ONLINE PATH PLANNING AND CONTROL SOLUTION FOR A COORDINATED ATTACK OF MULTIPLE UNMANNED AERIAL VEHICLES IN A DYNAMIC ENVIRONMENT

by

JUAN E. VEGA-NEVÁREZ
B.S. University of Puerto Rico, 2003

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
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

Fall Term
2012

Major Professor: Zhihua Qu

ABSTRACT

The role of the unmanned aerial vehicle (UAV) has significantly expanded in the military sector during the last decades mainly due to their cost effectiveness and their ability to eliminate the human life risk. Current UAV technology supports a variety of missions and extensive research and development is being performed to further expand its capabilities. One particular field of interest is the area of the low cost expendable UAV since its small price tag makes it an attractive solution for target suppression. A swarm of these low cost UAVs can be utilized as guided munitions or kamikaze UAVs to attack multiple targets simultaneously. The focus of this thesis is the development of a cooperative online path planning algorithm that coordinates the trajectories of these UAVs to achieve a simultaneous arrival to their dynamic targets. A nonlinear autopilot design based on the dynamic inversion technique is also presented which stabilizes the dynamics of the UAV in its entire operating envelope. A nonlinear high fidelity six degrees of freedom model of a fixed wing aircraft was developed as well that acted as the main test platform to verify the performance of the presented algorithms.

To my wife María and my beautiful daughter Monique.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Dr. Zhihua Qu for his guidance, support, and overall, for his infinite patience. I would also like to thank my dearest friends Ricardo Colón, Ricardo Elías, and Leticia Delgado for their continuous support and constant harassment to obligate me complete this thesis.

I would also like to offer my most heartfelt gratitude to my greatest mentor, my grandfather Juan Nevárez, since his guidance and example have been some of the main elements for defining who I am today.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

3-D	Three Dimensional
3-DOF	Three Degrees of Freedom
6-DOF	Six Degrees of Freedom
AOA	Angle of Attack
BLOS	Beyond Line of Sight
CAS	Close Air Support
CG	Center of Gravity
DoD	Department of Defense
ECEF	Earth Centered Earth Fixed
ECI	Earth Centered Inertial
ENU	East North Up
FRD	Front Right Down
GPS	Global Positioning System
INS	Inertial Navigation System
LOS	Line of Sight
LTP	Local Tangent Plane
NED	North East Down
NWU	North West Up
PID	Proportional Integral Derivative
SA-1976	Standard Atmosphere of 1976
SAM	Surface to Air Missile
SEAD	Suppression of Enemy Air Defenses
UAV	Unmanned Aerial Vehicle
UCAV	Unmanned Combat Aerial Vehicle
VV&A	Verification, Validation, and Accreditation
WGS-84	World Geodetic System of 1984

CHAPTER 1: **INTRODUCTION**

The unmanned aerial vehicle (UAV) has become a crucial asset for the military organizations across the globe since it is capable of executing high risk missions without jeopardizing human life as well as performing missions typically done by manned aircraft but at a fraction of the cost. This military interest in UAV technology has created an unprecedented growth in the research and development efforts of new hardware and software solutions that expand the capabilities of the UAV enabling it to perform a variety of new missions. One particular area of interest is the utilization of UAVs in combat roles allowing them to support missions such as strike, suppression of enemy air defenses (SEAD), and close air support (CAS). These UAV types, known as Unmanned Combat Air Vehicles (UCAV), can range from multi-million dollar re-deployable units equipped with weapons (such as the MQ-9 Reaper) to low cost expendable units where the UAV itself is the weapon. The research topics presented in this thesis are focused on the latter ones.

The concept of the kamikaze UAV is nothing new. Years before the invention of the airplane, Charles Perley registered a patent for an unmanned aerial bomber which consisted of a hot air balloon packed with explosives and a timing mechanism. This balloon was used during the American Civil War by both the Union and Confederate forces. In 1917, during World War I, inventors Elmer Sperry and Peter Hewitt developed an automatic airplane for the U.S. Navy to act as an aerial torpedo. This design became the first radio controlled UAV. Another example of an early kamikaze UAV is the V-1

Flying Bomb developed by the Nazis in World War II. The V-1, known as the predecessor of the modern cruise missile, was first deployed in 1944 against Britain killing more than 900 civilians showing the terrorizing capabilities of the UAV for the first time in history. Forwarding to present time, some examples of modern day kamikaze UAVs include the IAI Harpy, the L-3 Cutlass, and the Switchblade. The Harpy, made by Israel, is a ground vehicle launched UAV that detects radar emissions from hostile Surface to Air Missile (SAM) sites and automatically destroys them by flying directly into them making it a perfect choice for SEAD missions. The Cutlass and the Switchblade, on the other hand, are small UAVs packed in canisters that can be launched both from the ground by troops or from the air by other aircraft. These UAVs can provide beyond line of sight (BLOS) target recognition and suppression making them ideal for close air support missions.

The examples of the three modern day UAVs provided above provide excellent capabilities for target suppression missions. However; these systems currently lack the capability of synchronizing multiple UAVs to execute a coordinated attack on multiple targets simultaneously. Since most SEAD and strike missions consist of destroying multiple primary and secondary targets, a simultaneous attack is the preferred option since it maintains the element of surprise which one of the key components of a successful combat mission. The main objective of this thesis is to address this shortcoming by developing an online path planning algorithm that includes a coordination technique among multiple UAVs that allow them to reach their final targets

simultaneously. The trajectory planning algorithm is a modified version of Dr. Qu's real time path planning with collision avoidance techniques presented in [1]. The secondary objective of this thesis is the development of a nonlinear autopilot based on the dynamic inversion technique that will stabilize the UAV dynamics across its entire operating envelope without the need of developing multiple linear controllers like the gain scheduling control law requires. This nonlinear autopilot, combined with the path planning algorithm, provides a complete guidance and control solution that can be utilized by any kamikaze UAV platform. A detailed description of a high fidelity nonlinear six degrees of freedom (6-DOF) model is also provided in this thesis since it was used as the main tool to verify the performance of the path planning and control algorithms.

CHAPTER 2: MODELING AND NAVIGATION FUNDAMENTALS

This chapter presents some of the basic concepts required to understand the general problem of guiding and controlling an aerial vehicle flying in the vicinity of the Earth. The main topics to be discussed include: Earth modeling, coordinate frames, equations of motion, and basic aerodynamics. The first topic defines the concept of geodesy which is an important concept to vehicle guidance and navigation. The following topic provides an insight of the typical coordinate frames used in guidance algorithms as well as other frames that are normally used for modeling of vehicle dynamics. The next topics introduce the concepts of rigid body dynamics, kinematics, and aerodynamics which are crucial to understand the motion behavior of a vehicle flying within the Earth's atmosphere.

2.1 Geodesy

The term geodesy is defined as the branch of applied mathematics concerned with the determination of the size and shape of the Earth and its gravitational field. Several mathematical models have been developed in the past, however, the most recent model which is used by the United States and by the Global Positioning System (GPS) is known as the World Geodetic System of 1984 (WGS-84). This mathematical model, developed by the U.S. Department of Defense (DoD), represents the Earth as an oblate spheroid where the equatorial radius is defined as the semi-major axis (R_a) and the polar radius is defined as the semi-minor axis (R_b). The values for these two

parameters and other important constants related to the WGS-84 model are shown in Table 2-1.

Table 2-1: WGS-84 Constants

Parameter	Description	Value	Units
Ra	Equatorial Radius	6,378,137	meters
R_b	Polar Radius	6,356,753	meters
f	Earth Flattening	1 / 298.257223563	unitless
ε	Earth Eccentricity	0.0818191908426	unitless
ω_E	Earth Angular Velocity	7.292115 x 10-5	rad/s
g_{WGS_0}	Earth Gravity at Equator	9.7803267714	m/s^2
g_{WGS_1}	Gravity Formula Constant	0.00193185138639	m/s^2

2.1.1 Earth's Gravity Model

The term gravity is defined as the acceleration experienced by an object on a rotating Earth. This acceleration is the combination of the mass attraction between the object and the Earth, known as gravitation, and the centripetal acceleration caused by the rotation of the Earth. The WGS-84 model calculates the magnitude of the gravity vector, which is normal to the ellipsoid surface, using equation 2.1. It can be noticed that this magnitude is greatly influenced by the geodetic latitude (φ).

$$g = g_{WGS_0} \frac{\left(1 + g_{WGS_1} sin^2\varphi\right)}{\sqrt{1 - \varepsilon^2 sin^2\varphi}} \quad \textbf{(2.1)}$$

2.2 Coordinate Frames and Transformations

Aircraft navigation system states, such as position, velocity, and attitude, are always defined in reference to a particular coordinate system. In addition, important vectors such as the forces and moments exerted on the aircraft body are also measured in reference to a coordinate frame. This section provides an insight into the typical coordinate systems used in vehicle navigation and modeling.

2.2.1 Earth Centered Inertial (ECI) Frame

The Earth Centered Inertial (ECI) frame is a Cartesian coordinate frame that has its origin located at the center of the Earth. Conventionally, the vertical (Z) axis is aligned with the Earth's rotational axis pointing to the North Pole, the horizontal (X) axis is aligned with the vernal equinox, and the lateral (Y) axis completes the right handed orthogonal system. The vernal equinox is defined as the intersection between the Earth's orbit plane (the ecliptic) and the Earth's equatorial plane. The ECI frame is not a truly Newtonian inertial frame since the Earth is accelerating while traveling around the Sun. However, for objects moving within the Earth's atmosphere, it can be assumed to be a true inertial frame without detrimental effects. An illustration of the ECI frame is shown in Figure 2-1.

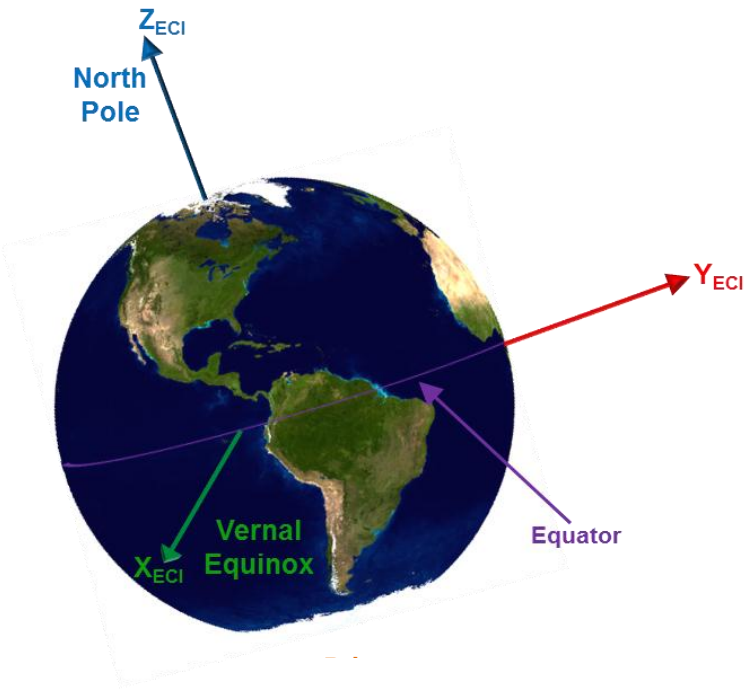

Figure 2-1: Earth Centered Inertial (ECI) Frame

2.2.2 Earth Centered Earth Fixed (ECEF) Frame

The Earth Centered Earth Fixed (ECEF) frame (shown in Figure 2-2) is another Cartesian coordinate frame that has its origin located at the center of the Earth; however, it differs from the ECI frame since this coordinate system rotates with the Earth maintaining a fixed orientation with the Earth's surface. In this frame, the vertical (Z) axis is aligned with the Earth's rotational axis pointing North, the horizontal (X) axis is pointing towards the intersection of the prime meridian and the Equator (0° latitude, 0° longitude), and the lateral (Y) axis completes the right handed coordinate system pointing outwards somewhere in the Indian Ocean. This frame is extremely useful to

represent the location of an object in relation with the Earth's surface. For this reason, it is widely used in navigation systems such as the GPS.

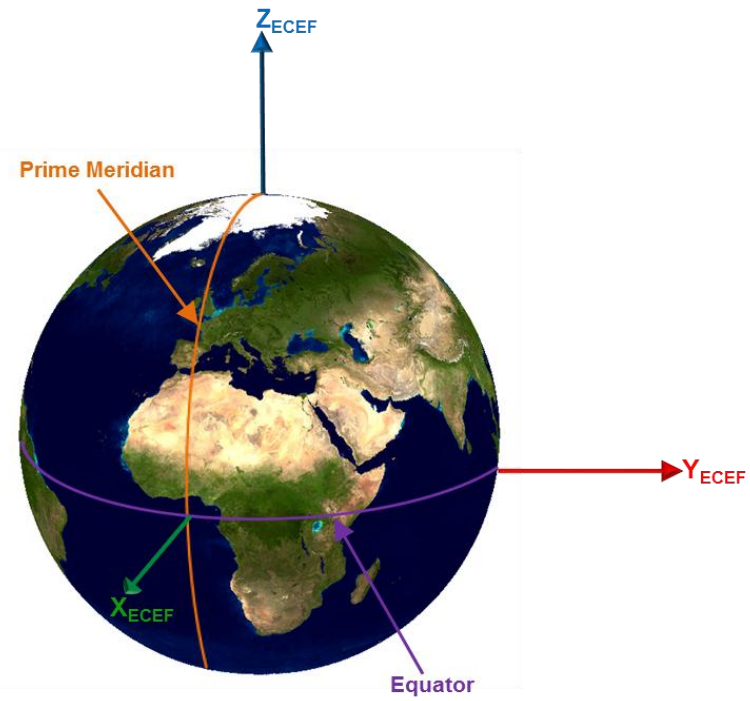

Figure 2-2: Earth Centered Earth Fixed (ECEF) Frame

2.2.3 Local Geodetic Plane

The local geodetic plane, also known as the local tangent plane, is one of the most intuitive reference frames since it is often used in our everyday life. This frame has its origin at an arbitrary location in the surface of the Earth and assumes a flat Earth around it. Several different conventions are used to define the axes orientation such as East-North-Up (ENU), North-West-Up (NWU), and North-East-Down (NED). The NED convention is widely used for aircraft simulation and it will be the convention used for

this thesis. In this convention, the X axis points towards Cardinal North, the Y axis points towards the East, and the Z axis points downwards towards the Earth surface. A portrayal of the NED frame is shown in Figure 2-3.

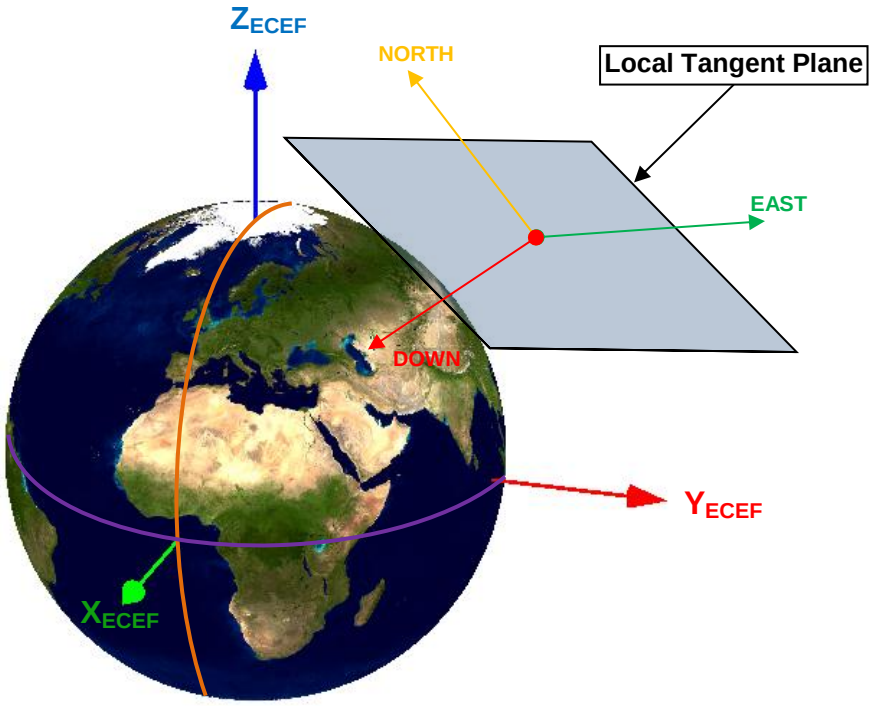

Figure 2-3: Local Tangent Plane with NED convention

2.2.4 Body Fixed Coordinate Frame

In the body fixed frame, the axes of the coordinate system are rigidly attached to the nominal geometry of the aircraft. Typically, the origin of this frame is located at the center of gravity (CG) of the aircraft. Also, the X-Z plane nominally lies around the plane of symmetry. The generic names given for the three axes are longitudinal (X), lateral (Y), and vertical (Z). There are several different sign conventions for axis definition. For

this thesis, the convention to be used is the Front-Right-Down (FRD) convention, where the X axis points forward towards the nose of the aircraft, the Y axis points towards the right wing, and the Z axis points downwards. Figure 2-4 illustrates the FRD coordinate system. This frame is widely used in the definition of aircraft dynamic forces and moments since it permits separation of the rotational and translational dynamics.

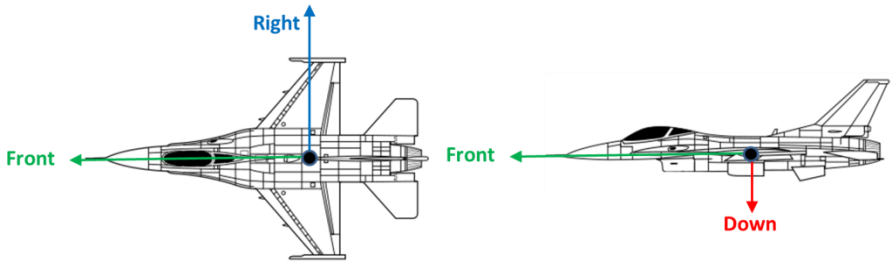

Figure 2-4: Body Fixed Coordinate Frame

2.2.5 ECI to ECEF Conversion

Vectors expressed in the ECI frame can easily be converted to the ECEF frame since the rotation between both frames is a plane rotation around the Z axis. Equation 2.2 shows the rotation matrix used for this conversion.

$$R_{e/i} = \begin{bmatrix} cos\,\eta & \sin\eta & 0 \\ -\sin\eta & \cos\eta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(2.2)}$$

The parameter η, defined as the rotation angle between both frames, can be determined using the following equation:

$$\eta = \lambda_0 + \omega_E t \quad \textbf{(2.3)}$$

The variable λ_0 is defined as the initial celestial longitude. The variable ω_E is defined as the Earth's rotational rate (see Table 2-1). Finally, t is defined as the elapsed time in seconds.

2.2.6 ECEF to Local Tangent Plane Conversion

The rotation matrix used for the conversion between the ECEF frame and the local tangent plane is dependent on the current geodetic latitude (φ) and geodetic longitude (ℓ) of the aircraft. The matrix is calculated as follows:

$$R_{n/e} = \begin{bmatrix} -\sin\varphi\cos\ell & -\sin\varphi\sin\ell & \cos\varphi \\ -\sin\ell & \cos\ell & 0 \\ -\cos\varphi\cos\ell & -\cos\varphi\sin\ell & -\sin\varphi \end{bmatrix} \quad \textbf{(2.4)}$$

2.2.7 Local Tangent Plane to Body Conversion

The relationship between the local tangent plane and the body fixed frame is typically defined by three successive rotations where the rotational angles are known as Euler angles (these angles will be explained in more detail in section 2.3.3). The names given

to these three angles are: yaw (ψ), pitch (θ), and roll (ϕ). The rotational matrix is defined as:

$$R_{b/n} = \begin{bmatrix} c\theta c\psi & c\theta s\psi & -s\theta \\ -c\phi s\psi + s\phi s\theta c\psi & c\phi c\psi + s\phi s\theta s\psi & s\phi c\theta \\ s\phi s\psi + c\phi s\theta c\psi & -s\phi c\psi + c\phi s\theta s\psi & c\phi c\theta \end{bmatrix} \quad \textbf{(2.5)}$$

where s and c are the abbreviations of sine and cosine respectively.

2.3 Rigid Body Equations of Motion

The equations of motion of a rigid body are represented by a set of mathematical formulas that describe the motion of a particle or body on an inertial frame. These set of equations are divided into two main subjects: dynamics and kinematics. Dynamics, from the Greek word dunamis (power), studies the causes and changes in motion produced by the forces and moments acting on a body. Kinematics, from the Greek word kinesis (motion), describes the motion of a body without the consideration of forces and moments. It is often referred as the geometry of motion. These two topics are further subdivided into two types of motion: rotational and translational. The rotational and translational equations are typically co-dependent, however, when the center of gravity of the body is used as the reference point, they can be decoupled from each other.

The equations of motion presented in this section will act as the heart of the nonlinear six degrees of freedom (6-DOF) model described in the following chapter. The

equations below are derived assuming a flat, non-rotating Earth. This assumption can be used for the simulation of a subsonic vehicle flying over a small region of the Earth which is sufficient for the purposes of this thesis.

2.3.1 Rotational Dynamics

The rotational dynamic equations are responsible for calculating the angular velocity vector of the vehicle based on the moments that are exerted on the body. The moments are mainly generated by the aerodynamic effects of the fuselage and the control surfaces as well as the components of the propulsion force that are not aligned with the longitudinal axis.

Using Newton's law, the sum of all moments on a body equals the derivative of the angular momentum. The angular momentum vector (h_b) is defined as the multiplication of the inertia matrix (J_b) by the angular velocity vector (ω_b). The inertia matrix of a rigid body that is symmetric on the X-Z plane is composed of six parameters: three axial moments of inertia, and three cross products of inertia. The following equation shows how to calculate the angular momentum.

$$h_b = \begin{bmatrix} J_{xx} & -J_{xy} & -J_{xz} \\ -J_{xy} & J_{yy} & -J_{yz} \\ -J_{xz} & -J_{yz} & J_{zz} \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} = J_b \omega_b \qquad \textbf{(2.6)}$$

The equation of Coriolis states that the derivative of a vector that is fixed in a specific frame (in this case the body frame) and that rotates with respect to a reference frame (NED frame) is equal to the rate of change of that vector in the body frame plus the cross product of this vector with the angular velocity vector. Using this equation, we are able to calculate the total moment (τ_b) acting on the vehicle as follows:

$$\tau_b = \dot{h}_b + \omega_b \times h_b = J_b\, \dot{\omega}_b + \omega_b \times (J_b\, \omega_b) \quad \textbf{(2.7)}$$

Rearranging equation 2.7 we can obtain the state equation for the angular velocity vector:

$$\dot{\omega}_b = (J_b)^{-1}[\tau_b - \Omega_b\, J_b\, \omega_b] \quad \textbf{(2.8)}$$

The term Ω_b is defined as the cross product matrix of the angular velocity vector. This matrix is calculated using equation 2.9.

$$\Omega_b = \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & P & 0 \end{bmatrix} \quad \textbf{(2.9)}$$

2.3.2 Translational Dynamics

The translational dynamic equations are responsible for calculating the linear velocity vector of the vehicle based on the forces acting on the body which typically consist of the aerodynamic forces (F_{A_b}), the engine thrust (F_{T_b}), and gravity (g). Based on Newton's law and the equation of Coriolis explained above, a linear relationship can be determined between the forces and the vehicle's acceleration as shown below:

$$\sum F_b = F_{A_b} + F_{T_b} + mg_b = m[\dot{v}_b + \omega_{b/n} \times v_b] \quad \textbf{(2.10)}$$

In the equation above, the term m is defined as the vehicle mass and the term g_b is the gravity vector expressed in the body frame. Utilizing the NED to body rotation matrix to calculate g_b and re-arranging equation 2.10 we can obtain the state equation for the linear velocity vector:

$$\dot{v}_b = \left(\frac{1}{m}\right)[F_{A_b} + F_{T_b}] + R_{b/n}\, g_n - \Omega_b v_b \quad \textbf{(2.11)}$$

2.3.3 Rotational Kinematics

The rotational kinematic equations are responsible for determining the attitude of the aircraft. The attitude of a vehicle can be represented in two different manners: Euler angles and quaternions.

The orientation of a coordinate system in reference to another can be represented by performing three successive rotations. The rotation angles that describe this orientation are known as Euler angles. In the case of aircraft attitude, the Euler angles provide the orientation of the aircraft body coordinate frame in reference to the local geodetic plane. For a NED frame, the typical rotation sequence used is the yaw-pitch-roll convention. The roll angle (ϕ) is the rotation that is performed along the longitudinal axis of the aircraft. Its positive notation is defined as a right roll (right wing down, left wing up). The pitch angle (θ) is the rotation along the lateral axis of the aircraft and its positive notation is defined as a nose up attitude. The yaw angle (ψ) is the rotation along the vertical axis of the aircraft and its positive notation is defined as a right yaw (nose to the right). The state equation of the aircraft attitude expressed in Euler angles can be computed from the angular body rates using the following equation:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \tan\theta \sin\phi & \tan\theta \cos\phi \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad \textbf{(2.12)}$$

As it can be noticed, equation 2.12 utilizes the tangent function to compute the Euler angles which produces singularities at pitch angles of ±90°. This issue can be considered insignificant for the simulation of most helicopters and cargo aircraft; however, it generates severe limitations in the simulation of highly maneuverable aircraft such as fighter planes and certain types of UAVs. This equation also requires a good amount of processing power due to the usage of trigonometric equations.

An alternate method of expressing the attitude of an aircraft is known as quaternions. Quaternions are commonly used today in the simulation of aircraft, missiles, spacecraft, and other types of 3-DOF and 6-DOF simulations due to their ease of use, low processing requirements, and lack of singularities. Quaternions are a four element unit vector composed of a rotation vector and a scalar value. Instead of performing three successive rotations on an orthogonal frame like the Euler angles require, the quaternion represents attitude by achieving a single rotation around a unique axis known as the quaternion vector. The state equation of the aircraft attitude expressed in quaternions can be computed from the angular body rates using the following equation:

$$\dot{q} = \frac{1}{2}\begin{bmatrix} 0 & -\omega_b^T \\ \omega_b & -\Omega \end{bmatrix} q = \begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 & -P & -Q & -R \\ P & 0 & R & -Q \\ Q & -R & 0 & P \\ R & Q & -P & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad \textbf{(2.13)}$$

The utilization of quaternions is computationally efficient; however, quaternions are not as intuitive as Euler angles. For this reason, even though quaternions are used for the internal simulation, the attitude is normally expressed in Euler angles. Equations used to convert quaternions to Euler angles and vice versa are very useful. For example, in a simulation the initial attitude of the aircraft is expressed in Euler angles and is then converted to quaternions to initialize the quaternion integrator using the following equation:

$$\begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) - \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) + \sin\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) \\ \cos\left(\frac{\phi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\psi}{2}\right) - \sin\left(\frac{\phi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\psi}{2}\right) \end{bmatrix} \quad \textbf{(2.14)}$$

The NED to Body directional cosine matrix is utilized to convert vehicle states typically expressed in the body frame such as angular rates and body velocity into the local NED frame which is the local navigation coordinate system. Equation 2.5 showed how this rotational matrix can be computed from the aircraft's Euler angles. In the case of quaternions, the rotational matrix can be computed using the equation below.

$$R_{b/n} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 \cdot q_2 + q_0 \cdot q_3) & 2(q_1 \cdot q_3 - q_0 \cdot q_2) \\ 2(q_1 \cdot q_2 - q_0 \cdot q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 \cdot q_3 + q_0 \cdot q_1) \\ 2(q_1 \cdot q_3 + q_0 \cdot q_2) & 2(q_2 \cdot q_3 - q_0 \cdot q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad \textbf{(2.15)}$$

The rotational matrix can be utilized to extract the Euler angles allowing for a simple conversion from quaternions to Euler angles. Equation 2.16 demonstrates how this is accomplished.

$$\phi = tan^{-1}\left(\frac{R_{23}}{R_{33}}\right) \quad \textbf{(2.16)}$$

$$\theta = -sin^{-1}(R_{13}) \quad \textbf{(2.17)}$$

$$\psi = tan^{-1}\left(\frac{R_{12}}{R_{11}}\right) \quad \textbf{(2.18)}$$

2.3.4 Translational Kinematics

The translational kinematic equations are mainly responsible of calculating the position of the aircraft in relation to the local navigation frame. Once the body velocity and the rotational matrix are calculated as explained in sections 2.3.2 and 2.3.3, the aircraft velocity in the local navigation frame can be determined as follows:

$$\dot{P}_{NED} = \left(R_{b/n}\right)^{T} v_b \qquad \textbf{(2.19)}$$

2.4 Basic Aerodynamics

The study of aircraft motion requires good understanding of the field of aerodynamics since the majority of the forces and moments acting on the aircraft body are generated by the air surrounding it. The four main forces acting on an aircraft are weight, thrust, lift, and drag (see Figure 2-5).

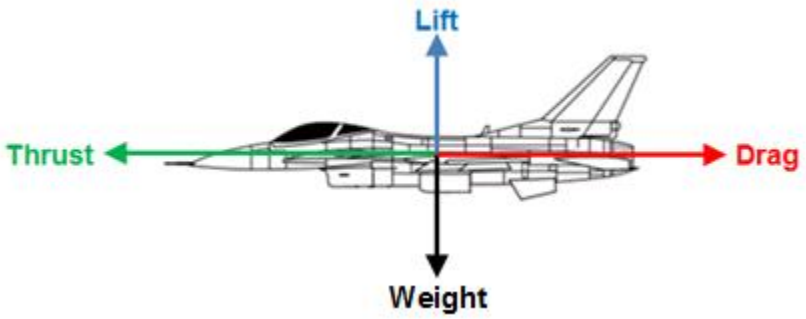

Figure 2-5: Basic Aircraft Forces

Weight is created by the Earth's gravitational pull as explained in section 2.1.1. Thrust is an aerodynamic force generated by the engine which propels the aircraft forward. The lift and drag forces are aerodynamic forces mainly generated by the fuselage and the wing airfoils. A sample airfoil cross-section is shown in Figure 2-6. As the airfoil moves through the air, a pressure differential between the top and the bottom is generated, creating a force perpendicular to the freestream velocity vector which is known as lift. The lift force opposes weight allowing the aircraft to stay airborne. Drag, on the other hand, is a force that is parallel to the freestream velocity vector. Drag is created by the air resistance generated as the airfoil moves through the air opposing its motion. Lift and drag are heavily dependent on the angle created between the freestream velocity vector and the chord line (the imaginary line between the leading edge and the trailing edge of the airfoil). This angle is known as the angle of attack and is represented with the symbol α.

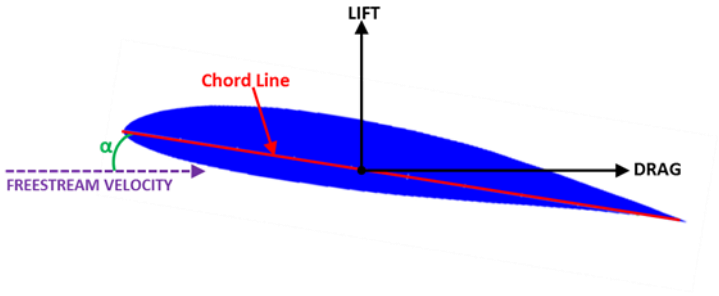

Figure 2-6: Wing Airfoil

Another important aerodynamic force is known as the sideforce which acts along the aircraft's lateral axis. For symmetric aircraft, this force is mainly created by sideslipping

motion and by deflections on the ailerons and the rudder surfaces. Sideslipping motion is caused when the aircraft's longitudinal axis is not aligned with the freestream velocity vector. The angle between these two vectors is known as the sideslip angle and it is represented by the symbol β (shown in Figure 2-7).

Besides the aerodynamic forces, the aircraft motion is also affected by angular moments in all three axes. These moments are known as rolling, pitching, and yawing moments. Rolling and yawing moment are typically created by the sideslip, the control action of the ailerons and the rudder, and the damping moments resisting rolling and yawing motion. The pitching moment, on the other hand, is mostly dependent on the angle of attack, the elevator deflection, the damping moment resisting the pitching motion, and the contribution of the lift force generated by the lever arm effect of the separation between the center of pressure and the center of gravity. An illustration of the aerodynamic moments and their notation is shown in Figure 2-7.

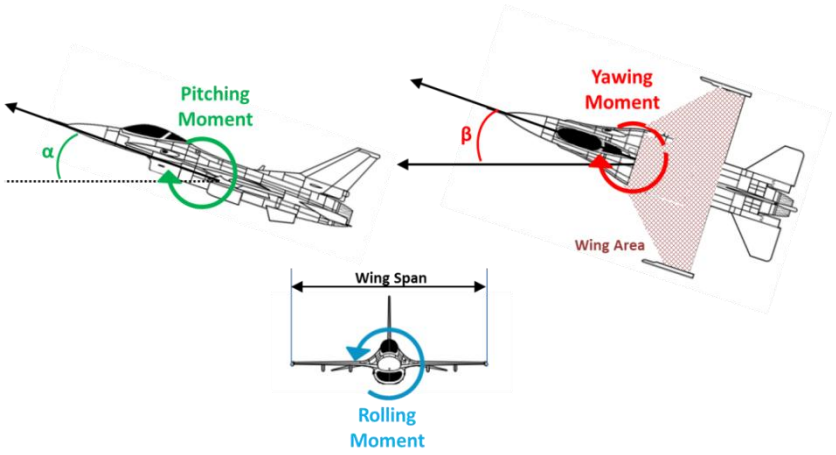

Figure 2-7: Definition of Aerodynamic Moments

The aerodynamic forces and moments can be expressed in terms of dimensionless coefficients known as the aerodynamic coefficients. These coefficients depend on the shape of the airfoil and the aerodynamic angles (α and β). Utilizing these coefficients, the aerodynamic forces can be then be defined as the product of the air dynamic pressure ($\bar{q}$), the wing reference area (S), and the lift, drag, and sideforce coefficients. The aerodynamic moments can be defined in a similar manner with the addition of a reference length in order to obtain the proper units. The reference length utilized for the rolling and yawing moments is the aircraft's wingspan (b) while the reference length utilized for the pitching moment is the mean aerodynamic chord ($\bar{c}$). Equations 2.20 to 2.25 show how to calculate the aerodynamic forces and moments.

$$F_{XA} = \bar{q} \cdot S \cdot C_X \quad \textbf{(2.20)}$$

$$F_{YA} = \bar{q} \cdot S \cdot C_Y \quad \textbf{(2.21)}$$

$$F_{ZA} = \bar{q} \cdot S \cdot C_Z \quad \textbf{(2.22)}$$

$$L = \bar{q} \cdot S \cdot b \cdot C_l \quad \textbf{(2.23)}$$

$$M = \bar{q} \cdot S \cdot \bar{c} \cdot C_m \quad \textbf{(2.24)}$$

$$N = \bar{q} \cdot S \cdot b \cdot C_n \quad \textbf{(2.25)}$$

CHAPTER 3: **MODELING THE NONLINEAR UAV**

The first step towards the development of a new guidance and control algorithm is to obtain a better understanding of the plant that needs to be stabilized and controlled. This can be accomplished by applying the concepts of modeling and simulation which have become an emerging discipline widely used in the engineering community due their ability of reducing the risks involved in a new design in a cost effective manner.

The term modeling is used to define the process of creating a physical, mathematical, or logical representation of a system. In the particular case of fixed wing aircraft, a mathematical model of the aircraft can be created that includes important features such as aerodynamics, propulsion, and the well-known 6-DOF equations of motion explained in the previous chapter. The term simulation, on the other hand, is the implementation of the mathematical model over time. Digital simulations are useful for testing the developed control algorithms and analyzing their performance.

One of the main advantages of modeling and simulation is that it allows testing the proposed solution without the need to build any hardware. One of its main disadvantages, however, is that simulations are prone to errors and can inadvertently omit important dynamics of the system, especially when dealing with complex ones. To minimize this inherited problem, models are often submitted to a verification, validation, and accreditation (VV&A) procedure where the model is checked for any potential errors and its results are compared against the outputs of the real system. Validated models

provide a higher degree of confidence on the simulation results which is the main reason behind the selection of a high fidelity NASA F-16 model to represent the UAV plant in this thesis. This chapter presents an overview of the selected aircraft model and provides a detailed explanation on how this model was implemented in the Matlab/Simulink environment.

3.1 Nonlinear 6-DOF Model Overview

The selected UAV model is based on a high fidelity, nonlinear 6-DOF model of an F-16 fighter aircraft developed by NASA. The model, described in [2], has been used to develop a simulation of the F-16 aircraft in the Matlab/Simulink platform. The main constant parameters of the F-16 model are shown in Table 3-1 and an overview of the Simulink model is shown in Figure 3-1.

Table 3-1: F-16 Nonlinear Model Constants

Parameter	Symbol	Value	Units
Mass	m	9295.44	kg
Moments of Inertia	J_{XX}	12874.8	kg·m^2
	J_{YY}	75673.6	kg·m^2
	J_{ZZ}	85552.1	kg·m^2
	J_{XZ}	1331.4	kg·m^2
Wing Span	b	9.144	m
Wing Area	S	27.87	m^2
Mean Aerodynamic Chord	$\bar{c}$	3.45	m
C.G. Location	X_{CG}	$0.3\bar{c}$	m
Reference C.G.	X_{CGR}	$0.35\bar{c}$	m
Engine Angular Momentum	h_E	216.9	kg·m^2/s

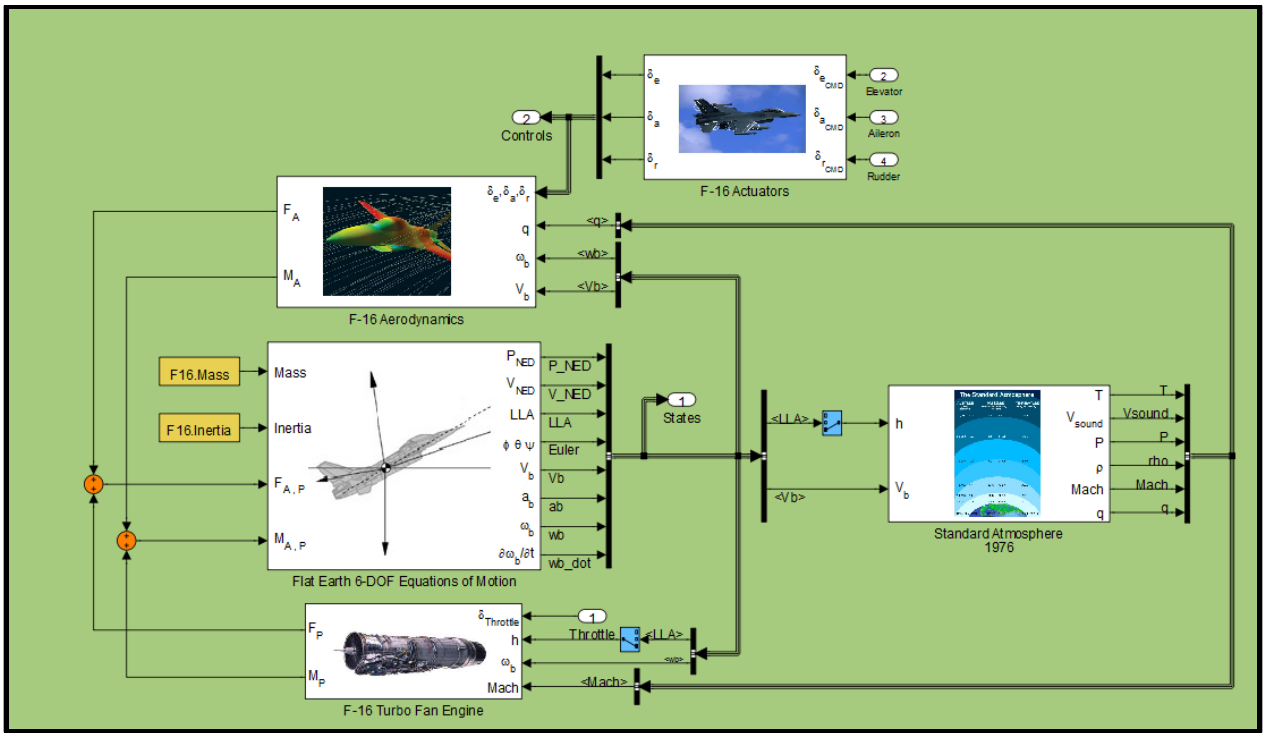

Figure 3-1: Nonlinear 6-DOF UAV Simulink Model

The UAV model is composed of an atmospheric model, an actuator model, an aerodynamics model, a propulsion model, and the 6-DOF equations of motion. The model only requires the pilot commands as inputs and generates the typical position, velocity, and attitude outputs. A list of the model inputs/outputs and their unit definitions are provided in Table 3-2.

Table 3-2: List of UAV model Inputs and Outputs

Inputs		Outputs	
Variable	**Units**	**Variable**	**Units**
Throttle Cmd	unitless (0-1)	Position NED	meters
Elevator Cmd	deg	Velocity NED	m/s
Aileron Cmd	deg	WGS-84 Coordinates	deg,deg,m
Rudder Cmd	deg	Euler Angles	deg
		Body Velocity	m/s
		Body Acceleration	m/s^2
		Angular Rates	deg/s
		Angular Acceleration	deg/s^2

The submodels contained inside the UAV Simulink model must be called in a specific order to satisfy the interdependencies among them. Once the vehicle states (position, velocity, attitude, etc.) are initialized, the simulation runs six recursive steps until it ends. The first step consists of reading the commands for the throttle and the control surfaces. In the next step, the actuator models are executed to calculate the position of the control surfaces based on the commanded inputs. The following step calls the atmospheric model which is responsible of calculating important environmental parameters such as air density and pressure based on the aircraft's altitude. In the fourth step, the aerodynamics model calculates the aero forces and moments exerted on the vehicle body using the aircraft's current state, the position of the control surfaces, and the atmospheric parameters. In the fifth step, the engine thrust is updated by calling the propulsion model. Finally, the aircraft states for the next time step are computed by adding all the forces and moments acting on the aircraft and using the 6-DOF equations of motion. Figure 3-2 shows a flow diagram of the UAV Simulink model. The following sections describe the details associated with each of these submodels.

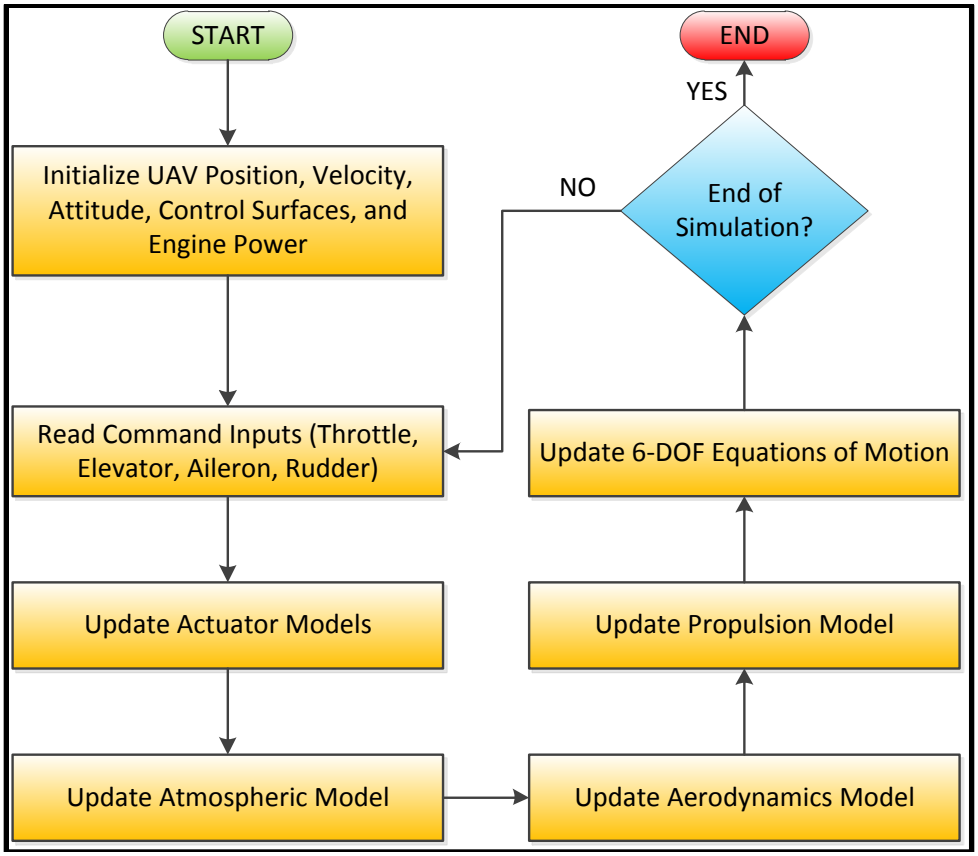

Figure 3-2: UAV Model Flow Diagram

3.2 Actuator Models

The actuator models simulate the dynamics of the control surfaces that are used to steer and stabilize the aircraft. The F-16, like most fixed wing aircraft, is equipped with three different types of control surfaces which are the elevator, the ailerons, and the rudder. The elevator is a control surface typically attached to the horizontal stabilizer of the aircraft that controls the pitch motion of the vehicle. In the case of the F-16, the entire horizontal stabilizer acts as the elevator which is known as a stabilator. The

ailerons are installed in the trailing edge of the aircraft wing and control the roll motion. The rudder, which is located in the vertical stabilizer, controls the yaw motion.

The dynamics of the servo actuators used to control these surfaces are represented as simple first order filters that simulate the inherited lag of the electro-hydraulic actuators. The models also include saturation blocks to limit the maximum deflections of each surface and rate limiter blocks that simulate the maximum mechanical speed of these devices. Figure 3-3 shows the implementation of the model in Simulink. The specific characteristics of each control surface are shown in Table 3-3.

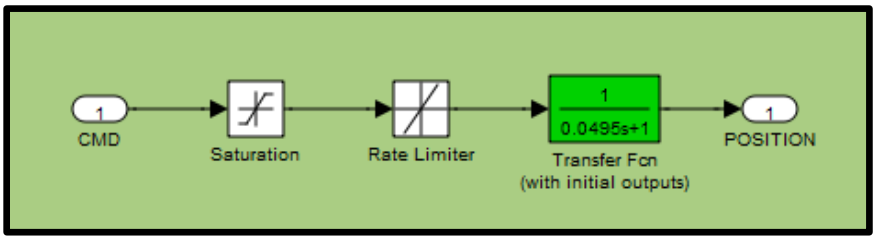

Figure 3-3: Simulink Actuator Model

Table 3-3: F-16 Control Surface Characteristics

Control Surface	Maximum Deflections	Time Lag Constant	Maximum Rate
Elevator	±25.0°	0.0495s	±60°
Aileron	±21.5°	0.0495s	±80°
Rudder	±30.0°	0.0495s	±120°

3.3 Atmospheric Model

The selected atmospheric model is based on the U.S. Standard Atmosphere Model of 1976. This is the latest version of the static atmospheric models that describe how the ideal gas properties of the atmosphere change as a function of altitude. The model calculates atmospheric temperature, pressure, and density at mean global and annual mid-latitude (45°N) conditions. The model assumes dry air that is homogeneously mixed at altitudes below 86km. These limitations are addressed by newer, dynamics models by taking into account other parameters such as geodetic position and time of the year; however, they are fairly complex making the U.S. Standard Atmosphere the best candidate for simulations that don't require highly accurate atmospheric modeling.

The Standard Atmosphere of 1976 divides the atmosphere in 8 main layers and provides the baseline temperature, temperature lapse rate, and static pressure for each layer. Table 3-4 shows the look-up tables that are defined by this atmospheric model and Table 3-5 shows the definitions for all the inputs, outputs, and internal parameters used by the model.

Table 3-4: U.S. Standard Atmosphere of 1976 Tables

Layer	Altitude (m)	Temperature (°K)	Lapse Rate (°K/m)	Static Pressure (Pa)
Troposphere	0	288.15	-0.0065	101,325
Tropopause	11,000	216.65	0	22,632.1
Lower Stratosphere	20,000	216.65	0.001	5,474.89
Upper Stratosphere	32,000	228.65	0.0028	868.019
Stratopause	47,000	270.65	0	110.906
Lower Mesosphere	51,000	270.65	-0.0028	66.9389
Upper Mesosphere	71,000	214.65	-0.002	3.9564
Mesopause	84,852	186.946	0	0.3734

Table 3-5: U.S. Standard Atmosphere of 1976 Parameters and Constants

Parameter	Description	Value	Units
h	Current Altitude	Model Input	meters
V_b	Aircraft Body Velocity	Model Input	m/s
T	Temperature	Model Output	Kelvin
P_S	Static Pressure	Model Output	Pascal
V_S	Speed of Sound	Model Output	m/s
Ma	Mach	Model Output	unitless
ρ	Air Density	Model Output	kg/m^3
$\bar{q}$	Dynamic Pressure	Model Output	Pascal
P_0	Atmospheric Layer Baseline Pressure	Output from Look-Up Table	Pascal
L_R	Atmospheric Layer Lapse Rate	Output from Look-Up Table	K/m
h_0	Atmospheric Layer Baseline Altitude	Output from Look-Up Table	meters
T_0	Atmospheric Layer Baseline Temperature	Output from Look-Up Table	Kelvin
g	Local gravity magnitude	9.80665	m/s^2
M_m	Molar Mass of Dry Air	0.0289644	kg/mol
R	Universal Gas Constant	8.31447	J/(mol•K)
γ	Adiabatic Index	1.4	unitless

The atmospheric model calculates the temperature based on simple linear interpolation between the layers, however the pressure is computed logarithmically. To accomplish this, equation 3.1 is utilized where the baseline atmospheric parameters (P_0, T_0, h_0, and L_R) are provided by the look-up tables which are programmed to output the values of the layer below the current aircraft altitude. Once the temperature and pressure are calculated, the other model output can be calculated using equations 3.2 to 3.5. An illustration of the Simulink model implementation is shown in Figure 3-4.

Static Pressure: $$P_S = P_0\left(1 + \frac{L_R \cdot (h - h_0)}{T_0}\right)^{\frac{g \cdot M_m}{-R \cdot L_R}} \quad \textbf{(3.1)}$$

Air Density: $$\rho = \frac{P_S}{T} \cdot \frac{M_m}{R} \quad \textbf{(3.2)}$$

Dynamic Pressure: $$\bar{q} = \frac{1}{2}\rho|V_b|^2 \quad \textbf{(3.3)}$$

Speed of Sound: $$V_S = \sqrt{\frac{\gamma RT}{M_m}} \quad \textbf{(3.4)}$$

Mach: $$\mathrm{Ma} = \frac{|V_b|}{V_S} \quad \textbf{(3.5)}$$

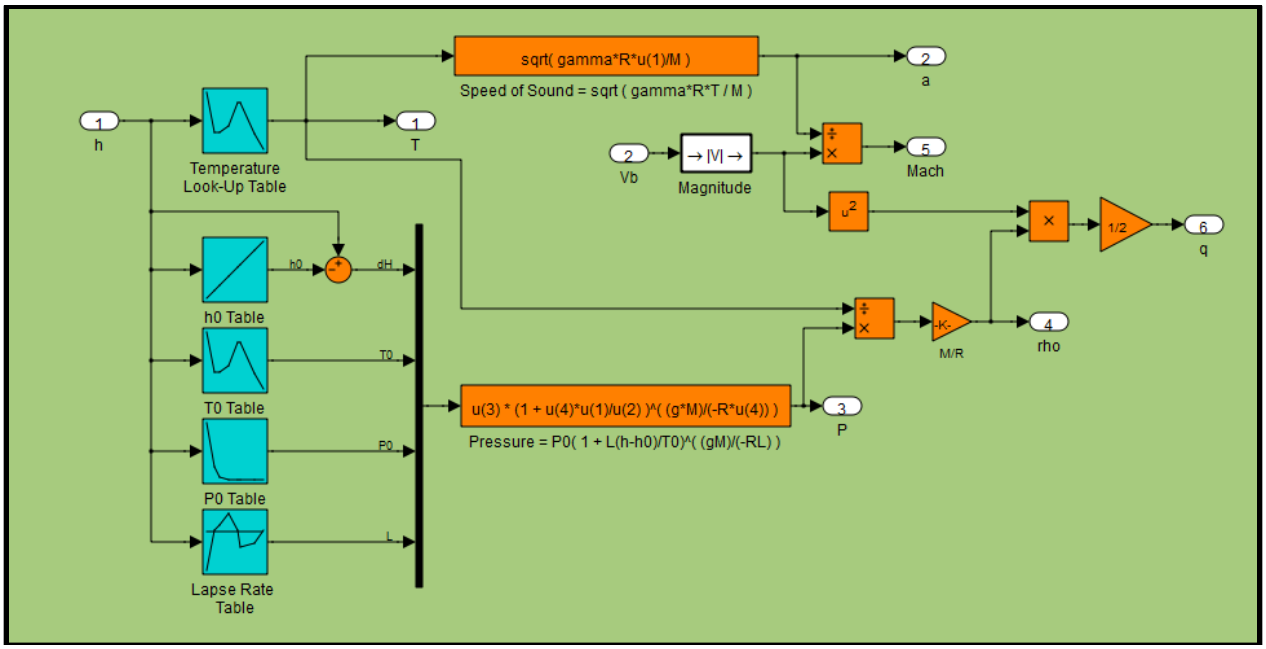

Figure 3-4: Standard Atmosphere 1976 Simulink Model

3.4 Aerodynamics Model

The NASA technical paper [2] includes high fidelity aerodynamic wind tunnel data that is valid for subsonic flight and for an ample angle of attack (AOA) range (-20° to 90°). This high fidelity aerodynamic model is very complex consisting of more than 50 look-up tables. A simpler version of this model was derived by Steven and Lewis [3] creating a reduced data set (AOA: -10° to 45°) which still provides adequate fidelity for the purposes of this thesis. The aerodynamics model presented in this section is mainly based in this simplified model. An overview of the Simulink aerodynamics model is presented in Figure 3-5.

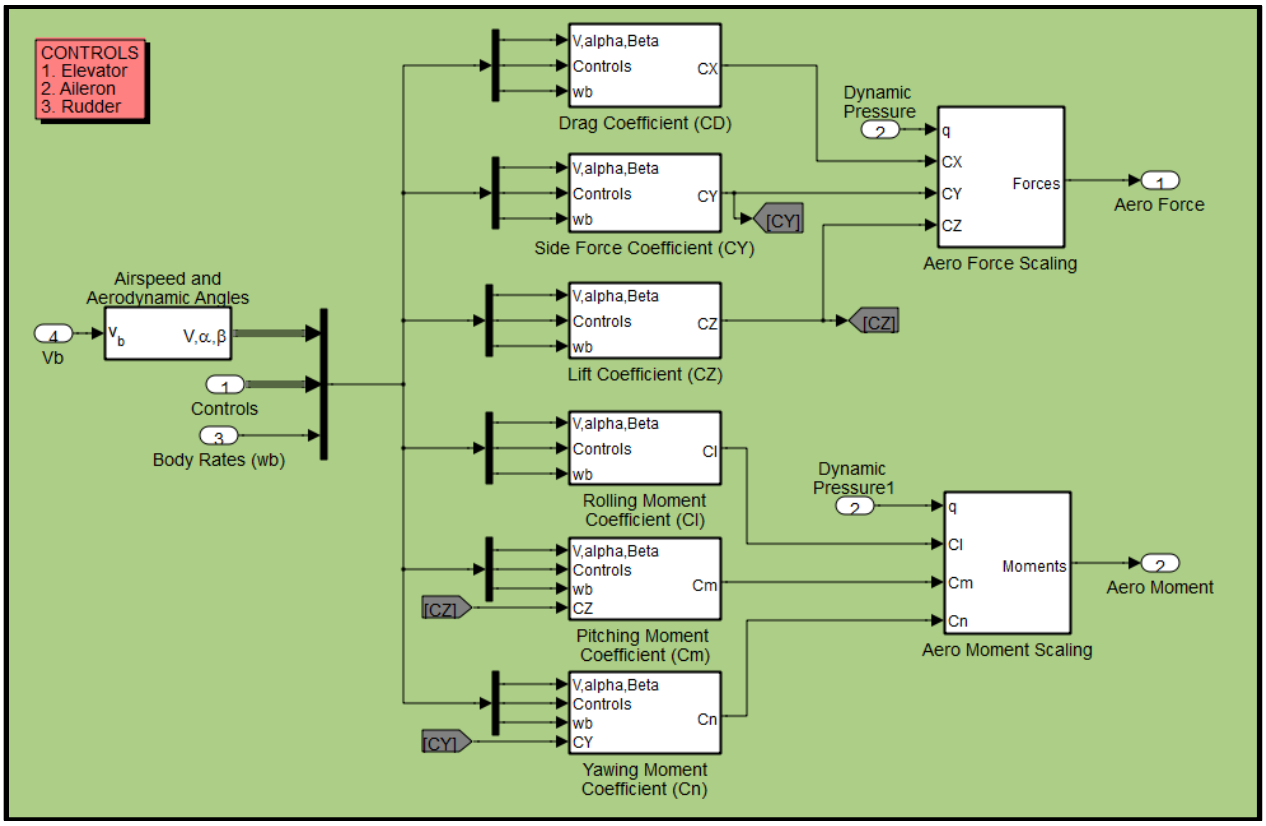

Figure 3-5: Aerodynamics Simulink Model

The model is composed of nine different modules. In the first module, the vehicle airspeed and the aerodynamic angles are calculated from the body velocity vector based on the equations shown below. The Simulink implementation is shown in Figure 3-6.

Velocity: $$V_T = \sqrt{U^2 + V^2 + W^2} \quad \textbf{(3.6)}$$

Angle of Attack: $$\alpha = tan^{-1}\frac{W}{U} \quad \textbf{(3.7)}$$

Sideslip Angle: $$\beta = sin^{-1}\frac{V}{V_T} \quad \textbf{(3.8)}$$

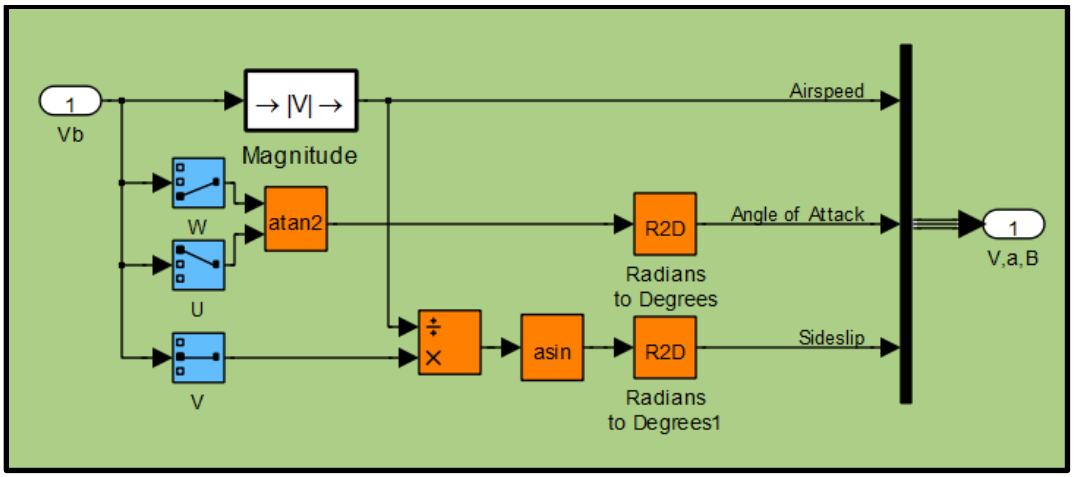

Figure 3-6: Airspeed and Aerodynamic Angles Simulink Implementation

The next six modules are responsible of calculating the aerodynamic coefficients which are known as drag, sideforce, lift, rolling moment, pitching moment, and yawing moment coefficients. The following sub-sections provide in depth detail on how to calculate these coefficients.

3.4.1 Aerodynamic Damping Derivatives

The rotational motion of an aircraft creates a restoring moment caused by the changed relative airflow. This restoring moment opposes the direction of the angular rotation attempting to nullify the angular rate in that axis. The coefficients that relate this restoring moment to the angular rate are known as damping derivatives. The mathematical formula used to calculate these damping force or damping moment coefficients is shown in Equation 3.9.

$$\Delta C = C(\alpha) \cdot \frac{k}{2V_T} \quad \textbf{(3.9)}$$

The equation above shows that the dimensionless restoring moment (ΔC) is equal to the multiplication of the damping coefficient and the helix angle. The helix angle is calculated as the division between a reference length and twice the magnitude of the translational velocity. This reference length is either the wingspan (b) when related to rolling and yawing rates, or the mean aerodynamic chord $(\bar{c})$ when related to pitching rate.

The F-16 aerodynamics model utilizes look-up tables for these damping coefficients which are only dependent on the angle of attack (α). Table 3-6 shows the look-up tables for all nine damping derivatives.

Table 3-6: Aerodynamic Damping Coefficients Look-Up Tables

α	C_{X_Q}	C_{Y_P}	C_{Y_R}	C_{Z_Q}	C_{l_P}	C_{l_R}	C_{m_Q}	C_{n_P}	C_{n_R}
-10°	-0.267	-0.108	0.882	-8.8	-0.360	-0.126	-7.21	0.061	-0.380
-5°	0.110	-0.108	0.852	-25.8	-0.359	-0.026	-0.54	0.052	-0.363
0°	0.308	-0.188	0.876	-28.9	-0.443	0.063	-5.23	0.052	-0.378
5°	1.340	0.110	0.958	-31.4	-0.420	0.113	-5.26	-0.012	-0.386
10°	2.080	0.258	0.962	-31.2	-0.383	0.208	-6.11	-0.013	-0.370
15°	2.910	0.226	0.974	-30.7	-0.375	0.230	-6.64	-0.024	-0.453
20°	2.760	0.344	0.819	-27.7	-0.329	0.319	-5.69	0.050	-0.550
25°	2.050	0.362	0.483	-28.2	-0.294	0.437	-6.00	0.150	-0.582
30°	1.500	0.611	0.590	-29.0	-0.230	0.680	-6.20	0.130	-0.595
35°	1.490	0.529	1.210	-29.8	-0.210	0.100	-6.40	0.158	-0.637
40°	1.830	0.298	-0.493	-38.3	-0.120	0.447	-6.60	0.240	-1.020
45°	1.210	-2.270	-1.040	-35.3	-0.100	-0.330	-6.00	0.150	-0.840

3.4.2 Drag Coefficient

The drag coefficient can be calculated using equation 3.10 which is composed of two main terms. The first term is known as the baseline drag and it is dependent on the angle of attack and the elevator deflection. It can be determined using the look-up table shown in Table 3-7. The second term is the pitch damping derivative associated with drag (see Table 3-6). A Simulink implementation of equation 3.10 is shown in Figure 3-7.

$$C_X = C_{X_0}(\alpha, \delta_e) + \frac{\bar{c}}{2V_T}\left[C_{X_Q}(\alpha) \cdot Q\right] \quad \textbf{(3.10)}$$

Table 3-7: Baseline Drag Coefficient (CX0) Look-Up Table

α \ δ_e	-25	-12.5	0.0	12.5	25
-10°	-0.099	-0.048	-0.022	-0.040	-0.083
-5°	-0.081	-0.038	-0.020	-0.038	-0.073
0°	-0.081	-0.040	-0.021	-0.039	-0.076
5°	-0.063	-0.021	-0.004	-0.025	-0.072
10°	-0.025	0.016	0.032	0.006	-0.046
15°	0.044	0.083	0.094	0.062	0.012
20°	0.097	0.127	0.128	0.087	0.024
25°	0.113	0.137	0.130	0.085	0.025
30°	0.145	0.162	0.154	0.100	0.043
35°	0.167	0.177	0.161	0.110	0.053
40°	0.174	0.179	0.155	0.104	0.047
45°	0.166	0.167	0.138	0.091	0.040

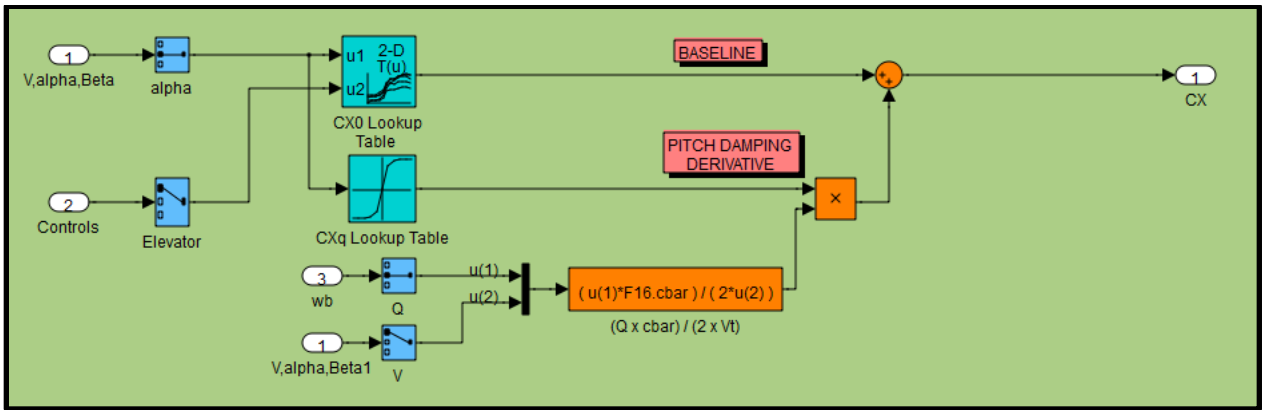

Figure 3-7: Drag Coefficient Simulink Implementation

3.4.3 Sideforce Coefficient

The sideforce coefficient can be computed using equation 3.11. The first term is the baseline sideforce. The reduced aerodynamic data set simplified this term as a linear equation dependent on the sideslip angle and the aileron and rudder deflections as shown in equation 3.12. The second term is the combination of the rolling and yawing damping derivatives (see Table 3-6). Figure 3-8 depicts the Simulink implementation for this coefficient.

$$C_Y = C_{Y_0}(\beta, \delta_a, \delta_r) + \frac{b}{2V_T}\left[C_{Y_P}(\alpha) \cdot P + C_{Y_R}(\alpha) \cdot R\right] \quad \textbf{(3.11)}$$

$$C_{Y0}(\beta, \delta_a, \delta_r) = -0.02\beta + 0.021\frac{\delta_a}{21.5} + 0.086\frac{\delta_r}{30} \quad \textbf{(3.12)}$$

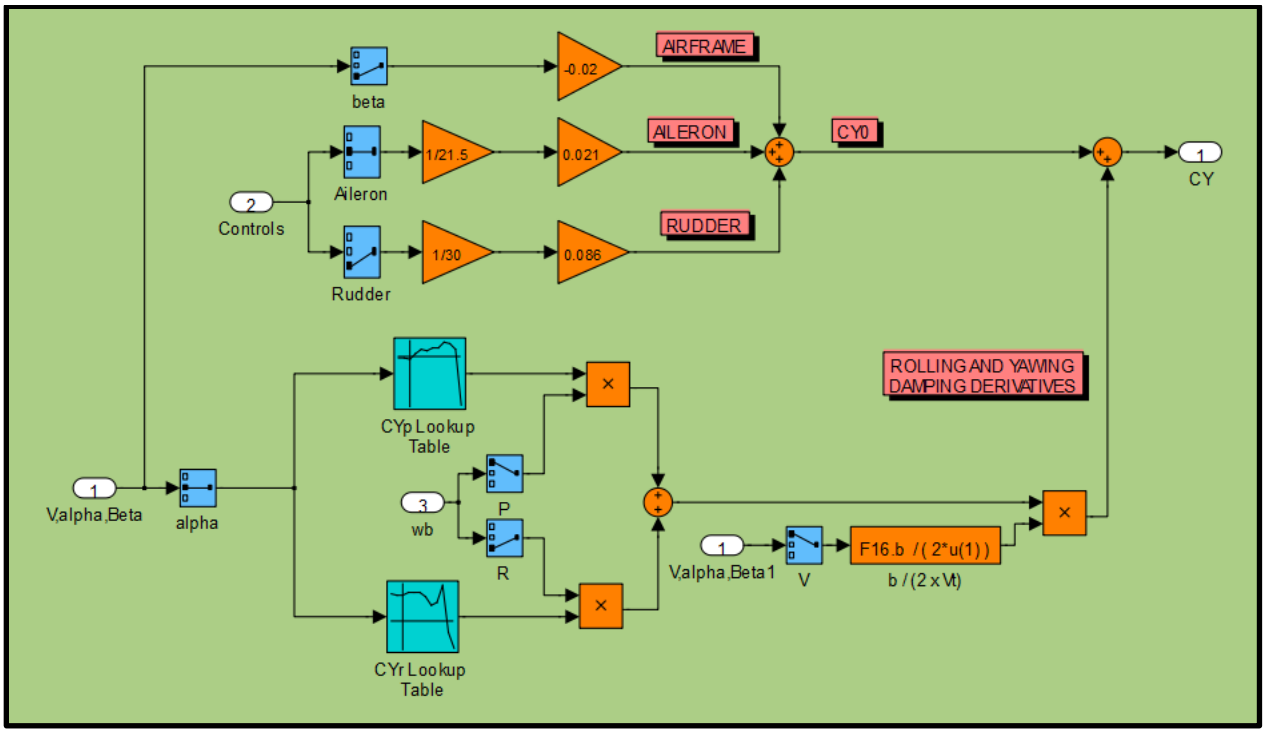

Figure 3-8: Sideforce Coefficient Simulink Implementation

3.4.4 Lift Coefficient

The lift coefficient formula, shown in equation 3.13, is composed of three terms. The first term is the baseline lift coefficient which is dependent on the aerodynamic angles. Its relationship with the angle of attack is determined using a look-up table shown in Table 3-8. The second term is the contribution of the elevator deflection and the third term is the pitch damping derivative (see Table 3-6). The Simulink implementation of this coefficient is shown in Figure 3-9.

$$C_Z = C_{Z_0}(\alpha)\left[1 - \left(\beta \cdot \frac{\pi}{180}\right)^2\right] - 0.19\left(\frac{\delta_e}{25}\right) + \frac{\bar{c}}{2V_T}\left[C_{Z_Q}(\alpha) \cdot Q\right] \qquad \textbf{(3.13)}$$

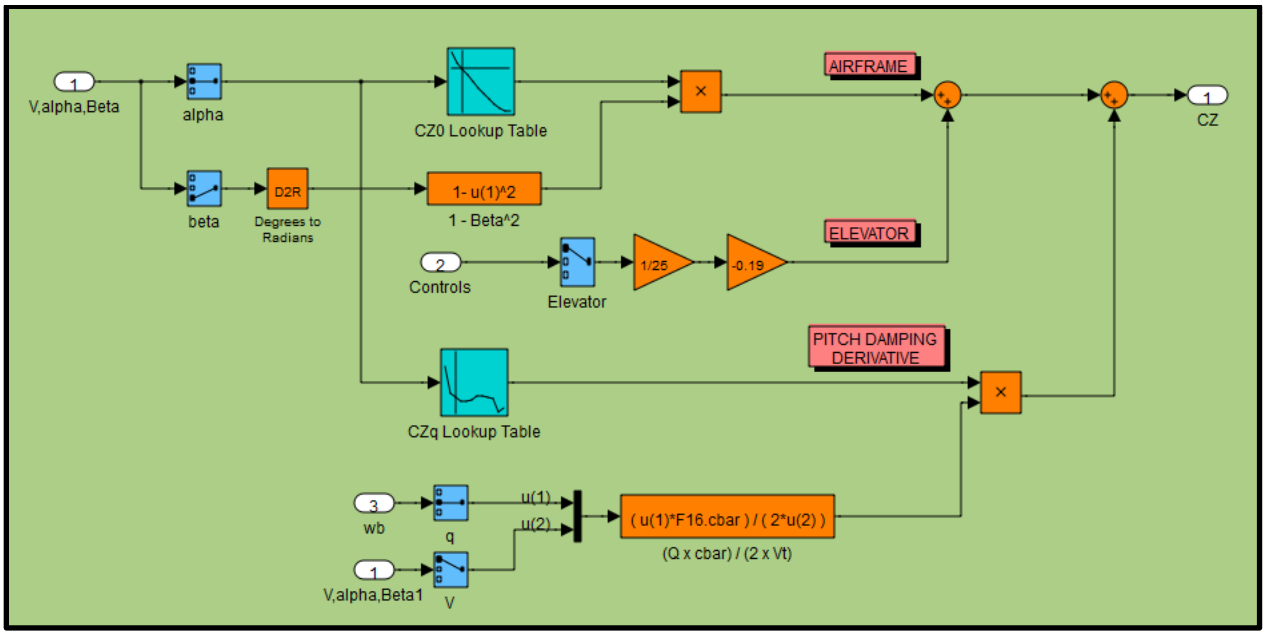

Figure 3-9: Lift Coefficient Simulink Implementation

Table 3-8: Baseline Lift Coefficient (CZ0) Look-Up Table

α	C_{z_0}
-10°	0.770
-5°	0.241
0°	-0.100
5°	-0.416
10°	-0.731
15°	-1.053
20°	-1.366
25°	-1.646
30°	-1.917
35°	-2.120
40°	-2.248
45°	-2.229

3.4.5 Rolling Moment Coefficient

The rolling moment coefficient can be computed using equation 3.14 and it is composed of four main terms. The first term is the baseline rolling moment which is dependent on

the aerodynamic angles. This term can be determined using Table 3-9. The second and third terms are known as the roll control derivatives since both the aileron and rudder deflections can directly affect the rolling moment. These two parameters can be determined using Table 3-10 and Table 3-11 respectively. The last term is the combination of the rolling and yawing damping derivative (see Table 3-6). The Simulink implementation is shown in Figure 3-10.

$$C_l = C_{l_0}(\alpha, |\beta|) \cdot sign(\beta) + C_{l_{\delta_a}}(\alpha, \beta)\left(\frac{\delta_a}{21.5}\right) + C_{l_{\delta_r}}(\alpha, \beta)\left(\frac{\delta_r}{30}\right) + \frac{b}{2V_T}\left[C_{l_P}(\alpha) \cdot P + C_{l_R}(\alpha) \cdot R\right] \quad \textbf{(3-14)}$$

Table 3-9: Baseline Rolling Moment (Cl0) Coefficient Look-Up Table

α | β |	0°	5°	10°	15°	20°	25°	30°
-10°	0.000	-0.001	-0.003	-0.001	0.000	0.007	0.009
-5°	0.000	-0.004	-0.009	-0.010	-0.010	-0.010	-0.011
0°	0.000	-0.008	-0.017	-0.020	-0.022	-0.023	-0.023
5°	0.000	-0.012	-0.024	-0.030	-0.034	-0.034	-0.037
10°	0.000	-0.016	-0.030	-0.039	-0.047	-0.049	-0.050
15°	0.000	-0.019	-0.034	-0.044	-0.046	-0.046	-0.047
20°	0.000	-0.020	-0.040	-0.050	-0.059	-0.068	-0.074
25°	0.000	-0.020	-0.037	-0.049	-0.061	-0.071	-0.079
30°	0.000	-0.015	-0.016	-0.023	-0.033	-0.060	-0.091
35°	0.000	-0.008	-0.002	-0.006	-0.036	-0.058	-0.076
40°	0.000	-0.013	-0.010	-0.014	-0.035	-0.062	-0.077
45°	0.000	-0.015	-0.019	-0.027	-0.035	-0.059	-0.076

Table 3-10: Aileron Roll Control Derivative (Clda) Look-Up Table

α \ β	-30°	-20°	-10°	0°	10°	20°	30°
-10°	-0.041	-0.041	-0.042	-0.040	-0.043	-0.044	-0.043
-5°	-0.052	-0.053	-0.053	-0.052	-0.049	-0.048	-0.049
0°	-0.053	-0.053	-0.052	-0.051	-0.048	-0.048	-0.047
5°	-0.056	-0.053	-0.051	-0.052	-0.049	-0.047	-0.045
10°	-0.050	-0.050	-0.049	-0.048	-0.043	-0.042	-0.042
15°	-0.056	-0.051	-0.049	-0.048	-0.042	-0.041	-0.037
20°	-0.082	-0.066	-0.043	-0.042	-0.042	-0.020	-0.003
25°	-0.059	-0.043	-0.035	-0.037	-0.036	-0.028	-0.013
30°	-0.042	-0.038	-0.026	-0.031	-0.025	-0.013	-0.010
35°	-0.038	-0.027	-0.016	-0.026	-0.021	-0.014	-0.003
40°	-0.027	-0.023	-0.018	-0.017	-0.016	-0.011	-0.007
45°	-0.017	-0.016	-0.014	-0.012	-0.011	-0.010	-0.008

Table 3-11: Rudder Roll Control Derivative (Cldr) Look-Up Table

α \ β	-30°	-20°	-10°	0°	10°	20°	30°
-10°	0.005	0.007	0.013	0.018	0.015	0.021	0.023
-5°	0.017	0.016	0.013	0.015	0.014	0.011	0.010
0°	0.014	0.014	0.011	0.015	0.013	0.010	0.011
5°	0.010	0.014	0.012	0.014	0.013	0.011	0.011
10°	-0.005	0.013	0.011	0.014	0.012	0.010	0.011
15°	0.009	0.009	0.009	0.014	0.011	0.009	0.010
20°	0.019	0.012	0.008	0.014	0.011	0.008	0.008
25°	0.005	0.005	0.005	0.015	0.010	0.010	0.010
30°	0.000	0.000	-0.002	0.013	0.008	0.006	0.006
35°	-0.005	0.004	0.005	0.011	0.008	0.005	0.014
40°	-0.011	0.009	0.003	0.006	0.007	0.000	0.020
45°	0.008	0.007	0.005	0.001	0.003	0.001	0.000

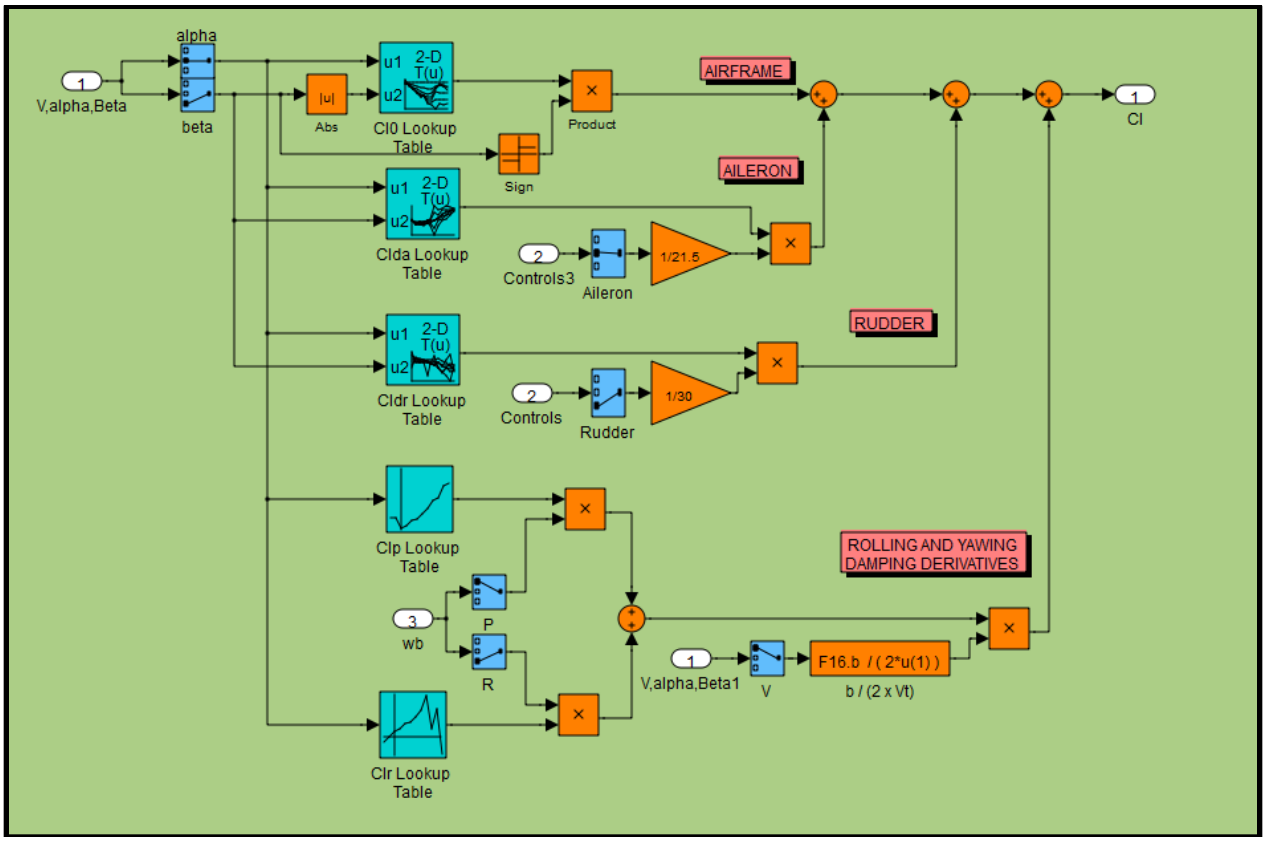

Figure 3-10: Rolling Moment Coefficient Simulink Implementation

3.4.6 Pitching Moment Coefficient

The pitching moment coefficient equation is shown in equation 3.15. The first term, known as the baseline pitching moment, is dependent on α and the elevator angle and it can be calculated using the look-up table shown in Table 3-12. The second term is the contribution of the lift force to the pitching moment caused by the displacement between the aerodynamic center of pressure and the aircraft's center of gravity. The last term is simply the pitching damping derivative. The Simulink implementation is depicted in Figure 3-11.

$$C_m = C_{m_0}(\alpha, \delta_e) + C_Z(X_{CGR} - X_{CG}) + \frac{\bar{c}}{2V_T}\left[C_{m_Q}(\alpha) \cdot Q\right] \quad \textbf{(3.15)}$$

Table 3-12: Baseline Pitching Moment Coefficient (Cm0) Look-Up Table

α \ δ_e	-25	-12.5	0.0	12.5	25
-10°	0.205	0.081	-0.046	-0.174	-0.259
-5°	0.168	0.077	-0.020	-0.145	-0.202
0°	0.186	0.107	-0.009	-0.121	-0.184
5°	0.196	0.110	-0.005	-0.127	-0.193
10°	0.213	0.110	-0.006	-0.129	-0.199
15°	0.251	0.141	0.010	-0.102	-0.150
20°	0.245	0.127	0.006	-0.097	-0.160
25°	0.238	0.119	-0.001	-0.113	-0.167
30°	0.252	0.133	0.014	-0.087	-0.104
35°	0.231	0.108	0.000	-0.084	-0.076
40°	0.198	0.081	-0.013	-0.069	-0.041
45°	0.192	0.093	0.032	-0.006	-0.005

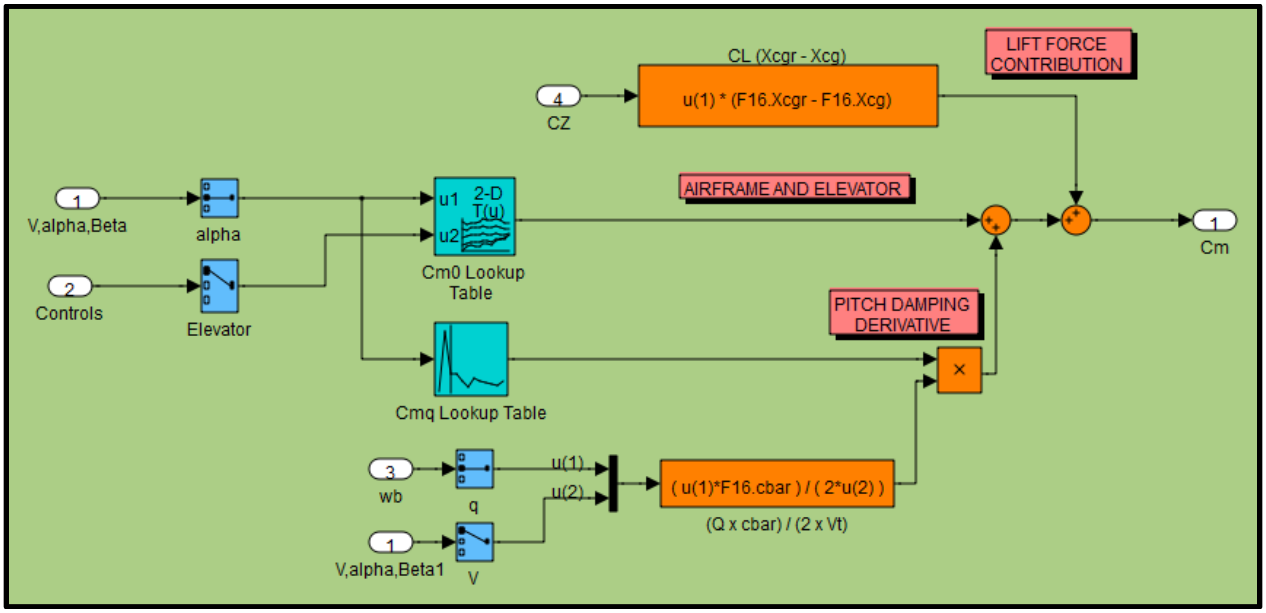

Figure 3-11: Pitching Moment Coefficient Simulink Implementation

3.4.7 Yawing Moment Coefficient

The yawing moment coefficient can be computed using equation 3.16. The first term is known as the baseline yawing moment which is dependent on the aerodynamic angles. This term can be determined using Table 3-13. The second and third terms are known as the yaw control derivatives since the aileron and rudder deflections can directly affect the yawing moment. These two parameters can be determined using Table 3-14 and Table 3-15 respectively. The last term is the combination of the rolling and yawing damping derivative (see Table 3-6). The Simulink implementation is shown in Figure 3-10.

$$C_n = C_{n_0}(\alpha, |\beta|) \cdot sign(\beta) + C_{n_{\delta_a}}(\alpha, \beta)\left(\frac{\delta_a}{21.5}\right) + C_{n_{\delta_r}}(\alpha, \beta)\left(\frac{\delta_r}{30}\right) + \frac{b}{2V_T}\left[C_{n_P}(\alpha) \cdot P + C_{n_R}(\alpha) \cdot R\right] - C_Y(X_{CGR} - X_{CG})\frac{\bar{c}}{b} \quad \textbf{(3.16)}$$

Table 3-13: Baseline Yawing Moment (Cn0) Coefficient Look-Up Table

α | β |	0°	5°	10°	15°	20°	25°	30°
-10°	0.000	0.018	0.038	0.056	0.064	0.074	0.079
-5°	0.000	0.019	0.042	0.057	0.077	0.086	0.090
0°	0.000	0.018	0.042	0.059	0.076	0.093	0.106
5°	0.000	0.019	0.042	0.058	0.074	0.089	0.106
10°	0.000	0.019	0.043	0.058	0.073	0.080	0.096
15°	0.000	0.018	0.039	0.053	0.057	0.062	0.080
20°	0.000	0.013	0.030	0.032	0.029	0.049	0.068
25°	0.000	0.007	0.017	0.012	0.007	0.022	0.030
30°	0.000	0.004	0.004	0.002	0.012	0.028	0.064
35°	0.000	-0.014	-0.035	-0.046	-0.034	-0.012	0.015
40°	0.000	-0.017	-0.047	-0.071	-0.065	-0.002	0.011
45°	0.000	-0.033	-0.057	-0.073	-0.041	-0.013	-0.001

Table 3-14: Aileron Yaw Control Derivative (Cnda) Look-Up Table

α \ β	-30°	-20°	-10°	0°	10°	20°	30°
-10°	0.001	0.002	-0.006	-0.011	-0.015	-0.024	-0.022
-5°	-0.027	-0.014	-0.008	-0.011	-0.015	-0.010	0.002
0°	-0.017	-0.016	-0.006	-0.010	-0.014	-0.004	-0.003
5°	-0.013	-0.016	-0.006	-0.009	-0.012	-0.002	-0.005
10°	-0.012	-0.014	-0.005	-0.008	-0.011	-0.001	-0.003
15°	-0.016	-0.019	-0.008	-0.006	-0.008	0.003	-0.001
20°	0.001	-0.021	-0.005	0.000	-0.002	0.014	-0.009
25°	0.017	0.002	0.007	0.004	0.002	0.006	-0.009
30°	0.011	0.012	0.004	0.007	0.006	-0.001	-0.001
35°	0.017	0.016	0.007	0.010	0.012	0.004	0.003
40°	0.008	0.015	0.006	0.004	0.011	0.004	-0.002
45°	0.016	0.011	0.006	0.010	0.011	0.006	0.001

Table 3-15: Rudder Yaw Control Derivative (Cndr) Look-Up Table

α \ β	-30°	-20°	-10°	0°	10°	20°	30°
-10°	-0.018	-0.028	-0.037	-0.048	-0.043	-0.052	-0.062
-5°	-0.052	-0.051	-0.041	-0.045	-0.044	-0.034	-0.034
0°	-0.052	-0.043	-0.038	-0.045	-0.041	-0.036	-0.027
5°	-0.052	-0.046	-0.040	-0.045	-0.041	-0.036	-0.028
10°	-0.054	-0.045	-0.040	-0.044	-0.040	-0.035	-0.027
15°	-0.049	-0.049	-0.038	-0.045	-0.038	-0.028	-0.027
20°	-0.059	-0.057	-0.037	-0.047	-0.034	-0.024	-0.023
25°	-0.051	-0.052	-0.030	-0.048	-0.035	-0.023	-0.023
30°	-0.030	-0.030	-0.027	-0.049	-0.035	-0.020	-0.019
35°	-0.037	-0.033	-0.024	-0.045	-0.029	-0.016	-0.009
40°	-0.026	-0.030	-0.019	-0.033	-0.022	-0.010	-0.025
45°	-0.013	-0.008	-0.013	-0.016	-0.009	-0.014	-0.010

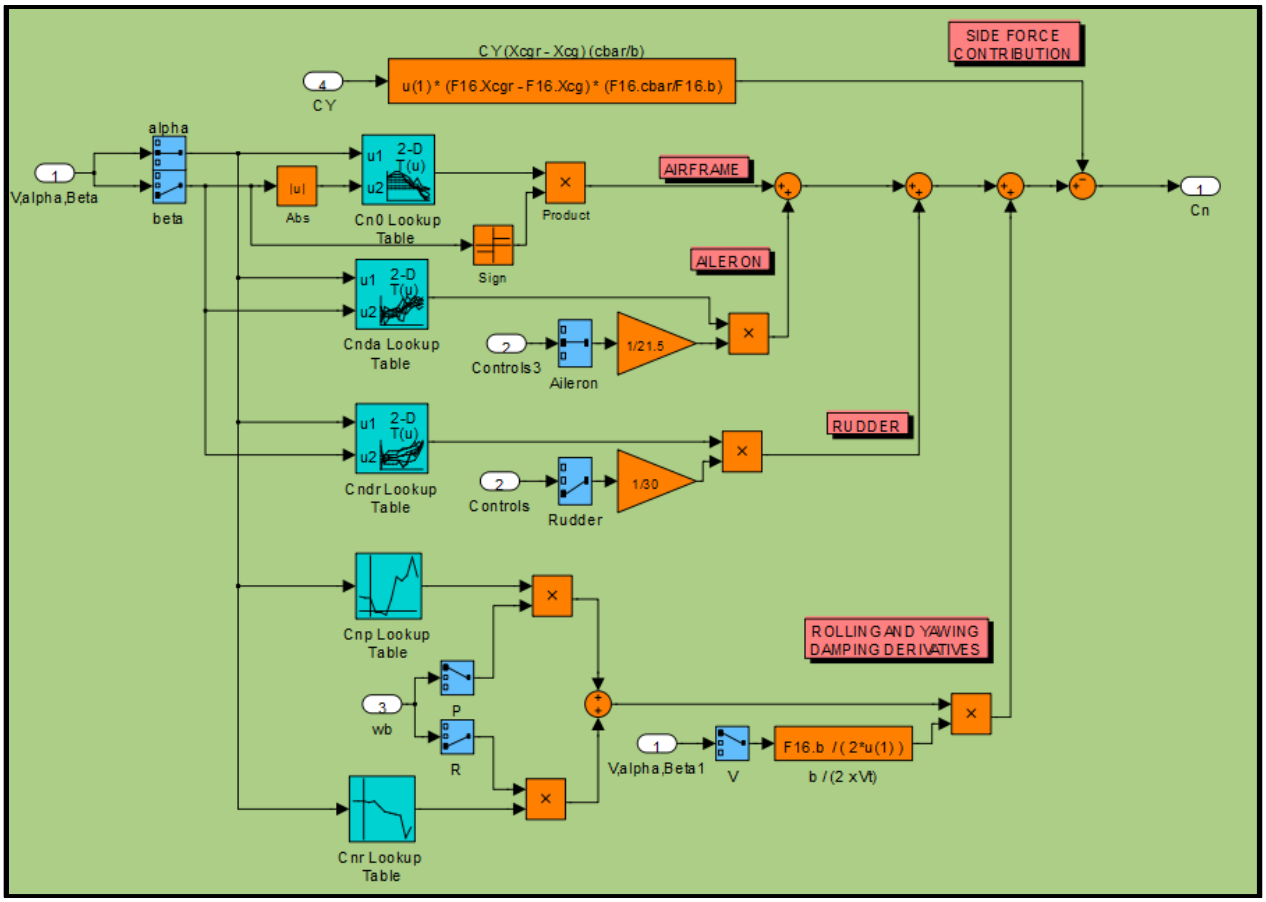

Figure 3-12: Yawing Moment Coefficient Simulink Implementation

3.4.8 Aerodynamic Forces and Moments

Once the aerodynamic coefficients are calculated, the final forces and moments can be determined using equations 2.20 to 2.25 presented in section 2.4. The Simulink implementation for these six equations is shown in Figure 3-13.

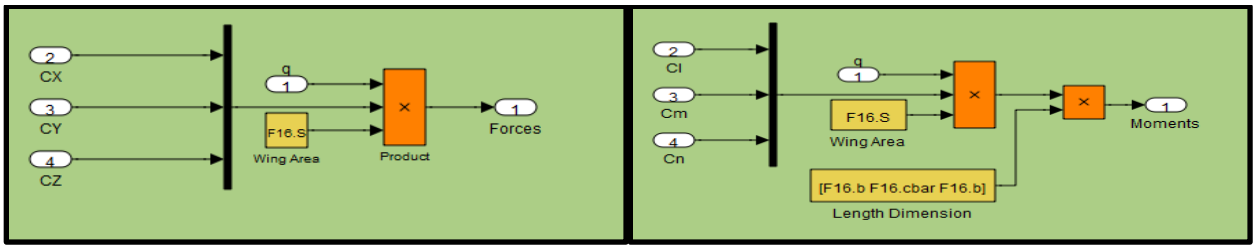

Figure 3-13: Implementation of Aerodynamic Forces and Moments Scaling

3.5 Propulsion Model

The F-16 is powered by a single afterburning turbofan engine. The propulsion model is responsible for simulating the dynamic response of this engine and for calculating its forces and moments exerted on the aircraft. The model is divided into three main sections: throttle gearing, engine power lag, and thrust look-up tables. Figure 3-14 shows a Simulink overview of the propulsion model and portrays how these sections are interconnected.

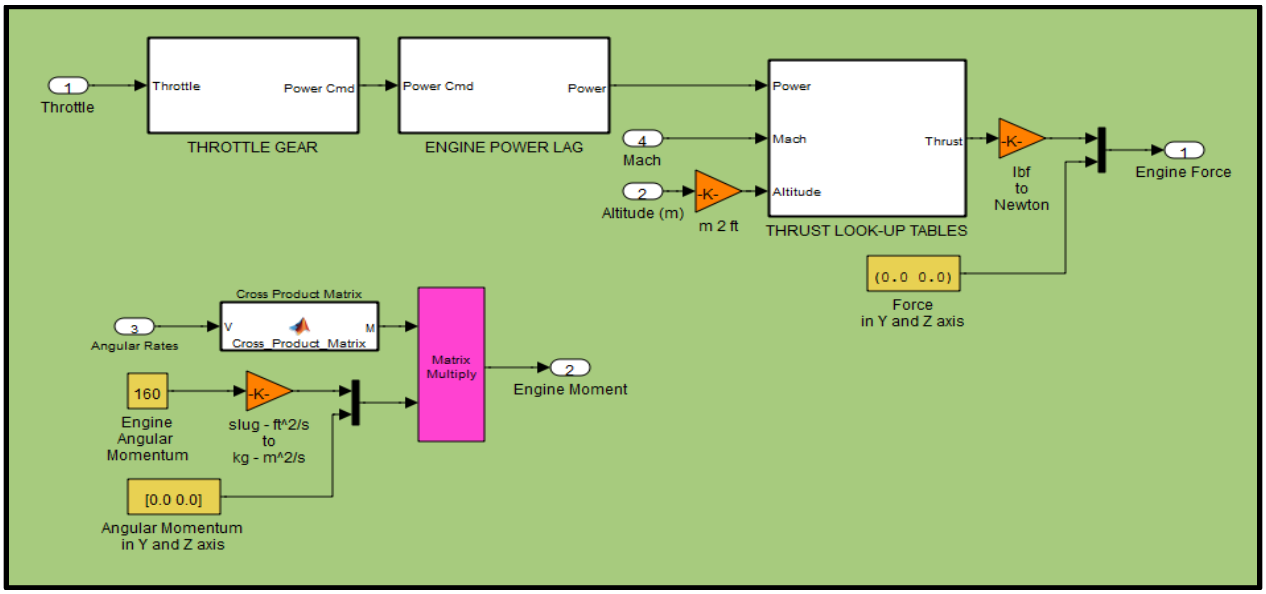

Figure 3-14: Propulsion Simulink Model

The throttle gearing section models the relationship between the commanded power and the throttle setting. These two parameters are correlated as a linear relationship with a change in slope when the engine reaches military power at a 77% throttle setting. Equation 3.17 shows the mathematical equation that relates these two parameters and it was implemented in Simulink as shown in Figure 3-15.

$$P_{CMD} = \begin{cases} 64.94\delta_T & \delta_T \leq 77\% \\ 217.38\delta_T - 117.38 & \delta_T > 77\% \end{cases} \quad \textbf{(3.17)}$$

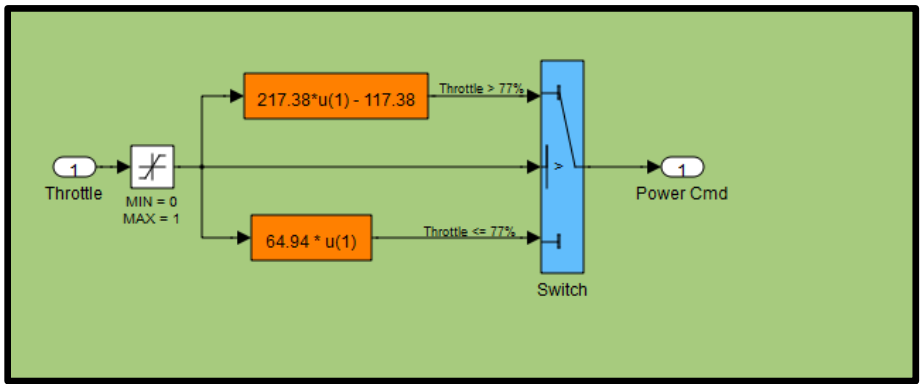

Figure 3-15: Propulsion Throttle Gearing Simulink Model

The engine power lag is modeled as simple first order lag with a variable time constant that is dependent on the current engine power (P_A) and the commanded power (P_C). The flow diagram shown in Figure 3-16 provides a functional description of the power lag model. The diagram shows that the power command is limited for specific combinations of commanded and available power. The diagram also shows that the engine time constant is defined as either 5 seconds or as a function of the difference

between the commanded and the available power. The output of this function is depicted in Figure 3-17.

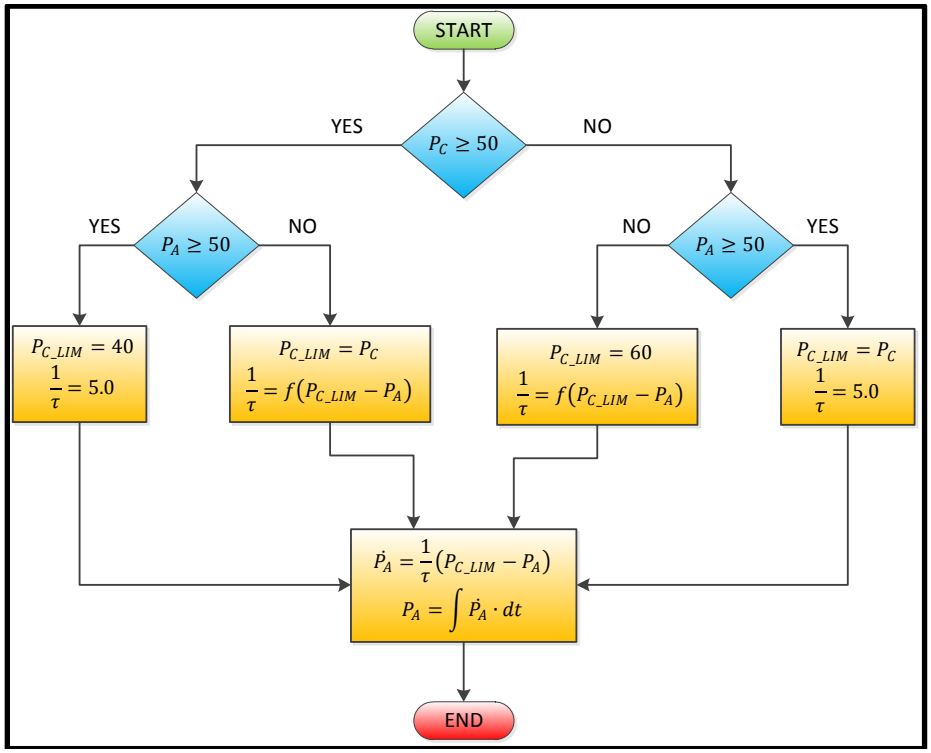

Figure 3-16: Engine Power Lag Flow Diagram

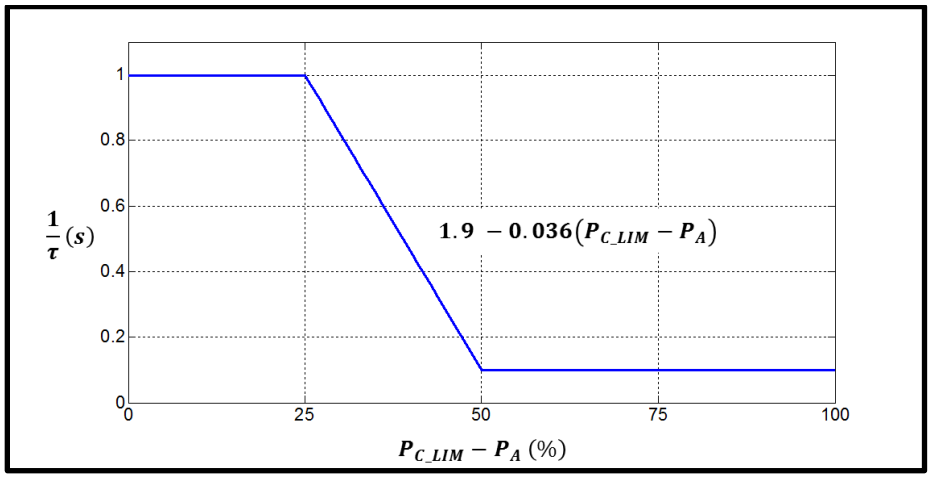

Figure 3-17: Engine Power Lag Time Constant Graph

The Simulink implementations of the engine power lag model and the time constant function are shown in Figure 3-18.

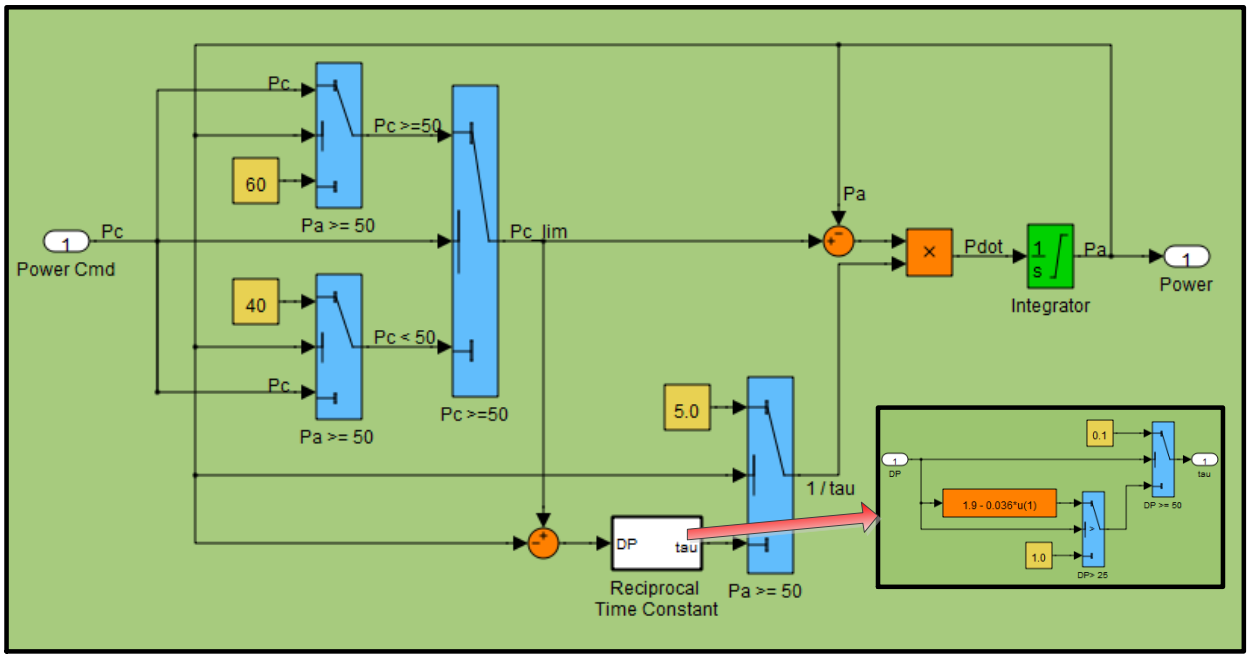

Figure 3-18: Engine Power Lag Simulink Implementation

Once the available engine power is determined, the engine thrust can be computed since it is a function of actual power, altitude, and Mach number. The NASA F-16 model calculates the thrust based on three look-up tables that represent the three main states of the engine: idle power, military power, and maximum power. Equation 3.18 shows how the thrust is calculated based on the outputs of the look-up tables and the actual engine power. A depiction of the Simulink model is shown in Figure 3-19 and the thrust tables, specified in pound force, are presented in Table 3-16, Table 3-17, and Table 3-18.

$$T = \begin{cases} T_{IDLE} + (T_{MIL} - T_{IDLE})\left(\frac{P_A}{50}\right) & P_A < 50\% \\ T_{MIL} + (T_{MAX} - T_{MIL})\left(\frac{P_A - 50}{50}\right) & \delta_T \geq 50\% \end{cases} \quad \textbf{(3-18)}$$

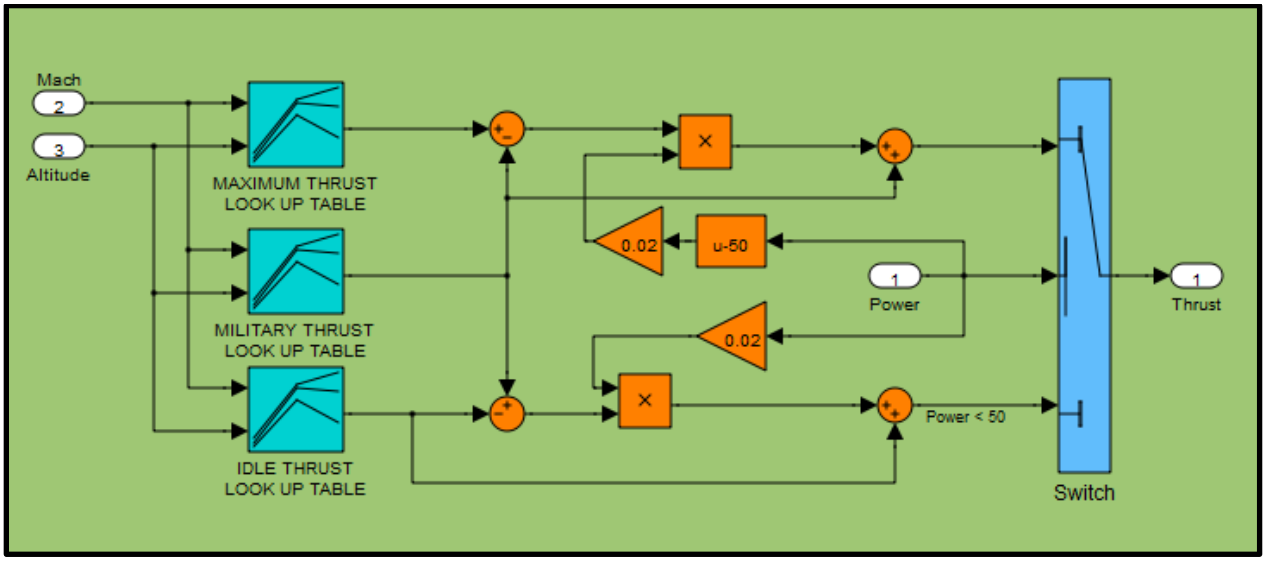

Figure 3-19: Engine Thrust Simulink Model

Table 3-16: F-16 Idle Thrust Look-Up Table in Pounds

		MACH NUMBER					
		0.0	0.2	0.4	0.6	0.8	1.0
ALTITUDE (ft)	0	1,060	635	60	-1,020	-2,700	-3,600
	10,000	670	425	25	-710	-1,900	-1,400
	20,000	880	690	345	-300	-1,300	-595
	30,000	1,140	1,010	755	350	-247	-342
	40,000	1,500	1,330	1,130	910	600	-200
	50,000	1,860	1,700	1,525	1,360	1,100	700

Table 3-17: F-16 Military Thrust Look-Up Table in Pounds

		MACH NUMBER					
		0.0	0.2	0.4	0.6	0.8	1.0
ALTITUDE (ft)	0	12,680	12,680	12,610	12,640	12,390	11,680
	10,000	9,150	9,150	9,312	9,839	10,176	9,848
	20,000	6,200	6,313	6,610	7,090	7,750	8,050
	30,000	3,950	4,040	4,290	4,660	5,320	6,100
	40,000	2,450	2,470	2,600	2,840	3,250	3,800
	50,000	1,400	1,400	1,560	1,660	1,930	2,310

Table 3-18: F-16 Maximum Thrust Look-Up Table in Pounds

		MACH NUMBER					
		0.0	0.2	0.4	0.6	0.8	1.0
ALTITUDE (ft)	0	20,000	21,420	22,700	24,240	26,070	28,886
	10,000	15,000	15,700	16,860	18,910	21,075	23,319
	20,000	10,800	11,225	12,250	13,760	15,975	18,300
	30,000	7,000	7,323	8,154	9,285	11,115	13,484
	40,000	4,000	4,435	5,000	5,700	6,860	8,642
	50,000	2,500	2,600	2,835	3,215	3,950	5,057

The main outputs of the propulsion model are the engine forces and moments expressed in body coordinates. The propulsion model assumes that the thrust force is co-aligned with aircraft's longitudinal axis and that the engine angular momentum (h_e) is fixed at a constant 216.9 kg-m2/s (160 slugs-ft2). From this momentum, the total engine moment can be calculated by performing a cross product vector between the aircraft's angular rate and the engine momentum as shown in equation 3.21.

$$F_{T_B} = [T \quad 0 \quad 0]\ lbs \tag{3.19}$$

$$h_e = [216.9 \quad 0 \quad 0]\ kg \cdot m^2/s \tag{3.20}$$

$$M_{P_B} = \omega_b \times h_e = \Omega_b \cdot h_e \tag{3.21}$$

3.6 6-DOF Equations of Motion Model

The six degrees of freedom equations of motion model represents the heart of the simulation. It is mainly responsible of computing all the important states of the vehicle such as position, velocity, and attitude based on the exerted forces and moments acting on the vehicle body. The fundamentals of these equations were explained in detail in section 2.3. The model presented in this section is mainly based on those equations. The 6-DOF equations of motion have numerous interdependencies among them and they must be executed in a certain order to satisfy such dependencies. Figure 3-20 shows a flow diagram that explains the execution order of this model while Figure 3-21 shows the overall view of the Simulink implementation.

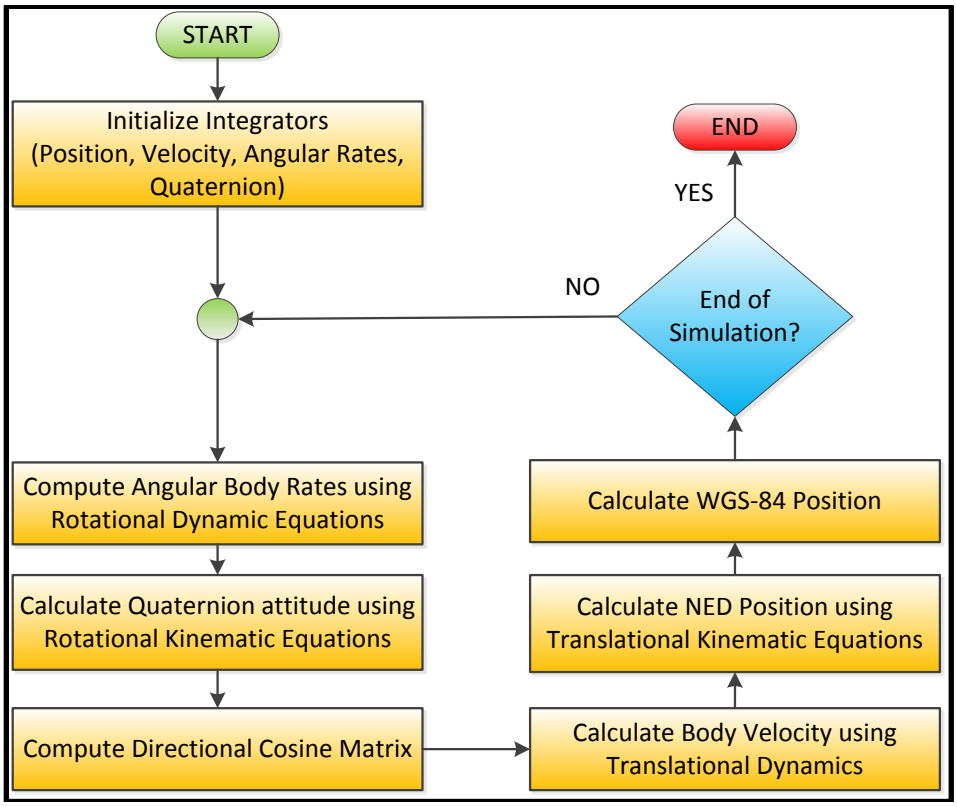

Figure 3-20: 6-DOF Equations of Motion Flow Diagram

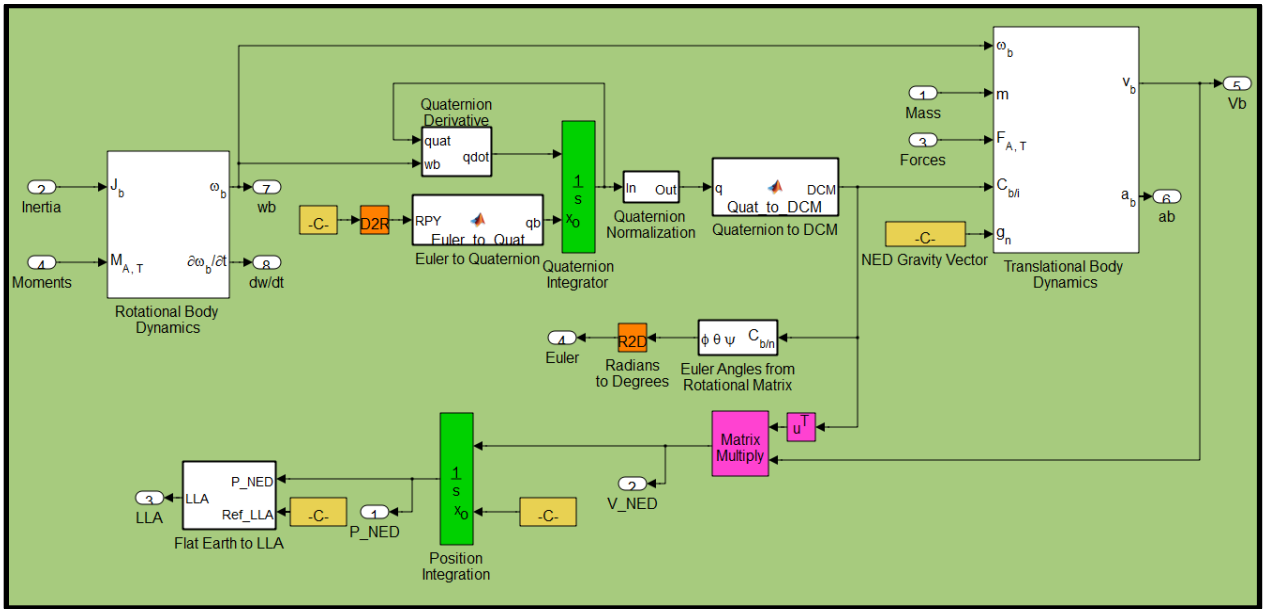

Figure 3-21: 6-DOF Equations of Motion Simulink Model

As it can be noticed in the flow diagram, the 6-DOF equations of motion model consists of one initialization step followed by six recursive steps. During the first step, all the integrators used to calculate the vehicle states (position, velocity, attitude, and angular rates) are properly initialized. One key point that must be emphasized in this is step is that since the initial attitude condition of the aircraft is typically expressed in Euler angles, equation 2.14 is used to convert these angles and calculate the initial quaternion vector.

The second step consists of updating the rotational dynamic equations (see equations 2.8 and 2.9 on section 2.3.1) which are responsible of calculating the angular accelerations and angular rates of the aircraft based on the vehicle's moment of inertia and the exerted moments acting on the body. A Simulink implementation of this step is depicted in Figure 3-22.

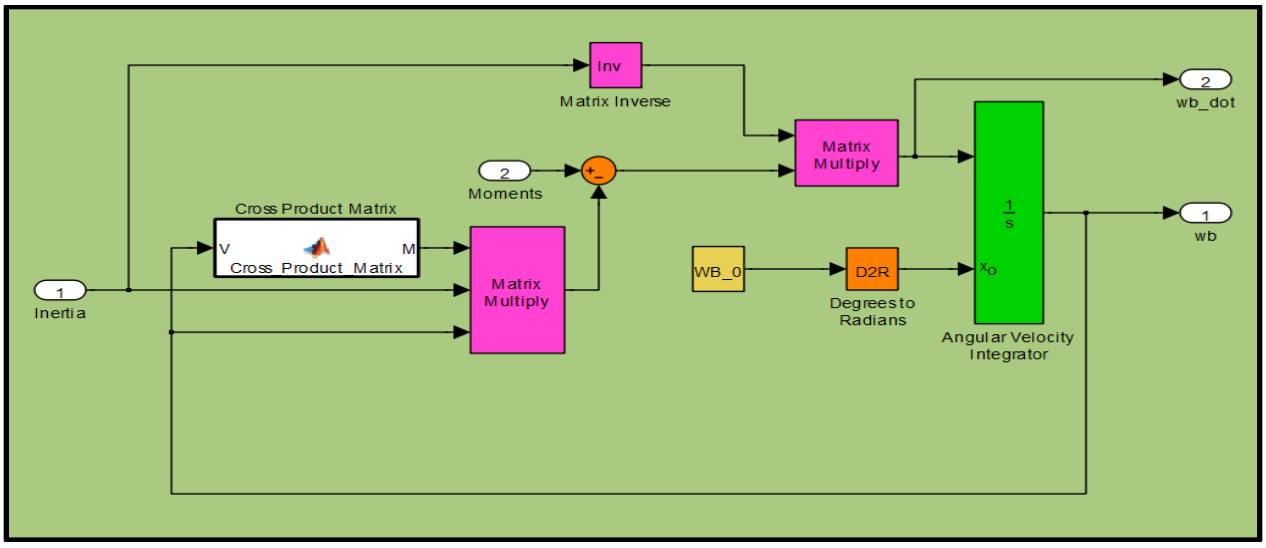

Figure 3-22: 6-DOF Rotational Dynamics Simulink Implementation

During the third step, the rotational kinematic equations are updated. This is accomplished by utilizing equation 2.13, found in section 2.3.3, which calculates the quaternion derivative. Once the quaternion integrator is updated, the quaternion vector is normalized to guarantee that it maintains a unity magnitude during the entire simulation. Also, the NED to Body Rotation matrix and the current Euler angles are calculated during this step using equations 2.15 to 2.18. A Simulink implementation of equation 2.13 is shown in Figure 3-23.

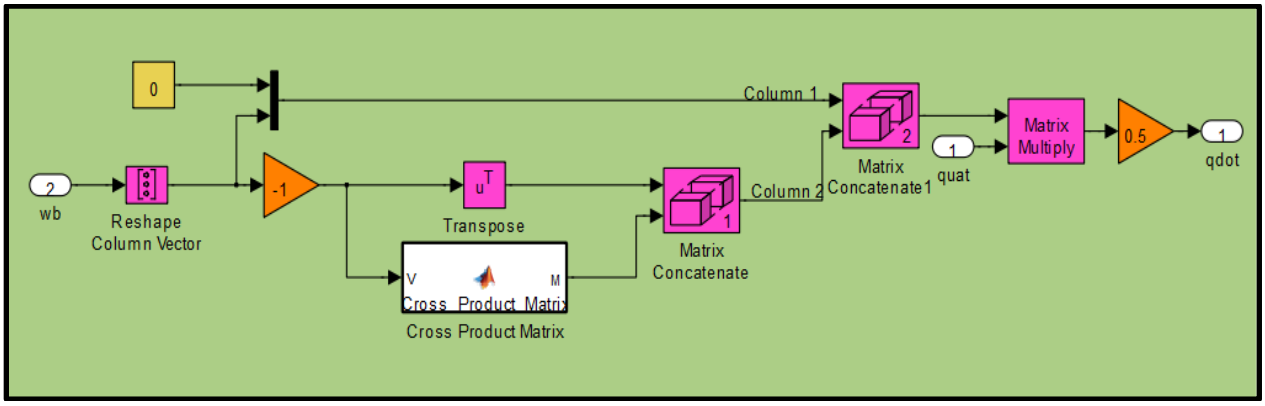

Figure 3-23: 6-DOF Rotational Kinematics Simulink Implementation

At the end of the third step, the entire set of the aircraft's rotational variables has been updated. The remaining steps are responsible of updating all of the aircraft's translational states. In the fourth step, the translational dynamic equations are calculated based on the aircraft's mass and the exerted forces acting of the vehicle. The linear acceleration and velocity in body coordinates are updated using equation 2.11 explained in section 2.3.2. Figure 3-24 shows the Simulink implementation of this equation.

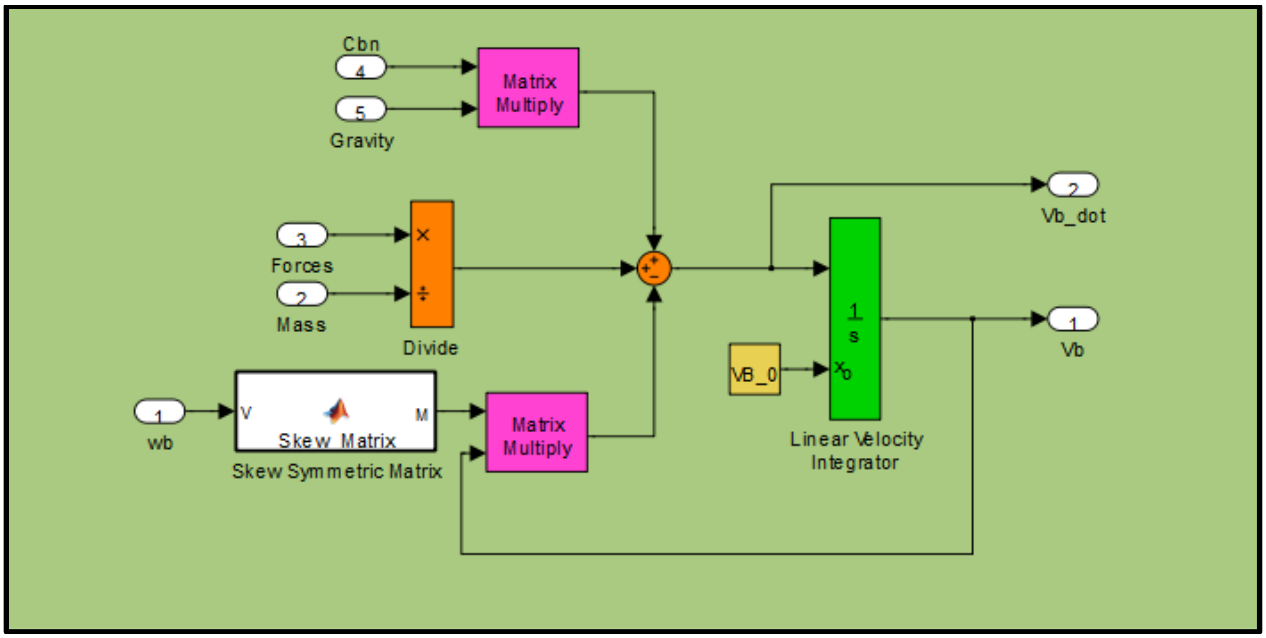

Figure 3-24: 6-DOF Translational Dynamic Equations Simulink Implementation

The fifth step is related to the computation of the local navigation frame states such as the NED position and the NED velocity by utilizing the translational kinematic equations shown in section 2.3.4. After updating the NED position, the WGS-84 coordinates can be calculated making this the sixth and last recursive step of the model.

The WGS-84 coordinates can be estimated from the flat Earth navigational frame by defining a reference latitude, longitude, and altitude, and converting the aircraft's NED position as a deviation from these reference coordinates. To convert the north and east coordinates into latitude and longitude deviations, the Earth radius of curvature in the prime vertical (R_N) and the radius of curvature in the prime meridian (R_M) are used. These two radii can be computed using the equations below.

$$R_N = \frac{R_a}{\sqrt{1-(2f-f^2)sin^2\varphi_0}} \quad \textbf{(3.22)}$$

$$R_M = R_N \frac{1-(2f-f^2)}{1-(2f-f^2)sin^2\varphi_0} \quad \textbf{(3.23)}$$

The terms R_a and f are the Earth equatorial radius and the Earth flattening defined in Table 2-1, while the term φ_0 is the selected reference latitude. The conversion between north and east positions to latitude and longitude deviations are shown below.

$$d\varphi = tan^{-1}\left(\frac{1}{R_M}\right)P_N \quad \textbf{(3.24)}$$

$$d\ell = tan^{-1}\left(\frac{1}{R_N cos\varphi}\right)P_E \quad \textbf{(3.25)}$$

Once the latitude and longitude deviations are determined, the aircraft's geodetic coordinates can be estimated by adding the deviations to the reference coordinates as demonstrated in the equations below.

$$\varphi = \varphi_0 + d\varphi \quad \textbf{(3.26)}$$

$$\ell = \ell_0 + d\ell \quad \textbf{(3.27)}$$

$$h = h_0 - P_D \quad \textbf{(3.28)}$$

CHAPTER 4: **CONTROLLING THE NONLINEAR UAV**

Unmanned aerial vehicles, like all other aircraft, are inherently nonlinear systems that must be properly stabilized and controlled. The most common approach to develop an autopilot is to linearize the aircraft dynamics on a particular operating point and develop a linear controller to stabilize it. The problem with this approach is that the aircraft states must not venture too far from this operating point or the designed autopilot could start becoming unstable. The typical design solution for an aircraft that operates in a wide flight envelope is to select multiple operating points, develop linear controllers for each point, and adjust the controller gains based on the aircraft's current states. This approach, known as gain scheduling, is widely used in aircraft and missile autopilot designs. Though this approach is generally acceptable, it requires extensive amount of work and testing.

Nonlinear control theory provides numerous techniques that can take into account the known nonlinearities of the UAV and apply them directly in the controller design to improve overall system performance. One of the main advantages of developing a nonlinear autopilot is that it is capable of stabilizing the UAV plant through the entire operating range without the need of multiple controller designs. In this chapter, a nonlinear controller is developed using the dynamic inversion technique, also known as feedback linearization, which has been successfully proven in many nonlinear applications and an abundance of information can be found in nonlinear textbooks such as in [4].

4.1 Fundamentals of the Dynamic Inversion Technique

A generic nonlinear system can be described as shown in equations 4.1 and 4.2.

$$\dot{x} = f(x,t) + g(x,t)u \quad \textbf{(4.1)}$$

$$y = h(x,t) \quad \textbf{(4.2)}$$

In the equations above, x is the state vector, u is the control input vector, and y is the output vector. The functions $f(x,t)$, $g(x,t)$, and $h(x,t)$ are matrices that are both state and time dependent. In order to make the system follow a reference trajectory, the tracking error is calculated as the difference between this trajectory and the output.

$$e(t) = r(t) - y(t) \quad \textbf{(4.3)}$$

Based on equation 4.3, the tracking error rate of change can be computed as follows:

$$\dot{e}(t) = \dot{r}(t) - \dot{y}(t) \quad \textbf{(4.4)}$$

The rate of change of the output can be obtained by differentiating equation 4.2.

$$\dot{y}(t) = \frac{\partial h}{\partial x}\frac{\partial x}{\partial t} = \frac{\partial h}{\partial x}\dot{x}(t) = \frac{\partial h}{\partial x}f(x) + \frac{\partial h}{\partial x}g(x)u = F(x) + G(x)u \quad \textbf{(4.5)}$$

Utilizing equation 4.5, the error rate of change can be rewritten as shown below.

$$\dot{e}(t) = \dot{r}(t) - [F(x) + G(x)u] \quad \textbf{(4.6)}$$

A virtual control input can be defined as the negative of the error rate of change.

$$\upsilon = -\dot{e}(t) \quad \textbf{(4.7)}$$

Utilizing this virtual control input, the dynamic inversion control law can be defined by solving for the control input in equation 4.6.

$$u = G^{-1}(x)[-F(x) + \dot{r}(t) + \upsilon] \quad \textbf{(4.8)}$$

By utilizing the dynamic inversion control law shown in equation 4.8, the transfer function between the virtual control υ and the system output y becomes a linear system with the poles located at the origin, making the system marginally stable. Because of this, the virtual control input can be designed using any linear control technique.

4.2 Derivation of Wind Axis Dynamic Equations

The utilization of the rigid body equations of motion expressed in the body coordinate frame, as explained in section 2.3, is convenient for the development of simulations due to its simple nature. However, due to the interdependencies between the aircraft motion and the aerodynamic effects, it is often more convenient to express these equations of motion in the wind coordinate frame. This coordinate frame makes it easier for the development of an autopilot that must control the position and attitude of an aircraft in the local tangent plane.

The translational dynamics of an aircraft expressed in the body coordinate frame define the aircraft's velocity vector as three independent velocity terms (U, V, and W). In the wind frame, on the other hand, the velocity vector is expressed as the aircraft's airspeed and the two aerodynamics angles (V_T, α, β). These terms can be derived from the body velocity terms using the following equations:

$$V_T = \sqrt{U^2 + V^2 + W^2} \quad \textbf{(4.9)}$$

$$\alpha = tan^{-1}\frac{W}{U} \quad \textbf{(4.10)}$$

$$\beta = sin^{-1}\frac{V}{V_T} \quad \textbf{(4.11)}$$

Vice versa, the body velocities can be determined from the airspeed and the aerodynamic angles using the following set of equations:

$$U = V_T \cos\alpha \cos\beta \quad \textbf{(4.12)}$$

$$V = V_T \sin\beta \quad \textbf{(4.13)}$$

$$W = V_T \sin\alpha \cos\beta \quad \textbf{(4.14)}$$

The translational dynamic equations expressed in the wind coordinate system can be calculated by deriving equations 4.9, 4.10, and 4.11 as shown below.

$$\dot{V}_T = \frac{U \cdot \dot{U} + V \cdot \dot{V} + W \cdot \dot{W}}{V_T} \quad \textbf{(4.15)}$$

$$\dot{\alpha} = \frac{U \cdot \dot{W} - \dot{U} \cdot W}{U^2 + W^2} \quad \textbf{(4.16)}$$

$$\dot{\beta} = \frac{\dot{V} \cdot V_T - V \cdot \dot{V}_T}{V_T \sqrt{U^2 + W^2}} \quad \textbf{(4.17)}$$

Substituting the terms shown in equations 4.15 to 4.17 with the ones shown in equation 2.11 and equations 4.12 to 4.14, we obtain the final derivatives of the airspeed, the angle of attack, and the sideslip angle as shown below.

$$\dot{V}_T = \frac{1}{m}[F_X \cos\alpha \cos\beta + F_Y \sin\beta + F_Z \sin\alpha \cos\beta] - g \sin\gamma \quad \textbf{(4.18)}$$

$$\dot{\alpha} = Q - \tan\beta\,[P \cos\alpha + R \sin\alpha] + \frac{1}{mV_T \cos\beta}[F_Z \cos\alpha - F_X \sin\alpha] + \frac{g \cos\gamma \cos\mu}{V_T \cos\beta} \quad \textbf{(4.19)}$$

$$\dot{\beta} = (P\sin\alpha - R\cos\alpha)` + \frac{1}{mV_T}[-F_X\cos\alpha\sin\beta + F_Y\cos\beta - F_Z\sin\alpha\sin\beta] + \frac{g\cos\gamma\sin\mu}{V_T} \quad \textbf{(4.20)}$$

The wind angles are composed of the bank angle (μ), the flight path angle (γ), and the heading angle (χ). The flight path and the heading angles provide a relationship between the aircraft's velocity vector and the local navigation frame. The manipulation of these angles is imperative towards effectively controlling the position of the aircraft. The derivatives of these angles can be determined utilizing the following equations:

$$\begin{bmatrix}\dot{\mu}\\ \dot{\gamma}\\ \dot{\chi}\end{bmatrix} = \begin{bmatrix}1 & \sin\mu\tan\gamma & \cos\mu\tan\gamma\\ 0 & \cos\mu & -\sin\mu\\ 0 & \sin\mu/\cos\gamma & \cos\mu/\cos\gamma\end{bmatrix}\begin{bmatrix}P_W\\ Q_W\\ R_W\end{bmatrix} \quad \textbf{(4.21)}$$

The angular velocity vector in wind coordinate frame can be calculated from the angular body velocity vector as follows:

$$\begin{bmatrix}P_W\\ Q_W\\ R_W\end{bmatrix} = \begin{bmatrix}\cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta\\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta\\ -\sin\alpha & 0 & \cos\alpha\end{bmatrix}\begin{bmatrix}P\\ Q\\ R\end{bmatrix} + \begin{bmatrix}-\sin\beta\cdot\dot{\alpha}\\ -\cos\beta\cdot\dot{\alpha}\\ \dot{\beta}\end{bmatrix} \quad \textbf{(4.22)}$$

Substituting equations 4.19, 4.20, and 4.22 into equation 4.21, we can obtain the final equations of the wind angle derivatives shown below.

$$\dot{\mu} = \frac{1}{\cos\beta}[P\cos\alpha + R\sin\alpha] - \frac{\tan\beta}{V_T} g\cos\gamma\cos\mu + \frac{1}{mV_T}[F_X(\sin\alpha\tan\beta + \sin\alpha\sin\mu\tan\gamma - \cos\alpha\sin\beta\cos\mu\tan\gamma) + F_Y(\cos\beta\cos\mu\tan\gamma) - F_Z(\cos\alpha\tan\beta + \cos\alpha\sin\mu\tan\gamma + \sin\alpha\sin\beta\cos\mu\tan\gamma)] \quad \textbf{(4.23)}$$

$$\dot{\gamma} = \frac{1}{mV_T}[F_X(\sin\alpha\cos\mu + \cos\alpha\sin\beta\sin\mu) - F_Y(\cos\beta\sin\mu) - F_Z(\cos\alpha\cos\mu - \sin\alpha\sin\beta\sin\mu)] - \frac{g\cos\gamma}{V_T} \quad \textbf{(4.24)}$$

$$\dot{\chi} = \frac{1}{mV_T\cos\gamma}[F_X(\sin\alpha\sin\mu - \cos\alpha\sin\beta\cos\mu) + F_Y(\cos\beta\cos\mu) - F_Z(\cos\alpha\sin\mu + \sin\alpha\sin\beta\cos\mu)] \quad \textbf{(4.25)}$$

4.3 Nonlinear Autopilot Overview

The UAV autopilot is mainly responsible of stabilizing the angular rates and controlling the attitude and the velocity of the aircraft. It accomplishes this by manipulating the four main aircraft control inputs which are the throttle, the ailerons, the elevator, and the rudder. The presented autopilot design is mainly based in the nonlinear dynamic inversion technique explained in section 4.1.

The aircraft dynamics are composed of a set of 12 equations that are controlled by only four inputs. In addition, some of these equations exhibit non-minimum phase behavior that prevents direct inversion of these equations since this behavior will create instability in the controller. To overcome these issues, certain approaches and assumptions must be taken to take advantage of the feedback linearization technique. One of the

approaches taken in [5], [6], and [7] is to divide the aircraft dynamics into a set of fast dynamics consisting of the angular rates and a set of slow dynamics that contain the aircraft attitude. Also, the solutions presented in these references assume that the control surfaces only generate angular moments and that the forces created by them (which are one of the main causes of non-minimum phase behavior) are negligible. The autopilot presented in this chapter is based in this two time scale separation concept and follows the assumptions presented in these references. The autopilot will be divided into three main loops (inner, middle, and outer) as shown in Figure 4-1.

The inner loop consists of an angular velocity controller that is mainly responsible of stabilizing the angular rates of the aircraft by computing the required amount of torque (τ_{CMD}) based on the desired angular rates (ω_{CMD}). This angular velocity controller, shaded in blue, is composed of a dynamic inversion algorithm combined with a linear PID controller that calculates the virtual control law required by the dynamic inversion controller. The inner loop also contains a control allocation algorithm, shown in red, which calculates the required angular deflections of the control surfaces from the commanded torque. This control allocation algorithm also receives a direct thrust command (T_{CMD}) from the velocity controller to determine the required throttle position.

The middle loop controller, shaded in yellow, controls the attitude of the aircraft and calculates the desired angular velocity vector (ω_{CMD}) which is the input of the inner loop controller. The aircraft's attitude is expressed in terms of the bank angle (μ) and the

aerodynamic angles which are the angle of attack and the sideslip angle (α, β). A derivation of the aerodynamic angle dynamics was explained in the previous section.

The outer loop controller is responsible of manipulating the velocity vector of the aircraft by tracking the guidance commands which are the commanded airspeed (V_{CMD}), the commanded heading (χ_{CMD}), and the flight path angle (γ_{CMD}). This outer loop controller calculates the desired attitude commands $([\mu, \alpha, \beta]_{CMD})$ which are the inputs to the middle loop controller. It also calculates the thrust command (T_{CMD}) which is sent directly to the control allocation algorithm. This outer loop controller is composed of a linear PI controller and a maneuver generation algorithm that calculates the desired attitude commands tracked by the middle loop.

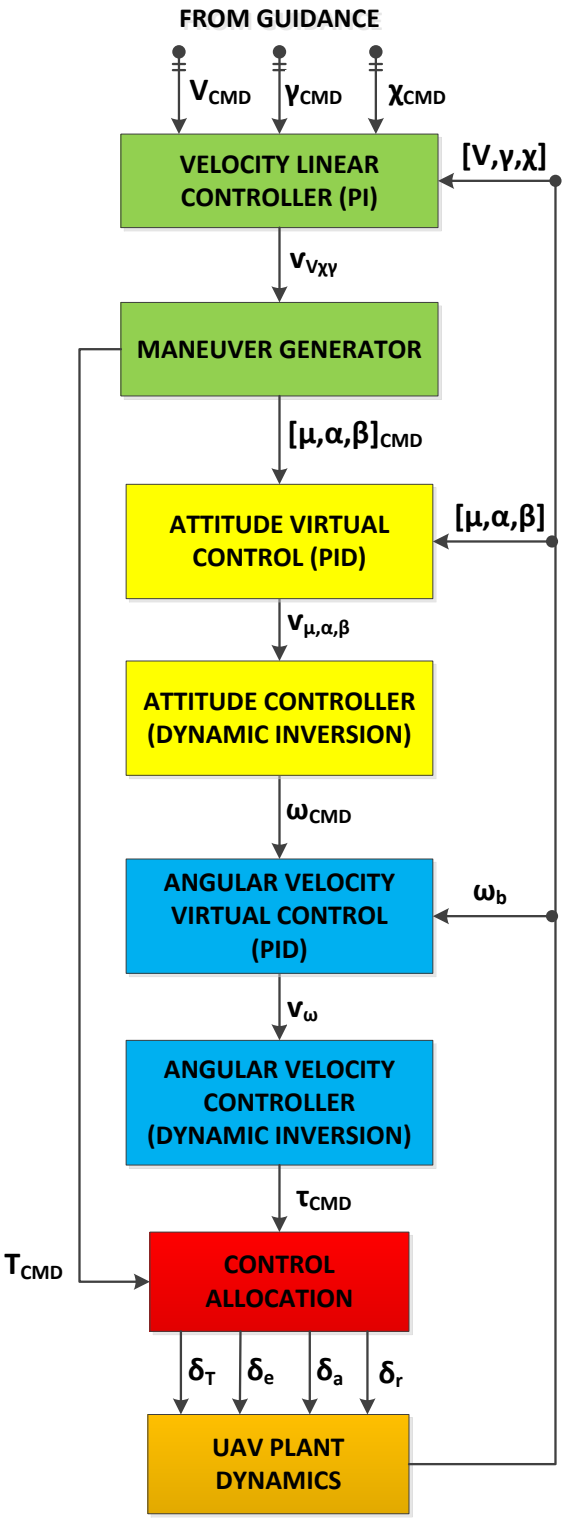

Figure 4-1: Nonlinear Autopilot Design Layout

4.4 Nonlinear Autopilot Design

The nonlinear autopilot is composed of three main control loops. The inner loop is responsible of stabilizing the angular rates of the aircraft by commanding the required torque to the control surfaces. The middle loop controls the attitude of the aircraft by calculating the angular velocity command needed by the inner loop. Finally, the outer loop controls the velocity vector of the aircraft by following the guidance commands that are composed of the airspeed, the heading, and the flight path angle. The following sub-sections provide a detailed explanation of the design of each controller.

4.4.1 Inner Loop: Angular Velocity Controller

The inner loop controller must be able to handle the nonlinearities found in the rotational dynamic equations explained in section 2.3.1. For this reason, the dynamic inversion technique previously explained has been selected as the base design for this loop. In this particular case, the state vector and the output vector can both be defined as the body angular velocity vector. The control inputs are defined as the exerted angular moments.

$$x = \omega_b = \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad , \quad y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad , \quad u = \tau_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_{CMD} \tag{4.26}$$

Using equation 2.8, we can define the derivative of the state vector as follows:

$$\dot{x} = \dot{\omega}_b = (J_b)^{-1}[\tau_b - \Omega_b J_b \omega_b] \quad \textbf{(4.27)}$$

As a reminder, the variable Ω_b is the cross product matrix of the angular velocity.

$$\Omega_b = \begin{bmatrix} 0 & -R & Q \\ R & 0 & -P \\ -Q & P & 0 \end{bmatrix} \quad \textbf{(4.28)}$$

Rearranging the terms in equation 4.27, we can represent the rotational dynamic equations in a format similar to the generic nonlinear system (equations 4.1 and 4.2).

$$\dot{x} = f(x,t) + g(x,t)u = -(J_b)^{-1}\Omega_b J_b \, x + (J_b)^{-1}u \quad \textbf{(4.29)}$$

$$y = h(x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x \quad \textbf{(4.30)}$$

Following equation 4.5 and utilizing the equations above, the functions $F(x)$ and $G(x)$ can be determined by differentiating $h(x)$ with respect to the state vector.

$$\frac{\partial h}{\partial x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(4.31)}$$

$$F(x) = \frac{\partial h}{\partial x} f(x) = -(J_b)^{-1}\Omega_b J_b \quad \textbf{(4.32)}$$

$$G(x) = \frac{\partial h}{\partial x} g(x) = (J_b)^{-1} \quad \textbf{(4.33)}$$

Substituting the determined $F(x)$ and $G(x)$ functions into equation 4.8, the dynamic inversion control law for the inner loop can be defined as follows.

$$\tau_{CMD} = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_{CMD} = J_b[(J_b)^{-1}\Omega_b J_b \omega_b + \upsilon_\omega] \quad \textbf{(4.34)}$$

The virtual control υ_ω has been selected as a combination of a proportional-integral-derivative (PID) controller and a feed-forward derivative of the desired angular rate command to satisfy the $\dot{r}(t)$ term shown in equation 4.8. Equation 4.35 shows the selected virtual control law.

$$\upsilon_\omega = \left[K_{FF}\dot{\omega}_{CMD} - \left(K_P + \frac{K_I}{s} + K_D s\right)(\omega_{CMD} - \omega_b)\right] \quad \textbf{(4.35)}$$

The dynamic inversion control law presented in equation 4.34 calculates the angular moment vector required to stabilize the rotational dynamics of the UAV; however, this torque command must be converted to the control inputs of the aircraft. A control allocation algorithm can be implemented to accomplish this conversion. This approach allows the utilization of this dynamic inversion control law in any rigid body vehicle by simply modifying the control allocation algorithm to the particular control inputs of the vehicle. The selected control allocation algorithm employs equations 2.23 to 2.25 and

the aerodynamic coefficients of the aircraft to calculate the deflections required in the control surfaces.

$$\tau_{CMD} = \begin{bmatrix} L \\ M \\ N \end{bmatrix}_{CMD} = \begin{bmatrix} \bar{q} \cdot S \cdot b \cdot C_l(\alpha, \beta, P, R, \delta_a, \delta_r) \\ \bar{q} \cdot S \cdot \bar{c} \cdot C_m(\alpha, \beta, Q, \delta_e) \\ \bar{q} \cdot S \cdot b \cdot C_n(\alpha, \beta, P, R, \delta_a, \delta_r) \end{bmatrix} \quad \textbf{(4.36)}$$

Separating the aerodynamic control derivatives from the other aerodynamic coefficients and rearranging equation 4.36, the control allocation equations can be determined.

$$\delta_e = \frac{1}{C_{m_{\delta_e}}} \left(\frac{M_{CMD}}{\bar{q} \cdot S \cdot \bar{c}} - C_m(\alpha, \beta, Q) \right) \quad \textbf{(4.37)}$$

$$\begin{bmatrix} \delta_a \\ \delta_r \end{bmatrix} = \begin{bmatrix} C_{l_{\delta_a}} & C_{l_{\delta_r}} \\ C_{n_{\delta_a}} & C_{n_{\delta_r}} \end{bmatrix}^{-1} \left(\frac{1}{\bar{q} \cdot S \cdot b} \begin{bmatrix} L \\ N \end{bmatrix}_{CMD} - \begin{bmatrix} C_l(\alpha, \beta, P, R) \\ C_n(\alpha, \beta, P, R) \end{bmatrix} \right) \quad \textbf{(4.38)}$$

4.4.2 Middle Loop: Attitude Controller

The middle loop controller's main responsibility is to stabilize the angle of attack (α) and the sideslip angle (β) of the aircraft and to control the bank (roll) angle (μ). The design of this controller is also based on the dynamic inversion technique explained in section 4.1 where the state vector and the output vector are defined as the aerodynamic angles and the bank angle. The control input is defined as the body angular velocity vector command. Equation 4.39 illustrates these variable definitions.

$$x = \begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix} \quad , \quad y = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix} \quad , \quad u = \omega_{CMD} = \begin{bmatrix} P \\ Q \\ R \end{bmatrix}_{CMD} \tag{4.39}$$

The state vector derivatives for the middle loop controller were derived in equations 4.19, 4.20, and 4.23. Following the standard format presented in equation 4.1, the state vector derivatives can expressed as follows:

$$\dot{x} = f(x,t) + g(x,t)u = \begin{bmatrix} \dot{\mu} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} f_\mu(x) \\ f_\alpha(x) \\ f_\beta(x) \end{bmatrix} + \begin{bmatrix} \cos\alpha /_{\cos\beta} & 0 & \sin\alpha /_{\cos\beta} \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix} u \tag{4.40}$$

The functions $f_\mu(x)$, $f_\alpha(x)$, and $f_\beta(x)$ shown in equation 4.40 are defined as:

$$\begin{aligned} f_\mu(x) = \frac{1}{mV_T}[&F_X(\sin\alpha\tan\beta + \sin\alpha\sin\mu\tan\gamma - \cos\alpha\sin\beta\cos\mu\tan\gamma) \\ &+ F_Y(\cos\beta\cos\mu\tan\gamma) \\ &- F_Z(\cos\alpha\tan\beta + \cos\alpha\sin\mu\tan\gamma + \sin\alpha\sin\beta\cos\mu\tan\gamma)] \\ &- \frac{\tan\beta}{V_T} g\cos\gamma\cos\mu \end{aligned} \tag{4.41}$$

$$f_\alpha(x) = \frac{1}{mV_T\cos\beta}[F_Z\cos\alpha - F_X\sin\alpha] + \frac{g\cos\gamma\cos\mu}{V_T\cos\beta} \tag{4.42}$$

$$f_\beta(x) = \frac{1}{mV_T}[-F_X\cos\alpha\sin\beta + F_Y\cos\beta - F_Z\sin\alpha\sin\beta] + \frac{g\cos\gamma\sin\mu}{V_T} \tag{4.43}$$

Differentiating the output equation shown in 4.39 with respect to the state vector we obtain:

$$\frac{\partial h}{\partial x} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(4.44)}$$

$$F(x) = \frac{\partial h}{\partial x} f(x) = \begin{bmatrix} f_\mu(x) \\ f_\alpha(x) \\ f_\beta(x) \end{bmatrix} \quad \textbf{(4.45)}$$

$$G(x) = \frac{\partial h}{\partial x} g(x) = \begin{bmatrix} \cos\alpha/\cos\beta & 0 & \sin\alpha/\cos\beta \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix} \quad \textbf{(4.46)}$$

By substituting equations 4.45 and 4.46 into equation 4.8, we can obtain the dynamic inversion control law for the middle loop controller as shown below.

$$\omega_{CMD} = \begin{bmatrix} P \\ Q \\ R \end{bmatrix}_{CMD} = \begin{bmatrix} \cos\alpha/\cos\beta & 0 & \sin\alpha/\cos\beta \\ -\cos\alpha\tan\beta & 1 & -\sin\alpha\tan\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix}^{-1} \left[-\begin{bmatrix} f_\mu(x) \\ f_\alpha(x) \\ f_\beta(x) \end{bmatrix} + \upsilon_{\mu\alpha\beta} \right] \quad \textbf{(4.47)}$$

The virtual controller $\upsilon_{\mu\alpha\beta}$ is defined as PID controller with a feed-forward signal similar to the one utilized in the inner loop controller. Equation 4.48 shows the virtual control law for the middle loop.

$$\upsilon_{\mu\alpha\beta} = \left[K_{FF} \begin{bmatrix} \dot{\mu} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix}_{CMD} - \left(K_P + \frac{K_I}{s} + K_D s \right) \left(\begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix}_{CMD} - \begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix} \right) \right] \quad \textbf{(4.48)}$$

4.4.3 Outer Loop: Velocity Controller

The outer loop of the nonlinear autopilot manipulates the velocity vector of the aircraft by tracking the airspeed, flight path, and heading commands received from the guidance algorithms. This controller is responsible of calculating the bank, angle of attack, and sideslip commands that are required by the attitude controller. It is also responsible of calculating the thrust command that is utilized by the control allocation algorithm to compute the required throttle command. The state vector and input and output definitions for the outer loop are shown in equation 4.49.

$$x = \begin{bmatrix} V_T \\ \gamma \\ \chi \end{bmatrix} \quad , \quad y = \begin{bmatrix} V_T \\ \gamma \\ \chi \end{bmatrix} \quad , \quad u = \begin{bmatrix} T \\ \mu \\ \alpha \\ \beta \end{bmatrix}_{CMD} \tag{4.49}$$

Equations 4.18, 4.24, and 4.25 define the derivatives of the state vector. One of the design constraints of this outer loop controller is the utilization of the coordinated turn maneuver. This type of maneuver is typically desired for fixed wing aircraft motion since it is the most energy efficient approach to execute turns. During a coordinated turn maneuver, the aircraft is banked at a specific angle which is dependent on the heading rate while maintaining a zero sideslip angle. Based on this constraint, the sideslip angle command must be set to zero.

$$\beta_{CMD} = 0 \tag{4.50}$$

The airspeed controller is defined as a simple PI control law with a feed forward gain as shown in equation 4.51.

$$\dot{V}_{DES} = \left[K_{FF}\dot{V}_{CMD} - \left(K_P + \frac{K_I}{s}\right)(V_{CMD} - V_T)\right] \quad \textbf{(4.51)}$$

The desired acceleration must be converted to a thrust command which is one of the responsibilities of the maneuver generator algorithm. Assuming that the engine thrust is co-aligned with the longitudinal axis of the aircraft, the total longitudinal force (F_X) can be defined as the sum of the longitudinal aerodynamic force (F_{XA}) and the total engine thrust (T). Since the sideslip angle is kept to a minimum, equation 4.18 can be simplified and utilized to convert the desired acceleration command presented in equation 4.51 into a thrust command as follows:

$$T_{CMD} = \frac{m\left(\dot{V}_{DES} + g\sin\gamma\right) - [F_{XA}\cos\alpha + F_Z\sin\alpha]}{\cos\alpha} \quad \textbf{(4.52)}$$

The thrust command is sent directly to the control allocation algorithm which converts this command into the necessary engine throttle input. A model of the aircraft engine is typically used to perform this throttle conversion. In other cases, a simple algebraic equation can do the trick as well. The selected throttle control allocation algorithm is presented in equation 4.53.

$$\delta_T = \frac{T_{CMD} - T_{MIN}}{T_{MAX} - T_{MIN}} \quad \textbf{(4.53)}$$

The variables T_{MIN} and T_{MAX} represent the minimum engine thrust (when the engine is idle) and the maximum engine thrust. These variables can be declared as constant values or as values that are dependent on key factors such as altitude and airspeed.

The flight path controller must be capable of tracking the flight path commands generated by the guidance algorithm. To accomplish this, a PI control law, shown in equation 4.54, is utilized.

$$\dot{\gamma}_{DES} = \left[K_{FF}\dot{\gamma}_{CMD} - \left(K_P + \frac{K_I}{s}\right)(\gamma_{CMD} - \gamma)\right] \quad \textbf{(4.54)}$$

Once the desired flight path rate is calculated, the maneuver generator will compute the required angle of attack command needed by the middle loop controller. Once again, based on the assumption that sideslip angle β is close to zero and that the side forces are minimal, the flight path derivative shown equation 4.24 can be simplified as follows:

$$\dot{\gamma} = \frac{1}{mV_T}[F_X(\sin\alpha\cos\mu) - F_Z(\cos\alpha\cos\mu)] - \frac{g\cos\gamma}{V_T} \quad \textbf{(4.55)}$$

As seen above, the flight path derivative is dependent on the aerodynamic lift force (F_Z) which itself is dependent on the angle of attack. Utilizing equation 2.20 and defining the

lift coefficient as a simple sum of a baseline lift coefficient (C_{Z0}) and an alpha dependent lift coefficient ($C_{Z\alpha}$) we can rewrite equation 4.55 as shown below:

$$\dot{\gamma} = \frac{1}{mV_T}[F_X(\sin\alpha\cos\mu) - \bar{q}S(C_{Z0} + C_{Z\alpha}\cdot\alpha)(\cos\alpha\cos\mu)] - \frac{g\cos\gamma}{V_T} \quad \textbf{(4.56)}$$

Solving equation 4.56 for the linear term of the angle of attack, a relationship between the desired flight path rate and the commanded angle of attack can be determined.

$$\alpha_{CMD} = \frac{\left(\frac{F_X(\sin\alpha\cos\mu) - mV_T\dot{\gamma}_{DES} - mg\cos\gamma}{\bar{q}S(\cos\alpha\cos\mu)}\right) - C_{Z0}}{C_{Z\alpha}} \quad \textbf{(4.57)}$$

The heading controller tracks the heading commands received from the guidance algorithm by utilizing a PI control law similar to the one used in the flight path controller.

$$\dot{\chi}_{DES} = \left[K_{FF}\dot{\chi}_{CMD} - \left(K_P + \frac{K_I}{s}\right)(\chi_{CMD} - \chi)\right] \quad \textbf{(4.58)}$$

The maneuver generator algorithm must convert this desired heading rate into a bank angle command required by the middle loop controller. Following the same assumptions used in the flight path controller ($\beta = 0\,, F_Y = 0$), the state equation of the heading angle shown in equation 4.25 can be simplified as shown below.

$$\dot{\chi} = \frac{1}{mV_T \cos\gamma}\left[F_X \sin\alpha \sin\mu - F_Z \cos\alpha \sin\mu\right] \quad \textbf{(4.59)}$$

Solving equation 4.59 for the bank angle, we can obtain the bank angle command from the desired heading rate as shown in equation 4.60.

$$\mu_{CMD} = \sin^{-1}\left(\frac{mV_T \cos\gamma\, \dot{\chi}_{DES}}{F_X \sin\alpha - F_Z \cos\alpha}\right) \quad \textbf{(4.60)}$$

4.5 Autopilot Simulation Implementation and Results

A simulation of the nonlinear autopilot presented in section 4.4 was developed to demonstrate its performance. This section provides an overview of the simulation implementation and the results obtained from it.

4.5.1 Autopilot Simulink Implementation

The autopilot design was implemented in the Simulink platform and interfaced with the F-16 model developed in chapter 3 as depicted in Figure 4-2.

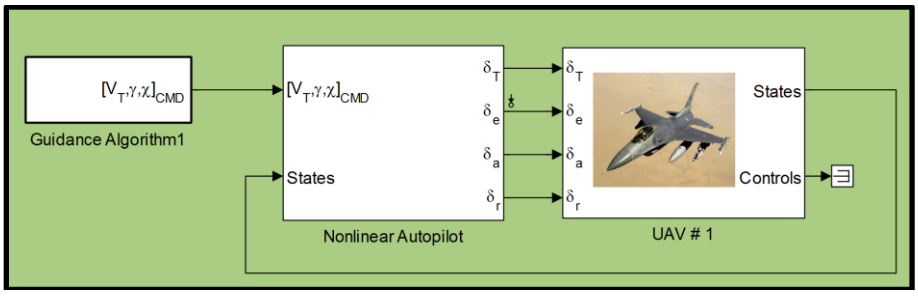

Figure 4-2: Nonlinear Autopilot Interface with F-16 Simulink Model

The main inputs of the autopilot are a vector containing the three guidance commands (airspeed, flight path angle, and heading) and the aircraft states. The autopilot utilizes these inputs and the equations presented in section 4.4 to calculate the four UAV control inputs which are the throttle, the elevator, the ailerons, and the rudder.

A detailed view of the Simulink autopilot subsystem is presented in Figure 4-3. This figure shows the cascaded loop configuration of the autopilot presented in Figure 4-1. The first six blocks on the top of Figure 4-3 (starting from the top left) represent the control laws for the three autopilot loops. The seventh block on the top handles the control allocation algorithms for the throttle and the control surfaces. The first block at the bottom of the figure (starting from the left) calculates the wind state variables from the aircraft states which are expressed in body coordinates. The other two blocks provide the aerodynamic coefficients required by the autopilot given the current aircraft states. For the purposes of this thesis, it is assumed that these coefficients are known. For cases when these coefficients are not known in advance, an online estimation algorithm such as the one presented in [8] can be utilized.

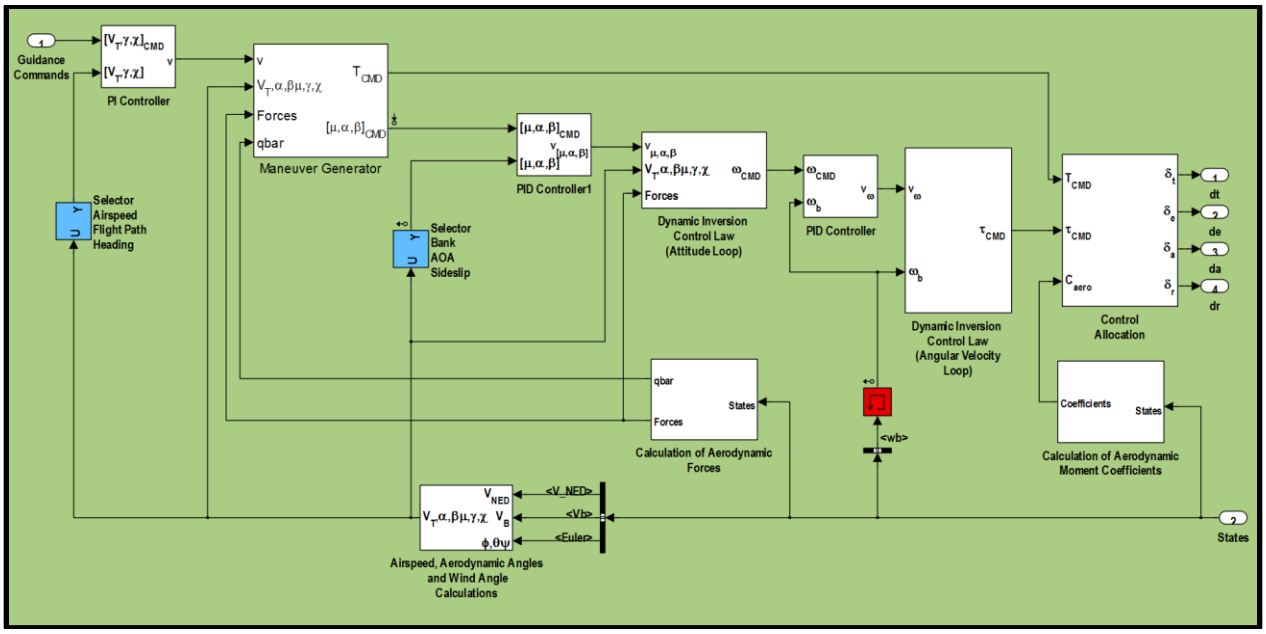

Figure 4-3: Nonlinear Autopilot Simulink Implementation

4.5.2 Angular Velocity Controller Results

The selected gains for the PID controller that is part of the angular velocity (inner) loop are presented in Table 4-1. A step response test was performed on all channels to verify the stability of the controller. The results for the independent channels are shown in Figure 4-4 to Figure 4-6. The results for all three channels excited simultaneously are shown in Figure 4-7. The results show successful tracking of the step commands.

Table 4-1: Angular Velocity PID Controller Gains

Angular Velocity Controller PID Gains	Values
Proportional Gain (K_P)	10.0
Integral Gain (K_I)	0.25
Derivative Gain (K_D)	1.0
Feed Forward Gain	1.0

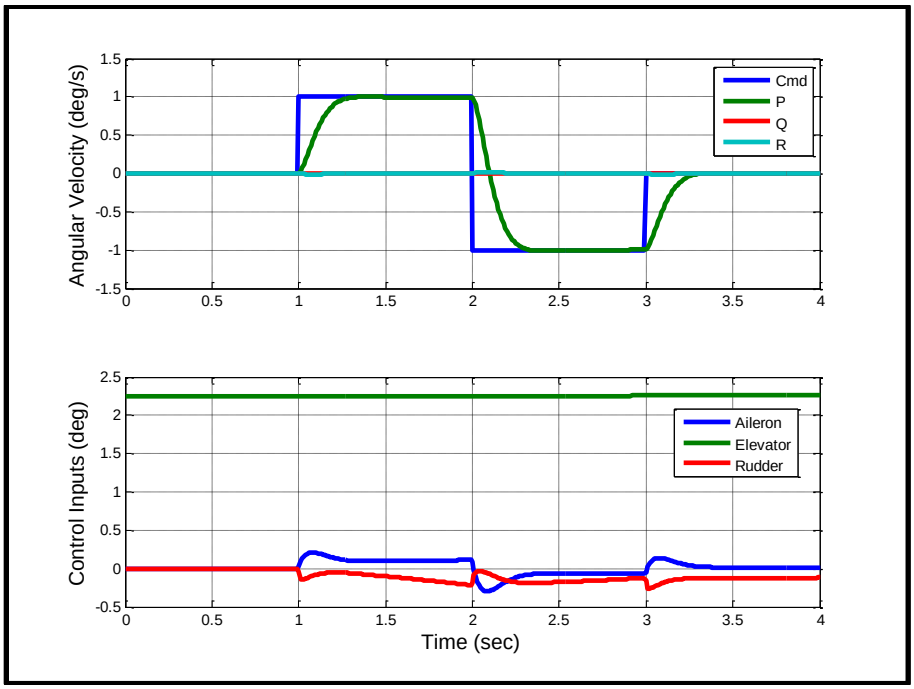

Figure 4-4: Roll Angular Velocity Step Response

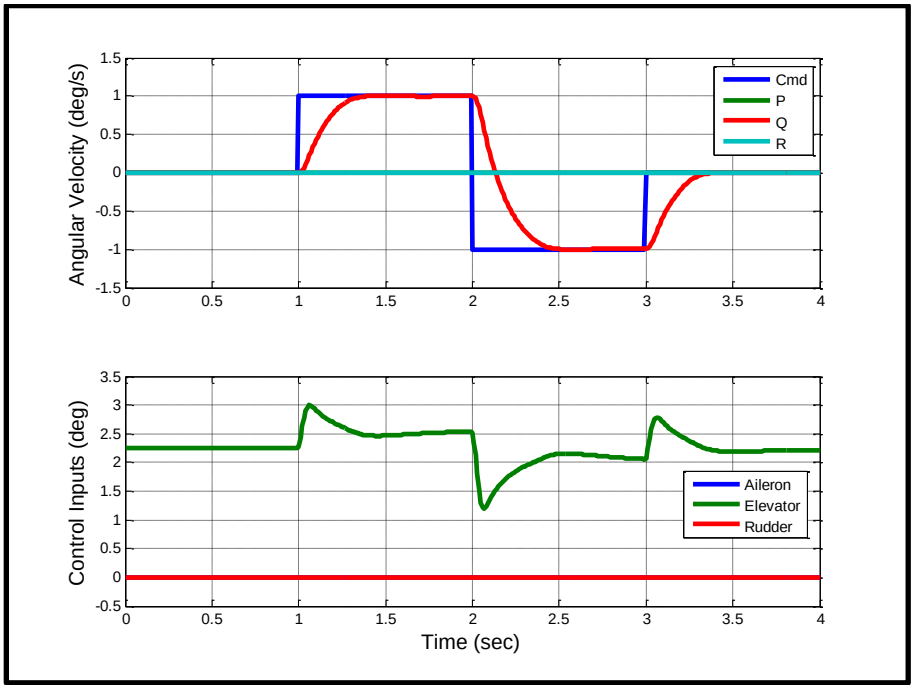

Figure 4-5: Pitch Angular Velocity Step Response

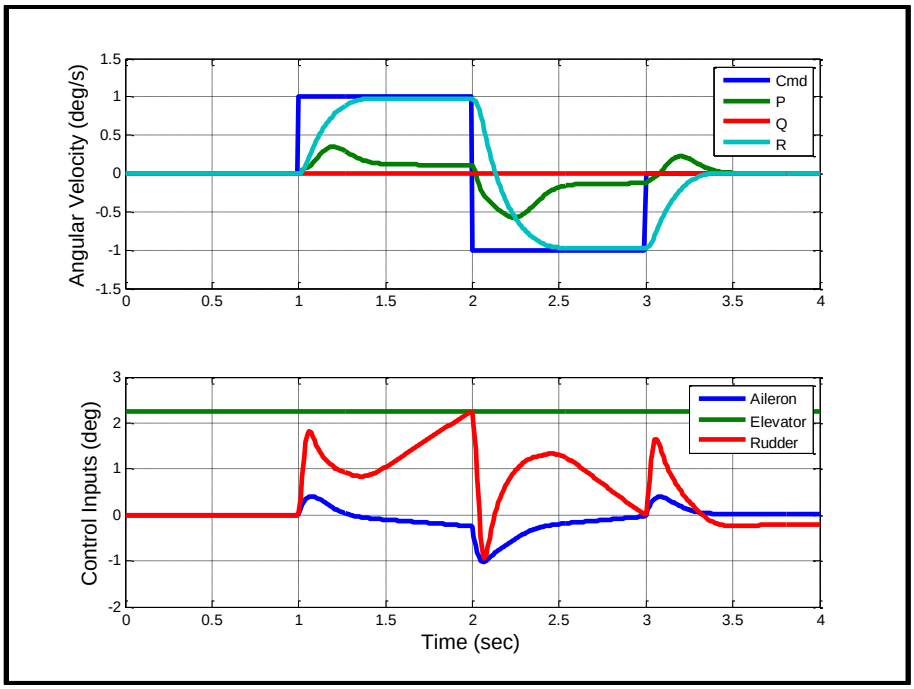

Figure 4-6: Yaw Angular Velocity Step Response

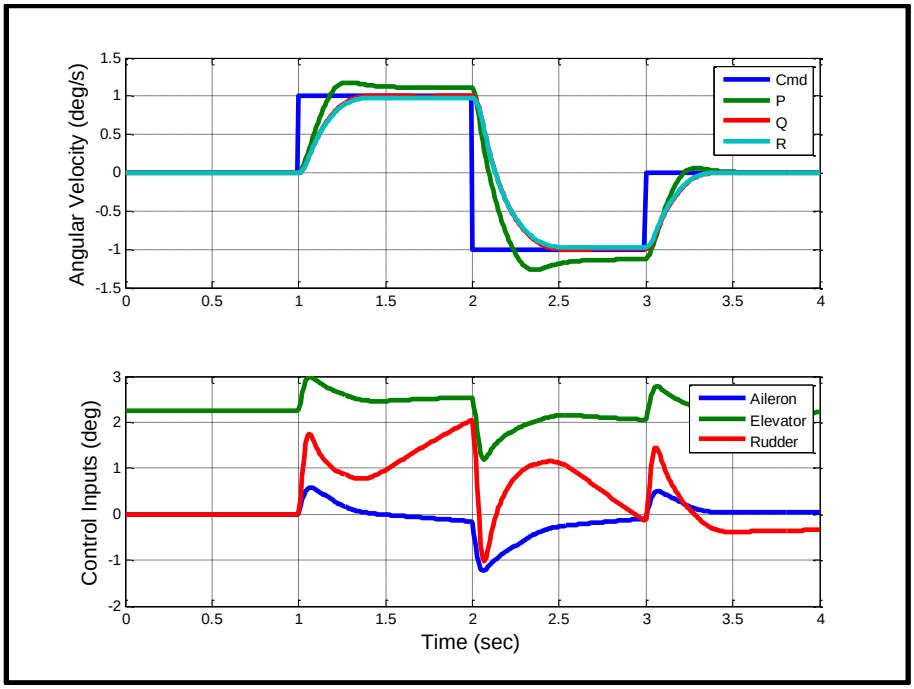

Figure 4-7: Coupled Angular Velocity Step Response

4.5.3 Attitude Controller Results

The selected gains for the PID controller that is part of the attitude (middle) loop are presented in Table 4-2. The integral and derivative channels were disabled since the controller demonstrated good performance with a simple proportional gain. The results for the step response tests for the independent channels are presented in Figure 4-8, Figure 4-9, and Figure 4-10 while the coupled test results are shown in Figure 4-11.

Table 4-2: Attitude Controller PID Gains

Attitude Controller PID Gains	Values
Proportional Gain (K_P)	**2.0**
Integral Gain (K_I)	**0.0**
Derivative Gain (K_D)	**0.0**
Feed forward Gain	**1.0**

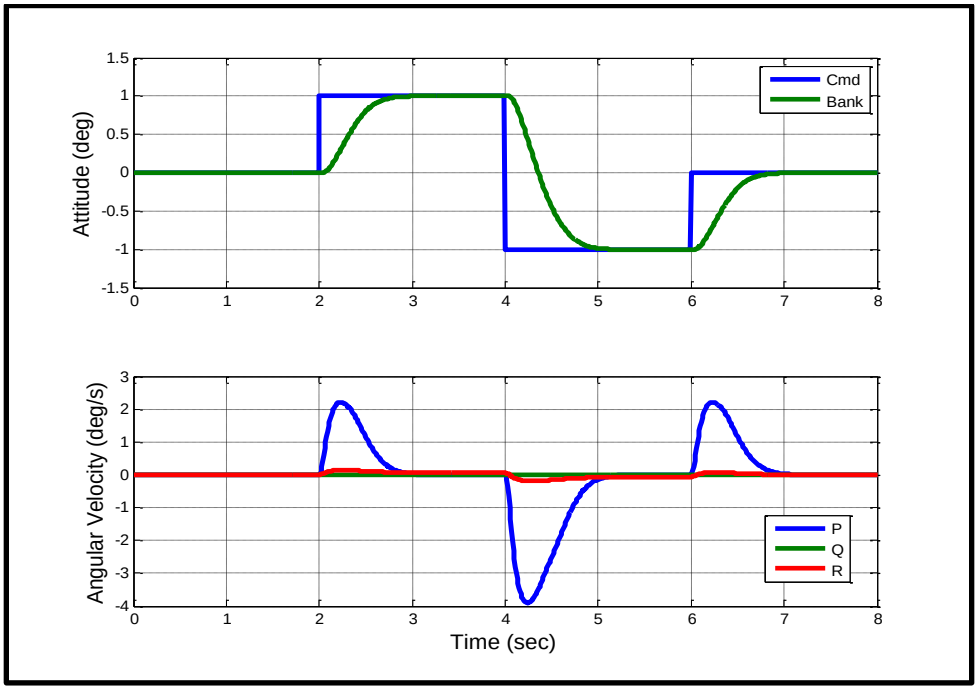

Figure 4-8: Bank Angle Step Response

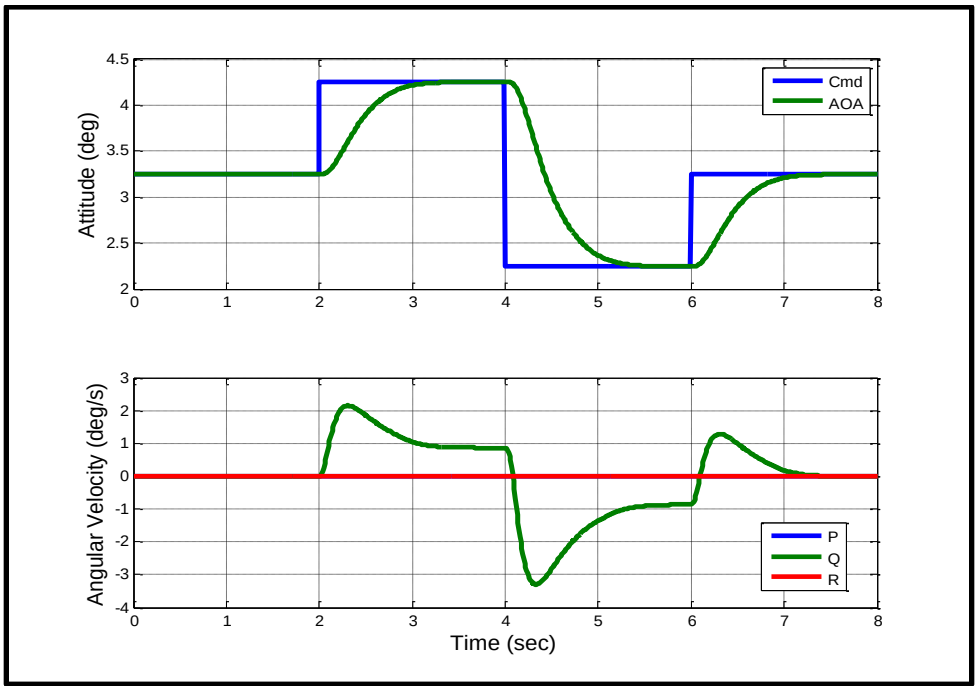

Figure 4-9: Angle of Attack Step Response

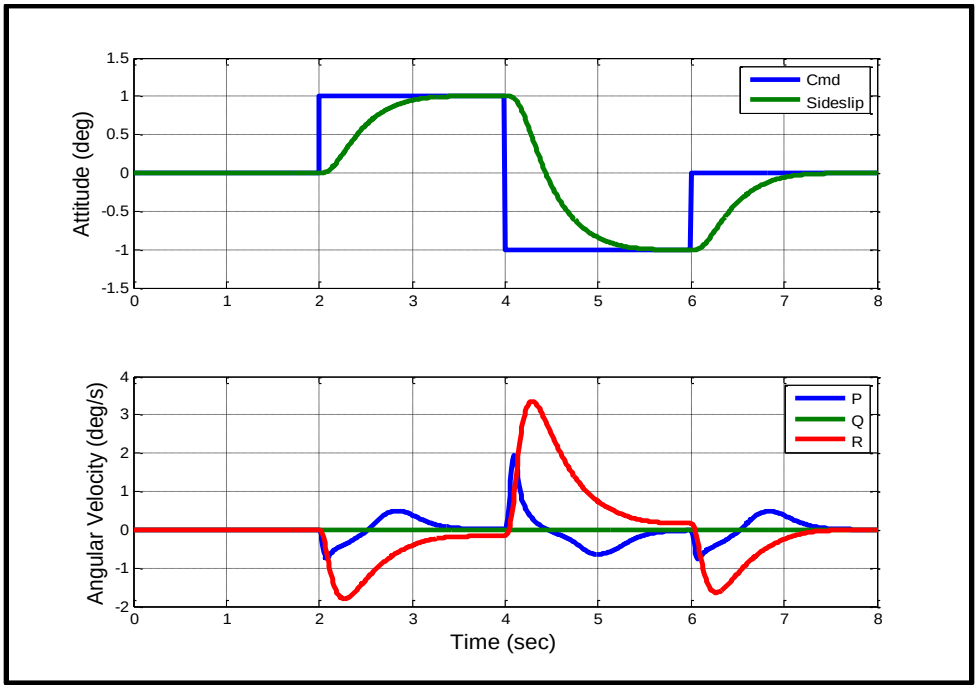

Figure 4-10: Sideslip Angle Step Response

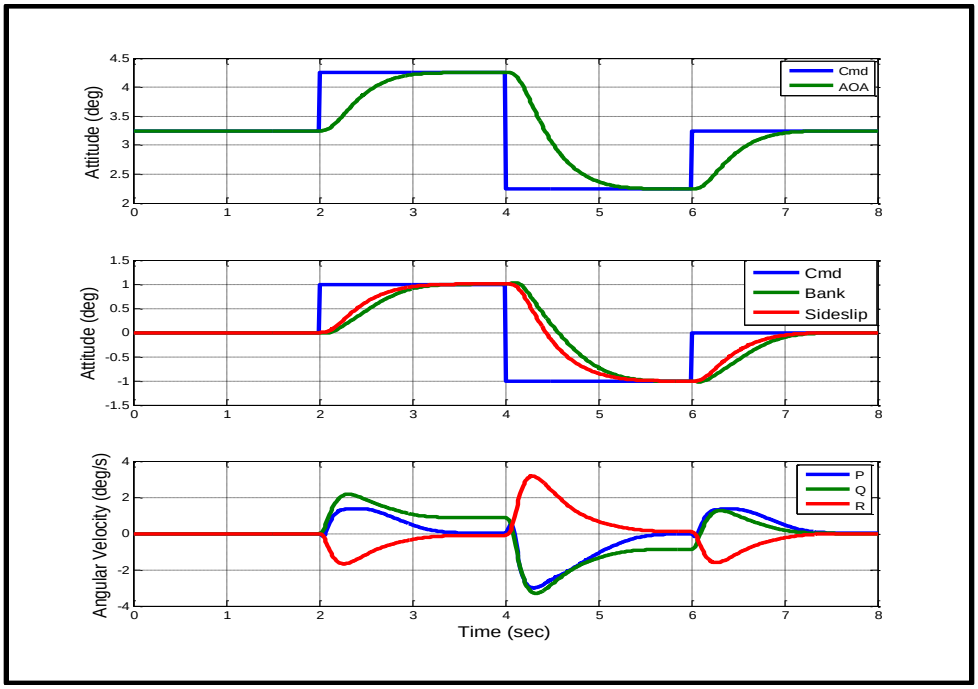

Figure 4-11: Coupled Attitude Loop Step Response

4.5.4 Velocity Controller Results

The selected gains for the velocity loop PI controller are presented in Table 4-3. The step response results for the independent channels are shown in Figure 4-12, Figure 4-13, and Figure 4-14 while the response for the coupled channels is shown in Figure 4-15. The coupled step response figure shows very good tracking performance of the guidance commands while maintaining the sideslip angle near zero demonstrating the ability of performing coordinated turn maneuvers.

Table 4-3: Position Controller PID Gains

Position Controller PID Gains	Values
Proportional Gain (K_P)	1.0
Integral Gain (K_I)	0.025
Feed Forward Gain	1.0

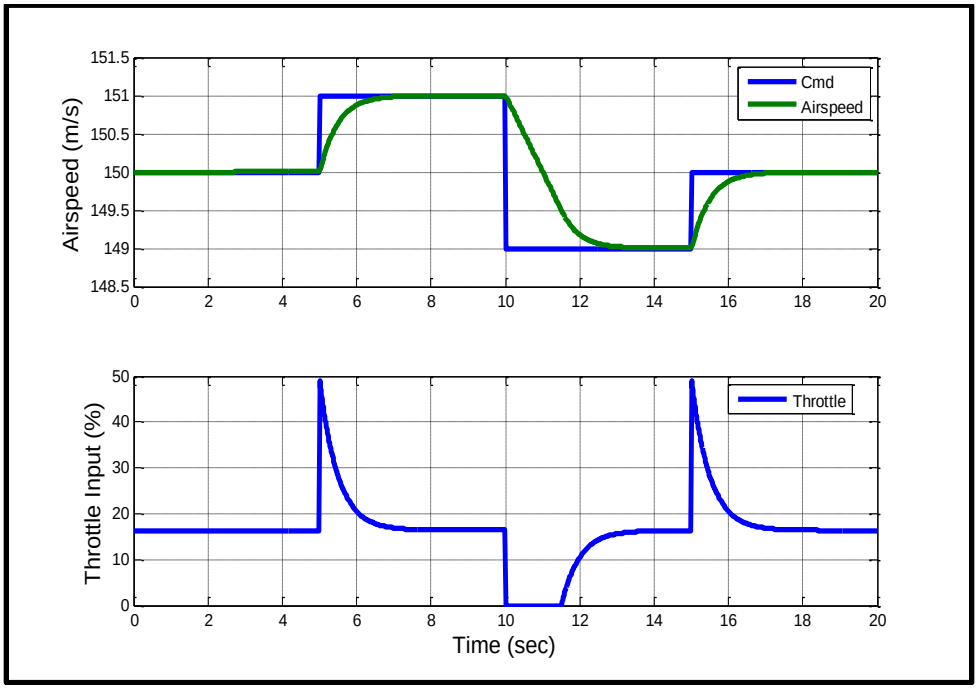

Figure 4-12: Airspeed Controller Step Response

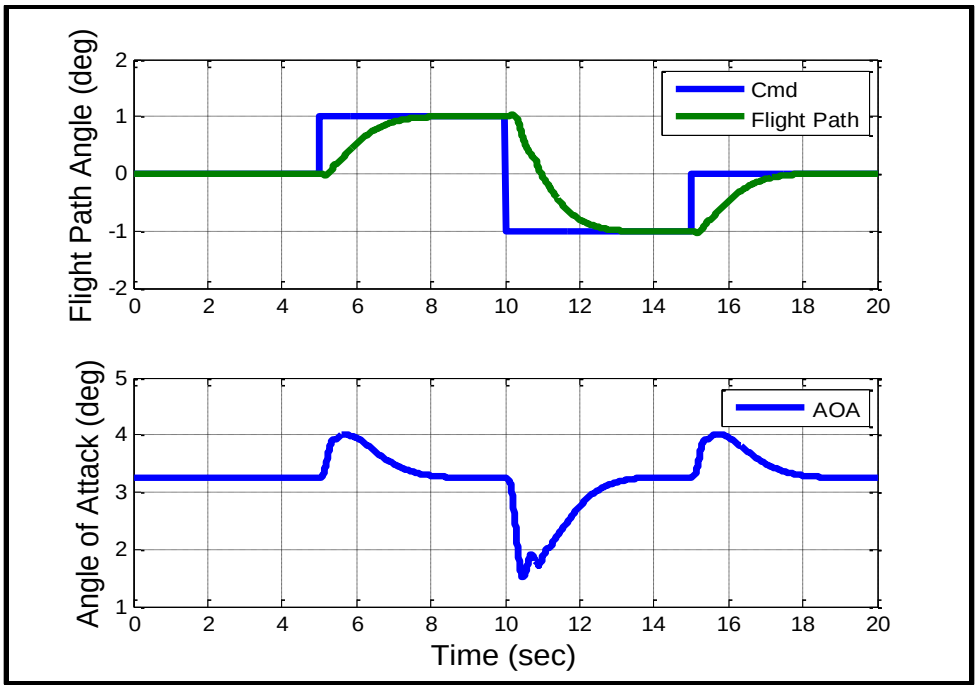

Figure 4-13: Flight Path Controller Step Response

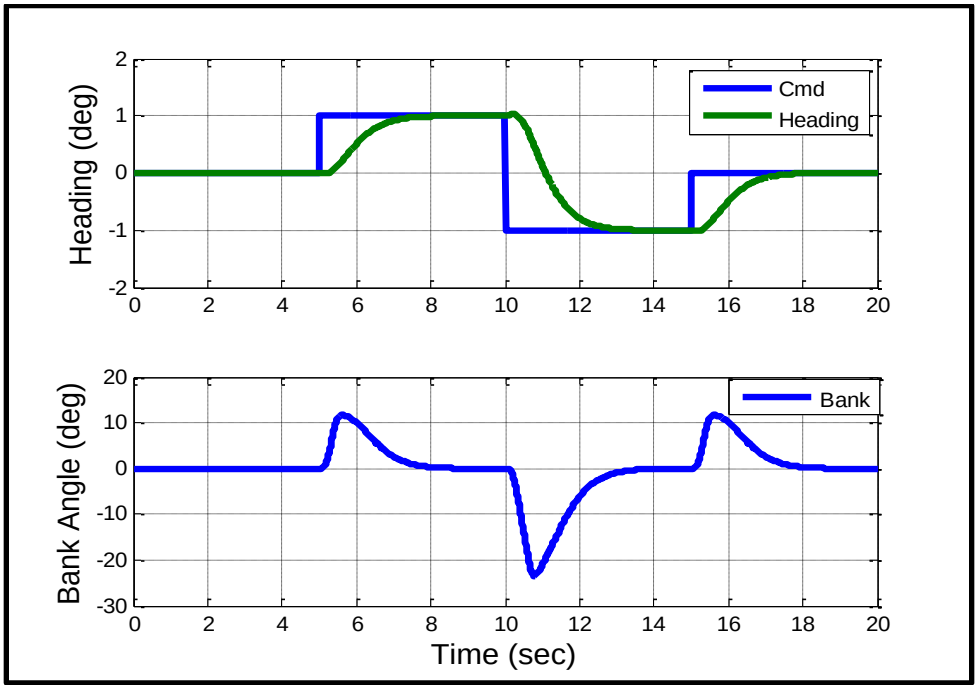

Figure 4-14: Heading Controller Step Response

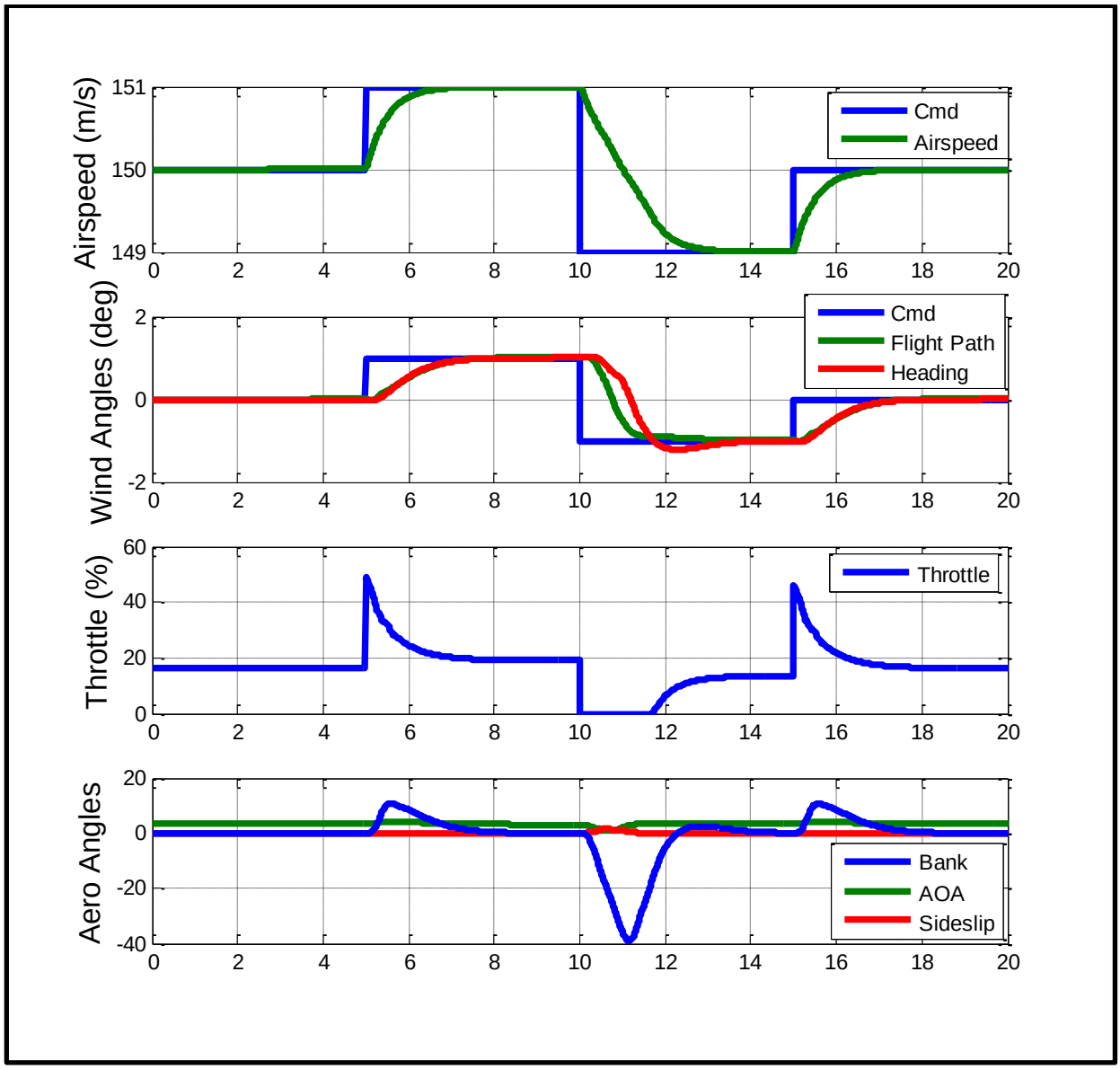

Figure 4-15: Coupled Velocity Controller Step Response

4.5.5 Trajectory Tracking Results

The nonlinear autopilot was tested against a sample guidance trajectory in order to check the actual performance of the autopilot when tracking continuous guidance commands. The sample trajectory consists of a combined climb and turn maneuver that was generated using the guidance algorithm presented in the following chapter. The initial and final conditions for this sample trajectory are shown in Table 4-4. Figure 4-16 shows the trajectory tracking performance of this autopilot design. It can be noticed that the commanded trajectory and the actual aircraft trajectory are on top of each other. Since this is a canned trajectory, the guidance commands are open loop which will create a drift between the commanded position and the actual aircraft position if the autopilot deviates from the guidance commands. For this particular case, the autopilot was capable of maintaining a position error of less than 1.5 meters during the entire trajectory. The response to the guidance commands and the attitudes and control inputs related to this sample trajectory are shown in figures Figure 4-17, Figure 4-18, and Figure 4-19.

Table 4-4: Sample Trajectory Initial and Final Parameters

PARAMETER	INITIAL CONDITION	FINAL CONDITION
North Position	**0 m**	**5,000 m**
East Position	**0 m**	**2,000 m**
Altitude	**1,000 m**	**2,000 m**
Airspeed	**150 m/s**	**150 m/s**
Flight Path Angle	**0°**	**0°**
Heading	**0°**	**60°**

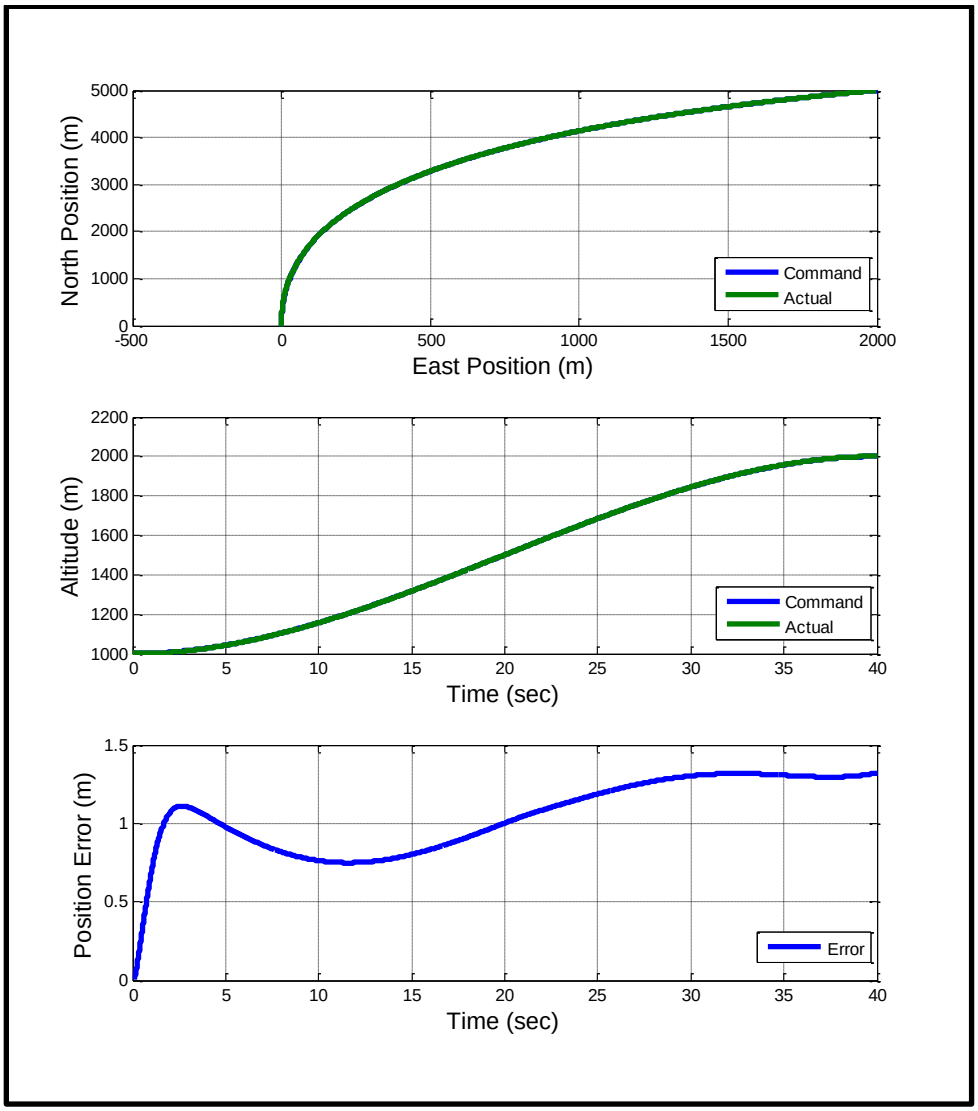

Figure 4-16: Sample Trajectory Tracking Response

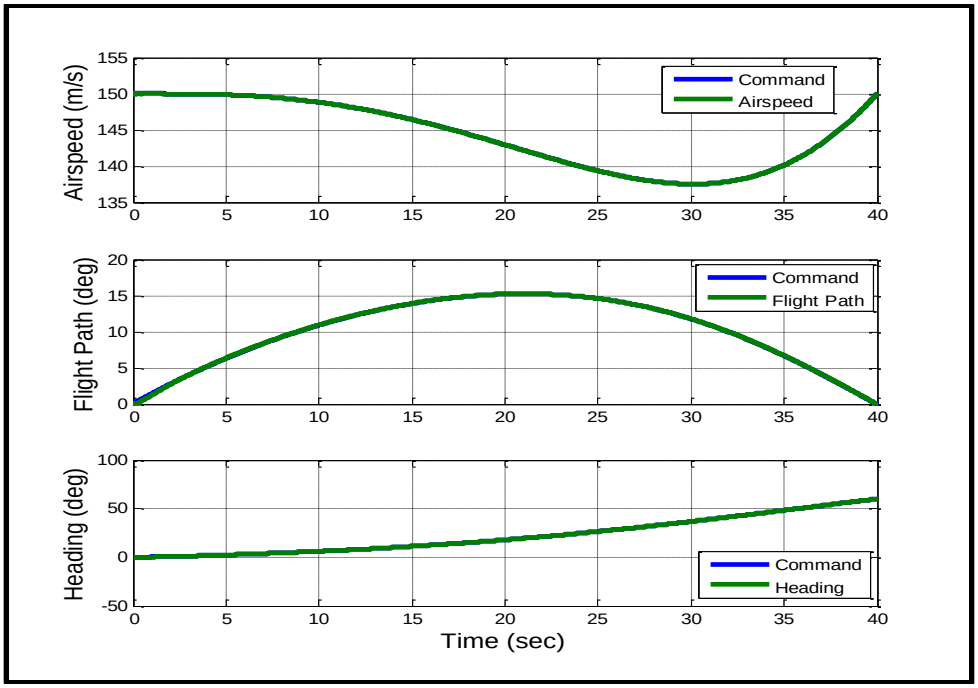

Figure 4-17: Sample Trajectory Guidance Commands and Response

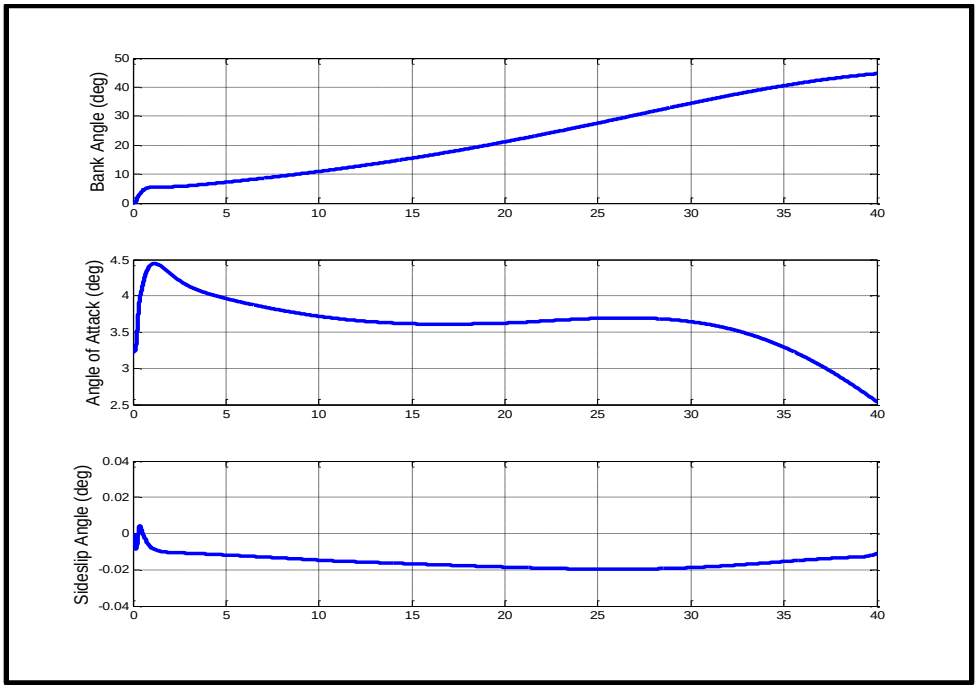

Figure 4-18: Sample Trajectory Attitude Response

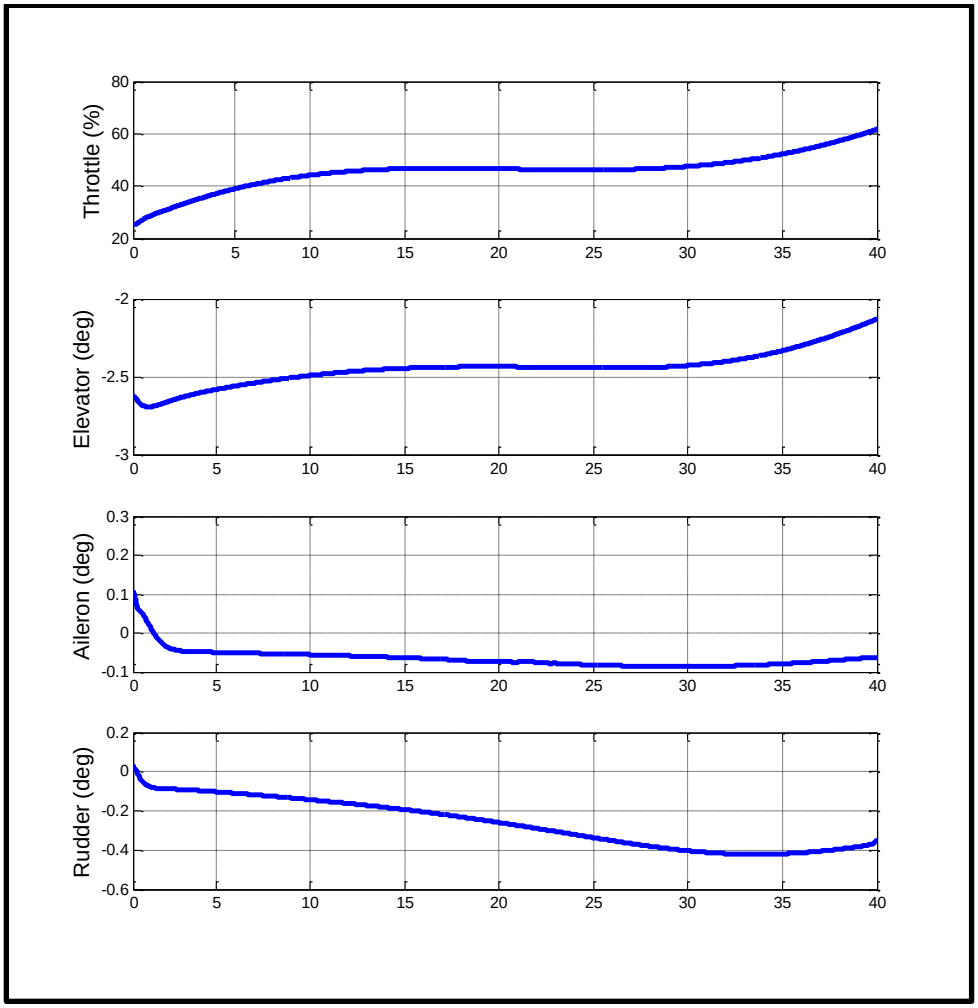

Figure 4-19: Sample Trajectory Control Inputs Response

CHAPTER 5: **COORDINATED ATTACK OF MULTIPLE UAVS**

The previous chapter provided a detailed solution to stabilize and control the dynamics of the UAV. With this accomplished, the next and final step is to develop a real time path planning algorithm that guides the UAV from its initial location to its final destination. This algorithm must generate trajectories that comply with all the necessary boundaries and constraints presented by the motion of an aircraft flying in a three dimensional (3-D) space. At the same time, these trajectories must also meet the requirements necessary to synchronize multiple UAVs to execute a coordinated attack on multiple targets which is one of the main objectives of this thesis.

The initial sections of this chapter will deal with the generation of a feasible trajectory for a single UAV that satisfies the kinematic constraints posed by fixed wing aircraft as well as the initial and final boundary conditions. The final boundary conditions can be known a priori if the UAV is attempting to suppress a fixed installation such as a SAM site, however, if the UAV is attacking a moving target such as a ground vehicle, an intercept point estimation algorithm will be utilized to determine these final conditions. After presenting the development of this single UAV trajectory design, the final sections will address how these independent trajectories are modified using optimization parameters to address the simultaneous arrival constraint required for the coordinated attack.

5.1 Aircraft Kinematic Constraints

The fixed wing aircraft kinematic equations play a crucial role in the development of feasible UAV trajectories since they impose a set of nonholonomic constraints that must be met by the selected path. These kinematic equations were explained in section 2.3, however, they were expressed in the body coordinate system. For guidance and control purposes, it is easier to express these equations in the wind coordinate frame. The translational kinematics can be expressed in the wind frame as follows:

$$\dot{P}_N = V_T \cos\gamma \cos\chi \quad \textbf{(5.1)}$$

$$\dot{P}_E = V_T \cos\gamma \sin\chi \quad \textbf{(5.2)}$$

$$\dot{P}_D = -V_T \sin\gamma \quad \textbf{(5.3)}$$

In the equations above, the airspeed (V_T) must be positive definite since fixed wing aircraft require forward speed to maintain flight. These equations will be utilized later to derive the desired guidance commands from the calculated trajectory.

5.2 Online Trajectory Planning for Single UAV

Before the creation of a coordination scheme that synchronizes multiple UAVs, it is necessary to develop a trajectory planning algorithm capable of guiding a single UAV from an initial boundary condition $q_0 = (x_0, y_0, z_0, V_{T_0}, \gamma_0, \chi_0)$ at time t_0 to its final boundary condition $q_f = (x_f, y_f, z_f, V_{T_f}, \gamma_f, \chi_f)$ at time t_f while meeting the fixed wing

aircraft nonholonomic constraints. Due to the dynamic environment in which these UAVs must fly (i.e. moving targets, obstacles, etc.), an online path planning algorithm that is capable of adjusting the UAV trajectory in real time must be employed for this application.

The designed path planning algorithm is mainly based in the findings portrayed on [1] which utilize polynomial parameterization to generate a set of smooth curves that meet the initial and final boundary constraints. An optimization parameter is used to select the best trajectory that meets the additional constraints imposed on the UAV such as collision avoidance criteria, minimum energy path, arc length requirements, etc. The minimum order of the polynomial required equals $d_0 + d_f + 1$ where d_0 and d_f represent the maximum derivative order of the initial and final boundary conditions. To meet the constraints q_0 and q_f presented above, the initial and final positions and velocities must be specified meaning that only first order derivatives are required. Therefore, a third order polynomial is the minimum order required to satisfy the boundary constraints. This third order polynomial provides a unique trajectory which may not necessarily meet the additional constraints required for coordination and collision avoidance so at least one optimization parameter must be added to the horizontal and vertical planes to create a group of potential trajectories from which a candidate can be selected. The 3-D parametric curves which define this group of trajectories are presented in the equations 5.4 to 5.6.

$$P_N(t) = x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad \textbf{(5.4)}$$

$$P_E(t) = y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 \quad \textbf{(5.5)}$$

$$-P_D(t) = z(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3 + c_4 t^4 \quad \textbf{(5.6)}$$

The coefficients b_4 and c_4 are the two optimization parameters which will be used to meet the coordination constraints and their calculations will be shown in the following sections. The remaining polynomial coefficients are computed from the boundary constraints as shown below.

$$[a_0 \quad a_1 \quad a_2 \quad a_3]^T = B^{-1} Y_1 \quad \textbf{(5.7)}$$

$$[b_0 \quad b_1 \quad b_2 \quad b_3]^T = B^{-1}(Y_2 - A b_4) \quad \textbf{(5.8)}$$

$$[c_0 \quad c_1 \quad c_2 \quad c_3]^T = B^{-1}(Y_3 - A c_4) \quad \textbf{(5.9)}$$

where,

$$A = \begin{bmatrix} t_0^4 & 4t_0^3 & t_f^4 & 4t_f^3 \end{bmatrix}^T \quad \textbf{(5.10)}$$

$$B = \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 \\ 0 & 1 & 2t_0 & 3t_0^2 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix} \quad \textbf{(5.11)}$$

$$Y_1 = \begin{bmatrix} x_0 & V_{T_0} \cos\gamma_0 \cos\chi_0 & x_f & V_{T_f} \cos\gamma_f \cos\chi_f \end{bmatrix}^T \quad \textbf{(5.12)}$$

$$Y_2 = \begin{bmatrix} y_0 & V_{T_0} \cos\gamma_0 \sin\chi_0 & y_f & V_{T_f} \cos\gamma_f \sin\chi_f \end{bmatrix}^T \quad \textbf{(5.13)}$$

$$Y_3 = \begin{bmatrix} z_0 & V_{T_0} \sin\gamma_0 & z_f & V_{T_f} \sin\gamma_f \end{bmatrix}^T \quad \textbf{(5.14)}$$

5.3 Estimation of Target Velocity and Intercept Point

The path planning algorithm presented in section 5.2 requires knowledge of the initial and final boundary conditions for the generation of the UAV trajectory. Most UAVs are capable of measuring the vehicle states such as position, velocity, and attitude with onboard sensors such as GPS and INS systems. Because of this, providing the initial conditions to the path planning algorithm is a straight forward task. Providing final boundary conditions of the UAV, on the other hand, could be a more difficult task since they are dependent on the behavior of the target that they intend to suppress. For the case of static targets, prior intelligence information can pre-programmed or communicated to the UAV satisfying the need for this final boundary data. However, in most cases, the UAV will be required to suppress a moving target or a target from which no prior information is available. In this case, it is necessary to utilize the onboard sensors of the aircraft to estimate the target position and velocity which will be used to determine the final boundary conditions required by the trajectory algorithm. For the purpose of this thesis, it is assumed that the UAV is equipped with a laser range finder or any other sensor capable of measuring the distance from the UAV to the target, as well as a gimbaled camera or similar device capable of tracking the target and measuring the relative line-of-sight (LOS) angles and rates. By having available the range and LOS information, it is possible to estimate the target velocity and the time and point on intercept.

5.3.1 Estimation of Target Position and Velocity

The first step towards determining the final conditions of the UAV is to characterize the behavior of the target which consists of estimating its current position and velocity. Assuming that the target's velocity is constant and that it is not maneuvering, a set of equations can be utilized to convert the range and relative line-of-sight measurements into estimated target motion. The equations presented here are based on [9]; however, they have been expanded to handle a 3-D environment. Figure 5-1 depicts the UAV to target 3-D engagement geometry.

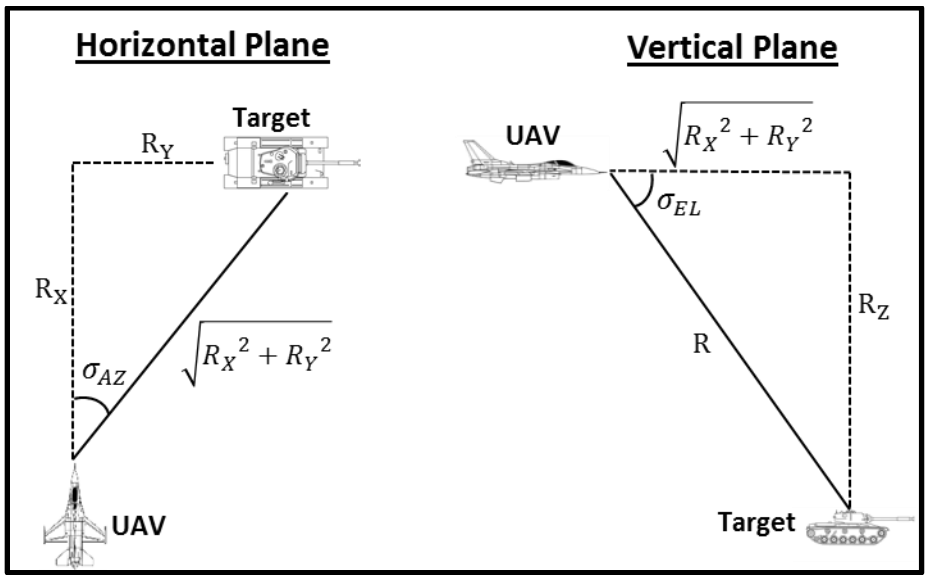

Figure 5-1: UAV to Target Geometry

The main three variables that define the UAV to target geometry are the range (R) and the azimuth (σ_{AZ}) and elevation (σ_{EL}) line-of-sight angles. These variables are calculated as shown in equations 5.15 to 5.17.

$$R = \sqrt{R_X^2 + R_Y^2 + R_Z^2} \quad \textbf{(5.15)}$$

$$\sigma_{AZ} = tan^{-1}\left(\frac{R_Y}{R_X}\right) \quad \textbf{(5.16)}$$

$$\sigma_{EL} = sin^{-1}\left(\frac{R_Z}{R}\right) \quad \textbf{(5.17)}$$

Assuming that the UAV is capable of measuring range and line-of-sight angles, the three individual range components can be calculated using equations 5.18 to 5.20.

$$R_X = R\cos\sigma_{AZ}\cos\sigma_{EL} \quad \textbf{(5.18)}$$

$$R_Y = R\sin\sigma_{AZ}\cos\sigma_{EL} \quad \textbf{(5.19)}$$

$$R_Z = R\sin\sigma_{EL} \quad \textbf{(5.20)}$$

These range components can then be added to the current UAV position $(x_{UAV}, y_{UAV}, z_{UAV})$ to estimate the target position as follows:

$$x_{TGT} = x_{UAV} + R\cos\sigma_{AZ}\cos\sigma_{EL} \quad \textbf{(5.21)}$$

$$y_{TGT} = y_{UAV} + R\sin\sigma_{AZ}\cos\sigma_{EL} \quad \textbf{(5.22)}$$

$$z_{TGT} = z_{UAV} + R\sin\sigma_{EL} \quad \textbf{(5.23)}$$

Once the target position is determined, the next step is to estimate its velocity. To accomplish this, it is first necessary to derive equations 5.15 to 5.17 to determine the closing velocity and the line-of-sight rates.

$$\dot{R} = \frac{R_X \dot{R}_X + R_Y \dot{R}_Y + R_Z \dot{R}_Z}{R} \quad \textbf{(5.24)}$$

$$\dot{\sigma}_{AZ} = \frac{R_X \dot{R}_Y - \dot{R}_X R_Y}{R_X^2 + R_Y^2} \quad \textbf{(5.25)}$$

$$\dot{\sigma}_{EL} = \frac{R\dot{R}_Z - \dot{R}R_Z}{R\sqrt{R_X^2 + R_Y^2}} \quad \textbf{(5.26)}$$

Equations 5.15 to 5.17 and 5.24 to 5.26 are the heart of the target velocity estimator algorithm. Substituting the range component terms shown in equations 5.24 to 5.26 with equations 5.18 to 5.20 and solving for the relative velocity components, equation 5.27 is obtained.

$$\begin{bmatrix} \dot{R}_X \\ \dot{R}_Y \\ \dot{R}_Z \end{bmatrix} = \begin{bmatrix} \cos\sigma_{AZ}\cos\sigma_{EL} & \sin\sigma_{AZ}\cos\sigma_{EL} & \sin\sigma_{EL} \\ -\sin\sigma_{AZ} & \cos\sigma_{AZ} & 0 \\ -\cos\sigma_{AZ}\sin\sigma_{EL} & -\sin\sigma_{AZ}\sin\sigma_{EL} & \cos\sigma_{EL} \end{bmatrix}^{-1} \begin{bmatrix} \dot{R} \\ R\cos\sigma_{EL}\dot{\sigma}_{AZ} \\ R\dot{\sigma}_{EL} \end{bmatrix} \quad \textbf{(5.27)}$$

Given the fact that the relative velocity components are equal to the subtraction between the target velocity vector and the UAV velocity vector, the target velocity components can be found using equation 5.28.

$$\begin{bmatrix} V_X \\ V_Y \\ V_Z \end{bmatrix}_{Target} = \begin{bmatrix} \dot{R}_X \\ \dot{R}_Y \\ \dot{R}_Z \end{bmatrix} + \begin{bmatrix} V_X \\ V_Y \\ V_Z \end{bmatrix}_{UAV} \quad \textbf{(5.28)}$$

5.3.2 Estimation of Intercept Point and Final Boundary Conditions

The first three elements of the final boundary conditions consist of the final coordinate (x_f, y_f, z_f) of the UAV trajectory which in this particular application is defined as the intercept point. Numerous techniques have been developed over the years to estimate this intercept point based on the current conditions of the target and the guided weapon. One of the oldest and simplest strategies is known as pure pursuit where the weapon aligns itself with the target and keeps flying toward the target until collision. This method proved to be very inefficient against moving targets since the weapon needs to maneuver during the entire engagement. The most common technique used today is known as lead collision where the weapon flies at an angle that leads the target trajectory. This technique is based on the fact that if both the target and the weapon are moving at a constant speed, collision between the two will occur if the weapon flies at a constant predetermined angle during the entire engagement. This is the main principle behind proportional navigation guidance which is the most used guidance algorithm in weapon technology. In proportional navigation, the turning acceleration command of the weapon is directly related to the relative line-of-sight rate since its main goal is to nullify this LOS rate and maintain a constant LOS angle during the engagement as required by the lead collision technique. Figure 5-2 illustrates the engagement geometry of a lead collision engagement.

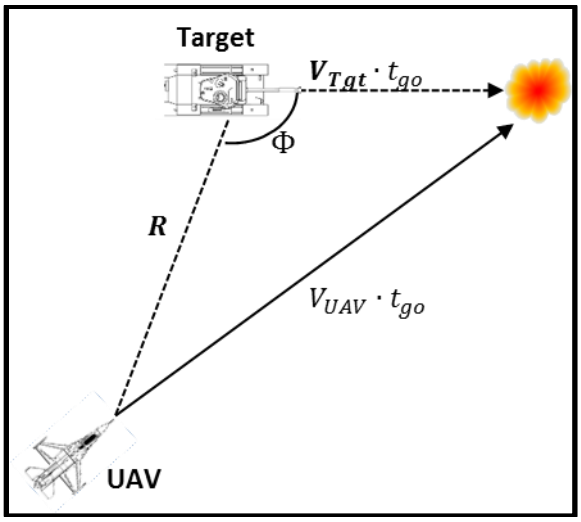

Figure 5-2: Lead Collision Geometry

The figure above shows that the lengths of two of the three sides of the engagement triangle are dependent on the time to go. Due to this reason, before being able to estimate the interception point, it is necessary to calculate this time to go. Using the law of cosines we obtain equation 5.29.

$$V_{UAV}^2 \, t_{go}^2 = |\boldsymbol{R}|^2 + \left|\boldsymbol{V_{tgt}}\right|^2 t_{go}^2 - 2\,|\boldsymbol{R}|\,\left|\boldsymbol{V_{tgt}}\right|\,t_{go}\,\cos\Phi \qquad \textbf{(5.29)}$$

The angle Φ can be determined by calculating the dot product of the range vector and the target velocity vector and solving for Φ as shown in equation 5.30.

$$\Phi = cos^{-1}\left(\frac{\boldsymbol{R}\cdot\boldsymbol{V_{tgt}}}{|\boldsymbol{R}|\,\left|\boldsymbol{V_{tgt}}\right|\,t_{go}}\right) \qquad \textbf{(5.30)}$$

Substituting equation 5.30 into equation 5.29, the time to go can be solved by using the quadratic formula as shown in equation 5.31.

$$t_{go} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad \textbf{(5.31)}$$

where,

$$a = \left|\boldsymbol{V_{tgt}}\right|^2 - V_{UAV}^2 \quad \textbf{(5.32)}$$

$$b = -2\left(\boldsymbol{R} \cdot \boldsymbol{V_{tgt}}\right) \quad \textbf{(5.33)}$$

$$c = |\boldsymbol{R}|^2 \quad \textbf{(5.34)}$$

In equation 5.31, only the positive definite root will be utilized as the time to go value. Also, in the equations above the scalar V_{UAV} is defined as the desired average UAV airspeed that must be achieved during the engagement timeline. By selecting an appropriate V_{UAV} value, the minimum (stall) and maximum UAV airspeed constraints can be implicitly met. After determining the time to go, the final boundary conditions can be calculated as shown in equations 5.32 to 5.38.

$$t_f = t_0 + t_{go} \quad \textbf{(5.35)}$$

$$x_f = x_{TGT} + V_X\, t_{go} \quad \textbf{(5.36)}$$

$$y_f = y_{TGT} + V_Y\, t_{go} \quad \textbf{(5.37)}$$

$$z_f = z_{TGT} + V_Z\, t_{go} \quad \textbf{(5.38)}$$

$$V_{T_f} = V_{UAV} \quad \textbf{(5.39)}$$

$$\chi_f = tan^{-1}\frac{(y_f - y_0)}{(x_f - x_0)} \quad \textbf{(5.40)}$$

$$\gamma_f = sin^{-1}\frac{(z_f - z_0)}{\sqrt{(x_f - x_0)^2 + (y_f - y_0)^2 + (z_f - z_0)^2}} \quad \textbf{(5.41)}$$

5.4 Trajectory Coordination for Multiple UAVs

The path planning algorithm presented in the sections above provide the necessary mechanisms to guide a single UAV from its initial location to the final intercept point allowing for a successful suppression of a single target. This might be sufficient for missions where only one target is engaged; however, other missions such as SEAD missions, multiple targets must be engaged requiring the use of multiple kamikaze UAVs. Since the element of surprise is always crucial in an attack, it is desired that these multiple UAVs arrive to their targets simultaneously. This simultaneous arrival requirement adds an additional constraint to the UAV trajectory design. Based on the trajectory planning algorithm presented in section 5.2, this constraint could be met by simply selecting the same final time (t_f) on all UAVs, however, since some UAVs might be farther away from their targets than others, the minimum velocity constraint might be violated causing some UAVs to stall. Due to this reason, besides selecting a common final time, it is also necessary to generate trajectories whose arc lengths are approximately equal. Therefore, the main coordination parameter required to synchronize the trajectories of multiple UAVs is defined as the reference arc length which is simply the maximum trajectory arc length of the UAV that is farthest away from

its target. Equation 5.42 shows how to calculate the reference arc length. The parameter L_i stands for the arc length of the i[th] UAV while N stands for the number of UAVs that must be coordinated. The common final arrival time can be calculated as shown in equation 5.42 where V_{UAV} is the average desired airspeed.

$$L_{ref} = \max(L_i) \quad , \quad i = 1 \dots N \tag{5.42}$$

$$t_f = \frac{L_{ref}}{V_{UAV}} \tag{5.43}$$

5.4.1 Estimation of Trajectory Arc Length

Since the trajectory arc length has been defined as the main coordination parameter, it is necessary to come up with a methodology to properly calculate this parameter. The arc length of a parametric curve is defined as the integral of the airspeed as shown in equation 5.44.

$$L = \int_{t_0}^{t_f} \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2 + \dot{z}(t)^2} \tag{5.44}$$

Equation 5.44 does not have an analytical solution so it can only be determined by utilizing a numerical method. A simple approach to this problem would be to simply estimate this arc length as the distance between the initial position and the final intercept point, however, if the UAV has enough processing power, a more accurate method should be used. The Newton-Cotes quadrature formula is a numerical

integration algorithm that can provide a better estimate of the arc length given that the value of the integrand is given at equally spaced points. For this estimation, the 5th order version of the Newton-Cotes method will be used whose formula is shown below.

$$L_{est} = \sum_{i=0}^{5} w_i \sqrt{\dot{x}(t_i)^2 + \dot{y}(t_i)^2 + \dot{z}(t_i)^2} \quad \textbf{(5.45)}$$

where,

$$w = \left[\frac{19}{288} \quad \frac{25}{96} \quad \frac{25}{144} \quad \frac{25}{144} \quad \frac{25}{96} \quad \frac{19}{288}\right] \quad \textbf{(5.46)}$$

$$t_i = t_0 + \frac{t_f - t_0}{5} i \quad , \quad i = 0 \dots 5 \quad \textbf{(5.47)}$$

$$\dot{x}(t_i) = a_1 + 2a_2 t_i + 3a_3 t_i^2 \quad \textbf{(5.48)}$$

$$\dot{y}(t_i) = b_1 + 2b_2 t_i + 3b_3 t_i^2 + 4b_4 t_i^3 \quad \textbf{(5.49)}$$

$$\dot{z}(t_i) = c_1 + 2c_2 t_i + 3c_3 t_i^2 \quad \textbf{(5.50)}$$

The arc length approximation consists of a sum of six points which are equally divided from t_0 to t_f as defined in equation 5.45. These points are multiplied by weight factors which are shown in equation 5.46. A higher accuracy approximation could be achieved if the interval $[t_0, t_f]$ is divided into smaller subintervals; however, the simulations results that are presented in section 5.6 prove that the approximations achieved by equation 5.45 are accurate enough.

5.4.2 Calculation of the Optimization Parameters

The parametric curves defined in equations 5.4 to 5.6 contain two optimization parameters which consist of the b_4 and c_4 polynomial coefficients. These parameters can modify the trajectory in the horizontal and vertical plane independently. In [1], these parameters are mainly utilized for collision avoidance but in this thesis their main focus is to modify the arc length of the trajectory to satisfy the requirements of the coordination constraint explained in equation 5.42. When no coordination is required, these parameters assume their nominal values of zero. Otherwise, these parameters must be determined to guarantee that the length of the planned trajectory matches the reference arc length. In this thesis only the b_4 parameter will be used for coordination. The c_4 coefficient will be maintained at its nominal value of zero though it could be used in the future to meet any additional constraints in the vertical plane.

An equation that relates the parameter b_4 with the reference arc length L_{ref} is needed to properly calculate the required value of b_4 for each UAV trajectory. Since equation 5.44 has no analytical solution, the next best candidate is equation 5.45. After estimating the arc lengths from all UAVs using the nominal b_4 value of zero, the reference arc length is determined using equation 5.42. Then, the left hand side of equation 5.42 is substituted by L_{ref} as shown in equation 5.51.

$$L_{ref} = \sum_{i=0}^{5} w_i \sqrt{\dot{x}(t_i)^2 + \dot{y}(t_i)^2 + \dot{z}(t_i)^2} \quad \textbf{(5.51)}$$

Unfortunately since equation 5.51 is a sum of six square root functions, solving directly for the parameter b_4 is not possible requiring the utilization of a numerical algorithm. By moving L_{ref} to the right hand side, the Newton-Raphson algorithm for finding the equation roots can be used to determine b_4. The Newton-Raphson is an iterative algorithm that quickly finds the approximate root of a function by evaluating the function and its first derivative at an initial guess of the root and then utilizes the resulting values to calculate a new estimated root. This new root is then utilized as the new guess and the process is repeated until the root converges to a certain tolerance. In this case, the iteration will be repeated a total of five times. Declaring the function of interest as equation 5.45 and the independent variable as b_4, the Newton-Raphson root finding algorithm for this application is defined by equation 5.52.

$$b_{4_{n+1}} = b_{4_n} - \frac{f(b_{4_n})}{f'(b_{4_n})} \quad , \quad n = 0 \ldots 5 \quad \textbf{(5.52)}$$

where,

$$f(b_{4_n}) = \sum_{i=0}^{5} w_i \sqrt{\dot{x}(t_i)^2 + \dot{y}(t_i)^2 + \dot{z}(t_i)^2} - L_{ref} \quad \textbf{(5.53)}$$

$$f'(b_{4_n}) = \sum_{i=0}^{5} w_i \frac{\dot{y}(t_i)\ddot{y}(t_i)}{\sqrt{\dot{x}(t_i)^2 + \dot{y}(t_i)^2 + \dot{z}(t_i)^2}} \quad \textbf{(5.54)}$$

The variables $w_i, t_i, \dot{x}(t_i), \dot{y}(t_i)$, and $\dot{z}(t_i)$ are defined by equations 5.46 to 5.50. The variable $\ddot{y}(t_i)$ is defined as the second order derivative with respect to the parameter b_4 as shown in equation 5.55.

$$\ddot{y}(t_i) = \frac{\partial[B^{-1}(Y_2 - Ab_4)[0 \quad 1 \quad 2t_i \quad 3t_i^2]^T + 4b_4 t_i^3]}{\partial b_4} \tag{5.55}$$
$$= B^{-1}(-A)[0 \quad 1 \quad 2t_i \quad 3t_i^2]^T + 4t_i^3$$

5.5 Derivation of Guidance Commands

The guidance commands required by the autopilot are composed of the airspeed, the flight path angle, and the heading. These commands can be determined from the calculated trajectory using the following equations:

$$V_{CMD} = \sqrt{\dot{x}(t) + \dot{y}(t) + \dot{z}(t)} \tag{5.56}$$

$$\gamma_{CMD} = \sin^{-1}\frac{\dot{z}(t)}{V_{CMD}} \tag{5.57}$$

$$\chi_{CMD} = \tan^{-1}\frac{\dot{y}(t)}{\dot{x}(t)} \tag{5.58}$$

5.6 Simulation Results

The online path planning algorithm design presented in the previous sections was implemented in the Simulink environment and interfaced to the nonlinear autopilot and the F-16 model in order to test its performance. This section provides a brief description of the Simulink model that was developed as well as the results obtained from it.

5.6.1 Path Planning Algorithm Simulink Implementation

A simulation consisting of a total of three UAVs and three targets was implemented to test the algorithms presented in this chapter. An overview of the Simulink model is shown in Figure 5-3.

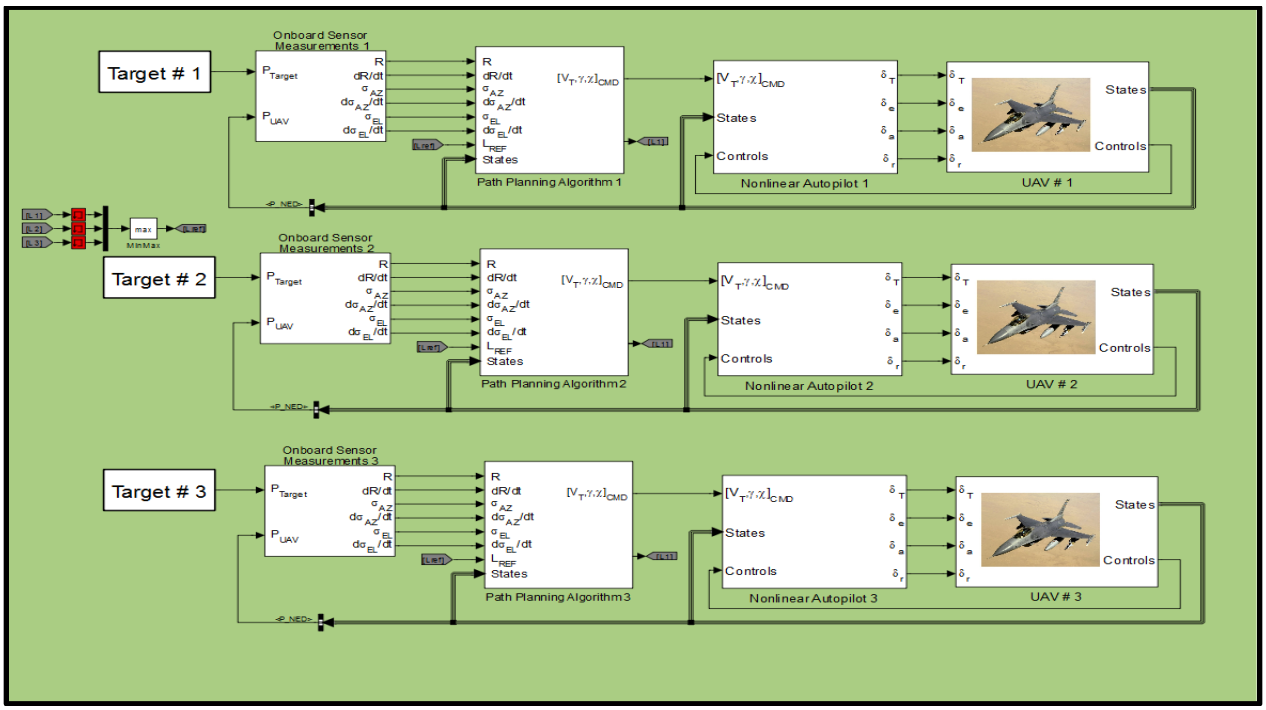

Figure 5-3: Coordinated Path Planning Simulink Model

As it can be noticed from the figure, the model contains five different block types. The first block from right to left is the UAV model presented in chapter 3. The second block models the autopilot presented in chapter 4. The third block contains the path planning algorithms that were explained in this chapter. The fourth block simulates the output signals of the UAV onboard sensors providing the range and LOS information required by the path planning algorithm. The inputs of this block are the current UAV position as well as the target position which is modeled by the last block. The target model (fifth block) consists of a simple kinematic model that generates a predefined trajectory.

An inside view of the path planning block is revealed in Figure 5-4 showing that it is mainly composed of two additional blocks. The block on the left side is responsible of estimating the target position and velocity from the measurements provided by the onboard sensors by using the equations shown in section 5.3.1. The second block calculates the UAV trajectory based on the equations provided in section 5.2 and the coordination constraints presented in section 5.4.

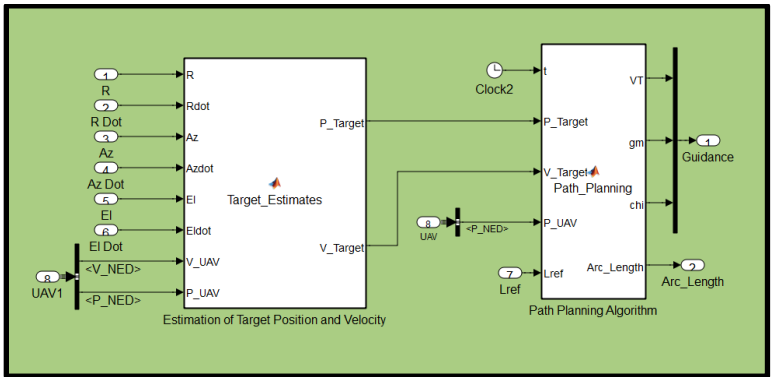

Figure 5-4: Path Planning Algorithm Model

5.6.2 Effects of the Optimization Parameters on the UAV Trajectory

The online path planning algorithm contains two optimization parameters that can be manipulated to modify the UAV trajectory in the horizontal and vertical planes. Even though only the b_4 parameter is used in this thesis, the results presented in this section provide a demonstration of how the trajectory shape and arc length are modified by selecting different values for b_4 and c_4. In addition, the accuracy of the arc length estimation algorithm presented in equation 5.45 is tested against this set of trajectories. The initial and final boundary conditions for this demonstration are shown in Table 5-1.

Table 5-1: Boundary Conditions for Demonstration of UAV Trajectory Modification

Parameter	Variable	Initial Conditions	Final Conditions
Time	t	0 s	36 s
North Position	x	0 m	5,000 m
East Position	y	0 m	0 m
Altitude	z	2,000 m	0 m
Velocity	V_T	150 m/s	150 m/s
Flight Path Angle	γ	0°	-21.8°
Heading	χ	0°	0°

The results shown in Figure 5-5 demonstrate how the b_4 parameter is capable of modifying the trajectory of the UAV by a simple change in value. In this particular example, it can be noted that the nominal value of $b_4 = 0$ achieves a direct path between the initial and final position while all other values of b_4 start deviating from this straight path increasing the trajectory arc length as shown in Table 5-2. This table also shows that the arc length estimation error of the sample trajectories is less than 1.3%.

Figure 5-6 shows the trajectory variations in the vertical plane caused by modifying the c_4 parameter. It can be noticed that the nominal value of zero creates a trajectory that is the closest to a straight path. All other values deviate from this closest path increasing the arc length as demonstrated by the results shown in Table 5-3.

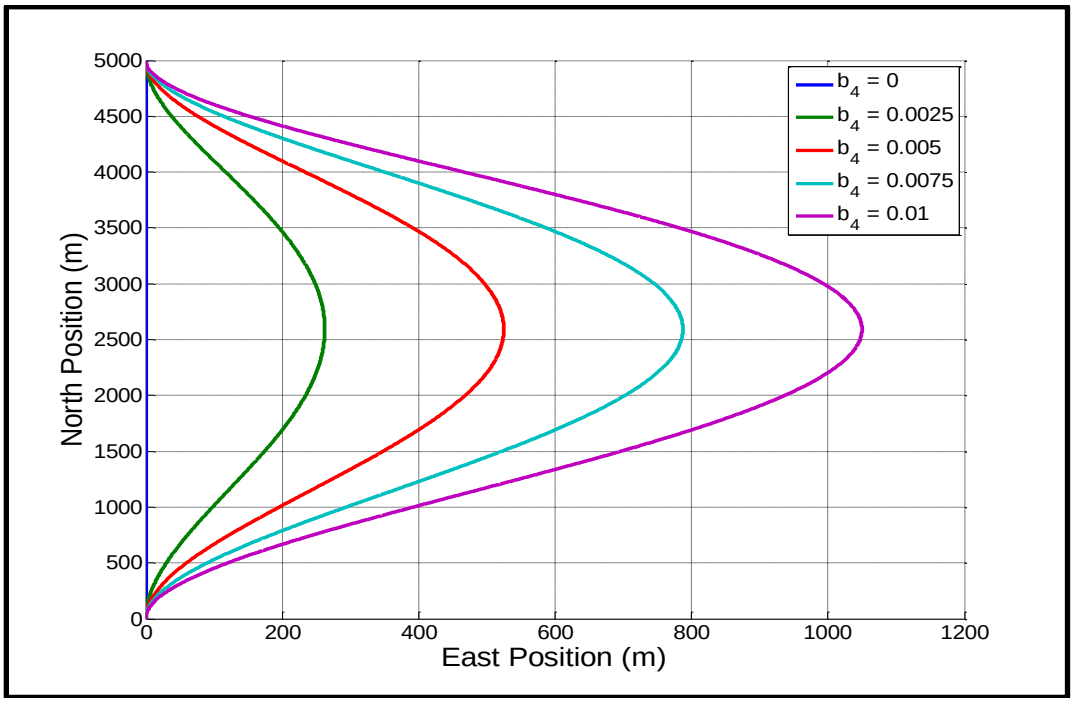

Figure 5-5: Variations on UAV Trajectory caused by b4 Parameter

Table 5-2: Effect of b4 Parameter on Arc Length and Estimation Accuracy

b_4 Value	Arc Length	Estimated Arc Length	Error
0.0000	5,454.0 m	5,452.5 m	0.03%
0.0025	5,484.7 m	5,488.6 m	0.07%
0.0050	5,575.4 m	5,594.8 m	0.35%
0.0075	5,721.9 m	5,765.6 m	0.76%
0.0100	5,918.5 m	5,993.5 m	1.27%

Table 5-3: Effect of c4 Parameter on Arc Length and Estimation Accuracy

b_4 Value	Arc Length	Estimated Arc Length	Error
0.0000	5,437.2 m	5,435.7 m	0.03%
0.0025	5,512.2 m	5,514.5 m	0.04%
-0.0025	5,415.1 m	5,418.8 m	0.07%
0.005	5,638.3 m	5,653.3 m	0.26%
-0.005	5,446.4 m	5,464.8 m	0.34%

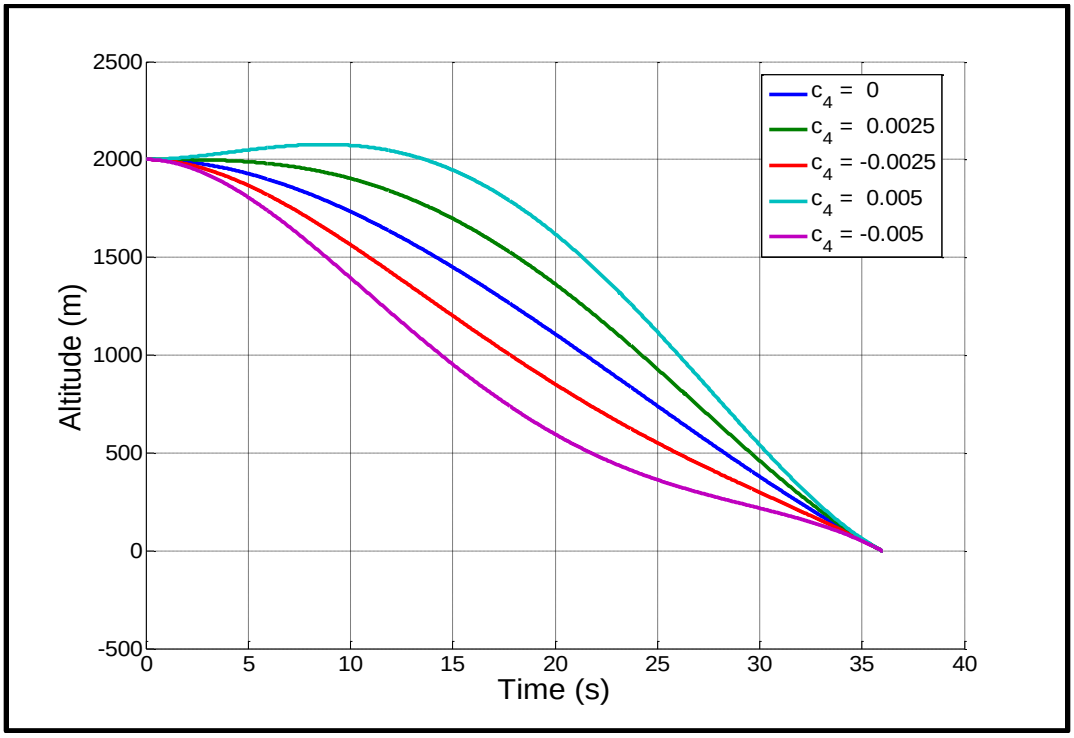

Figure 5-6: Variations on UAV Trajectory caused by c4 Parameter

5.6.3 Simulation Results for Single UAV Attack

The first test performed to evaluate the path planning algorithm was the simple case of a single UAV attempting to suppress a moving target. The initial conditions for this scenario are presented in Table 5-4. As it was explained in section 5.3, the path planning algorithm requires knowledge of the target position and velocity which can be

estimated using the range and LOS information provided by the UAV sensors. The target velocity estimation results for this scenario are shown in Figure 5-7 which demonstrate very good estimation accuracy. Figure 5-8 and Figure 5-9 show the trajectory followed by the UAV to intercept its target. The guidance commands associated with this trajectory are shown in Figure 5-10.

Table 5-4: Single UAV Scenario Initial Conditions

SCENARIO PARAMETER	SYMBOL	UAV	TARGET
Initial North Position (m)	x_0	0	5,000
Initial East Position (m)	y_0	0	0
Initial Altitude (m)	z_0	1,000	0
Initial Velocity (m/s)	V_0	150	25
Initial Flight Path Angle (°)	γ_0	0	0
Initial Heading (°)	χ_0	0	90

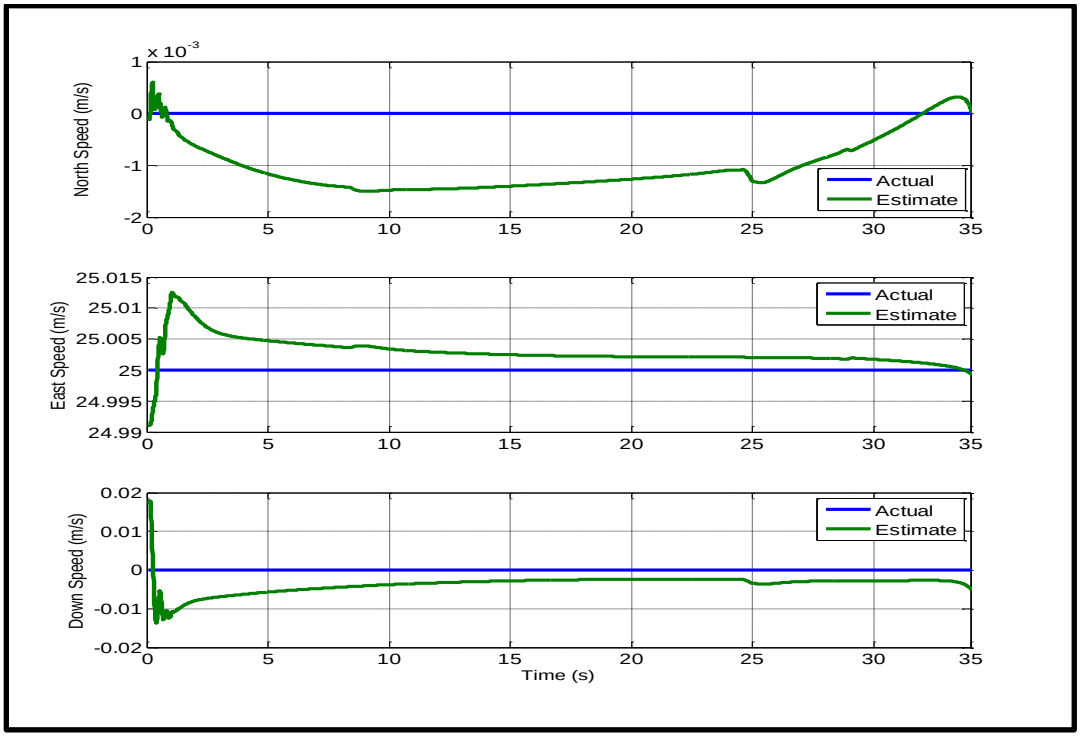

Figure 5-7: Target Velocity Estimation Results

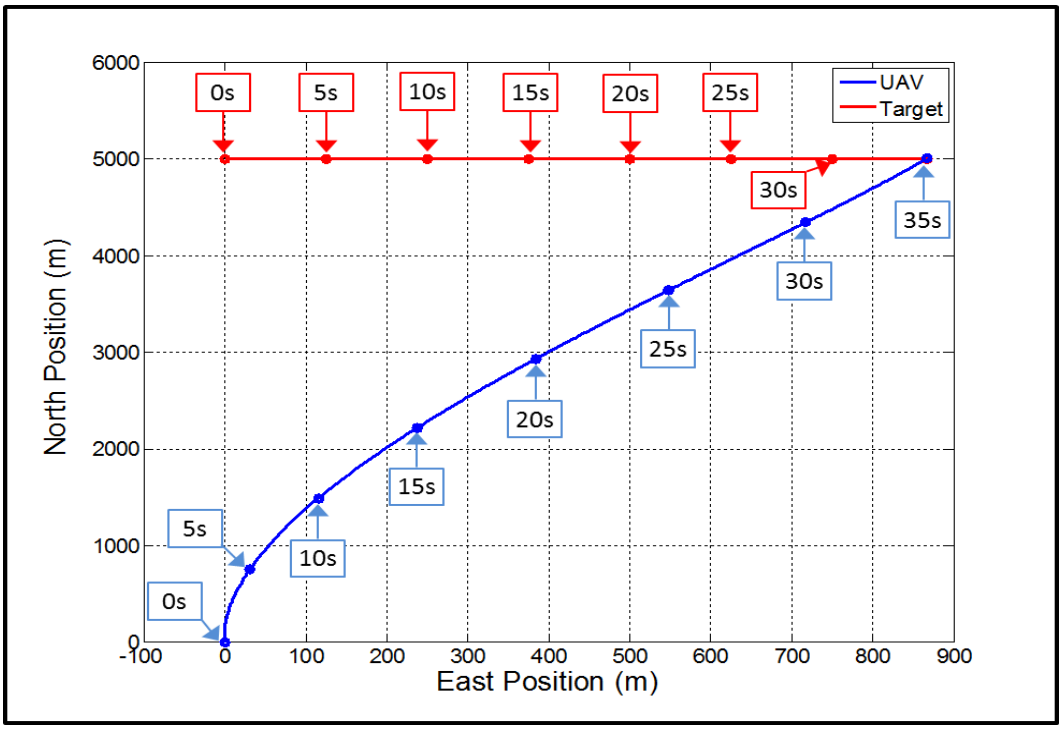

Figure 5-8: Single UAV Attack Scenario Trajectory Plot

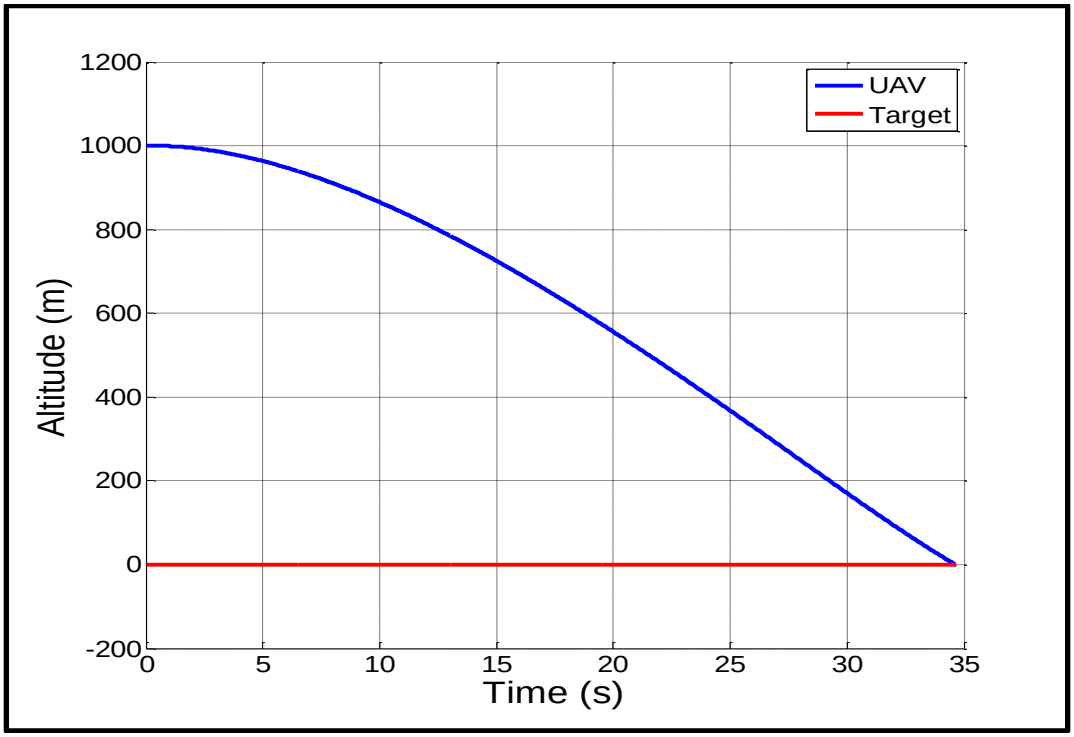

Figure 5-9: Single UAV Attack Scenario Altitude Plot

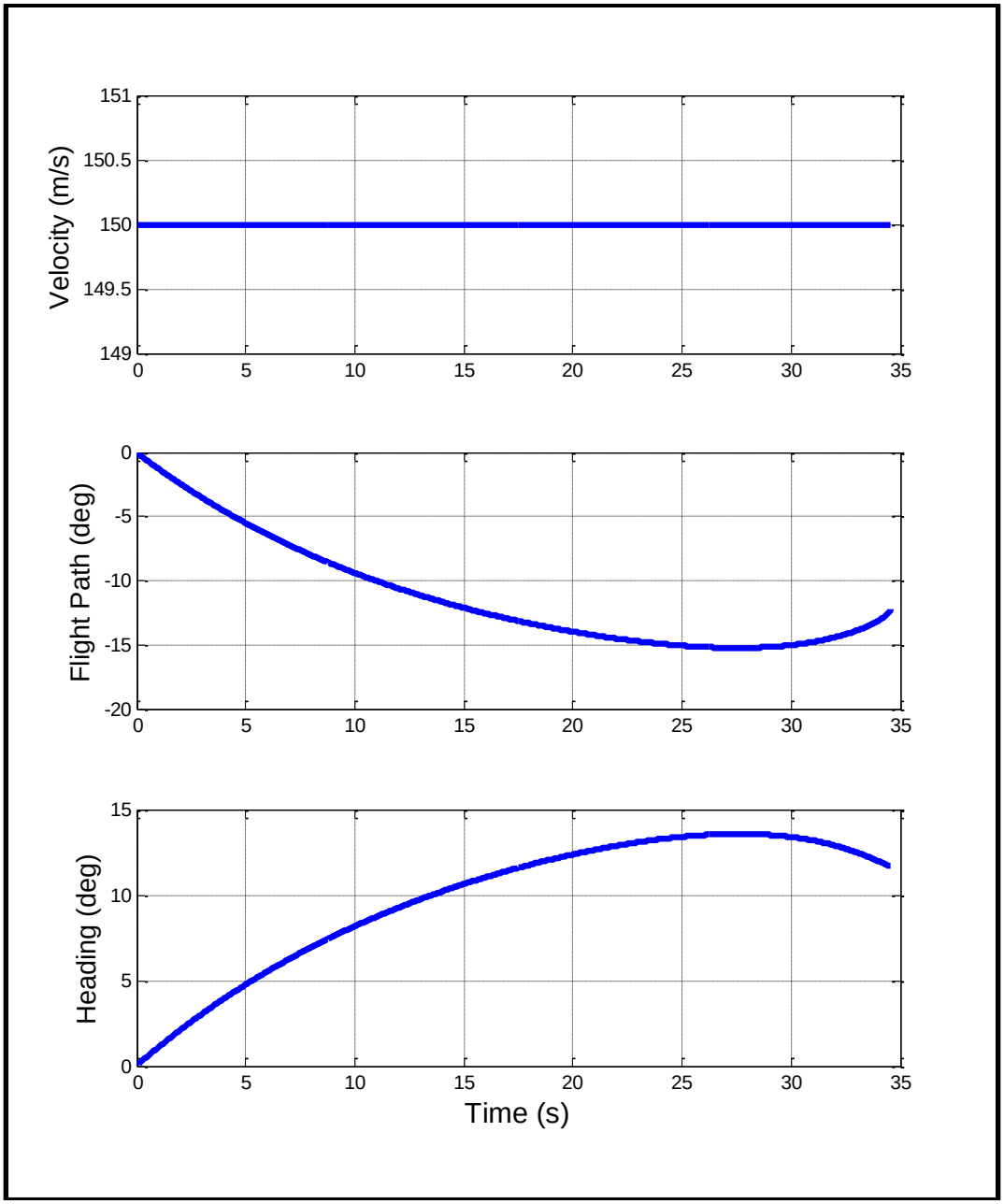

Figure 5-10: Single UAV Attack Scenario Guidance Commands

5.6.4 Simulation Results for Coordinated Attack of Multiple UAVs

After proving the performance of the online path planning algorithm against a single target, the next step was to simulate an engagement of multiple UAVs attacking three targets. In this scenario, the UAVs are executing a SEAD mission where one of the targets is static which is representative of a fixed SAM site installation or a radar system. The other two targets simulate launch vehicles where one of them is moving north while the other is moving eastbound. The initial conditions for this scenario are presented in Table 5-5.

The results presented in Figure 5-11 and Figure 5-12 show the trajectories followed by the three UAVs and the moving targets. Notice that the static target is located at the origin. In Figure 5-11, UAV 1 is located the farthest away from its target. For this reason, it follows a direct lead collision path similar to the one shown in the results for the single UAV attack. UAVs 2 and 3, on the other hand, are closer to their targets causing them to deviate from this lead collision path to match their arc lengths to the reference length which in this case is defined as the arc length of the UAV 1 trajectory. Figure 5-12 shows the altitude plots for all three UAVs. It can be noticed in this plot that all vertical trajectories follow their nominal shape since the c_4 parameters remained at zero during the simulations. Figure 5-13 shows the UAV to target ranges as a function of time. Once again, it can be noticed that the range UAV 1 is linearly decreasing since it is following a direct path while the other two UAVs were commanded to initially deviate from their

targets. The guidance commands for all three UAVs associated with this coordinated attack scenario are shown in Figure 5-14.

Table 5-5: Coordinated Attack Scenario Initial Conditions

Parameter	Symbol	UAV 1	UAV 2	UAV 3	TGT 1	TGT 2	TGT 3
North Position (m)	x_0	1,000	-2,000	-6,000	100	0	0
East Position (m)	y_0	9,000	-8,000	5,000	0	0	100
Altitude (m)	z_0	1,000	1,500	1,250	0	0	0
Velocity (m/s)	V_0	150	150	150	25	0	15
Flight Path Angle (°)	γ_0	0	0	0	0	0	0
Heading (°)	χ_0	-90	20	0	0	0	0

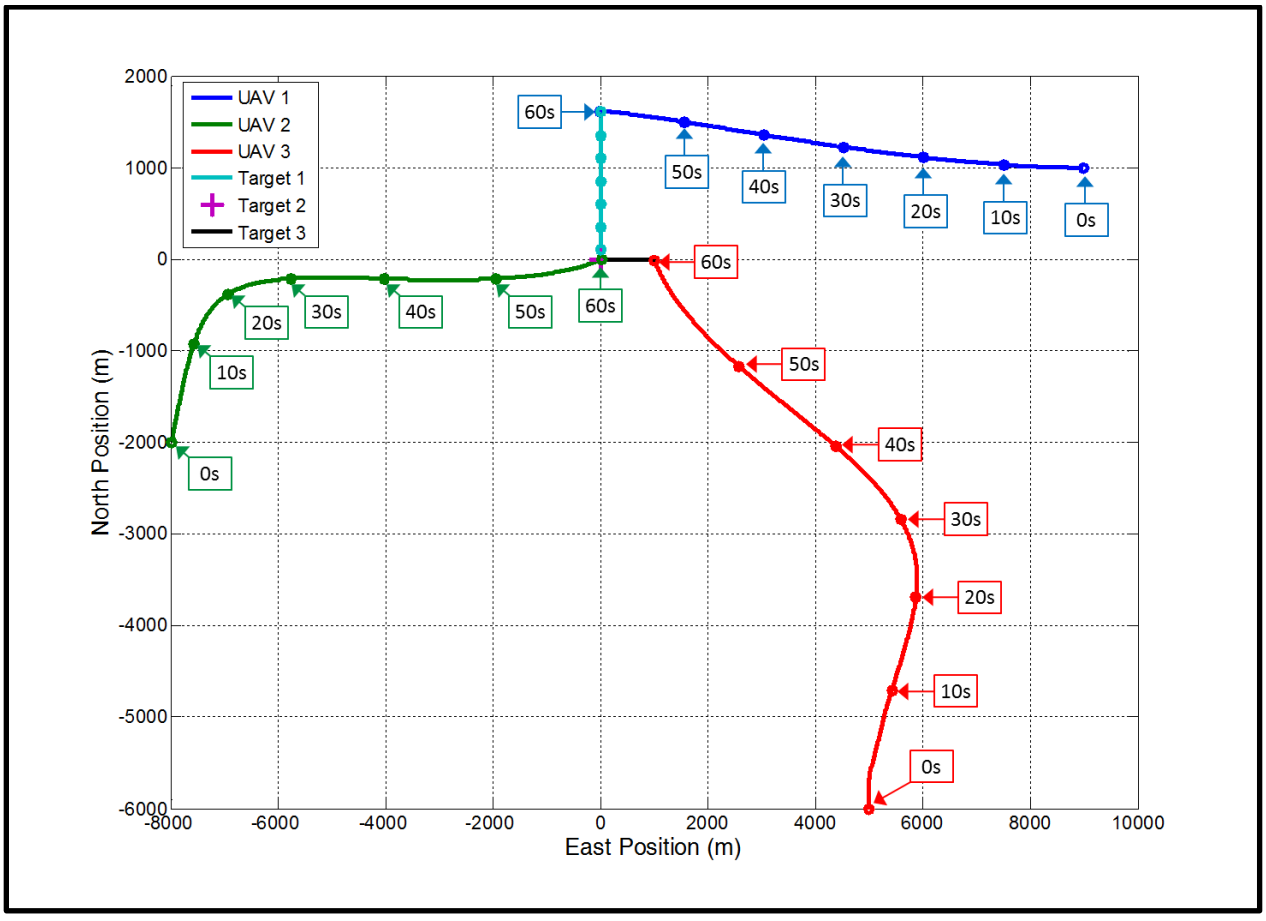

Figure 5-11: Coordinated Attack Scenario Trajectory Plot

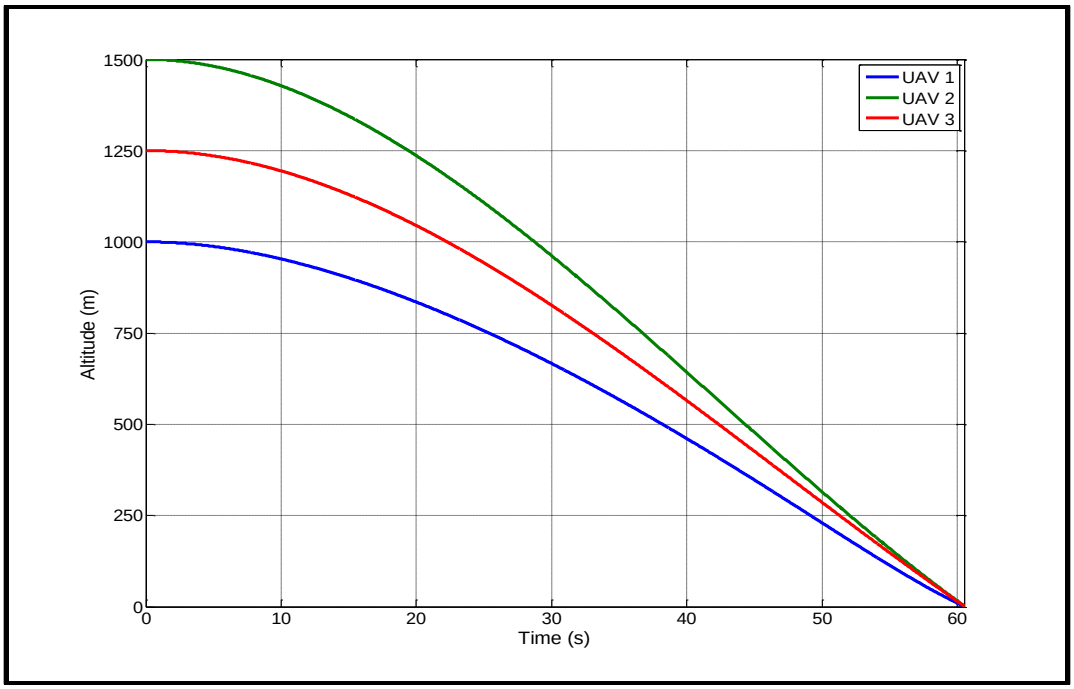

Figure 5-12: Coordinated Attack Scenario Altitude Plot

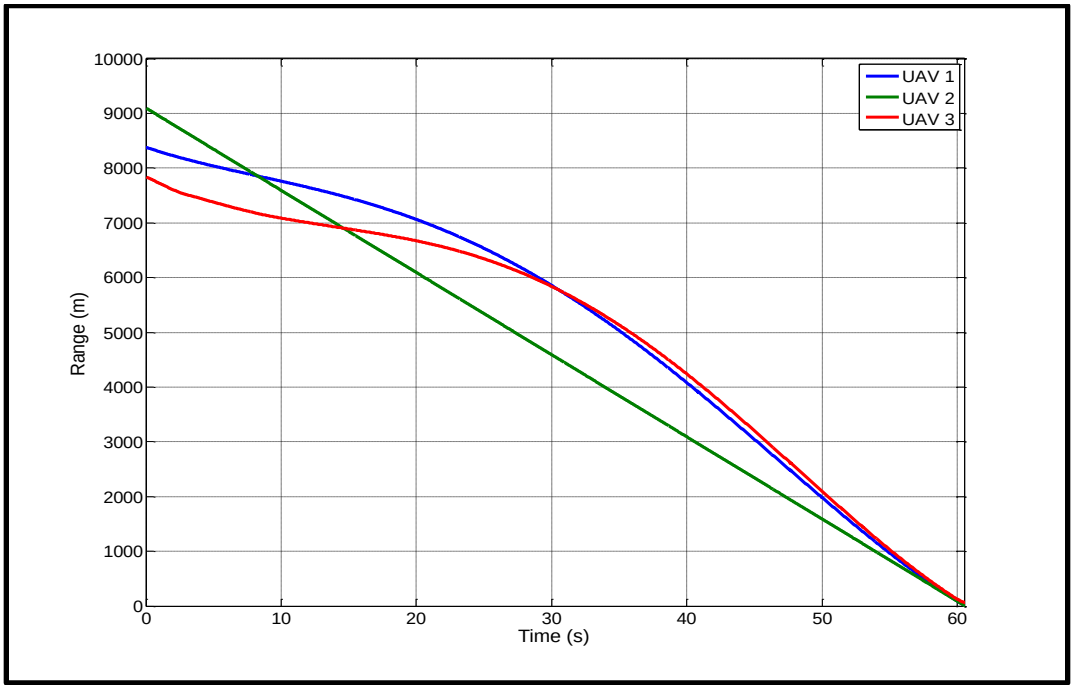

Figure 5-13: Coordinated Attack Scenario Range Comparison

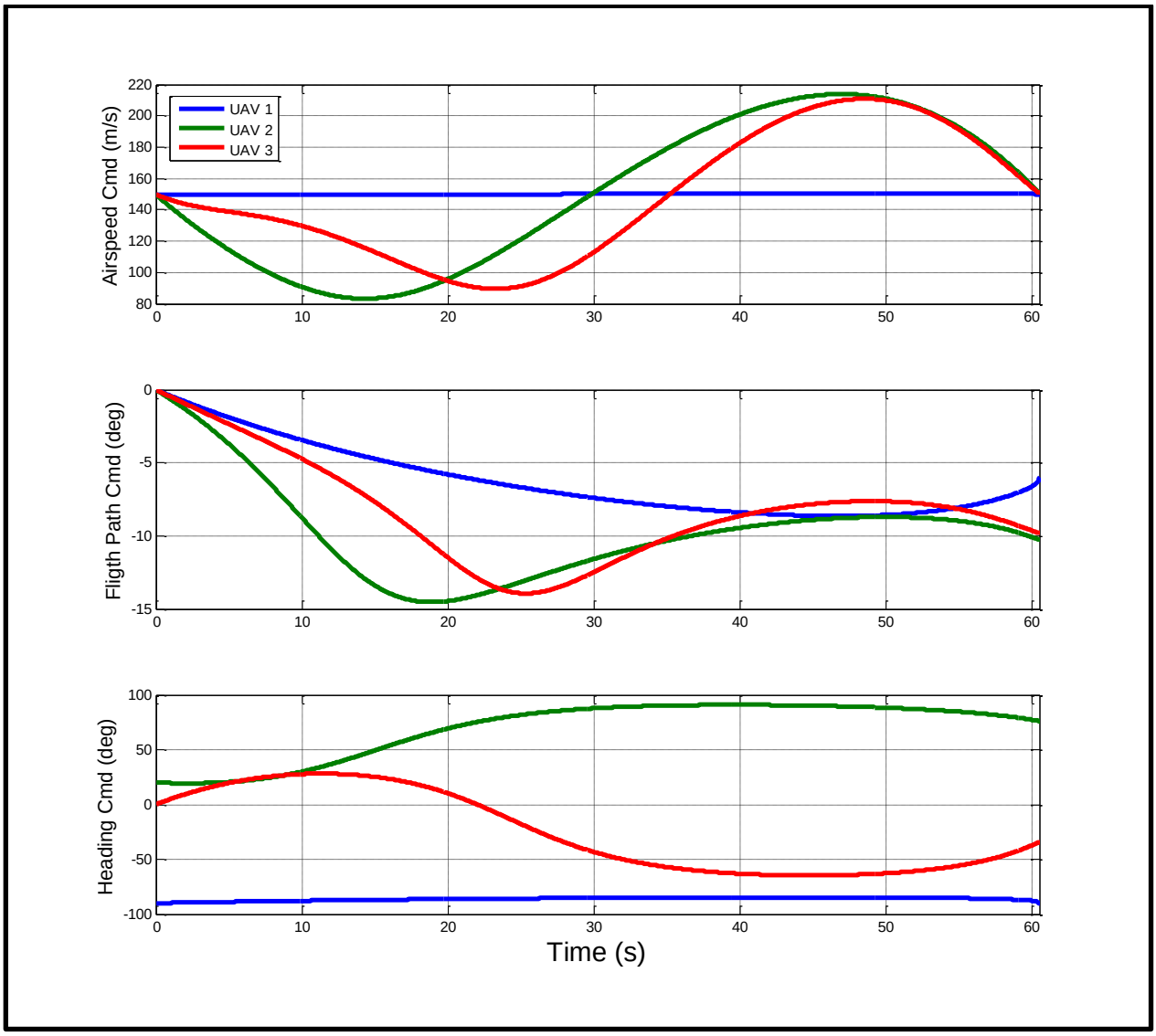

Figure 5-14: Coordinated Attack Scenario Guidance Commands

5.7 Future Considerations

The online path planning algorithm presented in this chapter provided a solution to generate coordinated trajectories for multiple UAVs that suffice the kinematic constraints of fixed wing aircraft and the simultaneous arrival time constraint. However, this path planning algorithm currently does not take into account the dynamic constraints of fixed wing aircraft with the exception that the velocity must be positive definite. Additional work must be performed to include the dynamic constraints such as the minimum and maximum airspeeds as well as maximum turn rate and the maximum climb rate. This can be accomplished by including an additional set of constraints in the b_4 and c_4 optimization parameters as shown in the equations below.

$$V_{MIN} < V_{CMD}(t, b_4, c_4) = \sqrt{\dot{x}(t) + \dot{y}(t, b_4) + \dot{z}(t, c_4)} < V_{MAX} \quad \textbf{(5.59)}$$

$$-\dot{\chi}_{MAX} < \dot{\chi}_{CMD}(t, b_4) = \frac{\dot{x}(t)\ddot{y}(t, b_4) - \ddot{x}(t)\dot{y}(t, b_4)}{\dot{x}(t)^2 + \dot{y}(t, b_4)^2} < \dot{\chi}_{MAX} \quad \textbf{(5.60)}$$

$$-\dot{h}_{MAX} < \dot{z}(t, c_4) < \dot{h}_{MAX} \quad \textbf{(5.61)}$$

CHAPTER 6: **CONCLUSION**

The main topics presented in this thesis consisted of the development of a high fidelity nonlinear UAV model, a nonlinear autopilot capable of stabilizing and control the UAV dynamics, and a path planning algorithm that permits the synchronization of multiple UAV trajectories to coordinate a simultaneous attack of multiple targets.

The nonlinear model, presented in chapter 3, showed the details on how to implement a nonlinear 6-DOF model of the F-16 aircraft in the Matlab/Simulink simulation environment. This model, based on a NASA mathematical model [2], provided a high fidelity simulation of the F-16 which was the UAV platform selected for this thesis. The aerodynamics model was replaced with a reduced version presented in [3] since the model shown by NASA was highly complex and deemed unnecessary for the purposes of this thesis. A future upgrade to this model could be to implement the numerous look-up tables presented in the NASA paper which provides a wider range of aerodynamic angles allowing the simulation of high angle of attack maneuvers. Also, utilizing these tables would provide more accurate representation of the cross coupling between the longitudinal and lateral dynamics of the aircraft.

The nonlinear autopilot design, presented in chapter 4, showed an approach on how to stabilize the UAV plant and how to calculate the control inputs (throttle, aileron, elevator, and rudder) necessary to follow a commanded trajectory by properly tracking the

guidance commands. The results presented in this chapter demonstrated the successful tracking abilities of the autopilot design which is mainly based on the dynamic inversion technique. This autopilot design, however, requires prior knowledge of the aircraft aerodynamic coefficients which would needed to be pre-programmed for each specific type of UAV. An alternative approach to this, which can be considered in the future, is to implement a real time aircraft parameter estimation [8] in conjunction with the autopilot algorithm making it a very flexible algorithm which could be installed on any UAV without the need of providing any specific parameters of the aircraft.

The online path planning algorithm, presented in chapter 5, provided a solution for the main objective of this thesis which is to perform a coordinated attack on multiple targets, which can be either static or moving, by manipulating the trajectories of the UAVs to guarantee a simultaneous arrival of all UAVs into their targets. The simulation results provided demonstrated that the presented algorithm is capable of synchronizing the trajectories with only one communication parameter which is the estimated arc length of the nominal UAV trajectory. Since only one parameter is used, the communication bandwidth can be kept at a minimum which is ideal for this type of applications. The results also showed that the UAV that is farthest away from its target follows the nominal lead collision trajectory while the other UAVs deviate from this trajectory to achieve a common arrival time. One of the current limitations of this algorithm is that since the velocity is directly related to the derivative of the calculated position values $\left(V = \sqrt{\dot{x}(t) + \dot{y}(t) + \dot{z}(t)}\right)$, the velocity command cannot be controlled independently of

the generated trajectory which can cause deviations from the desired nominal velocity. Future work can be performed in this area by attempting to decouple space and time following an approach similar to the ones shown in [10], [11], and [12]. Also, the integration of the dynamic constraints explained in section 5.7 and the collision avoidance criterion presented in Dr. Qu's paper [1] can be implemented in conjunction with the algorithms presented in this thesis to provide a more ruggedized solution.

LIST OF REFERENCES

[1] J. Yang, Z. Qu, J. Wang and R. A. Hull, "A Real-Time Optimized Path Planning for a Fixed Wing Flying in a Dynamic and Uncertain Environment," in *IEEE International Conference on Advanced Robotics*, 2005.

[2] L. T. Nguyen, M. E. Ogburn, W. P. Gilbert, K. S. Kibler, P. P. Brown and P. L. Deal, "NASA Technical Paper 1538: Simulator Study of Stall/Post Stall Characteristics of a Fighter Plane with a Relaxed Longitudinal Static Stability," NASA, Hampton, VA, 1979.

[3] B. L. Stevens and F. L. Lewis, Aircraft Control and Simulation, 2nd ed., Hoboken, NJ: John Wiley & Sons, Inc., 2003.

[4] H. K. Khalil, Nonlinear Systems, 3rd ed., Upper Saddle River, NJ: Prentice Hall, 2002.

[5] Z. Xie, Y. Xia and M. Fu, "Robust Trajectory Tracking Method for UAV using Nonlinear Dynamic Inversion," in *IEEE 5th International Conference on Cybernetics and Intelligent Systems (CIS)*, Qingdao, China, 2011.

[6] X. Wang, R. Xie and Y. Li, "Nonlinear Controller for a Supermaneuverable Aircraft," in *International Joint Conference on Computational Sciences and Optimization*, Sanya, China, 2009.

[7] W. Lei and L. Wang, "Nonlinear Flight Control Design of a Combat Flying Wing with High Aspect-ratio," in *The 2nd International Conference on Computer Application and System Modeling*, Shenyang, China, 2012.

[8] G. Campa, M. Napolitano, B. Seanor, M. Fravolini and Y. Song, "Application of an Improved LWR Method to Real-Time Aircraft Parameter Identification Problems," in *Proceedings of the American Control Conference*, Anchorage, AK, 2002.

[9] R. A. Bednar, "A Method for Estimating Target Acceleration and its Use in a Missile Guidance Law," Naval Weapons Center, China Lake, CA, 1977.

[10] I. Kaminer, O. Yakimenko, V. Dobrokhodov, A. Pascoal, N. Hovakimyan, C. Cao, A. Young and V. Patel, "Coordinated Path Following for Time-Critical Missions of Multiple UAVs via L1 Adaptive Output Feedback Controllers," in *AIAA Guidance, Navigation, and Control Conference*, Hilton Head, SC, 2007.

[11] O. A. Yakimenko, "Direct Method for Rapid Prototyping of Near-Optimal Aircraft Trajectories," *Journal of Guidance, Control, and Dynamics,* vol. 23, no. 5, pp. 865-875, 200.

[12] M. Shanmugavel, A. Tsourdos, R. Zbikowski, B. A. White, C. A. Rabbath and N. Lechevin, "A Solution to Simultaneous Arrival of Multiple UAVs using Pythagorean Hodograph Curves," in *Proceedings of the 2006 American Control Conference*, Minneapolis, MN, 2006.