, \dot{q}_2$ and $\dot{q}_3$. Inspection of the scalar equations for the quaternion differential equations reveals the following,

\[
\begin{align*}
    \dot{q}_0 &= \frac{1}{2}((0q_0 - pq_1 - qg_2 - rq_3) \\
    \dot{q}_1 &= \frac{1}{2}(pq_0 + 0q_1 + rq_2 - qg_3) \\
    \dot{q}_2 &= \frac{1}{2}((qq_0 - rq_1 + 0q_2 + pq_3) \\
    \dot{q}_3 &= \frac{1}{2}(rq_0 + qg_1 - pq_2 + 0q_3)
\end{align*}
\]

After the appropriate substitutions and reductions are made, we arrive at the final expressions for each of the gravity bias rate of change components,

\[
\begin{align*}
    i_{13} &= 2rq_0 q_1 + qq_0 q_2 - 2pq_1 q_2 - qg_2 q_3 - qg_0 q_0 + qg_2 q_2 \\
    i_{23} &= 2rq_0 q_2 - 2rq_1 q_3 - pq_2 q_2 + pq_3 q_3 + pq_0 q_0 - pq_0 q_1 \\
    i_{33} &= -2pq_0 q_1 - 2qq_0 q_2 + 2qq_0 q_3 - 2pq_2 q_3
\end{align*}
\]
At this point, all critical mathematical equations that comprise the system support data sets have been established. From this point, a comprehensive build-up type approach will ensue, ultimately leading us to the development of autopilot control laws. Key design issues must first be laid out in chapters 3, 4, and 5 and analyzed in order to reach the proper format conducive to employing dynamic inversion.
CHAPTER 3
NON MINIMUM PHASE FOR TAIL CONTROLLED MISSILES

Non Minimum Phase

Tail controlled missiles exhibit non minimum phase, which means the airframe can become unstable under certain conditions. Under normal conditions, that is, dynamic systems with no presence of non minimum phase, dynamic inversion can be successfully employed as a means of nonlinear control; however, dynamic inversion fails in the presence of non minimum phase. These instabilities manifest themselves in the aerodynamic equations that represent force in the normal plane. Referring back to equation 2.3, the non minimum phase enters the equation through the aerodynamic normal force. Basically, non minimum phase in this context can be characterized as an un-commanded movement of the vehicle in a direction opposite to that which is desired. For instance, under the steering policy chosen for this simulation study, a negative pitch deflection of the tail control fins must give rise to a negative increase in the normal force and a positive pitching moment (assuming a body coordinate representation). There is a brief moment however, after the negative deflection occurs that a positive or downward normal force will occur. The positive normal force quickly dissipates and builds into a negative (upward in body coordinates) force as the angle of attack builds. It is this small transient period of un-commanded motion that wreaks havoc on autopilots. The reasons are fairly clear. An autopilot must receive the difference of two signals, called the error signal, between the commanded acceleration and normal acceleration. This allows the autopilots to determine the proper output control signal to apply to the plant. Under the non minimum phase conditions, the difference of
two signals becomes a sum of two signals. Once this situation unfolds, the autopilots immediately become unstable causing actuators to saturate, resulting in a complete loss of control of the vehicle. Let’s illustrate this point with a simple example. If the missile is traveling straight down the 1L axis and it detects the target above it, the guidance system issues a negative acceleration command to steer the vehicle up. The autopilots issue a negative pitch deflection command to the fin actuators. Because a momentary positive increase in the normal force occurs due to the nonminimum phase and is ultimately sensed by the accelerometer cluster, the autopilots are now faced with the sum of two signals as illustrated in figure 3.

Figure 3: Error Signal

Nonminimum phase is a relatively short-lived condition - provided the vehicle has enough forward thrust. Underpowered vehicles may have a significantly harder time dealing with nonminimum phase because reducing nonminimum phase and driving it toward zero is dependent on building sufficient enough normal force to stop the downward free fall and begin the desired upward motion. The dynamics behind building adequate normal force to counteract this motion depends solely on angle of attack. What we must see in body coordinates is a negative angle of
attack build resulting in more of the windward side of the vehicle pressing against the atmosphere. The scenario that unfolds is this; the aerodynamic effecters (tail control fins) create a small aerodynamic force that “kicks” the tail of the vehicle downward; however, insufficient angle of attack exists initially to cause a significant increase in force on the airframe. Thus, due to the effects of the gravity bias incident on the vehicle, a brief period of downward acceleration results, causing the sum of two negative signals between the guidance command and the vehicle’s actual acceleration within the computed error calculation.

**Separation of Time Scales**

One way of dealing with the non minimum phase issue is the separation of vehicle dynamics into separate time scales of different “speed rates”. One time scale is considered to be slow while the other is considered fast. The translational acceleration of the vehicle, equations 2.1 - 2.3, is considered the slow time scale while the rotational body rates, equations 2.4 – 2.6, and actuator dynamics are lumped into the fast time scale. The two time scale approach is only effective if there is a clear speed difference between these two dynamic entities and, fortunately for missile designers, there is. The two time scale approach controls the non minimum phase by counting on the fact that the fast time scale is capable of being stabilized and controlled much faster than the slow time scale. The non minimum phase, in fact, appears within the slow time scale which makes for a lucky coincidence. If we can be guaranteed the fast time scale can be stabilized and controlled in a much quicker fashion than the slow time scale, actuator commands can be issued by the autopilots to execute vehicle maneuvers even though the slow time scale may be briefly bounding towards instability. If the fast time scale remains stabilized during non minimum
phase, the mounting normal force due to elevating angle of attack will subsequently stabilize and drive the non minimum phase condition to zero resulting in lateral vehicle acceleration in the proper direction.

**Time Scale Objectives**

Each time scale serves its own control purpose. The main objective in missile control is stabilization of the angular body rates and control of lateral acceleration; therefore, if we consider the design of the pitch channel autopilot, the fast time scale is designed to stabilize the pitch rate of the vehicle while the slow time scale is designed to control normal or lateral acceleration. Since the fast time scale is free of any non minimum phase and operates much faster than the slow time scale, the pitch rate can be controlled and stabilized while the slow time scale is dealing with the non minimum phase. The generalized form of an autopilot/plant system is shown in figure 4. Inputs to a dynamic inversion autopilot are guidance commands and feedback of the vehicles states. Autopilot outputs are control commands to the rear aerodynamic fins.

![Figure 4: General Autopilot/Plant Form](image)
The most important point to make about figure 3.2 is the dynamic inversion data vector that forms the feedback network from the missile airframe/navigation computer to the autopilots. The dynamic inversion relies on a tremendous supply of information in real time to facilitate the construction of the linearizing equations which are responsible for the cancellation of all nonlinear terms. In robotics, this method is better known as computed torque control where all known dynamics are calculated online and then fed forward into the plant. The dynamic inversion data vector seen in figure 4 consists not only of vehicle states but also the critical support data previously discussed in chapter 2. It supplies everything needed to implement dynamic inversion effectively.
CHAPTER 4
VEHICLE AERODYNAMICS

Representing the aerodynamics of any aerospace vehicle can be a formidable task, especially if predictive software algorithms such as MISSILE DATCOM are not available. Typically, aerodynamic equations are given as a Taylor series expansion in which the coefficients for each of the terms are given as a function of mach number and angle of attack. The reasons for this are simple. Much of the aerodynamic testing that occurs for a vehicle in development takes place inside wind tunnels and it makes sense that the data collected are functions of the parameters used during the tests.

In the case of this simulation study however, wind tunnel data was not available but MISSILE DATCOM was used to generate appropriate aerodynamic stability data. Aerodynamic coefficients can be mathematically expressed through partial derivative relationships and hence they are given the name “stability derivatives”. The aerodynamic coefficient equations appear below in equations 4.1 through 4.4

YAW CHANNEL FORCE (\(C_y\)) AND MOMENT (\(C_n\)) COEFFICIENT EQUATIONS:

\[
C_y^{\text{FORCE}} = \frac{\partial C_y}{\partial \beta} \beta + \frac{\partial C_y}{\partial \delta_n} \delta_n \tag{4.1}
\]

\[
C_{\text{MOMENT}} = \frac{\partial C_n}{\partial \beta} \beta + \frac{\partial C_n}{\partial \gamma} \frac{rl}{2V} + \frac{\partial C_n}{\partial \delta_n} \delta_n \tag{4.2}
\]

PITCH CHANNEL FORCE AND MOMENT COEFFICIENT EQUATIONS:
\[ C_{N\text{FORCE}} = \frac{\partial C_Y}{\partial \alpha} \alpha + \frac{\partial C_Y}{\partial \delta_n} \delta_n \]  
\[ (4.3) \]

\[ C_{N\text{MOMENT}} = \frac{\partial C_n}{\partial \beta} \beta + \frac{\partial C_n}{\partial r} \frac{rl}{2V} + \frac{\partial C_n}{\partial \delta_n} \delta_n \]  
\[ (4.4) \]

In both the yaw and pitch moment equations above, 4.2 and 4.4 respectively, the center terms represent damping and typically can be ignored because the synthetic damping provided by the autopilots overwhelm these small quantities. In equations 4.1 and 4.3 the right most term describes the force exerted by the aerodynamic control fins. The right-most terms in 4.2 and 4.4 describes the moment applied to the airframe by the control fins. The left-most side of these same equations describes the amount of moment due to sideslip (4.2) and angle of attack (4.4) and finally the left-most side of equations 4.1 and 4.3 describes the amount of airframe force due to sideslip (4.1) and angle of attack (4.3).

The methods for predicting aerodynamic stability derivatives are an extremely complex field and are not the intended scope of this document. In fact, in keeping with ITAR regulations no aerodynamic data will be revealed in this document. It is important, though, to point out the issues that arise when designing and simulating a dynamic inversion type control system because of the differences between the two main types of coordinate systems used on the missiles body, i.e., stability versus body coordinates. Seemingly insignificant errors in usage of these coordinate systems in the main design can result in corrupted simulation outputs. The problems caused by overlooking these slight differences in coordinate systems plagued the early stages of this project and created bugs in the simulation that were next to impossible to find. As stated before, to implement dynamic inversion, we must create the linearizing equation which requires taking a
portion of the dynamic model and calculating that model “online” as data streams in from the Inertial Measurement Unit (IMU). This online calculated model is then fed-forward into the plant and at least ideally cancels all nonlinear terms leaving behind a linear system which is easily controlled. Determining the linearizing equation for the proper coordinate system is very tricky and must be done with great care because if the proper transformations are not utilized in the linearizing equations, hard to find errors are sure to result.

**Stability Derivatives**

In equations 4.1 – 4.4, the partial derivatives are called stability derivatives and can be produced from wind tunnel testing or as direct outputs from predictive software routines such as MISSILE DATCOM. As discussed in the last section, it is of extreme importance to know what coordinate system the stability derivatives were computed for. Sometimes the data is provided in stability axes and other times it is given in body axes. It is crucial to the success of the final autopilot design to make the proper transformations between body axes and stability axes. The transformations are quite simple in nature but require great care and mathematical exactness when merged with a dynamic inversion type control scheme.

**Setting the Stage for Dynamic Inversion**

The overall concept of dynamic inversion is quite simple. A portion of the known vehicle dynamics are calculated on line from the instantaneous states of the vehicle and fed forward from the autopilots into the plant which consists of the dynamic equations of motion; the six differential equations that represent translation and rotation of the vehicle. The result is all
known nonlinear effects being canceled which leaves a simple linear system. Typically a linear control law such as a PID is more than sufficient to control the remaining dynamics. The known dynamics calculated online and fed forward into the plant is known as the linearizing equation and will be referred to many times within this document. Looking back to chapter 2, the reader will note the dynamic equations of motion are expressed in body axes. This is a simple yet crucially important point to keep in mind. It is clear from this observation that the linearizing equation, which comprises the autopilot logic, must be formulated in body coordinates as well, such that the dynamic inversion process can effectively cancel the known nonlinear dynamics. As stated previously, many times stability derivatives are provided in stability axes and not body axes; therefore, any aerodynamic data used as inputs to the linearizing equation must be expressed in body coordinates. Figure 5 illustrates the two coordinates systems of primary interest to us.

![Figure 5: Body/Aerodynamic Axes](image-url)

Figure 5: Body/Aerodynamic Axes
If aerodynamic stability data is provided in stability axes, a conversion must be made to transform the given data into body axes before being used in the linearizing equation. The transformations are as follows,

\[
\begin{bmatrix}
1B \\
2B \\
3B
\end{bmatrix} = \begin{bmatrix}
-1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & -1
\end{bmatrix}
\begin{bmatrix}
1A \\
2A \\
3A
\end{bmatrix}
\] (4.5)

An analytic proof of these design concepts pertaining to the proper implementation of the linearizing equation with respect to the coordinate systems will be given near the beginning of chapter 7. The subscripts in 4.5 stand for body and stability.
CHAPTER 5
DYNAMIC MODEL

In chapter 2, Kinematics of Translation and Rotation, the vehicle dynamics were introduced along with all of the support equations necessary to determine inertial attitudes, sideslip, and angle of attack, all of which are integral parts of any missile stabilization and control system. This chapter will focus on the dynamic model and its implementation within the Matlab environment. We begin this chapter by reintroducing the dynamic equations for translation and rotation of a missile airframe.

Translational Dynamics:

\[
\begin{align*}
\dot{u} &= rv - qw + \frac{f_{ap1}}{m} + t_{13}g \\
\dot{v} &= pw - ru + \frac{f_{ap2}}{m} + t_{23}g \\
\dot{w} &= qu - pv + \frac{f_{ap3}}{m} + t_{33}g
\end{align*}
\]  

(5.1)  

(5.2)  

(5.3)

Rotational Dynamics:

\[
\begin{align*}
\dot{p} &= I_1^{-1}m_{g1} \\
\dot{q} &= I_2^{-1}\left((I_3 - I_1)pr + m_{g2}\right) \\
\dot{r} &= I_3^{-1}\left((I_1 - I_2)pq + m_{g3}\right)
\end{align*}
\]  

(5.4)  

(5.5)  

(5.6)

We begin building the airframe dynamics around the translational and rotational dynamic equations. Both of these sets are first order, coupled nonlinear differential equations. Notice that state variables \( p, q, \) and \( r \) of the rotational equations are coupled into the translational equations. Also, notice the input to the translational equations require data from the direction cosine. The
sequence of calculations that must occur are as follows: the rotational differential equation must be solved first and the resulting state vector \( p \), \( q \), and \( r \) must be supplied to both the translational differential equation and the quaternion differential equation. Before any other calculations can be processed, the quaternion DE must be solved, at which time the quaternion state vector is handed off to the direction cosine equations which will then compute the gravity bias incident on the airframe. This calculated bias is then input to the translational mathematics, at which time the associated differential equation can be solved. The resulting state vector comprises \( u \), \( v \), and \( w \), the axial, side, and normal velocities respectively. It is not, however, as simple as connecting the Matlab simulation blocks together and hitting the run button. One of the most aggravating problems involved with the simulation of feedback control systems is the presence of algebraic loops buried deep within the simulation. Algebraic loops arise when the output of a simulation block is some function of the input to the block. These types of blocks exhibit “direct feed-through” in which part of the input must be used to calculate part of the output. The problem occurs when Matlab calls up loop solvers in an attempt to approximate the output of the block at time \( t=0 \). Simply setting the proper initial conditions most likely will not solve the problem, although in some rare circumstances it can. This type of issue is a prime example of the headaches that can await anyone attempting to simulate a complex missile control system. A six d.o.f. missile simulation will contain many algebraic loops and most of them will not cause large enough problems to prevent a simulation run. Most of the time a warning will be issued by the simulation environment; however, it will be the algebraic loop within the dynamic model that will require special attention. The translational dynamics, due to the coupling of all six differential equations, are the last equations in line to be solved. Critical data needed for the
solution of the translational differential equations reach the translational block at different points in time than does data to the rotational block. This causes serious algebraic loop problems which will most definitely prevent the simulation from running. Algebraic loop issues can require significant time to resolve and some of the methods for resolving them may be as simple as setting the proper initial conditions, rearranging the order of the dynamics or even using memory blocks to synchronize the arrival of data to certain blocks. Transport delays with a small time delay setting will help resolve the algebraic loop by creating a slight break in the circuit path. Rearranging all of the above dynamic equations is basically out of the question. The best alternative to solving the algebraic loops issue is to use memory blocks or a transport delay. Figure 6 illustrates this usage. Upon examination of the structure in figure 6 it is clear that computation of the translational equations contained within the “translation” block cannot occur until the rotational block has issued its output. The memory block serves as a means to input a zero condition until valid data begins reporting from the rotation block.

Figure 6: Vehicle Dynamics Block
Figure 6 shows the Simulink circuit structure used to solve the differential equations for the dynamics of motion. The output of this block is the state vector for the missile; u, v, w, p, q, and r. The circuit structure in figure 7 shows the calculation of the critical support data discussed in chapter 2 such as the quaternion state vector, angle of attack and sideslip, direction cosine and inertial attitude.

The mathematical model contained within the vehicle dynamics block in figure 6 is an accurate representation of the behavior of a missile airframe when external stimuli are applied such as a gravity field, thrust and aerodynamic forces and moments. The result is acceleration and velocity of the airframe which behave nonlinearly. It is the behavior of the dynamic model that the linearizing equations seek to cancel out in the dynamic inversion process. The development of this unique stabilization and control structure will begin in chapter 7, however,
one final issue must be addressed before finally getting to the heart of this topic; the addition of actuator dynamics to the closed loop system must be considered since their presence can have significant impact on the closed loop. We begin this discussion next in chapter 6.
CHAPTER 6
ACTUATOR DYNAMICS

All too often, simulation studies conveniently ignore actuator dynamics and the profound
effects they can have on control systems, especially a nonlinear one. Reference [1] states that the
nonlinear effects contributed to the closed loop system by aerodynamic surface actuators can
have a significantly destabilizing effect on nonlinear control designs. It makes no sense to go to
the tremendous lengths of modeling a missile in 6 degrees of freedom and then disregard the
presence of actuator dynamics. Actuator dynamics are a very real part of any real life missile
control system and to simply ignore their presence would be negligent. Entire theses can be
written on the effects of actuators on closed loop control systems; however, this is not the
intended scope of this project. Several actuator designs should be presented and the performance
impacts on the nonlinear control design should be noted.

Actuator Types

The main issue to consider when actuator dynamics are included in any simulation is where
exactly these added dynamics place the closed loop poles of the total system: that is, the
combination of control law, actuator dynamics and plant. Different actuator designs added to a
system can force the closed loop poles into an unstable region causing the total system to fail. At
the same time, a different actuator model can change pole positioning while remaining stable.
There are essentially three different types of actuator models which can be used in simulations.
The first is a simple first order lag (6.1). Typically the addition of a first order lag to the closed
loop system will not force pole positions into an unstable region and these models therefore make a great place to start with the controller design. A first order actuator model was used throughout the development of this control design. In the end, once testing and evaluation of the system was complete, a far more realistic nonlinear second order actuator model was applied and the performance contributions to the closed loop were analyzed as well. The results will be given in chapter 9.

The second type of actuator model is a second order lag (equation 6.2) and forms the basis for the second order nonlinear lag (the third type) seen in figure 9. It is primarily the nonlinear effects posed by deflection rate limitation and travel saturation that impact the closed loop the greatest amount. The second order lag itself without the nonlinear characteristics can push the closed loop poles into the unstable region. Add the nonlinear characteristics and the control design falls apart.

Reference [1] proposes a rather unique approach to dealing with second order nonlinear models. It is proposed that a secondary controller situated between the autopilot output and the actuator input be implemented to not only cancel out the nonlinear characteristics of saturation and rate deflection limitation but to also provide added gains to prevent the closed loop poles from being forced unstable. The design for this project will implement a dynamic inversion controller at the input of each actuator. Since there are 4 actuators (4 fins total) there will be 4 dynamic inversion controllers dedicated solely to elimination of nonlinear characteristics produced by the actuators.

At this point we begin looking at the mathematical structure of each of the actuator models. The linear first order actuator model currently in use within the simulation structure is shown in
(6.1). It is simply,

\[ \dot{\delta} = 150(\delta_c - \delta) \]  

(6.1)

The term \( \delta_c \) represents the autopilot command to the actuator and \( \delta \) is the actuator output with 150 being the actuator’s natural frequency. Current simulation results prove the two time scale control system remains stable in the presence of first order lags (such as that in equation 6.1) in the closed loop. Each fin is controlled by one of these actuator models; there are 4 total.

**Second Order Nonlinear Actuator with Dynamic Inversion Compensator**

The basic nonlinear actuator model is built directly from the linear second order model. They are essentially the same with the exception that one has rate limitation and travel deflection. Equation (6.2) is a basic second order linear model which the nonlinear model will be built from. Before converting the linear model to a nonlinear one we must first prepare the linear model and take a look at some of the mathematical quantities involved. Typical values for natural frequency and damping ratio are \( \omega_n \) and \( \varsigma \) respectively. \( \delta_c \) and \( \delta \) variables maintain the same representation as in equation 6.1

\[ \frac{\ddot{\delta}}{\delta_c} = \frac{\omega_n^2}{s^2 + 2\varsigma \omega_n s + \omega_n^2} \]  

(6.2)

The main objective is to convert the transfer function to state space format which makes implementation in Simulink easier. First it is necessary to convert the transfer function into differential equation form,

\[ \delta(S) \left[ s^2 + 2\varsigma \omega_n s + \omega_n^2 \right] = \omega_n^2 \delta_c(S) \]  

(6.3)
\[ \ddot{\delta} + 2\zeta \omega_n \dot{\delta} + \omega_n^2 \delta = \omega_n^2 \delta_c \]  

(6.4)

At this point we begin the conversion to state space format by assigning phase variables as follows,

\begin{align*}
    x_1 &= \delta \\
    x_2 &= \dot{\delta} \\
    \dot{x}_1 &= \dot{\delta} \\
    \dot{x}_2 &= \ddot{\delta}
\end{align*}

(6.5)

After substituting the phase variables in 6.5 into 6.4, we arrive at the second order linear actuator model in 6.6. Equation 6.6 is still a linear representation of a second order actuator. The next step is to transform 6.6 into a simulation diagram which is a suitable form to implement in Matlab and Simulink.

\begin{align*}
    \dot{x}_1 &= x_2 \\
    \dot{x}_2 &= -2\zeta \omega_n x_2 - \omega_n^2 x_1 + \omega_n^2 \delta_c
\end{align*}

(6.6)

Equation 6.6 leads us to the following simulation diagram in figure 8 for the linear version of our second order actuator dynamics.

![Figure 8: Second Order Linear Actuator Structure](image)

For the simulation in figure 8, In 1 is the deflection command \( \delta_c \) from the autopilot. Out 1 is the deflection command \( \delta \) to the airframe, Gain 2 is the square of the actuators natural
frequency $\omega_n^2$ and Gain 1 is the damping term $2\zeta\omega$. Careful analysis of equation 6.4 shows the actuator velocity is $x_2 = \dot{\delta}$ which is the output of integrator 1; therefore, the solution of the state vector component $x_2$ yields the rate of deflection.

**Creating the Nonlinear Actuator Model**

In order to capture the nonlinear effects of a true actuator and analyze the potential impact they may have on autopilot designs, we begin by modifying the above simulation to limit the rate of deflection and also the range of travel. By doing so, we can accurately evaluate the real life impact actuators have on control performance. All actuators have a certain amount of linear travel until they reach a limit and saturate. The most important concept to consider when modeling actuator performance is limits on rate of deflection. Even the fastest actuating systems have a maximum speed at which they can execute a command, so in this case, we can think of an actuator as a type of mechanical low pass filter dampening out a rapidly changing autopilot signal. The next simulation shown in figure 6.2 captures these nonlinear characteristics. So, in essence what we have done is take a linear second order model, convert it to state space and then implement the design in Simulink with all of the nonlinearities present. Figure 6.2 shows the final nonlinear second order actuator model.
An important point can be made at this time regarding the modeling of deflection rate limits. If a simulation diagram is not used and a purely mathematical model is to be constructed, hyperbolic tangent functions are good ways to model this highly nonlinear term.

Figure 9 shows all of the necessary modifications that must be made to the linear model in figure 8 in order to attain the nonlinear version of a second order actuator. Since it is safe to say the plant has been modeled to acceptable levels of fidelity, we turn our attention to the design of a dynamic inversion controller. The nonlinear dynamic states in figure 9 can have nasty transient characteristics especially when driven by a rapidly changing autopilot signal. The entire basis for designing a controller for each actuator in the missile system is very similar to the autopilot design concept in chapter 7, to cancel all nonlinear dynamics online and control the plants remaining dynamics with linear controllers. However, remember that the scope of actuator controller design in this respect is to shift the poles of the missiles closed loop back into the stable region since the second order actuator pole contribution caused the problem to begin with.
Figure 10 shows the block diagram for one actuator/controller pair and the respective inputs and outputs. We shall now derive the linearizing function for the dynamic inversion process and apply a linear PID control law.

We repeat part of equation set 6.6 here again but all that is needed to develop the linearizing equation for the actuator plant is the expression for $\dot{x}_2$ given by 6.7. We assign a pseudo control variable to $\dot{x}_2$ in 6.8.

\[ \dot{x}_2 = -2\zeta\omega_n x_2 - \omega_n^2 x_1 + \omega_n^2 \delta_c \]  
(6.7)

\[ \hat{P} = \dot{x}_2 \]  
(6.8)

Next, we solve for $\delta_c$ yielding,

\[ \frac{\hat{P} + 2\zeta\omega_n x_2 + \omega_n^2 x_1}{\omega_n^2} = \delta_c \]  
(6.9)

Equation 6.9 is the final form of the linearizing control law to be used for actuator control. The pseudo control variable will be replaced by a PID linear controller in the actual implementation. Figures 11 and 12 show the Simulink diagrams that implement the control law in equation 6.9. In figure 10, the block labeled DYNAMIC INVERSION CONTROLLER...
houses both of the simulation diagrams shown in figures 11 and 12 and in fact, figure 12 is contained within the block labeled PID LINEAR FEED FORWARD CONTROL. This sort of Pandora’s Box approach to simulation design seems confusing at first but is truly a great way to model dynamic systems. To help understand the structure more, $\hat{P}$ in equation 6.9 is shown in figure 12. The rest of the control law in equation 6.9 is implemented in figure 11 and both reside in the left-most block shown in figure 10.

Figure 11: Linearizing Control Law

Figure 12: Proportional Plus Integral Plus Derivative Controller
We have now arrived at our nonlinear second order actuator model but our mathematical development for the overall actuator structure is still not complete. Regardless of which actuator type is used within the simulation structure, linear or nonlinear, a circuit structure must be developed to drive 4 fin actuators with only three autopilot signals; roll, pitch and yaw plus a feedback network to the autopilots must be provided. Upon inspection of the actuator pod located at the rear of the vehicle it is clear that each of the 3 autopilot (APL) signals from the roll, pitch and yaw APL’s are combined, applied to the actuators and then recovered once again at the output of the actuators, at which point the fully separated signals are applied to the airframe aerodynamics block. The problem that arises is any internal actuator states used as feedback to APL’s contain components from all three APL commands. This is true for both linear and nonlinear actuator designs. For example, upon inspection of the actuator system below it is apparent that the input to the actuator of fin 1 is a mixed combination of roll, pitch and yaw commands or,

\[ \delta_{\text{fin1_input}} = \delta_{\text{pitch_command}} - \delta_{\text{yaw_command}} - \delta_{\text{roll_command}} \]  

(6.9)

Therefore if states \( x_1 \) and or \( x_2 \) are used as feedback to the APL’s the design runs the risk of delivering erroneous data corrupted with additional signals that should not be appearing in the calculations. For example, the design of the pitch APL’s slow and fast time scale equations must receive clean feedback states containing input from pitch dynamics only. It is imperative that any actuator feedback states required by the autopilots prepare the feedback signals accordingly. If a linear actuating system is used, such as the one implemented within the blocks shown in figure 13, output feedback signals are all that is necessary to satisfy the first order actuator model of
equation 6.1 used within the autopilots linearizing function. If a second order nonlinear actuator model is used, special circuits must be utilized to filter out any unwanted command signals.

Figure 13 shows the overall actuator structure employed within this simulation structure. It is valid for both linear and nonlinear actuator systems. The feedback signal from the actuators to the autopilots for the linear first order version can be seen labeled as AF at the bottom right of figure 13.

![Actuator Network](image)

**Figure 13: Actuator Network**

At the left side of the circuit diagram, the autopilot command inputs are shown, that is $\delta_c$ for the pitch, yaw and roll channels. At the right side of the circuit diagram, the pitch, yaw and roll signals are recovered and applied to the airframe mathematical block.

This chapter completes the comprehensive build up to the most important chapter within this thesis; the two time scale nonlinear stabilization and control autopilot. The first 6 chapters have laid the necessary groundwork to implement and test this design. The information presented thus far should provide the reader with basic understanding of the dynamics, aerodynamics and all of the associated sub-systems required for an autopilot design.
CHAPTER 7
NONLINEAR AUTOPILOT

The derivation of nonlinear control systems for missiles is tedious at best. The mathematics involved is extremely complex and difficult to derive. Most papers dedicated to nonlinear missile controllers are done in three degrees of freedom for obvious reasons; the mathematics is far simpler. This thesis delves deep into the mathematical construction of a full 6 degree of freedom nonlinear controller for a surface to air interceptor missile. Many months were spent deriving and re-deriving the mathematical structure of this type of controller. To briefly re-cap chapters 3 and 4, the basis for this autopilot design is to break the missile dynamics into two time scales and force the separation between the two time scale entities. Within the dynamic model, there exists a fast time scale dynamic (rotational equations) and slow time scale dynamic (translational equations). Once the separations of time scales are enforced, they are used to form the linearizing equations needed to cancel all known dynamics of the plant. A pseudo control variable is then assigned to the remaining dynamics which allows the implementation of a standard linear control law. Developing a linearizing equation is the key to successful implementation of dynamic inversion because it is this equation that is fed forward into the plant and essentially removes the known non-linear dynamics. The dynamics that remain can be controlled by PID linear controllers. The linearizing equation is essentially a mirror image of the plant dynamics whereas all of the required mathematical inputs to the linearizing equation are calculated online. Online calculation of known non-linear dynamics requires tremendous computing power in a fast, real time environment.
When considering the nature of the slow time scale, it would seem evident that a PI controller may be perfect for the job because of the tracking ability on the slow acceleration signal the integral portion of the control law can provide. The effects of dynamic pressure, especially during the non minimum phase condition, induce a total loss of control for the PI control law. For these dynamics, a PID control law is a far better choice.

**Effects of Rapid Dynamic Pressure**

During the progression of this research document, simulation models of the missile airframe reached higher and higher levels of fidelity to ensure the most accurate outcome possible. During tuning of the autopilots in earlier stages, dynamic pressure on the airframe was held constant until atmospheric models could be developed which would allow for a variable dynamic pressure on the airframe. Once the atmosphere was modeled and incorporated into the simulation, profound impacts on the stability of the control system, especially the slow time scale were observed. The large magnitude and rapidly changing nature of dynamic pressure introduced new issues to contend with that presented a “make or break” moment for the validity of the two-time-scale control approach. The effects were so profound that complete destabilization of the vehicle resulted when the guidance processor began commanding elevated acceleration signals to the autopilots. Even under low acceleration commands the transient phase in the initial moments of launch was quite oscillatory. The original idea of using a PI control law for tracking the slow acceleration error was called into question and a more robust linear control law, such as PID, was sought. The entire theory of the two time scale approach for overcoming non minimum phase relies on the existence of a clear and distinct separation of time scales where one set of dynamics is actually much faster than the other and in the case of a missile this does in fact exist. The fast
time scale is able to stabilize the rotational rates before the non minimum phase within the slow
time scale becomes a serious threat to vehicle stability. Once the destabilizing effects of dynamic
pressure were observed two questions arose: 1) have we in fact destroyed the notion of slow and
fast dynamics by introducing a rapidly changing dynamic pressure and 2) is the two time scale
control approach even valid at this point? The answer to each of the questions is NO and YES,
respectively. It is important to note that dynamic pressure not only enters into the translational
acceleration of the vehicle but also into the rotational moments as well. So, in fact, each of the
separate time scales is affected in the exact same manner and no destruction of the slow/fast
relationship occurs. In other words, each of the time scales is scaled in an equally large manner.
It became clear, however, that as the dynamic pressure on the airframe increases to very large
values, such as the vehicle approaching Mach 5, the closed loop poles can move into the unstable
region. This suggests design limitations on the airframe itself and aerodynamic designs better
suited for Mach 5 speeds are required. Rising dynamic pressure had an effect even at lower
speeds in the seconds just after launch. Oscillations, although damped, during post-launch
conditions became unmanageable as acceleration commands from the guidance processor grew
larger. This reinforced the idea of adding error rate control to the slow time scale; that is, the
choice of PID control over PI as previously mentioned. One additional problem, however, was
inherent in the design itself. In order to account for the change in dynamic pressure, the autopilot
design required data representing this condition and a mathematical expression was developed. A
simple mistake was made in the mathematical expression of the change in dynamic pressure and
this caused some of the initial performance problems observed early in the testing phase. In the
end, a numerical differentiation routine with first order low pass filtering was utilized to provide
proper data to the dynamic inversion process. Transient response problems still existed though.
Through extensive testing, it was determined that autopilot gains became even more critical than
before and one gain in particular held the key to stabilizing the transient response. That gain is
the derivative gain setting for the fast time scale. This gain setting has a major impact on both the
slow and fast time scale and should be the first gain set when making initial test runs on this type
of autopilot design because no acceleration control can take place until rate stabilization is
achieved.

**Slow Time Scale Linearizing Equation and Controller-
Controlling Normal Acceleration**

The slow time scale dynamics are built upon the normal (and side in the case of yaw autopilot)
acceleration equations and comprise the slow time scale. Remember, only in cases where a clear
separation between time scales exists can this approach be used. Fortunately, missile dynamic
equations 2.1 through 2.6 exhibit this clear separation of time scales even though the dynamic
equations are fully coupled through coriolis effects. In this report, we will derive the linearizing
equations for the fast and slow time scale dynamics for the pitch/normal plane. The derivation of
the linearizing functions for the yaw channel, even though very similar to the pitch channel will
comprise the second half of this chapter.

For the slow time scale, we start with the normal acceleration equation and begin designing
the linearizing equation by taking the time derivative of 7.1. This allows us to accommodate the
input of actuator dynamics for either linear first order or nonlinear second order models; both can
be used in this simulation structure to evaluate different levels of performance provided by each
type of actuating system. Once the time derivative is taken and the appropriate substitutions are
made, we solve for q, which becomes the pitch rate command of the vehicles airframe. By solving for q, and developing an expression for the pitch rate command, we are then in a position to use this command as input to the fast time scale controller which ultimately satisfies the rate stabilization requirement. The overall control structure should be clear at this point. Pitch rate command q is developed from the slow time scale dynamics and is part of the rate stabilizing structure of the overall control system. The pitch rate command is then fed-forward into the linear control law for the fast time scale where a deflection command for the normal acceleration plane is calculated. This completes the objective of stabilizing the body rates and controlling acceleration; the two fundamental aspects of missile/rocket control. Keep in mind that two linearizing equations must be developed for the pitch/normal plane; one for each of the time scales. We now begin the painstaking task of deriving the linearizing equation for the slow time scale/normal acceleration dynamics. The dynamic equation for the normal acceleration in the pitch plane is displayed in equation 7.1. Keep in mind all derivations are conducted in body coordinates.

\[
\begin{align*}
\dot{a}_N &= \ddot{w} = qu - pv + \frac{F_{wp3}}{m} + t_{33}g \\
\end{align*}
\]  

(7.1)

Taking the time derivative yields,

\[
\begin{align*}
\dot{a}_N &= \dot{\ddot{w}} = \dot{qu} + \dot{pv} + \left(\frac{F_{wp3}}{m}\right) + \left(t_{33}\right)g \\
\end{align*}
\]  

(7.2)

The following equations will be substituted into equation 7.2 but first, special attention must be given to the normal force components of equation 7.2. Also, refer to chapter 2 for equation 7.7 derivation.
\[ \dot{u} = rv - qw + \frac{F_{wp1}}{m} + t_{13g} \]  
(7.3)

\[ \dot{q} = I^{-1}_z \left( (I_3 - I_1) pr + I^{-1}_z m_{g2} \right) \]  
(7.4)

\[ \dot{v} = pw - ru + \frac{F_{wp2}}{m} + t_{23g} \]  
(7.5)

\[ \dot{p} = I^{-1}_1 m_{g1} \]  
(7.6)

\[ \dot{\alpha}_3 = -2pq_\alpha q_1 - 2qq_\alpha q_2 + 2qq_\alpha q_3 - 2pq_\alpha q_3 \]  
(7.7)

When deriving the expression for the time rate of change of normal force, \( \dot{F}_{wp3} \) in equation 7.2, the dynamic pressure incident on the vehicle airframe changes too rapidly to ignore, as discussed previously. Since this particular vehicle is of high speed, long duration type flight, the dynamic pressure can rise to great levels at a rapid pace. In addition to this, once the area of maximum atmospheric dynamic pressure is passed, the dynamic pressure on the vehicle can begin to drop rapidly as well. Therefore, the time rate of change of dynamic pressure must be accounted for in the linearizing equation for the slow time scale dynamics. This results in the final expression for the time rate of change of airframe normal force.

\[ \dot{F}_{wp3} = SC_{Xwp} \left( \bar{q} \ddot{\alpha} + \alpha \ddot{q} \right) + SC_{Xwp} \left( \bar{q} \dot{\delta} + \delta \dot{q} \right) \]  
(7.8)

While deriving this equation is straight forward, special attention must be given to the coordinate transforms between aerodynamic axes and body axes. Proper implementation of these equations relies solely on the correct application of the associated coordinate transforms. The transforms are very simple but applying them properly can be quite confusing. Figure 7.1 illustrates the important differences that exist between body axes and aerodynamic axes; the two coordinate systems of primary concern. Early in the development of this project, modeling errors between
the two coordinate systems were made resulting in erroneous simulation results and difficult to trace errors that plagued the closed loop system. Although the differences between the two coordinate systems are slight, proper performance of the overall closed loop system depends on the proper transformation between body and aerodynamic axes. It is also extremely important to note that the calculation of angle of attack will result in a 90 degree difference depending on which coordinate system it is calculated in. Before any substitutions are made, transformations between body axes and aerodynamic axes will be established. This extra time is necessary to ensure these crucial transformations enter the autopilot linearizing equations properly. If they do not, the closed loop performance will be invalid. First, the differences between body axes and aerodynamic axes discussed in chapter 4 will be restated here.

Upon examination of figure 14, it is clear that the -3B and +3A axes coincide but are in opposite directions. The same can be said for the +1B and -1A axes. Depending on the coordinate system used, calculation of the angle of attack can be 90 degrees out of phase if the process is not
properly thought out. The underlying issue here is, within the aerodynamics block the force coefficients are calculated in aerodynamic axes and then later converted to body axes before being delivered to the vehicle dynamics block and the autopilot linearizing functions. If the angle of attack, which is used in the force coefficient calculation, has been calculated in body axes and then used to determine normal force coefficient, the data will be incorrect and other crucial data outputs of the simulation such as the normal force vectors will be oriented in the wrong direction. This tiny mistake results in giant headaches during testing. The scalar transformations between coordinate systems are given below in 7.9.

\[
A \quad N \quad A \quad a p \quad B \quad a p \quad S C \quad q \quad F \quad F
- = - = 33 \quad (7.9)
\]

Where \( C_N \) is expressed as,

\[
C_N = \left(C_{Na} \left[ \alpha \right]^4 + C_{Na} \left[ \delta \right]^4 \right) \quad (7.10)
\]

In this simulation structure, the autopilot output and angle of attack are produced in body coordinates. As stated before, aerodynamic coefficient terms must be calculated in aerodynamic coordinates; therefore, additional transformations are given for the conversion of angle of attack and fin deflection from aerodynamic axes to body axes.

\[
\left[ \alpha \right]^4 = \left[ - \alpha \right]^b \quad (7.11)
\]
\[
\left[ \delta \right]^d = \left[ - \delta \right]^b \quad (7.12)
\]

The expression for the time rate of change of normal airframe force, \( \dot{F}_{ap3} \), is,

\[
\dot{F}_{ap3} = SC_{Na} \left( \ddot{\alpha} \dot{\alpha} + \alpha \ddot{\alpha} \right) + SC_{Na} \left( \ddot{\delta} \dot{\delta} + \delta \ddot{\delta} \right) \quad (7.13)
\]

Next, we express the force equations with the proper axis transformations included so we can arrive at the correct form for implementation in the simulation. Since the coefficient equations must be calculated using aerodynamic axes but the simulation structure provides \( \alpha \) and \( \delta \) in body
axes, those transforms are included as well. The final form of the time rate of change of normal force taking into account all of the proper transformations is given in equation 7.16. The first order linear actuator model in 7.17 must also enter into the force equation of 7.16.

\[
\begin{align*}
[F_{ap3}]^a &= \left[SC_{Na} (q \dot{q} \dot{\alpha} + \{\alpha\}^a \dot{q}) + SC_{Na} \left(\dot{\delta}^a \dot{q} + \{\delta\}^a \dot{q}\right)\right]^a \\
[F_{ap3}]^b &= \left[SC_{Na} (q \dot{q} (-\dot{\alpha}) + \{-\dot{\alpha}\}^b \dot{q}) + SC_{Na} \left(\dot{\delta}^b \dot{q} + \{-\dot{\delta}\}^b \dot{q}\right)\right]^b \\
[F_{ap3}]^c &= SC_{Na} (q \dot{q} \dot{\alpha}) + \{\alpha\}^c \dot{q} + SC_{Na} \left(\dot{\delta}^c \dot{q} + \{\delta\}^c \dot{q}\right)
\end{align*}
\]

\[
[\delta]^c = 150(\delta_C - \delta)
\]

Now that proper care has been taken to correctly model the normal force, we can begin building the autopilot linearizing equations. Substitution of equations 7.3 through 7.7 and 7.16 through 7.17 into equation 7.2 result in the base linearizing equation for the slow time scale.

\[
\begin{align*}
\dot{q} = \dot{w} &= q \left(rv - qw + \frac{F_{ap1}}{m} + t_{15}g\right) + u \left(I_2^{-1} ((I_3 - I_1) pr + m_{B2}) \right) - \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \} \\
\delta = g(-2pq_0q_1 - 2qq_0q_2 + 2qq_0q_3 - 2pq_2q_3)
\end{align*}
\]

$I_1, I_2, I_3$ are the principal moments of inertia in the axial, side and normal directions respectively.

$p, q, r$ is the vehicle roll rate, pitch rate and yaw rate respectively.
\( u, v, w \) is the vehicle’s inertial velocity in body coordinates for the axial, side and normal directions.

\( q_0, q_1, q_2, q_3 \) are the components of the quaternion state vector.

\( F_{ap1}, F_{ap3} \) are the aerodynamic forces incident on the vehicle airframe in the axial and normal directions.

\( t_{33}, \dot{t}_{33}, t_{13}, t_{23} \) are the gravitational components in the normal direction, rate of gravitational change in the normal direction, gravitational components in the axial and side directions.

\( \alpha, \dot{\alpha} \) is the angle of attack rate and deflection rate.

At this point it must be noted that it is highly important all mathematical expressions containing the pitch rate be exposed in the equations, for it is this value we are trying to solve for to reach the proper expression for the pitch rate command. For instance, we could have simply substituted into the first equation for \( \dot{w} \) the numerical values for \( \dot{u}, \dot{q}, \dot{v}, \) and \( \dot{p} \) streaming in from the translational and rotational differential equation blocks and solve for the one \( q \) in the original equation. This would be a grave error because contained within those rates of change are pitch rate values (\( \dot{q} \)) that must be solved for. Therefore, it is of the utmost importance that all values of \( q \) be exposed and are observable within the equations.

The linearizing equation, in body coordinates, for the slow time scale is,

\[
[\ddot{\mathbf{x}}_N] = \dot{\mathbf{w}} = q\left((rv - qw + \frac{F_{ap1}}{m} + t_{13}g) + u\left(t_2^{-1}(0 - \mathbf{I})pr + m_{g2}\right)\right) - \cdots \quad (7.22)
\]
\[
\begin{align*}
\tau_{\text{B}} & = \left( p\left( p_{w} - ru + \frac{F_{\text{ap}}}{m} + t_{23}g \right) + vl_{I_{1}}^{-1}m_{b1} \right) + \ldots \quad (7.23) \\
\ldots + m^{-1}SC_{N_{a}}\left( q[\dot{\alpha}]^{q} + [\alpha]^{q} \dddot{q} \right) + m^{-1}SC_{N_{a}}\left( q[\dot{\delta}]^{q} + [\delta]^{q} \dddot{q} \right) + \ldots \quad (7.24) \\
\ldots + g\left( -2pq_{0}q_{1} - 2qq_{0}q_{2} + 2qq_{1}q_{3} - 2pq_{2}q_{3} \right) \quad (7.25)
\end{align*}
\]

Expansion of terms yields,
\[
\begin{align*}
\left[ \dot{\alpha} \right]^{q} & = \dot{w} = qrv - q^{2}w + q\frac{F_{\text{ap}}}{m} + qt_{13}g + I_{2}^{-1}(I_{3} - I_{1})wpr + I_{2}^{-1}um_{g2} - p^{2}w + pru - p\frac{F_{\text{ap}}}{m} \ldots \quad (7.26) \\
\ldots - pt_{23}g - pvl_{I_{1}}^{-1}m_{b1} + m^{-1}SC_{N_{a}}\left( q[\dot{\alpha}]^{q} \right) + m^{-1}SC_{N_{a}}\left[ \alpha \right]^{q} \dddot{q} \ldots \quad (7.27) \\
\ldots + m^{-1}SC_{N_{a}}\left[ \dot{\delta} \right]^{q} + m^{-1}SC_{N_{a}}\left[ \delta \right]^{q} \dddot{q} - 2gpq_{0}q_{1} \ldots \quad (7.28) \\
\ldots - 2qq_{0}q_{2} + 2qq_{1}q_{3} - 2gpq_{2}q_{3} \quad (7.29)
\end{align*}
\]

We are quickly converging on the final form for this equation; however, we must first turn our attention to the time rate of change of angle of attack. Once again, this quantity can be computed in the kinematics block and fed directly into this equation, but it contains q values that must be exposed and ultimately solved for.

Cartesian angle of attack expressed in body coordinates is,
\[
\left[ \alpha \right]^{q} = \tan^{-1}\left( \frac{[w]^{q}}{[u]^{q}} \right) \quad (7.30)
\]

Where \( w \), the inertial velocity, is expressed in body coordinates in the normal direction and \( u \) is the inertial velocity expressed in body coordinates in the axial direction. For the rest of the angle of attack derivation the superscripts denoting body coordinates will be dropped. Finding the rate of change of the angle of attack is accomplished by taking the time derivative of equation 7.30 as follows,
\[
\dot{\alpha} = \frac{d}{dt} \left( \frac{w}{u} \right) \frac{1}{1 + \left( \frac{w}{u} \right)^2} \tag{7.31}
\]

We must next apply the quotient rule to the numerator which results in,

\[
\frac{uw - wu}{u^2} \tag{7.32}
\]

For a total angle of attack rate expression of,

\[
\dot{\alpha} = \frac{uw - wu}{u^2} = \frac{uw - wu}{u^2 + w^2} \tag{7.33}
\]

where

\[
\dot{u} = rv - qw + \frac{F_{ap1}}{m} + t_{13}g \tag{7.34}
\]

\[
\dot{w} = qu - pv + \frac{F_{ap2}}{m} + t_{33}g \tag{7.35}
\]

After making the substitutions of 7.34 and 7.35 into 7.33, we arrive at the final angle of attack rate equation suitable for substitution into \( \dot{\alpha}_N \),

\[
\left[ \dot{\alpha} \right]_{\theta} = \frac{qu^2 - pvu + \frac{F_{ap3}}{m} + ut_{33}g - wrv + qw^2 - w \frac{F_{ap1}}{m} - wt_{13}g}{u^2 + w^2} \tag{7.36}
\]
Next, we substitute equation 7.36 into 7.27 and continue the process which will eventually result in solving for \( q \), which becomes the pitch rate command \( q_c \).

\[
\dot{q}_v = \ddot{w} = qrv - q^2w + q\frac{F_{ap1}}{m} + qt_{13}g + I_2^{-1}(I_3 - I_1)upr + I_2^{-1}um_{g2} - p^2w + pru - p\frac{F_{ap2}}{m} - \ldots \quad (7.37)
\]

\[
\ldots - pt_{23}g - pvI_1^{-1}m_{b1} + \frac{\bar{q}S}{m}C_{Na} \left( \frac{qu^2 - pvu + u - \frac{F_{ap3}}{m} + ut_{33}g - wrv + qw^2 - w - \frac{F_{ap4}}{m} - wt_{13}g}{u^2 + w^2} \right) + \ldots \quad (7.38)
\]

\[
\ldots + m^{-1}SC_{Na} [\alpha]^B \ddot{\alpha} + m^{-1}SC_{Na} \ddot{\alpha} + m^{-1}SC_{Na}[\delta]^B \ddot{\delta} - \ldots \quad (7.39)
\]

\[
\ldots - 2gpq_0q_1 - 2gpq_0q_2 + 2gpq_1q_3 - 2gpq_2q_3 \quad (7.40)
\]

Now that all substitutions have been made, our next move, as mentioned before, is to assign a pseudo control variable and apply a linear control law to the problem. Keep in mind, all derivations up to this point are for the slow time scale only. We must still derive the fast time scale mathematics. In addition, both derivations for the fast and slow time scales are for the pitch channel dynamics only.

What we have finally arrived at is a linearizing function that represents the required pitch rate that must be executed and tracked in order to maintain rate stabilization of the vehicles pitch dynamics. This equation, along with the linear control law will be fed forward into the second half of the control system which consists of the linearizing equation for the fast time scale. So, we have essentially designed a mathematical function based on airframe dynamics that allows us to cancel all known nonlinear terms contained in the plant. It is important to note that each of the time scales will not only have a linearizing function, but also a linear control law associated with
each of them. Since a linearizing equation has been developed, we turn our focus to incorporating a linear control law into the slow time scale linearizing equation.

We assign pseudo control variable \( \hat{P} \) to the right hand side of the massive equation that spans 7.37 through 7.40.

\[
\dot{a}_N = \hat{P} \tag{7.41}
\]

At this point, we can assign a Proportional plus Integral plus Derivative (PID) control law to the right hand side,

\[
\dot{P} = K_1 (a_{NC} - a_N) + K_2 \int_o^t (a_{NC} - a_N) dt + K_3 \frac{d}{dt} (a_{NC} - a_N) \tag{7.42}
\]

We will, however, keep \( \hat{P} \) in the equation and substitute it at the very end of the derivation. Now, we are at the point where we can begin to solve for \( q_c \). The input arguments to the PID control law will be discussed shortly. For now, we must distribute terms in order to extract \( q \) from the equations.

\[
\dot{a}_N = \ddot{w} = qrv - q^2 w + q \frac{F_{ap1}}{m} + q t_{13} g + I_2^{-1} (I_3 - I_1) upr + I_2^{-1} um_{b2} - p^2 w + pru - p \frac{F_{ap2}}{m} + 
\]

\[
..... + \frac{\bar{q} S}{m} C_{Na} \left( \frac{qu^2 + qw^2}{u^2 + w^2} \right) + \frac{\bar{q} S}{m} C_{Na} \left\{ \frac{-p vu + u \frac{F_{ap3}}{m} + ut_{33} g - wrv - w \frac{F_{ap1}}{m} - wt_{13} g}{u^2 + w^2} \right\} - 
\]

\[
..... - m^{-1} S C_{Na} \left[ \alpha \right]^\theta \ddot{\bar{q}} + m^{-1} S C_{Nd} \ddot{\bar{\delta}} + m^{-1} S C_{Na} \left[ \delta \right]^\theta \ddot{q} - 2 gpq_0 q_1 - 2 gpq_0 q_2 + 
\]

\[
..... + 2 gpq_1 q_3 - 2 gpq_2 q_3 - pt_{23} g - pv t_{1}^{-1} m_{gi} \tag{7.45}
\]

\[
..... + 2 gpq_1 q_3 - 2 gpq_2 q_3 - pt_{23} g - pv t_{1}^{-1} m_{gi} \tag{7.46}
\]
Since \( \dot{a}_N = \dot{P} \), we can express the following,

\[
\dot{P} = q v r - q^2 w + q \frac{F_{ap1}}{m} + q t_{13} g + I_2^{-1}(I_3 - I_1)u_{pr} + I_2^{-1}um_{\beta 2} - p^2 w + pru - p \frac{F_{ap2}}{m} + \ldots \tag{7.47}
\]

\[
\ldots + \frac{\bar{q}S}{m} C_{Na} \left( \frac{q u^2 + q v^2}{u^2 + w^2} \right) + \frac{\bar{q}S}{m} C_{Na} \left( - p v u + u \frac{F_{ap3}}{m} + u t_{33} g - w r v - w \frac{F_{ap1}}{m} - w t_{13} g \right) \tag{7.48}
\]

\[
\ldots - m^{-1}S_{Na} [\alpha]^q \dot{\bar{q}} + m^{-1}S_{Na} q \dot{\bar{q}} + m^{-1}S_{Na} [\delta]^q \ddot{\bar{q}} - 2gqq_0 q_1 - 2gqq_0 q_2 + \ldots \tag{7.49}
\]

\[
\ldots + 2gqq_4 q_3 - 2gqq_2 q_3 - pt_{23} g - pv l_{1}^{-1} m_{\beta 1} \tag{7.50}
\]

Next, we set the equation equal to zero and group \( q \) terms.

\[
0 = -\dot{P} + I_2^{-1}(I_3 - I_1)u_{pr} + I_2^{-1}um_{\beta 2} - p^2 w + pru - p \frac{F_{ap2}}{m} + \ldots \tag{7.51}
\]

\[
\ldots + \frac{\bar{q}S}{m} C_{Na} \left( - p v u + u \frac{F_{ap3}}{m} + u t_{33} g - w r v - w \frac{F_{ap1}}{m} - w t_{13} g \right) + \ldots \tag{7.52}
\]

\[
\ldots - m^{-1}S_{Na} [\alpha]^q \dot{\bar{q}} + m^{-1}S_{Na} q \dot{\bar{q}} + m^{-1}S_{Na} [\delta]^q \ddot{\bar{q}} - 2gqq_0 q_1 + \ldots \tag{7.53}
\]

\[
\ldots - \bar{q}SC_{Na} \delta - 2gqq_0 q_1 - 2gqq_2 q_3 - pt_{23} g - pv l_{1}^{-1} m_{\beta 1} + \ldots \tag{7.54}
\]

\[
\ldots + q \left( rv + \frac{F_{ap1}}{m} + \frac{\bar{q}S}{m} C_{Na} + t_{13} g - 2gq_0 q_2 + 2gq_1 q_3 \right) - q^2 w \tag{7.55}
\]

**Quadratic Slow Time Scale Form**

Unfortunately, as we can see from equation 7.55 a quadratic expression exists. To maintain the fidelity of the system, the quadratic formula will be utilized to solve for \( q \). Early in the development of this project, the \( q^2 w \) term was linearized which resulted in that term becoming zero. Solving for \( q \) was then a straight forward task. Much testing was completed without this
term, ultimately making the final form of the control law a bit simpler to deal with. However, it was determined that the performance of the system suffered to a degree and the quadratic form of equations 7.51 through 7.55 was implemented using the quadratic formula. Further testing was required to “tune in” the equations and several problems arose due to the quadratic implementation. There are typically two solutions to quadratic equations but which solution is appropriate for any given simulation time step within an autopilot? Testing has determined the proper quadratic solution to execute for the pitch rate command and it will be discussed shortly. Furthermore, once the above equation is placed in standard quadratic form, normal velocity now makes up the denominator of two major terms within the autopilot equations. This is obviously extremely problematic since the normal velocity can quite frequently pass through zero. Therefore, during final testing of the design, zero crossing detection was developed to prevent autopilot saturation as \( w \) approached zero from either the negative end or the positive end of the velocity field. It is clear that implementation of quadratic autopilot logic posed numerous coding challenges to overcome and great amounts of time were consumed tuning system performance. We continue completing the final form by presenting the standard quadratic equation form in 7.56, in terms of the pitch rate command,

\[
aq^2_c + bq_c + c = 0 \quad (7.56)
\]

The quadratic formula is then,

\[
q_c = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (7.57)
\]

Dividing the \( a \) term through the expression yields the final form of the autopilot slow time scale output signal,
\[ q_c^2 + \left( \frac{b}{a} \right) q_c + \left( \frac{c}{a} \right) = 0 \]  

(7.58)

\[ q_c = \frac{-\left( \frac{b}{a} \right) \pm \sqrt{\left( \frac{b}{a} \right)^2 - 4\left( \frac{c}{a} \right)}}{2} \]  

(7.59)

As discussed a moment ago, the question arose during initial implementation of the quadratic form as to which solution to use for the pitch rate command output. Testing of the design has shown that a difference between the two numerator terms must always occur. Since solutions to quadratic expressions occur in pairs, typically one solution may be out of line with a more desirable solution. In this case, for instance, one solution may yield .210 and the other solution may be 670.0. It is clear that the 670.0 solution to the quadratic expression is not reasonable since the solution represents a pitch rate command that the vehicle must track. A pitch rate command of 670.0 radians per second is ridiculous. Therefore, the need for the following switching logic within the autopilot structure is given in 7.60 and 7.61,

If \( \left( \frac{b}{a} \right) \) is negative, then we must have,

\[ q_c = \frac{-\left( \frac{b}{a} \right) - \sqrt{\left( \frac{b}{a} \right)^2 - 4\left( \frac{c}{a} \right)}}{2} \]  

(7.60)

If \( \left( \frac{b}{a} \right) \) is positive, then we must have,

\[ q_c = \frac{-\left( \frac{b}{a} \right) + \sqrt{\left( \frac{b}{a} \right)^2 - 4\left( \frac{c}{a} \right)}}{2} \]  

(7.61)

If the above switching logic is maintained, the autopilot will generate the proper output signal without fail. One major issue with this quadratic format, which was mentioned before, is a likely division by zero condition within the quadratic structure. Careful software control must be maintained in order avoid this potentially catastrophic problem. Failure to properly implement a
fix for this problem will result in autopilot signal saturation and all loss of vehicle control with no way to recover performance.

In order to place the linearizing equation into a form suitable for implementation in the quadratic formula, equations 7.51 through 7.55 are broken up into suitable components,

\[ c = -\dot{P} + I_2^{-1}(I_3 - I_1)upr + I_2^{-1}um_{g2} - p^2w + pru - p\frac{F_{ap2}}{m} + \ldots \]  
(7.62)

\[ \ldots + \frac{\bar{q}S}{m}C_{\mathrm{Na}}\left( -pvu + u\frac{F_{ap3}}{m} + ut_{33}g - wrv - w\frac{F_{ap1}}{m} - wt_{13}g \right) + \ldots \]  
(7.63)

\[ \ldots - m^{-1}SC_{\mathrm{Na}}[\alpha]q_{\hat{q}} - m^{-1}SC_{\mathrm{Na}}q_{\hat{\delta}} + m^{-1}SC_{\mathrm{Na}}[\delta]q_{\hat{q}} - 2gpq_0q_{11} + \ldots \]  
(7.64)

\[ \ldots - \bar{q}SC_{\mathrm{Na}}\delta - 2gpq_0q_{11} - 2gpq_0q_{13} - pt_{23}g - pv\tau_1^{-1}m_{g1} \]  
(7.65)

\[ b = \left( rv + \frac{F_{ap1}}{m} + \frac{\bar{q}S}{m}C_{\mathrm{Na}} + t_{13}g - 2gq_0q_{22} + 2gq_0q_{13} \right) \]  
(7.66)

\[ a = -w \]  
(7.67)

Where \( \dot{P} \) in 7.48 is the linear PI control law,

\[ \dot{P} = K_1(a_{NC} - a_N) + K_2 \int_0^t(a_{NC} - a_N)dt + K_3 \frac{d}{dt}(a_{NC} - a_N) \]  
(7.68)

We have now reached the final form of the slow time scale derivation. Let us take a moment to look closer at the PID control law. The proportional part of the control law multiplies the proportional gain, \( K_1 \), by the error that exists between the commanded acceleration and the vehicle’s actual acceleration. The commanded accelerations are produced by the guidance computers and an appropriate guidance law. The guidance computers produce acceleration commands for the pitch and yaw planes that act as the “steering signals” for the interceptor. If
calculated properly, the acceleration commands will place the interceptor vehicle onto a collision triangle with the target. The second part of the control law, the integral portion, attempts to produce zero tracking error of the guidance signals. The derivative part of the control provides system damping due to a rapidly changing error signal. The derivative control can be implemented in two ways; 1) a numerical differentiation of the error signal can be done; or, 2) the rate of change of vehicle acceleration can be calculated assuming the guidance command signals change slowly enough to be considered constant. The drawback to the first choice is Simulink’s derivative blocks are numerical differentiation routines and react to noise or erratic input signals in a very undesirable manner and demonstrated extremely destabilizing effects on the controls. The second choice was ultimately implemented, which involved coding a routine that calculates equation 7.2, (which is in fact the rate of change of vehicle acceleration) and injecting those calculations into the derivative portion of the control law. It was assumed that the guidance acceleration commands varied slow enough to be considered constant, which left just the vehicle acceleration rate calculations as input to the derivative control. This provided the required transient rate damping in the slow time scale and proved to be far more reliable than a numerical differentiation to the error signal.

As stated before, the slow time scale derivation is only half of the total control system design. The fast time scale derivation is conducted in a similar manner to that of the slow time scale process; only this time, the starting equation used for the derivation process is the pitch rate dynamic equation and the end result will be an expression for \( \delta_c \), the autopilot command which is applied directly to the servo actuators that operate the rear control fins. A second linear control law is also used for the fast time scale system in which the pitch command \( \alpha \) becomes one of the
input arguments to the linear control law. By this time, it is probably quite apparent how mathematically intense a nonlinear control system design is for a missile.

**Fast Time Scale Linearizing Equation and Controller-Stabilizing Body Rate \( q \)**

The derivation begins with the pitch rate dynamic equation in 7.69. Refer to chapter 5, “Dynamic Model” for more information on this equation. As was done for the slow time scale, we begin the development of a linearizing equation for the fast time scale by taking the time derivative of 7.69 which allows actuator dynamics to be included in the final form. Equation 7.69 is actually the angular acceleration found in the pitch channel. We now calculate the time derivative and find the rate of change of angular acceleration in 7.70,

\[
\begin{align*}
\ddot{q} &= I_2^{-1}((I_3 - I_1)p \dot{r} + m_{\beta 2}) \\
\ddot{\dot{q}} &= I_2^{-1}((I_3 - I_1)(p \ddot{r} + \dot{p} \dot{r}) + \dot{m}_{\beta 2}) \\
\dddot{q} &= I_2^{-1}(I_3 - I_1)p \dddot{r} + I_2^{-1}(I_3 - I_1)\dddot{p} + I_2^{-1}\dddot{m}_{\beta 2}
\end{align*}
\]  

(7.69)  

(7.70)  

(7.71)

Where the following substitutions can be made,

\[
\begin{align*}
\dot{p} &= I_1^{-1}m_{\gamma 1} \\
\dot{r} &= I_3^{-1}((I_1 - I_2)pq + m_{\beta 3})
\end{align*}
\]  

(7.72)  

(7.73)

After substituting 7.72 and 7.73 into 7.70, we arrive at 7.74,

\[
\ddot{q} = I_2^{-1}(I_3 - I_1)p(I_3^{-1}((I_1 - I_2)pq + m_{\beta 3})) + I_2^{-1}(I_3 - I_1)(I_1^{-1}m_{\gamma 1})\dot{r} + I_2^{-1}\dddot{m}_{\beta 2}
\]  

(7.74)
In order to find the correct expression for $\dot{m}_{B2}$, we must once again consider the rate of change of dynamic pressure on the vehicle, therefore,

$$\dot{m}_{B2} = I_2^{-1}SC_{ma} (\tilde{q} \dot{\alpha} + \alpha \tilde{q}) + I_2^{-1}SC_{ma} (\tilde{q} \dot{\alpha} + \alpha \tilde{q})$$  \hspace{1cm} (7.75)

Where the first order actuator lag is,

$$\dot{\delta} = 150(\delta - \delta_c)$$  \hspace{1cm} (7.76)

Substituting 7.76 into 7.75 leads to the final form of the rate of change of pitch moment,

$$\dot{m}_{B2} = I_2^{-1}SC_{ma} (\tilde{q} \dot{\alpha} + \alpha \tilde{q}) + I_2^{-1}SC_{ma} (\tilde{q} (150 \delta_c - 150 \delta) + \delta \ddot{q})$$  \hspace{1cm} (7.77)

Expansion of terms in 7.77 leads to,

$$\dot{m}_{B2} = I_2^{-1}SC_{ma} \tilde{q} \dot{\alpha} + I_2^{-1}SC_{ma} \alpha \tilde{q} + 150 I_2^{-1}SC_{ma} \tilde{q} \delta - 150 I_2^{-1}SC_{ma} \dot{\delta} + I_2^{-1}SC_{ma} \delta \ddot{q}$$  \hspace{1cm} (7.78)

Equation 7.78 can then be substituted into 7.74 taking into consideration we have already accounted for the inverse inertial component. This final step leads us closer to the complete expression for the fast time scale command. After substitutions are made we arrive at,

$$\dot{q} = I_2^{-1} I_3^{-1} (I_3 - I_1)(I_1 - I_2) p^2 q + I_2^{-1} I_3^{-1} (I_3 - I_1)pm_{B3} + I_1^{-1} I_2^{-1} (I_3 - I_1) rm_{B1} + .......$$  \hspace{1cm} (7.79)

$$...... + I_2^{-1}SC_{ma} \tilde{q} \dot{\alpha} + I_2^{-1}SC_{ma} \alpha \tilde{q} + 150 I_2^{-1}SC_{ma} \tilde{q} \delta - 150 I_2^{-1}SC_{ma} \dot{\delta} + I_2^{-1}SC_{ma} \delta \ddot{q}$$  \hspace{1cm} (7.80)

$I_1, I_2, I_3$ are the principal moments of inertia in the axial, side and normal directions respectively.

$p, q, r$ is the vehicle roll rate, pitch rate and yaw rate respectively.
\( u, v, w \) is the vehicles inertial velocity in body coordinates for the axial, side and normal directions.

\( q_0, q_1, q_2, q_3 \) are the components of the quaternion state vector.

\( F_{ap1}, F_{ap3} \) are the aerodynamic forces incident on the vehicle airframe in the axial and normal directions.

\( t_{33}, \dot{t}_{33}, t_{13}, t_{23} \) are the gravitational components in the normal direction, rate of gravitational change in the normal direction, gravitational components in the axial and side directions.

\( \dot{\alpha}, \dot{\delta} \) is the angle of attack rate and deflection rate.

During all of these mathematical derivations it is imperative that all terms being solved for be revealed in the math. Take for instance the \( \dot{\alpha} \) term in equation 7.80. Since we are solving for \( \delta_c \), could any of these terms be hidden within \( \dot{\alpha} \)? \( \dot{\alpha} \), shown in 7.81, does not appear to contain any terms of immediate interest.

\[
\dot{\alpha} = \frac{qu^2 - puv + u \frac{F_{ap3}}{m} + ut_{33}g - wrv + qw^2 - w \frac{F_{ap1}}{m} - wt_{13}g}{u^2 + w^2}
\]  

(7.81)

Since none of the terms possess a time rate of change of \( \delta \) or \( \dot{\delta}_c \), we are safe to proceed and finish the derivation leaving \( \dot{\alpha} \) in its current state.

We follow the same process as was done for the slow time scale in that we assign a pseudo control variable \( \hat{P} \), assign a linear control law and then solve for \( \delta_c \),
\begin{equation}
\dot{P} = I_2^{-1}I_3^{-1}(I_3 - I_1)(I_1 - I_2)p^2q + I_2^{-1}I_3^{-1}(I_3 - I_1)pm_{B3} + I_1^{-1}I_2^{-1}(I_3 - I_1)rm_{B1} + \ldots \tag{7.82}
\end{equation}

\begin{equation}
\ldots + I_2^{-1}SLC_{m_a}\bar{q}\dot{\alpha} + I_2^{-1}SLC_{m_a}\alpha\ddot{q} + 150I_2^{-1}SLC_{m_0}\bar{\delta} + 150I_2^{-1}SLC_{m_0}\delta + I_2^{-1}SLC_{m_0}\delta\dot{\delta} \tag{7.83}
\end{equation}

We are now set to solve for \(\delta_c\) to finalize the output command expression for the fast time scale,

\begin{equation}
\delta_c = \frac{150I_2^{-1}SLC_{m_0}\bar{\delta}}{-I_2^{-1}SLC_{m_a}\bar{q}\dot{\alpha} - I_2^{-1}SLC_{m_a}\alpha\ddot{q} + 150I_2^{-1}SLC_{m_0}\delta - I_2^{-1}SLC_{m_0}\delta\dot{\delta}} \tag{7.84}
\end{equation}

Where \(\dot{P}\) equals,

\begin{equation}
\dot{P} = K_1(q_C - q) + K_2 \int (q_C - q)dt - K_3\dot{q} \tag{7.85}
\end{equation}

This is a Proportional plus Derivative linear control law; note the input argument to the Proportional component. It just happens to be the \(q_C\) command we derived for the slow time scale! For the Derivative part of the control law, \(\dot{q}\) equals,

\begin{equation}
\dot{q} = I_2^{-1}((I_3 - I_1)pr + m_{a_2}) \tag{7.86}
\end{equation}

We will not mathematically substitute \(\dot{q}\) into the PID control law since there are no deflection command variables \(\delta_c\) contained within \(\dot{q}\) we must solve for. This data will be delivered by the rotational differential equation block which was addressed in chapter 5, “Dynamic Model”.

**Implementing the Controls**

The equations 7.84, 7.85 and 7.60 through 7.68 comprise the final controller form to be implemented within the pitch channel autopilot. Although numerous equations are shown, \(\delta_c\)
forms the deflection command output for the pitch channel autopilot. $\hat{P}$ in equation 7.68 substitutes into $q_c$ at 7.51 and $q_c$ substitutes into $\delta_c$ at 7.85 forming one giant equation. Keep in mind, this control design is implemented only for the pitch channel, we must still derive a two time scale nonlinear controller for the yaw channel and then another for the roll channel. The roll channel autopilot will be addressed in chapter 8, Vehicle Roll Stabilization. The design method for stabilizing the roll axis is different from the pitch and yaw derivations.

Figure 15: Complete Autopilot Structure (Pitch Channel)

Figure 15 shows the complete form of the pitch channel autopilot comprised of slow and fast time scale controller dynamics. Moving from left to right in figure 15, the first block contains the linear control law for the slow time scale; the inputs to this block can be seen in the figure and are composed of guidance command signals, vehicle normal acceleration and vehicle acceleration rate. The output of the linear control law is fed forward into the next block labeled SLO_PRC, short for slow time scale pitch rate command. The dynamic inversion data vector is also injected into this block. The dynamic inversion data vector contains all of the current vehicle states and other required data such as dynamic pressure and dynamic pressure rate to facilitate the cancellation of all nonlinear dynamics. The output of SLO-PRC is the pitch rate command $q_c$.
given in equations 7.60 and or 7.61 depending on internal controller conditions. Command \( q_c \) then enters the next block named Fast Time Scale PID. As the name implies, this block contains the linear control law used to control the fast time scale dynamics. In this case, inputs to this block are of course \( q_c \), vehicle pitch rate and vehicle pitch acceleration, which are angular velocity and acceleration respectively. The final block in this control sequence is named FAS_PCMD which stands for fast time scale pitch command. Notice the dynamic inversion data vector is also injected into this block and serves the same purpose as it does for the slow time scale. The output of this block is \( \delta_c \), which comprises the fin deflection command delivered to the actuators.

This now completes the development of a pitch channel two time scale nonlinear controller. The second half of this chapter is dedicated to the development of an autopilot structure for the yaw channel. It is essentially the exact same process as for the pitch channel so the last half of this chapter will be a somewhat briefer.

**Slow Time Scale Linearizing Equation and Controller- Controlling Side Acceleration**

The process for this derivation is basically the same as for the pitch channel; however, many of the quantities used in the process are different. The rate of change of gravity bias, for example, is vastly different. Since this mathematical derivation process is exactly the same as for the pitch channel, less detail will be dedicated to the process. It is important to note that in this case, we seek to develop linearizing equations for the slow time scale from the linear side acceleration equation 5.2. The Fast time scale linearizing equation will be developed from the rotational yaw
acceleration equation of 5.6. We begin the derivation with the expression for the linear acceleration (side velocity) of the vehicle.

\[ a_S = \ddot{v} = p\dot{w} - ru + \frac{F_{ap2}}{m} + t_{23}g \]  

(7.87)

Taking the derivative yields,

\[ \dot{a}_S = \ddot{v} = p\dot{w} + \dot{p}\dot{w} - (ru + \dot{r}u) + \frac{1}{m}(\dot{F}_{ap2}) + (i_{23})g \]  

(7.88)

Next, we state all of the relevant equations requiring substitution into the above,

\[ \dot{u} = rv - qw + \frac{F_{ap1}}{m} + t_{13}g \]  

(7.89)

\[ \dot{r} = I_3^{-1}((I_1 - I_2)pq + m_{b3}) \]  

(7.90)

\[ \dot{w} = qu - pv + \frac{F_{ap3}}{m} + t_{33}g \]  

(7.91)

\[ \dot{p} = I_1^{-1}m_{b1} \]  

(7.92)

\[ \dot{F}_{ap2} = \bar{q}SC_{\gamma \beta} \dot{\beta} + \bar{q}SC_{\gamma \delta} \dot{\delta} \]  

(7.93)

\[ i_{23} = 2rq_0q_2 - pq_2q_2 - 2rq_3q_3 + pq_3q_3 + pq_0q_0 - pq_1q_1 \]  

(7.94)

After the appropriate substitutions are made, we have the following,

\[ \dot{a}_y = \ddot{v} = p \left( qu - pv + \frac{F_{ap1}}{m} + t_{33}g \right) + w(I_1^{-1}m_{b1}) \left[ r \left( rv - qw + \frac{F_{ap3}}{m} + t_{33}g \right) + u \left( I_3^{-1}((I_1 - I_2)pq + m_{b3}) \right) \right] + ..... \]  

(7.95)

\[ ..... + \frac{\bar{q}S}{m} \left( C_{\gamma \beta} \dot{\beta} + C_{\gamma \delta} \dot{\delta} \right) + g \left( 2rq_0q_2 - pq_2q_2 + pq_3q_3 + pq_0q_0 - pq_1q_1 \right) \]  

(7.96)
We can now expand all terms,

\[ \dot{a}_r = \ddot{v} = pq u - \rho \dot{v} + \rho \frac{F_{\text{ap}}}{m} + pt_{33} g + w t_{13} m_{13} - r \dot{v} + rw - \rho \frac{F_{\text{ap}}}{m} - r t_{13} g - r t_{3}(I_1 - I_2) pq u - \ldots \]

(7.97)

\[ \ldots - u I_{1} m_{13} + \frac{q S}{m} C_{\beta \beta} \dot{\beta} + \frac{q S}{m} C_{\rho \\
\beta} \dot{\beta} + 2 gr q_0 q_2 - 2 gr q_3 q_3 - gp q_2 q_2 + gp q_3 q_3 + gp q_0 q_0 - gp q_1 q_1 \]

(7.98)

The next step in the derivation process is to find the time rate of change of sideslip, \( \dot{\beta} \). The expression of sideslip is as follows,

\[ \beta = \tan^{-1} \left( \frac{v}{u} \right) \]

(7.99)

Where \( v \) is inertial velocity expressed in body coordinates in the side direction and \( u \) is the inertial velocity expressed in body coordinates in the axial direction. Finding the rate of change of sideslip is as follows,

\[ \dot{\beta} = \frac{d \left( \frac{v}{u} \right)}{dt} \]

(7.100)

We must next apply the quotient rule to the numerator which results in,

\[ \frac{u \dot{v} - v \dot{u}}{u^2} \]

(7.101)

For a total angle of attack rate expression of,

\[ \dot{\beta} = \frac{u \dot{v} - v \dot{u}}{u^2 + v^2} \]

(7.102)
where,

\[
\dot{u} = rv - qw + \frac{F_{ap1}}{m} + t_{13}g
\]  
(7.103)

\[
\dot{v} = pw - ru + \frac{F_{ap2}}{m} + t_{23}g
\]  
(7.104)

\[
\dot{\beta} = \frac{u \left( pw - ru + \frac{F_{ap2}}{m} + t_{23}g \right) - v \left( rv - qw + \frac{F_{ap1}}{m} + t_{13}g \right)}{u^2 + v^2}
\]  
(7.105)

\[
\dot{\beta} = \frac{upw - ru^2 + u \frac{F_{ap2}}{m} + ut_{23}g - rv^2 + qvw - v \frac{F_{ap1}}{m} - vt_{13}g}{u^2 + v^2}
\]  
(7.106)

Therefore,

\[
\dot{\beta} = \dot{v} = pqu - p^2v + p \frac{F_{ap3}}{m} + pt_{33}g + wt_{1}^{-1}m_{g1} - r^2v + rqw - r \frac{F_{ap1}}{m} - rt_{15}g - I_3^{-1}(I_1 - I_2)pqu - ....
\]  
(7.107)

\[
....-uI_3^{-1}m_{g3} + \frac{\bar{q}S}{m} C_{\gamma\beta} \left( \frac{upw - ru^2 + u \frac{F_{ap2}}{m} + ut_{23}g - rv^2 + qvw - v \frac{F_{ap1}}{m} - vt_{13}g}{u^2 + v^2} \right) + \frac{\bar{q}S}{m} C_{\gamma\beta} \dot{\beta} + 2grq_0q_2 - ....
\]  
(7.108)

\[
....-2grq_1q_3 - gpq_2q_2 + gpq_3q_3 + gpq_6q_0 - gpq_5q_1
\]  
(7.109)

At this point, we have exposed all of the variables we must solve for in the yaw channel linear acceleration equation. In the previous section, we developed equations for the pitch command \(q_c\).

For the yaw channel, we must solve for yaw command \(r_c\). We can proceed exactly as before in the case of the pitch channel, by assigning a pseudo control variable, choosing a linear control law and solving for \(r\), which becomes the yaw rate command \(r_c\).

68
\[ \hat{P} = \dot{\alpha}_y \]  

Group terms containing \( r \) yields,

\[ 0 = -\hat{P} + pqu - p^2v + \frac{F_{ap}}{m} + pt_{33}g + wI_1^{-1}m_{g1} - I_3^{-1}(I_1 - I_2)pqu - \ldots \]  

\[ \ldots uI_3^{-1}m_{g3} + \frac{\bar{q}S}{m} C_{\gamma\delta} \left( \frac{upw + u \frac{F_{aq}}{m} + ut_{33}g + qvw - v \frac{F_{ap}}{m} - vt_{13}g}{u^2 + v^2} \right) + \frac{\bar{q}S}{m} C_{\gamma\delta} \hat{\delta} + \ldots \]  

\[ \ldots - gpq_1q_2 + gpq_1q_3 + gpq_0q_0 - gpq_1q_1 - r^2v + rw - r \frac{F_{ap}}{m} - rt_{13}g - \frac{r\bar{q}SC_{\gamma\beta}}{m} + 2grq_0q_2 - 2grq_1q_3 \]  

At this point we can apply the quadratic method in the same manner it was applied to the slow time scale-pitch channel autopilot. Notice that even though the equations used here were very different, the general form remains quite the same, which requires us to use the quadratic formula to solve for the yaw rate command \( r_c \). Using the same format from the last section we have the quadratic expression for the yaw rate command as,

\[ r_c = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \]  

Where in this case,

\[ a = -v \]  

\[ b = +qw - \frac{F_{ap}}{m} - t_{13}g - \frac{\bar{q}SC_{\gamma\beta}}{m} + 2grq_0q_2 - 2grq_1q_3 \]
\[
c = -\dot{P} + pqu - p^2v + p \frac{F_{\text{ap}3}}{m} + pt_{33}g + wt_{11}^{-1}m_{81} - I_1^{-1}(I_1 - I_2)pqu - \ldots \tag{7.117}
\]

\[
\ldots - uI_3^{-1}m_{83} + \frac{\bar{q}S}{m} C_{\gamma \beta} \left( \frac{upw + u F_{\text{ap}2}}{m} + ut_{33}g + qvw - v \frac{F_{\text{ap}1}}{m} - vt_{13}g}{u^2 + v^2} \right) + \frac{\bar{q}S}{m} C_{\gamma \beta} \delta - \ldots \tag{7.118}
\]

\[
\ldots - gpq \dot{q}_2 + gpq \dot{q}_3 + gpq \dot{q}_6 - gpq \dot{q}_1 \tag{7.119}
\]

Once again the exact same simulation structure exists for this portion of the autopilot as did for the pitch channel. Please refer to section “QUADRATIC SLOW TIME SCALE FORM” for a refresher of the details.

Equations 7.14 through 7.19 comprise the complete expression for the yaw rate command, slow time scale controller/linearizing equation; however, we still must deal with the fast time scale equations. Our next task is to derive the expression for the fast time scale which centers around the rotational acceleration dynamics associated with the yaw channel. This discussion and derivation begins the next section.

**Fast Time Scale Linearizing Equation and Controller: Controlling Body Rate r**

The development of the fast time scale equations yield the deflection command for the aerodynamic surface actuators. This process comprises the second half of the nonlinear control law that will control the vehicles yaw plane. We begin this final half of the derivation with the angular acceleration equation of motion for the yaw channel,
\[ \dot{r} = I_3^{-1}((I_1 - I_2)pq + m_{g3}) \]  
(7.120)

The next step is to find the rate of change of angular acceleration from the above equation which results in,

\[ \dot{r} = I_3^{-1}(I_1 - I_2)(pq + \dot{q}p) + I_3^{-1}m_{g3} \]  
(7.121)

Where the time rate of change of the yaw moment is,

\[ \dot{m}_{g3} = \ddot{q}SI(C_{y\beta} \dot{\beta} + C_{y\delta} \dot{\delta}) \]  
(7.122)

This equation reveals the need for a rate of change of beta, (or sideslip), and a time rate of change for the aerodynamic surface actuators. We have already derived the time rate of change of sideslip and repeat it here,

\[ \dot{\beta} = \frac{u_{pw} - ru^2 + u F_{aw1} + ut_{g3}g - rv^2 + qvw - v F_{aw2} + vt_{g3}g}{u^2 + v^2} \]  
(7.123)

We also need to make substitutions for roll rate and pitch rate, \( \dot{p} \) and \( \dot{q} \) respectively, which are given below. Notice the moment equation was included in the roll rate \( \dot{p} \).

\[ \dot{p} = I_1^{-1}m_{g1} = I_1^{-1}\bar{q}SIc_c + I_1^{-1}\bar{q}SIc_\phi \delta_{ROLL} \]  
(7.124)

\[ \dot{q} = I_2^{-1}((I_3 - I_1)pr + m_{g2}) \]  
(7.125)

\[ \dot{\gamma} = I_3^{-1}(I_1 - I_2)((I_1^{-1}\bar{q}SIc_c + I_1^{-1}\bar{q}SIc_\phi \delta_p)\dot{\gamma} + (I_2^{-1}(I_3 - I_1)pr + m_{g2})\dot{p}) + I_3^{-1}\bar{q}SI(C_{y\beta} \dot{\beta} + C_{y\delta} \dot{\delta}) \]  
(7.126)

Where actuator rate is,
\[ \hat{\delta} = 150(\delta_C - \delta) \]  

(7.127)

Expansion of equations yields,

\[ \ddot{\delta} = I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_A + I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_{\varphi}\delta_{ROLL} + \ldots \]  

(7.128)

\[ \ldots + I_2^{-1}I_3^{-1}(I_1 - I_2)(I_3 - I_1)p^2r + I_1^{-1}I_3^{-1}(I_1 - I_2)pm_{B2} + \ldots \]  

(7.129)

\[ \ldots + I_3^{1\dagger}\bar{q}SlC_{\gamma\delta} \left( \frac{upw - ru^2 + u \frac{F_{ap2}}{m} + ut_{23}g - rv^2 + qvw - v \frac{F_{ap1}}{m} - vt_{13}g}{u^2 + v^2} \right) + I_3^{1\dagger}\bar{q}SlC_{\gamma\delta}\hat{\delta} \]  

(7.130)

The next step in this derivation process is to assign a pseudo control variable and apply a linear control law, in this case a PD controller. Therefore,

\[ \ddot{\delta} = \hat{P} \]  

(7.131)

Assigning the pseudo control variable and separating terms yields the following,

\[ \hat{P}_{FAST\_YAW} - I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_A - I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_{\varphi}\delta_{ROLL} - \ldots \]  

(7.132)

\[ \ldots - I_2^{-1}I_3^{-1}(I_1 - I_2)(I_3 - I_1)p^2r - I_2^{-1}I_3^{-1}(I_1 - I_2)pm_{B2} - \ldots \]  

(7.133)

\[ \ldots - I_3^{1\dagger}\bar{q}SlC_{\gamma\delta} \left( \frac{upw - ru^2 + u \frac{F_{ap2}}{m} + ut_{23}g - rv^2 + qvw - v \frac{F_{ap1}}{m} - vt_{13}g}{u^2 + v^2} \right) = I_3^{1\dagger}\bar{q}SlC_{\gamma\delta}\delta_C \]  

(7.134)

We are now in a position to solve for the yaw channel deflection command \( \delta_C \).

\[ \hat{P}_{FAST\_YAW} - I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_A - I_1^{-1}I_3^{-1}(I_1 - I_2)\bar{q}SlqC_{\varphi} - I_2^{-1}(I_3 - I_1)p^2r - pm_{B2} - \ldots \]
At this point we have completed the derivation for the entire yaw channel autopilot. That is, each of the two time scale controllers, fast and slow, have been fully derived for the autopilot that will control the lateral accelerations for the yaw channel of the vehicle. Below are the fast and slow time scale control laws that will be implemented.

The linear control law for the slow time scale is given in equation 7.136 and is to be used in equation 7.117. The equation in 7.137 is the linear control law for the fast time scale and is to be used in equation 7.135. Both control laws are of PID form.

\[
\delta_c = \frac{I_3 \hat{qSly} - \frac{(u_{pp} - r_x^2 + u F_{up} \frac{2}{m} + u t_{23} g - rv - qv - v \frac{F_{up}}{m} - v t_{12} g)}{u^2 + v^2}}{I_3 \hat{qSly}}
\]  

(7.135)

This ends the chapter on the two time scale autopilot design. The same implementation processes for the pitch channel can be used for the yaw channel as well. Considerable time is required for gain tuning, however, and can become tedious at times during development. If all equations have been implemented properly and the autopilot logic properly coded to avoid division by zero issues, gain tuning becomes a matter of time and a little patience. Therefore, it is
advantageous to take plenty of time when implementing these equations. Small errors can create huge headaches when tuning the vehicle’s performance.
In the previous section, we derived equations for two time scale nonlinear control for the pitch and yaw channels, capable of stabilizing body rates and controlling vehicle acceleration of a missile under command from the guidance computers. In this chapter, we will derive a nonlinear control system to stabilize the longitudinal roll axis of the vehicle. This is an extremely important concept in the study of guided missiles. Tail controlled skid-to-turn cruciform missiles must maintain strict control over roll, pitch and yaw channels. Uncontrolled or un-commanded roll excursions are absolutely unacceptable for interceptor missiles attempting to destroy a target for obvious reasons; in order to steer the missile towards a target the rear control fins must remain in a stable position such that control deflection commands can be executed properly. The aerodynamics affecting the longitudinal roll axis can be quite complex and difficult to model. Under moderate angles of attack in the pitch channel, un-commanded roll excursions can become quite distinct. Reasons for this can be attributed to the vortex fields that arise at and flow from the missiles leeward surface under said angle of attack. This affect is called “vortex shedding” and is highly nonlinear in nature. We are all familiar with this phenomenon. Speeding down the interstate in a Ford Mustang at 135 mph with the window open is a prime example of vortex shedding. The turbulent air rushing into and past the window is a result of violent vortices emanating from the forward edges of the vehicle. Missiles traversing the lower atmosphere experience, in general, the same phenomenon. In addition to vortex shedding, the pressure field surrounding a missile executing an angle of attack can be irregular and uneven as well. This
causes the control effectiveness of the two windward fins (assuming zero degree roll orientation in X configuration) to be greater than the two leeward fins. The low leeward air pressure results in an unstable rocking back and fourth of the roll axis as the shedding vortices begin interacting with the leeward fins. Two points now become clear. One, tight roll stabilization is a must. Two, since the aforementioned aerodynamics is extremely difficult to model, robustness of the roll control autopilot must be guaranteed in order to ensure stability under the worst of unforeseen nonlinear vortex shedding. The design method used here will be direct feedback linearization. During development of autopilots for the pitch and yaw channels, direct feedback linearization fails because of the non minimum phase of tail controlled missiles. This instability in the airframe necessitates the two time scale approach. However, no such instabilities exist in the dynamic equations for the vehicles roll axis; the dynamics we seek to stabilize. We must maintain roll orientation at zero degrees as well zero roll rate. In other words, if there is no angular velocity about the longitudinal axis there will be no change in position from the desired zero degree body orientation. Since no instabilities exist in the vehicle roll dynamics, we are well within rights to employ direct feedback linearization.

The rotational dynamic equation for the roll axis is,

\[ \dot{p} = I_1^{-1} m_{B1} \]  

(8.1)

Where \( I_1^{-1} \) is the inverse axial principal moment of inertia and \( m_{B1} \) is the moment about the longitudinal axis. The moment about the longitudinal axis contains all of the aerodynamic effects previously mentioned and is of the form,
\[ m_{\text{bi}} = \bar{q}SIC_c \alpha^2 + \bar{q}SIC_{\psi p} \delta \dot{p} \]  

(8.2)

The first term involving angle of attack on the right hand side of equation 8.2 is an attempt to model vortex shedding previously discussed at the beginning of this chapter. As mentioned before, vortex shedding is highly nonlinear in nature becomes much more profound at higher angles of attack which can be seen in the quadratic term. The second term on the right hand side of 8.2 is the vehicle roll control effectiveness which receives its command from the autopilot to correct any deviations from desired roll position. The roll axis dynamics complete with aerodynamic stability coefficients are presented in equation 8.3 and will form the basis for the roll stabilization autopilots.

\[ \dot{p} = I_1^{-1} \left( \bar{q}SIC_c + \bar{q}SIC_{\psi p} \delta \right) \]  

(8.3)

In order to include actuator dynamics in the linearizing control law, we take the first derivative with respect to time which yields equation 8.4,

\[ \dot{p} = I_1^{-1} SIC_c \left( 2\bar{q} \alpha + \alpha^2 \dot{\bar{q}} \right) + I_1^{-1} SIC_{\psi p} \left( \bar{q} \dot{\delta}_p + \delta \dot{p} \dot{\bar{q}} \right) \]  

(8.4)

Where \( \dot{\delta} \) will take on the form of a first order actuator lag as,

\[ \dot{\delta} = 150 \left( \delta_c - \delta_p \right) \]  

(8.5)

It is important to remember from chapter 6 that first order actuator dynamics typically do not pose destabilizing effects on nonlinear autopilots; therefore, autopilot design will begin with this type of actuator. Once the design is tuned and performance evaluated, nonlinear second order effects will be investigated. For now, this build-up type method is a crucial step in swift,
competent autopilot design. We now embed actuator dynamics of equation 8.5 into 8.4 which results in,

\[
\dot{p} = I_1^{-1}SLC_c \left( 2\bar{q} \alpha + \alpha^2 \bar{q} \right) + I_1^{-1}SLC_{\phi_p} \left[ \bar{q} \left( 150 \delta_c - \delta_p \right) \right] + \delta_p \bar{q}
\]  

(8.6)  
\[
\dot{p} = I_1^{-1}SLC_c 2\bar{q} \alpha + I_1^{-1}SLC_c \alpha^2 \bar{q} + 150I_1^{-1}SLC_{\phi_p} \bar{q} \delta_c - 150I_1^{-1}SLC_{\phi_p} \bar{q} \delta_p + I_1^{-1}SLC_{\phi_p} \delta_p \bar{q}
\]  

(8.7)  

An important point must be made at this time. \( \delta_p \) in equation 8.4 is output from the actuators and not the autopilot! This distinction must be made in equations 8.6 and 8.7 as well.

As was done in chapter 7, we define a pseudo control variable to the left hand side of equation 8.7,

\[
\dot{Z} = I_1^{-1}SLC_c 2\bar{q} \alpha + I_1^{-1}SLC_c \alpha^2 \bar{q} + 150I_1^{-1}SLC_{\phi_p} \bar{q} \delta_c - 150I_1^{-1}SLC_{\phi_p} \bar{q} \delta_p + I_1^{-1}SLC_{\phi_p} \delta_p \bar{q}
\]  

(8.8)  

This pseudo control variable will become the linear control law which we will turn our attention to at this point. We will make use of a PID control law in which angular position and velocity will be controlled since these variables are directly observable in the system. \( \dot{Z} \) then takes the form of equation 8.9,

\[
\dot{Z} = K_pe + K_i \int e \, dt + K_d \dot{e}
\]  

(8.9)  

Let us take a look at the error dynamics within equation 8.9. The error can be specified as the difference between the desired roll position in body coordinates and the actual roll position,

\[
e = r_D - r_A
\]  

(8.10)  

The time rate of change of error is,

\[
\dot{e} = \dot{r}_D - \dot{r}_A
\]  

(8.11)
Upon inspection of the error rate in equation 8.11, we see that the time rate of change of the roll position is actually the body roll rate $p$, therefore,

$$\dot{r}_A = p \quad (8.12)$$

Other determinations can be made as well. Since we seek to roll stabilize the vehicle at zero degrees, not only will the desired roll position $r_D$ be zero but $\dot{r}_D$ will be zero as well. Therefore,

$$r_D = \dot{r}_D = 0 \quad (8.13)$$

At this point, we can finalize the form of the linear control law as,

$$\dot{Z} = -K_p r_A - K_i \int r_d dt - K_D p \quad (8.14)$$

We are now in a position to derive the complete form of the roll control autopilot which begins by substituting equation 8.14 into the linearizing control law in 8.8 and solving for the deflection command $\delta_c$,

$$\delta_c = \frac{-K_p r_A - K_i \int r_d dt - K_D p - I_1^{-1} SIC_e 2q \alpha - I_1^{-1} SIC_e \alpha^2 \dot{q} + 150I_1^{-1} SIC_{\phi} \delta \dot{q} - I_1^{-1} SIC_{\phi} \delta \dot{q}}{150I_1^{-1} q SIC_{\phi}} \quad (8.15)$$

Equation 8.15 is the fin deflection command which roll stabilizes the vehicle’s roll axis. At this point, between this chapter and chapter 7, three separate autopilots have been designed which rounds out our objective of developing an autopilot structure in 6 degrees of freedom. Once all three autopilots are implemented, stabilization of all body rates $p, q$ and $r$ are possible.
while at the same time controlling acceleration in the normal and side directions in the presence of non minimum phase. In the next chapter we look at vehicle performance on several fronts.
CHAPTER 9
ANALYSIS OF VEHICLE PERFORMANCE

This Chapter begins a comprehensive look at several different levels of vehicle performance. The first level comprises vehicle performance in the pitch channel with no roll or yaw dynamics involved. The tests will be conducted with linear first order actuators and will report vehicle response to different acceleration commands. The second level of performance tests will involve a full 6 degree of freedom simulation run involving roll, pitch and yaw dynamics. The third level of performance tests will determine how well the vehicle stabilization and control system can track a varying guidance command signal. Since this vehicle is of a high speed long duration type, rapidly varying guidance commands are not expected as would be with a short range air to air missile like the Sidewinder; however, visibly good performance under these circumstances gives indication as to the robustness of the designed control. Finally, a comprehensive look at vehicle performance with nonlinear second order actuators will be conducted and evaluations made.

TEST 1 - PITCH CHANNEL TESTS

These test results demonstrate the ability of the Pitch autopilot to effectively stabilize and control the vehicle motion in the normal/axial plane under non minimum phase conditions.

TEST CRITERION
Roll channel disabled: 0 degree roll orientation enforced.
Yaw channel disabled: motion occurs in pitch plane only.
Launch angle: 75 degrees with respect to downrange.
Acceleration command: 15 meters per second squared.

*Note: For all graphs, the horizontal axis is time in seconds unless specified otherwise.
Figure 16: Fast Time Scale Response
Vertical axis: Degrees of fin deflection

Figure 17: Slow Time Scale Response
Vertical axis: Angular rate in degree
Figure 18: Normal Acceleration
Vertical axis: Meters/second squared

Figure 19: Angle of Attack
Vertical Axis: Degrees
Figure 20: Pitch Moment
Vertical axis: Kilogram meters squared

Figure 21: Airframe Normal Force
Vertical Axis: Newtons
Figure 22: Normal Velocity
Vertical Axis: Meters per second

Figure 23: Non minimum Phase in Normal Velocity
Vertical Axis: Meters per second
TEST 2 - FULLY COUPLED ROLL PITCH AND YAW TEST

These test results demonstrate the ability of the roll, pitch and yaw autopilots to operate in unison with no adverse effects from dynamic coupling.

TEST CRITERION
Roll channel enabled: Vehicle roll orientation stabilized at zero degrees.
Yaw channel enabled: Full 6 degree of freedom motion enabled.
Launch angle: 75 degrees with respect to downrange.
Pitch and Yaw acceleration command: 15 meters per second squared.
Figure 25: Roll Command
Vertical axis: Degrees of fin deflection

Figure 26: Roll Position
Vertical axis: Degrees
Figure 27: Angle of Attack
Vertical Axis: Degrees

Figure 28: Sideslip
Vertical Axis: Degrees
Figure 29: Normal Acceleration
Vertical axis: Meters/second squared

Figure 30: Side Acceleration
Vertical axis: Meters/second squared
Figure 31: Fast Time Scale Response – Pitch
Vertical axis: Degrees of fin deflection

Figure 32: Slow Time Scale Response – Pitch
Vertical axis: Angular rate in degree
Figure 33: Fast Time Scale Response – Yaw
Vertical axis: Degrees of fin deflection

Figure 34: Slow Time Scale Response - Yaw
Vertical axis: Angular rate in degree
TEST 3 - FULLY COUPLED PITCH AND YAW TEST

These test results demonstrate the ability of the roll, pitch and yaw autopilots to operate in unison with no adverse effects from dynamic coupling under high acceleration command from the guidance processor. These tests will determine the acceleration command limit of the design. Roll channel enabling is not necessary.

TEST CRITERION
Roll channel disabled: Vehicle roll orientation stabilized at zero degrees.
Yaw channel enabled: Full 6 degree of freedom motion enabled.
Launch angle: 75 degrees with respect to downrange.
Pitch and Yaw acceleration command: 30 meters per second squared.

Figure 35: Fast Time Scale Response – Pitch
Vertical axis: Degrees of fin deflection

Figure 36: Slow Time Scale Response – Pitch
Vertical axis: Angular rate in degree
Figure 37: Normal Acceleration
Vertical axis: Meters per second squared

Figure 38: Angle of Attack
Vertical axis: Degrees
Figure 39: Fast Time Scale Response – Yaw
Vertical axis: Degrees of fin deflection

Figure 40: Slow Time Scale Response – Yaw
Vertical axis: Angular rate in degree
Figure 41: Side Acceleration

Vertical axis: Meters per second squared

Figure 42: Sideslip

Vertical axis: Degrees
TEST 4 - NONLINEAR ACTUATOR PITCH CHANNEL TESTS

These test results demonstrate the ability of the Pitch autopilot to effectively stabilize and control the vehicle motion in the normal/axial plane under non minimum phase conditions with a nonlinear second order actuator model present.

TEST CRITERION
Roll channel disabled: 0 degree roll orientation enforced.
Yaw channel disabled: motion occurs in pitch plane only.
Launch angle: 75 degrees with respect to downrange.
Acceleration command: 15 meters per second squared.

Figure 43: Fast Time Scale Response
Vertical axis: Degrees of fin deflection

Figure 44: Slow Time Scale Response
Vertical axis: Angular rate in degree
**Analysis of Test Results**

Overall, testing yielded very promising results. Test 1, which was conducted to evaluate the performance of the pitch plane only, shows very good results. Stable operation of both the slow and fast time scales can be observed. Test 2 was the most important performance test of the group and demonstrates the ability of all three autopilots to work in unison without any apparent problems even in the presence of dynamic coupling due to coriolis terms. Test 3 shows the autopilots ability to track larger acceleration commands provided by the guidance computers with little effect on performance. However, destabilization of the pitch channels slow time scale began to occur as the acceleration commands exceeded 50 meters per second squared. The yaw channel seemed to be unaffected by the higher acceleration commands but the instabilities from the pitch channel began showing up in the yaw channel simply because of dynamic coupling. During stable lower level acceleration command operation of the pitch channel, yaw channel performance could be pushed well in excess of 100 meters per second squared. This suggests that the pitch channel, due to non minimum phase, is less tolerable of high acceleration commands. In fact the fast time scale remained stable throughout the entire test; only the slow time scale exhibited problems and this is where the non minimum phase resides. Tuning of the pitch channels gains showed improved tolerance and better performance during higher acceleration commands. Test 4 shows the results of the non linear actuator model performance. Performance was poor but possibly shows potential for good performance. Once again it is the slow time scale that exhibits the instability. The gains for this autopilot were adjusted many times and the best performance attained was displayed here. While the results of this test show promise, more time and resources must be spent in order to fully work out the problems associated with second order
nonlinear actuators present in the closed loop. Steady state error in tests 1, 2, and 3 was relatively good but became almost zero under low changes in dynamic pressure. Previous tests showed that as the vehicle exited the area of maximum dynamic pressure in the atmosphere, the vehicle controls began to settle down and steady state error reduced to very small levels.
CONCLUSION

The many months of research on this topic has shown great potential for the two time scale nonlinear control method. While it is felt that many more months of research could have been conducted, a line must be drawn at some point and work brought to a conclusion. The areas of potential research within this topic are plentiful to say the least. The design itself has shown great promise although some issues did arise during development. A great sensitivity to the rapid rise in dynamic pressure made autopilot gain adjustment very difficult. If the dynamic pressure was held constant as it was during the initial phases of testing, the autopilots tracked acceleration commands with almost no steady state error, however, with the massive solid rocket engine thrust, the vehicle reached great speeds quickly and the steady state error did deteriorate to some degree while traversing the denser lower atmosphere. Adding to this problem was a lack of smooth transitioning atmospheric mathematical models, namely at the atmospheric boundary of 11km. At the 11km point and above, temperature should remain constant. This switching point, while subtle, caused a very slight, almost unperceivable elbow point in the dynamic pressure profile. However, since dynamic pressure comprises part of the input to the dynamic equations, the solution to the differential equations reveals a large transient response in the acceleration output due to this switching point. This in turn caused a rather large transient period within the slow time scale portion of the pitch channel autopilot that no amount of gain adjustment could dampen out. A better atmospheric model is definitely needed if further work is ever done on this project.

Even though dynamic inversion cancels out all known nonlinear terms leaving a controllable linear system, sensitivity to the rapid rise in dynamic pressure was still ever present even though
every step was taken to mathematically account for this. This nonlinear control method shows great promise but improvements could vastly enhance performance such as adding an adaptive control component to this system or adding robustness. Possibly, near zero steady state error could be achieved under any dynamic pressure condition with more research and work. The same can be said for incorporating a non linear second order actuator model into the closed loop system. Test results show promise but once again more time is required.

As I sit here typing this, I think of the time required to resolve all of these problems and sometimes wonder how old I might be or how much hair I would have left when I finally make this system perfect and or explore all of the additional area’s of research associated with this system . I can say with certainty another year, possibly two or three, might get me close. I’ve always entered design projects with lots of zeal and big plans but building a six dof missile simulation is a tremendous task and the monumental nature of what you have gotten yourself into quickly sets in. Adding nonlinear actuators to the closed loop is enough to keep a person busy for months much less trying to find the bugs in chapter 7’s equations; believe me, there were plenty of them too. I’ve worked diligently on this project for just over two years at this point and still never got to address the changing center of mass issue. Over all, I’m pretty proud of my work on this project but it is time to move on. I have been in the Ph.D. program for a little over a year now but will soon be starting on the core of my dissertation research in the coming months. I will begin working on the cooperative control of interceptor missiles aimed at controlling multiple long range interceptor missiles to strike multiple inbound I.C.B.M’s to the United States. That ought to keep me busy for a long time.

Project dates: 15 February, 2007 to 06 April , 2009
REFERENCES