Electron Rocket Launch and Recovery Performance Analysis

Development of Simulink Simulation Tools for Ascent and Descent Trajectories with Parafoil and Unguided Circular Parachute Recovery Mechanisms

Yama Shewan

Image credit: Mack Crawford [1]

Mack Crawford | NASASpaceFlight.com

Electron Rocket Launch and Recovery Performance Analysis

Development of Simulink Simulation Tools for Ascent and Descent Trajectories with Parafoil and Unguided Circular Parachute Recovery Mechanisms

by

Yama Shewan

To obtain the degree of Master of Science in Aerospace Engineering
at the Delft University of Technology.
To be defended publicly on Monday March 31, 2025 at 9:30 AM.

Student number:	4109686	
Project duration:	November 15, 2023 – August 28, 2024	
Thesis committee:	Ir. M.C. Naeije,	TU Delft, supervisor
	Ir. WJ. F. Simons,	TU Delft, examiner
	Dr. Ir. E. Mooij,	TU Delft, chair

TU Delft

Version Control

Table 1: Version Table

Version	Date	Changes
1.0	10-07-2024	Creation of draft version ascent phase of the trajectory
1.1	3-03-2025	Creation of draft version ascent and descent phase of the trajectory
2.0	21-03-2025	Final version

Preface

This thesis report follows the literature study report and is submitted in partial fulfillment of the requirements for the degree of Master of Science in Aerospace Engineering at Delft University of Technology.
First and foremost, I would like to express my deepest gratitude to my supervisor, Marc Naeije, for providing me with an excellent thesis topic and for his unwavering support throughout the entire process, especially during the academic and personal challenges I encountered. I would also like to extend my heartfelt thanks to my family and friends for their constant moral support and encouragement, which has been invaluable throughout this journey.

Yama Shewan
Delft, March 21, 2025

Abstract

Rocket Lab has recovered the first stage of its dedicated small satellite launcher Electron twice by means of splashdown. The ability to reuse the entire first stage can reduce production cost at the current launch frequency and enable a higher launch rate. To lower the launch cost further, Rocket Lab is developing Mid-Air Recovery system for the first stage of the Electron. The research objective of this thesis is to develop a simulation tool for analyzing the launch performance of the Electron rocket and its first-stage recovery, involving mid-air recovery using an air-ram parachute and splashdown recovery using an unguided circular parachute. For the ascent phase, a two-stage rocket Simulink model is developed that implements quaternion representation of six-degrees-of-freedom (6-DOF) equations of motion for custom variable mass in the Earth-centered Earth-fixed (ECEF) reference frame to simulate ascent trajectories. Another Simulink model is created to simulate the ballistic trajectory of the first stage after separation using quaternion representation of 6-DOF equations of motion for constant mass in the ECEF reference frame, particularly when first stage recovery is not implemented. It was found that he additional mass, which is required to transform the Electron's first stage into a reusable one is 100 kg, with the parafoil accounting for 74 kg, the ballute 9 kg, and the remaining mass attributed to modifications such as the parachute attachment and release mechanism and for the circular parachute 71.5 kg with 22.8 kg for canopy, 35.3 kg for suspension lines and 13.4 kg actuators. The parafoil provides superior control, whereas the circular parachute is lighter and more straightforward in design. Consequently, if refurbishment costs are identical for both, the circular parachute emerges as the more advantageous choice.

Nomenclature

Abbreviation	Description
AGU	Airborne Guidance Unit
C.O.M	Centre of Mass
DA	Direct Ascent
DARE	Delft Aerospace Rocket Engineering
DCM	Direction Cosine Matrix
DOF	Degrees of Freedom
DRL	Down Range Landing
E.O.M	Equation of Motion
ECEF	Earth-Centered, Earth-Fixed Position to Latitude, Longitude, and Altitude (LLA)
ELV	Expendable Launch Vehicle
FOS	Factors of Safety
GSO	Geosynchronous Orbit
GTOW	Gross Take-off Weight
HAT	Hohmann Transfer Ascent
HTHL	Horizontal Takeoff and Horizontal Landing
IAC	In-Air-Capturing
ICAO	International Civil Aviation Organization
IP	Interior-Point
LEO	Low Earth Orbit
LLA	Latitude, Longitude, and Altitude
LOX	Liquid Oxygen
LV	Launch Vehicle
MAR	Mid-Air Recovery
NED	North-East-Down
NRLMSISE	Naval Research Laboratory Mass Spectrometer and Incoherent Scatter Radar Exosphere
PID	Proportional Integral Derivative
PMA	Pneumatic Muscle Actuator
PSO	Particle Swarm Optimization
RLV	Reusable Launch Vehicles
RP-1	Rocket Propellant One (refined kerosene)
RTLS	Return to Launch Site
SMILE	Small Innovative Launcher for Europe
SISO	Single-Input Single-Output
SSO	Sun-Synchronous Orbit
TE	Trailing Edge
TUDAT	TU Delft Astrodynamics Toolbox
US76	The US Standard Atmosphere 1976
VTHL	Vertical Takeoff and Horizontal Landing
VTVL	Vertical Takeoff, Vertical Landing

List of Symbols

Symbol	*Description*	*Unit*
α	angle of attack	rad
a_{sp}	spatial angle of attack	rad
a_0	zero-lift angle of attack	rad
b	wingspan	m
β	angle of sideslip	rad
C_D	drag force coefficient	–
C_{D_0}	Drag coefficient at zero lift	–
$C_{D_{\alpha^2}}$	Proportionality factor for induced drag	–
$C_{D_{\delta_s}}$	Drag coefficient due to asymmetric control deflection	–
C_I	rolling moment coefficient	–
C_L	lift force coefficient	–
C_{L_0}	Lift coefficient at zero angle of attack	–
C_{L_α}	Lift-curve slope coefficient versus angle of attack	–
$C_{L_{\delta_s}}$	Lift coefficient due to symmetric control deflection	–
C_m	pitching moment coefficient	–
C_{m_0}	Pitching-moment coefficient at zero lift	–
C_{m_α}	Pitching-moment coefficient due to angle of attack	–
C_{m_q}	Pitch-damping coefficient due to pitch rate	–
C_n	yawing moment coefficient	–
C_{n_β}	Yawing-moment coefficient due to sideslip	–
C_{n_p}	Yawing-moment coefficient due to roll rate	–
C_{n_r}	Yawing-moment coefficient due to yaw rate	–
$C_{n_{\delta_\alpha}}$	Yawing-moment coefficient due to control deflection	–
C_S	side force coefficient	–
C_{Y_β}	Side-force coefficient due to sideslip	–
C_{Yb}	Side-force coefficient due to sideslip	–
C_{Yda}	Side-force coefficient due to asymmetric control input	–
C_{Yp}	Side-force coefficient due to roll rate	–
C_{Yr}	Side-force coefficient due to yaw rate	–
Cp	Pressure coefficient (P/Q)	–
D	drag	N
e	eccentricity	–
$\mathbf{F}$	force vector	N
$\mathbf{F}_I$	all external forces expressed in the inertial frame	N
$\mathbf{F}_C$	coriolis force as a result of time variations in mass distribution	N
$\mathbf{F}_{rel}$	relative force as a result of time variations in mass distribution	N
f	flattening (or oblateness) of the Earth	–
g	acceleration due to Earth gravity	m/s^2
G	universal gravity constant	m^3/kgs^2
h	altitude (h = -z)	m

i	inclination	rad
J_2, J_3, J_4	gravity-field harmonics	–
L	lift	N
L	rolling moment	Nm
$L_{\mathbf{S}}$	length of suspension lines	ft
m	mass	kg
$\dot{m}_f$	mass flow	kg/s
$\mathbf{M}$	moment vector	Nm
M	Mach number	rad
M	mean anomaly	Nm
M	pitching moment	rad/s
$\mathbf{M}_{cm}$	all external moment about c.o.m. of the vehicle	Nm
$\mathbf{M}_C$	coriolis moment as a result of time variations in mass distribution	Nm
$\mathbf{M}_{rel}$	relative moment as a result of time variations in mass distribution	Nm
N	yawing moment	Nm
$N_{\mathbf{G}}$	number of gores (radials) in the canopy	–
N_{SL}	number of suspension lines	–
p	roll rate	rad/s
q	pitch rate	rad/s
$\boldsymbol{Q}$	quaternion vector	–
r	yaw rate	rad/s
r	modulus of position vector	m
$\boldsymbol{r}$	position vector	m
$\tilde{\boldsymbol{r}}$	location vector of a mass element	m
R_e	equatorial radius	m
S	side force	N
S_{ref}	aerodynamic reference area	m^2
S_0	surface area of the finished canopy	ft^2
t	time	s
T	thrust vector	N
u, v, w	cartesian velocity components	m/s
v	velocity component	m/s
$\mathbf{V}$	velocity vector	m/s
$\bar{V}_e$	mean exhaust velocity	m/s
x, y, z	cartesian position components	m
X, Y, Z	aerodynamic forces in body frame	N
X, Y, Z	axes	–
λ_s	geocentric latitude at mean sea level	rad
γ	Earth-referenced flight-path angle	rad
μ	rigging angle	deg
${}^n_bR$	rotation matrix from lower reference frame to upper reference frame	–

List of Figures

List of Tables

Contents

1 Introduction

Launching satellites into space is very expensive. The number of space launches is increasing year by year. This growth is due to technological advancements in space technology and the development of commercial launch systems, which have drastically reduced the cost of space launches. For example, launching 1 kilogram (kg) of payload into Low Earth Orbit (LEO) with NASA's Space Shuttle would have cost $54,500, whereas with SpaceX's Falcon 9 it costs only $2,720. This is a cost reduction by a factor of 20 for LEO. For SpaceX's Falcon Heavy, the space launch cost per kg to LEO is $1,400, which is a cost reduction by a factor of 40. To increase the access to space for scientists, universities and companies further reduction in space launch cost is needed. Reusability of the launch vehicle parts is the most effective way to reduce the launch cost. As such, more and more private and national companies are developing partially and/or fully reusable launch vehicles. The current frontrunners are SpaceX (with Falcon 9 and Falcon Heavy) and Blue Origin (with New Shepard), who are focusing on Vertical Takeoff, Vertical Landing (VTVL). The miniaturization of actuators, sensors, and electronic components has made small satellites more capable [2]. These satellites can replace large, expensive ones and are useful for communication, remote sensing, Earth observation, and espionage [3]. This advancement has created a new market for dedicated small satellite launch vehicles. Small satellites are typically launched into Low Earth Orbit (LEO) through rideshare agreements or piggybacking, but these methods often lack flexibility in terms of launch dates and desired orbits. To address this issue, Rocket Lab developed Electron, a dedicated small satellite launch vehicle designed for the global market. Electron provides customers with schedule control, responsive launch capabilities, and tailored orbits [4]. Initially recovering the Electron's first stage via splashdown, Rocket Lab is now advancing to Mid-Air Recovery (MAR) using air-ram parachutes (parafoil) and helicopters. Reusing the first stage will significantly reduce launch costs and increase launch frequency without requiring additional production.

The initial idea for this thesis was to investigate cost savings and launch cost reduction by comparing two methods of recovery: deploying an unguided circular parachute for splashdown and employing MAR using a helicopter to catch a fully steerable parafoil in the air. However, Peter Beck, the CEO of Rocket Lab, addressed nearly all of these cost-related questions in interviews. In his interview with Madison Reidy, Peter Beck stated that a rocket launch costs $7.5 million, with 70% of the total cost ($5.25 million) attributed to the first stage. Additionally, the Electron rocket's gross margin aim is 50% [5]. In another interview with Jack Beyer and Sawyer R., he explained that initially, there was a belief that rockets should avoid touching water. However, after several splashdowns and recoveries, it became evident that with minor modifications, rockets could be waterproofed and marinized, enabling easy retrieval from water. This significantly simplified operations [6]. He also highlighted that refurbishing costs for the splashdown method are comparable to the expenses in-

volved in mid-air recovery using a helicopter. This cost balance results in neutral overall economics, thereby favoring the adoption of the simpler method.
As a result, the decision was made to omit cost estimation and instead focus solely on the performance analysis of the Electron rocket as part of a feasibility study. This includes developing simulation tools for both ascent and descent trajectories, covering methods of recovery such as unguided circular parachutes and fully steerable parafoils. While the primary focus of this thesis is the Electron rocket, the simulation tool can be adapted for any two-stage rocket by adjusting input parameters such as mass, thrust, geometry, aerodynamic coefficients, etc., to correspond with new data. The tools will be developed with Simulink software and programmed using MATLAB.

For the ascent phase, a two-stage rocket Simulink model is developed that implements quaternion representation of six-degrees-of-freedom (6-DOF) equations of motion for custom variable mass in the Earth-centered Earth-fixed (ECEF) reference frame to simulate ascent trajectories. The model draws inspiration from the Simulink tutorial [7], although it differs significantly in design and scope, offering more depth and detail within this thesis. Additionally, another Simulink model is created to simulate the ballistic trajectory of the first stage after separation using quaternion representation of 6-DOF equations of motion for constant mass in the ECEF reference frame, particularly when first stage recovery is not implemented.
For the descent phase, two distinct Simulink models are developed. One implements Euler angle representation of 6-DOF equations of motion for custom variable mass with an unguided circular parachute to simulate the descent trajectory, while the other simulates the same equations for a steerable (guided) parafoil. The purpose of these models is to address the research objective, as formulated below:

Research Objective: *The research objective of this thesis is to develop a simulation tool for analyzing the launch performance of the Electron rocket and its first-stage recovery, involving mid-air recovery using an air-ram parachute and splashdown recovery using an unguided circular parachute.*

To achieve the research objective, this thesis examines the following research question:

Research Question: *Is the implementation of mid-air and splashdown recovery of the Electron's first stage viable for transforming it into a reusable first stage?*

To answer the research question precisely, the following sub-questions are formulated:

- *What is the additional mass, which is required to transform the Electron's first stage into a reusable one?*
- *What are the comparative advantages and disadvantages of air-ram parachutes (parafoils) versus circular parachutes for the recovery of Electron rocket's first stage?*
- *What impact do wind patterns and atmospheric conditions have on the performance of air-ram parachutes and circular parachutes?*
- *How does the simulated data from the developed tool compare with empirical data gathered from actual Electron rocket launches, particularly in terms of trajectory prediction, parachute performance, and recovery outcomes?*

The structure of the report is as follows: First, Electron's specifications and trajectory outline are presented in Chapter 2. Chapter 3 elaborates on ascent trajectory reference frames, frame transformations, environmental conditions, aerodynamic forces, thrust mechanisms, equations of motion, the Hohmann transfer orbit, ascent trajectory constraints, and the guidance, navigation, and control (GNC) model. Simulink architecture for the ascent trajectory model is detailed in Chapter 4. Ascent trajectory validation is provided in Chapter 5. Results of the ascent phase are presented in Chapter 6. The sensitivity analysis of the ascent phase is presented in Chapter 7. Chapters 8 through 10 focus on the parafoil system. Chapters 11 through 13 discuss the unguided circular parachute, while Chapter 14 offers the conclusion and recommendations.

2
Launch Vehicle Electron

This chapter is divided into multiple sections. Electron specifications, propulsion, and launch sites are presented in Section 2.1. The trajectory outline and recovery of the first stage are discussed in Section 2.2.

2.1. Electron Specifications

Electron is a two-stage (+ kick stage), partially recoverable launch vehicle designed for small satellites (see Figure 2.1). It can deliver a payload of 300 kg to Low Earth Orbit (LEO) or a 200 kg payload to a 500 km Sun-Synchronous Orbit (SSO) [4]. Due to its carbon-composite structure, Electron's overall structural mass is 40 % less compared to traditional aluminum structures. An overview of Electron's specifications is presented in Table 2.1, based on Electron's payload user's guide [4] and information provided by Space Launch Report [8].

Table 2.1: Electron Specifications [4] [8]

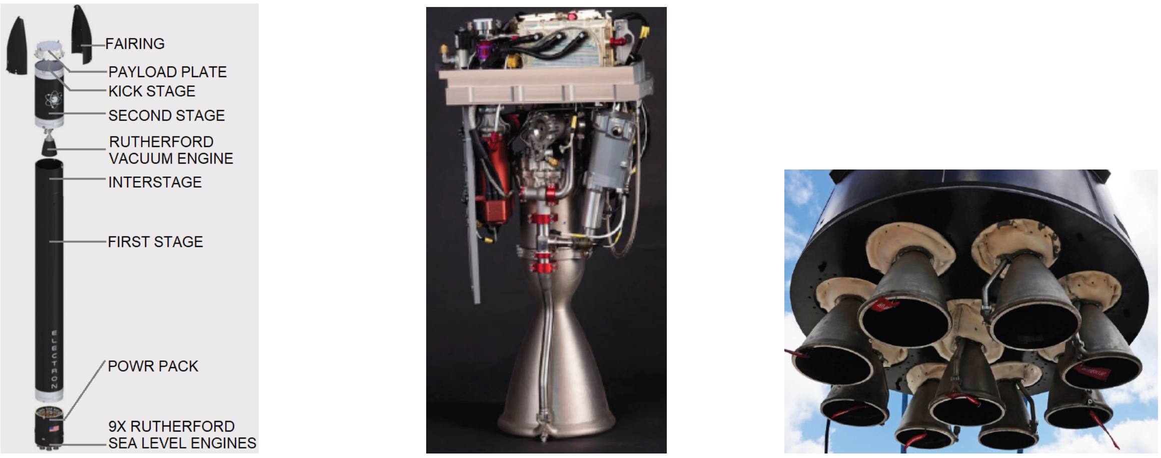
Specification	First Stage	Second Stage	Kick Stage	Overall
Length [m]	12.1	2.4	3.5	18
Diameter [m]	1.2	1.2	1.2	1.2
Structure	Monocoque	Monocoque	Monocoque	Monocoque
Material	Carbon fiber composite	Carbon fiber composite	Carbon fiber composite	Carbon fiber composite
Liftoff mass [kg]	10,200	2,300	500	13,000
Fuel mass [kg]	9,250	2,050	-	11,300
Dry mass [kg]	950	250	-	1,200
Engine	9x Rutherford sea level	1x Rutherford vacuum	1x Curie	-
Single engine mass [kg]	35	35	40	-
Propellant	LOx/RP-1	LOx/RP-1	Liquid bi-propellant	-
Thrust per engine [kN]	24	25.8	0.12	-
Specific impulse [s]	311	343	-	-

2.1.1. Propulsion

For the first stage, the Electron rocket uses nine sea-level Rutherford engines, while the second stage employs a single Rutherford vacuum engine (see Figure 2.1 [4]). The engines in both stages are identical, except that the second stage features a larger expansion ratio nozzle optimized for near-vacuum conditions. Both types of Rutherford engines use liquid oxygen (LOx) and refined kerosene (RP-1) as propellants. The kick stage, powered by the Curie engine, uses a liquid bi-propellant. The thrust and specific impulse of these engines are provided in Table 2.1.

2.1.2. Launch Site

Rocket Lab has two launch sites (see Figure 2.2) [4]. Launch Complex 1, located in Mahia, New Zealand, is the only private orbital launch range in the world. The site is situated at 39.262°S latitude and 177.865°E longitude, supporting launches to inclinations between 39° and 120°. This complex can handle up to 120 launches per year. Launch Complex 2, located on Wallops Island, Virginia, USA, is situated at 37.834°N (latitude) and 75.488°W (longitude), and it supports up to 12 missions per year. This site can accommodate launches to inclinations between 38° and 60°.



(a) Electron's Exploded Views (b) First Stage Rutherford Engine (c) Stage 1 Engine Configuration

Figure 2.1: Electron's Components [4]

(a) Launch Complex 1 in Mahia, New Zealand (b) Launch Complex 2 on Wallops Island, Virginia, USA

Figure 2.2: Rocket Lab's Launch Sites [4]

2.2. Trajectory Outline

This section outlines the trajectory of the Electron rocket, dividing it into the ascent and descent phases. Detailed explanations and designs for these phases will follow in the subsequent chapters of the report.

2.2.1. Ascent Trajectory Outline

For the ascent trajectory, one can choose either a Direct Ascent (DA) trajectory or an indirect ascent trajectory. In a DA trajectory, the rocket is launched from the launch pad on Earth directly to its final destination without entering a preliminary orbit around the Earth. In a DA trajectory, the injection conditions of the desired orbit are identical to the burn-out condition of the final stage. For this method, restart capabilities are not required. The flight time is shorter, which is advantageous for the cryogenic liquid oxygen oxidizer to limit boil-off. In contrast, an indirect ascent trajectory commonly involves placing the rocket into an orbit around the Earth first and then additional maneuvers are performed to reach the desired final orbit. One of the most fuel-efficient indirect ascent trajectories is the Hohmann Transfer Ascent (HTA) trajectory, as depicted in Figure 2.3 [9]. In an HTA trajectory, the spacecraft is first launched into a typical 200 km altitude parking orbit. Then, the spacecraft is injected into the final orbit. In an HTA trajectory, the engines are fired twice. Therefore, restartable engines are needed. The flight time is longer, but this method minimizes the required energy for reaching the desired final orbit. For the above-mentioned reasons, the HTA trajectory is chosen for this thesis.

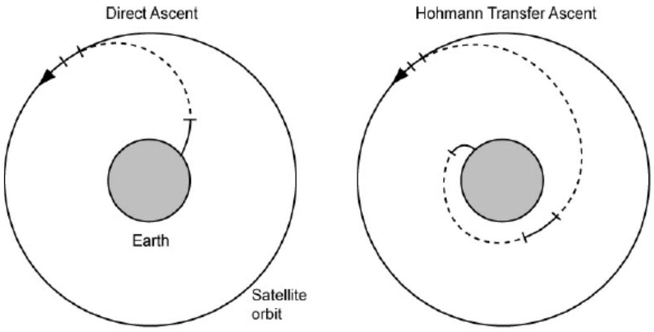


Figure 2.3: Direct Ascent and Hohmann Transfer Ascent Trajectory [9]

Figure 2.4 illustrates the ascent trajectory of the Electron rocket. The launch begins with liftoff from Earth (green path), followed by the second stage ignition (red path) to achieve the initial circular orbit (blue circle), known as the parking orbit. The spacecraft then uses its kick stage Curie engine to enter a Hohmann transfer orbit (yellow path), eventually reaching its final circular orbit (light blue circle) for mission operations. If first stage recovery attitude control is not implemented, the first stage will follow a ballistic trajectory after first stage separation, as illustrated by the green line.

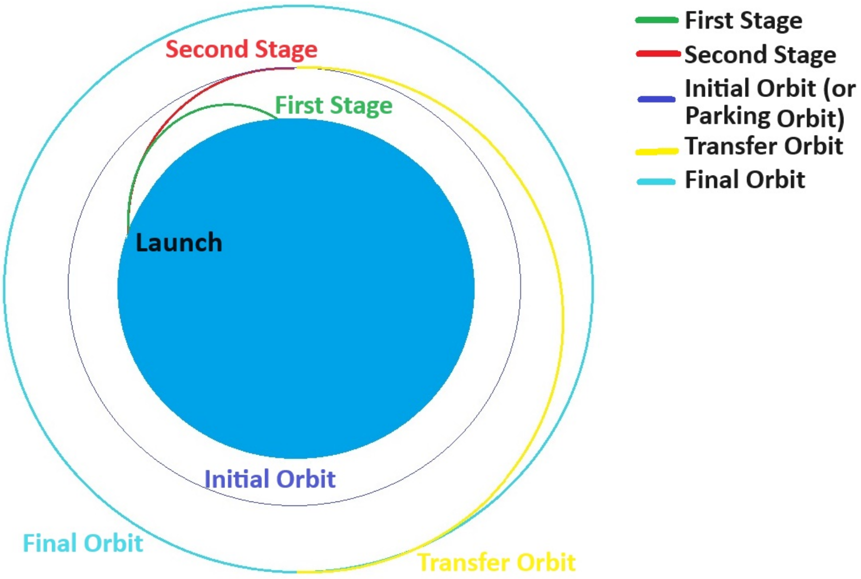


Figure 2.4: Ascent Trajectory Outline Illustration

An example flight profile for a standard dedicated LEO mission is shown in Figure 2.5 [4]. Stage one engine cutoff occurs at $L+155$ seconds, where L represents lift-off. Stage one separation follows at $L+158$ seconds at an altitude of 78 km. At $L+162$ seconds and an altitude of 82 km, the second stage engine ignites. The second stage places the kick stage into a low elliptical orbit. Fairing separation occurs at an altitude of 126 km after $L+184$ seconds. At $L+388$ seconds, the batteries powering the Rutherford engine pumps are jettisoned. The second stage engine cutoff and separation happen at $L+535$ and $L+539$ seconds, respectively. Depending on mission requirements, the kick stage engine (Curie) ignites to circularize the elliptical orbit into the final orbit.

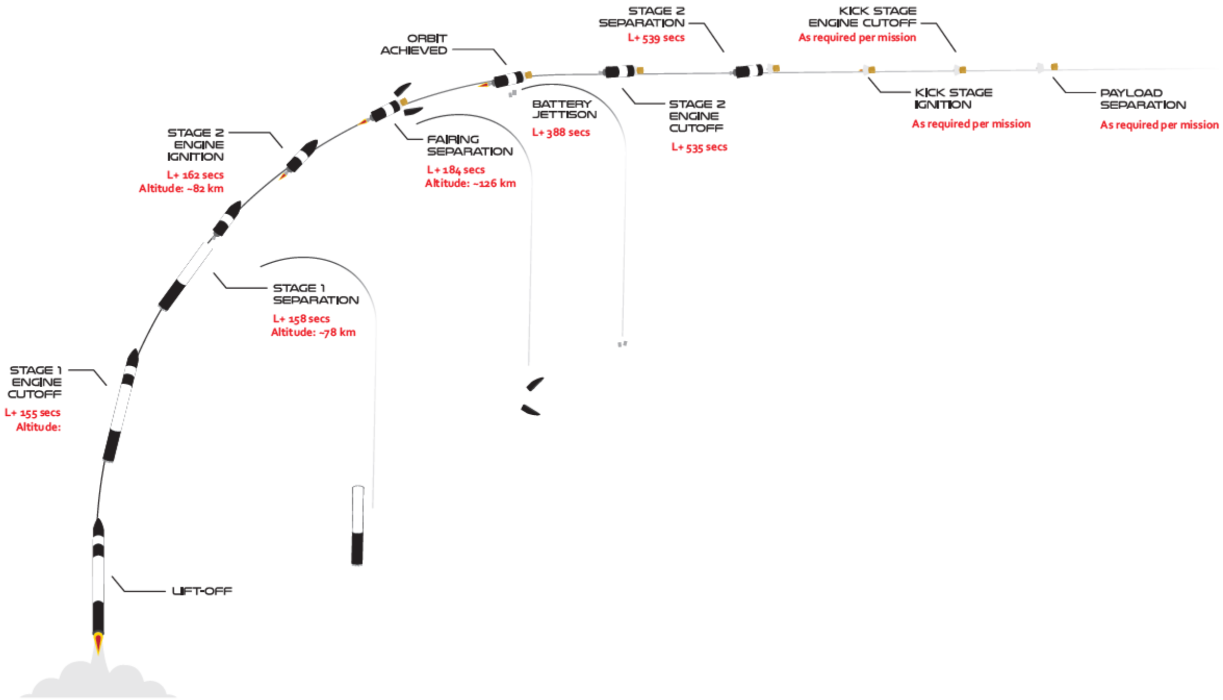


Figure 2.5: Electron Flight Profile Example [4]

2.2.2. Descent Trajectory Outline

After the first stage separation, the second stage will go to the desired orbit and the first stage will reach its apogee. Then, it will start falling back to Earth. Its velocity will increase and it begins to heat up due to the Earth's atmosphere. The reaction control system is activated to reorient the first stage for re-entry through the atmosphere. Keeping the ideal orientation is paramount as the first stage is subjected to very high pressures and temperatures (well above 2,400 °C [10]) and the protective heat shield is at the base of the first stage. To reduce the velocity, Electron will deploy a supersonic decelerator called ballute, which is optimized to generate drag at supersonic velocities and high altitudes. This increases the surface area and rapidly slows down the first stage until it reaches the lower atmosphere and subsonic velocities. For main parachute deployment, Rocket Lab has the option to choose between two methods to recover the first stage: deploying an unguided circular parachute for splashdown, or employing mid-air recovery using a helicopter to catch a fully steerable parafoil in the air.

The first option is straightforward: after deploying the unguided circular parachute, the first stage descends under the parachute and splashes down. A recovery ship promptly arrives to retrieve the first stage and transport it back to port for investigation and refurbishment.

The second method is more complex and relies heavily on pilot skills. A fully steerable parafoil is deployed, and a helicopter equipped with a special grappling hook device takes off to intercept the descending first stage from the rear. As the stage descends with controlled and stable velocity, the grappling hook captures the in-flight drogue line trailing behind the parafoil. The helicopter then gently climbs and accelerates past the parafoil, pulling the slider up and steadily deflating the parafoil until the payload mass transfers from the parafoil to the helicopter [11]. The recovered first stage is then transported to land (or a ship) for inspection and refurbishment in preparation for the next launch. This approach avoids high impact accelerations and prevents engine contamination from harsh saltwater. Additionally, the lifting parafoil provides the ability to glide to a predetermined point, reducing the dispersion associated with descent trajectories and minimizing the landing area.

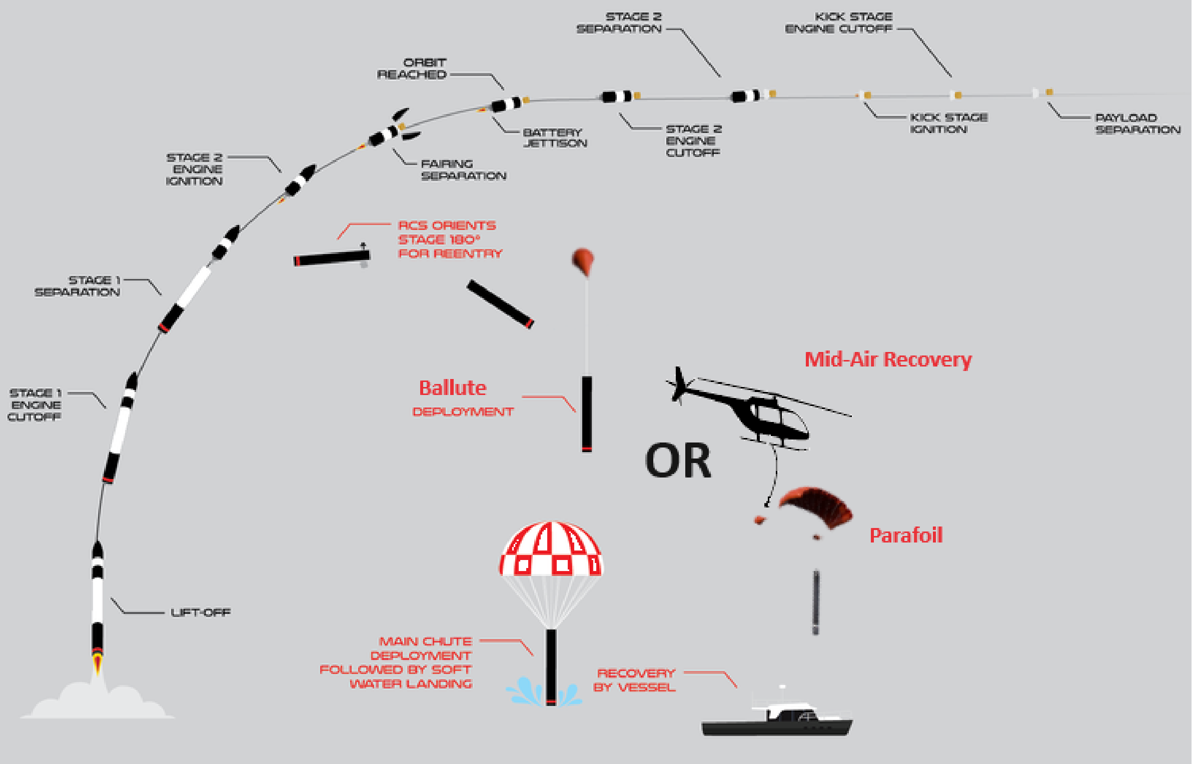


Figure 2.6: Illustration of First Stage Recovery Methods: Mid-Air Recovery with Parafoil and Helicopter, or Unguided Circular Parachute Splashdown (Adapted from Rocket Lab's Press Kit and Mack Crawford's Rendering [1])

The modified Sikorsky S-92 helicopter is used for recovery missions and operated by a crew of three: a pilot, a co-pilot, and a rocket spotter. With a lifting capacity of up to 5000 kg, it features several key modifications for its specialized role. These include larger fuel tanks for extended-range open ocean flights, additional cargo capacity for specialized equipment, a capture hook on a long line to catch the parachute line attached to Electron's first stage, and numerous minor enhancements to improve performance and functionality during missions.

3

Ascending Phase of the Electron Rocket

This chapter focuses on the ascending phase of the Electron rocket. It discusses various aspects including reference frames, state variables, frame transformations, atmospheric and gravity models, aerodynamic forces, thrust mechanisms, fundamental equations of motion, the Hohmann transfer orbit, ascent trajectory constraints, and the Guidance, Navigation, and Control (GNC) model.

3.1. Reference Frames

Reference frames are needed for the derivation of the equations of motion and for describing the position and orientation of the vehicle. Since the vehicle will be orbiting the Earth, Earth-bound reference frames will be used. Multiple reference frames will be needed, as expressing the equations of motion, position, and orientation in some reference frames is easier than in others.

Earth-Centered Inertial Reference Frame

The Earth-centered inertial (ECI) frame is a global reference frame that does not rotate with the Earth, as illustrated in Figure 3.1 [12]. Even though the equinox and equatorial plane experience small movement over time, for most aerospace applications, this reference frame can be assumed inertial. When the epoch J2000 is used to define the equator and equinox, this reference frame can be considered truly inertial for precise orbit determination [13]. The epoch J2000 refers to a specific standardized moment in time, chosen arbitrarily as the starting point corresponding to January 1st, 2000, 12:00 noon UTC.

The origin of this reference frame is at the center of the Earth. The Earth is assumed to be an ellipsoid. The X_{ECI}-axis points towards the vernal equinox also known as the First Point of Aries ♈. The equinoxes are the locations in space where the Earth's equatorial plane intersects with its ecliptic plane. The Y_{ECI}-axis is perpendicular to both the X_{ECI} and Z_{ECI} axes and lies in the equatorial plane, pointing 90 degrees eastward from the X_{ECI}-axis. The Z_{ECI}-axis points toward the north pole along the Earth's rotation axis at the epoch J2000 [14].

Earth-Centered Earth-Fixed Reference Frame

The Earth-Centered Earth-Fixed (ECEF) reference frame is a non-inertial global reference frame that rotates with the Earth at approximately 15° per hour (or 360° per 24 hours). It is used to determine the absolute position of the launch vehicle on the Earth. The origin of this reference frame is at the center of the Earth, and its axes are fixed to the Earth as shown in the Figure 3.1. The Earth is assumed to be an ellipsoid. The X_{ECEF}-axis intersects the Earth's equatorial plane and the Greenwich meridian at 0° latitude and 0° longitude, corresponding to positive values [15]. The Y_{ECEF}-axis is perpendicular to the X_{ECEF}-Z_{ECEF} plane, completing the right-handed system. The intersection of the Earth's surface at 0° latitude and 90° longitude corresponds to positive Y_{ECEF} values. The Z_{ECEF}-axis points toward the north pole along the Earth's rotation axis. The intersection of the Earth's surface at 90° latitude and 0° longitude corresponds to positive Z_{ECEF} values [16].

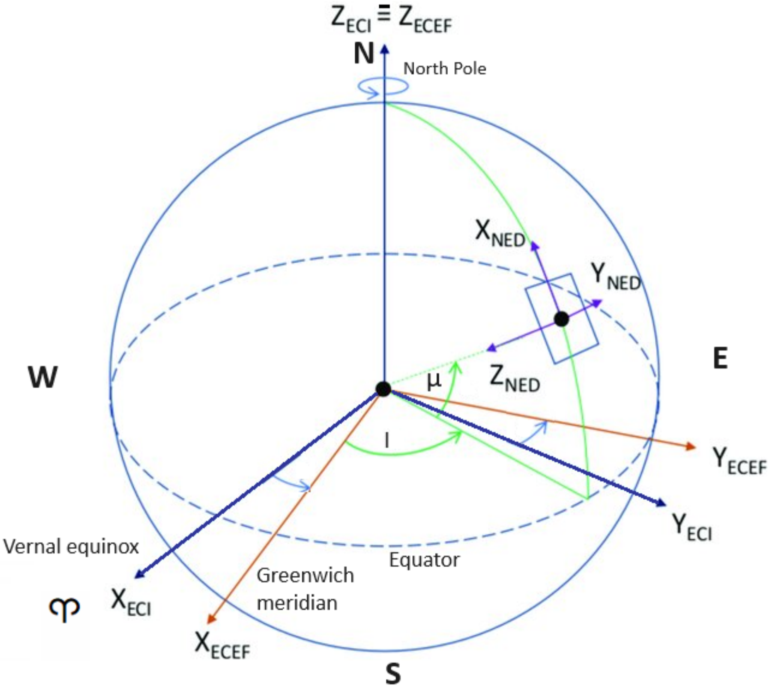


Figure 3.1: Earth-Centered Earth-Fixed (ECEF), Earth-Centered Inertial (ECI) and North-East-Down (NED) Reference Frames (adapted from[12])

North-East-Down Reference Frame

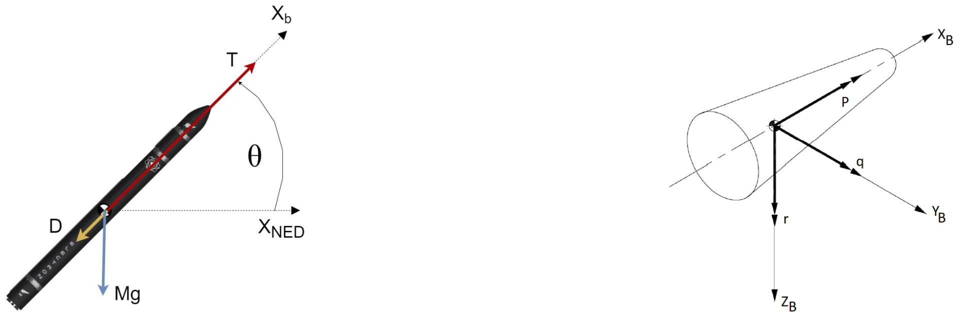
The North-East-Down (NED) frame is a non-inertial, local reference frame used to express velocities and relative positions with respect to a local origin as illustrated in the Figure 3.1. The current position serves as the origin of the reference frame and is located at the center of gravity of the spacecraft. Hence, the origin and axes of the NED frame undergo continuous changes. The positive X_{NED}-axis points toward the geodetic reference ellipsoid's north, running parallel to the latitude line at that location, thus aligning with the true north direction. The positive Y_{NED}-axis points towards the geodetic reference ellipsoid's east, running parallel to the longitude line at that location and perpendicular to the X_{NED}-axis. The positive Z_{NED}-axis points downward towards the Earth's center along the normal of the reference ellipsoid [16].

Body Reference Frame

The body reference frame (index B) is a non-inertial local frame fixed to the moving spacecraft, which is assumed to be rigid as depicted in Figure 3.2a. The origin and orientation of the body reference frame are located at the center of mass of the vehicle. The X_B-axis is positive in a forward direction and lies in the symmetry plane. The Z_B-axis is positive in a downward direction and lies in the plane of symmetry perpendicular to the X_B-Y_B plane. The right-handed system is completed by the Y_B-axis [17].

Wind Reference Frame

The Wind Reference Frame is also a non-inertial local frame with its origin fixed to the moving rigid spacecraft. However, the orientation of the wind reference frame is defined with respect to the velocity vector of the spacecraft. Its x, y, and z axes align with those of the body reference frame (see Figure 3.2b). Its orientation is determined by the Euler angles: Φ (bank angle around the x-axis), γ (flight path angle around the y-axis), and χ (heading angle around the z-axis) [17].



(a) Body Reference Frame (adapted from[7])

(b) The Vehicle's Angular Rate $\omega = (\mathrm{p}, \quad \mathrm{q}, \quad \mathrm{r})^T$ [18]

Figure 3.2: Body Reference Frame and Orientation

Propulsion Reference Frame

The propulsion reference frame can be defined using spherical components. However, thrust force components are generally defined in the body reference frame. Therefore, the body reference frame will be used for defining propulsion.

3.2. State Variables

The state variables describes the position, velocity, attitude and angular rates of the launch vehicle in a dynamic system.

3.2.1. Position And Velocity

The state variables for **position** and **velocity** will be expressed in Cartesian or spherical components. However, since the launch vehicle is orbiting a celestial body (Earth), classical orbital elements will also be utilized.

Orbital Elements

Six parameters are used to define the position and velocity of a spacecraft in an elliptical orbit with respect to an inertial frame, as shown in Figures 3.3 and 3.4 [18].The definition of the semi-major axis α, eccentricity e, true anomaly θ, and the distance of the moving spacecraft r is indicated in Figure 3.3. The size of the elliptical orbit is defined by the semi-major axis, while its shape is defined by the eccentricity. The argument of pericentre ω specifies the orientation of the ellipse in the orbital plane, as shown in Figure 3.4.

The angle longitude of the ascending node Ω specifies the position of the ascending node in the XY-plane. Eastward measurements along the X-axis are defined as positive. To determine the position of the spacecraft in orbit at a given time, it's necessary to know the time of pericentre passage τ. However, this parameter can be replaced by the mean anomaly at the epoch t_0 [18]. The range of these parameters is provided in Table 3.1. The launch vehicle target orbits are defined by these orbital elements; therefore, their definitions are paramount.

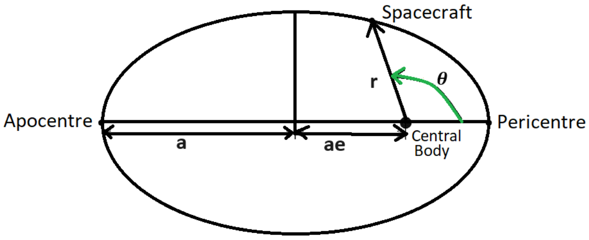


Figure 3.3: Elliptic Orbit [18]

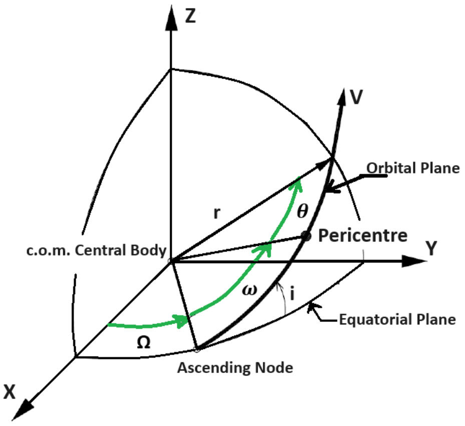


Figure 3.4: The definition of the three orbital parameters Ω, ω, and i [18]

Table 3.1: The Six Orbital Parameters Required for Position and Velocity Definition [18]

Symbol	Parameter	Range
e	Eccentricity	$0 \leq e < 1$
α	Semi-major axis	$\alpha > R_e$
i	Inclination	$0° \leq i \leq 180°$
ω	Argument of pericentre	$0° \leq \omega < 180°$
Ω	Longitude of the ascending node	$0° \leq \Omega < 180°$
M	Mean anomaly	$0° \leq M < 180°$

Cartesian Components

Cartesian components of the state variables are expressed in Cartesian coordinate system. The state variables for **position** and **velocity** expressed in Cartesian components arex, y, z and $\dot{x}, \dot{y}, \dot{z}$, respectively. The definition of the Cartesian position and velocity components with respect to the ECEF frame is shown in the Figure 3.5. When expressing the position and velocity with respect to other reference frames, the same variables are used with different subscript (ECI, b and NED). When expressing the velocity components in the ECEF frame, the notation u, v, and w is commonly used instead of $\dot{x}$, $\dot{y}$, and $\dot{z}$. The advantage of this coordinate system is that it avoids singularities and is frequently utilized in numerical simulations.

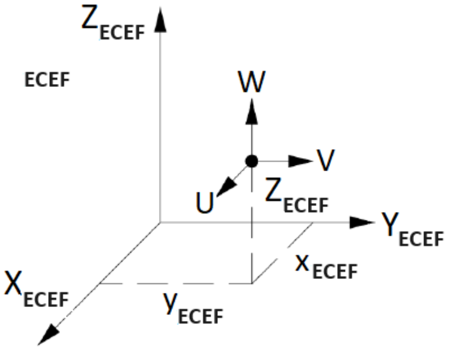


Figure 3.5: Cartesian Coordinate System w.r.t. the ECEF Reference Frame

Spherical Components

The state variables for **position** expressed in spherical components with respect to the ECEF reference frame are ellipsoidal height h, geodetic latitude μ and geodetic longitude l (see Figure 3.1). l is measured positively in the east direction, typically within the range of $[-180° \leq l < 180°]$ or [0°

$\leq l < 360°]$. μ is measured along the appropriate meridian starting at the equator. It is positive in the north and negative in the south direction, with a range of $[-90° \leq \mu \leq 90°]$. The **velocity** of the spacecraft with respect to the ECEF frame is derived in Section 3.7. Coordinate transformation techniques are employed to express the body velocity in various reference frames, including the ECEF frame. For more information on frame transformation, please refer to Section 3.3.

3.2.2. Attitude

The attitude of a vehicle refers to the orientation of a body-fixed reference frame relative to another one. While there are multiple ways to express attitude, this discussion will focus on the three commonly used methods.

- **Classical Attitude Angles**

The classical attitude angles are the roll angle ϕ, the pitch angle θ, and the yaw angle ψ. These angles define the attitude of the body frame with respect to the NED frame. The range of yaw, pitch, and roll angles is as follows: $[-180° \leq \psi < 180°]$, $[-90° \leq \theta \leq 90°]$, and $[-180° \leq \phi < 180°]$ respectively. The frame transformation can be executed in different orders and about different axes, resulting in a total of 12 possible combinations. There are six Type-1 representations, which involve rotations about all three axes, and six Type-2 representations, which involve rotations about only two axes. In aerospace applications, the Type-1 sequence (3-2-1) is commonly used. In this sequence, a yaw rotation about the Z-axis is followed by a pitch rotation about the (local) Y-axis, and finally, a roll rotation about the (local) X-axis [19].

- **Aerodynamic Angles**

Aerodynamic angles are the angle of attack α, the angle of sideslip β and the bank angle σ (see Figure 3.6). These angles form a set of Euler angles, which is a Type-1 sequence with order 2-3-1. The range of α, β and σ are; $[-180° \leq \alpha < 180°]$, $[-90° \leq \beta \leq 90°]$ and $[-180° \leq \sigma < 180°]$ respectively. $\alpha > 0°$ for a 'nose-up' attitude. 'Nose-left' attitude corresponds with a positive β. Banking to the right corresponds with a positive σ.

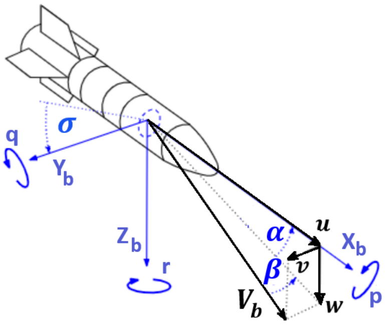


Figure 3.6: Aerodynamic Angles α, β and σ (adapted from [20])

- **Quaternions**

The four Euler parameters (q_0, q_1, q_2 and q_4) or quaternions are a 4-dimensional hyper-complex number, which consist of one real and three imaginary numbers. Quaternions are used to describe rotations in three-dimensional space. The advantage of quaternions is that they avoid singularities (i.e., gimbal lock) for any rotation. For more information about quaternions, readers are referred to the book *Quaternions and Rotation Sequences* [21]. How quaternions are used for coordinate transformation and attitude determination is explained in Section 3.3 and Section 3.7.

3.2.3. Angular Rates

The vehicle's angular rate is defined as the rotational velocity with respect to the body frame. This is defined in components along the body axes as shown in Figures 3.2b and 3.6. ω is the rotation vector, which is defined by the roll rate p, the pitch rate q and the yaw rate r.

3.3. Frame Transformations

This Section explores the use of Simulink utilities for various frame transformations, including Direction Cosine Matrices (DCM), quaternions, rotation angles, and conversions between different reference frames. Each subsection provides detailed explanations of these transformation. The information used for these methods is based on the work of Stevens and Lewis [22], Zipfel [23] and AIAA recommendation [24].

3.3.1. LLA to ECEF Position

The Latitude, Longitude and Altitude (LLA) to Earth-centered Earth-Fixed (ECEF) Position block in Simulink converts geodetic latitude ($\bar{\mu}$), longitude ($\bar{\iota}$), and altitude ($\bar{h}$) above the reference ellipsoid into an ECEF position vector ($\bar{P}$) using Equation 3.1.

$$\bar{p} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} = \begin{bmatrix} r_s \cos\lambda_s \cos\iota + h\cos\mu\cos\iota \\ r_s \cos\lambda_s \sin t + h\cos\mu\sin\iota \\ r_s \sin\lambda_s + h\sin\mu \end{bmatrix} \tag{3.1}$$

Where, λ_s is the geocentric latitude at mean sea level, defined by Equation 3.2, and r_s is the radius at the surface point, defined by Equation 3.3.

$$\lambda_s = \operatorname{atan}\left((1-f)^2 \tan\mu\right) \tag{3.2}$$

$$r_s = \sqrt{\frac{R^2}{1+\left(1/(1-f)^2 - 1\right)\sin^2\lambda_s}} \tag{3.3}$$

In the above equation, R is the equatorial radius, and f represents the flattening (or oblateness) of the Earth, which can be computed using Equation 3.4.

$$f = \frac{a-b}{a} \tag{3.4}$$

where, a represents the semi-major axis and b represents the semi-minor axis.

3.3.2. ECEF Position to LLA

The Earth-centered, Earth-fixed (ECEF) Position to Latitude, Longitude, and Altitude (LLA) block in Simulink converts an ECEF position vector ($\bar{P}$) into geodetic latitude ($\bar{\mu}$), longitude ($\bar{\iota}$), and altitude ($\bar{h}$) above the reference ellipsoid. Longitude can be calculated with Equation 3.5.

$$\iota = \operatorname{atan}\left(\frac{P_y}{P_x}\right) \tag{3.5}$$

Geodetic latitude can be calculated using Bowring's method, which is an iterative technique for converting geocentric coordinates to geodetic coordinates [25].

$$\bar{\mu} = \operatorname{atan}\left(\frac{P_z + \frac{e^2(1-f)}{(1-e^2)} R(\sin\beta)^3}{s - e^2 R(\cos\beta)^3}\right) \tag{3.6}$$

Where:

$$s = \sqrt{P_x^2 + P_y^2} \tag{3.7}$$

$$e^2 = 1 - (1 - f)^2 \tag{3.8}$$

The Simulink block implements the following approach to calculate the geodetic latitude: first, an initial guess for the geodetic latitude is made, followed by another initial guess for the reduced latitude defined by Equation 3.9.

$$\beta = \operatorname{atan}\left(\frac{p_z}{(1-f)s}\right) \tag{3.9}$$

Once the initial guesses are determined, Equation 3.10 is utilized to recalculate the reduced latitude.

$$\beta = \operatorname{atan}\left(\frac{(1-f)\sin\mu}{\cos\mu}\right) \tag{3.10}$$

Using this updated value for the reduced latitude, the geodetic latitude (Equation 3.6) is recalculated. This process is iterated until $\bar{\mu}$ converges, typically achieved after two or three iterations. The altitude can be calculated with Equation 3.11.

$$h = s\cos\mu + \left(P_z + e^2 N \sin\mu\right)\sin\mu - N \tag{3.11}$$

Where, N is the radius of curvature in the vertical prime, as defined by the Equation 3.12.

$$N = \frac{R}{\sqrt{1 - e^2(\sin\mu)^2}} \tag{3.12}$$

3.3.3. Direction Cosine Matrix ECEF to NED

The Direction Cosine Matrix Earth-centered Earth-Fixed (ECEF) to North East Down (NED) block in Simulink converts the geodetic latitude ($\bar{\mu}$)and longitude ($\bar{\iota}$) into a Direction Cosine Matrix. This can be achieved by first rotating the initial coordinates about the z-axis by the longitude, followed by rotating the intermediate coordinates about the y-axis by the geodetic latitude. The combined transformation matrices for the two axes are defined by the Equation 3.13, representing the Direction Cosine Matrix from the ECEF to the NED frame.

$$DCM_{ef} = \begin{bmatrix} -\sin\mu\cos\iota & -\sin\mu\sin\iota & \cos\mu \\ -\sin\iota & \cos\iota & 0 \\ -\cos\mu\cos\iota & -\cos\mu\sin\iota & -\sin\mu \end{bmatrix} \tag{3.13}$$

3.3.4. Direction Cosine Matrix ECEF to NED to Latitude and Longitude

The Direction Cosine Matrix Earth-Centered, Earth-Fixed (ECEF) to North-East-Down (NED) to Latitude and Longitude (LL) block in Simulink converts DCM_{ef} (see Equation 3.13) into geodetic latitude and longitude using Equations 3.14 and 3.15.

$$\mu = \arcsin(-DCM_{ef}(3,3)) = \arcsin(\sin\mu) \tag{3.14}$$

$$\iota = \arctan\left(\frac{-DCM_{ef}(2,1)}{DCM_{ef}(2,2)}\right) = \arctan\left(\frac{\sin\iota}{\cos\iota}\right) \tag{3.15}$$

3.3.5. Direction Cosine Matrix Body to Wind

The Direction Cosine Matrix Body to Wind block in Simulink converts the angle of attack(α) and sideslip angle(β) into a Direction Cosine Matrix. This can be achieved by first rotating the initial coordinates about the Y-axis by the angle of attack, followed by rotating the intermediate coordinates about the Z-axis by the sideslip angle. The combined transformation matrices for the two axes are defined by the Equation 3.16, representing the Direction Cosine Matrix from the Body to the Wind frame.

$$DCM_{wb} = \begin{bmatrix} \cos\alpha\cos\beta & \sin\beta & \sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{3.16}$$

3.3.6. Direction Cosine Matrix Body to Wind to Alpha and Beta

The Direction Cosine Matrix Body to Wind to Alpha and Beta block in Simulink converts DCM_{wb} (see Equation 3.16) into angle of attack and sideslip angle using Equations 3.17 and 3.18.

$$\alpha = \arcsin(-DCM_{wb}(3,1)) = \arcsin(\sin\alpha) \tag{3.17}$$

$$\beta = \arcsin(DCM_{wb}(1,2)) = \arcsin(\sin\beta) \tag{3.18}$$

3.3.7. Direction Cosine Matrix NED to Body

The Direction Cosine Matrix North-East-Down (NED) to Body frame (DCM_{be}) utilizes the Rotation Angles to Direction Cosine Matrix block in Simulink. This block determines DCM_{be} from a given set of rotation angles denoted by R3, R2, and R1, as illustrated in Equations 3.19, 3.20, and 3.21. In Section 3.2.2, it was explained that the rotation sequence follows the ZYX order. The product of three rotation matrices constitutes $\mathrm{DCM_{be}} = \mathrm{R}_\phi \mathrm{R}_\theta \mathrm{R}_\psi$ and is defined by Equation 3.22.

$$\mathrm{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \quad \text{(Rotation about X-axis)} \tag{3.19}$$

$$\mathrm{R}_\theta = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \quad \text{(Rotation about Y-axis)} \tag{3.20}$$

$$\mathrm{R}_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{(Rotation about Z-axis)} \tag{3.21}$$

$$\mathrm{DCM_{be}} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \tag{3.22}$$

The DCM_{be} is an orthogonal matrix, meaning its columns and rows consist of orthonormal unit vectors. For any orthogonal matrix, its inverse is equal to its transpose. Therefore, to compute the DCM from the ECI frame to the body frame (DCM_{bi}), Equation 3.23 can be used.

$$DCM_{bi} = (DCM_{be})^T \times DCM_{be} \tag{3.23}$$

3.3.8. Direction Cosine Matrix to Rotation Angles

The Direction Cosine Matrix to Rotation Angles block in Simulink transforms the $\mathrm{DCM_{be}}$ (see Equation 3.22) into the rotation angles $R1$, $R2$, and $R3$ (see Equations 3.19, 3.20, and 3.21) if the rotation order is known.

3.3.9. Direction Cosine Matrix to Quaternions

The Direction Cosine Matrix to Quaternions block in Simulink converts a DCM into a four-element unit quaternion vector (q_0, q_1, q_2, q_3). Simulink uses the scalar-first convention, meaning the real part (scalar) of the quaternion is listed first, followed by the imaginary part (vector). The DCM_{be} (see Equation 3.22) defined by a unit quaternion vector is represented in Equation 3.24. Using this equation, quaternions and rotation angles can be calculated.

$$\mathrm{DCM_{be}} = \begin{bmatrix} \left(q_0^2+q_1^2-q_2^2-q_3^2\right) & 2\left(q_1q_2+q_0q_3\right) & 2\left(q_1q_3-q_0q_2\right) \\ 2\left(q_1q_2-q_0q_3\right) & \left(q_0^2-q_1^2+q_2^2-q_3^2\right) & 2\left(q_2q_3+q_0q_1\right) \\ 2\left(q_1q_3+q_0q_2\right) & 2\left(q_2q_3-q_0q_1\right) & \left(q_0^2-q_1^2-q_2^2+q_3^2\right) \end{bmatrix} \tag{3.24}$$

3.3.10. Quaternions to Rotation Angles

The Direction Cosine Matrix to Rotation Angles block in Simulink converts the DCM_{be} (see Equation 3.24) into the rotation angles R3, R2, and R1, respectively. This is valid for the ZYX rotation order.

$$\phi = \mathrm{atan2}(\mathrm{DCM_{be}}(2,3), \mathrm{DCM_{be}}(3,3)) = \mathrm{atan2}\left(2\left(q_2q_3+q_0q_1\right), \left(q_0^2-q_1^2-q_2^2+q_3^2\right)\right) \tag{3.25}$$

$$\theta = \mathrm{asin}(-DCM_{be}(1,3)) = \mathrm{asin}\left(-2\left(q_1q_3-q_0q_2\right)\right) \tag{3.26}$$

$$\psi = \mathrm{atan2}(DCM_{be}(1,2), DCM_{be}(1,1)) = \mathrm{atan2}\left(2\left(q_1q_2+q_0q_3\right), \left(q_0^2+q_1^2-q_2^2-q_3^2\right)\right) \tag{3.27}$$

Where, ϕ is R3, θ is R2, and ψ is R1. These are the same rotation angles as described by Equations 3.19, 3.20, and 3.21 respectively.

3.3.11. Rotation Angles to Quaternions

The Rotation Angles to Quaternions block in Simulink converts the three rotation angles R_3, R_2, and R_1 described by Equations 3.19, 3.20, and 3.21 into a four-element unit quaternion vector (q_0, q_1, q_2, q_3). This conversion is performed by first calculating the half-angles of R_1, R_2, and R_3, and then multiplying them by sine and cosine. The quaternions are subsequently determined using Equations 3.28, 3.29, 3.30, and 3.31.

$$q_0 = \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) + \sin\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right) \tag{3.28}$$

$$q_1 = \cos\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right) - \sin\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) \tag{3.29}$$

$$q_2 = \cos\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) + \sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right) \tag{3.30}$$

$$q_3 = \sin\left(\frac{\psi}{2}\right)\cos\left(\frac{\theta}{2}\right)\cos\left(\frac{\phi}{2}\right) - \cos\left(\frac{\psi}{2}\right)\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{\phi}{2}\right) \tag{3.31}$$

3.3.12. Quaternions to Direction Cosine Matrix

The Quaternions to Direction Cosine Matrix block in Simulink converts the four-element unit quaternion vector (q_0, q_1, q_2, q_3) into a direction cosine matrix, as described by Equation 3.24.

3.4. Environment

To accurately replicate the physical world, it's essential to define the environment in which the simulation occurs. Simulink offers various models for the atmosphere, wind and gravity. Brief descriptions of these models are provided in the following subsections. During the ascent phase, wind effects are not considered.

3.4.1. Atmospheric Models

For atmospheric models, a distinction can be made between a *reference* atmosphere and a *standard atmosphere*. The standard atmospheric model is defined by a hypothetical vertical distribution of atmospheric properties such as pressure, temperature, and density. It represents the conditions of an annual average at mid-latitude. Therefore, it does not take into account temporal and spatial influences except for altitude. Thus, the results of the numerical simulation would be the same for different epochs and launch locations. The *reference atmosphere* does include temporal and spatial influences as it takes into account seasonal, geomagnetic, latitudinal, and solar effects.

NRLMSISE-00 Atmosphere Model

NRLMSISE-00 is an empirical *reference atmospheric model* that implements the mathematical representation of the 2001 United States Naval Research Laboratory Mass Spectrometer and Incoherent Scatter Radar Exosphere (NRLMSISE-00) [26]. This is the most accurate model available in Simulink. It provides values for atmospheric properties (temperatures and densities) from the surface of the Earth up to the lower exosphere (0 to 1,000 km).The NRLMSISE-00 model is most accurate as it takes into account the actual observations (measured data). Therefore, the NRLMSISE-00 provides different values for different locations on the Earth, which makes it ideal to investigate the advantages or disadvantages of different launch locations on the Electron's performance. However, this level of accuracy requires a lot of computational power. Furthermore, this model requires a lot of input, including year and day, time of day, geodetic altitude, latitude and longitude, local apparent solar time, 81-day average of F10.7 solar flux, daily F10.7 solar flux for the previous day, and daily magnetic index. Additionally, it only provides two outputs: temperature and density, which encompass the densities of Helium, Oxygen, Nitrogen, Argon, and Hydrogen, as well as the total mass density. While other available models also provide pressure and speed of sound, this model will not be used.

COESA 1976 Atmosphere Model

The COESA 1976 Atmosphere Model was developed by the United States Committee on Extension to the Standard Atmosphere (COESA) in 1976. For this reason, the COESA Atmosphere Model 1976 is also known as the United States Standard Atmosphere 1976 (US76). It is a representation of an idealized steady-state of the Earth's atmosphere [27]. The model utilizes mathematical algorithms to represent the lower atmospheric conditions specified by the US76. It takes geopotential altitude as input and provides values for absolute temperature, pressure, density, and speed of sound as outputs. This model provides values for the atmospheric properties from the surface of the Earth up to an altitude of 1000 km at 45° North latitude. Below an altitude of 32 km, the data of COESA 1976 is identical to the standard atmosphere of the International Civil Aviation Organization (ICAO). Above the geopotential altitude of 84,852 m, temperature values are linearly extrapolated, while pressure values are extrapolated logarithmically. Speed of sound and density are computed using a perfect gas relationship. In the COESA 1976 model, the atmosphere below 86 km is divided into 8 layers. The defined reference levels and temperature gradients for the different linearly segmented layers from the surface of the Earth to 86 geometric km are provided in Table 3.2 [27]. Using these values, the temperature below 86 km can be determined by Equations 3.32 and 3.33.

$$T_M = T_{M_i} + L_{z_i}(z - z_i) \tag{3.32}$$

$$T = T_M \frac{M}{M_0} \tag{3.33}$$

In the above equations, T is the kinetic temperature, M is the molecular mass, T_M is the molecular scale temperature, and M_0 is the molecular mass of air at sea level. The molecular mass ($M = M_0 =$ 28.9644 kg/kmole) is assumed to be constant below 80 km. Therefore, $T = T_M$. The relation between

Table 3.2: Defined Reference Levels and Temperature Gradients (adapted from the U.S. Standard Atmosphere 1976 [27])

Subscript i	z_i [km]	L_{z_i} [K/km]	p_i [N/m^2]
0	0	-6.5	101,325
1	11	0	22,632
2	20	+1	5,475
3	32	+2.8	868
4	47	0	111
5	51	-2.8	79
6	71	-2.0	4
7	84.85	-6.5	4

the geopotential altitude (z) and the geometric altitude (h) is given by the Equation 3.34, where R^* (8314.32 J/kmole K) is the universal gas constant, g_0 (9.80665 m/s^2) is the Earth's gravity at sea level, and R_0 (6356.766 km) is the Earth's radius at the latitude, which corresponds with g_0.

$$z = \int_0^h \frac{g}{g_0} dh \approx \frac{R_0 h}{R_0 + h} \tag{3.34}$$

The pressure below 86 km can be determined using Equation 3.35 when $L_{z_i} = 0$ and Equation 3.36 when $L_{z_i} \neq 0$. L_{z_i} is the molecular scale temperature gradient, and it is constant per interval. K_i is defined by Equation 3.37. The index i represents the i^{th} interval.

$$p = p_i \exp\left[\frac{-g_0 M_0 (z - z_i)}{R^* T_{M_i}}\right] \tag{3.35}$$

$$p = p_i \left[\frac{T_{M_i}}{T_{M_i} + L_{z_i}(z - z_i)}\right]^{K_i} \tag{3.36}$$

$$K_i = \frac{g_0 M_0}{R^* \cdot L_{z_i}} \tag{3.37}$$

Above the geometric altitude of 86 km, T_M values are not defined, nor is geopotential the primary argument anymore. Instead, the temperature-altitude profile is defined in terms of four successive functions as follows:

- An isothermal layer for 86 km ≤ h < 91 km, where $T = 186.8673K$ = constant.
- A layer where T(z) has an elliptical form for 91 km ≤ h < 110 km and can be determined by Equation 3.38.

$$T = T_c + A\sqrt{1 - \left(\frac{z - z_8}{b}\right)^2} \tag{3.38}$$

  Where $T_c = 263.1905K$, $A = -76.3232K$, $b = 19.9429km$ and $z_8 = 89.72km$.

- A layer with constant and positive gradient for 110 km ≤ h < 120 km, which can be computed with Equation 3.39.

$$T = T_9 + L_9 (z - z_9) \tag{3.39}$$

  Where $T_9 = 240K$, $L_9 = 12K/km$ and $z_9 = 108.13km$.

- The fourth temperature-altitude profile function has an exponential form from 120 km to 1000 km and can be calculated using Equation 3.40.

$$T = T_\infty - (T_\infty - T_{10}) \cdot \exp(-\lambda\xi) \tag{3.40}$$

The temperature gradient can be computed with Equation 3.41.

$$\frac{dT}{dz} = L_{z_i} = \lambda \cdot (T_\infty - T_{10}) \cdot \left(\frac{R_0 + z_{10}}{R_0 + z}\right)^2 \cdot \exp(-\lambda \cdot \xi) \tag{3.41}$$

Where,

$$\begin{aligned} \lambda &= L_{K,9} / (T_\infty - T_{10}) = 0.01875 \\ \xi &= \xi(z) = (z - z_{10}) \cdot (R_0 + Z_{10}) / (R_0 + z) \end{aligned} \tag{3.42}$$

The atmospheric pressure between 86 and 1000 geometric km is determined with Equation 3.43.

$$p = \frac{NR^* T}{N_A} \tag{3.43}$$

Where, N_A is the Avogadro's constant ($6.0225 \cdot 10^{26} kmol^{-1}$) and N is the total number of density (m^{-3}). N is determined based on linear interpolation and its values are provided in the US76 report [27]. The last parameter needed is the atmospheric density, which can be computed with Equation 3.44.

$$\frac{\rho}{\rho_0} = e^{-\frac{h}{H_s}} \tag{3.44}$$

Where, h is the altitude, ρ_0 is the atmospheric density at sea level ($1225 kg/m^3$) and H_s is the scale height and its value is in the range of 7,050 - 7,200 m [28]. The speed of sound α in an ideal gas can be calculated with the following equation:

$$\alpha = \sqrt{\gamma \frac{RT}{M}} \tag{3.45}$$

where, γ is the adiabatic index, R is the universal gas constant, T is the absolute temperature in Kelvin and M is the molar mass of the gas in kilograms per mole. This model has been validated and is available in Simulink. A tabulated version of this model has been validated and used by Miranda [29], Rozemeijer [30], Vandamme [31], Kesteren [32], Contant [33], and other students at TU Delft.

3.4.2. Gravity Models

Gravity is one of the main forces that influence the launch vehicle during the ascent and descent phases of the launch. Multiple gravity models are available in Simulink, but only two models will be addressed in this section: central gravity (point mass gravity) and spherical harmonics gravity model.

Central Gravity

Using Newton's law of universal gravitation and the fact that the mass of the launch vehicle is negligible compared to the mass of the Earth results in the central gravity model as described by Equation 3.46.

$$F_G = -G\frac{M_E}{r^3}\mathbf{r} \tag{3.46}$$

Where, F_G is the gravity force, M_E is the mass of the Earth ($5.9722 \cdot 10^{24}$ kg), G is the universal gravitational constant ($6.674 \cdot 10^{-11} m^3 \cdot kg^{-1} \cdot s^{-2}$) and **r** is distance between the launch vehicle and the center of the Earth. This model assumes a perfect sphere and a homogeneous mass distribution for the Earth. This model was used first until all the bugs of the Simulink model were resolved and the models were working fine; then the switch was made to a more accurate gravity model namely Spherical Harmonic Gravity Model.

Spherical Harmonics Gravity

In reality, the Earth is neither a perfect sphere nor does it have a homogeneous mass distribution. Near the Earth, the assumption of central gravity does not hold. The inhomogeneous mass distribution, represented by spherical harmonic expansion, needs to be taken into account. This can be achieved by implementing the mathematical representation of the spherical harmonics of Earth's gravity based on Earth's gravitational potential, as described by Equation 3.47.

$$U = -\frac{\mu}{r}\left[1+\sum_{n=2}^{\infty}\sum_{m=0}^{n}\left(\frac{R}{r}\right)^{n} P_{n,m}(sin(\Phi)\left\{C_{n,m}cos(m\Lambda)+S_{n,m}sin(m\Lambda)\right\}\right] \tag{3.47}$$

Where r, Φ, and Λ are the spherical coordinates in the geocentric rotating reference frame. μ is the Earth's gravitational parameter, r is the distance from the center of mass, R is the mean equatorial radius of the Earth, and $P_{n,m}(\sin\Phi)$ are associated Legendre functions of the first kind of degree n and order m. Furthermore, Φ is the geocentric latitude, Λ is the geocentric longitude, and $C_{n,m}$ and $S_{n,m}$ are constant model parameters [34]. Data for $C_{n,m}$ and $S_{n,m}$ are obtained from other satellites in orbit, and their values are provided in the *ITG-Grace2010 gravity-field model* [35]. This model can distinguish between different mass distributions as represented in Figure 3.7 [9].

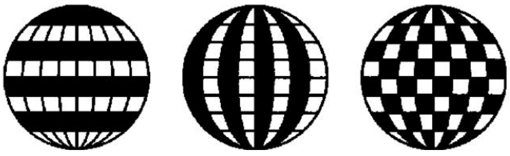


Figure 3.7: Zonal Harmonics ($n \neq 0$ m = 0) Sectorial Harmonics (m = n ≠0) Tesseral Harmonics (m ≠ n ≠ 0; right) [9]

In Figure 3.7, elevation above and below the mean spherical surface is represented by white and black areas, respectively. It should be noted that this model does not take into account the centrifugal effects of planetary rotation and the influence of a precessing reference frame. For the simulations, it is usually sufficient to include only a few terms for n (such as $n = 2$) [28] as the J_2 term is the dominant zonal term. According to D. Y. Hsu, the difference between the spherical harmonic gravity model with the J_2 term and the central gravity model is only $5\mu g$ [36]. For this reason, Vandamme [31], Kesteren [32], and Contant [33] used the central gravity model in their theses. However, it is not known how this will affect the desired end accuracy in terms of velocity and position. Since the Spherical Harmonic Model is better suited for high-precision and detailed studies, this model is chosen for the final simulation.

This model is validated and available in Simulink. It takes ECEF coordinates as input and provides an array of gravity values in the x, y, and z-axis of the ECEF coordinates. For the central body, Earth Gravitational Model 2008 (EGM2008) has been selected, which represents the latest spherical harmonic gravitational model for Earth from the National Geospatial-Intelligence Agency (NGA) [37]. The default degree of the harmonic model, specified as a scalar, is 120, implying that the model includes terms up to the 120th zonal harmonic coefficient (J_{120}). The gravity vector must be transformed from the ECEF frame to the body frame. This process involves two steps: first, transforming the gravity vector from the ECEF frame to the NED frame, and then transforming it from the NED frame to the body frame, as illustrated by the Equation 3.48.

$$g_b = DCM_{\text{be}}\cdot DCM_{\text{ef}}\cdot g_f \tag{3.48}$$

Where g_b is the gravity vector in the body frame, DCM_{be} is the Direction Cosine Matrix from the ECEF frame to the body frame, DCM_{ef} is the Direction Cosine Matrix from the ECEF frame to the NED frame, and g_f is the gravity vector in the ECEF frame.

3.5. Aerodynamic Model

During the ascent trajectory, the vehicle will be flying through the Earth's atmosphere. As a result of this motion between the air and the vehicle, aerodynamic forces will be developed, which have a significant influence on the vehicle's trajectory. The magnitude of aerodynamic forces depends on the vehicle's velocity, orientation, and shape, as well as atmospheric properties such as temperature, pressure, and density. To compute aerodynamic forces and moments, the *Aerodynamic Forces and Moments* block in Simulink is utilized (see Figure 3.8). This block requires input parameters such as aerodynamic coefficients in the wind reference frame, dynamic pressure, center of gravity, center of pressure and velocity. Output of the block is the aerodynamic forces and moments in the body frame.

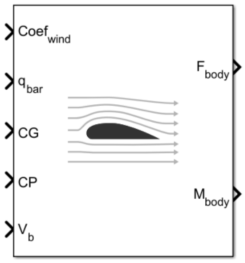


Figure 3.8: Simulink Block of Aerodynamic Forces and Moments

Aerodynamic Forces

Since the input axis of the block is in the wind reference frame, a Direction Cosine Matrix from wind to body (DCM_{bw}) is required to transform the aerodynamic forces in the wind reference frame into the body reference frame. The Direction Cosine Matrix from body to wind (DCM_{wb}) is already provided in Section 3.3.5. Since DCM_{wb} is an orthogonal matrix, which means its columns and rows consist of orthonormal unit vectors. For any orthogonal matrix, its inverse is equal to its transpose. Therefore:

$$DCM_{bw} = DCM_{wb}^{-1} = DCM_{wb}^{T} \tag{3.49}$$

The aerodynamic forces in the body reference frame ($\mathbf{F}_A^b$) can be divided into three components: lift (L), drag (D), and side force (S), and can be computed with Equation 3.50. These forces are defined with respect to the body reference frame and are positive in the negative axis, as shown in Figure 3.2 [18]. The term $\frac{1}{2}\rho V_b^2$ is, in fact, the definition of *dynamic pressure*.

$$\mathbf{F}_A^b = \begin{pmatrix} -D \\ -S \\ -L \end{pmatrix} = DCM_{wb}^T \cdot \overbrace{\begin{pmatrix} -C_D \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \\ -C_S \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \\ -C_L \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \end{pmatrix}}^{\mathbf{F}_A^w} \tag{3.50}$$

In the above equation, C_D, C_S, and C_L are the drag, side force, and lift coefficients, respectively. $\mathbf{F}_A^w$ is the aerodynamic forces in wind frame, ρ is the atmospheric density (kg/m^3), V_b is the velocity of the body (m/s), and S_{ref} is the aerodynamic reference area, which is the cross-sectional area of the vehicle's body (m^2). C_D, C_S, and C_L are functions of the Mach number (M) and aerodynamic angles such as the angle of attack (α) and the angle of sideslip (β) (see Figure 3.6). M is defined by Equation 3.51, where a is the local speed of sound and can be calculated with Equation 3.45.

$$M = \frac{V_b}{\alpha} \tag{3.51}$$

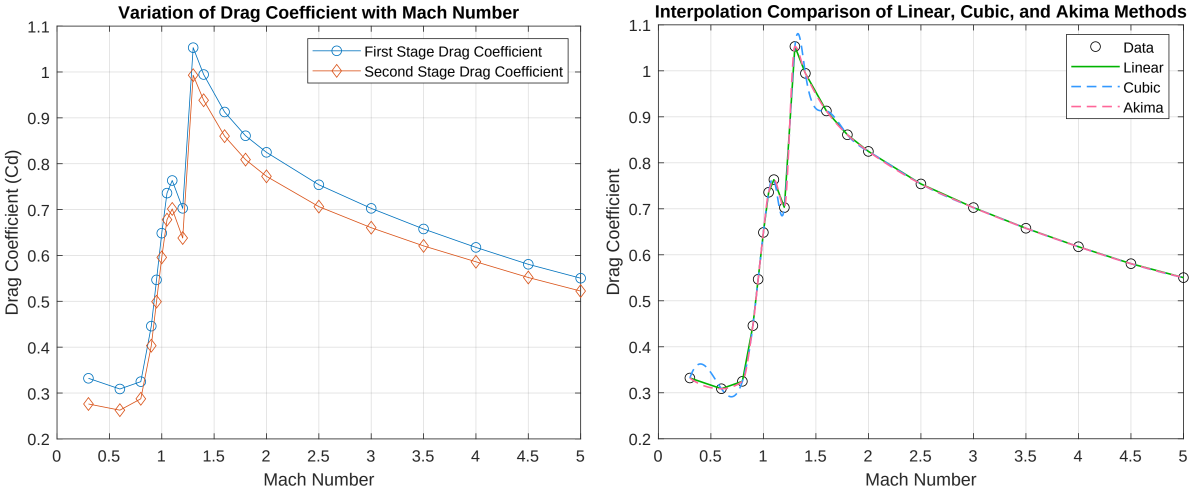
Aerodynamic Moments

The aerodynamic moments ($M_{A_M,b}$) due to aerodynamic forces can be divided into three components: the rolling moment around the x-axis (L), the pitching moment around the y-axis (M), and the yawing moment around the z-axis (N). These moments can be computed using Equation 3.52. They are defined with respect to the body frame.

$$M_{A_M,b} = \begin{pmatrix} M_{X,b} \\ M_{Y,b} \\ M_{Z,b} \end{pmatrix} = \begin{pmatrix} L \\ M \\ N \end{pmatrix} = DCM_{wb}^{T} \cdot \overbrace{\begin{pmatrix} C_l \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \cdot b_{ref} \\ C_m \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \cdot c_{ref} \\ C_n \cdot \frac{1}{2} \cdot \rho \cdot V_b^2 \cdot S_{ref} \cdot b_{ref} \end{pmatrix}}^{\mathbf{M_{A_M},w}} \tag{3.52}$$

Where, C_l, C_m, and C_n are the rolling moment, pitching moment, and yawing moment coefficients, respectively. $\mathbf{M_{A_M},w}$ is the aerodynamic moment in the wind frame. b_{ref} and c_{ref} are the aerodynamic reference lengths (m), which are equal to the height of the vehicle and the maximum diameter of the vehicle, respectively. The moment due to gravity arises because the center of gravity does not coincide with the center of mass. However, these differences are significantly small compared to thrust and aerodynamic moments. Therefore, gravitational moments are neglected. The moments due to the difference between the center of gravity and the center of pressure are also neglected. Furthermore, it is assumed that the vehicle is symmetric, and therefore only the above-mentioned moments (L, M, and N) are taken into account in the Simulink model.

To obtain the aerodynamic coefficients, **Missile DATCOM** software is used. The Missile DATCOM program can be implemented directly as part of the simulation. However, direct implementation significantly increases the computational load, as discovered by Castellini [38] and Vandamme [31]. To circumvent this drawback, Vandamme constructed a database of several launch vehicle configurations based on the length, diameter, and shape of the nosecone of the launch vehicle. Stijn Hoefsloot [39] provided the database, which has been used to create a lookup table with Akima spline interpolation in Simulink for this thesis. The drag coefficient at zero angle of attack for both stages of the Electron launch vehicle is presented in Figure 3.9a. After creating the lookup table, a decision needed to be made on which interpolation method to use: Linear, Cubic, or Akima. According to Ozdemir [40], the Akima method offers the best results due to its lowest interpolation error. Although the data presented in Figure 3.9a includes numerous data points, which leads to good performance across all interpolation methods, the Akima method stands out when tested for the first stage drag coefficient interpolation. Its superior performance is evident from the smooth graph shown in Figure 3.9b. The Akima spline method is particularly well-suited for data sets with abrupt changes. This is why the Akima spline interpolation method was chosen.



(a) Drag Coefficients of the Electron Launch Vehicle for Both Stages

(b) Interpolation Comparison of Linear, Cubic, and Akima Method

Figure 3.9: Drag Coefficients and Interpolation Methods for the Electron Launch Vehicle

3.6. Thrust Model

The thrust is provided by the propulsion system of the vehicle. The thrust (T) consists of momentum and pressure components, as defined by Equation 3.53 [18] [9] [41].

$$T = \overbrace{\dot{m} \cdot U_e}^{\text{momentum comp.}} + \overbrace{A_e \cdot (p_e - p_a)}^{\text{pressure comp.}} \tag{3.53}$$

Where, $\dot{m}$ is the mass flow rate (kg/s), U_e is the exhaust velocity (m/s), A_e is the nozzle exit area (m^2), p_e is the nozzle exit pressure (Pa), and p_a is the ambient pressure (Pa). Since the ambient pressure is zero in a vacuum, the thrust in a vacuum (T_{vac}) is defined by Equation 3.54. The specific impulse is defined by Equation 3.55, and the effective exhaust velocity (C_{eff}) is defined by Equation 3.56.

$$T_{vac} = \overbrace{\dot{m} \cdot U_e}^{\text{momentum comp.}} + \overbrace{A_e \cdot p_e}^{\text{pressure comp.}} \tag{3.54}$$

$$I_{\text{sp}} = \frac{T}{\dot{m} \cdot g_0} = \frac{C_{\text{eff}}}{g_0} \tag{3.55}$$

$$C_{\text{eff}} = U_e + \frac{p_e - p_a}{\dot{m}} A_e \tag{3.56}$$

If the propulsion force does not act through the vehicle's center of mass, a propulsion moment is introduced. However, it is assumed that the propulsion force acts through the vehicle's center of mass, thus negating the introduction of any propulsion moment. The thrust force in the body reference frame is defined by Equation 3.57.

$$F_{T,b} = \begin{pmatrix} T \\ 0 \\ 0 \end{pmatrix} \tag{3.57}$$

Since the thrust at liftoff and maximum thrust of the launch vehicle are known (see Table 2.1), a better approach in this case would be to use throttling to increase or decrease the produced thrust to achieve the desired velocity with respect to the reference velocity at each epoch. The reference velocity and throttling control are explained in detail in Section 3.10. In essence, if the vehicle's velocity falls below the reference velocity, additional thrust is applied for correction. This adjustment is facilitated by a Saturation block, which serves to cap the output thrust. By setting a minimum thrust value of zero and an upper limit equal to the maximum thrust capacity of the launch vehicle, the Saturation block effectively prevents both excessive and negative thrust outputs. The corresponding Simulink architecture is presented in Chapter 4. With the thrust known at each time step, the mass flow rate can be determined using Equation 3.58.

$$\dot{m} = \frac{T}{g_0 \cdot I_{sp}} \tag{3.58}$$

The propellant mass is known. When the mass flow rate reaches zero, it signifies the complete combustion of all propellant. The corresponding time is referred to as the burnout time. A MATLAB function is written to continuously check when the mass flow rate becomes zero, and the corresponding time is saved as the burnout time. After the main engine cutoff, first stage separation takes place. Based on past missions, this coasting phase lasts approximately 5 seconds. Following this, the second-stage engine is ignited. Similar to the first stage, thrust control via throttling is applied. Upon complete combustion of the second-stage propellant, the second-stage engine cutoff occurs. Using another MATLAB function, the corresponding burnout time of the second stage is calculated. Based on these burnout times and coasting time, the Stage Selector Block determines when to utilize the thrust of the first stage and the second stage, along with their corresponding masses. Both MATLAB functions used for calculating the burnout times of the first and second stages are provided in Appendix B (see B.1 and B.2).

The achievable delta-v ($\Delta v_{\text{achievable}}$) of a launch vehicle represents the actual change in velocity, accounting for losses due to gravity and atmospheric drag. This value is lower than the ideal delta-v, which does not consider these losses. For simplification, losses due to steering and engine misalignment are also neglected. To compute the cumulative delta-achievable, the net acceleration is integrated over the burn time of both stages, accounting for thrust and losses due to gravity and drag as defined by Equation 3.59.

$$\Delta V_{\text{achievable}} = \overbrace{\int_0^{t_b} \left(\frac{\vec{T}(t)}{m(t)} \right) dt}^{\Delta V_{\text{ideal}}} - \overbrace{\int_0^{t_b} \vec{g}(t) \cdot \vec{v}(t)\, dt}^{\Delta V_{\text{gravity}}} - \overbrace{\int_0^{t_b} \left(\frac{\vec{D}(t)}{m(t)} \right) dt}^{\Delta V_{\text{drag}}} \tag{3.59}$$

Where, $\vec{v}(t)$ is the unit vector in the direction of the velocity at time t, $m(t)$ represents the instantaneous mass of the vehicle and t_b denotes the burnout time of the second stage. The dot product accounts for the directionality of gravity's influence on the trajectory. Integration begins at liftoff (t_0) and concludes at the cutoff of the second stage engine. During the coasting phase, there is no thrust, but the effects of gravity and drag are considered. The total delta-v loss is the sum of gravity and drag losses. The integral formulation accurately captures the delta-v, not only due to its consideration of delta-v losses, but also because it accounts for the variability of mass, thrust, gravity and drag. This contrasts with simpler models that assume constant values for these factors.

The value of $\Delta V_{\text{achievable}}$ can also be obtained by integrating the accelerations from Equation 3.68. Since $\Delta V_{\text{achievable}}$, ΔV_{ideal}, and ΔV_{drag} must be calculated in any case (see Figure 4.11), there is no need to compute $\Delta V_{\text{gravity}}$ using the dot product and unit vector. Instead, $\Delta V_{\text{gravity}}$ can be directly determined using Equation 3.60.

$$\Delta V_{\text{gravity}} = \Delta V_{\text{ideal}} - \Delta V_{\text{drag}} - \Delta V_{\text{achievable}} \tag{3.60}$$

3.7. Equation Of Motion

The equations of motion are based on Newton's Laws of Motion. The motion of the vehicle consists of two components: translational motion and rotational motion. The first component describes the translational motion of the center of mass (c.o.m.), which provides information about position and velocity in all three directions, representing three degrees of freedom. The second component describes rotation about its c.o.m., which provides information about the angular rates around all three axes, also accounting for three degrees of freedom. Since the angular rates are defined in components along the body axes, this information also contributes to understanding the attitude of the body. Incorporating these three translational and three rotational motions forms the six degrees of freedom (6-DOF).

The equations of motion can be expressed in different reference frames, as described in the book *Rocket Propulsion and Spaceflight Dynamics* [9] and in the report *The Motion of a Vehicle in a Planetary Atmosphere* [18]. However, in this section, only the formulation of the quaternion representation of the 6-DOF equations of motion for a custom variable mass in the ECEF reference frame will be presented.

The rigid body assumption is applied for the Electron rocket. This means the elastic behavior is not modeled, thus eliminating the need to take into account the forces acting between the individual elements of the mass. It is assumed the applied forces act at the body's center of gravity. The rotation of the Earth is taken into account. However, the Earth's polar motion, nutation and precession are neglected. To derive the equations of motion an arbitrary vehicle with variable mass (m) is assumed, which is subjected to external forces while moving about a celestial body as shown in Figure 3.1. The corresponding **translational equation of motion** is described by the Equation 3.61 [42].

$$\begin{aligned}\bar{F_b} = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} &= m\left(\dot{\bar{V_b}} + \bar{\omega}_b \times \bar{V_b} + DCM_{bf}\bar{\omega}_e \times \bar{V_b} + DCM_{bf}\left(\bar{\omega}_e \times \left(\bar{\omega}_e \times \bar{X_f}\right)\right)\right) \\ &+ \dot{m}\left(\bar{V}re_b + DCM_{bf}\left(\bar{\omega}_e \times \bar{X_f}\right)\right)\end{aligned} \tag{3.61}$$

Where:
$\left[F_x\ F_y\ F_z\right]^T$ are the applied external forces given in the body reference frame. $\bar{V}re_b$ is the relative velocity in the wind reference frame, where the mass flow ($\dot{m}$) is expelled from the body in the body reference frame. DCM_{bf} is the Direction Cosine Matrix for coordinate transformation from the ECEF reference frame to the NED reference frame. $\bar{V_b}$ is the velocity of the body relative to the ECEF reference frame, expressed in the body reference frame. It is defined by the Equation 3.62.

$$\bar{V_b} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3.62}$$

$\bar{\omega}_{\text{rel}}$ represents the body's relative angular rates with respect to the NED reference frame, expressed in the body reference frame. It is described by the Equation 3.63.

$$\bar{\omega}_{rel} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.63}$$

$\bar{\omega}_e$ is the rotation rate of the Earth and it is expressed by the Equation 3.64.

$$\bar{\omega}_e = \begin{bmatrix} 0 \\ 0 \\ \omega_e \end{bmatrix} \tag{3.64}$$

$\bar{X_f}$ is the position in the ECEF reference frame, and its time derivative ($\dot{\bar{X_f}}$) represents the change in position in the same ECEF reference frame. This can be calculated using Equation 3.65.

$$\dot{\bar{x}}_f = DCM_{fb}\bar{V_b} \tag{3.65}$$

$\bar{\omega}_b$ is the body's angular rates with respect to the ECI reference frame, expressed in the body reference frame.

$$\bar{\omega}_b = \bar{\omega}_{rel} + DCM_{bf}\bar{\omega}_e + DCM_{be}\bar{\omega}_{NED} \tag{3.66}$$

In the above equation DCM_{be} stands for the Direction Cosine Matrix for the coordinate transformation from the NED reference frame to the body reference frame. The angular rate of the NED reference frame with respect to the ECEF reference frame, expressed in the NED reference frame ($\bar{\omega}_{\text{NED}}$) is defined by the Equation 3.67.

$$\bar{\omega}_{\text{NED}} = \begin{bmatrix} \dot{i}\cos\mu \\ -\dot{\mu} \\ -\dot{i}\sin\mu \end{bmatrix} = \begin{bmatrix} V_E/(N+h) \\ -V_N/(M+h) \\ -V_E \bullet \tan\mu/(N+h) \end{bmatrix} \tag{3.67}$$

In the above equation, μ represents the geodetic latitude and l represents the geodetic longitude. The derivatives of the geodetic latitude $\dot{\mu}$ and geodetic longitude $\dot{l}$ have also been expressed in terms of NED velocity components, where V_E represents the eastward velocity component and V_N represents the northward velocity component, respectively. h is the height above the ellipsoid. N and M are the radii of curvature in the prime vertical and meridian, respectively.

The accelerations of the body relative to the ECEF reference frame, expressed in the body reference frame, are provided by the Equation 3.68.

$$A_{bb} = \begin{bmatrix} \dot{u}_b \\ \dot{v}_b \\ \dot{w}_b \end{bmatrix} = \frac{\bar{F}_b - \dot{m}\left(\bar{V}_{re_b} + DCM_{bf}\left(\bar{\omega}_e \times \bar{X}_f\right)\right)}{m} \tag{3.68}$$
$$- \left[\bar{\omega}_b \times \bar{V}_b + DCM_{bf}\bar{\omega}_e \times \bar{V}_b + DCM_{bf}\left(\bar{\omega}_e \times \left(\bar{\omega}_e \times \bar{X}_f\right)\right)\right]$$

The accelerations of the body relative to the ECEF reference frame, expressed in the ECEF reference frame, are provided by the Equation 3.69.

$$A_{b_{\text{ECEF}}} = \frac{\bar{F}_b - \dot{m}\left(\bar{V}_{re_b} + DCM_{bf}\left(\bar{\omega}_e \times \bar{X}_f\right)\right)}{m} \tag{3.69}$$

The **equation of rotational motion** of the body, defined in the body reference frame, is described by the Equation 3.70 [42].

$$\bar{M}_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = \bar{I}\dot{\bar{\omega}}_b + \bar{\omega}_b \times \left(\bar{I}\bar{\omega}_b\right) + \dot{I}\bar{\omega}_b \tag{3.70}$$

Where:
$[L \;\; M \;\; N]^T$ represents the applied external moments given in the body reference frame with respect to the origin O within the same reference frame. The inertia tensor (I) with respect to the origin O can be calculated with the Equation 3.71.

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{3.71}$$

The inertia tensor's rate of change is defined by the Equation 3.72.

$$\dot{I} = \begin{bmatrix} \dot{I}_{xx} & -\dot{I}_{xy} & -\dot{I}_{xz} \\ -\dot{I}_{yx} & \dot{I}_{yy} & -\dot{I}_{yz} \\ -\dot{I}_{zx} & -\dot{I}_{zy} & \dot{I}_{zz} \end{bmatrix} \tag{3.72}$$

The rate of change of the quaternion vector, which is used for integration, is defined by Equation 3.73. The elements of the matrix are constructed using the components of the angular rate of the body $(\bar{\omega}_b)$.

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = -1/2 \begin{bmatrix} 0 & \omega_b(1) & \omega_b(2) & \omega_b(3) \\ -\omega_b(1) & 0 & -\omega_b(3) & \omega_b(2) \\ -\omega_b(2) & \omega_b(3) & 0 & -\omega_b(1) \\ -\omega_b(3) & -\omega_b(2) & \omega_b(1) & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.73}$$

3.8. Hohmann Transfer Orbit

The Hohmann transfer is defined by two coplanar maneuvers, which transfer the spacecraft between two orbits within the same plane as depicted in Figure 3.10. The shape of the initial and desired final orbits can be either elliptical or circular. In this thesis, a circular orbit is assumed for both the initial and final orbits. The elliptical transfer orbit is tangential to both the initial and final orbits and its semimajor axis is calculated by the Equation 3.74.

$$a_T = \frac{(a_1 + a_2)}{2} \tag{3.74}$$

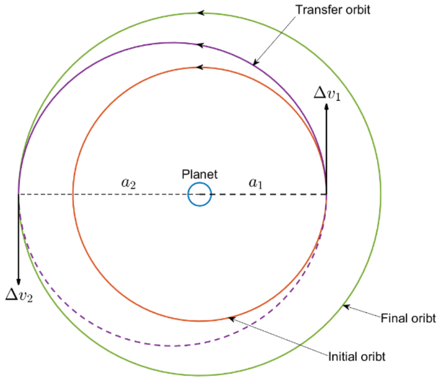


Figure 3.10: The Hohmann Transfer Orbit Depiction [43]

Where, a_1 and a_2 are the radii of the initial and final circular orbits, respectively. This transfer orbit is created by the first maneuver, where its perigee has the same altitude as the initial circular orbit, and its apogee has the same altitude as the final orbit. The second maneuver takes place at the tangent point of the final orbit (apogee), which circularizes the transfer orbit. Without the second maneuver, the spacecraft would follow the trajectory of the dashed line in Figure 3.10. The velocity magnitude of the initial circular orbit is defined by the Equation 3.75

$$v_1 = \sqrt{\frac{\mu}{a_1}} \tag{3.75}$$

The velocity magnitude of the final circular orbit is defined by the Equation 3.76

$$v_2 = \sqrt{\frac{\mu}{a_2}} \tag{3.76}$$

where, μ is the standard gravitational parameter of the Earth. The velocity magnitude of the transfer orbit can be derived from the so-called Vis-viva equation:

$$\frac{v_{T,1}^2}{2} - \frac{\mu}{a_1} = -\frac{\mu}{2a_T} \tag{3.77}$$

The velocity magnitude of the transfer orbit at the tangent point of the initial orbit (first transfer point) can be calculated using the Equation 3.78.

$$v_{T,1} = \sqrt{2\mu\left(\frac{1}{a_1} - \frac{1}{2a_T}\right)} \tag{3.78}$$

The velocity magnitude of the transfer orbit at the tangent point of the final orbit (second transfer point) can be calculated using the Equation 3.79.

$$v_{T,2} = \sqrt{2\mu\left(\frac{1}{a_2} - \frac{1}{2a_T}\right)} \tag{3.79}$$

The corresponding delta-v at the first transfer point, which is the difference between the initial circular orbit and the transfer orbit at the same transfer point, can be obtained using Equation 3.80.

$$\Delta v_1 = v_{T,1} - v_1 \tag{3.80}$$

The delta-v at the second transfer point can be obtained using the Equation 3.81.

$$\Delta v_2 = v_2 - v_{T,2} \tag{3.81}$$

The transfer duration, which is the coasting time between the first and the second transfer points, can be calculated by taking half of the transfer orbit's orbital period, as given by the Equation 3.82.

$$t_T = \pi\sqrt{\frac{a_T^3}{\mu}} \tag{3.82}$$

In order to visualize the initial orbit in Simulink, it is assumed that the spacecraft stays for one full orbital period on the initial orbit and then starts the transfer orbit. The orbital period of the initial orbit at the first transfer point can be obtained using the Equation 3.83.

$$t_1 = 2\pi\sqrt{\frac{a_1^3}{\mu}} \tag{3.83}$$

The corresponding time to the second transfer point can be obtained using the Equation 3.84.

$$t_2 = t_1 + t_T \tag{3.84}$$

The simulation end upon completing one orbital period on the desired final orbit. The corresponding time is determined by the Equation 3.85.

$$t_f = t_2 + 2\pi\sqrt{\frac{a_2^3}{\mu}} \tag{3.85}$$

Both maneuvers are assumed to be impulsive maneuvers and are in the same direction as the orbital motion. Therefore, the change in velocity is assumed to occur instantaneously. This assumption can be achieved by assuming a high thrust acceleration in Simulink, which reduces the burn duration of the propellant. The amount of used propellant will remain the same; only the burn time is reduced. The thrust can be calculated using the Equation 3.86.

$$T = \dot{m} g_0 I_{sp} \tag{3.86}$$

Where, $\dot{m}$ represents the mass flow rate, g_0 denotes Earth's gravitational acceleration at sea level, and I_{sp} is the specific impulse. To calculate the burn duration for each transfer point, one needs the initial and final masses of the spacecraft after each maneuver, which can be determined using the rocket equation. For the first maneuver, this is obtained using the Equation 3.87.

$$m_1 = \frac{m_0}{e^{\left(\frac{\Delta v_1}{v_e}\right)}} \tag{3.87}$$

Where, m_0 and m_1 represent the initial and final mass of the spacecraft after the first maneuver, respectively. v_e denotes the exhaust velocity, which can be obtained using Equation 3.88.

$$v_e = g_0 I_{sp} \tag{3.88}$$

The burn duration of the first maneuver is obtained using the Equation 3.89

$$b_1 = \frac{m_0 - m_1}{\dot{m}} \tag{3.89}$$

The final mass of the spacecraft after the second maneuver is determined using the Equation 3.90.

$$m_2 = \frac{m_1}{e^{\left(\frac{\Delta v_2}{v_e}\right)}} \tag{3.90}$$

The burn duration of the second maneuver is calculated using the Equation 3.91.

$$b_2 = \frac{m_1 - m_2}{\dot{m}} \tag{3.91}$$

So far, maneuver times, burn durations, and delta-v magnitudes have been established. To characterize circular velocity in three-dimensional space, one need to determine the direction of delta-v, which is provided by the Equation 3.92

$$\begin{aligned}\hat{\mathbf{v}} = &(-\cos\Omega\sin(\omega+\theta_*) - \sin\Omega\cos i\cos(\omega+\theta_*))\,\hat{x}\\ &+(-\sin\Omega\sin(\omega+\theta_*) + (\cos\Omega\cos i\cos(\omega+\theta_*)))\,\hat{y}\\ &+\sin i\cos(\omega+\theta_*)\,\hat{z}\end{aligned} \tag{3.92}$$

Where, $\hat{\mathbf{v}}$ is the Δv direction, Ω is the right ascension of ascending node (RAAN), θ_* is the true anomaly, ω is the argument of perigee and i is the inclination. These orbital parameters are provided in the International Celestial Reference Frame (ICRF), which is characterized by the axes spanned by the unit vectors $\hat{x}, \hat{y}$, and $\hat{z}$ as shown in the Figure 3.11. It should be noted that MATLAB's ECI frame aligns with the ICRF.

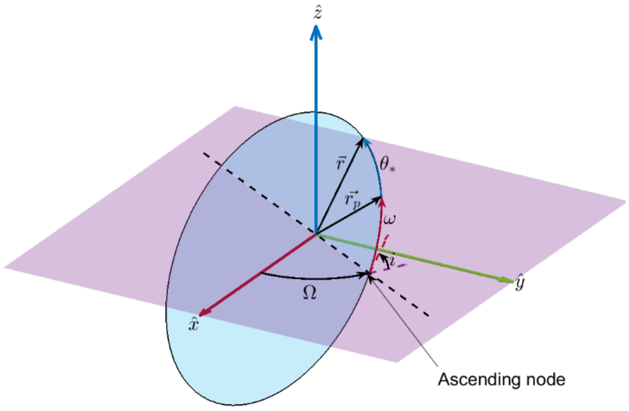


Figure 3.11: International Celestial Reference Frame [43]

The true anomaly of the transfer orbit is 0 degrees at the first transfer point and 180 degrees at the second transfer point. Since the transfer orbit is coplanar with the initial and final orbits, its inclination and RAAN are the same as that of the initial and final orbits. Furthermore, the transfer orbit has the same argument of perigee as the initial orbit. With the equations and data provided, it is now possible to accurately simulate the trajectory of the launch vehicle.

Fortunately, there is already a validated Simulink model for the Hohmann transfer orbit available in MathWorks. Therefore, designing or validating another Simulink model for this purpose is unnecessary. This model will be used to raise the orbit from the initial (parking) orbit to the final orbit.

3.9. Ascent Trajectory Constraints

In order to have a realistic trajectory, certain trajectory constraints must be met. The first constrains is the maximum dynamic pressure (q_{max}), which can be calculated with Equation 3.93. This constraint is related to the the structural integrity of the launch vehicle and keeping the dynamic pressure below this constraint is paramount.

$$q = \frac{1}{2}\rho V^2 \tag{3.93}$$

The q_{max} value varies significantly for different launchers. For example, it is 27.0 kPa for Falcon 9, 57.5 kPa for Pegasus, 57.0 kPa for Ariane V and 90 kPa for PSLV C19 [32] [33]. However, maximum dynamic pressure of the Electron rocket is not know. Therefore, it must be calculated based on past mission. This is addressed in Section 5.1. As a result, the upper limit for the maximum dynamic pressure is constrained to 35 kPa.

The second constraint is the axial acceleration (α_x), which can be calculated with the Equation 3.94. The launch vehicle will be subjected to a range of axial accelerations during flight. This constraint is implemented in the Simulink model for the ascent trajectory. The upper limit for the axial acceleration is constrained at 7.5g per Electron's payload user's guide [4].

$$a_x = \frac{T(t) - D(t)}{m(t) g_0} \leq a_{x_{\max}} \tag{3.94}$$

Another important constraint for the ascent trajectory is the heat flux ($\dot{Q}$), which is generated by the interaction between the atmosphere and the launch vehicle, resulting from convection and radiation. The heat flux can be calculated using the Equation 3.95.

$$\dot{Q} = \frac{1}{2} \rho v^3 \tag{3.95}$$

The payload is protected from this heat flux by the fairing. Therefore, no constraints are applied while the fairing is attached. After the fairing is jettisoned, constraints on the heat flux must be considered [44]. According to the Electron payload user's guide, at the time of fairing deployment, a standard mission will experience free molecular heating of approximately 1135 W/m^2 [4]. Therefore, the heat flux constraint will be set at 1135 W/m^2. The free-stream enthalpy convective model, known for its accuracy, is widely used and has also been employed by Kesteren [32] and Contant [33].

3.10. Guidance, Navigation and Control (GNC) model

For the ascent trajectory, the angle of attack is assumed to be 0 degrees. The relationship between the angle of attack (α), flight path angle (γ), and pitch angle (θ) is given by Equation 3.96.

$$\theta = \alpha + \gamma \tag{3.96}$$

Assuming a zero angle of attack simplifies this relationship, making the flight path angle equal to the pitch angle. This simplification makes it easier to estimate the aerodynamic coefficients using the software Missile DATCOM (see Section 3.5). Additionally, this simplification makes the guidance and control systems easier to design and implement, as the vehicle's orientation directly aligns with its velocity vector. While numerous methods exist for guidance and control, this thesis adopts a straightforward approach, implementing parallel open-loop altitude control and closed-loop velocity control to shape the ascent trajectory.

Typically, ascent trajectory guidance and control for rockets involve two stages: first, generating a simplified 3 DOF optimized reference trajectory. It's optimized for factors like payload delivery accuracy, fuel efficiency, and stress minimization. In the second stage, a more complex 6 DOF guidance system is designed, which uses the simplified 3 DOF trajectory as a reference, for precise control during ascent. In this thesis, rather than employing an optimized 3 DOF trajectory, past mission data is utilized as the reference trajectory. This dataset comprises altitude and velocity at various time steps. However, due to the unavailability of the pitch profile from past missions, the pitch angle is manually adjusted until the desired altitude trajectory is achieved. Unfortunately, the constrained timeline prevents the design of an optimized reference trajectory generation model. Consequently, the trajectory is not optimized, leaving room for improvement in future work. The closed-loop velocity control is depicted in Figure 3.12. The source of the images in Figures 3.12 and 3.13 is [45]. In Figure 3.12, past mission velocity data is used as reference velocity. Section 5.1

explains past missions in detail. The two-degree-of-freedom (2-DOF) Proportional Integral Derivative (PID) controller block generates an output signal (referred to as *G* in Figure 3.12) based on the difference between the reference velocity and the measured simulated velocity of the rocket. The equation for the 2-DOF PID controller in its ideal form is presented by Equation 3.97.

$$u = P\left[(br - y) + I\frac{1}{s}(r - y) + D\frac{N}{1 + N\frac{1}{s}}(cr - y)\right] \tag{3.97}$$

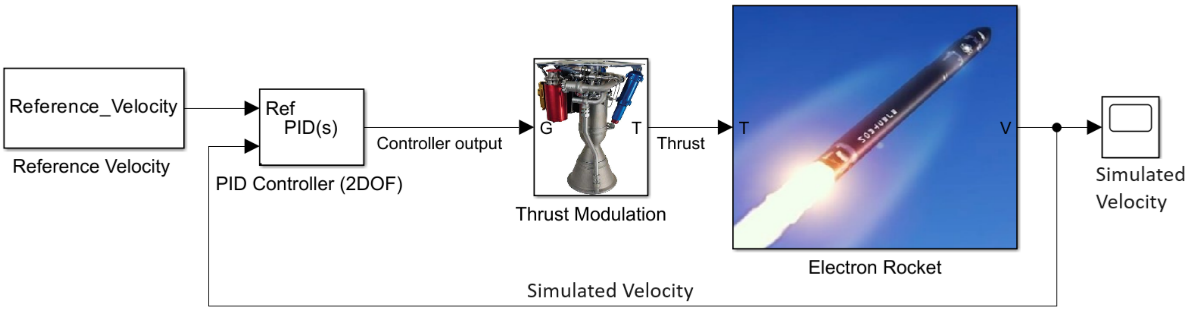


Figure 3.12: Schematic of Closed-Loop Velocity Control System

Where, u is the controller output, b and c are the setpoint weights, N is the filter coefficient, r is the reference signal and y is the measured output signal (simulated Velocity). The output signal G is the aggregate of the proportional (P), integral (I), and derivative (D) actions based on the difference between reference velocity and the simulated velocity, with each action being adjusted according to the weighted gain parameters P, I, and D. The default setting is utilized, and the parameters are $P = 250$, $I = 10$, $D = 100$, $N = 100$, and $b = c = 1$. P acts on the sum of all actions. The signal G is used as a form of gain to modulate the thrust of the rocket engine by increasing or decreasing it, directly influencing the rocket's simulated velocity. The system ensures that the simulated velocity matches the reference velocity as closely as possible. The open-loop altitude control system cannot directly influence the altitude. Therefore, in parallel with the closed-loop velocity control (see Figure 3.12), a closed-loop pitch angle control is used to shape the altitude trajectory, as depicted in Figure 3.13.

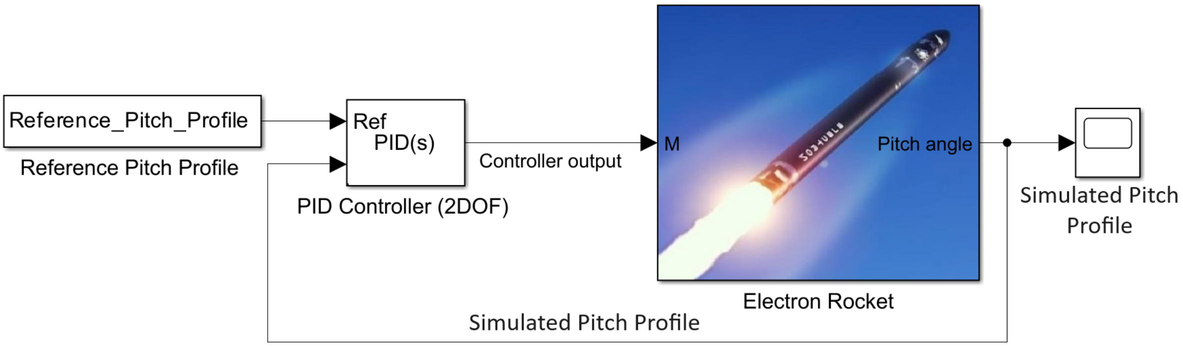


Figure 3.13: Schematic of Closed-Loop Pitch Angle Control System

The closed-loop pitch angle control operates similarly to the closed-loop velocity control. A 2-DOF PID controller block is implemented to generate an output signal (referred to as M) based on the difference between the reference pitch profile and the simulated pitch profile. Signal M is utilized as applied moments to alter the pitch angle and consequently, the trajectory's shape. As previously mentioned, due to the unavailability of pitch profiles from past missions, the pitch angle is manually adjusted until the desired trajectory is attained. The Simulink block diagram of the control system is presented in Chapter 4.

4

Simulink Architecture for the Ascent Trajectory Model

This chapter presents the Simulink architecture for the ascent trajectory model. The model comprises many subsystems, each of which will be presented in detail. The corresponding MATLAB code for each subsystem is provided in Appendix B.

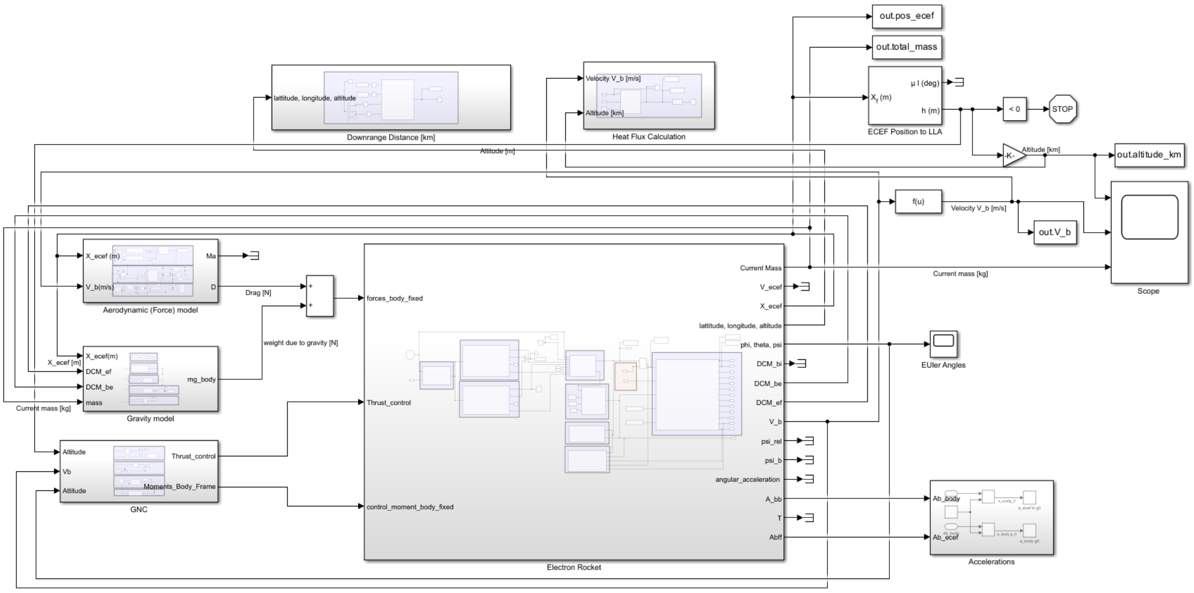


Figure 4.1: Simulink Ascent Trajectory Model Overview

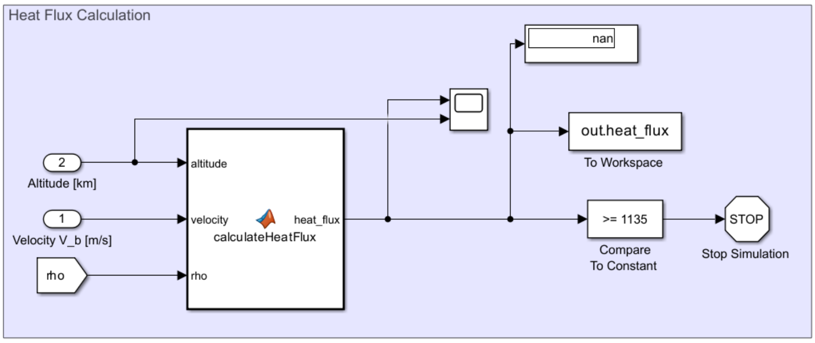


Figure 4.2: Heat Flux Calculation

The subsystem for heat flux calculation is shown in Figure 4.2. It takes altitude, velocity, and density as inputs and calculates the heat flux. If the heat flux constraint is violated, the simulation will be stopped.

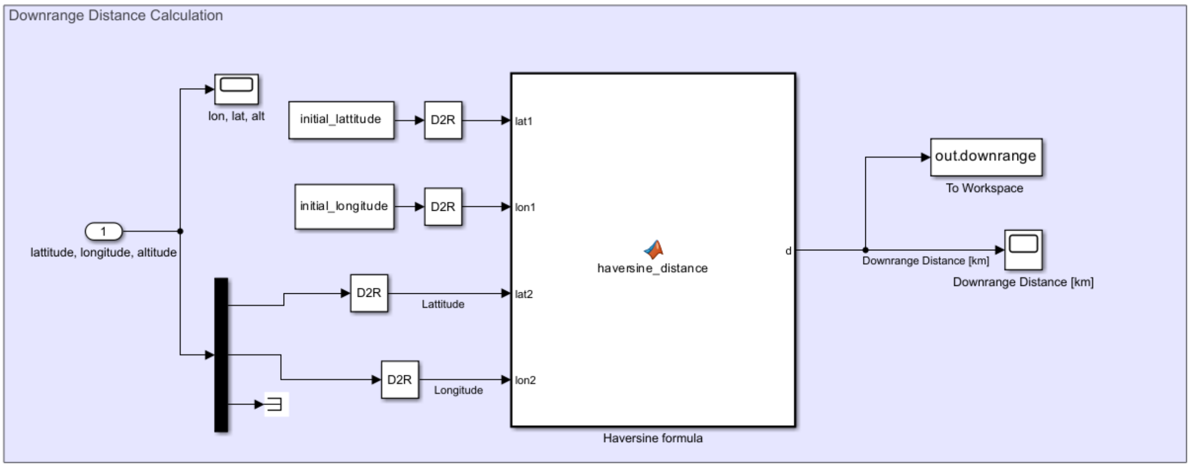


Figure 4.3: Downrange Distance Calculation

The subsystem for downrange distance calculation is illustrated in Figure 4.3. This subsystem uses the haversine formula to compute the great-circle distance between two points on a sphere, based on their initial and final latitudes and longitudes.

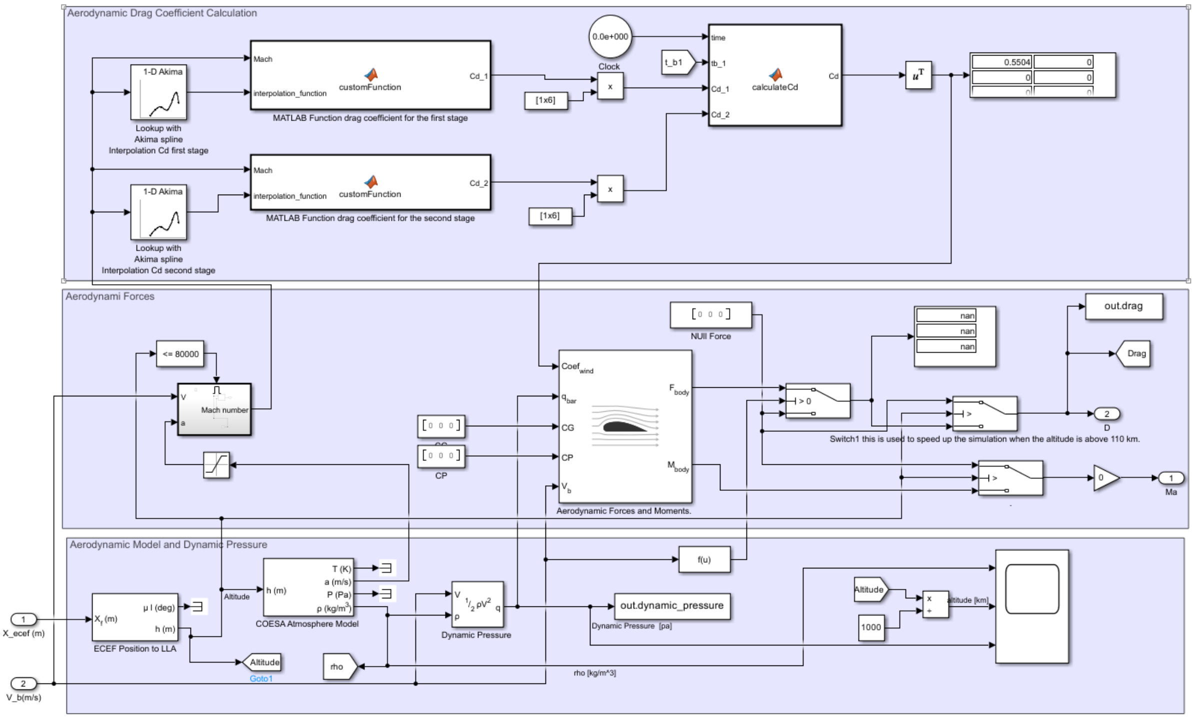


Figure 4.4: Aerodynamic Model

The subsystem for the aerodynamic model is illustrated in Figure 4.4. It consists of three main parts: estimation of aerodynamic coefficients using Missile DATCOM data, calculation of aerodynamic forces and moments, and computation of dynamic pressure. Inputs to the model include altitude, velocity, simulation time, burnout time, and Missile DATCOM data. Outputs include aerodynamic

dynamic pressure, as well as forces and moments. The aerodynamic model assumes that the center of pressure and center of gravity are at the center of the body. Aerodynamic moments are not used for control and guidance, but the generated moments from GNC system (see Section 3.10 for more information). Furthermore, drag calculation stops after fairing separation due to its negligible impact, reducing computational load.

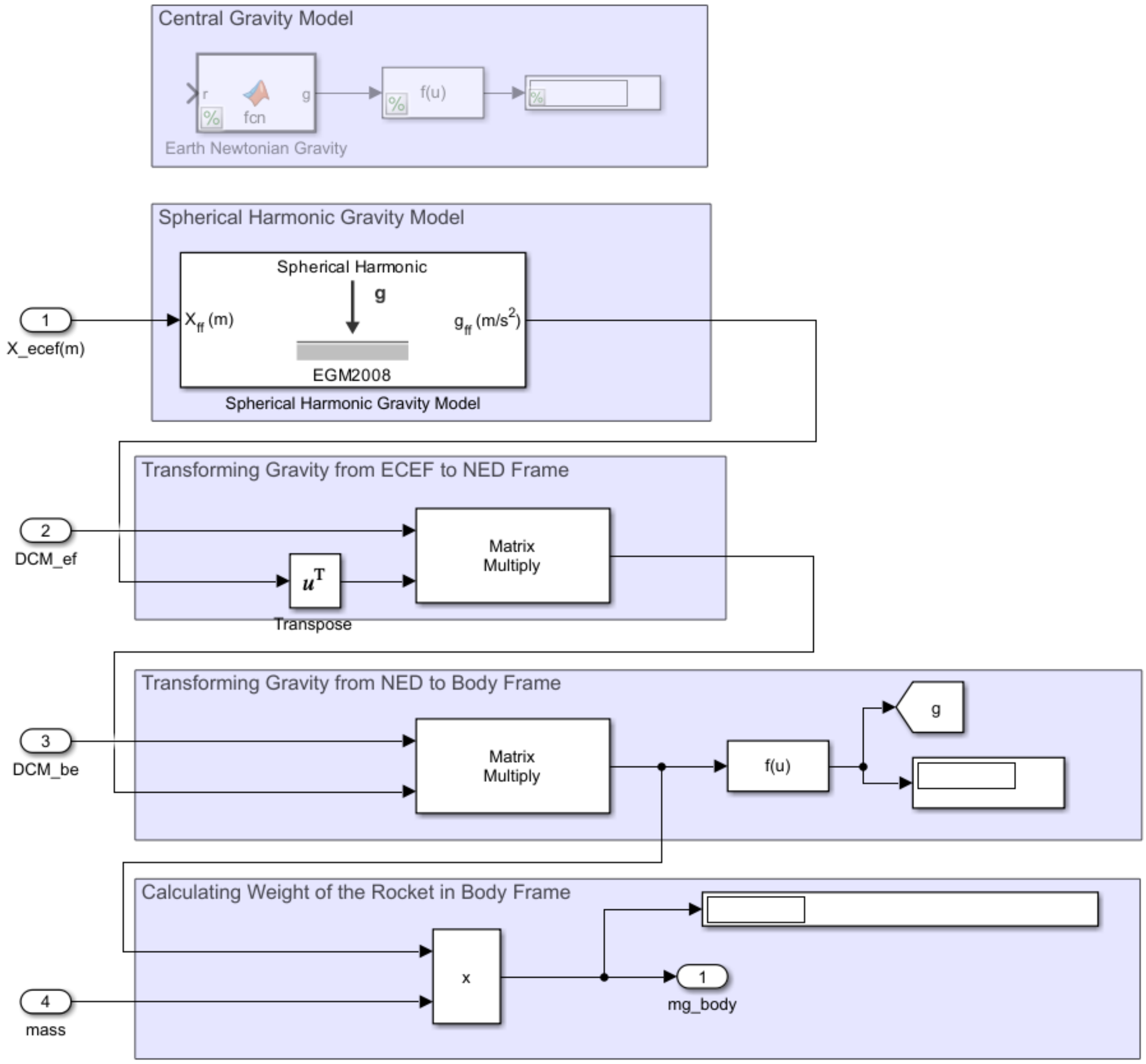

Central Gravity Model
r
fcn
g
f(u)
Earth Newtonian Gravity
Spherical Harmonic Gravity Model
Spherical Harmonic
g
1
X_ecef(m)
X_{ff} (m)
g_{ff} (m/s^2)
EGM2008
Spherical Harmonic Gravity Model
Transforming Gravity from ECEF to NED Frame
2
DCM_ef
Matrix Multiply
u^T
Transpose
Transforming Gravity from NED to Body Frame
3
DCM_be
Matrix Multiply
f(u)
g
Calculating Weight of the Rocket in Body Frame
x
1
mg_body
4
mass

Figure 4.5: Gravity Model

Figure 4.5 illustrates the subsystem for the gravity model. Initially, the Simulink model utilized a central gravity model until all issues were resolved, after which it transitioned to a more precise model known as the Spherical Harmonic Gravity Model.

Figure 4.6 illustrates the subsystem for the GNC model. It consists of three main parts: thrust control, velocity and altitude comparison with the reference velocity and altitude, and attitude control as described in Section 3.10. Additionally, an alternative attitude control for rapid iteration is provided. The principal operation remains the same as regular attitude control. However, now the provided pitch profile and constant roll and yaw are first converted to radians and used as input in the ’Rotation Angles to Quaternions’ block with ZYX rotation order. This is then compared with the simulated attitude, which is also converted to quaternions with ZYX rotation order. The error is multiplied by the gains of a PID controller to obtain the required moments in the body frame for attitude control. Implementation of quaternions speeds up the simulation.

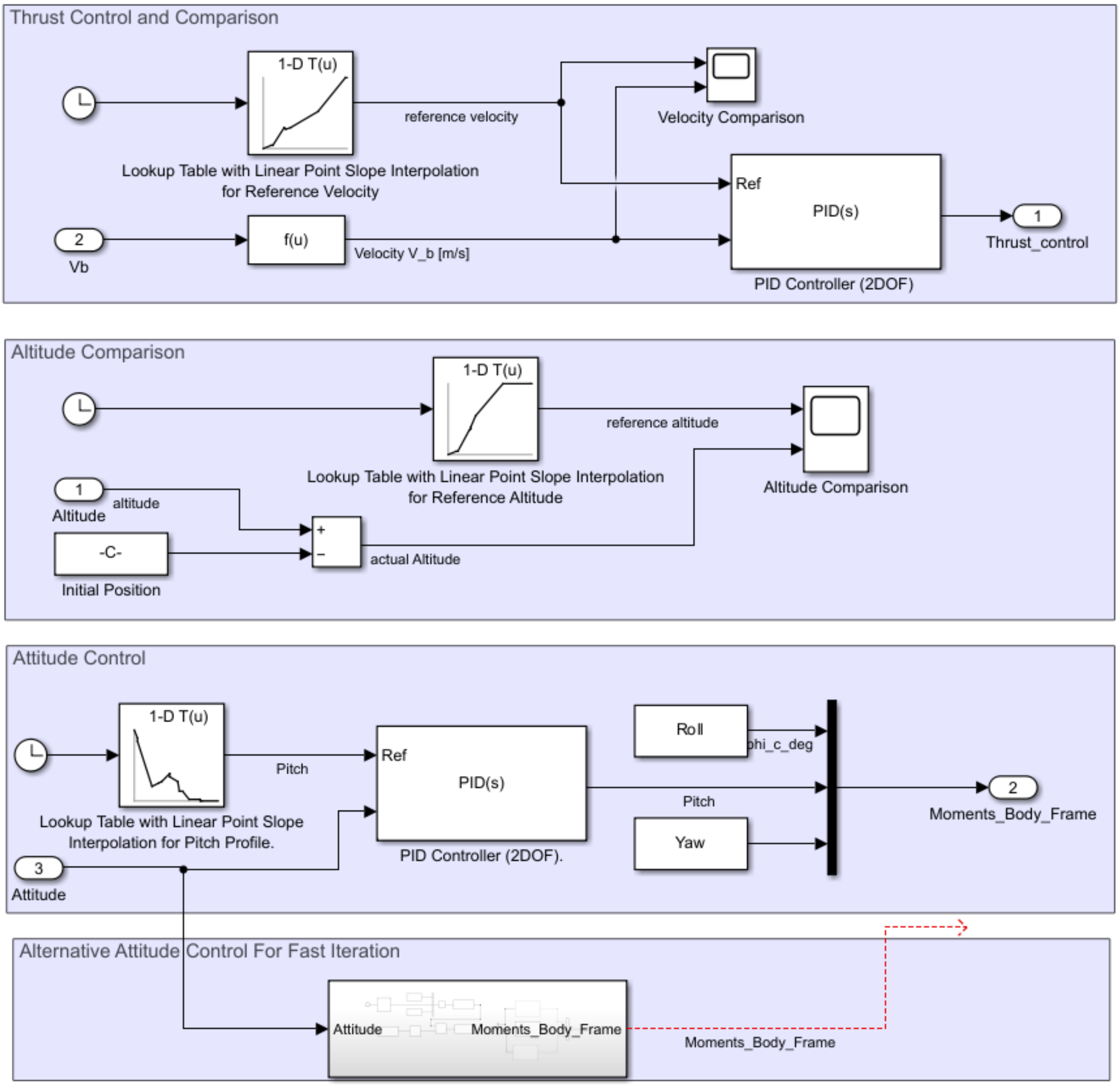


Figure 4.6: Guidance Navigation and Control (GNC) Model

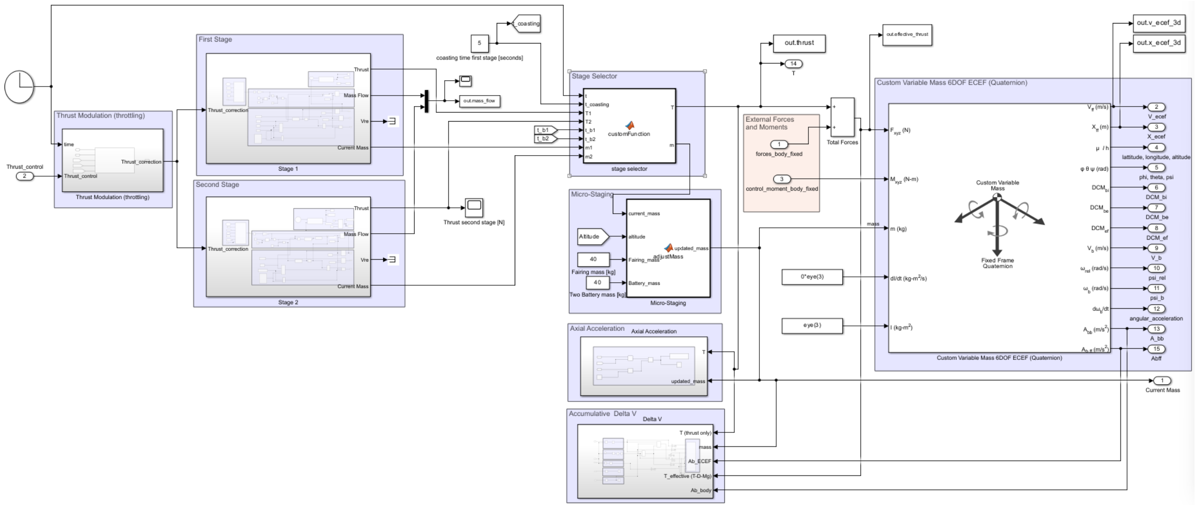


Figure 4.7: Electron Rocket Subsystem

Figure 4.7 illustrates the subsystems of the Electron rocket. It comprises multiple components including the first stage, second stage, thrust modulation, stage selector, micro-staging, quaternion representation of the 6-DOF equations of motion for a custom variable mass in the ECEF reference frame, axial acceleration, and ΔV calculation. Each of these subsystems is clearly presented below.

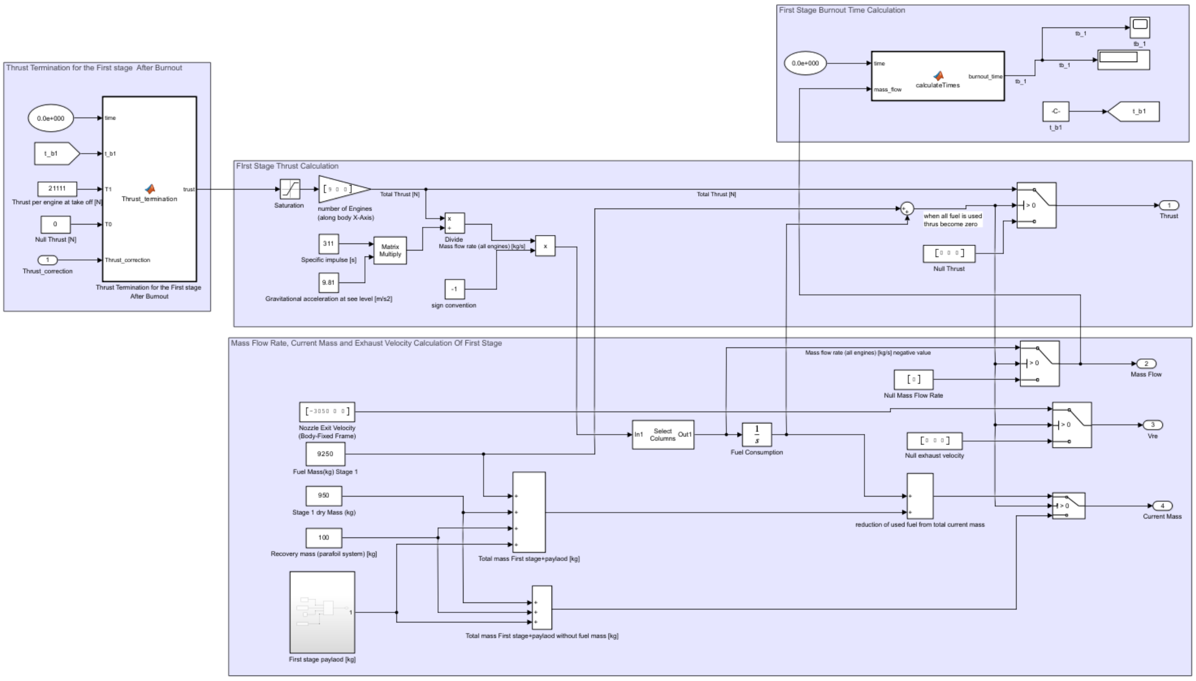

First Stage Burnout Time Calculation
0.0e+000
time
mass_flow
calculateTimes
burnout_time
tb_1
-C-
t_b1
Thrust Termination for the First stage After Burnout
0.0e+000
t_b1
21111
Thrust per engine at take off [N]
0
Null Thrust [N]
1
Thrust_correction
Thrust_termination
Thrust Termination for the First stage After Burnout
First Stage Thrust Calculation
Saturation
number of Engines (along body X-Axis)
Total Thrust [N]
311
Specific impulse [s]
Matrix Multiply
9.81
Gravitational acceleration at see level [m/s2]
Divide
Mass flow rate (all engines) [kg/s]
-1
sign convention
when all fuel is used thrus become zero
Null Thrust
Thrust
Mass Flow Rate, Current Mass and Exhaust Velocity Calculation Of First Stage
Mass flow rate (all engines) [kg/s] negative value
Null Mass Flow Rate
Mass Flow
[-3050 0 0]
Nozzle Exit Velocity (Body-Fixed Frame)
9250
Fuel Mass(kg) Stage 1
950
Stage 1 dry Mass (kg)
100
Recovery mass (parafoil system) [kg]
First stage payload [kg]
Select Columns
Fuel Consumption
Null exhaust velocity
Vre
Total mass First stage+paylaod [kg]
reduction of used fuel from total current mass
Total mass First stage+paylaod without fuel mass [kg]
Current Mass

Figure 4.8: First Stage Model

Figure 4.8 illustrates the first stage of the Electron rocket, depicting calculations for thrust, mass flow rate, burnout time, and current mass.

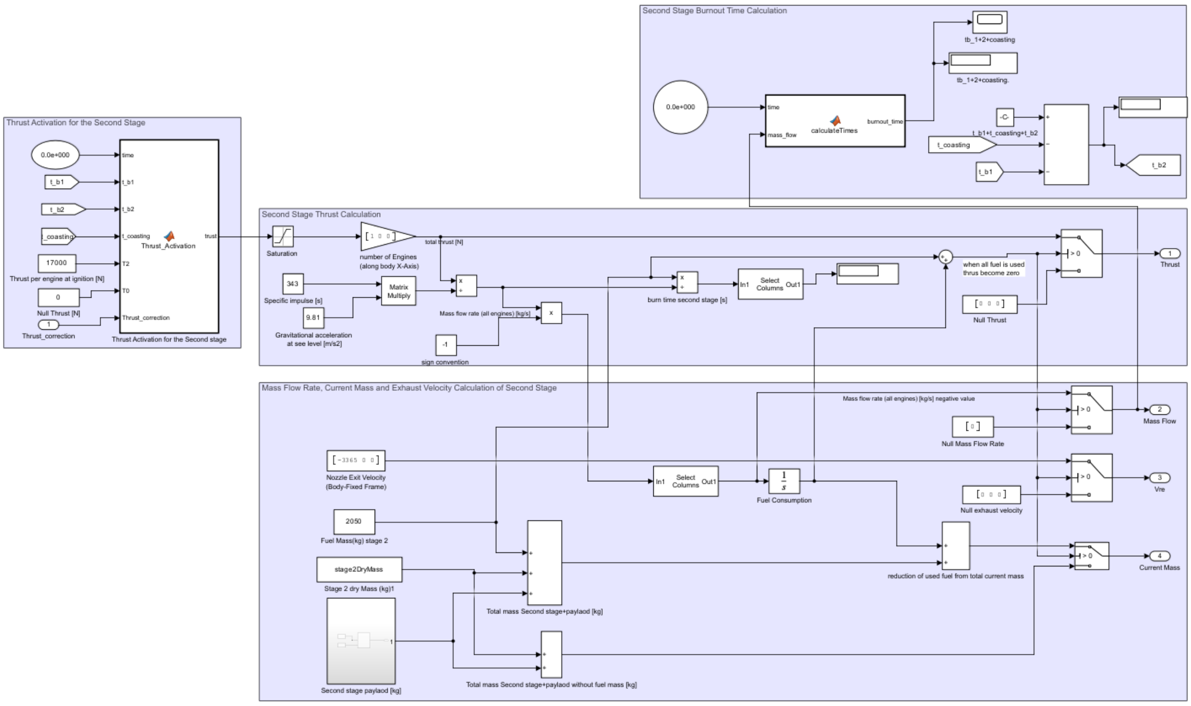


Figure 4.9: Second Stage Model

Figure 4.9 illustrates the second stage of the Electron rocket, detailing key parameters such as thrust, mass flow rate, burnout time, and current mass.

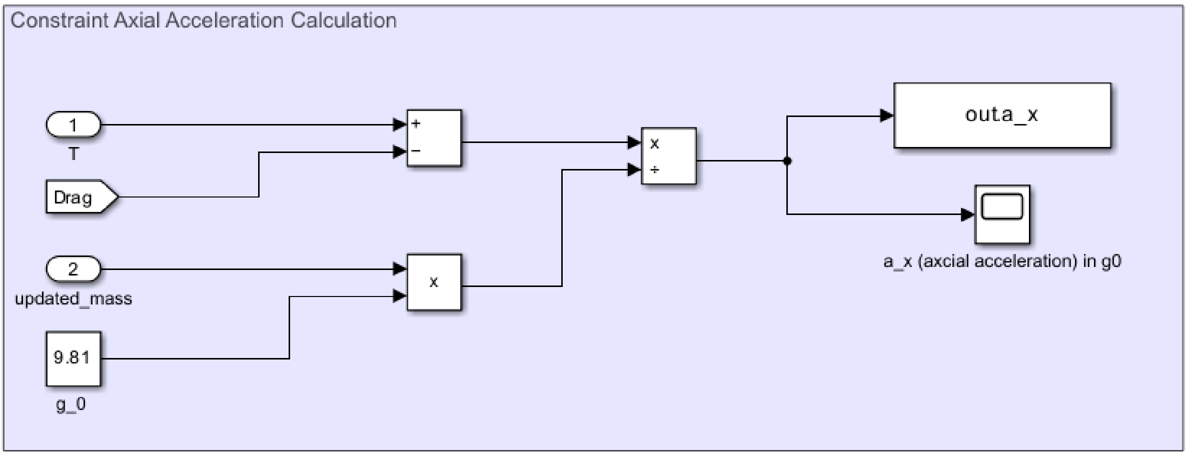

Constraint Axial Acceleration Calculation
1
T
Drag
2
updated_mass
9.81
g_0
+
−
x
x
÷
outa_x
a_x (axcial acceleration) in g0

Figure 4.10: Axial Acceleration Calculation in g

Figure 4.10 illustrates the axial acceleration, expressed in terms of g-forces, of the Electron rocket. This acceleration is crucial as it serves as a constraint in the ascent trajectory.

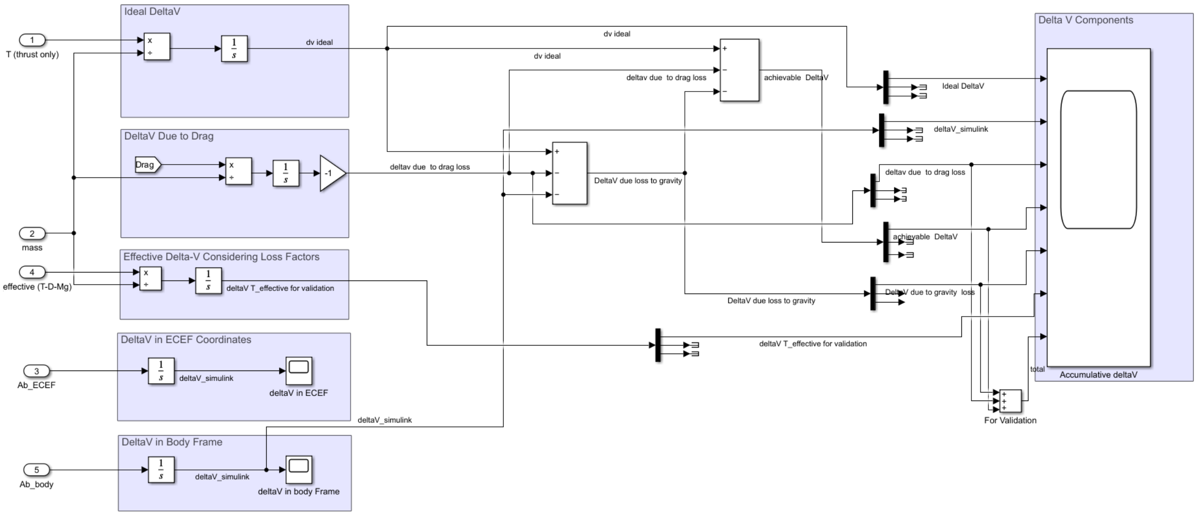


Figure 4.11: Accumulative ΔV Calculation

Figure 4.11 illustrates the accumulative ΔV calculation of the Electron rocket.

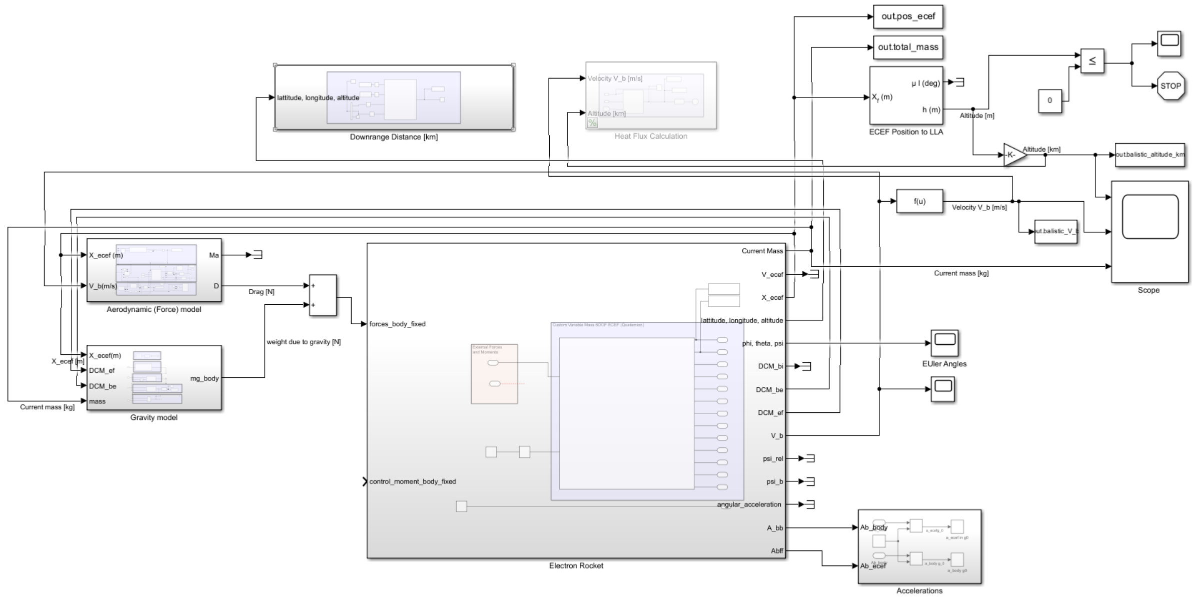


Figure 4.12: Simulink Ballistic Trajectory Model Overview

Figure 4.12 illustrates the Simulink ballistic trajectory model for the first stage. This model is a simplified version of the Simulink ascent trajectory model, focusing solely on the fixed mass condition. It eliminates the need for calculations involving first and second stage variable mass, thrust, and burnout time. Furthermore, the aerodynamic model has been streamlined by removing second stage data, and notably, there is no GNC model, resulting in a purely ballistic trajectory. The remaining subsystems mirror those of the ascent trajectory model.

5 Ascent Trajectory Validation

This chapter elaborates on past missions and the data acquired to establish ascent trajectory constraints in Section 5.1. The validation of the ascent trajectory model is presented in Section 5.2.

5.1. Past Missions

To validate the ascent trajectory of the Electron rocket, it is essential to compare simulation results with actual flight data. The most cost-effective method is to use data from past missions. Unfortunately, Rocket Lab does not publicly share detailed flight data from previous missions. However, they do stream parts of their launches on their YouTube channel. Additionally, on the Rocket Lab's website, under the "Missions" tab, selecting a past mission and clicking on the "Media Press Kit" provides a PDF with useful information about the mission, such as payload mass, orbit altitude, inclination, and event timings as illustrated in Figure 2.5 (e.g., lift-off, maximum dynamic pressure, stage separation etc). However, this data only includes event timings and does not provide velocity, altitude, or pitch profiles.

By performing video analysis of their streams and matching the velocity and altitude to the time of each event, the trajectory of each mission, including velocity profiles can be reconstructed. Subsequently, this reconstructed trajectory can be used to validate the simulation results. Presenting all the data from past missions, which includes tabulated data, velocity profiles, and trajectory plots, would take up too many pages. Therefore, it is presented in Appendix A. To gain a deeper understanding of the typical trajectory of the Electron rocket, altitude and velocity profiles from past missions listed in Table 5.1 are plotted against time, as depicted in Figures 5.1 and 5.2 (see Appendix A for more detailed information).

Table 5.1: Past Missions of the Electron Rocket

Mission Name	Payload [kg]	Orbit	Inclination [°]
The Owl's Night Begins	150	500 km SSO	97.3
The Owl's Night Continues	150	561 km SSO	97
There And Back Again	134	520 km SSO	97
The Owl Spreads Its Wings	100	563 km SSO	97
It Argos Up From Here	118	750 km SSO	98
Catch Me If You Can	50	585 km SSO	98
We Love The Nightlife	165	640 km LEO	53
Virginia is for Launch Lovers	50	550 km circular	40.5
Return to Sender	200	500 km SSO	97.3
Baby Come Back	86	1000 km SSO	99.45

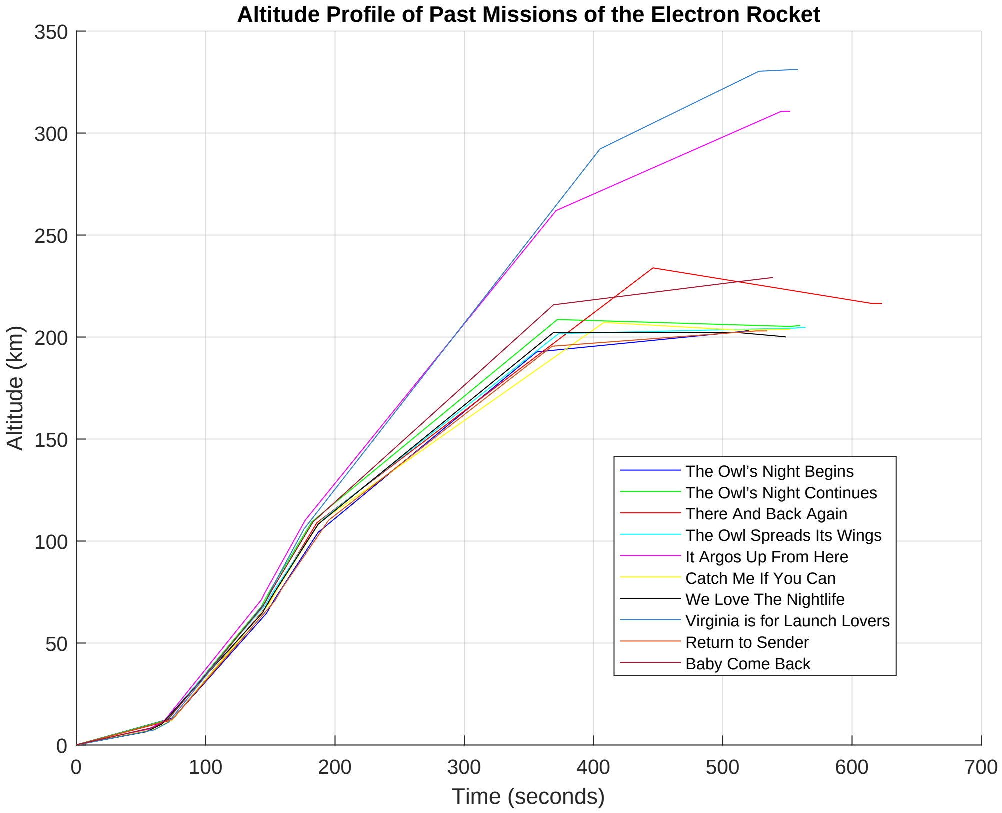


Figure 5.1: Combined Altitude Profile of Past Missions of the Electron Rocket

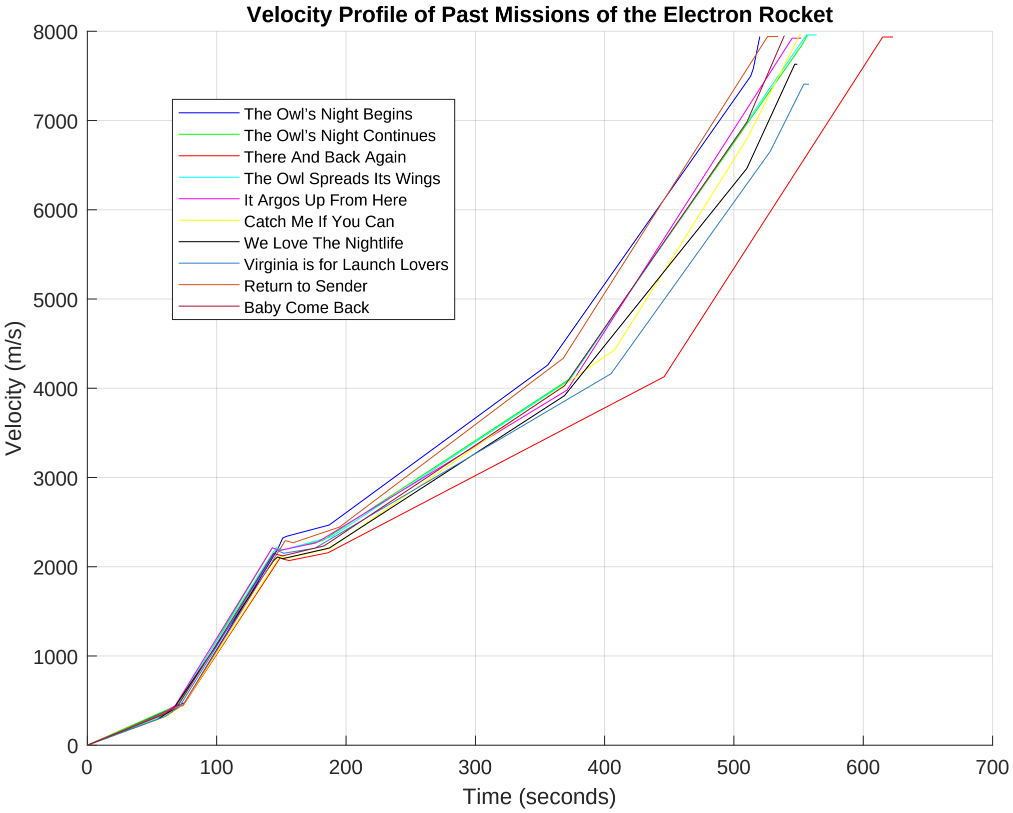


Figure 5.2: Combined Velocity Profile of Past Missions of the Electron Rocket

Based on the data presented in Appendix A about past Electron rocket missions, several similarities can be observed. Stage 1 separation occurs at an altitude between 70 and 80 km, as shown in Figure 5.2 around 150 seconds into the flight. Fairing separation typically happens at an altitude of around 110 km, followed by battery jettison at an altitude between 193 and 200 km for most missions. Figure 5.2 indicates that the velocity profile of past missions follows a consistent pattern until circular velocity is achieved. For instance, in the mission "We Love The Night Life" (black color), stage 2 separation from the Kick stage occurs at 200 km altitude with a corresponding velocity of around 7700 m/s, which is approximately 2% less than the theoretical circular velocity at 200 km altitude. Referring to Figure 5.1, a noticeable pattern emerges among most missions, with a few exceptions. For instance, missions at higher altitudes like "It Argos Up From Here" (magenta color) or those launched from Launch Complex 2, such as "Virginia is for Launch Lovers", deviate from this pattern. It's important to note that the majority of launches originate from Launch Complex 1. Given the consistent pattern in these trajectories, they provide a reliable basis for establishing the maximum dynamic pressure constraint. Appendix A details the velocity and altitude of the Electron rocket for each mission at the point of maximum dynamic pressure. This data enables the calculation of maximum dynamic pressure for each mission, with the highest value determining the constraint for this simulation. Figure 5.3 illustrates how this is determined in Simulink, and the corresponding results are provided in Table 5.2.

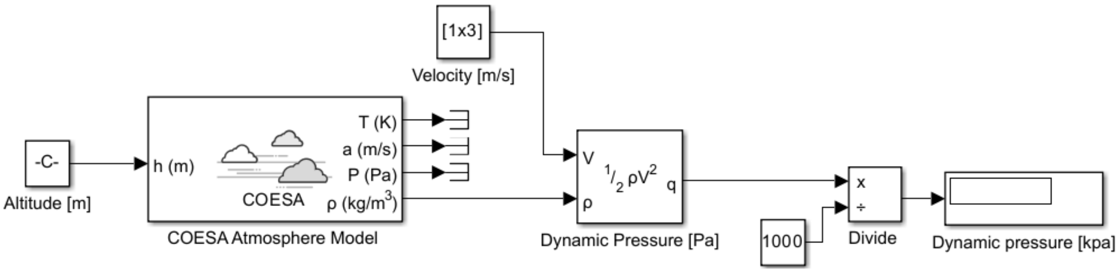


Figure 5.3: Maximum Dynamic Pressure Estimation Block

The highest value is around 34 kPa; therefore, the maximum dynamic pressure constraint is rounded to 35 kPa.

Table 5.2: Maximum Dynamic Pressure Across Past Electron Rocket Missions

Mission Name	Maximum Dynamic Pressure [kPa]
The Owl's Night Begins	28.6
The Owl's Night Continues	30.24
There And Back Again	29.25
The Owl Spreads Its Wings	31.28
It Argos Up From Here	30.93
Catch Me If You Can	29.59
We Love The Nightlife	31.19
Stronger Together	33.61
Virginia is For Launch Lovers	32.9
Baby Come Back	30.16
Return To Slender	28.88

Any of the aforementioned missions can be used to validate the results. Since the pitch profile needs to be adjusted manually to achieve the desired trajectory, as explained in Section 3.10, only one mission will be selected for validation. For no particular reason, "The Owl Spreads Its Wings" (see Figure 5.4) will be used.

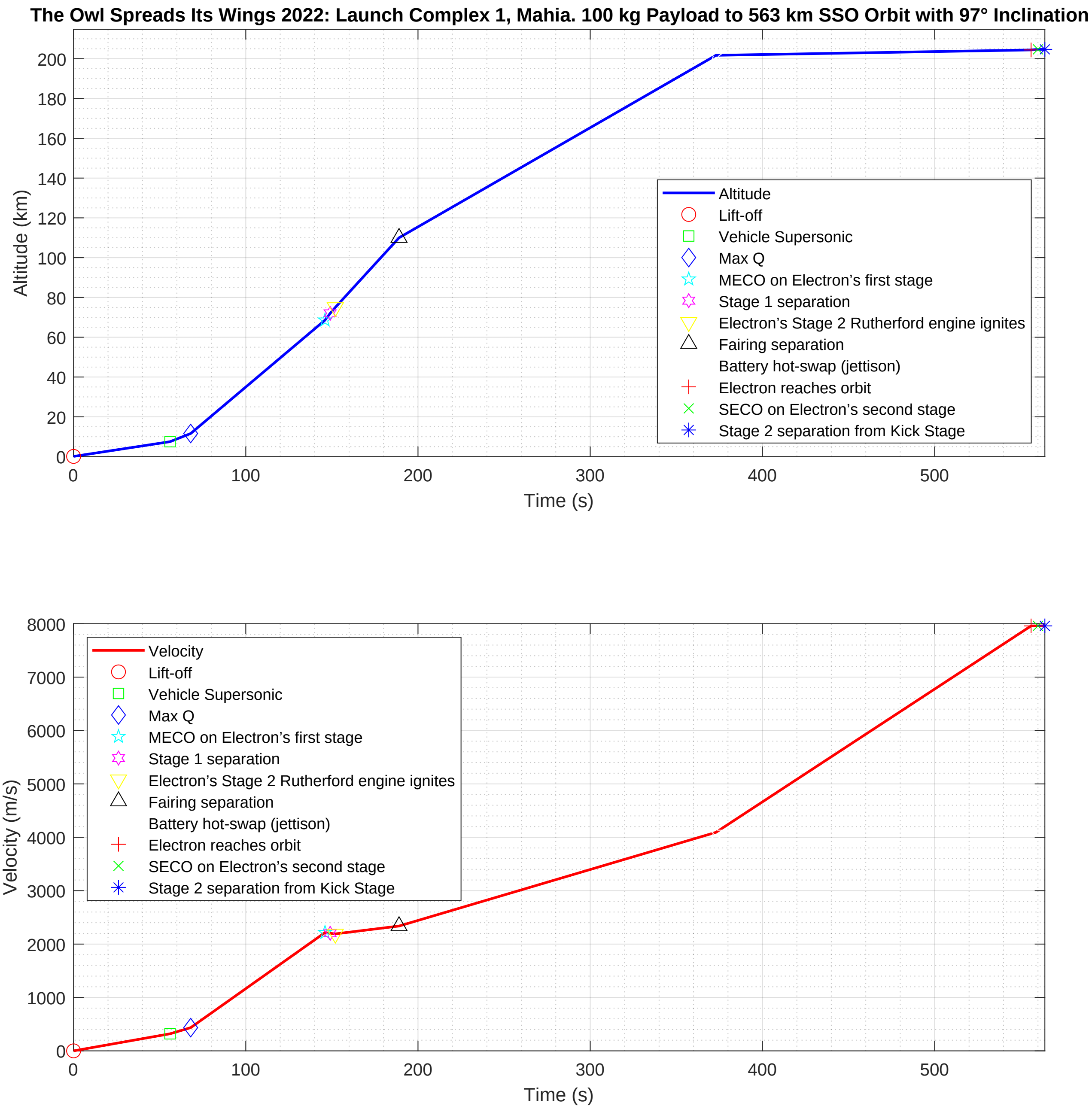
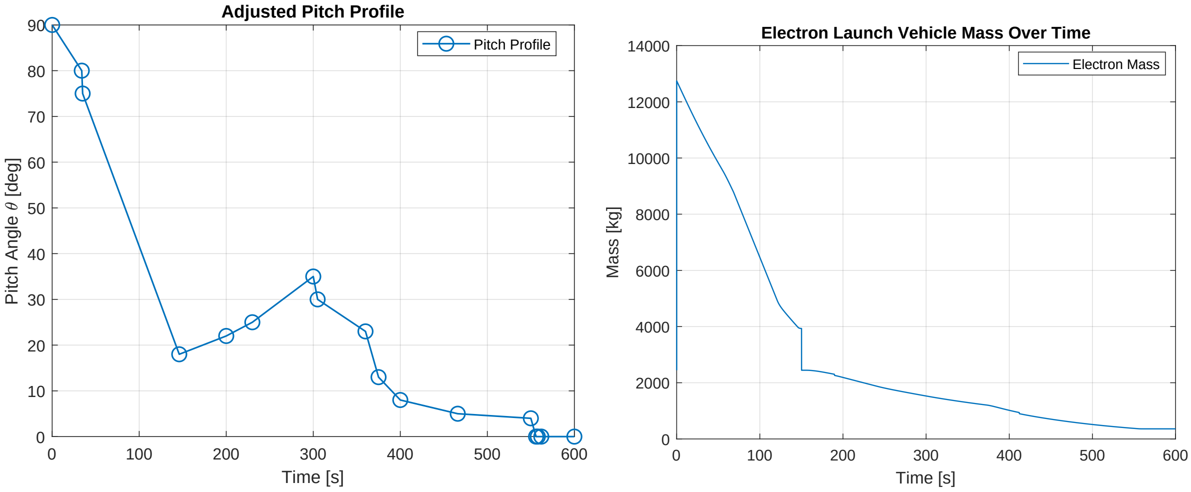


Figure 5.4: Altitude and Velocity Profile of the Mission "The Owl Spreads Its Wings"

5.2. Trajectory Validation

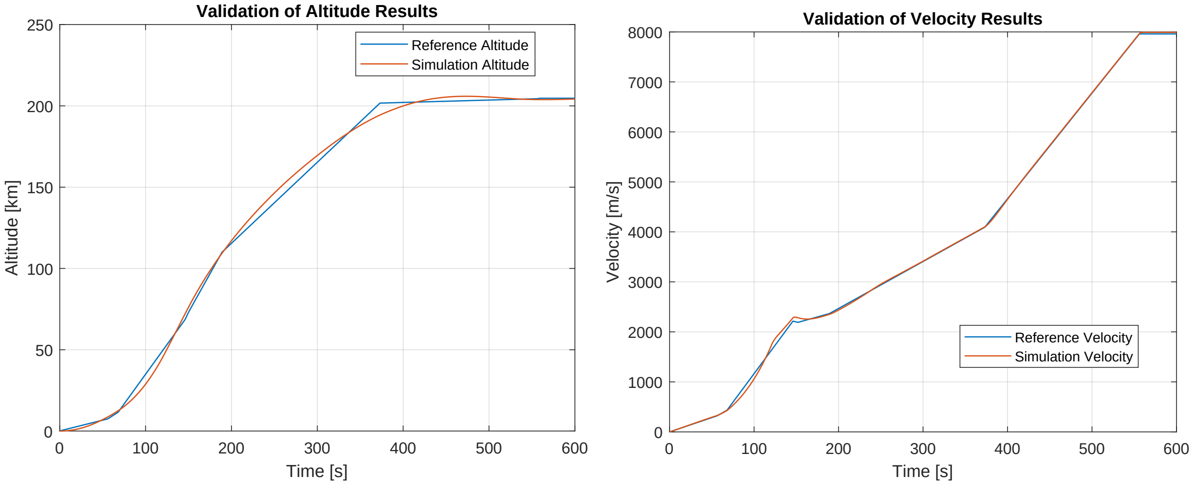
To validate the ascent trajectory model, the simulation results will be compared to the actual data from the mission *The Owl Spreads Its Wings*, as shown in Figures 5.4 and A.4, and detailed in Table A.4. All the input data is sourced from Table 2.1. The recovery mechanism mass is 100 kg and the pitch profile used to generate the results is shown in Figure 5.5a. The payload mass is assumed to be 100 kg. The mass profile is shown in Figure 5.5b. The trajectory results are presented in Figure 5.6, 5.8 and 5.7. The simulation results do not violate the ascent trajectory constraints discussed in Section 3.9. The maximum dynamic pressure experienced is approximately 30 kPa, which is lower than the estimated 31.28 kPa for the same mission, as shown in Table 5.2. After fairing separation, the maximum heat flux ($\dot{Q}$) is approximately 230 W/m^2, well within the constraint of 1135 W/m^2. The maximum axial acceleration is approximately 4g, which is below the constraint value of 7.5g.



(a) Pitch Profile for Ascent Trajectory Validation

(b) Electron Launch Vehicle Mass Profile

Figure 5.5: Electron Launch Vehicle Pitch and Mass Profiles of Ascent Trajectory model Validation



(a) Altitude Profile of Ascent Trajectory

(b) Velocity Profile of Ascent Trajectory

Figure 5.6: Electron Launch Vehicle Altitude and Velocity Profiles of Ascent Trajectory model Validation

Figure 5.5b illustrates the mass changes after takeoff. The mass steadily decreases until the first stage burnout at around 150 seconds. The sudden drop in mass at this point corresponds to the first stage separation. Following a 5-second coasting period where the mass remains constant, it begins to decrease again. When zoomed in, at approximately 190 seconds, a small drop in mass is observed due to fairing separation, and another small drop at around 412 seconds is due to battery jettison. In Figure 5.6, the simulation results for the altitude and velocity profiles closely align with the reference profiles. This strong correlation between the reference and simulation data across all phases underscores the model's reliability. Figures 5.7 and 5.8 offer detailed insights into the velocity and altitude profiles, as well as the percentage error between the simulation results and the reference profiles. Furthermore, these figures use color coding to clearly distinguish between graph sections: red denotes the first stage, green represents the second stage and blue indicates the coasting phase. The discrepancy between the simulated altitude and the reference altitude is significant during takeoff but diminishes over time. At the first stage burnout (150 seconds), it measures approximately -4.3%, decreasing to 0.3% by the end of the simulation. Similarly, the error between the simulated velocity and the reference value peaks at around 84 seconds, reaching approximately 14%, before narrowing to -0.3% at the simulation's conclusion. Overall, the simulated results look good, and the error is low. Therefore, it is concluded that the ascent trajectory model is validated.

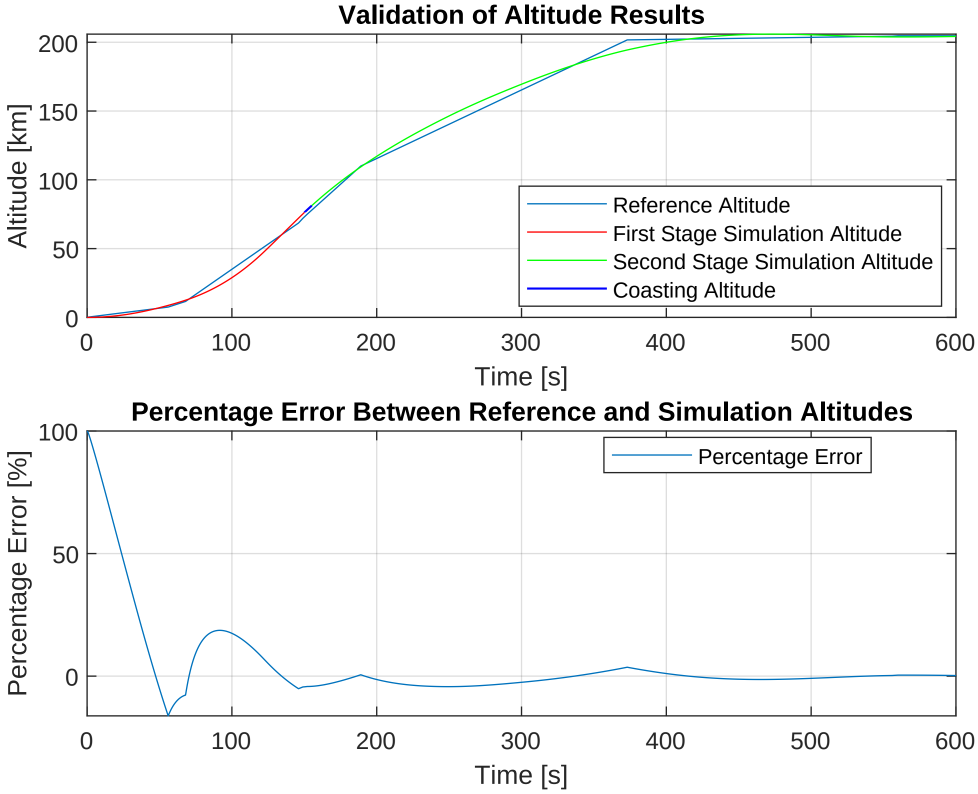


Figure 5.7: Electron Altitude and Error Profiles of Ascent Trajectory model Validation

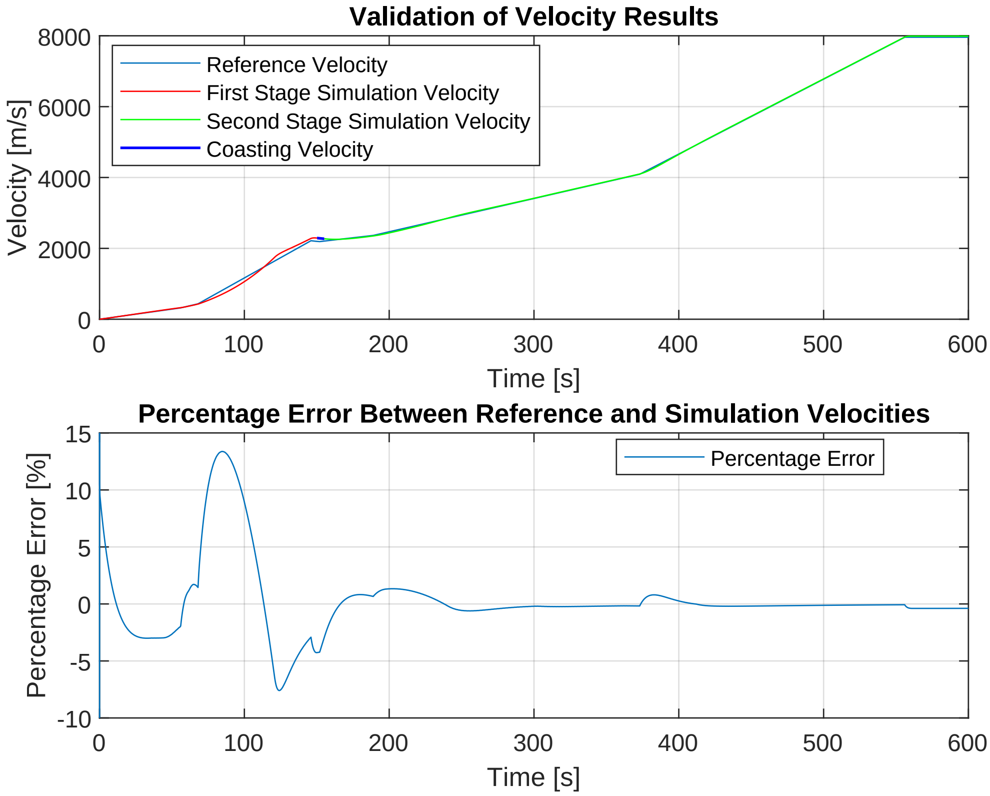


Figure 5.8: Electron Velocity and Error Profiles of Ascent Trajectory model Validation

6

Results of the Ascent Phase

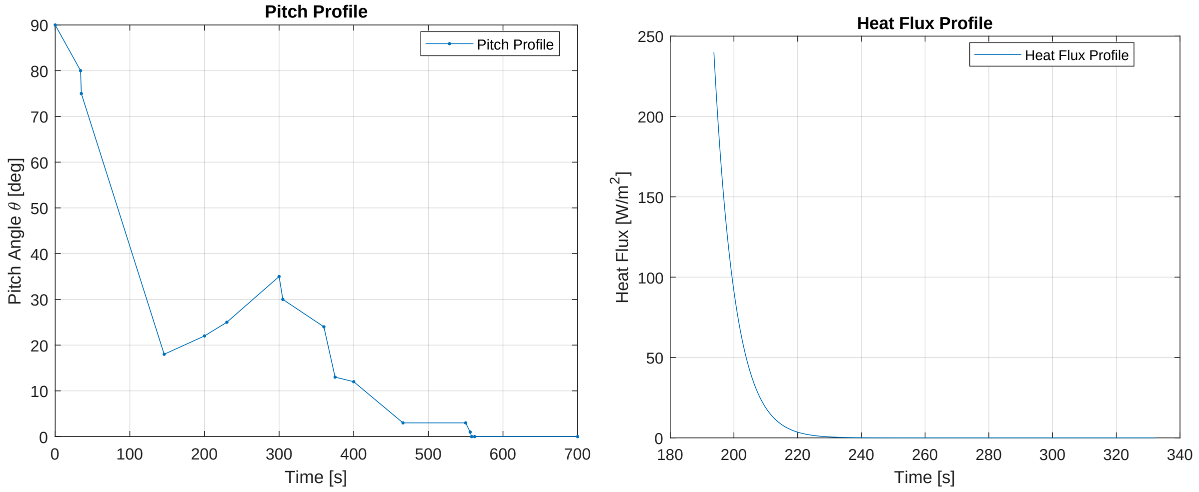
This chapter presents the results of the ascent phase of the Electron rocket as described in Section 6.1, divided into three sections. Section 6.2 presents the results of the ascent trajectory model for the first and second stages, while Section 6.3 discusses the results of the Hohmann transfer orbit. Section 6.4 provides the results of the first stage ballistic trajectory.

6.1. Mission Outline for the Ascent phase

The mission to be simulated is a launch from Launch Complex 1, located in Mahia, New Zealand. The site is situated at 39.262°S latitude and 177.865°E longitude. The payload is a 100 kg satellite to be delivered into a sun-synchronous orbit at 500 km altitude. The recovery mechanism mass is 100 kg (see Section 8.5). The inclination for this mission is chosen to be 97°. Other data for the Electron launch vehicle is taken from Table 2.1. The payload is initially inserted into a circular parking orbit (initial orbit) at around 200 km altitude using the first and second stages of the Electron launch vehicle. Subsequently, the orbit is raised to 500 km altitude using the kick stage's Curie engine. It is crucial to ensure that the ascent trajectory constraints detailed in Section 3.9 are not violated. As mentioned in Section 3.10, the simulation model does not include a reference trajectory generation model to be used in guidance, navigation, and control. Therefore, an adjusted trajectory similar to past missions like *The Owl's Night Begins*, *The Owl's Night Continues*, and *The Owl Spreads Its Wings* will be utilized as a reference trajectory for guidance, navigation and control. The trajectories of these missions are shown in Figures 5.1 and 5.2 and are detailed in tables and graphs in Appendix A. The adjustment to the past mission's trajectory mentioned above is necessary because, at second-stage burnout, the orbit should be circular to facilitate the transition to a Hohmann transfer orbit leading to the final 500 km orbit. Furthermore, based on past missions, fairing separation and battery jettison are to take place at altitudes of 110 km and 195 km, respectively.

6.2. Results of the Ascent Trajectory Model

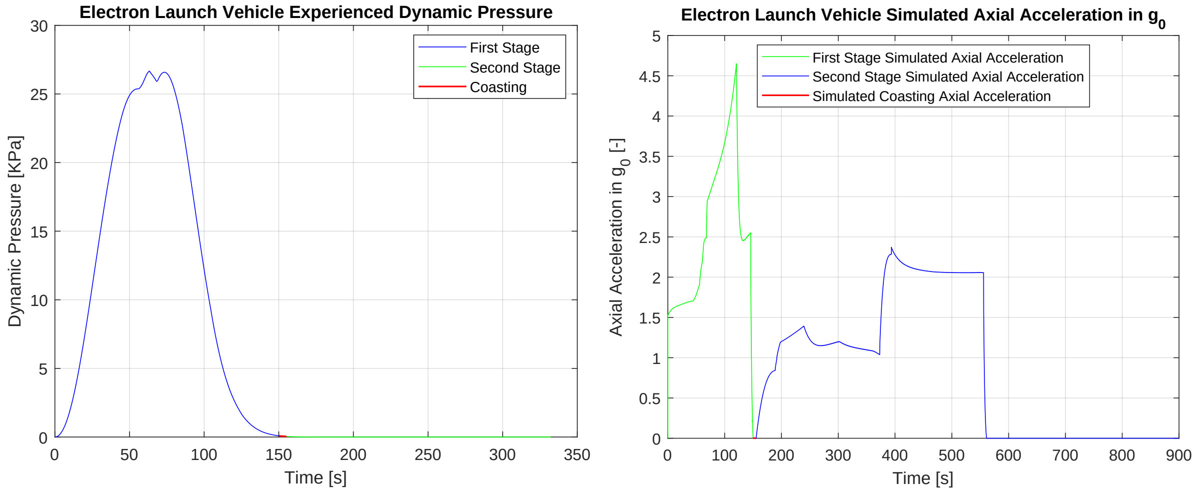
The simulation results do not violate the ascent trajectory constraints discussed in Section 3.9.The maximum dynamic pressure experienced is 26.7 kPa as shown in Figure 6.2a, which is lower than the constraint value of 35 kPa. The maximum heat flux ($\dot{Q}$) experienced After fairing separation, is 240 W/m^2 as presented in Figure 6.1b, well within the constraint of 1135 W/m^2. The maximum axial acceleration experienced is 4.65g as shown in Figure 6.2b, which is below the constraint value of 7.5g. The mass flow rate and thrust profiles of ascent trajectory is provided in Figure 6.3. The negative sign in Figure 6.3a is due to the sign convention used in the reference frame. The first stage burns out at 150.2 seconds, followed by a 5-second coasting phase. The second stage operates for 406 seconds. Consequently, after 561.2 seconds, the vehicle ceases to have thrust available. Since the first stage has more engines than the second stage, its thrust and mass flow rate are also higher.



(a) Pitch Profile for Ascent Trajectory

(b) Heat Flux Profile After Fairing Separation

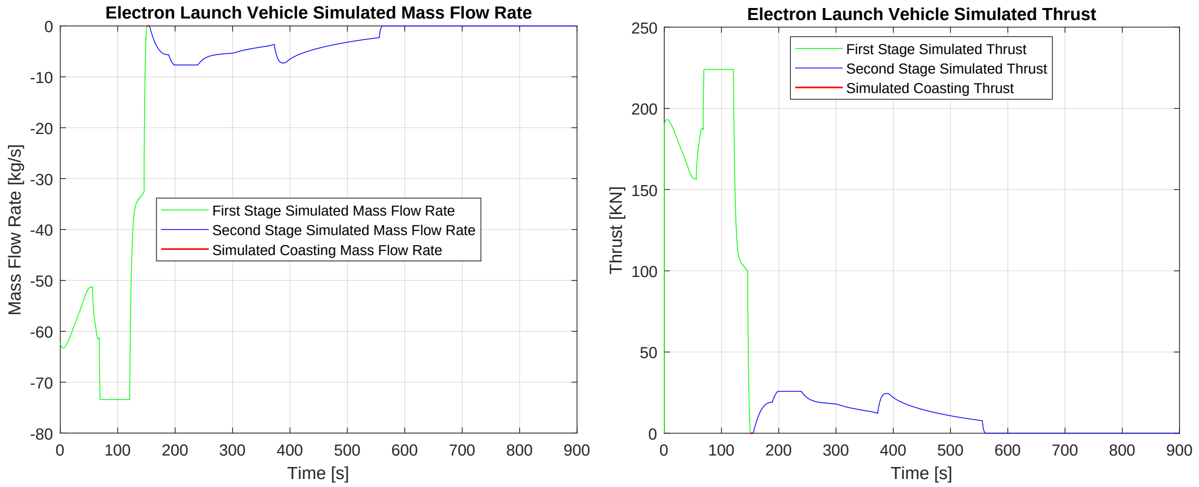
Figure 6.1: Electron Launch Vehicle Pitch and Heat Flux Profiles of Ascent Trajectory



(a) Dynamic Pressure Profile of Ascent Trajectory

(b) Axial Acceleration of Ascent Trajectory

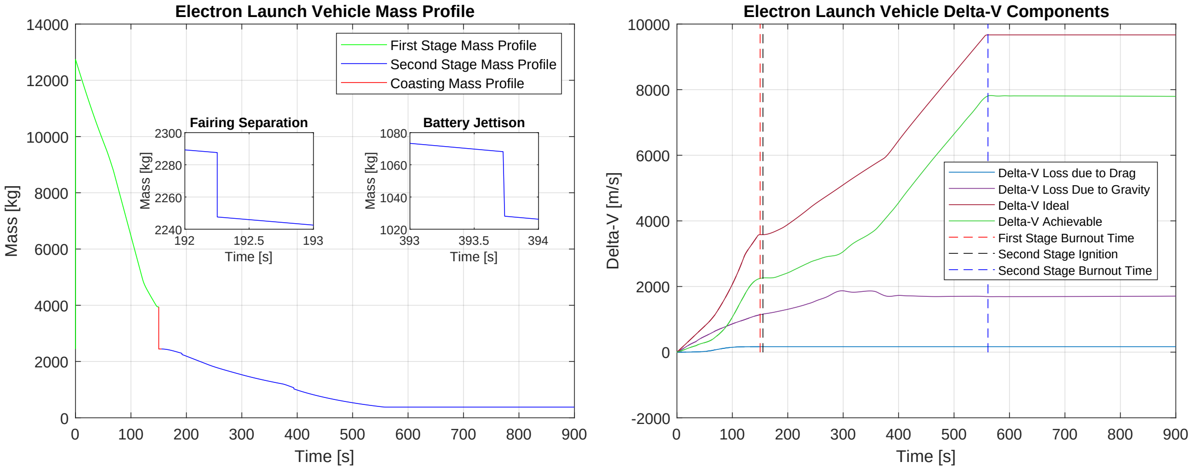
Figure 6.2: Electron Launch Vehicle Dynamic Pressure and Axial Acceleration Profiles of Ascent Trajectory



(a) Mass Flow Rate Profile of Ascent Trajectory

(b) Thrust Profile of Ascent Trajectory

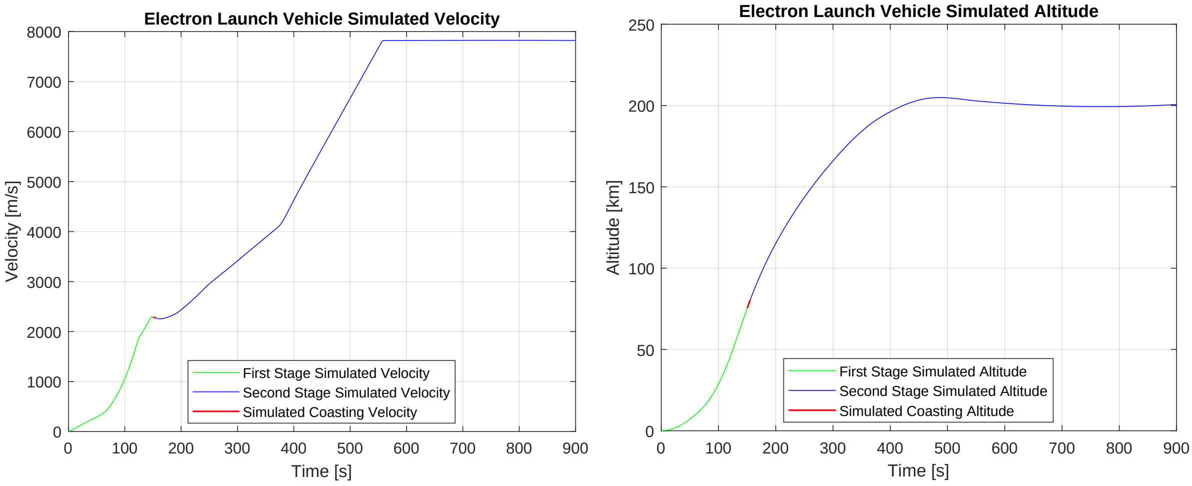
Figure 6.3: Electron Launch Vehicle Mass Flow Rate and Thrust Profiles of Ascent Trajectory



(a) Mass Profile of Ascent Trajectory

(b) ΔV Profile of Ascent Trajectory

Figure 6.4: Electron Launch Mass and ΔV Profiles of Ascent Trajectory



(a) Velocity Profile of Ascent Trajectory

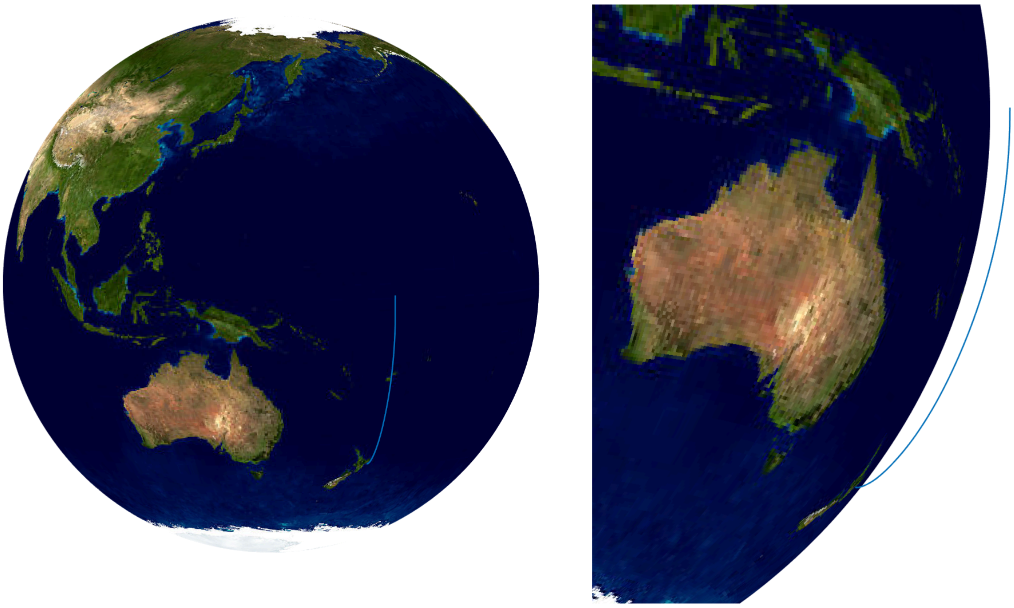
(b) Altitude Profile of Ascent Trajectory

Figure 6.5: Electron Launch Vehicle Velocity and Altitude Profiles of Ascent Trajectory

The Electron launch vehicle mass profile is shown in Figure 6.4a. The simulated mass profile shows the mass changes after takeoff. The mass steadily decreases until the first stage burnout. The sudden drop in mass at this point corresponds to the first stage separation. Following a 5-second coasting period, during which the mass remains constant, it begins to decrease again. At approximately 192.2 seconds, a small drop in mass is observed due to the fairing separation, which has a mass of 40 kg. A zoomed-in region is provided in the same figure. Another small drop at around 393.7 seconds is due to the battery jettison, which also has a mass of 40 kg. A zoomed-in region of this event is provided as well.

Figure 6.4b presents the ΔV components of the ascent trajectory. The blue line represents ΔV loss due to drag, while the purple line represents ΔV loss due to gravity. The dark red graph depicts the ideal ΔV, which exhibits the highest values. Subtracting the ΔV losses due to drag and gravity from the ideal ΔV yields the achievable ΔV, indicated by the green line. The graph originally exhibited oscillations, as shown in Figure C.1; however, a Savitzky-Golay finite impulse response (FIR) smoothing filter was implemented to remove them [46]. Additionally, three vertical dashed lines mark key moments: first stage burnout, second stage ignition, and second stage burnout. The ΔV loss due to drag at the second stage burnout is 1.73% of the ideal ΔV, while the ΔV loss due to gravity is 17.5%. Therefore, the achievable ΔV is only 80.8% of the ideal ΔV.

The simulation results for the altitude and velocity profiles are presented in Figure 6.5. The velocity profile exhibits a similar shape to past missions, as shown in Figure 5.2. At second stage burnout, the velocity reaches 7.821 km/s, which is slightly higher than the orbital velocity of approximately 7.79 km/s. Therefore, the percentage error between the simulated velocity at second stage burnout and the theoretical orbital velocity is approximately 0.4%. The altitude profile exhibits a slight overshoot around 450 to 550 seconds. After this period, the altitude overshoot diminishes and stabilizes at 200 km altitude. The eccentricity of the initial orbit is calculated and is equal to 0.0069. The drag data is depicted in Figure C.5. The effective thrust, which is obtained by subtracting drag and weight from the thrust, is provided in Figure C.6. For a more detailed view of the ascent trajectory results, enlarged figures are available in Appendix C. Figure 6.6 presents the 3D visualization of the simulated altitude profile depicting a launch trajectory from launch Complex 1, located in Mahia, New Zealand, from south to north direction. Conversely, Figure C.7 demonstrates that launches in the opposite direction, from north to south, are feasible with yaw angle control.



(a) Ascent Trajectory Visualization

(b) Zoomed in from the Left Side of the Trajectory Line

Figure 6.6: Electron Launch Vehicle Ascent Trajectory Visualization in 3D

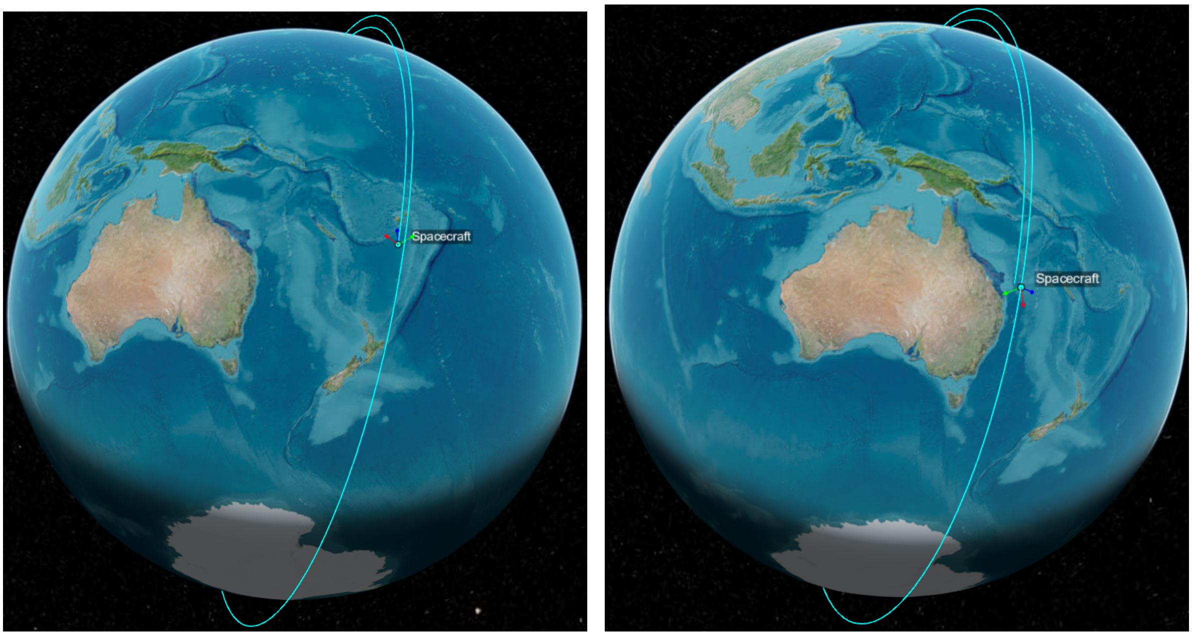
6.3. Results of the Hohmann Transfer Orbit

The Kick Stage, including its cold gas thrusters, has a mass of approximately 45 kg [4]. These thrusters are used for attitude control and stabilization, ensuring precise orientation and positioning of the payload during deployment. The payload itself has a mass of 100 kg. For the ascent trajectory, a total mass of 250 kg was purposefully overestimated to account for transfer orbit, potential deorbiting and to allow for a even higher final orbit if required. The orbital inclination is 97 degrees, with an initial orbit eccentricity of 0.0069. As discussed in Section 3.8, the true anomaly of the transfer orbit is 0 degrees at the first transfer point and 180 degrees at the second transfer point. The MathWorks Simulink model for Hohmann transfer orbit can accept inputs as Keplerian orbital elements and or as fixed-frame (ECEF) positions in meters and fixed-frame velocities in m/s [43]. The second stage burnout position and velocity in ECEF reference frame are provided in table C.1. These values are used as inputs for the Hohmann transfer orbit model. To achieve the assumed impulsive maneuvers described in Section 3.8, where the change in velocity is considered instantaneous, a high mass flow rate of 400 kg/s is assumed. This results in burn durations of 0.017 seconds for the first transfer point and 0.0163 seconds for the second. The results for the other orbital parameters of the Hohmann transfer orbit are presented in Table 6.1. The total ΔV for the Hohmann transfer orbit is 171.9 m/s, which correspond to 13.32 kg of propellant.

Table 6.1: Results of Hohmann Transfer Orbit Parameters

Symbol	Parameter	Value
a_T	Semimajor axis of the transfer orbit	6721 km
t_T	Transfer duration	2.742×10^3 s ($\approx$ 45.7 min)
v_1	Circular velocity on the initial orbit	7.797 km/s
v_2	Circular velocity on the final orbit	7.617 km/s
v_{T1}	Velocity on the transfer orbit at the first transfer point	7.875 km/s
v_{T2}	Velocity on the transfer orbit at the second transfer point	7.531 km/s
Δv_1	Magnitude of Δv at the first transfer point	86.43 m/s
Δv_2	Magnitude of Δv at the second transfer point	85.47 m/s
$\hat{\mathbf{v}}_1$	Direction of ΔV at the first transfer point	$[0.122, 0, 0.993]^T$
$\hat{\mathbf{v}}_2$	Direction of ΔV at the second transfer point	$[-0.122, 0, -0.993]^T$
m_1	Mass after maneuver at the first transfer point	243.21 kg
m_2	Mass after maneuver at the second transfer point	236.68 kg
$m_{\text{propellant}}$	Total propellant burned	13.32 kg

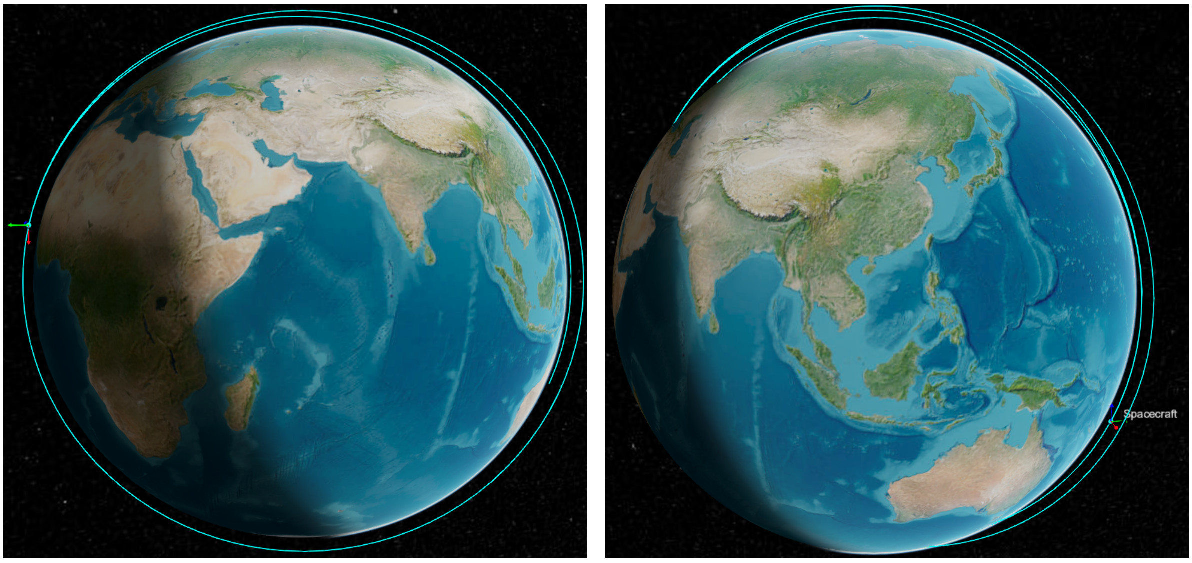
The trajectory of the transfer orbit is visualized in Figure 6.7 and 6.8. The body's x-, y-, and z-axes are indicated by the red, green, and blue arrows, respectively.



(a) The Spacecraft is in its Initial Orbit

(b) The Spacecraft is in its Final Orbit

Figure 6.7: Hohmann transfer Orbit Visualization Using Simulink Aerospace Toolbox satellite Scenario Object



(a) Completion of Transfer Orbit

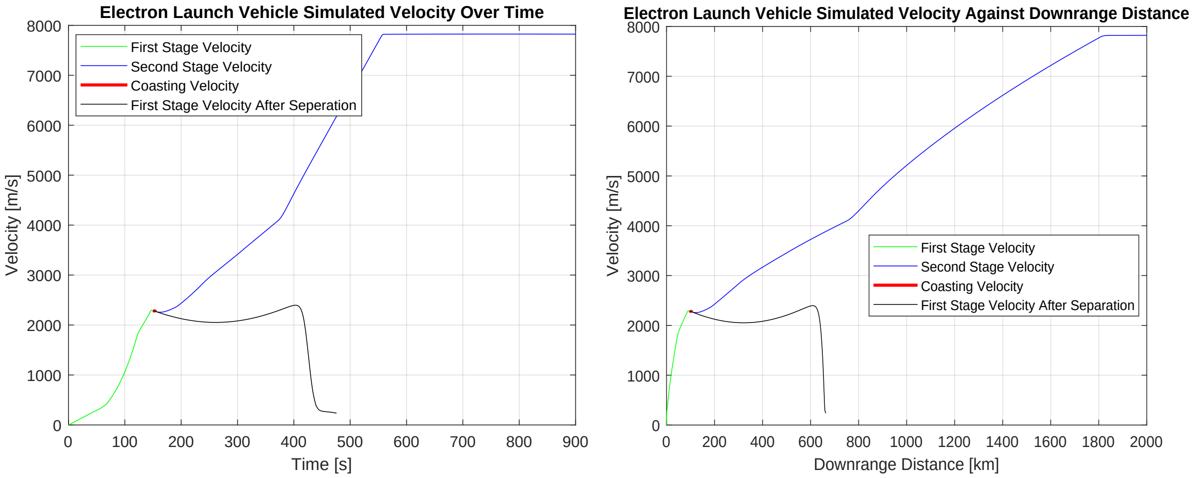
(b) Initial, Transfer, and Final Orbits of the Spacecraft

Figure 6.8: Visualizing Initial, Transfer, and Final Orbits of the Spacecraft

In Figure 6.7a, the spacecraft is shown just after the second stage burnout, representing its initial orbit. Moving to Figure 6.7b, the spacecraft is depicted in its final orbit. The influence of Earth's rotation is evident as the launch site in Mahia, New Zealand, has shifted eastward and is no longer directly beneath the spacecraft's trajectory. Figure 6.8b depicts the initial, transfer, and final orbits of the spacecraft. The transfer orbit, shown as the middle trajectory line, connects the initial orbit to the final orbit. Figure 6.8a depicts the completion of the transfer orbit.

6.4. Results of the First Stage Ballistic Trajectory

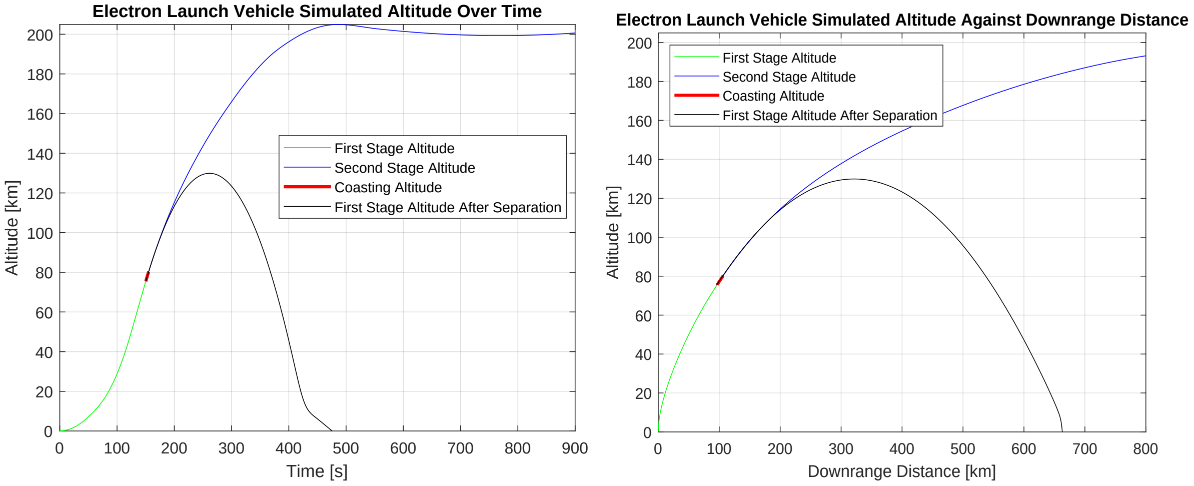
What happens if the first stage is not recovered? The altitude and velocity profiles of the Electron with the first stage ballistic trajectory are presented in Figures 6.9 and 6.10. At first stage burnout, the velocity is 2291 m/s (8248 km/h) (see Figure 6.9a) and the corresponding altitude is 75.48 km. After separation, the velocity starts to decrease, but the first stage still has significant momentum and continues to gain altitude. At 262 seconds, the first stage reaches its apogee at an altitude of 130 km (see Figure 6.10a), with the velocity decreasing to 2053 m/s (7391 km/h). The first stage then begins its descent back to Earth in a ballistic trajectory. Due to gravitational acceleration, the first stage starts to accelerate, causing the velocity to increase again. At 403 seconds, the maximum velocity of 2397 m/s (8630 km/h) is reached at an altitude of 41.88 km. Thereafter, the velocity decreases rapidly due to drag from air resistance. The apparent sudden change in the horizontal direction of the velocity and altitude at the end of the graphs in Figures 6.9a and 6.10a is due to the x-axis being in the time domain. As the velocity decreases, the descent of the first stage takes longer, causing the velocity and altitude lines to seem as though they abruptly change direction. However, when the velocity and altitude profiles are plotted against downrange distance, as shown in Figures 6.9b and 6.10b, this sudden change in direction is not observed, confirming the ballistic trajectory. Moreover, the velocity does not reach zero because when the first stage impacts the ground (or ocean in this case), it is traveling at 236 m/s (850 km/h). The simulation terminates once the altitude reaches zero. The apogee is reached at a downrange distance of 321 km, while the first stage impacts the ocean at 663 km. The 3D visualization of the first stage ballistic trajectory is shown in Figure 6.11. Overall, the results look promising, signaling readiness to proceed with the recovery of the first stage.



(a) Velocity Profile and First Stage Ballistic Trajectory over Time

(b) Velocity Profile and First Stage Ballistic Trajectory Against Downrange Distance

Figure 6.9: Electron Launch Vehicle Velocity Profile and First Stage Ballistic Trajectory



(a) Altitude Profile and First Stage Ballistic Trajectory over Time

(b) Altitude Profile and First Stage Ballistic Trajectory Against Downrange Distance

Figure 6.10: Electron Launch Vehicle Velocity Profile and First Stage Ballistic Trajectory

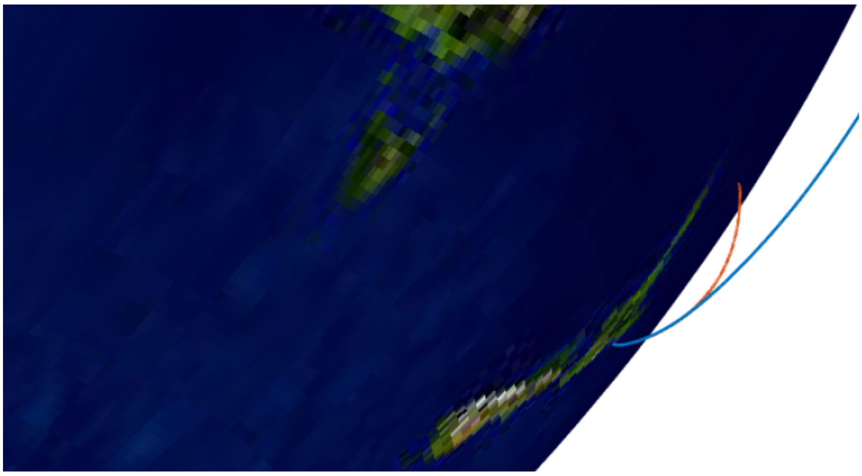


Figure 6.11: Electron Launch Vehicle Ascent and First Stage Ballistic Trajectory Visualization in 3D

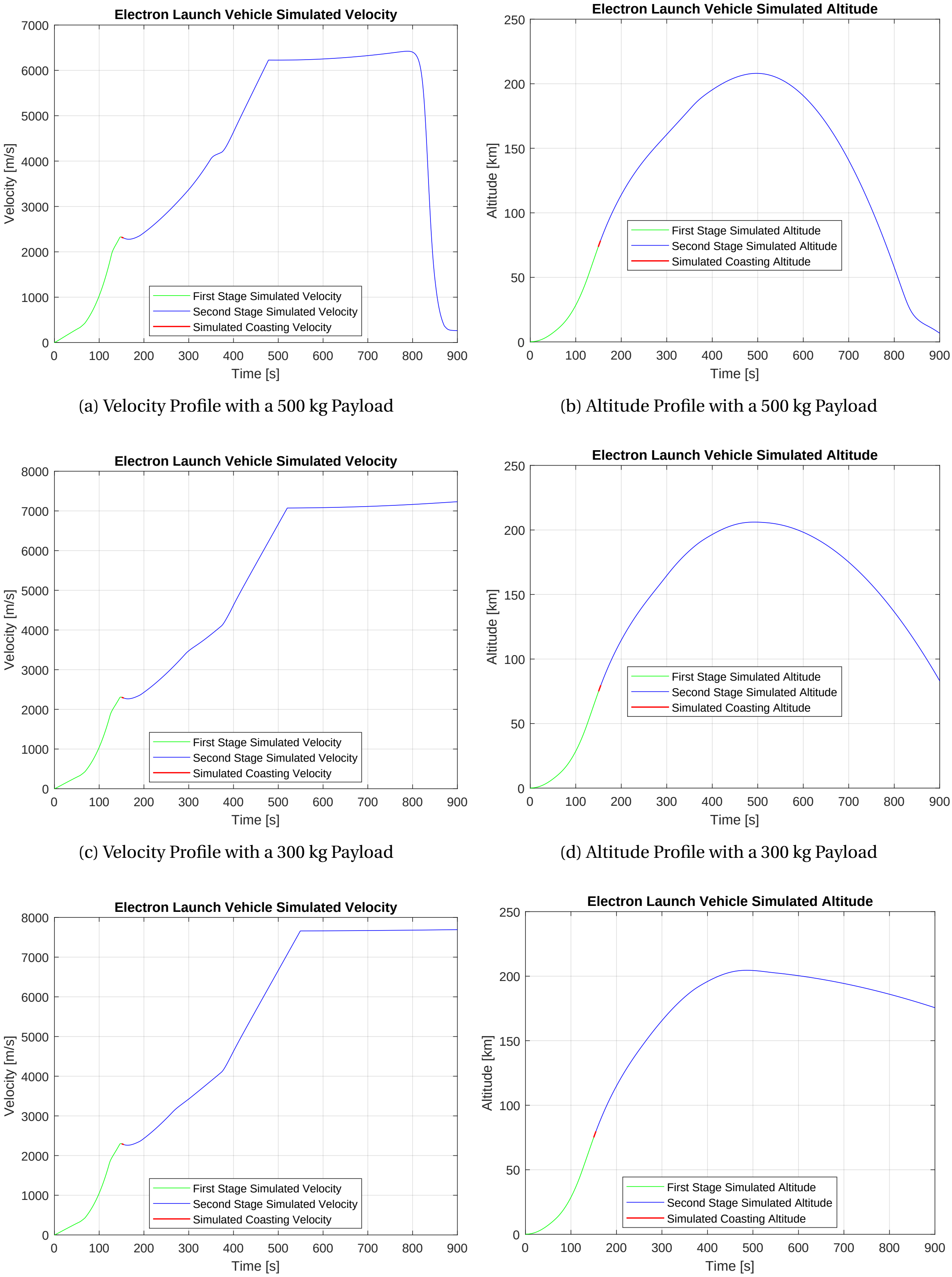
7
Sensitivity Analysis of the Ascent Phase

Two types of sensitivity analyses can be employed: Monte Carlo analysis and the one-at-a-time approach. In the one-at-a-time approach, a single parameter is varied while all other parameters remain constant. This analysis aims to determine the system's sensitivity to each individual parameter. Although this method is straightforward, it does not account for potential interactions between different parameters. For the ascent trajectory model, the important parameters are pitch profile, thrust, and payload. The one-at-a-time approach will be used for sensitivity analysis. First, while keeping the other parameters unchanged, the payload will be changed to 500 kg, 300 kg, and 200 kg. The results are shown in Figure 7.1. Figure 7.1a presents the velocity profile for a 500 kg payload. At the second stage burnout, occurring at 478 seconds, the velocity reaches 6226 m/s. The corresponding altitude at this point is 207 km, as shown in Figure 7.1b. The burnout velocity is significantly lower than the required orbital velocity of 7790 m/s. With no propellant remaining, the vehicle cannot gain additional velocity. Consequently, it begins to fall back towards Earth. The velocity initially increases due to gravitational pull but then decreases drastically as air resistance becomes too strong. The vehicle is unable to reach a stable circular orbit at an altitude of 200 km.

A similar phenomenon is observed with a 300 kg payload, although it develops more gradually, as shown in Figures 7.1c and 7.1d. At the second stage burnout, occurring at an altitude of 205 km and 520 seconds, the vehicle achieves a velocity of 7070 m/s. However, this is still below the required orbital velocity of 7790 m/s, preventing the vehicle from reaching a stable circular orbit at the targeted altitude of 200 km.

The Electron user manual indicates that the vehicle can deliver a 200 kg payload to a 500 km SSO. Unfortunately, it does not specify whether this capability applies to its expendable or reusable configuration. Figures 7.1e and 7.1f illustrate the results for a 200 kg payload. At burnout, which occurs at 549 seconds and an altitude of 202 km, the velocity reaches 7660 m/s. Despite this, the velocity remains below the required orbital velocity for achieving a stable orbit. This issue may arise from a suboptimal pitch profile and the added mass required for recovery. Since this model does not incorporate pitch profile optimization, as detailed in Section 3.10, it is not possible to definitively conclude whether a 200 kg payload can be delivered to a 500 km SSO. Addressing this question would require pitch profile optimization, which is beyond the scope of this thesis. The sensitivity analysis demonstrates that variations in payload mass have a significant effect on the vehicle's capability to reach its intended orbit. Payload mass is therefore a crucial parameter that profoundly impacts the ascent trajectory and overall mission performance

Figure 7.1 suggests that a sensitivity analysis of thrust is unnecessary. Reducing thrust will lower the velocity, thereby hindering the ability to reach the target orbit. On the other hand, increasing thrust beyond the engines' capacity is not feasible in practice. Additionally, any changes in thrust are automatically corrected by the GNC model through thrust modulation. Consequently, the focus will shift to investigating the pitch profile.



(a) Velocity Profile with a 500 kg Payload

(b) Altitude Profile with a 500 kg Payload

(c) Velocity Profile with a 300 kg Payload

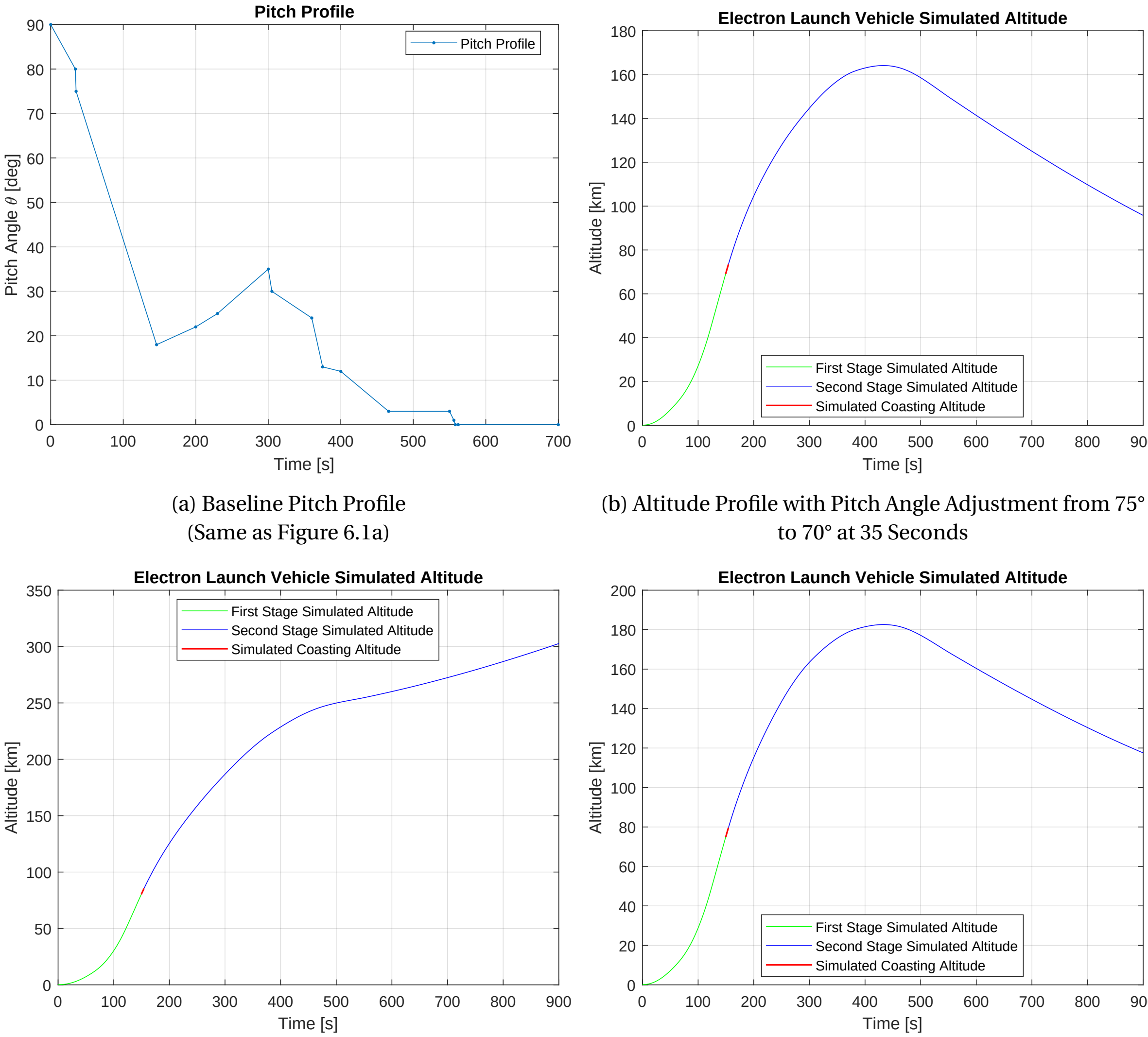
(d) Altitude Profile with a 300 kg Payload

(e) Velocity Profile with a 200 kg Payload

(f) Altitude Profile with a 200 kg Payload

Figure 7.1: Velocity and Altitude Profiles of Ascent Trajectory with Different Payloads

When the pitch profile is adjusted, the velocity profile remains largely unaffected, as the vehicle has sufficient propellant to reach the required reference velocity. However, the altitude profile exhibits a significant deviation from the target orbit. Figure 7.2 illustrates the effects of varying a single pitch angle randomly from the baseline pitch profile depicted in Figure 7.2a.



(a) Baseline Pitch Profile (Same as Figure 6.1a)

(b) Altitude Profile with Pitch Angle Adjustment from 75° to 70° at 35 Seconds

(c) Altitude Profile with Pitch Angle Adjustment from 75° to 80° at 35 Seconds

(d) Altitude Profile with Pitch Angle Adjustment from 35° to 20° at 300 Seconds

Figure 7.2: Altitude Profiles of Ascent Trajectory with Different Pitch Profiles

Consider the scenario where the pitch angle is adjusted from 75° to 70° at 35 seconds. The resulting trajectory is depicted in Figure 7.2b, showing that a 5° change in pitch angle causes the vehicle to deviate significantly from its intended path, preventing it from achieving a 200 km circular orbit. Increasing the pitch angle by 5° from the baseline profile yields the outcome shown in Figure 7.2c, where the altitude overshoots to 300 km, resulting in an unstable and non-circular orbit. Thus far, only two examples of pitch angle adjustments before the first stage separation have been analyzed. To further illustrate the system's sensitivity, consider changing the pitch angle from 35° to 20° at 300 seconds, as shown in Figure 7.2d, which causes the vehicle to descend towards Earth. These examples highlight the system's high sensitivity to pitch profile adjustments. Further tests with various pitch angles consistently reveal similar issues: unstable orbits, failure to reach the target orbit, or significant overshooting. Therefore, it can be concluded that variations in the pitch profile critically impact the vehicle's performance, underscoring its essential role in the ascent trajectory.

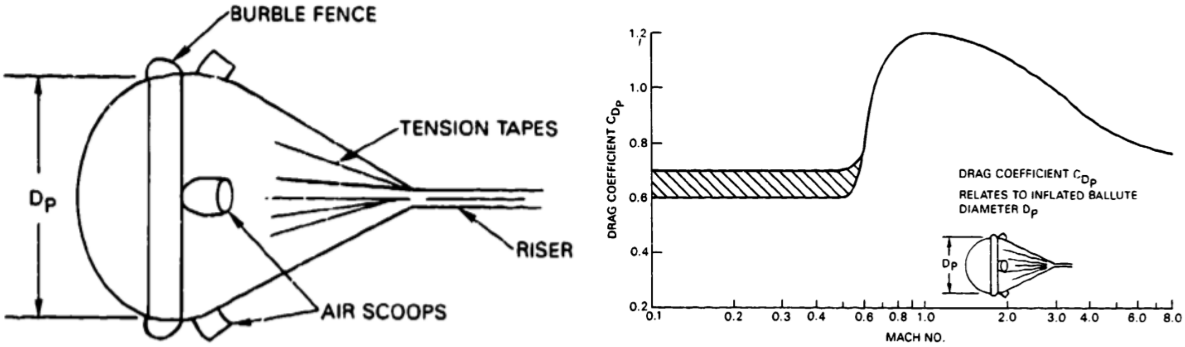
8
Descending Phase of the First Stage: Parafoil-Based Recovery

This chapter presented the descending phase and parafoil-based recovery of the first stage.

8.1. Ballute Deployment

When initiating the recovery mechanism, the vehicle's supersonic velocity must first be reduced to subsonic levels using a drogue parachute. This reduction allows for the subsequent deployment of the main parachute, whether it is a parafoil or a circular type. Unfortunately, detailed data on the recovery phase is less comprehensive compared to the ascent phase. However, insights from previous missions, such as "Baby Come Back," "There and Back Again," and "Running Out of Toes," reveal that a drogue parachute is deployed at an approximate altitude of 13 km. This drogue parachute decreases the vehicle's velocity from around Mach 2 to approximately Mach 0.3 (≈ 100 m/s) [47]. The main parachute is then deployed at around 6 km, further reducing the velocity to 10 m/s. At its peak, the dynamic pressure and temperature can reach up to 120 kPa and 2400°C, respectively [47]. Before deploying the drogue parachute, the reaction control system will reorient the first stage by 180 degrees to position it at the ideal angle for re-entry. This adjustment is crucial for withstanding the intense heat and pressure encountered during the re-entry.
No data is available regarding this ideal angle. Determining it requires extensive CFD analysis, which is beyond the scope of this thesis. Additionally, re-entry technology is subject to knowledge embargo by the United Nations and the European Union. As a result, individuals wishing to study or conduct research in this field must apply for an exemption. Without this exemption, one cannot initiate a research project or work in this area. The application review process can take up to eight weeks, and this period may be extended if a decision is not reached within that time-frame. Therefore, it has been decided to exclude re-entry GNC and aeroheating from this thesis.

The shape of the drogue parachute discussed above differs from the one used in this thesis. Therefore, the altitude at which it is deployed differs as well. The Electron rocket will employ a balloon-type aerodynamic decelerator known as a ballute to slow the vehicle's velocity. The ballute provides several advantages over traditional aerodynamic decelerators, including rapid and uniform inflation, reduced opening shock load, and exceptional stability during inflation [48]. To simplify the simulation, it is assumed that inflation happens instantaneously. Ballutes are generally deployed at higher altitudes. For Mars aerocapture missions ballutes are often deployed at around 30 km altitude [49]. For Earth re-entry and lunar missions, ballutes can be deployed in space prior to reaching the atmospheric interface and are utilized throughout the entry phase [50]. Based on the provided values, the descent velocity at an altitude of 6 km should be about 100 m/s, decreasing to approximately 10 m/s after the main parachute is deployed. Figure 8.1a depicts the shape and characteristic of the ballute more clearly [48]. Air enters the ballute through air scoops, inflating it quickly and uniformly. The drag coefficient of the Goodyear ballute (C_{D_P}) as a function of Mach number is depicted in Figure 8.1b. This figure illustrates that the ballute is optimized to generate high drag at supersonic velocities.



(a) Depiction of Inflatable Ballute

(b) Drag Coefficient as a Function of Mach Number

Figure 8.1: Illustration of Goodyear Ballute and Its Drag Coefficient [48]

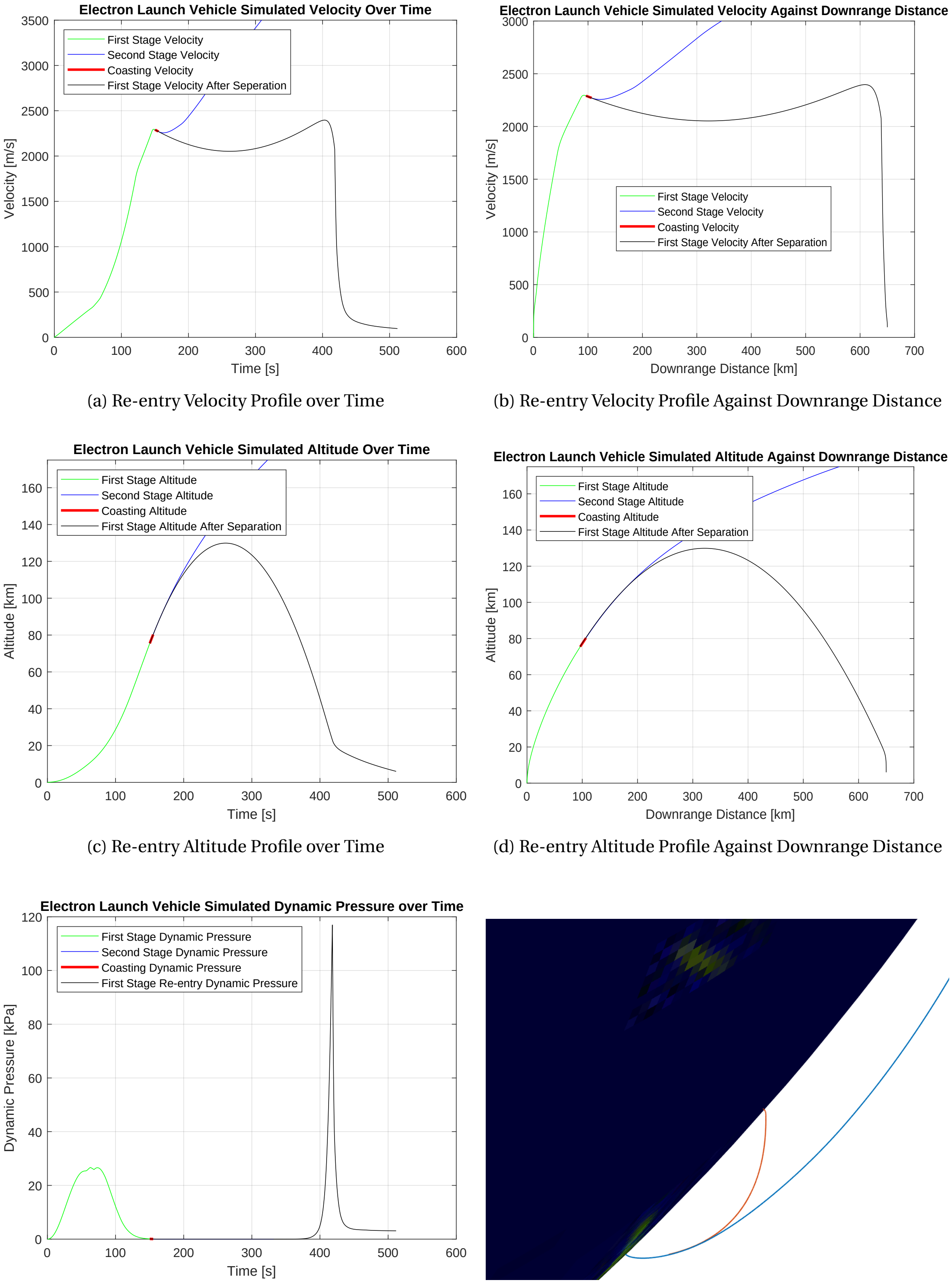
It is important to note that the drag coefficient pertains to the cross-sectional area of the inflated ballute (S_p), rather than the total surface area (S_o) as is customary with parachutes. The drag of the ballute can be calculated using Equation 8.1.

$$F_D = \frac{1}{2} c_D \rho_\infty \vec{v}_\infty^2 S_p \tag{8.1}$$

Where, ρ_∞ and $\vec{v}_\infty$ are the free stream density and velocity respectively. The drag coefficient data for the Goodyear ballute has been used to create a lookup table with Akima spline interpolation in Simulink for calculating ballute drag. For the calculation itself, the *Aerodynamic Forces and Moments* block in Simulink is utilized, which is the same block shown in Figure 3.8 in Section 3.5. The same Simulink model used for the ballistic trajectory in Section 6.4 has been adapted slightly to include ballute deployment. The Simulink architecture is provided in Chapter 9. The corresponding MATLAB functions and code are provided in the Appendix B.

By adjusting the radius of the ballute and the deployment altitude, the re-entry velocity and dynamic pressure can be controlled to meet the desired maximum dynamic pressure and velocity at an altitude of 6 km. With a radius of 1.2 m and a deployment altitude of 23 km, these requirements are satisfied. Assuming a spherical shape, a material thickness of 5 mm, and a material density of 90 kg/m³ for thermal resistant coated nylon fabric, the volume, total surface area, and mass of the ballute can be calculated as 7.24 m³, 18.2 m², and 8.14 kg, respectively.

Figure 8.2 shows the re-entry dynamic pressure, altitude and velocity profiles with the ballute deployed. At the end of the simulation, at 511.76 seconds and at an altitude of 6 km, the velocity is 96.7 m/s. The downrange distance is 650 km, consistent with the value reported by Rocket Lab [51]. The apogee altitude is 130 km. Figure 8.2e shows the dynamic pressure profile. The peak dynamic pressure recorded is 117 kPa, which is slightly lower than the actual value of 120 kPa [47]. Figure 8.2f shows a 3D visualization of the simulated ascent and re-entry trajectories from Launch Complex 1. The first-stage re-entry parameters at an altitude of 6 km will be used as input for the main parachute Simulink model, whether it is a ram-air (parafoil) or circular parachute. These parameters include the initial position in inertial axes (flat Earth reference frame), initial velocity in body axes, initial Euler angles (roll, pitch, and yaw), and initial body rotation rates (p, q, and r). These data are provided in Table D.1.



(a) Re-entry Velocity Profile over Time

(b) Re-entry Velocity Profile Against Downrange Distance

(c) Re-entry Altitude Profile over Time

(d) Re-entry Altitude Profile Against Downrange Distance

(e) Dynamic Pressure Profile During Ascent and Re-entry Trajectories

(f) 3D Visualisation of Ascent and Re-entry Trajectories

Figure 8.2: Electron Launch Vehicle Re-entry Dynamic Pressure, Velocity and Altitude Profiles with Ballute Deployment

8.2. First Stage Recovery Using a Parafoil

The following sections will describe the recovery of the first stage using a parafoil, which is assumed to be fully inflated. First, the reference frame, frame transformation, and environment will be explained. This will be followed by the dimensions, mass properties, and inertia of the parafoil-payload system, along with the governing equations of motion for controlled descent.

8.2.1. Reference Frames

This section details the reference frames used for the first stage's recovery with a parafoil.

Flat Earth Reference Frame

The flat Earth reference frame (subscript I) assumes the Earth as a flat disk and is treated as an inertial frame. The origin of the body-fixed coordinate system is located at the body's center of gravity as depicted in Figure 8.3. the body's center of gravity refers to the center of gravity of the parafoil system with payload as depicted in Figure 8.8. The applied forces are assumed to act at this center of gravity. The positive X-axis points toward true North, the positive Y-axis points toward the East, and the positive Z-axis points toward the ground. This also aligns with the NED reference frame, as detailed in Section 3.1.

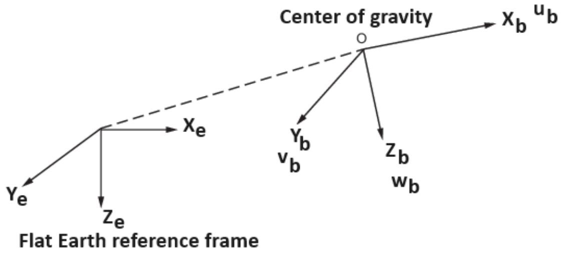


Figure 8.3: Flat Earth Reference Frame [52]

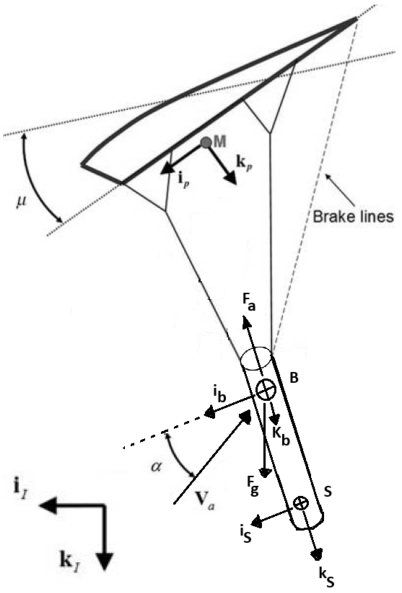
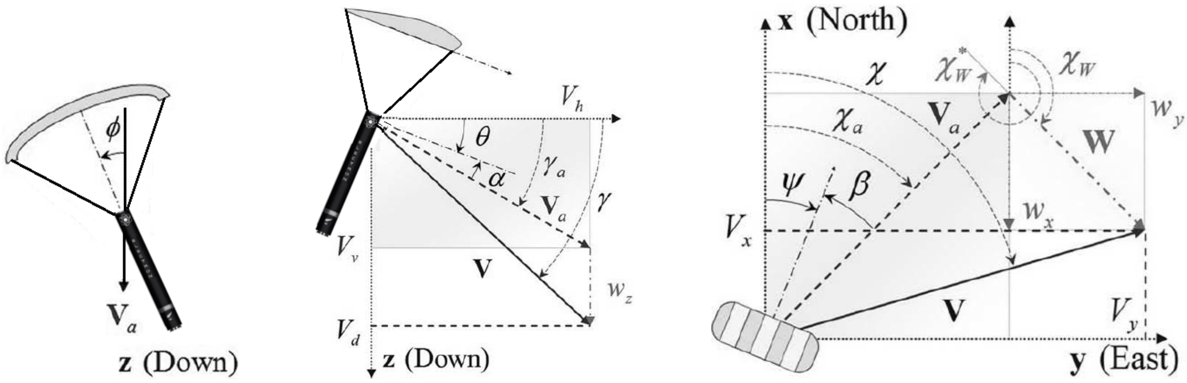


Figure 8.4: 6-DOF Parafoil-Payload System (adapted from [53])



(a) Front View

(b) Side View

(c) Bird's-Eye View

Figure 8.5: 6-DOF Parafoil-Payload System Views (adapted from [53])

The body reference frame and wind reference frame definitions, as described in Sections 3.1, remain applicable. However, to adapt these frames to the new body configuration (parafoil with payload) and to clearly illustrate its orientation, Figure 8.4 and 8.5 are provided [53]. As per the right-hand rule, the roll (ϕ), angle of attack (α), yaw (ψ), heading angle (χ_a), and ground-track angle (χ) in Figure 8.5 are positive, while the pitch (θ), flight-path angle (aka air-referenced flight-path angle) (γ_a), Earth-referenced flight path angle (γ), and sideslip angle (β) are negative. Furthermore, the wind vector (W) along with its components in Earth-reference frame is presented as well.

Body Reference Frame
The origin of the body reference frame is positioned at the center of gravity of the parafoil-payload system, denoted as B. The X_b axis is positive along the longitudinal axis of the parafoil with payload, lying in the plane of symmetry. The Z_b axis is perpendicular to the X_b axis, within the plane of symmetry, and points downward. The Y_b axis is perpendicular to the X_b–Z_b plane, with its positive direction determined by the right-hand rule, pointing to the right (East).

Wind Reference Frame
The origin of the wind reference frame is also located at the center of gravity of the parafoil and payload configuration. The X_w axis is positive in the direction of the velocity vector relative to the air, denoted as the airspeed vector $\mathbf{V}_a$ (pointing towards the the canopy's leading edge). The Z_w axis is perpendicular to the X_w axis and lies in the plane of symmetry, with the positive direction pointing downward. The Y_w axis is perpendicular to both the X_w and Z_w axes, with the positive direction determined by the right-hand rule, extending to the right. The wind frame provides a convenient reference for describing the aerodynamic forces and moments acting on the parafoil components. Specifically, the net aerodynamic force can be decomposed into components along the axes of the wind frame, with the drag force D aligned with the $-X_w$ axis and the lift force L aligned with the $-Z_w$ axis.

Canopy-Fixed Reference Frame
The canopy-fixed reference frame (subscript p) shares the same axis definitions as the body reference frame, but its origin is located at the apparent mass center, denoted as M. It is rotated relative to the body reference frame by the negative rigging angle μ, as shown in Figure 8.4.

Payload-Fixed Reference Frame
The payload-fixed reference frame (subscript s) has the same axis definitions as the body reference frame, but its origin is positioned at the payload's mass center, labeled S (see Figure 8.4).

Navigational Reference Frame
The navigational reference frame (subscript n) is parallel to the flat Earth reference frame (subscript I), with its origin positioned at the body's center of gravity.

8.3. Frame Transformations

This section elaborates on the frame transformation used for the recovery of the first stage using a parafoil. The information used for these methods is based on the work of Stevens and Lewis [22], Zipfel [23], Oleg Yakimenko [53] and AIAA recommendation [24].

8.3.1. Flat Earth to Latitude, Longitude and Altitude (LLA)

The Flat Earth to Latitude, Longitude and Altitude (LLA) block in Simulink converts flat Earth position vector ($\bar{P}$) into geodetic latitude ($\bar{\mu}$), longitude ($\bar{\iota}$), and altitude ($\bar{h}$) [22] [54]. In flat Earth reference frame, the z-axis is positive in downward direction. First, the position vector is converted to

North (N) and East (E) coordinates using Equation 8.2. In this transformation, ψ denotes the angle in degrees measured clockwise from the flat Earth x-axis to the north.

$$\begin{bmatrix} N \\ E \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} \tag{8.2}$$

The Earth's radius of curvature in the prime vertical, R_N, and in the meridian, R_M, are defined by Equations 8.3 and 8.4, respectively.

$$R_N = \frac{R}{\sqrt{1-\left(2f-f^2\right)\sin^2\mu_0}} \tag{8.3}$$

$$R_M = R_N \frac{1-\left(2f-f^2\right)}{1-\left(2f-f^2\right)\sin^2\mu_0} \tag{8.4}$$

where R is the Earth's equatorial radius and f is the flattening (or oblateness) of the Earth, which can be computed using Equation 3.4. The geodetic latitude and longitude are estimated by starting with the initial latitude (μ_0) and longitude (ι_0) and then adding the incremental changes in latitude ($\delta\mu$) and longitude ($\delta\iota$) derived from small changes in the North and East positions.

$$\mu = \mu_0 + \overbrace{\left(\frac{\delta N}{R_M}\right)}^{\delta\mu} \tag{8.5}$$

$$\iota = \iota_0 + \overbrace{\left(\frac{\delta E}{R_N\cos\mu_0}\right)}^{\delta\iota} \tag{8.6}$$

The altitude is determined by subtracting the reference height (h_{ref}) from the negative flat Earth z-axis value, as shown in Equation 8.7.

$$h = -p_z - h_{ref} \tag{8.7}$$

8.3.2. Rotation Matrix from Navigational Reference Frame to Body Reference Frame

The transformation from the navigational reference frame to the body reference frame is represented by the product of three elementary rotation matrices as $R_n^b = R_\phi R_\theta R_\psi$. Where, R_ϕ, R_θ, and R_ψ are defined by Equations 3.19, 3.20, and 3.21, respectively. Here, R_n^b is the rotation matrix from the frame indicated by the lower index to the frame indicated by the upper index. This matrix is detailed in Equation 8.8 [53].

$$R_n^b = \begin{bmatrix} \cos(\psi)\cos(\theta) & \sin(\psi)\cos(\theta) & -\sin(\theta) \\ \cos(\psi)\sin(\theta)\sin(\phi)-\sin(\psi)\cos(\phi) & \sin(\psi)\sin(\theta)\sin(\phi)+\cos(\psi)\cos(\phi) & \cos(\theta)\sin(\phi) \\ \cos(\psi)\sin(\theta)\cos(\phi)+\sin(\psi)\sin(\phi) & \sin(\psi)\sin(\theta)\cos(\phi)-\cos(\psi)\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \tag{8.8}$$

8.3.3. Rotation Matrix from Navigational Reference Frame to Wind Reference Frame

The rotation matrix that describes the transformation from the navigational reference frame to the wind reference frame is given in Equation 8.9.

$$\begin{aligned} R_n^w R = R_{\gamma_a} R_{\chi_a} &= \begin{bmatrix} \cos\left(\gamma_a\right) & 0 & -\sin\left(\gamma_a\right) \\ 0 & 1 & 0 \\ \sin\left(\gamma_a\right) & 0 & \cos\left(\gamma_a\right) \end{bmatrix} \begin{bmatrix} \cos\left(\chi_a\right) & \sin\left(\chi_a\right) & 0 \\ -\sin\left(\chi_a\right) & \cos\left(\chi_a\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\left(\gamma_a\right)\cos\left(\chi_a\right) & \cos\left(\gamma_a\right)\sin\left(\chi_a\right) & -\sin\left(\gamma_a\right) \\ -\sin\left(\chi_a\right) & \cos\left(\chi_a\right) & 0 \\ \sin\left(\gamma_a\right)\cos\left(\chi_a\right) & \sin\left(\gamma_a\right)\sin\left(\chi_a\right) & \cos\left(\gamma_a\right) \end{bmatrix} \end{aligned} \tag{8.9}$$

The angles are described in Section 8.2.1 and 8.5.6.

8.3.4. Rotation Matrix from Wind Reference Frame to Canopy-Fixed Reference Frame

The rotation matrix that describes the transformation from the wind reference frame to the canopy-fixed reference frame is detailed in Equation 8.10 [55].

$$\begin{aligned} R_w^p = R_\alpha R_\beta &= \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{bmatrix} \begin{bmatrix} \cos(\beta) & \sin(\beta) & 0 \\ -\sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos(\alpha)\cos(\beta) & \cos(\alpha)\sin(\beta) & -\sin(\alpha) \\ -\sin(\beta) & \cos(\beta) & 0 \\ \sin(\alpha)\cos(\beta) & \sin(\alpha)\sin(\beta) & \cos(\alpha) \end{bmatrix} \end{aligned} \tag{8.10}$$

The angles α and β, as defined in Section 8.2.1, can be calculated using Equations 8.59 and 8.60, respectively.

8.3.5. Rotation Matrix from Body Reference Frame to Canopy-Fixed Reference Frame

The rotation matrix from the body reference frame to the canopy-fixed reference frame is achieved through a single-axis transformation using the rigging angle μ, as detailed in Equation 8.11.

$$R_b^p = \begin{bmatrix} \cos(\mu) & 0 & -\sin(\mu) \\ 0 & 1 & 0 \\ \sin(\mu) & 0 & \cos(\mu) \end{bmatrix} \tag{8.11}$$

8.3.6. Rotation Matrix from Wind Reference Frame to Payload-Fixed Reference Frame

Since the only difference between the body frame and the payload frame is a translation along the z-axis, the rotation between them is simply the identity matrix ($R_b^S = I$). Equation 8.12 defines the rotation matrix that transforms coordinates from the wind reference frame to the payload-fixed reference frame.

$$R_w^S = R_w^p \left(R_b^p\right)^T R_b^S = R_w^p \left(R_b^p\right) = \begin{bmatrix} \cos(\alpha)\cos(\beta+\mu) & \cos(\alpha)\sin(\beta) & \cos(\alpha)\sin(\beta+\mu) \\ -\sin(\beta+\mu) & \cos(\beta) & \cos(\beta+\mu) \\ \sin(\alpha)\cos(\beta+\mu) & \sin(\alpha)\sin(\beta) & \sin(\alpha)\sin(\beta+\mu) \end{bmatrix} \tag{8.12}$$

8.4. Environment

This section describes the models used to define the environment for the descent phase trajectory simulation. The validated models, available in Simulink, include those for atmospheric conditions, gravity, wind, wind turbulence, and wind gusts. Brief descriptions of each model are provided in the following subsections.

8.4.1. Atmospheric Model

The **COESA 1976 Atmosphere Model**, as outlined in Section 3.4.1, is also applied during the descent phase of the trajectory to accurately describe the atmospheric conditions.

8.4.2. Gravity Model

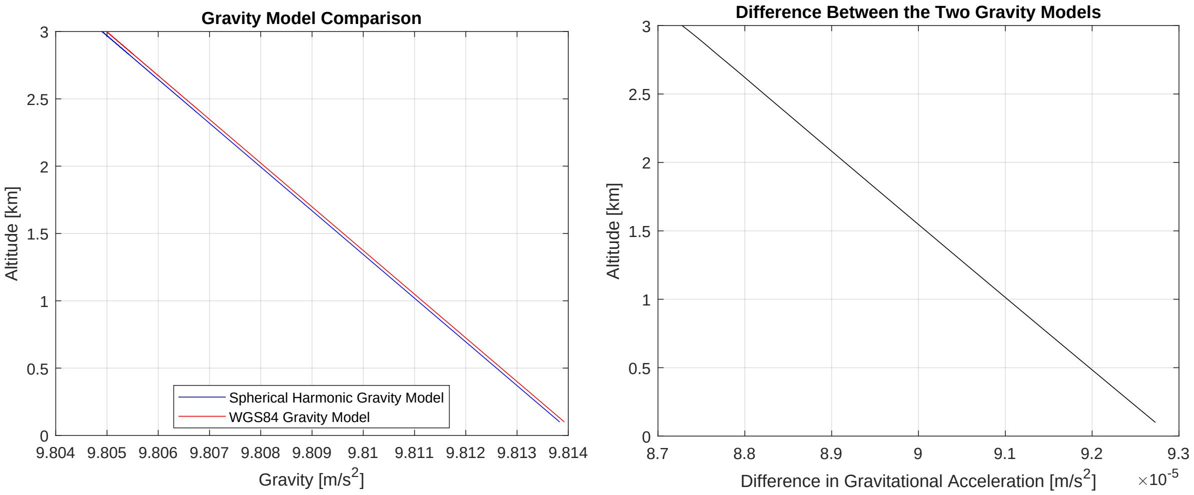
During the ascent phase of the trajectory, the Spherical Harmonics Gravity Model, as described in Section 3.4.2, was employed. However, since this model operates in the Earth-centered, Earth-fixed (ECEF) reference frame, it is not directly applicable to the descent phase. The descent trajectory model, which uses Euler angles in its 6-DOF equations of motion, employs a flat Earth reference frame, as explained in Section 8.6. Two approaches can be used to address this issue: coordinate transformation to adapt the Spherical Harmonic Gravity Model for the flat Earth reference model, or the use of an alternative gravity model. Both approaches are explored to determine their effectiveness and identify the most suitable solution.

The first approach involves converting the flat Earth coordinates to latitude, longitude, and altitude (LLA) using the Simulink *Flat Earth to LLA* block, as detailed in Section 8.3.1. These LLA coordinates are then transformed into Earth-Centered, Earth-Fixed coordinates using the Simulink *LLA to ECEF Position* block, as described in Section 3.3.1. The resulting ECEF coordinates are fed into the Spherical Harmonic Gravity Model, which outputs an array of gravity vectors in the ECEF frame. To convert the gravity vector from the ECEF frame to the North-East-Down (NED) frame, Equation 8.13 is applied:

$$g_{NED} = DCM_{\text{ef}} \cdot g_f \tag{8.13}$$

where DCM_{ef} is the Direction Cosine Matrix that transforms coordinates from the ECEF frame to the NED frame, and g_f represents the gravity vector in the ECEF frame. The DCM_{ef} matrix can be obtained using the Simulink *Direction Cosine Matrix ECEF to NED* block, as outlined in Section 3.3.3. This sequence ensures that the gravity vector is accurately represented in the NED frame, which is typically more convenient for local navigation and simulation tasks.

The **WGS84 Gravity Model** can serve as an alternative gravity model. This model uses the 1984 World Geodetic System (WGS84) to represent Earth's gravity. It is defined as a geocentric equipotential ellipsoid, meaning the gravity potential is assumed equal at every point on the ellipsoid [56]. This model takes geodetic latitude, longitude, and altitude as input and provides gravity values in the NED reference frame. The precision of the gravity calculation can be adjusted by selecting from three methods: WGS84 Taylor Series, WGS84 Close Approximation, or WGS84 Exact. Higher precision methods requires increased computational power and time. When using the WGS84 Close Approximation or WGS84 Exact methods and disregarding the effects of atmospheric mass and centrifugal forces, the gravity result in the NED reference frame closely aligns with that of the Spherical Harmonic Gravity Model as seen in Figure 8.6.



(a) WGS84 Gravity Model vs. Spherical Harmonic Gravity Model

(b) Difference Between WGS84 Gravity Model and Spherical Harmonic Gravity Model

Figure 8.6: Comparison and Difference Between WGS84 Gravity Model and Spherical Harmonic Gravity Model

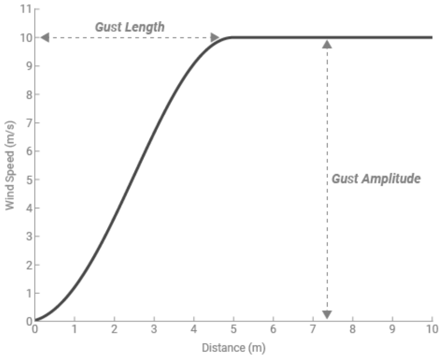
The difference in gravitational acceleration between the two models is approximately $9 \times 10^{-5}\,\text{m/s}^2$, which is considered negligible for this application. For efficiency, iterations during the simulation are carried out with the WGS84 Taylor Series method, while the final results are obtained using the WGS84 Close Approximation method. Although the WGS84 Exact method would provide the same results, it is less efficient due to its higher computational costs. Both gravity models are retained in the descent trajectory model, available for future students to use if needed for their projects.

8.4.3. Horizontal Wind Model

Wind can significantly influence the descent trajectory. Therefore, an appropriate wind model is necessary for the simulation. Fortunately, the Horizontal Wind Model is available in Simulink [57]. The Horizontal Wind Model calculates the wind velocity in the body reference frame. The wind speed and direction, which can be constant or variable over time, are defined in Earth axes. The direction of the wind is specified as the direction from which the wind is coming, measured clockwise from the Earth's x-axis (north) in degrees. The wind velocities are then transformed into body-axes coordinates using the direction cosine matrix from flat Earth reference frame to body reference frame. The model takes as inputs the direction cosine matrix, wind speed, and wind direction, and provides the wind velocity in the body reference frame.

8.4.4. Discrete Wind Gust Model

The Discrete Wind Gust Model uses a mathematical function to represent a smooth, bell-shaped wind gust, commonly referred to as the standard '1-cosine' shape, as specified in Military Specification MIL-F-8785C [58]. The gust can be applied to each axis individually or simultaneously to all three axes. Equation 8.14 presents the mathematical representation for the discrete gust (V_{wind}). The gust start time, length (d_m), and amplitude (V_m) are parameters that can be specified within the model, with x denoting the traveled distance. The gust length denotes the distance over which the gust develops, while the gust amplitude represents the wind speed produced by the gust, as illustrated in Figure 8.7 [59]. The model takes airspeed as input and generates discrete wind gusts in the form of wind velocity in body reference frame.



$$V_{\text{wind}} = \begin{cases} 0 & x < 0 \\ \frac{V_m}{2}\left(1 - \cos\left(\frac{\pi x}{d_m}\right)\right) & 0 \leq x \leq d_m \\ V_m & x > d_m \end{cases} \tag{8.14}$$

Figure 8.7: Shape of the Gust with Governing Parameters [59]

8.4.5. Wind Turbulence Model

The Von Karman Wind Turbulence Model (Continuous) introduces turbulence into the descending main parachute of Electron's first stage by filtering band-limited white noise through filters based on the Von Karman spectral representation, creating the desired turbulence characteristics as specified in U.S. military specifications [58] [60] [61]. The model uses altitude, vehicle velocity, and the Direction Cosine Matrix (which transforms from the flat Earth reference frame to the body reference frame) as inputs. It outputs turbulence velocities and angular rates, both in the body reference frame [62].

8.5. Parafoil-Payload System

The next step involves developing a comprehensive 6-DoF model for the main parachute. The first scenario examines an air-ram parachute or parafoil. Figure 8.8 illustrates a parafoil-payload system. Once fully inflated, the parafoil canopy retains a fixed shape, with the exception of the adjustable brakes. Deformation of the canopy due to aerodynamic forces and weight is neglected. It is also assumed that the deflection of the parafoil brakes does not affect the aerodynamic coefficients or alter the shape of the parafoil, apart from the brakes themselves. Additionally, the inflation process is assumed to be instantaneous, meaning that the parafoil is considered fully inflated when it is deployed. Rather than estimating the parafoil's weight through complex mathematical relations, detailed drawings, or CFD analysis, which would be time-consuming, the weight and dimensions are instead based on commercially available, off-the-shelf parafoils.

The FireFly is a cargo delivery parafoil system with a payload capacity sufficient to recover the weight of the Electron's first stage (see Figure 8.9). The specifications of the FireFly parafoil system are provided in Table 8.1 [63]. The parafoil is assumed to have the same mass and dimensions as the FireFly. Although the specific increase in the first stage's mass isn't explicitly detailed in the literature, it is noted that the interstage was reinforced, and a custom parachute attachment and release mechanism was added [47]. Given these modifications, the total mass of the recovery system, including the ballute, parafoil, and airborne guidance unit, is estimated to be approximately 100 kg. Of this, 74 kg is allocated for the parafoil, 9 kg for the ballute, with the remainder for the aforementioned modifications.

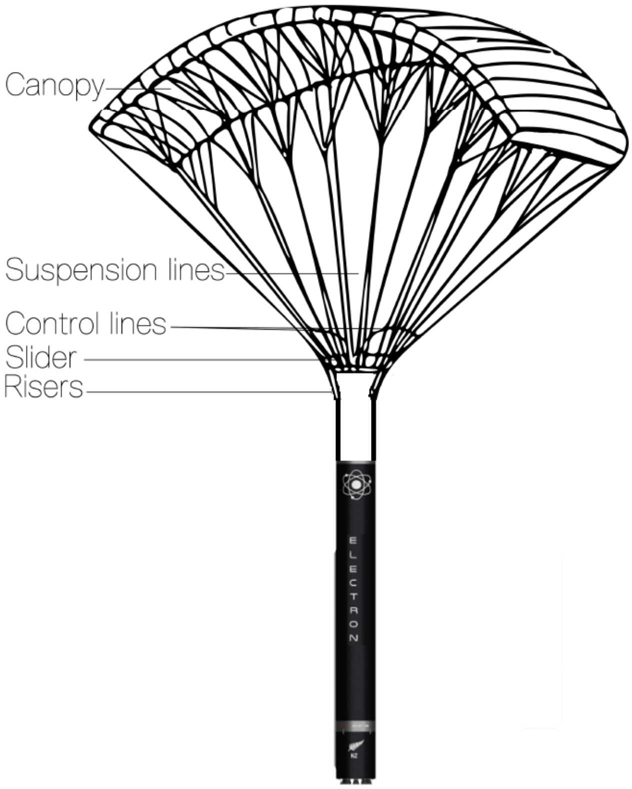


Figure 8.8: Depiction of the Parafoil with the First Stage (adapted from [64])

Table 8.1: Specifications of FireFly Parafoil [63]

Specification	Value
Maximum Gross Rigged Weight	1,089 kg
System Mass	73.5 kg
Span	17.1 m
Surface Area	95.2 m^2
Chord	5.5 m
Cell Count	19
Maximum Release Altitudes	7,468 m
Minimum Release Altitudes	1,524 m
Max Glide (L/D, No Wind)	3.25 : 1

Figure 8.9: FireFly Parafoil [63]

8.5.1. Apparent Mass and Inertia Tensors for the Parafoil

As the parafoil moves through the fluid (air), it causes the fluid to flow around it, generating a pressure field known as apparent mass pressure. This effect can be modeled as added mass and inertia tensors. The geometric representation of the parafoil, apparent masses and inertias are presented in Figure 8.10, 8.11 and 8.12. The apparent masses and inertias operate within the canopy-fixed reference frame (subscript p), as detailed in Section 8.2.1. The resulting apparent forces and moments act at the apparent mass center M, as illustrated in Figure 8.4.

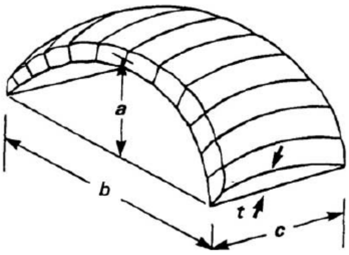


Figure 8.10: Parafoil Geometry [65]

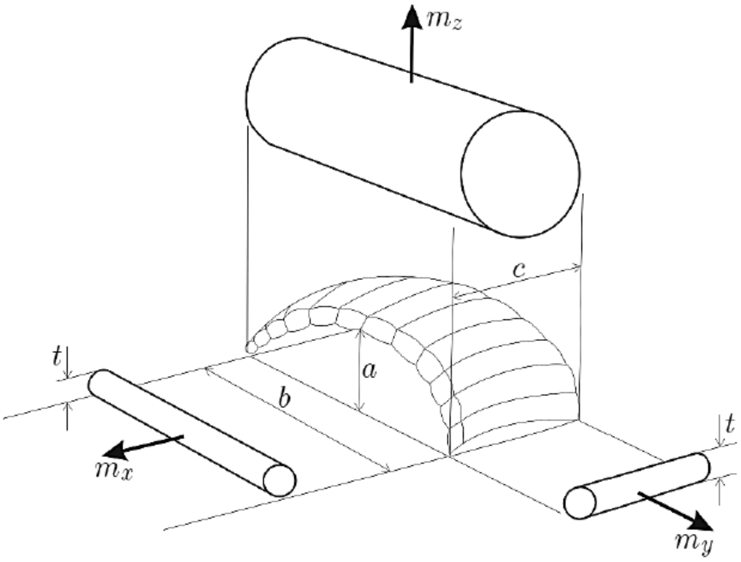


Figure 8.11: Apparent Masses for Translational Motion [65]

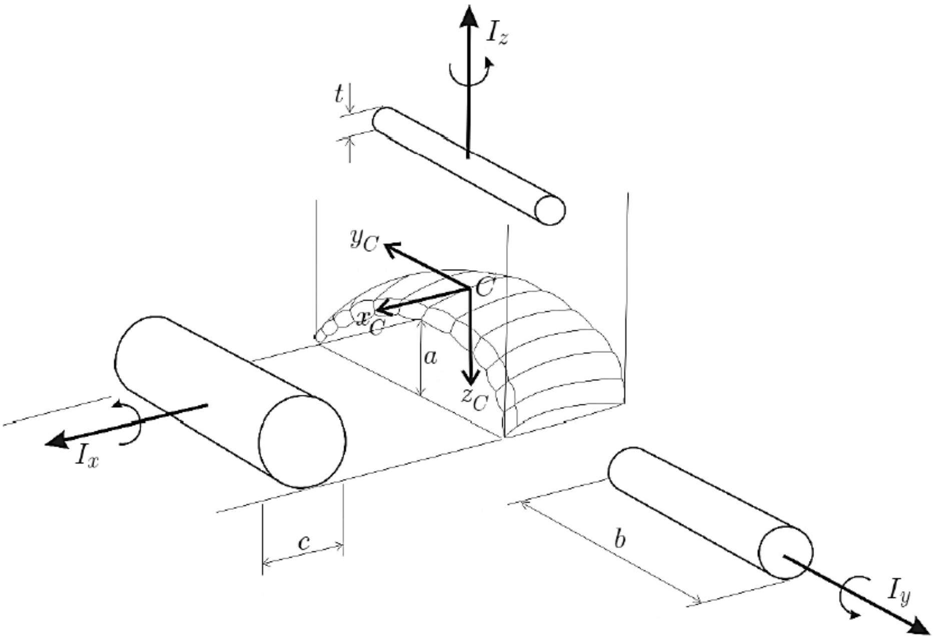


Figure 8.12: Apparent Masses for Rotational Motion [65]

The apparent masses and inertias can be determined using CFD analysis. However, a more straightforward and simpler approach is to calculate the apparent masses ($\boldsymbol{M_f}$) and inertias ($\boldsymbol{I_f}$) with the following mathematical equations [65].

$$\boldsymbol{M_f} = \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} \tag{8.15}$$

$$\boldsymbol{I_f} = \begin{bmatrix} I_A & 0 & 0 \\ 0 & I_B & 0 \\ 0 & 0 & I_C \end{bmatrix} \tag{8.16}$$

Where A, B, and C represent the apparent masses in the x, y, and z directions, respectively. These are denoted as m_x, m_y, and m_z in Figure 8.11. Similarly, I_A, I_B, and I_C represent the apparent inertias in the x, y, and z directions, respectively, and are denoted as I_x, I_y, and I_z in Figure 8.12. The parameters A, B, C, I_A, I_B, and I_C can be estimated using the following equation [65].

$$A = 0.666 \cdot \rho \left(1 + \frac{8}{3} a^{*2}\right) \cdot t^2 \cdot b \tag{8.17}$$

$$B = 0.267 \cdot \rho \cdot \left(1 + 2 \cdot \frac{a^{*2}}{t^{*2}} \cdot AR \cdot \left(1 - t^{*2}\right)\right) \cdot t^2 \cdot c \tag{8.18}$$

$$C = 0.785 \cdot \rho \cdot \sqrt{1 + 2 \cdot a^{*2} \cdot \left(1 - t^{*2}\right)} \cdot \frac{AR}{1 + AR} \cdot c^2 \cdot b \tag{8.19}$$

$$I_A = 0.055 \cdot \rho \cdot \frac{AR}{1 + AR} \cdot c^2 \cdot b^3 \tag{8.20}$$

$$I_B = 0.0308 \cdot \rho \cdot \frac{AR}{1 + AR} \cdot \left[1 + \frac{\pi}{6} \cdot (1 + AR) \cdot AR \cdot a^{*2} \cdot t^{*2}\right] \cdot c^4 \cdot b \tag{8.21}$$

$$I_C = 0.0555 \cdot \rho \cdot \left(1 + 8 \cdot a^{*2}\right) \cdot t^2 \cdot b^3 \tag{8.22}$$

With:

$$AR = \frac{b}{c} \quad , t^* = \frac{t}{c} \quad \text{and} \quad a^* = \frac{a}{b} \tag{8.23}$$

where a is the height of the parafoil canopy, b is the canopy span, c is the canopy chord length, t is the canopy thickness, and AR is the aspect ratio.

8.5.2. Geometry and Dimensions of the Parafoil-Payload System

During this preliminary design, the geometrical characteristics of the parafoil-payload system are estimated based on the data in Table 8.1. The mathematical expressions for these estimations are summarized in Table 8.2. Any remaining parameters will be derived using the mathematical formulations and geometrical relationships provided in the following equations.

Table 8.2: Estimation of Parafoil Geometrical Characteristics [53]

Parameter (with units)	Formula / Definition
Line diameter [m]	$D_{\text{line}} = 0.0025$
Aspect ratio [-]	$AR = \frac{b}{c}$
Canopy mean line length or suspension mean line length [m]	$R = 0.8 \cdot b$
Parafoil canopy mean thickness [m]	$t = 0.1 \cdot c$
Parafoil canopy height [m]	$a = \frac{b \cdot t}{c}$
Anhedral angle [deg]	$\epsilon = \frac{b}{2R}$
Harness length [m]	$l_{\text{ha}} = 0.45 \cdot R$
Inlet height of canopy [m]	$h = 0.14 \cdot c$
Canopy volume [m³]	$v_{\text{vol}} = 0.09 \cdot b \cdot c^2$
Apparent thickness of parafoil [m]	$h_{\text{mean}} = \frac{v_{\text{vol}}}{b \cdot c} \approx \frac{v_{\text{vol}}}{S_P}$

In the table above, S_P stands for the canopy area. For calculating the apparent thickness of the parafoil (h_{mean}), both the parafoil mean thickness (t) and the parafoil chamber are taken into account. It is assumed that the parafoil is modeled as a parallelepiped with dimensions b, c, and h_{mean}. The parafoil profile is designed with an inlet that facilitates the inflation of the canopy. This inlet captures a mass of air known as added mass within the ram-air, and the resulting drag is referred to as inlet drag. The added mass is determined using Equation 8.24.

$$m_e = 0.09 \cdot b \cdot c^2 \cdot \rho \tag{8.24}$$

where, ρ is the air density. Calculating the mass moments of inertia requires defining the dimensions of the parafoil-payload system. Relevant specifications and equations are provided in Tables 8.1 and 8.2, while Figure 8.13 illustrates the system's geometry and dimensions.

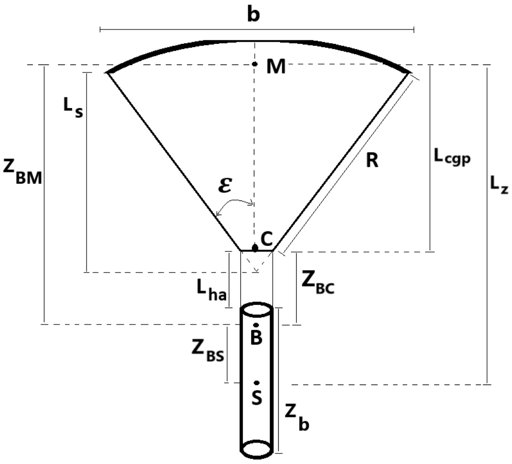


Figure 8.13: Schematic of Parafoil-Payload System Geometry

The vectors from the body center of mass (B) to the parafoil center of mass (M), the confluence point (C), and the payload center of mass (S) are given by Equations 8.25, 8.26, and 8.27, respectively.

$$\mathbf{r}_{BM} = \begin{bmatrix} 0 \\ 0 \\ z_{BM} \end{bmatrix} \tag{8.25}$$

$$\mathbf{r}_{BC} = \begin{bmatrix} 0 \\ 0 \\ z_{BC} \end{bmatrix} \tag{8.26}$$

$$\mathbf{r}_{BS} = \begin{bmatrix} 0 \\ 0 \\ z_{BS} \end{bmatrix} \tag{8.27}$$

The distances from the body center of mass (B) to the payload center of mass (S), the parafoil center of mass (M), and the confluence point (C) along the z-axis are given by Equations 8.28, 8.29, and 8.30, respectively.

$$z_{BS} = L_z \cdot \frac{m_{\text{par}} + m_e}{m_{\text{par}} + m_{\text{s}}} \tag{8.28}$$

$$z_{BM} = z_{BS} - L_z \tag{8.29}$$

$$z_{BC} = L_z - L_{c_{gp}} - z_{BS} \tag{8.30}$$

where, m_{par} is the mass of the parafoil system, m_s is the payload mass (i.e., the empty mass of the first stage), m_e is the added mass, and l_z is the distance between the parafoil center of mass (M) and the payload center of mass (S), as defined by Equation 8.31.

$$L_z = \frac{z_b}{2} + L_{ha} + L_{c_{gp}} \tag{8.31}$$

where, z_b represents the payload length (i.e., the length of the first stage), L_{ha} is the harness length, as defined in Table 8.2, and L_{cgp} is the distance between the parafoil center of mass (M) and the confluence point (C), as defined by Equation 8.32 [55].

$$L_{c_{gp}} = \frac{2}{3} \cdot \frac{(L_s + h_{\text{mean}})^3 - L_s^3}{(L_s + h_{\text{mean}})^2 - L_s^2} \cdot \frac{\sin(\epsilon)}{\epsilon} \tag{8.32}$$

Where, h_{mean} and ϵ denote the apparent thickness of the parafoil and the canopy anhedral angle, respectively, as defined in Table 8.2. L_s is defined by Equation 8.33, where R represents the mean suspension line length, as specified in Table 8.2.

$$L_s = R \cdot \frac{\sin(\epsilon)}{\epsilon} \tag{8.33}$$

8.5.3. Skew-Symmetric Matrices

In Equation 8.65, the cross-product term arises from angular velocities about the two axes perpendicular to the given axis. In three-dimensional space, the cross-product operation can be represented using skew-symmetric matrices. Therefore, the cross product can be expressed as the multiplication of a vector by a skew-symmetric matrix, which is constructed from the components of the same vector. The skew-symmetric matrix that replaces the vector product of the parafoil-payload system's rate is as follows [53]:

$$\mathbf{S}(\omega) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \tag{8.34}$$

The skew-symmetric matrices that replace the vector products of $\mathbf{r_{BS}}$ and $\mathbf{r_{BM}}$ are provided by Equations 8.35 and 8.36.

$$\mathbf{S}(\mathbf{r_{BS}}) = \begin{bmatrix} 0 & -z_{BS} & y_{BS} \\ z_{BS} & 0 & -x_{BS} \\ -y_{BS} & x_{BS} & 0 \end{bmatrix} \tag{8.35}$$

$$\mathbf{S}(\mathbf{r_{BM}}) = \begin{bmatrix} 0 & -z_{BM} & y_{BM} \\ z_{BM} & 0 & -x_{BM} \\ -y_{BM} & x_{BM} & 0 \end{bmatrix} \tag{8.36}$$

8.5.4. Mass Moments of Inertia of the Parafoil-Payload System

The mass moments of inertia of the parafoil about the x-, y-, and z-axes are given by equations 8.37, 8.38, and 8.39, respectively and are combined in Equation 8.40 [66].

$$I_{x_p} = \frac{m_{\text{par}} + m_e}{12} \left(b^2 + h_{\text{mean}}^2\right) \tag{8.37}$$

$$I_{y_p} = \frac{m_{\text{par}} + m_e}{12} \left(c^2 + h_{\text{mean}}^2\right) \tag{8.38}$$

$$I_{z_p} = \frac{m_{\text{par}} + m_e}{12} \left(b^2 + c^2\right) \tag{8.39}$$

$$I_{\text{par}_p} = \begin{bmatrix} I_{x_p} & 0 & 0 \\ 0 & I_{y_p} & 0 \\ 0 & 0 & I_{z_p} \end{bmatrix} \tag{8.40}$$

The parafoil is defined in the canopy-fixed frame (p) and must be transformed to the body frame (b). Equation 8.11 provides the orthogonal rotation matrix (R_b^p) from body to canopy. Since the inverse of an orthogonal matrix equals its transpose, the transpose of this matrix transforms from canopy to body frame. Due to the x_b-z_b symmetry plane of the parafoil-payload system, the inertia matrix contains only four unique components. The following equation calculates the moment of inertia tensor for the parafoil in the body frame [53].

$$\begin{aligned} I_{\text{par}_b} = I_{\text{par}_p} \cdot R_b^{pT} &= \begin{bmatrix} I_{x_p}\cos^2(\mu) + I_{z_p}\sin^2(\mu) & 0 & \frac{1}{2}(I_{x_p} - I_{z_p})\sin(2\mu) \\ 0 & I_{y_p} & 0 \\ \frac{1}{2}(I_{z_p} - I_{x_p})\sin(2\mu) & 0 & I_{z_p}\cos^2(\mu) + I_{x_p}\sin^2(\mu) \end{bmatrix} \\ &= \begin{bmatrix} I_{x_{pb}} & 0 & I_{xz_{pb}} \\ 0 & I_{y_{pb}} & 0 \\ I_{zx_{pb}} & 0 & I_{z_{pb}} \end{bmatrix} \end{aligned} \tag{8.41}$$

To calculate the mass moment of inertia of the payload, the empty first stage of the Electron is modeled as a thick-walled cylindrical tube with both ends open. It has mass m_s, inner and outer radii r_1 and r_2, uniform mass distribution, and length L as depicted in Figure 8.14. The wall thickness is assumed to be 5 mm. Equation 8.42 is used to calculate the mass moment of inertia of the payload. The diagonal terms, from left to right, represent the mass moments of inertia about the x-, y-, and z-axes, denoted as I_{x_s}, I_{y_s}, and I_{z_s}, respectively.

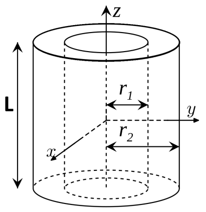


Figure 8.14: Schematic of the Thick-Walled Cylindrical Tube Representing the Electron's First Stage

$$I_s = \begin{bmatrix} \frac{1}{12}m_s\left(3\left(r_2^2 + r_1^2\right) + L^2\right) & 0 & 0 \\ 0 & \frac{1}{12}m_s\left(3\left(r_2^2 + r_1^2\right) + L^2\right) & 0 \\ 0 & 0 & \frac{1}{2}m_s\left(r_2^2 + r_1^2\right) \end{bmatrix} = \begin{bmatrix} I_{x_s} & 0 & 0 \\ 0 & I_{y_s} & 0 \\ 0 & 0 & I_{z_s} \end{bmatrix} \tag{8.42}$$

Thus far, the moment of inertia tensors for the parafoil and the payload have been calculated separately. Next, the moment of inertia tensor for the **entire system**, comprising the parafoil-payload assembly, will be computed in the body reference frame using the parallel axis theorem [55].

$$\begin{aligned} I_{\text{sys}} &= \begin{bmatrix} I_{x_{pb}} + (m_{\text{par}} + m_e)\cdot z_{BM}^2 + I_{x_s} + m_s \cdot z_{BS}^2 & 0 & I_{xz_{pb}} \\ 0 & I_{y_{pb}} + (m_{\text{par}} + m_e)\cdot z_{BM}^2 + I_{y_s} + m_s \cdot z_{BS}^2 & 0 \\ I_{zx_{pb}} & 0 & I_{z_{pb}} + I_{z_s} \end{bmatrix} \\ &= \begin{bmatrix} I_x & 0 & I_{xz} \\ 0 & I_y & 0 \\ I_{xz} & 0 & I_z \end{bmatrix} \end{aligned} \tag{8.43}$$

8.5.5. Aerodynamic Coefficients of the Parafoil-Payload System

The aerodynamic coefficients of the parafoil can be estimated through wind tunnel experiments or CFD analysis. Both approaches are resource-intensive and time-consuming. As a result, the dimensionless aerodynamic coefficients will be derived from existing data available in the literature. A significant amount of data is available for parafoils with the commonly used Clark-Y airfoil profile (see Figure 8.15). The dimensionless aerodynamic coefficient for the parafoil, based on the Snowflake parafoil from Table 5.3 of *Precision Aerial Delivery Systems: Modeling, Dynamics, and Control*, is shown in Table 8.3 [53]. The payload drag coefficient is taken from the cylinder drag coefficient lookup table in the *Applied Fluid Dynamics Handbook* [67].

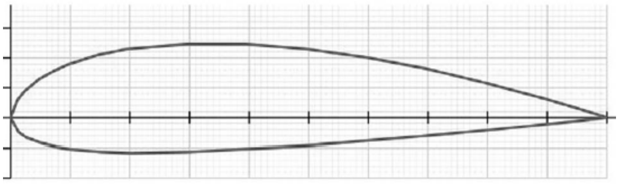


Figure 8.15: Clark-Y Airfoil Profile [53]

Table 8.3: Aerodynamic Coefficients for the Parafoil [53] [67]

Parameter	Value [-]	Description
C_{D_0}	0.25	Drag coefficient at zero lift
$C_{D_{\alpha^2}}$	0.12	Proportionality factor for induced drag
$C_{D_{\delta_s}}$	0.30	Drag coefficient due to asymmetric control deflection
C_{Y_β}	-0.23	Side-force coefficient due to sideslip
C_{L_0}	0.091	Lift coefficient at zero angle of attack
C_{L_α}	0.90	Lift-curve slope coefficient versus angle of attack
$C_{L_{\delta_s}}$	0.21	Lift coefficient due to symmetric control deflection
C_{m_0}	0.35	Pitching-moment coefficient at zero lift
C_{m_α}	-0.72	Pitching-moment coefficient due to angle of attack
C_{m_q}	-1.49	Pitch-damping coefficient due to pitch rate
C_{l_β}	-0.036	Rolling-moment coefficient due to sideslip
C_{l_p}	-0.84	Rolling-damping coefficient due to roll rate
C_{l_r}	-0.082	Rolling-damping coefficient due to yaw rate
$C_{l_{\delta_\alpha}}$	-0.0035	Rolling-moment coefficient due to asymmetric control deflection
C_{n_β}	-0.0015	Yawing-moment coefficient due to sideslip
C_{n_p}	-0.082	Yawing-moment coefficient due to roll rate
C_{n_r}	-0.27	Yawing-moment coefficient due to yaw rate
$C_{n_{\delta_\alpha}}$	0.0215	Yawing-moment coefficient due to control deflection
C_{D_S}	0.82	Payload drag coefficient

Based on the provided data, simplified panel methods can be used to derive mathematical expressions for aerodynamic coefficients, yielding a consistent set that varies with angle of attack (α) and airspeed [53] [68]. The mathematical expression for the lift coefficient is given by Equation 8.44.

$$C_L = C_{L_0} + C_{L_\alpha} \cdot \alpha + C_{L_{\bar{\delta}_S}} \cdot \bar{\delta}_S \tag{8.44}$$

The drag coefficient is given in Equation 8.45.

$$C_D = C_{D_0} + C_{D_{\alpha^2}} \cdot \alpha^2 + C_{D_{\bar{\delta}_S}} \cdot \bar{\delta}_S \tag{8.45}$$

To account for the various drag components associated with the parafoil system, additional drag coefficient values must be added to Equation 8.45 afterwards. These include the inlet drag, which is associated with the trapped air inside the parafoil and can be calculated using Equation 8.46.

$$C_{D_{\text{inlet}}} = 0.5 \cdot \frac{h}{c} \tag{8.46}$$

where, h is the inlet height of canopy and c is the canopy chord length. The inlet drag is approximately 39% of the total drag. Furthermore, an additional 30% of drag is added to Equation 8.45 to account for the contribution of the parafoil suspension lines, along with an extra 2.2% drag to account for parafoil roughness [53]. The side force, axial force, and normal force coefficients are determined using equations 8.47, 8.48, and 8.49, respectively [53].

$$C_Y = C_{Y_\beta} \cdot \beta \tag{8.47}$$

$$C_X = \frac{-C_D \cdot u_p + C_L \cdot w_p}{|V_p|} \tag{8.48}$$

$$C_Z = \frac{-C_D \cdot w_p - C_L \cdot u_p}{|V_p|} \tag{8.49}$$

In the equations above, u_p and w_p represent the components of the parafoil velocity (V_p) along the x- and z-axes, respectively, as defined in the canopy reference frame. The rolling-moment, pitching-moment, and yawing-moment coefficients are given by equations 8.50, 8.51, and 8.52, respectively.

$$C_l = C_{l_\beta} \cdot \beta + \frac{b}{2 \cdot |V_p|} \cdot C_{l_p} \cdot p + \frac{b}{2 \cdot |V_p|} \cdot C_{l_r} \cdot r + C_{l_{\delta_\alpha}} \cdot \bar{\delta}_\alpha \tag{8.50}$$

$$C_m = C_{m_0} + C_{m_\alpha} \cdot \alpha + \frac{c}{2 \cdot |V_p|} \cdot C_{m_q} \cdot q \tag{8.51}$$

$$C_n = C_{n_\beta} \cdot \beta + \frac{b}{2 \cdot |V_p|} \cdot C_{n_p} \cdot p + \frac{b}{2 \cdot |V_p|} \cdot C_{n_r} \cdot r + C_{n_{\delta_\alpha}} \cdot \bar{\delta}_\alpha \tag{8.52}$$

Where, V_P, α, and β are defined by Equations 8.57, 8.59, and 8.60, respectively. $\bar{\delta}_S$ and $\bar{\delta}_\alpha$ represent the normalized symmetric and asymmetric control deflections of the trailing edge, given by equations 8.53 and 8.54, respectively.

$$\bar{\delta}_s = \frac{\delta_s}{\delta_{s_{max}}} \quad \in \quad [0;1] \tag{8.53}$$

$$\bar{\delta}_a = \frac{\delta_a}{\delta_{s_{\max}}} \quad \in \quad [-1;1] \tag{8.54}$$

8.5.6. Computation of Aerodynamic Angles and Velocities

The *Incidence, Sideslip, & Airspeed block* in Simulink is used to calculate the total airspeed and the angles between the velocity vector and the body. These angles are described in Section 8.2.1. The body velocity in the body frame is given by Equation 8.66, where u_b, v_b, and w_b are the components of the velocity vector V_b along the x-, y-, and z-axes, respectively. The magnitude of the velocity vector in the body frame follows from Equation 8.55.

$$|V_b| = \sqrt{u_b^2 + v_b^2 + w_b^2} \tag{8.55}$$

The airspeed vector V_a is computed using Equation 8.56 [69].

$$V_a = V_b - R_n^b \cdot V_{\text{wind}} \tag{8.56}$$

Here, R_n^b represents the rotation matrix from the navigational reference frame to the body reference frame, as given by Equation 8.8, and V_{wind} denotes the wind velocity vector. The parafoil velocity vector, V_p, is determined using Equation 8.57 [53].

$$V_p = (R_b^p)^\top \left(V_a + \mathbf{S}(\omega) \cdot \mathbf{r}_{BM}\right) \tag{8.57}$$

Where R_b^P is the rotation matrix from the body reference frame to the canopy reference frame, as given by Equation 8.11. $\mathbf{S}(\omega)$ is the skew-symmetric matrix representing the rate of the parafoil-payload system, as defined by Equation 8.34. $\mathbf{r}_{BM}$ is the vector from the body center of mass (B) to the parafoil center of mass (M), as given by Equation 8.25. The magnitude of the parafoil velocity vector in the canopy frame follows from Equation 8.58.

$$|V_P| = \sqrt{u_P^2 + v_P^2 + w_P^2} \tag{8.58}$$

The incidence angle (also known as the angle of attack) is calculated using Equation 8.59.

$$\alpha = \tan^{-1}\left(\frac{w_P}{u_P}\right) \tag{8.59}$$

The sideslip angle is calculated using Equation 8.60.

$$\beta = \sin^{-1}\left(\frac{v_P}{|V_P|}\right) = \tan^{-1}\left(\frac{v_P}{\sqrt{u_P^2 + w_P^2}}\right) \tag{8.60}$$

The payload velocity vector, V_s, is determined using Equation 8.61 [53].

$$V_s = (V_a + \mathbf{S}(\omega) \cdot \mathbf{r}_{BC}) \tag{8.61}$$

where, $\mathbf{r}_{BC}$ is the vector from the body center of mass (B) to the confluence point (C), as given by Equation 8.26. To compute the flight path angle and heading angle, the airspeed vector V_a, initially expressed in the body frame, must be transformed into the navigational frame. The navigational frame is defined with respect to the NED frame. The rotation matrix R_n^b (see Equation 8.8) defines the transformation from the navigational reference frame to the body reference frame. Being an orthogonal matrix, R_n^b has orthonormal unit vectors as its rows and columns. Since R_n^b is orthogonal, its inverse is equal to its transpose. As a result, the airspeed vector V_a in the NED frame can be expressed as:

$$V_a^{\text{NED}} = (R_n^b)^T V_a \tag{8.62}$$

The magnitude of V_a remains invariant under rotation. Thus, the magnitude of V_a remains unchanged when it is rotated from the body frame to the NED frame. The flight-path angle γ_a is determined using Equation 8.63.

$$\gamma_a = \arcsin\left(\frac{v_z}{|V_a|}\right) \tag{8.63}$$

Where, v_z represents the descent rate or sink rate (the vertical component of airspeed in NED frame). Since V_a already accounts for wind velocity, there is no need to add it again to v_z. The heading angle χ_a is determined using Equation 8.64.

$$\chi_a = \text{atan2}(v_y, v_x) \tag{8.64}$$

Here, v_y and v_x represent the eastward and northward components of the airspeed vector in the NED frame (V_a^{NED}).

8.6. Equations of Motion for Parafoil-Based Descent Systems

This section derives the equations of motion for the descent phase of the first stage and its recovery using a parafoil

8.6.1. Equation Of Motion

In Section 3.7, the ascent phase equations of motion were derived using the quaternion representation of the 6-DOF equations for a variable mass in the ECEF reference frame. For the descending phase of the Electron's first stage, the Euler angle representation of the 6-DOF equations of motion for a custom variable mass will be implemented. The primary reason for using Euler angles is that they are more intuitive and easier to visualize and understand compared to quaternions. They provide a direct representation of orientation in terms of rotation about the primary axes. Since the Euler angles are not expected to reach ±90 degrees, gimbal lock is not a significant concern anymore. The body is assumed to be rigid, with applied external forces acting at its center of gravity. The origin of the body reference frame is fixed in the center of gravity of the body. Furthermore, the Earth is assumed to be flat and non-rotating, and the flat Earth reference frame is considered to be inertial. Therefore, only the rotation of the body reference frame (body-fixed frame) relative to the flat Earth reference frame (X_e, Y_e, Z_e) is taken into account. The Earth's polar motion, nutation, and precession are neglected, along with forces such as centrifugal and Coriolis forces. The **translational equation of motion** of the body (fixed) reference frame is described by the Equation 8.65 [52].

$$\bar{F}_b = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m\left(\dot{\bar{V}}_b + \bar{\omega}_b \times \bar{V}_b\right) + \dot{m}\bar{V}re_b \tag{8.65}$$

Where:
$\left[F_x \;\; F_y \;\; F_z\right]^T$ are the applied external forces, given in the body reference frame. These forces are defined in Section 8.6.2. $\bar{V}re_b$ is the relative velocity in the body reference frame, where the mass flow $(\dot{m})$ is expelled from the body in the body reference frame. $\bar{V}_b$ is the velocity of the body expressed in the body reference frame. It is defined by the Equation 8.66.

$$\bar{V}_b = \begin{bmatrix} u_b \\ v_b \\ w_b \end{bmatrix} \tag{8.66}$$

$\bar{\omega}_\mathrm{b}$ represents the body's angular rates expressed in the body reference frame. It is described by the Equation 8.67.

$$\bar{\omega}_b = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{8.67}$$

The accelerations of the body expressed in the body reference frame are provided by Equation 8.68.

$$A_{bb} = \begin{bmatrix} \dot{u}_b \\ \dot{v}_b \\ \dot{w}_b \end{bmatrix} = \frac{\bar{F}_b - \dot{m}\bar{V}_{re_b}}{m} - \bar{\omega}_b \times \bar{V}_b \tag{8.68}$$

The accelerations of the body relative to flat Earth reference frame (inertial) are provided by the Equation 8.69.

$$A_{be} = \frac{\bar{F}_b - \dot{m}\bar{V}_{re_b}}{m} \tag{8.69}$$

The **equation of rotational motion** of the body, defined in the body reference frame, is described by the Equation 8.70 [52].

$$\bar{M}_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = I\dot{\bar{\omega}}_b + \bar{\omega}_b \times (I\bar{\omega}_b) + \dot{I}\bar{\omega}_b \tag{8.70}$$

Where:
$[L \;\; M \;\; N]^T$ represents the applied external moments given in the body reference frame with respect

to the origin O within the same reference frame. These moments are defined in Section 8.6.2. The inertia tensor (I) with respect to the origin O can be calculated with the Equation 8.71 [52].

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix} \tag{8.71}$$

Due to the x_b-z_b symmetry plane of the parafoil-payload system, the inertia matrix contains only four unique components. As a result, Equation 8.71 simplifies to Equation 8.43, which represents the inertia tensor of the entire system. The inertia tensor's rate of change is defined by the Equation 8.72 [53].

$$\dot{I} = \begin{bmatrix} \dot{I}_x & 0 & \dot{I}_{xz} \\ 0 & \dot{I}_y & 0 \\ \dot{I}_{xz} & 0 & \dot{I}_z \end{bmatrix} \tag{8.72}$$

The rate of change of the Euler angles, $[\dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]^{\mathrm{T}}$ is defined by the Equation 8.73 [53] [52].

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & (\sin\phi\tan\theta) & (\cos\phi\tan\theta) \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{8.73}$$

The rigid body translational kinematics can be defined as follow [53]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = {}^{b}_{n}R^{T} \begin{bmatrix} u_b \\ v_b \\ w_b \end{bmatrix} \quad \text{or} \quad \dot{p} = {}^{b}_{n}R^{T}\, v_b \tag{8.74}$$

Where the components of the groundspeed vector, defined in the body frame, are projected onto the axes of the inertial frame using the transpose of the rotation matrix as defined in Equation 8.8.

8.6.2. External Forces and Moments

In Section 8.6.1, the applied external forces and moments were discussed during the derivation of the equation of motion. This section will now provide a detailed elaboration on the equations governing these external forces and moments. The external forces and moments acting on the parafoil and payload arise from aerodynamic loads, gravitational forces, and the canopy's apparent mass. The gravitational forces acting on the entire system, expressed in the body frame, are detailed in Equation 8.75 [53].

$$\mathbf{F}_g = \overbrace{(m_{\text{par}} + m_{\text{s}} + m_e)}^{m} \cdot g \begin{bmatrix} -\sin(\theta) \\ \cos(\theta)\sin(\phi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} \tag{8.75}$$

Where, m represents the mass of the entire system, and g denotes Earth's gravitational acceleration. The added mass (m_e) generates a small upward force known as the buoyancy force. Due to its distance from the system's center of gravity and its effect on the total moments, this force must be taken into account. The buoyancy force acting on the parafoil, expressed in the body frame, is represented by Equation 8.76 [55].

$$\mathbf{F}_b^P = -m_e \cdot g \begin{bmatrix} -\sin(\theta) \\ \cos(\theta)\sin(\phi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} \tag{8.76}$$

The aerodynamic forces acting on the payload, expressed in the body frame, are defined by Equation 8.77 [55].

$$\mathbf{F}_a^S = \frac{1}{2} \cdot \rho \cdot |V_S|^2 \cdot S_S \cdot R_w^S \begin{bmatrix} -C_{D_S} \\ 0 \\ 0 \end{bmatrix} \tag{8.77}$$

where, ρ denotes the air density, V_S is the airspeed vector experienced by the payload, as defined by Equation 8.61. S_S is the surface area of the payload (i.e., the first stage), and R_w^S is the rotation matrix from the wind reference frame to the payload-fixed reference frame, as defined by Equation 8.12.
The aerodynamic forces acting on the parafoil, expressed in the body frame, are given by Equation 8.78 [68].

$$\mathbf{F}_a^P = \frac{1}{2} \cdot \rho \cdot |V_P|^2 \cdot S_P \cdot R_w^P \cdot (R_b^P)^\top \begin{bmatrix} C_X \\ C_Y \\ C_Z \end{bmatrix} \tag{8.78}$$

Here, V_P denotes the airspeed vector experienced by the parafoil at the apparent mass center M, as defined in Equation 8.57. S_P represents the surface area of the canopy, and R_w^P is the rotation matrix that transforms coordinates from the wind reference frame to the canopy-fixed reference frame, as defined in Equation 8.10. R_b^P is the rotation matrix that converts coordinates from the body reference frame to the canopy-fixed reference frame, as detailed in Equation 8.11. The coefficients C_Y, C_X, and C_Z, representing the side force, axial force, and normal force, respectively and are determined using Equations 8.47, 8.48, and 8.49. The aerodynamic moments acting on the parafoil, expressed in the body frame, are given by Equation 8.79 [68].

$$\mathbf{M}_a^P = \frac{1}{2} \cdot \rho \cdot |V_P|^2 \cdot S_P \cdot (R_b^P)^\top \begin{bmatrix} b \cdot C_l \\ c \cdot C_m \\ b \cdot C_n \end{bmatrix} \tag{8.79}$$

Here, b and c denote the parafoil's span and chord, respectively. The coefficients C_l, C_m, and C_n represent the rolling-moment, pitching-moment, and yawing-moment coefficients, as defined in Equations 8.50, 8.51, and 8.52 respectively. The moment due to the buoyancy force acting on the parafoil, expressed in the body frame, is given by Equation 8.76.

$$\mathbf{M}_b^P = S(r_{BM}) \cdot F_b^P \tag{8.80}$$

Where $S(r_{BM})$ is the skew-symmetric matrix that replaces the vector cross product of r_{BM}, as defined by Equation 8.36. The buoyancy force, F_b^P, is described by Equation 8.76.
After describing the individual forces and moments, the total external force and moment can now be presented. These forces and moments are utilized in the Simulink 6-DOF EOM, as described by equations 8.65 and 8.70. The total external forces are represented by $\mathbf{B_1}$, while the external moments are denoted by $\mathbf{B_2}$ [53] [68].

$$\begin{aligned} \mathbf{B_1} = F_a^P + F_b^P + F_a^S + F_g - m \cdot S(\omega) \cdot \begin{bmatrix} u \\ v \\ w \end{bmatrix} - S(\omega) \cdot M_f' \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - S(r_{BM}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right) \\ + S(\omega) \cdot M_f' \cdot (R_n^b)^\top \cdot V_{\text{wind}} \end{aligned} \tag{8.81}$$

$$\begin{aligned} \mathbf{B_2} = M_a^P + M_b^P + S(r_{BS}) \cdot F_a^S + S(r_{BM}) \cdot F_a^P - S(\omega) \cdot I \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix} - S(\omega) \cdot I_f' \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix} \\ -S(r_{BM}) \cdot S(\omega) \cdot M_f' \cdot \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - S(r_{BM}) \cdot \begin{bmatrix} p \\ q \\ r \end{bmatrix} - \left(R_n^b\right)^\top \cdot V_{\text{wind}} \right) \end{aligned} \tag{8.82}$$

Where,

$$m = m_{\text{par}} + m_{\text{s}} + m_e \tag{8.83}$$

$$M_f' = (R_b^P)^\top \cdot M_f \cdot R_b^P \tag{8.84}$$

$$I'_f = (R_b^P)^\top \cdot I_f \cdot R_b^P \tag{8.85}$$

The velocity specified in the above equations corresponds to the body velocity in the body frame. M_f and I_f represent the parafoil's apparent mass and inertia, as detailed in Equations 8.15 and 8.16, respectively. M'_f and I'_f represent the apparent mass and inertia tensors after being rotated by the rigging angle μ. I is the moment of inertia tensor for the entire system, which includes the parafoil-payload assembly, as described by Equation 8.43.

8.7. Parafoil Guidance, Navigation and Control Model

During the ascent phase, a 2-DOF PID controller block was used, as represented by Equation 3.97. For the parafoil-based descent, a similar controller will be employed, but without the integral component, leaving only the Proportional and Derivative (PD) terms. The primary objective of this controller is to maintain a straight descent trajectory as much as possible. The specific direction of flight is not critical, as the helicopter pilot will maneuver the helicopter to intercept and retrieve the parafoil during descent, as explained in Section 2.2.2. The GNC model for parafoil descent is shown in Figure 8.16. To ensure a straight and stable descent trajectory, the reference (desired) yaw angle is set to zero, which simplifies the pilot's task of intercepting the parafoil. As outlined in Equation 8.54, the normalized asymmetric control deflection of the trailing edge, $\bar{\delta}_a$, is constrained within the range of [-1, 1]. Next, the controller output (normalized asymmetric deflection) is fed into the dynamic model of the parafoil-payload system, along with environmental data and disturbance factors, as outlined in Section 8.4. These inputs are then used to generate the simulation results. The simulated yaw angle (referred to as the actual yaw angle) is subsequently compared with the desired yaw angle to generate new controller output, and so on.

In the PD controller, the P term corrects the error based on its current value, while the D term predicts future error by evaluating its rate of change, thus helping to prevent overshooting. Based on observation, the I term is omitted to avoid windup. Typically, the I term eliminates steady-state error by accumulating past errors over time. However, in the case of windup, the accumulation of errors leads to excessive correction, resulting in slow system response. This causes the system to overshoot and become unstable.

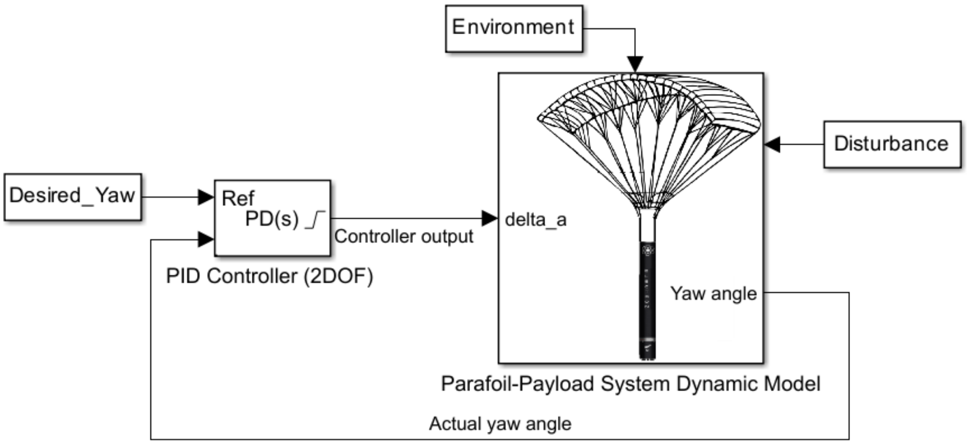


Figure 8.16: GNC System for Parafoil Descent

8.8. Verification and Validation

Now that all the models have been described and are ready for simulation, it is crucial to ensure that they undergo both verification and validation. Verification ensures that the simulation model accurately represents the physical model, while validation confirms that the simulation results reliably reflect the real-world problem being modeled. To validate the descent trajectory of the Electron rocket, it is essential to compare simulation results with actual flight data. Unfortunately, Rocket Lab does not publicly share detailed flight data from its mid-air recovery missions. The only available information is that the velocity at the end of the parafoil deployment, just before midair recovery by a helicopter, is 10 m/s, as explained in Section 8.1. Based on the *Catch Me If You Can* and *There and Back Again* missions, the parafoil's airborne duration is estimated to be approximately 10 minutes. This information is available on Rocket Lab's website; however, it only includes event timings and does not provide data on velocity, altitude, or pitch profiles. Additionally, I was unable to verify this information using the live-stream footage, as the video did not display the relevant details. As a result, direct validation of the simulation results is not feasible. While there is data available for parafoil-based aerial delivery systems, the GNC models used in these systems are significantly different. Their approach is more complex, as it involves maneuvering to a specific landing zone. This process includes several phases: a loitering phase and an energy management phase to reduce altitude and energy, followed by a homing phase (in a straight line) and a terminal phase, which involves turning to align with the landing zone and performing a flaring maneuver as depicted in Figure 8.17 [53]. Due to these substantial differences, the available data is not directly applicable to this preliminary study.However, the mathematical equations used in the models covering the environment, aerodynamic forces and moments have been verified and align with those of the precision aerial delivery system [53] [66] [55]. Additionally, the environment models are validated and available in Simulink. Therefore, in the absence of real flight data, this approach is considered the best available option and is assumed to be valid for the purposes of this study.

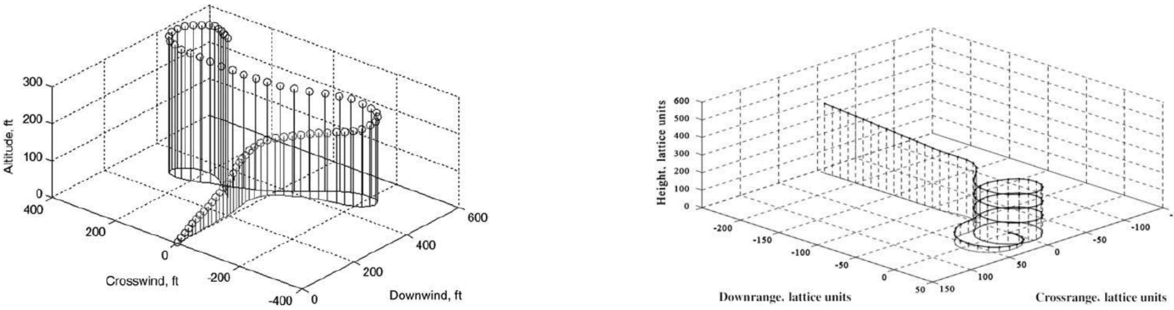


Figure 8.17: Example Trajectories of a Precision Aerial Delivery System [53]

9
Simulink Architecture for Ballute Deployment and Parafoil-Payload System Descent Trajectory Model

This chapter presents the Simulink architecture for ballute deployment and parafoil-payload system descent trajectory model. The model comprises many subsystems, each of which will be presented in detail. The corresponding MATLAB code for each subsystem is provided in Appendix B.

9.1. Simulink Architecture for Ballute Deployment

The same Simulink model used for the ballistic trajectory in Section 6.4 has been slightly adapted to include ballute deployment.

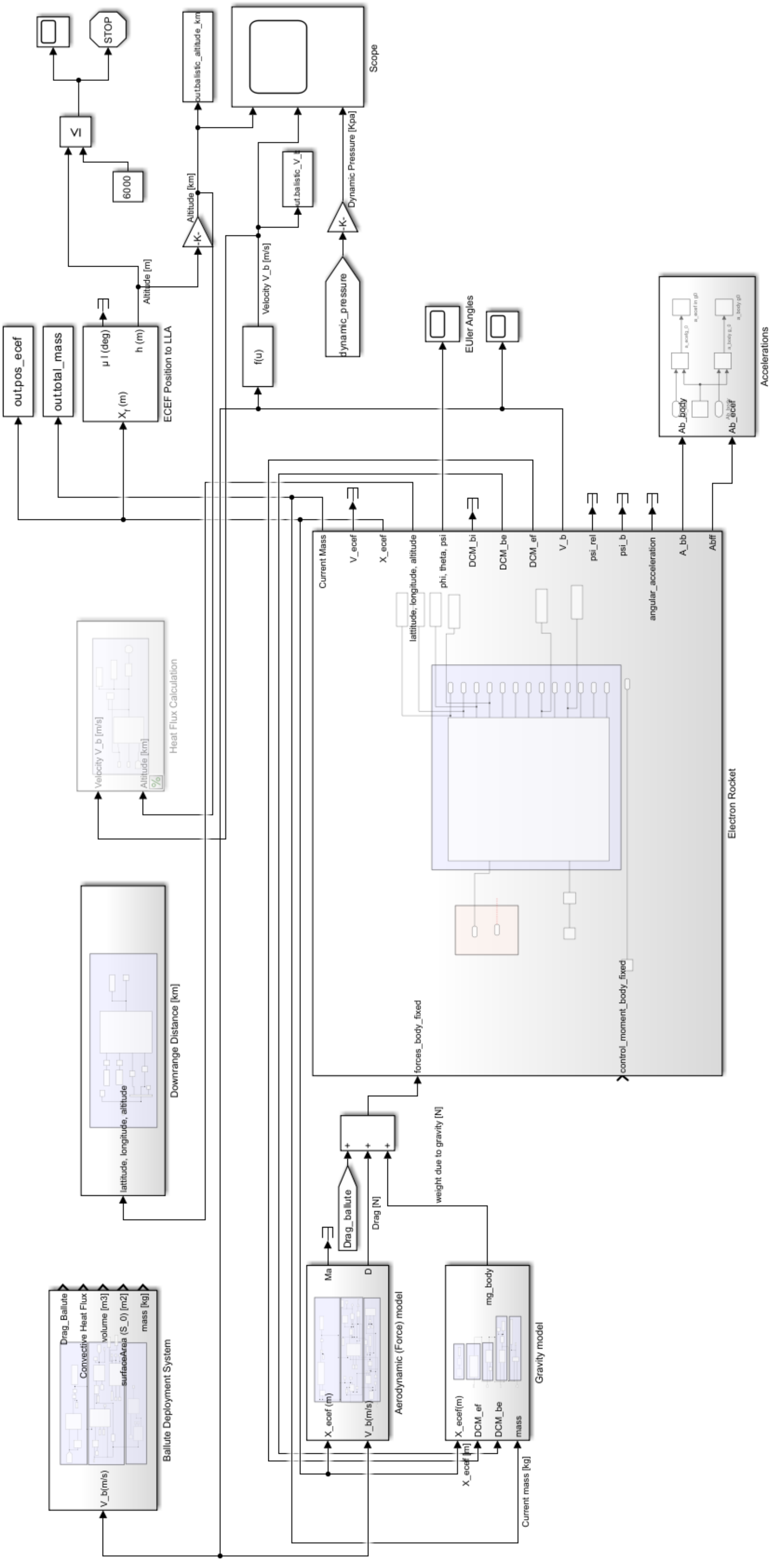


Figure 9.1: Simulink Re-entry Trajectory Model Overview

The subsystem for ballute deployment is shown in Figure 9.2. It takes altitude, velocity, mach number, ballute radius and density as inputs and calculates the ballute drag, surface area, volume and mass.

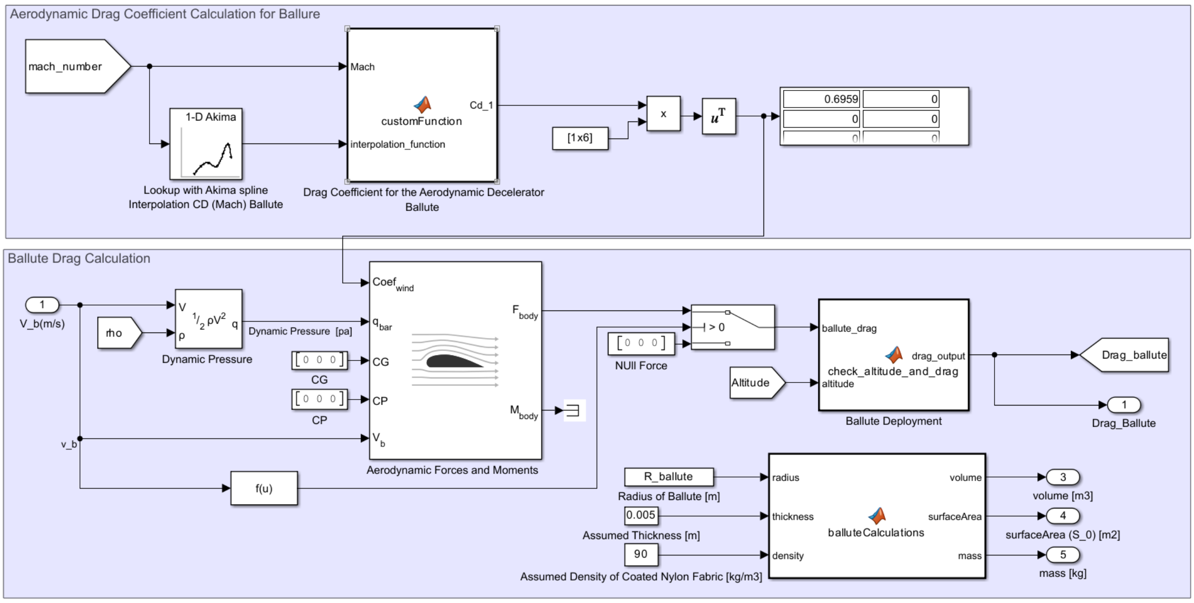


Figure 9.2: Ballute Deployment System

9.2. Simulink Architecture for Parafoil-Payload System Descent Model

Figure 9.3 presents an overview of the Simulink architecture for the parafoil-payload system descent model. Detailed overviews of each subsystem will also be presented.

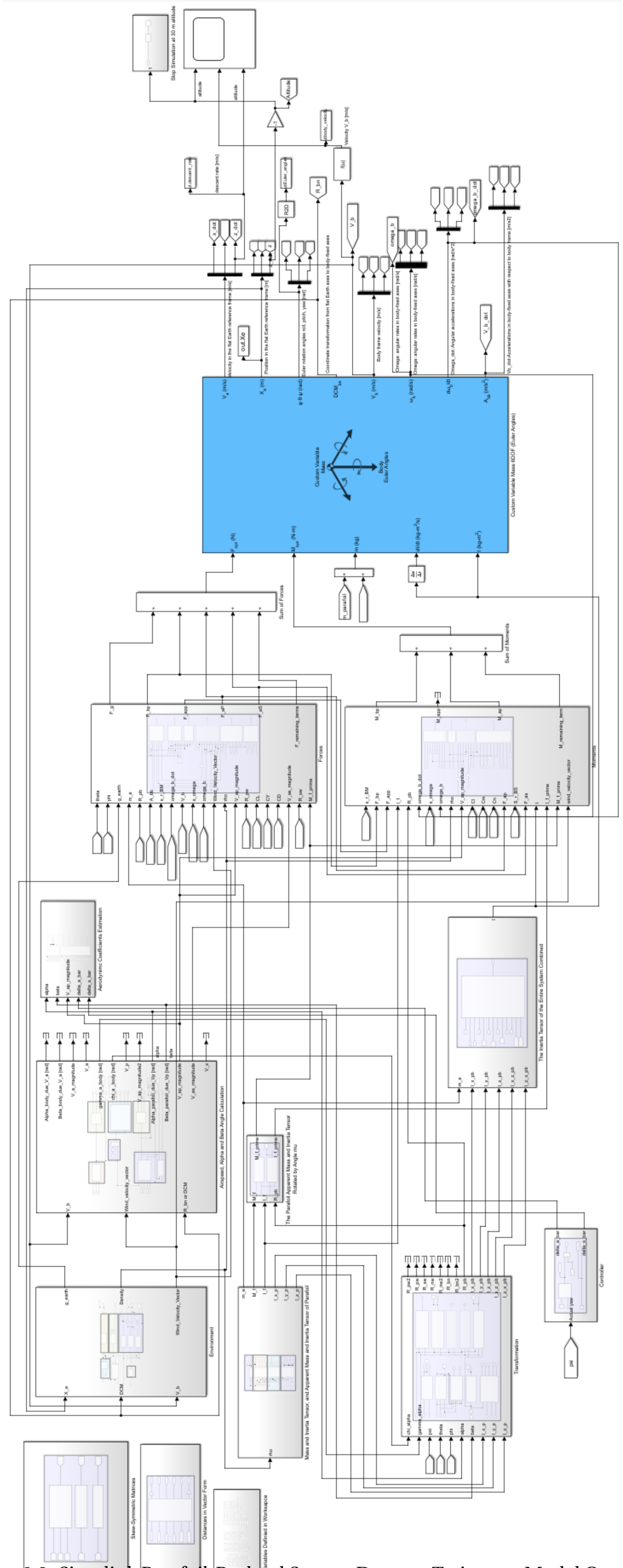

Stop Simulation at 30 m altitude
Custom Variable Mass 6DOF (Euler Angles)
Sum of Forces
Sum of Moments
Forces
Moments
Aerodynamic Coefficients Estimation
Airspeed, Alpha and Beta Angle Calculation
The Inertia Tensor of the Entire System Combined
The Parafoil Apparent Mass and Inertia Tensor Rotated by Angle mu
Environment
Mass and Inertia Tensor, and Apparent Mass and Inertia Tensor of Parafoil
Transformation
Controller
Skew-Symmetric Matrices
Distances in Vector Form
Variables Defined in Workspace

Figure 9.3: Simulink Parafoil-Payload System Descent Trajectory Model Overview

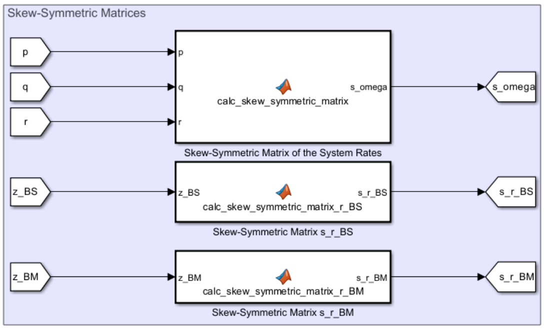

Skew-Symmetric Matrices
p
q
r
calc_skew_symmetric_matrix
s_omega
Skew-Symmetric Matrix of the System Rates
z_BS
calc_skew_symmetric_matrix_r_BS
s_r_BS
Skew-Symmetric Matrix s_r_BS
z_BM
calc_skew_symmetric_matrix_r_BM
s_r_BM
Skew-Symmetric Matrix s_r_BM

Figure 9.4: Skew-Symmetric Matrices

The subsystem for Skew-Symmetric Matrices calculations is shown in Figure 9.4. It takes the angular rotation vector of the body about the center of mass and the distances from the body's center of mass to the point of interest to compute the Skew-Symmetric Matrices. The subsystem calculating the distance from the payload center to the combined system's center of mass is shown in Figure 9.5.

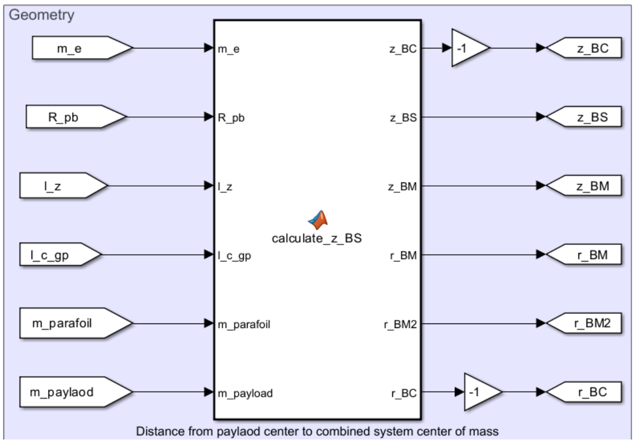


Figure 9.5: Distance from payload center to combined system center of mass

The environment model is presented in Figure 9.6. The environment model comprises the gravity model, atmospheric model, and wind model. The wind model itself consists of three sub-models: the Horizontal Wind Model, the Discrete Wind Gust Model, and the Von Karman Wind Turbulence Model. Each sub-model of the wind model can be manually activated or deactivated using switches. Additionally, there are three subsystems that perform the following transformations: converting flat Earth positions to geodetic latitude, longitude, and altitude, converting gravity from ECEF to the Flat Earth Gravity Model, and calculating the magnitude of the body's velocity.

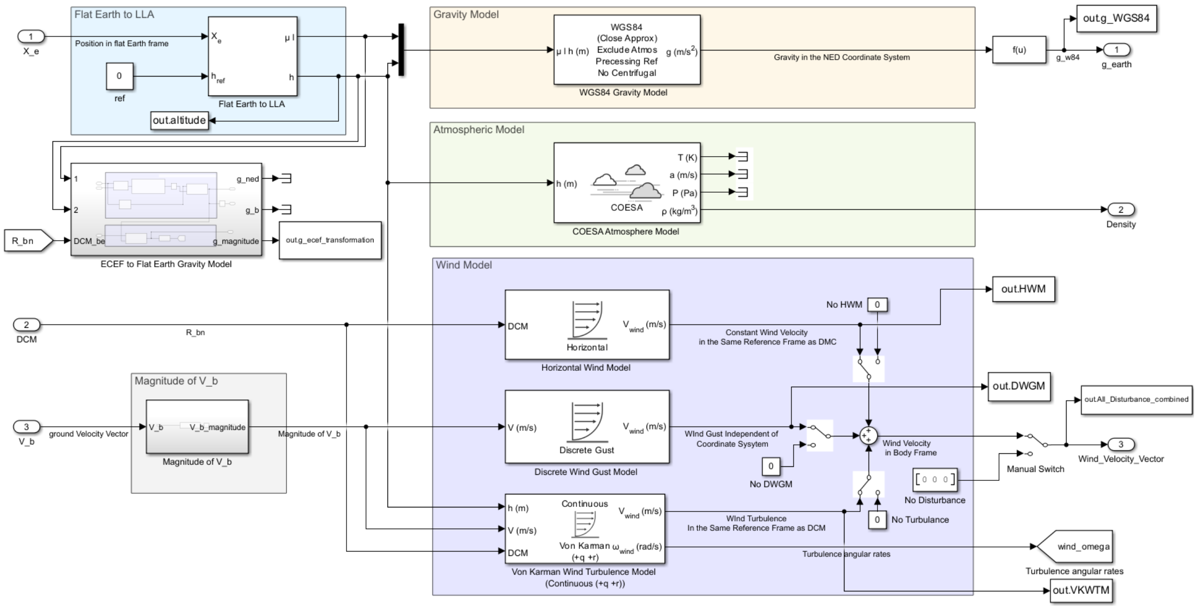


Figure 9.6: Environment Model

The subsystem responsible for converting ECEF (Earth-Centered, Earth-Fixed) gravity to the Flat Earth gravity model is shown in Figure 9.7.

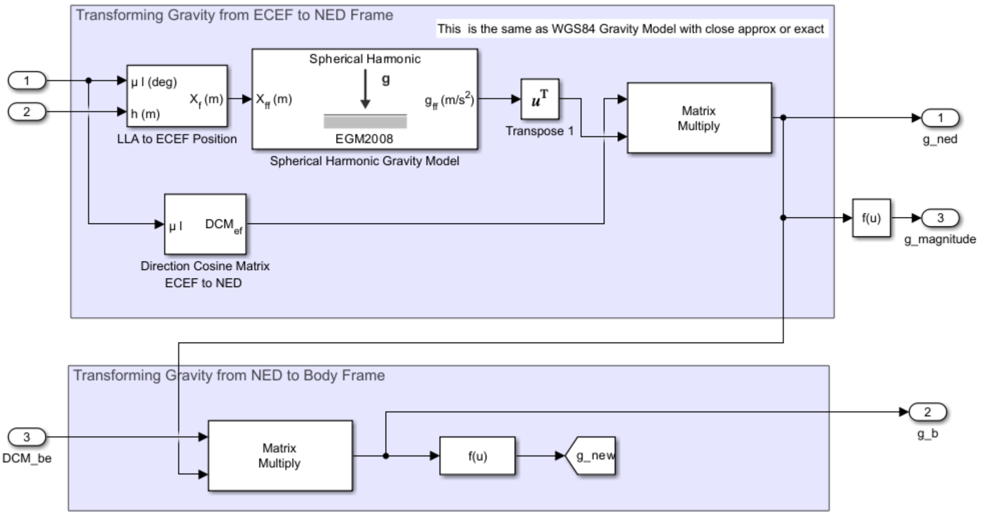

Transforming Gravity from ECEF to NED Frame
This is the same as WGS84 Gravity Model with close approx or exact
1
2
μ l (deg)
h (m)
X_f (m)
LLA to ECEF Position
Spherical Harmonic
g
X_ff (m)
g_ff (m/s²)
EGM2008
Spherical Harmonic Gravity Model
u^T
Transpose 1
Matrix Multiply
1
g_ned
μ l
DCM_ef
Direction Cosine Matrix ECEF to NED
f(u)
3
g_magnitude
Transforming Gravity from NED to Body Frame
3
DCM_be
Matrix Multiply
f(u)
g_new
2
g_b

Figure 9.7: ECEF Gravity to Flat Earth Gravity Model

The system responsible for calculating airspeed, parafoil and body velocities, as well as attitude angles, is shown in Figure 9.8.

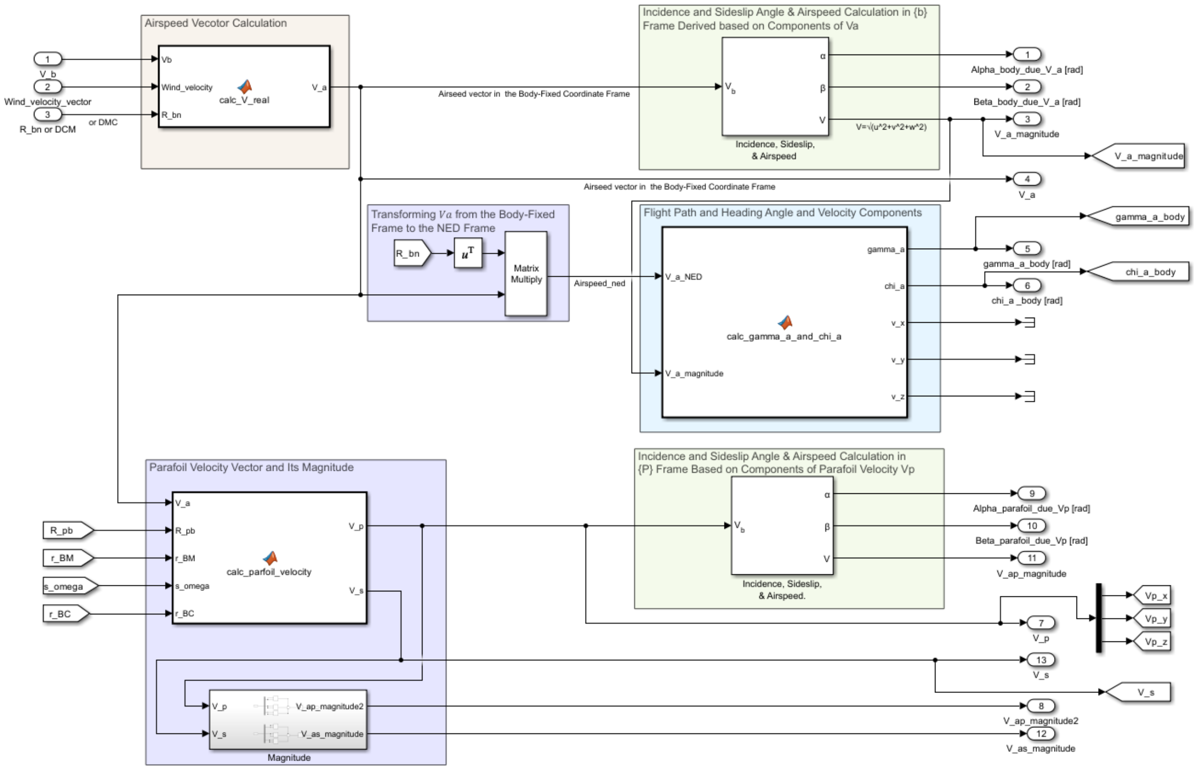

Airspeed Vecotor Calculation
V_b
Wind_velocity_vector
R_bn or DCM
or DMC
calc_V_real
Airseed vector in the Body-Fixed Coordinate Frame
Incidence and Sideslip Angle & Airspeed Calculation in {b} Frame Derived based on Components of Va
Incidence, Sideslip, & Airspeed
V=√(u^2+v^2+w^2)
Alpha_body_due_V_a [rad]
Beta_body_due_V_a [rad]
V_a_magnitude
V_a
Transforming Va from the Body-Fixed Frame to the NED Frame
Matrix Multiply
Airspeed_ned
Flight Path and Heading Angle and Velocity Components
calc_gamma_a_and_chi_a
gamma_a_body [rad]
chi_a_body [rad]
gamma_a_body
chi_a_body
Parafoil Velocity Vector and Its Magnitude
calc_parfoil_velocity
R_pb
r_BM
s_omega
r_BC
Incidence and Sideslip Angle & Airspeed Calculation in {P} Frame Based on Components of Parafoil Velocity Vp
Incidence, Sideslip, & Airspeed.
Alpha_parafoil_due_Vp [rad]
Beta_parafoil_due_Vp [rad]
V_ap_magnitude
V_p
V_s
Vp_x
Vp_y
Vp_z
Magnitude
V_ap_magnitude2
V_as_magnitude

Figure 9.8: Airspeed, Parafoil and body velocities, and Attitude Angles Calculation

The system responsible for calculating aerodynamic coefficients, is shown in Figure 9.9.

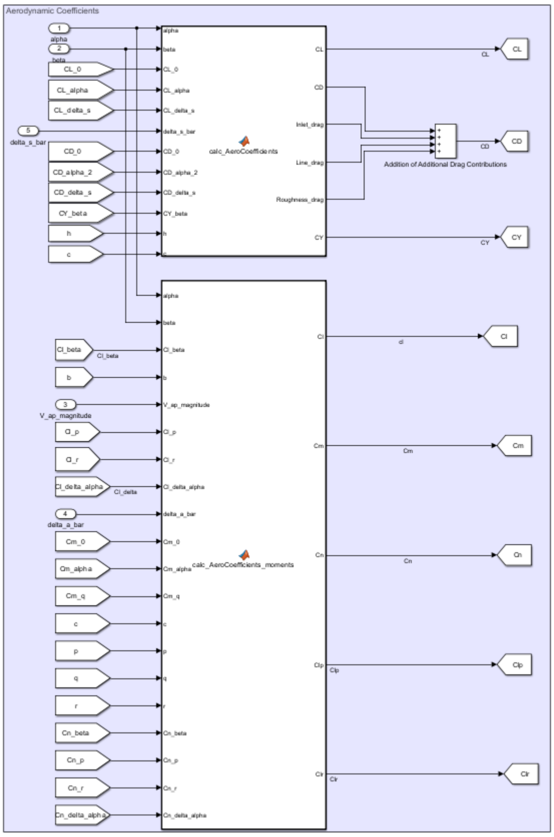


Figure 9.9: Aerodynamic Coefficients Calculation

The system responsible for rotating the parafoil's apparent mass and inertia tensor by the angle μ is shown in Figure 9.10.

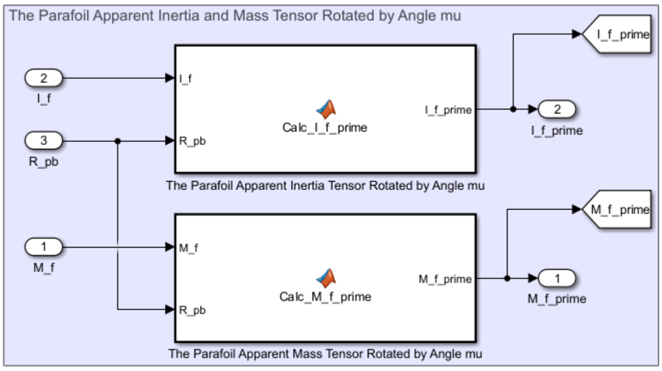

The Parafoil Apparent Inertia and Mass Tensor Rotated by Angle mu
2
I_f
3
R_pb
1
M_f
I_f
R_pb
Calc_I_f_prime
I_f_prime
The Parafoil Apparent Inertia Tensor Rotated by Angle mu
M_f
R_pb
Calc_M_f_prime
M_f_prime
The Parafoil Apparent Mass Tensor Rotated by Angle mu
I_f_prime
2
I_f_prime
M_f_prime
1
M_f_prime

Figure 9.10: The Parafoil Apparent Mass and Inertia Tensor Rotated by Angle mu

The system responsible for calculating the added mass, inertia tensor, and apparent mass and inertia tensor of the parafoil is shown in Figure 9.11. Figure 9.12 shows the system that is responsible for calculating The Combined Inertia Tensor of the Entire System. 9.12 illustrates the system used to compute the combined inertia tensor of the entire system.

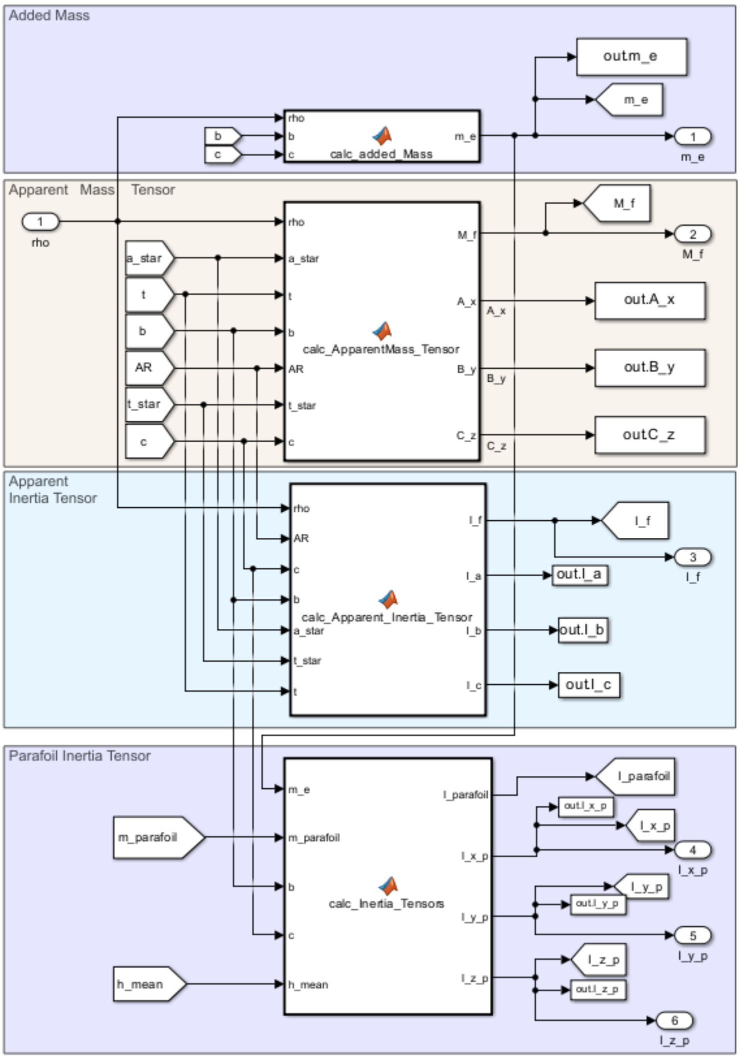


Figure 9.11: Added Mass, Inertia Tensor, and Apparent Mass and Inertia Tensor of the Parafoil

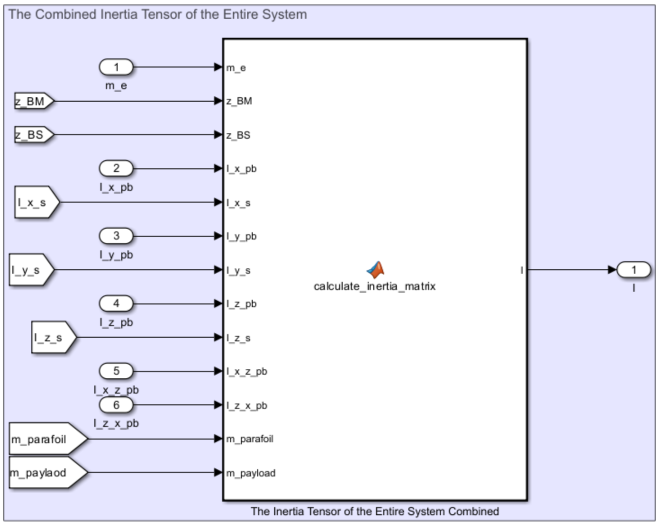


Figure 9.12: The Inertia Tensor of the Entire System Combined

The transformation matrices applied to the parafoil descent model are shown in Figure 9.13. The commented-out matrices are included for additional testing or alternative use cases.

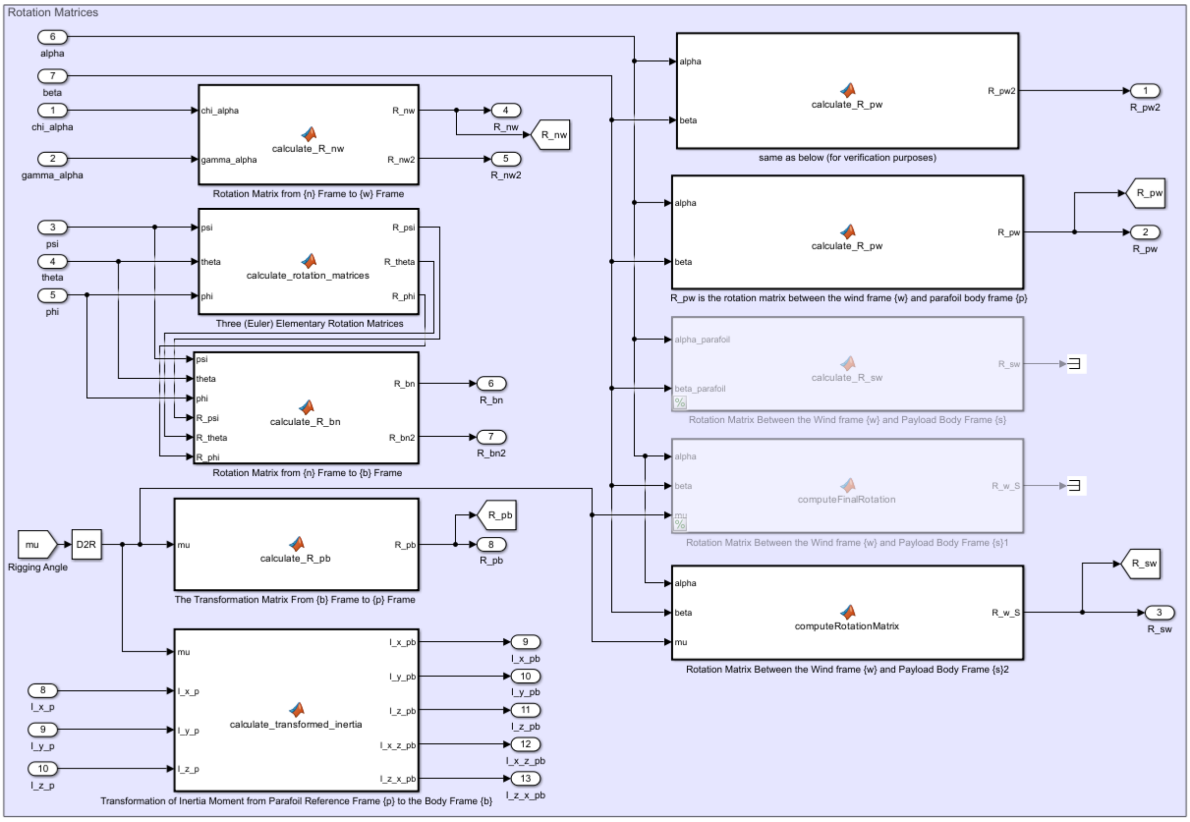

Rotation Matrices
6 alpha
7 beta
1 chi_alpha
2 gamma_alpha
chi_alpha
gamma_alpha
calculate_R_nw
R_nw
R_nw2
Rotation Matrix from {n} Frame to {w} Frame
4 R_nw
R_nw
5 R_nw2
3 psi
4 theta
5 phi
psi
theta
phi
calculate_rotation_matrices
R_psi
R_theta
R_phi
Three (Euler) Elementary Rotation Matrices
psi
theta
phi
R_psi
R_theta
R_phi
calculate_R_bn
R_bn
R_bn2
Rotation Matrix from {n} Frame to {b} Frame
6 R_bn
7 R_bn2
mu
Rigging Angle
D2R
mu
calculate_R_pb
R_pb
The Transformation Matrix From {b} Frame to {p} Frame
R_pb
8 R_pb
mu
I_x_p
I_y_p
I_z_p
calculate_transformed_inertia
I_x_pb
I_y_pb
I_z_pb
I_x_z_pb
I_z_x_pb
Transformation of Inertia Moment from Parafoil Reference Frame {p} to the Body Frame {b}
8 I_x_p
9 I_y_p
10 I_z_p
9 I_x_pb
10 I_y_pb
11 I_z_pb
12 I_x_z_pb
13 I_z_x_pb
alpha
beta
calculate_R_pw
R_pw2
same as below (for verification purposes)
1 R_pw2
alpha
beta
calculate_R_pw
R_pw
R_pw is the rotation matrix between the wind frame {w} and parafoil body frame {p}
R_pw
2 R_pw
alpha_parafoil
beta_parafoil
calculate_R_sw
R_sw
Rotation Matrix Between the Wind frame {w} and Payload Body Frame {s}
alpha
beta
mu
computeFinalRotation
R_w_S
Rotation Matrix Between the Wind frame {w} and Payload Body Frame {s}1
alpha
beta
mu
computeRotationMatrix
R_w_S
Rotation Matrix Between the Wind frame {w} and Payload Body Frame {s}2
R_sw
3 R_sw

Figure 9.13: Transformation Matrices

The guidance, navigation, and control (GNC) model for the parafoil-payload system is shown in Figure 9.14. A PD controller is employed to regulate the yaw angle. The manual switch allows for the activation or deactivation of the asymmetric control input.

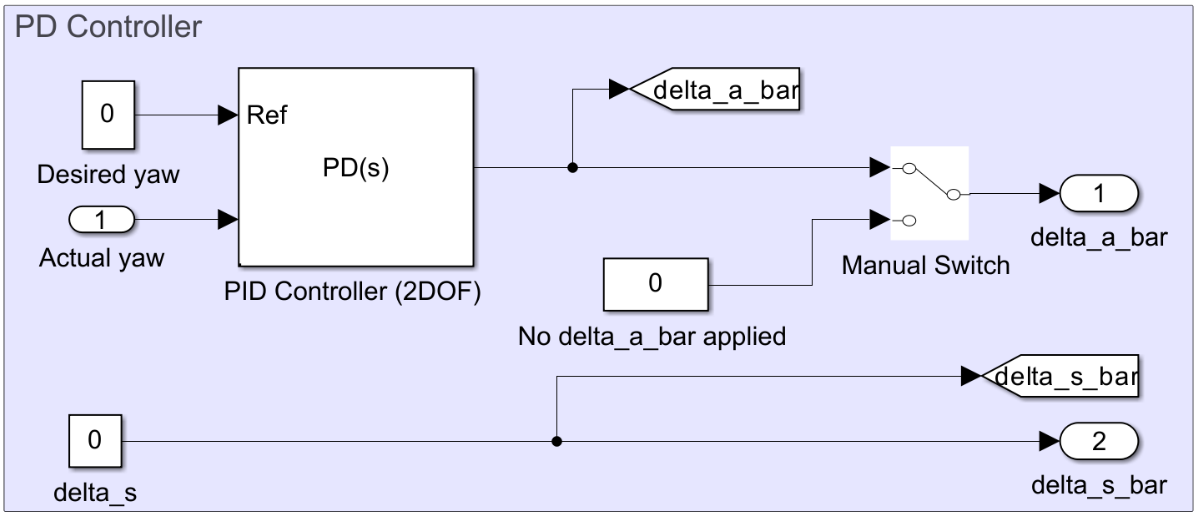


Figure 9.14: Parafoil Guidance, Navigation and Control Model

The forces acting on the parafoil-payload system are illustrated in Figure 9.15.

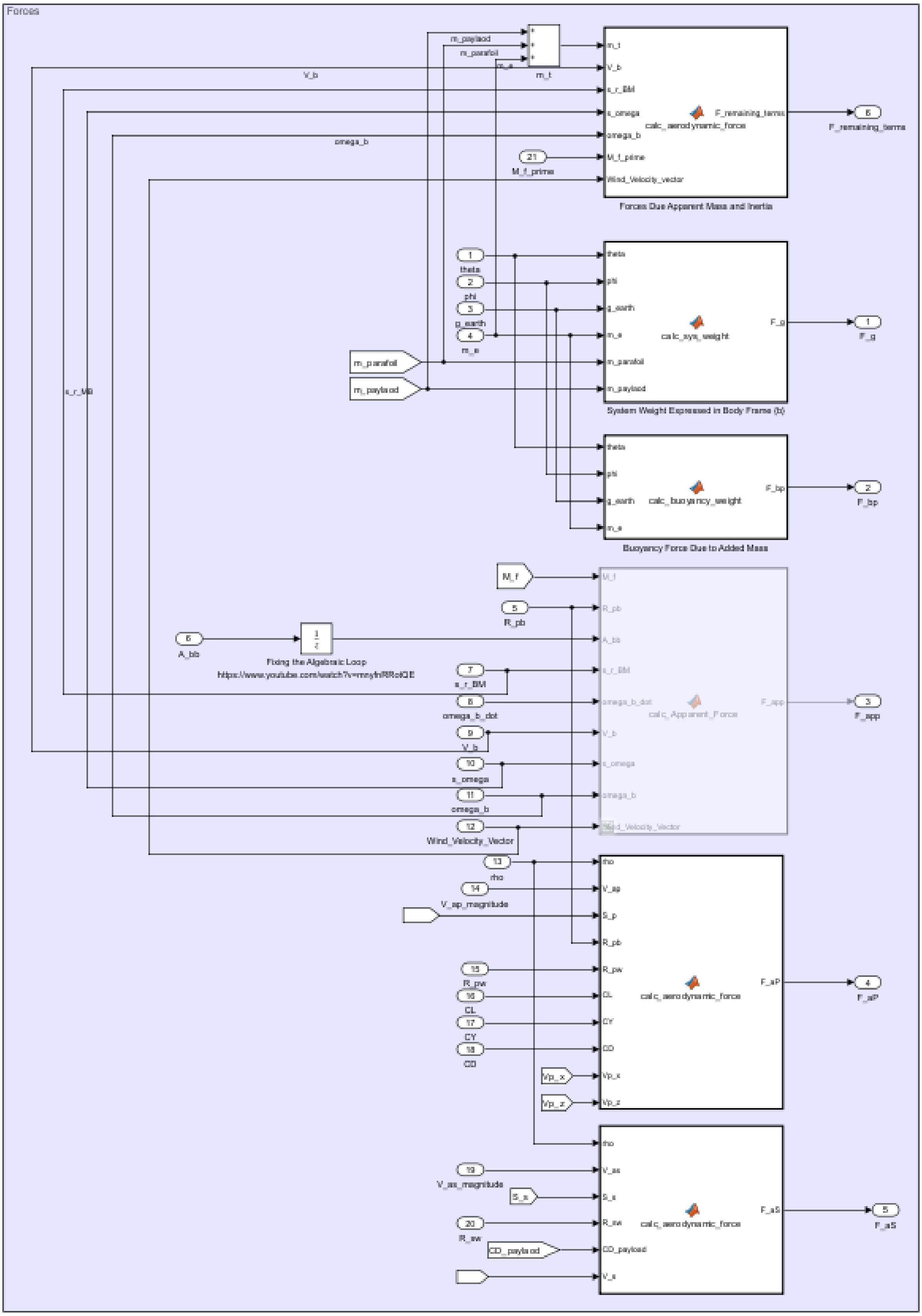


Figure 9.15: Forces Acting on the Parafoil-Payload System

The moments acting on the parafoil-payload system are illustrated in Figure 9.16.

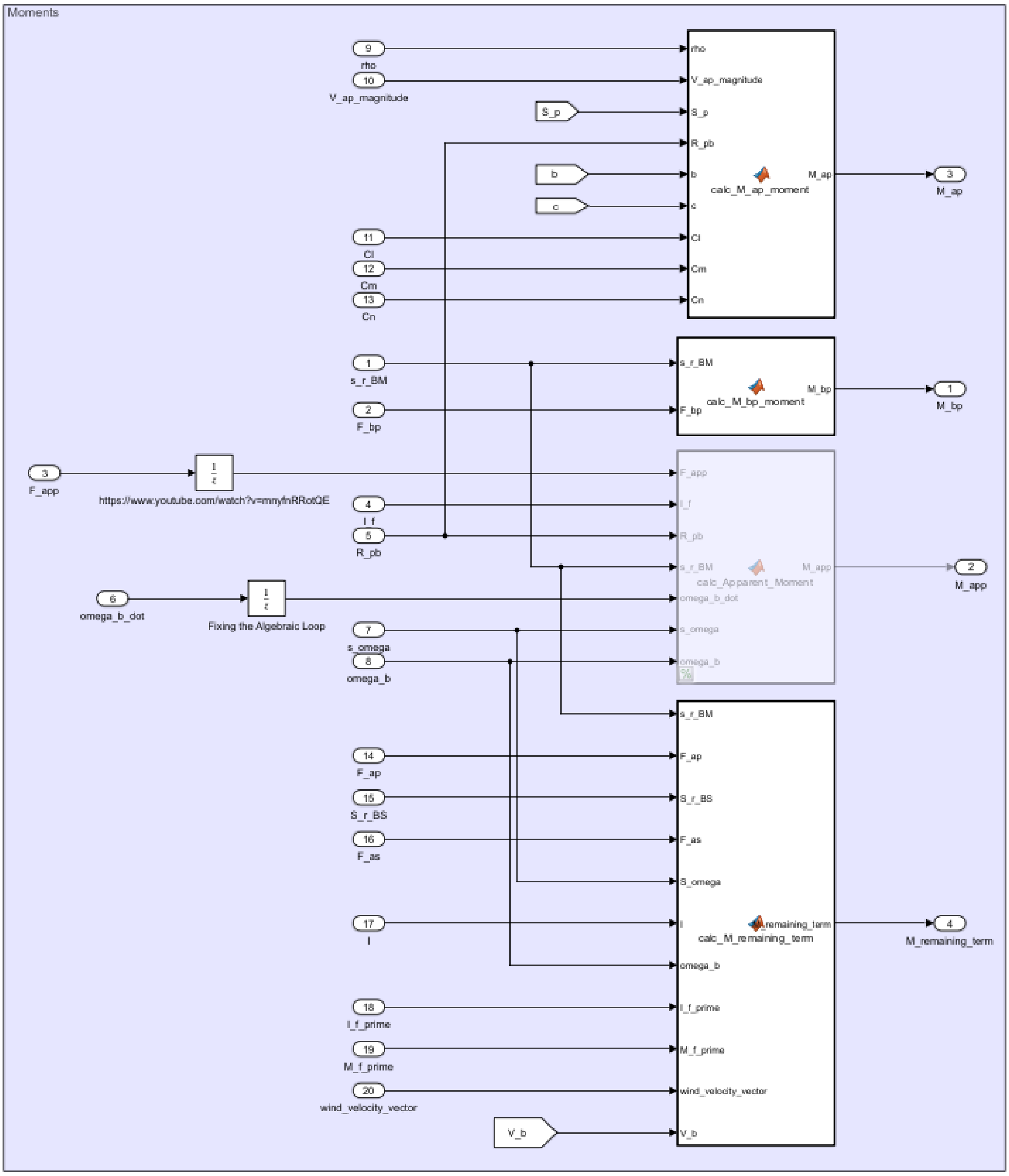


Figure 9.16: Moments Acting on the Parafoil-Payload System

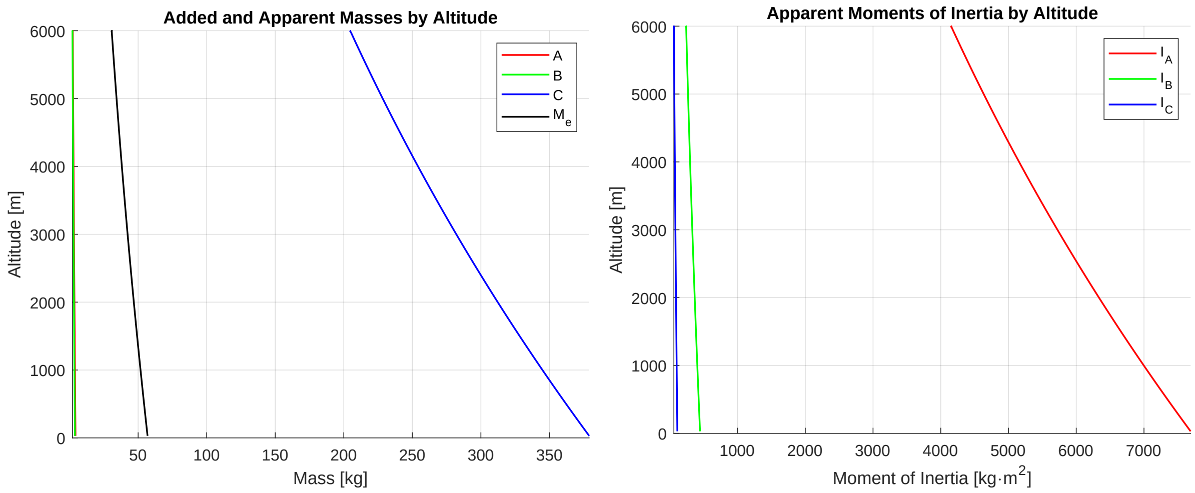
10

Results of the Descent Phase: Parafoil Based First Stage Recovery

This chapter presents the results and sensitivity analysis of the descent phase of the Electron rocket, focusing specifically on the parafoil-based recovery system for the first stage. Table D.1 presents the initial conditions, which include position, velocity, Euler orientation, and angular velocity, derived from the descent trajectory of the first stage with the deployed ballute, just prior to parafoil deployment, at an altitude of 6 km, as described in Section 8.1. The results are detailed in Sections 10.1 and 10.2, while the sensitivity analysis is discussed in Section 10.3.

10.1. Results of the Uncontrolled Parafoil Descent Trajectory Model

Figure 10.1 presents the simulation results for the apparent masses and moments of inertia in the x, y, and z directions, denoted as A, B, and C for masses, and I_A, I_B, and I_C for moments of inertia, respectively. m_e is the added mass. As shown in Figure 10.1a, the apparent mass in the z-direction is the dominant component, significantly larger than the apparent masses in the x- and y-directions. In fact, it exceeds the parafoil's mass of 73.5 kg. The apparent mass in the y-direction is smaller than that in the x-direction. This aligns with the geometric representation of the apparent masses, as shown in Figure 8.11, which illustrates their proportionality. The plots in the graph do not reach zero altitude, as the simulation is designed to terminate at an altitude of 30 m. The added mass, while not as large as C, ranges from 30 kg to 57 kg (40% to 77% of the parafoil's mass).



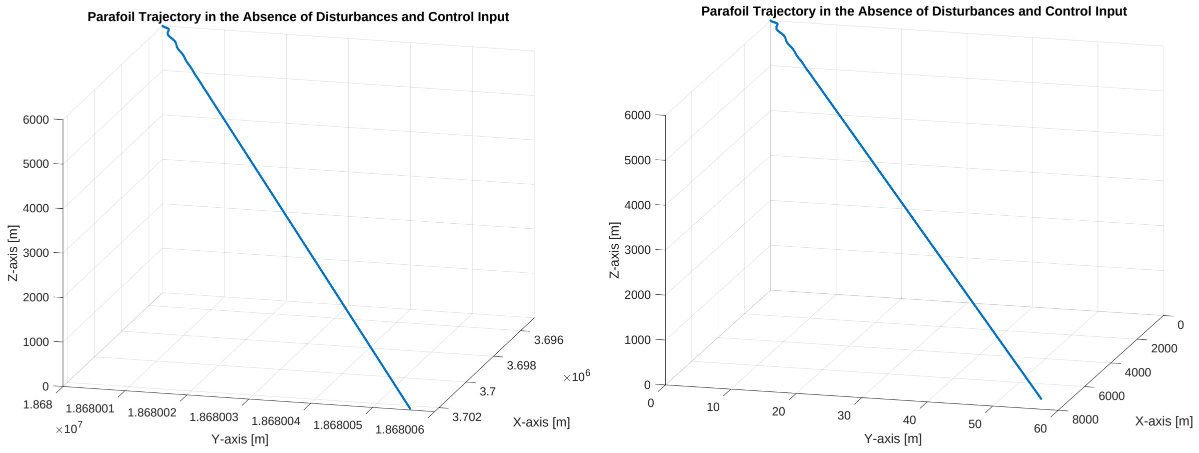
(a) Apparent & Added Masses

(b) Apparent Moments of Inertia

Figure 10.1: Apparent Masses and Moments of Inertia Profiles in Parafoil Descent Trajectory

Based on Figure 10.1b, the apparent moment of inertia in the x-direction is the dominant component, significantly larger than the apparent moments of inertia in the x- and y-directions. This also aligns with the geometric representation of the apparent moments of inertia, as shown in Figure 8.12. Hence, the added and apparent masses and apparent moments of inertia must be incorporated into the parafoil's EOM.

First, the descent trajectory, as shown in Figure 10.2, is presented without considering disturbances such as wind velocity, wind gusts, and turbulence, and without applying any control input. The rigging angle, μ, is -6 degrees. Figure 10.2a illustrates the trajectory with axis values relative to the launch position coordinates. For better clarity and readability, the same trajectory is shown in Figure 10.2b, where the initial X- and Y-axes are reset to zero. It can be observed that the trajectory initially experiences small oscillations due to the initial conditions (body velocity, Euler orientation, and angular velocity along the three axes). However, these oscillations quickly dampen, and the trajectory settles into a straight glide descent as expected. This behavior, of course, changes when disturbances are introduced.

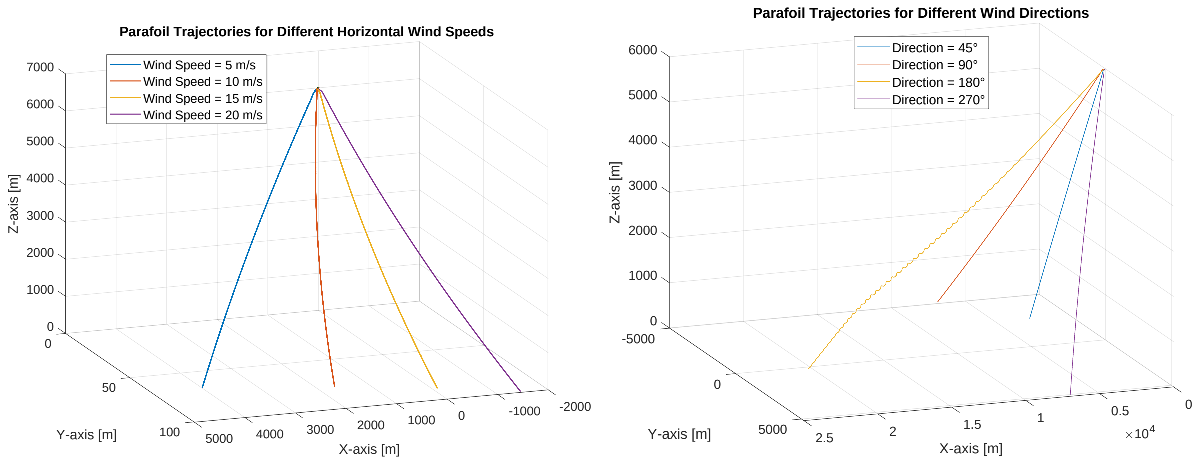


(a) Descent Trajectory Relative to Launch Coordinates

(b) Descent Trajectory with X- and Y-axis Reset to Zero

Figure 10.2: Parafoil-Payload System Descent Trajectory in the Absence of Disturbances and Control Input

Disturbances are now introduced without applying control input to examine their effects. The rigging angle, μ, is maintained at $-6°$. Figure 10.3 illustrates the results of parafoil-payload system descent trajectories under varying horizontal wind speeds and wind directions.



(a) Variation in Horizontal Wind Speeds at 0° Wind Direction

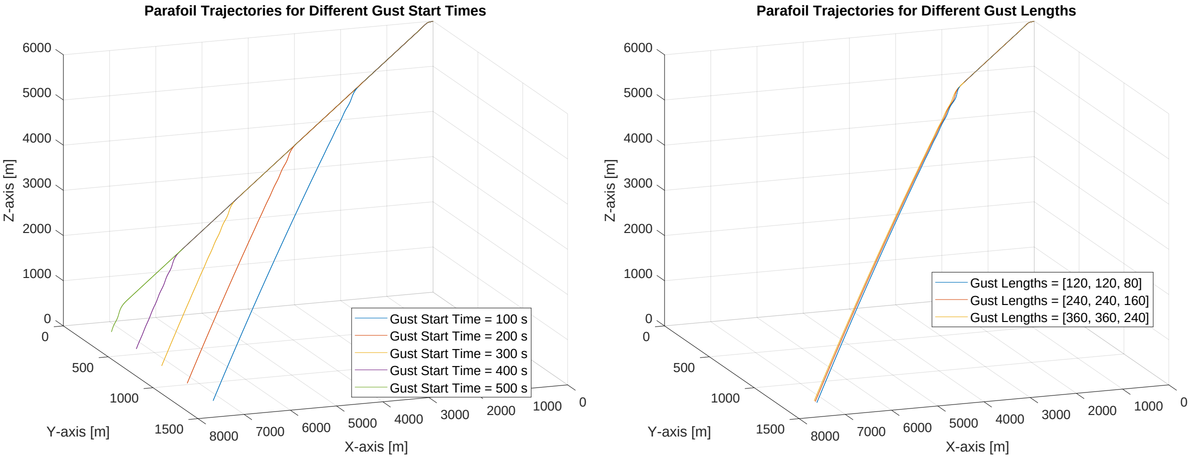
(b) Variation in Wind Directions Measured Clockwise from the Earth's x-axis (North) at 10 m/s Horizontal Wind Speed

Figure 10.3: Parafoil-Payload System Descent Trajectories under Different Horizontal Wind Speeds and Wind Directions

Figure 10.3a illustrates the effect of horizontal wind speeds ranging from 5 to 20 m/s at a wind direction of 0°. The wind direction is defined as the direction from which the wind is coming, measured in degrees clockwise from the Earth's x-axis (north) (see Section 8.4.3). Since the wind velocity is purely horizontal and coming from the north (corresponding to a 0° wind direction), and the rigging angle μ is negative, this causes the parafoil-payload system to move forward. The slight movement along the y-axis (east direction) is attributed to initial conditions. At the lowest wind speed, the traveled distance along the x-axis (north) is maximized. As the wind speed increases, the traveled distance along the x-axis decreases until the horizontal wind speed matches the parafoil's velocity, around 16 m/s. When the wind speed reaches 20 m/s, the parafoil is effectively pushed backward along the x-axis. Thus, it can be concluded that the horizontal wind speed primarily determines how far the parafoil can glide, in the absence of other disturbances.

Figure 10.3b illustrates the effect of wind direction, ranging from 45° to 270°, at a fixed horizontal wind speed of 10 m/s. The wind direction has a significant influence on the parafoil's trajectory. When the wind direction is 90° (i.e., the wind is coming from the positive y-axis in the NED reference frame), the traveled distance along the y-axis becomes -500 m. Conversely, when the wind direction is 270° (i.e., the wind is coming from the negative y-axis), the y-axis component of the trajectory reaches 5000 m. These values are equal in magnitude but opposite in direction, as expected. For a wind direction of 180°, corresponding to a tailwind (where the wind pushes the parafoil forward), the traveled distance along the x-axis (north direction) is maximized. This distance is considerably greater than in the case of a headwind, as illustrated in Figure 10.3a. A tailwind increases the parafoil's relative velocity with respect to the ground, enabling it to travel a greater distance. Hence, both the horizontal wind speed and wind direction have a profound effect on the parafoil's trajectory, highlighting their critical role in determining parafoil's flight performance.

Next, the effect of discrete wind gusts will be analyzed. The rigging angle is held constant at -6°, with the wind direction set to 0°and the horizontal wind speed at 0 m/s. Additionally, no control input is applied. Initially, the gust length and amplitude will be held constant at $[120, 120, 80]$ m and $[3.5, 3.5, 3.0]$ m/s along the x-, y-, and z-axes, respectively, while the starting time will be varied as shown in Figure 10.4a. As detailed in Section 8.4.4, the wind gust model operates such that when a wind gust is initiated, it must travel the defined gust length distance before stabilizing at a constant output. This makes the gust start time a critical factor. The earlier the gust starts, the more dispersed the trajectory becomes, resulting in increased travel distances along all three axes. Due to the very small velocity along the y-axis, the earliest start time produces the greatest traveled distance.

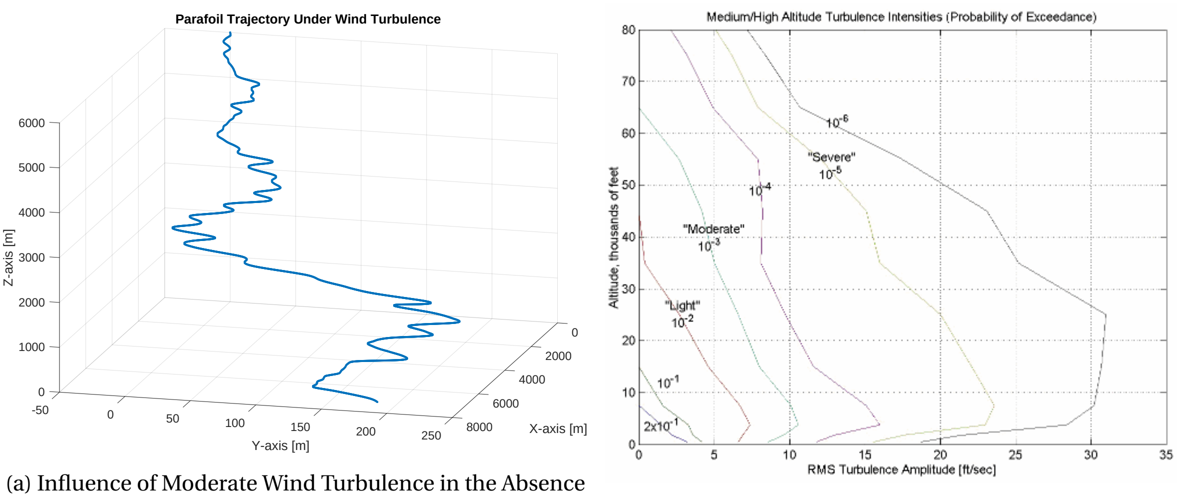


(a) Variation in Gust Start Time

(b) Variation in Gust Length

Figure 10.4: Impact of Gust Start Time and Gust Length on Parafoil Descent Trajectories in the Absence of Control Inputs

With the gust start time set at 100 s and the amplitude fixed at $[3.5, 3.5, 3.0]$ m/s, the effect of gust length is minimal. The trajectories remain nearly identical, even when the gust length is doubled and tripled, as illustrated in Figure 10.4b. The gust amplitude is clearly a significant parameter, as it determines the output wind gust, which in turn increases the dispersion of the trajectory when it is higher. Its impact is partially shown in Figure 10.4a, making a separate figure unnecessary.
Figure 10.5a illustrates the impact of the wind turbulence model on the trajectories. To isolate its effect, previously mentioned disturbances are excluded. The default settings are applied: a wind speed of 15 m/s for low-altitude turbulence intensity, a wind direction of 0°, a scale length of 762 m at medium/high altitudes, and a noise seed of $[23341, 23342, 23343, 23344]$. This four-element vector is used to generate turbulence signals for the three velocity components and the roll rate, respectively.



(a) Influence of Moderate Wind Turbulence in the Absence of Control Inputs

(b) Wind Turbulence Intensity Levels [62]

Figure 10.5: Impact of Wind Turbulence on Parafoil Descent Trajectories in the Absence of Control Inputs with Moderate Wind Turbulence Intensity Level

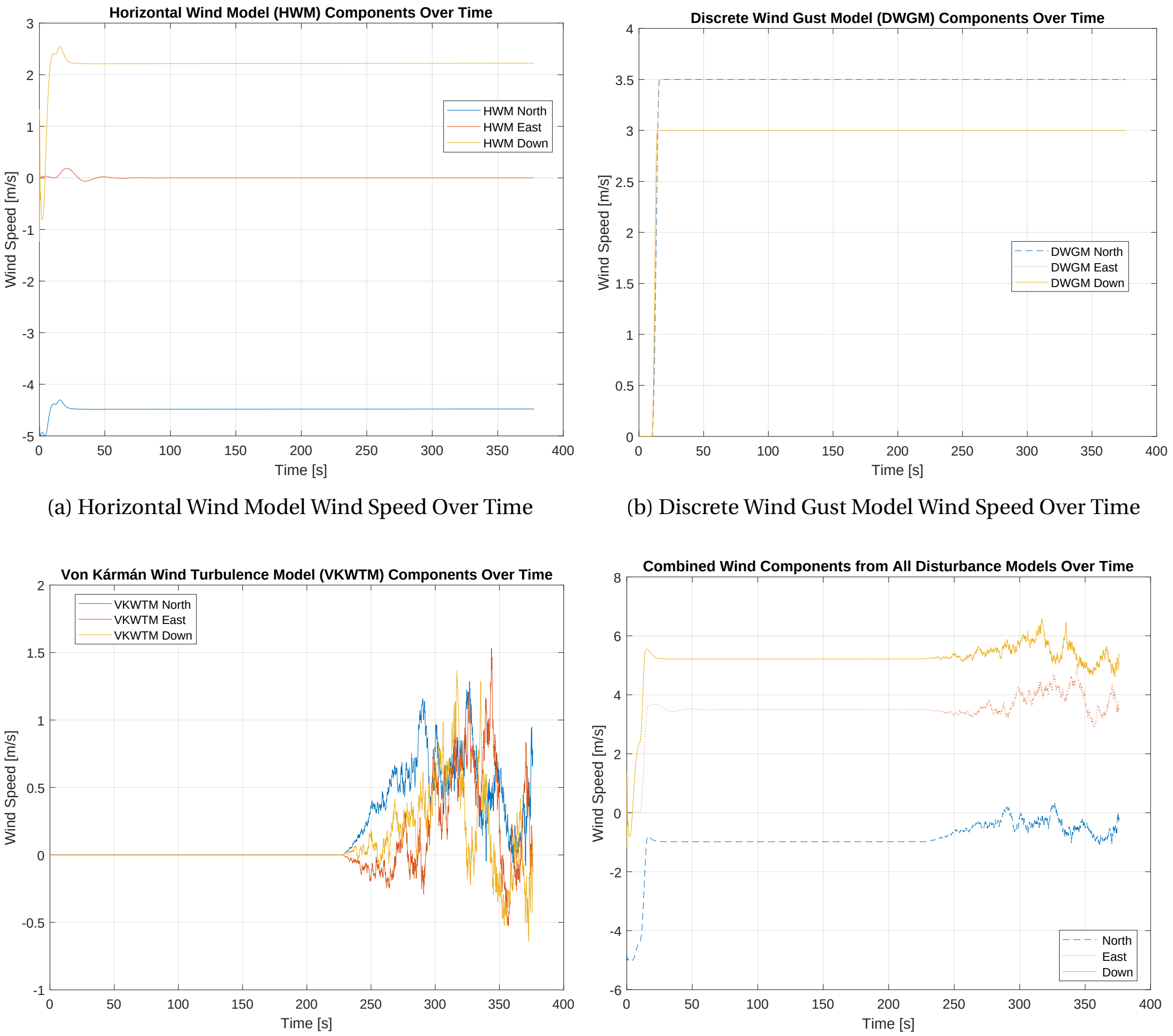
When wind turbulence is introduced, the trajectory becomes more irregular, exhibiting additional edges and curves. The changes in travel distance are not simply an increase or decrease but occur randomly, varying across different layers as defined by the Von Kármán Wind Turbulence Model. This model is specified in U.S. military standards [58] [60] [61] (see Section 8.4.5). Combining all these models results in a more accurate and realistic representation of the environment. Until now, the impact of each model has been analyzed independently and without control inputs. In the next step, the trajectory will be examined with all disturbances applied while ensuring a controlled straight descent flight.

10.2. Results of the Controlled Parafoil Descent Trajectory Model

This section presents the results of the controlled parafoil descent trajectory. The input parameters, including those for the *Horizontal Wind Model*, *Discrete Wind Gust Model*, and *Von Kármán Wind Turbulence Model*, are summarized in Table 10.1. The plots for each model in the NED reference frame are shown in Figure 10.6. In these plots, “North,” “East,” and “Down” correspond to the x-, y-, and z-directions, respectively. The initial jump in Figure 10.6a is attributed to the initial conditions. After the wind gust is applied, the wind speed does not return to zero but remains at a defined amplitude, as illustrated in Figure 10.6b. Figure 10.6c presents the generated wind turbulence, and Figure 10.6d displays the combined effects of all the models. Notably, the wind turbulence affects all three axes. Its impact is greater than its own amplitude because it causes the behavior of the other models to change. In effect, the turbulence modifies the constant horizontal wind speed, leading to variations in amplitude such that the overall combined effect exceeds the simple sum of the individual contributions.

Table 10.1: Wind Model Parameters

Parameter	Value [Unit]
Horizontal wind speed	5 m/s
Wind direction	0°
Gust start time	10 s
Gust length (dx, dy, dz)	[120, 120, 80] m
Gust amplitude (ug, vg, wg)	[3.5, 3.5, 3.0] m/s
Wind speed for low-altitude intensity	3 m/s
Rigging angle μ	-6°
Scale length at medium/high altitudes (above 6010 m)	762 m
Noise seed	[23341, 23342, 23343, 23344] [-]



(a) Horizontal Wind Model Wind Speed Over Time

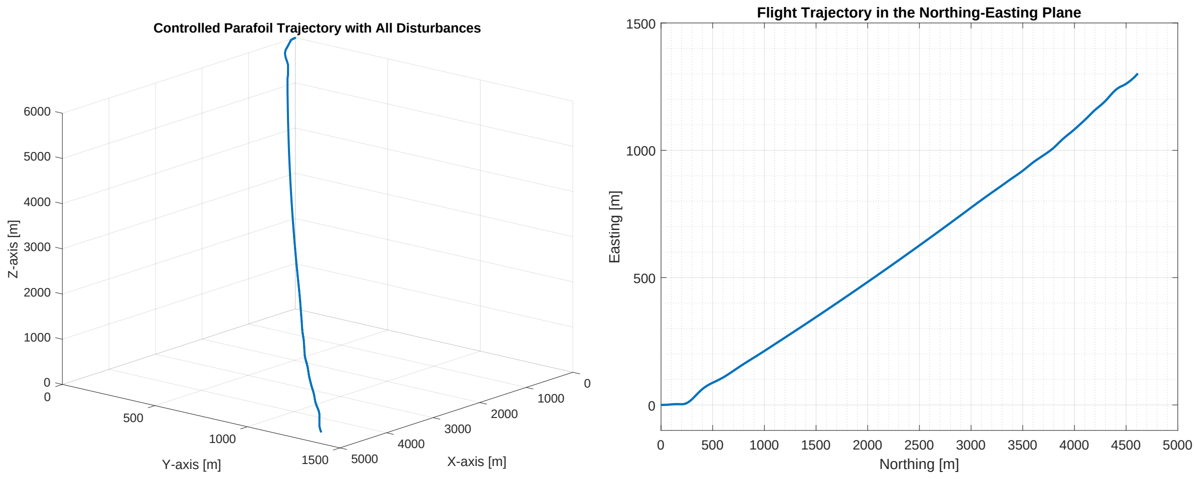
(b) Discrete Wind Gust Model Wind Speed Over Time

(c) Von Kármán Wind Turbulence Model Wind Speed Over Time

(d) Combined Wind Speed Components from All Disturbance Models Over Time

Figure 10.6: Wind Speed Profiles Over Time from Disturbance Models: (a) Horizontal Wind Model, (b) Discrete Wind Gust Model, (c) Von Kármán Wind Turbulence Model, and (d) Combined Wind Speed Components from All models

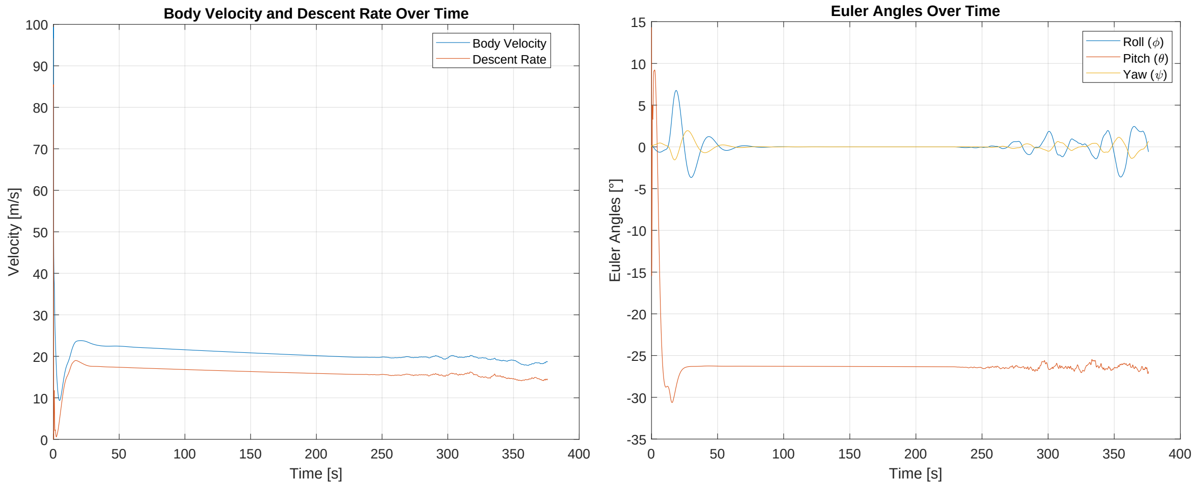
The controlled parafoil descent trajectory under applied disturbances is shown in Figure 10.7. As observed in Figure 10.7a, the trajectory closely approximates a straight-line path, which is crucial for mid-air recovery. This predictable motion simplifies interception using a helicopter, making it significantly easier compared to an uncontrolled trajectory. The slight deviation at the beginning of the trajectory results from initial conditions and wind gust activation, while the minor deviation at the end is attributed to wind turbulence. Figure 10.7b presents the Northing and Easting components of the trajectory. Initially, the trajectory moves northward, and as the wind gust reaches its peak amplitude (see Figure 10.6b), the trajectory begins shifting eastward while maintaining a linear course. The trajectory and its direction will naturally be influenced by changes in wind direction and speed. A more detailed analysis of this is provided in Section 10.3.



(a) Controlled Parafoil-Payload System Trajectory with Applied Disturbances

(b) Northing and Easting Components of the Trajectory with Applied Disturbances

Figure 10.7: Controlled Parafoil-Payload System Trajectory with Applied Disturbances (Left) and Northing and Easting Components of the Trajectory (Right)



(a) Controlled Parafoil-Payload System Velocity and Descent Rate with Applied Disturbances

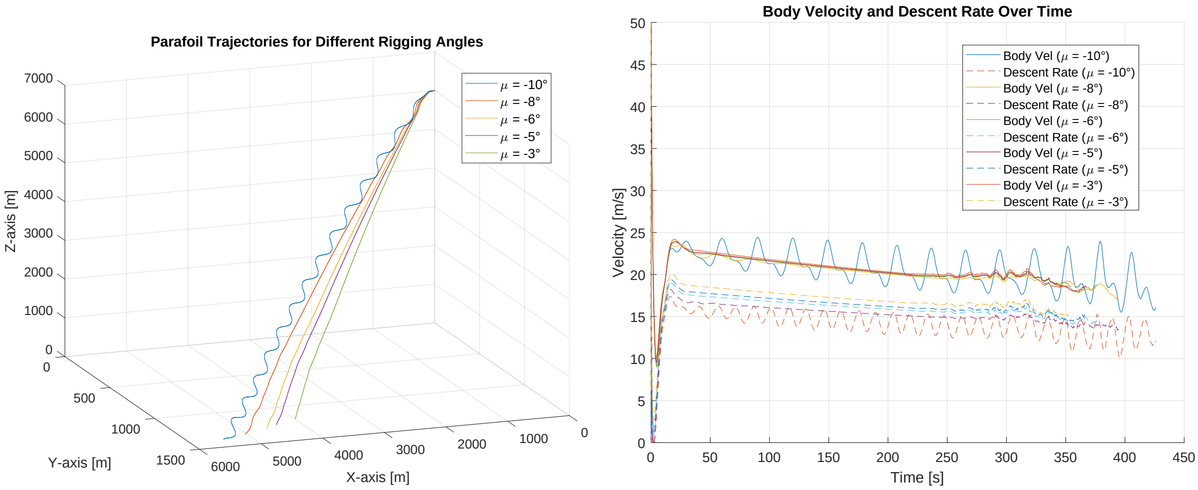
(b) Controlled Parafoil-Payload System Euler Angles with Applied Disturbances

Figure 10.8: Controlled Parafoil-Payload System Velocity, Descent Rate and Euler Angles Over Time

The parafoil-payload system velocity profile, descent rate, and Euler orientation over time are illustrated in Figure 10.8. The parafoil velocity decreases from an initial value of 94 m/s to approximately 20 m/s, while the descent rate stabilizes around 15 m/s. Although both the velocity and descent rate are not strictly constant, their fluctuations are minimal throughout the descent. When comparing the fluctuations at the beginning and end of the descent, the parafoil velocity fluctuates by approximately 5 m/s, and the descent rate varies by around 3 m/s. The total descent time to an altitude of 30 m, where the simulation concludes, is 376 seconds (roughly 6.27 minutes). This is notably shorter than the 10-minute descent time of the actual mission, as detailed in Section 8.8. This discrepancy can be partially attributed to the fact that the simulation was conducted with a headwind (0° wind direction). Additionally, the actual size of the parafoil may be larger than the estimated size used in the simulation. Moreover, the control model in the simulation only accounts for the yaw angle, whereas actual flights likely employ more sophisticated and optimized control mechanisms. As seen in Figure 10.8b, the yaw angle settles to 0°, resulting in a steady, straight descent trajectory. At approximately 230 seconds, due to wind turbulence, the yaw angle begins to fluctuate. The same behavior is observed for the roll angle. The pitch angle settles at approximately -26°, remaining consistent throughout the descent trajectory, with small fluctuations due to turbulent wind at the end of the trajectory. The negative sign of the pitch angle aligns with the definition provided in Section 8.2.1.

10.3. Sensitivity Analysis of Parafoil Performance

Similarly to the ascent trajectory sensitivity analysis in Chapter 7, the one-at-a-time approach is applied. This method varies only a single parameter at a time while keeping all others constant, thus neglecting potential interactions between parameters. To address this limitation, the second part of this section employs Monte Carlo analysis to visualize parameter interactions. The parameters are kept constant as specified in Table 10.1, while the rigging angle μ is varied. The results are presented in Figure 10.9.



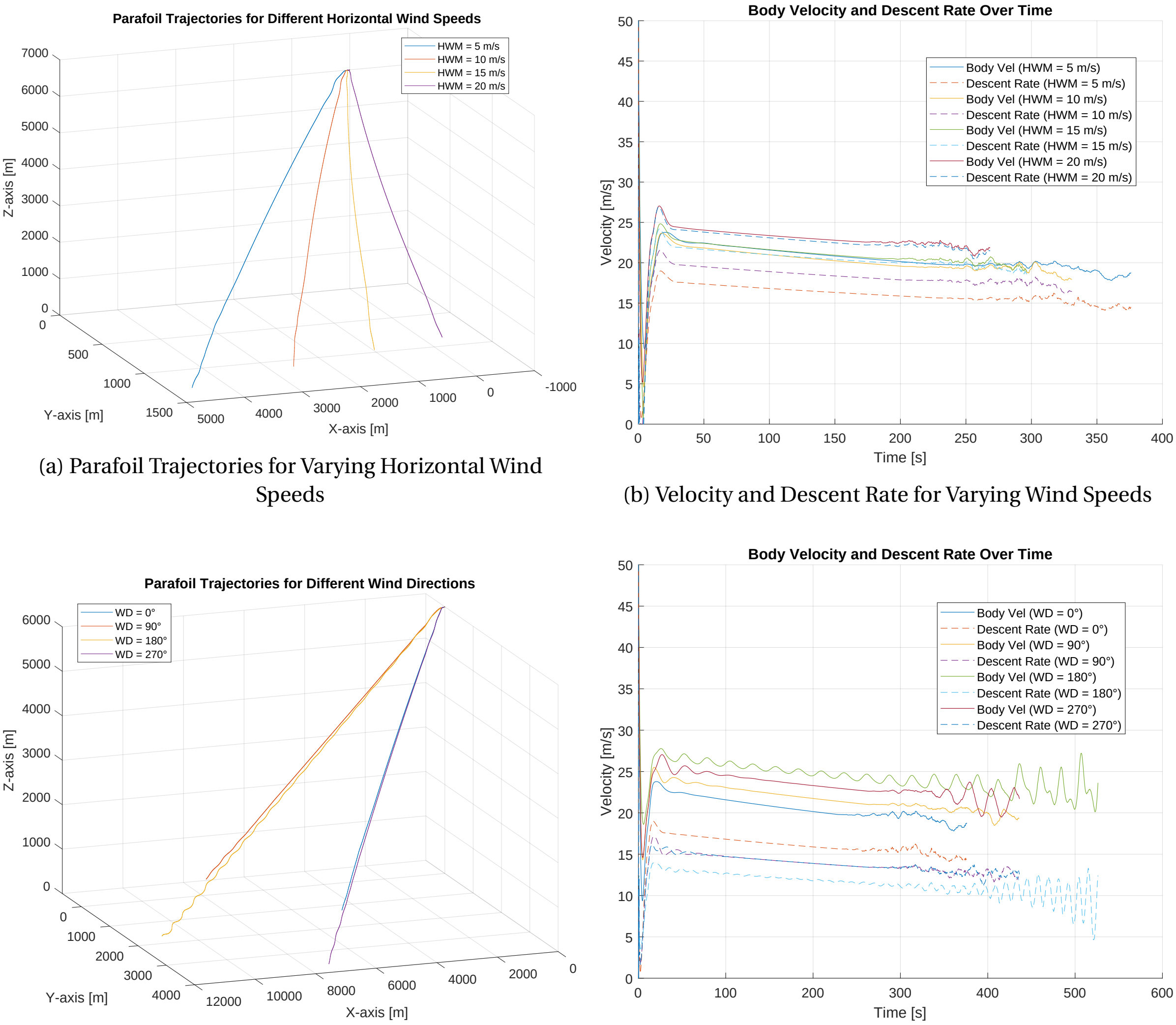
(a) Trajectories of the Parafoil-Payload System for Varying Rigging Angles

(b) Velocity and Descent Rate of the Parafoil-Payload System for Varying Rigging Angles

Figure 10.9: Sensitivity Analysis of Rigging Angle μ: Effects on Trajectory, Velocity, and Descent Rate

As shown in Figure 10.9a, increasing the rigging angle μ results in a greater traveled distance. However, at -10°, the trajectory exhibits oscillations, which can complicate mid-air recovery. (Note: In this context, increasing the rigging angle refers to making the value more negative, such as moving from -3° to -10°). Therefore, it is recommended to keep the rigging angle above -10° (e.g., -6°) to ensure stability. Figure 10.9b illustrates that a larger rigging angle leads to higher velocity and a lower descent rate. At -10°, oscillations appear in both velocity profiles. A more negative rigging angle extends flight time, as increased velocity allows for greater travel distance. For a -10° rigging angle, the flight time is approximately 7.1 minutes.

The sensitivity of wind speed and its direction on the parafoil trajectory, velocity, and descent rate is analyzed by keeping all other parameters constant, as defined in Table 10.1, while varying the horizontal wind speed and its direction. The results are presented in Figure 10.10.



(a) Parafoil Trajectories for Varying Horizontal Wind Speeds

(b) Velocity and Descent Rate for Varying Wind Speeds

(c) Parafoil Trajectories for Varying Wind Directions

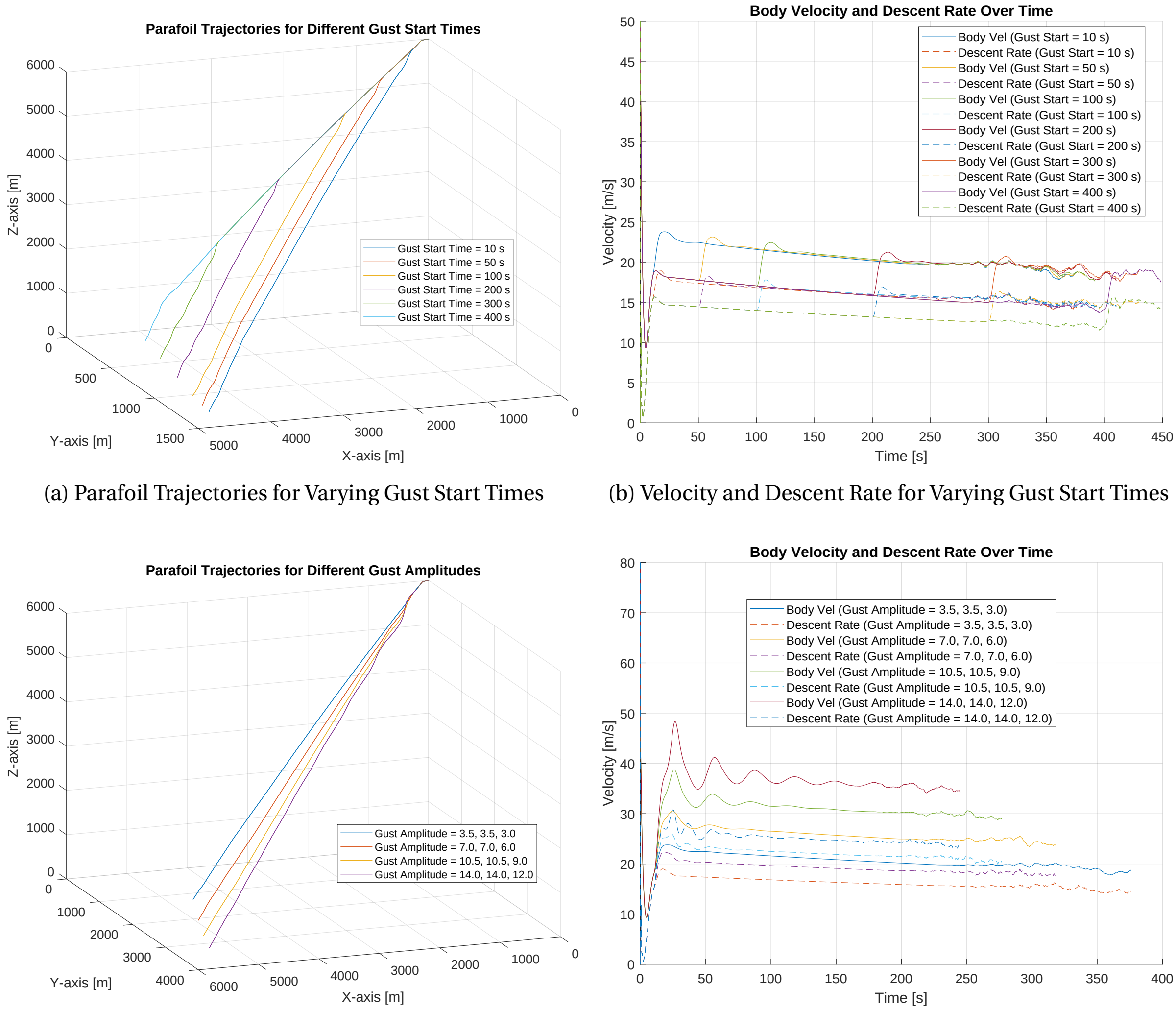
(d) Velocity and Descent Rate for Varying Wind Directions

Figure 10.10: Sensitivity Analysis of Horizontal Wind Speed and Direction: Effects on Trajectory, Velocity, and Descent Rate

Figure 10.10a illustrates the effect of horizontal wind speeds ranging from 5 to 20 m/s at a wind direction of 0°. At the lowest wind speed, the parafoil travels the greatest distance along the x-axis (north). As the wind speed increases, the traveled distance along the x-axis decreases until the horizontal wind speed matches the parafoil's velocity. When the wind speed reaches 20 m/s, the parafoil is effectively pushed backward along the x-axis. Thus, it can be concluded that a headwind reduces the traveled distance, and when the wind speed exceeds the parafoil's velocity, it causes the parafoil to move backward. Therefore, the horizontal wind speed plays a crucial role in determining both the distance the parafoil can glide and its direction of travel. As illustrated in Figure 10.10b, an increase in horizontal wind speed reduces the disparity between the body velocity and descent rate. This is due to the strong headwind, which decreases the x-component of the velocity while simultaneously increasing the downward component.

Figure 10.10c illustrates the effect of wind direction, ranging from 0° to 270°, at a fixed horizontal wind speed of 5 m/s. When the wind direction is from the left or right (corresponding to 90° and 270°), the traveled distance along the x-axis remains the same. However, the distance along the y-axis differs due to the influence of control input, gusts, and the turbulence model. In the absence of these factors, the traveled distance along the y-axis would have been equal in magnitude but opposite in direction, as shown in Figure 10.3b. A tailwind, corresponding to a wind direction of 180° (where the wind pushes the parafoil forward), results in the maximum traveled distance along the x-axis. This distance is significantly greater than that observed with a headwind, as shown in Figure 10.10c. By increasing the parafoil's relative velocity with respect to the ground, a tailwind enables it to cover more ground.

As shown in Figure 10.10d, a tailwind causes the parafoil's velocity to increase. At higher velocities, the lift and drag forces become more dynamic but also fluctuate more, due to variations in airflow over the surface. These fluctuations can result in oscillations in both velocity and displacement trajectory, driven by unsteady aerodynamic effects. Additionally, turbulent airflow intensifies as altitude decreases. As the velocity in the x-direction increases and the descent rate decreases, the parafoil stays in the air longer, encountering stronger turbulence over a longer period of time. This leads to greater fluctuations in both vertical and horizontal directions. Consequently, both horizontal wind speed and wind direction play a critical role in shaping the parafoil's trajectory, underscoring their importance in determining flight performance.



(a) Parafoil Trajectories for Varying Gust Start Times

(b) Velocity and Descent Rate for Varying Gust Start Times

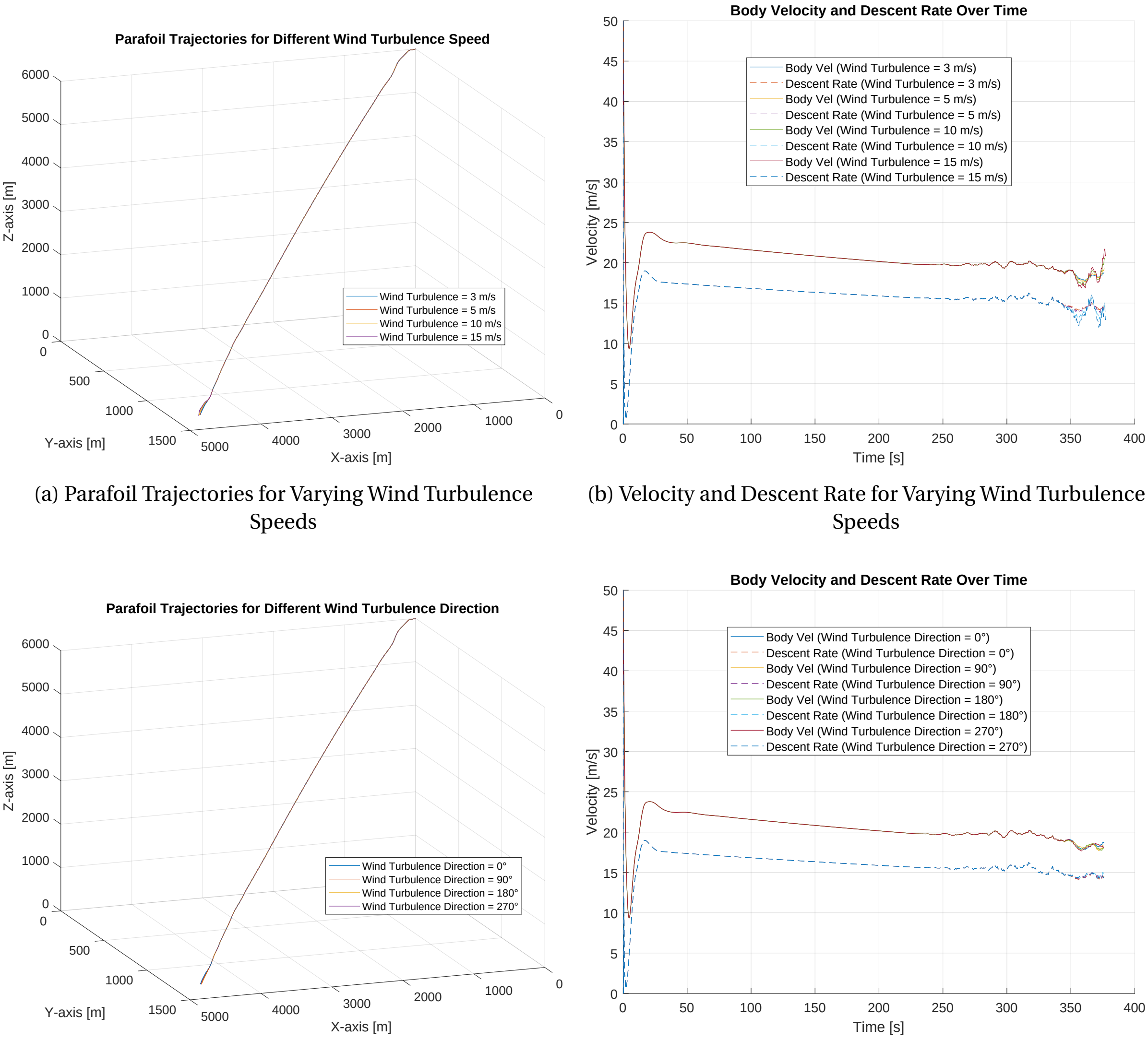
(c) Parafoil Trajectories for Varying Gust Amplitudes

(d) Velocity and Descent Rate for Varying Gust Amplitudes

Figure 10.11: Sensitivity Analysis of Gust Start Times and Amplitudes: Effects on Parafoil Trajectory, Velocity, and Descent Rate

The sensitivity of gust start time and gust amplitude on the parafoil trajectory, velocity, and descent rate is analyzed by keeping all other parameters constant, as defined in Table 10.1, while varying the gust start time and gust amplitude. The results are presented in Figure 10.11. The effect of gust start time on the trajectory is limited. When activated early, it has minimal impact on the traveled distance in the x-axis but increases the traveled distance in the y-axis, as shown in Figure 10.11a. As soon as the gust start time is triggered, there is a noticeable jump in body velocity and descent rate, as presented in Figure 10.11b. Previous analysis has shown that gust length has a minimal effect on the trajectories, as they remain nearly identical even when the gust length is doubled or tripled, as illustrated in Figure 10.4b. Therefore, further analysis is unnecessary.

An increase in gust amplitude enhances parafoil velocity and, consequently, travel distance, though the effect is not significant. However, the traveled distance in the y-axis becomes more pronounced. At the highest gust amplitude, both body velocity and descent rate increase. While this results in greater traveled distances in both the x- and y-axes, the flight time is reduced, as shown in Figures 10.11c and 10.11d.



(a) Parafoil Trajectories for Varying Wind Turbulence Speeds

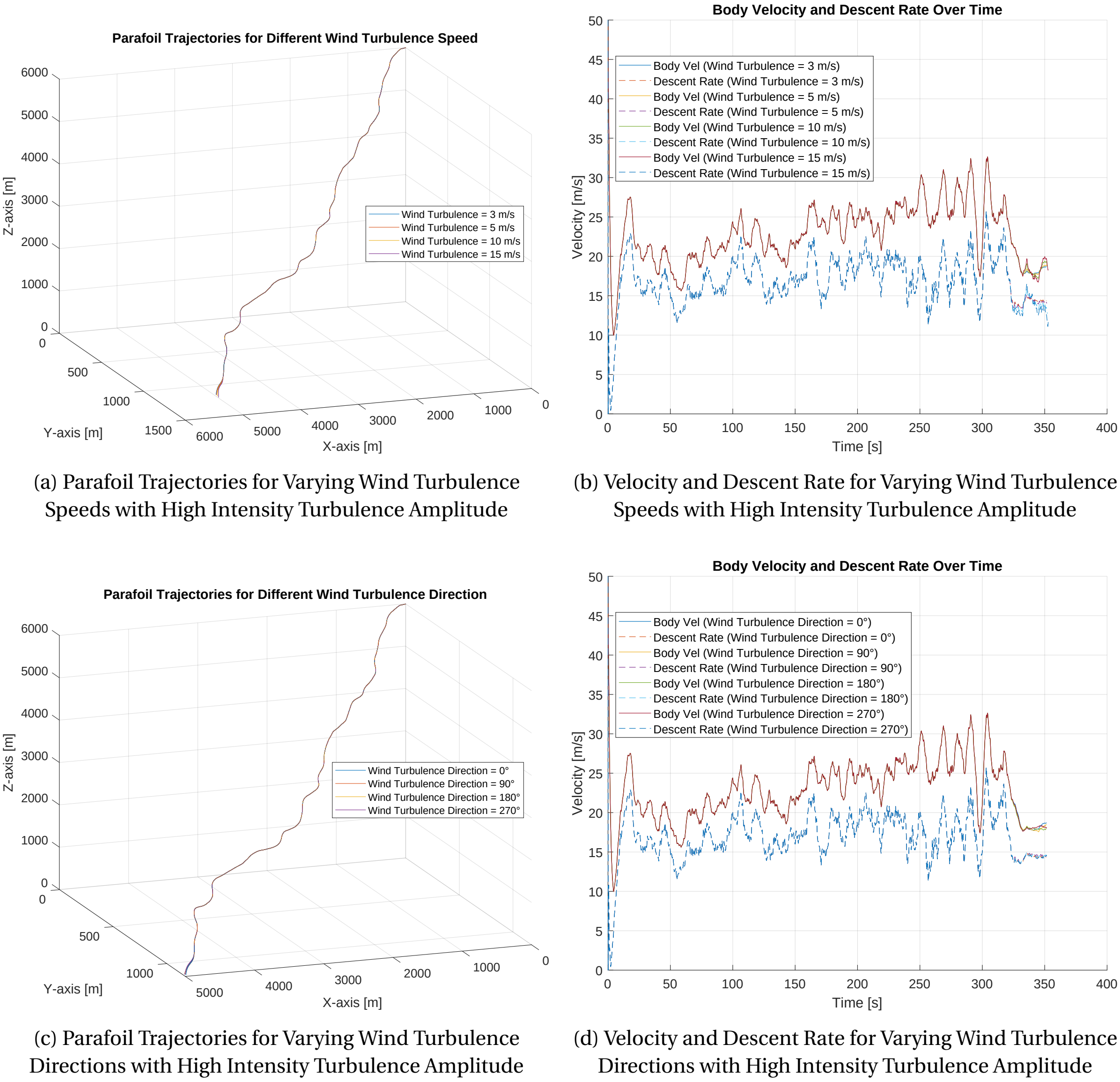
(b) Velocity and Descent Rate for Varying Wind Turbulence Speeds

(c) Parafoil Trajectories for Varying Wind Turbulence Directions

(d) Velocity and Descent Rate for Varying Wind Turbulence Directions

Figure 10.12: Sensitivity Analysis of Wind Turbulence Speeds and Directions: Effects on Parafoil Trajectory, Velocity, and Descent Rate

The sensitivity of wind turbulence speed and its direction on the parafoil trajectory, velocity, and descent rate is analyzed by keeping all other parameters constant, as defined in Table 10.1, while varying the wind turbulence speed and its direction. The results are presented in Figure 10.12. It is clear that increasing the wind turbulence speeds and changing its direction has minimal effect on the trajectory. However, velocity profiles near the Earth's surface exhibit oscillations due to the wind turbulence speeds. However, when the turbulence intensity (magnitude) is set to a high value (see Figure 10.5b), the trajectories and velocity profiles are significantly impacted. The turbulence intensity is obtained from a Simulink lookup table, which provides the intensity as a function of altitude and the probability of exceeding that intensity. Figure 10.13 presents the sensitivity analysis of wind turbulence speed and its direction with high intensity turbulence amplitude on the parafoil trajectory, velocity, and descent rate. As observed, varying the wind turbulence speeds and changing its direction have minimal impact on the trajectory, as the graphs for different wind turbulence speeds and directions remain similar in both magnitude and direction. However, increasing turbulence intensity causes the trajectory to deviate from a straight line, altering both its gradient and direction. The velocity profiles exhibit continuous oscillations. As a result, high turbulence intensity (amplitude) significantly complicates mid-air recovery.



(a) Parafoil Trajectories for Varying Wind Turbulence Speeds with High Intensity Turbulence Amplitude

(b) Velocity and Descent Rate for Varying Wind Turbulence Speeds with High Intensity Turbulence Amplitude

(c) Parafoil Trajectories for Varying Wind Turbulence Directions with High Intensity Turbulence Amplitude

(d) Velocity and Descent Rate for Varying Wind Turbulence Directions with High Intensity Turbulence Amplitude

Figure 10.13: Sensitivity Analysis of Wind Turbulence Speeds and Directions with High Intensity Turbulence Amplitude: Effects on Parafoil Trajectory, Velocity, and Descent Rate

In summary, wind direction significantly influences the trajectory. A tailwind increases the traveled distance along the flight path, while crosswinds cause lateral displacement. Additionally, the impact of wind gusts depends on their starting time, earlier gusts result in a more dispersed trajectory and greater travel distance along all axes. When wind turbulence is introduced, the trajectory becomes more irregular, exhibiting sharper deviations and unpredictable variations, making the flight path less stable. Increasing the rigging angle enhances travel distance, but angles below -10° introduce oscillations. It also raises body velocity, lowers descent rate, and extends flight time. Wind speed and direction significantly affect trajectory.

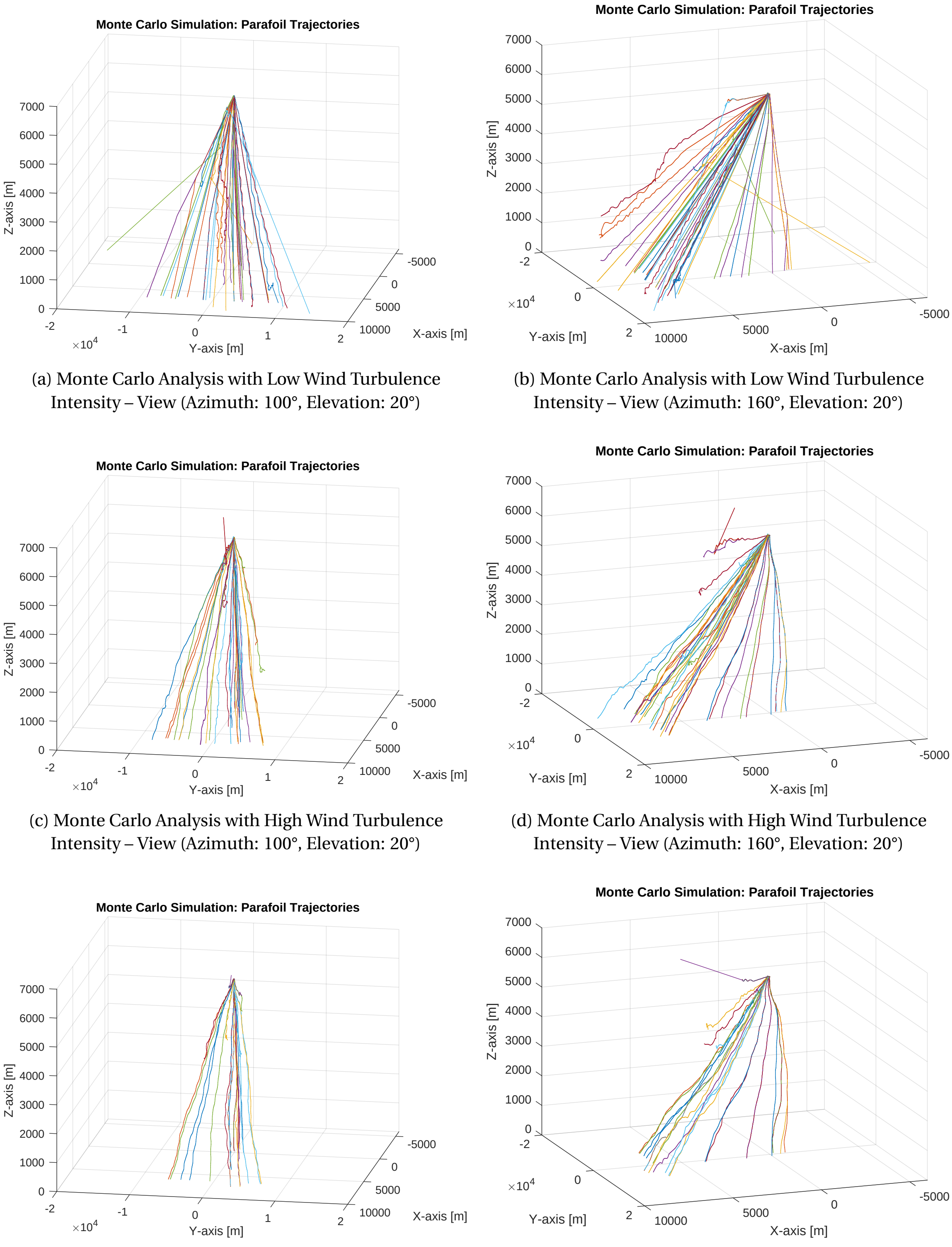
Up to this point, all parameters were held constant, while only one was varied within its specified range. Now, using Monte Carlo analysis, all parameters are varied simultaneously and randomly, with each parameter adjusted within the same specified range as specified in Table 10.2. To speed up the simulation, parallel processing is used. However, when the number of simulations is increased to 100, it still takes over an hour to complete. Figure 10.14 presents the Monte Carlo simulation results for the parameter variations specified in Table 10.2 with different wind turbulence intensities (Low, High, and Severe). Each subfigure corresponds to a different intensity level and is visualized from two distinct viewpoints: (Azimuth: 100°, Elevation: 20°) and (Azimuth: 160°, Elevation: 20°).

Table 10.2: Monte Carlo Simulation Parameters

Parameter	Specified Range
Horizontal Wind Speed [m/s]	[5, 10, 15, 20]
Horizontal Wind Speed Direction [°]	[0, 90, 180, 270]
Gust Start Time [s]	[50, 100, 200, 300]
Gust Amplitude (ug, vg, wg) [m/s]	[3.5, 3.5, 3.0]
Scaling Factor [–]	[1, 2, 3, 4]
Wind Turbulence Speed [m/s]	[3, 5, 10, 15]
Wind Turbulence Speed Direction [°]	[0, 90, 180, 270]
Rigging Angle μ [°]	[-3, -5, -8, -10]

Figures 10.14a and 10.14b present the Monte Carlo simulation results for variations in the parameters outlined in Table 10.2, considering low wind turbulence intensity. In general, the trajectory predominantly follows a straight-line descent. When the wind speeds and directions are aligned, the travel distance increases. Conversely, when the wind is in the opposite direction, the distance traveled is reduced. The sudden change in direction in some trajectories is due to the opposing wind direction, the magnitude of the wind speed, and the start time of the wind gust. When a strong and persistent gust wind is activated, the direction of the trajectory changes abruptly. The same phenomena are observed in Figures 10.14c, 10.14d, 10.14e, and 10.14f; however, in these cases, most trajectories no longer follow a straight line. Instead, they exhibit increased oscillations, specially when compared to Figures 10.14a and 10.14b as the wind turbulence intensity is increased to 'high' and 'severe' levels. Furthermore, the interaction between the varied parameters becomes more pronounced as the trajectories deviate in direction and travel distance. In some cases, unrealistic trajectories emerge, where instead of descending, the parafoil ascends (see Figures 10.14d and 10.14f). While such unrealistic trajectories can be disregarded, the remaining trajectories still provide valuable insights into travel distance and dispersion, making them useful for the purposes of this study.

In summary, Monte Carlo analysis allows for the simultaneous and random variation of all parameters, capturing a wide range of possible trajectories, both realistic and unrealistic. It helps identify maximum travel distances, chaotic behaviors, and the most likely trajectory patterns. This makes it a valuable tool for gaining an overall understanding. However, the conclusions align with previous findings on the model's effects.



(a) Monte Carlo Analysis with Low Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(b) Monte Carlo Analysis with Low Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

(c) Monte Carlo Analysis with High Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(d) Monte Carlo Analysis with High Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

(e) Monte Carlo Analysis with Severe Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(f) Monte Carlo Analysis with Severe Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

Figure 10.14: Monte Carlo Simulation Results for the Parameter Variations Specified in Table 10.2 with Different Wind Turbulence Intensities (Low, High, and Severe) and Viewpoints

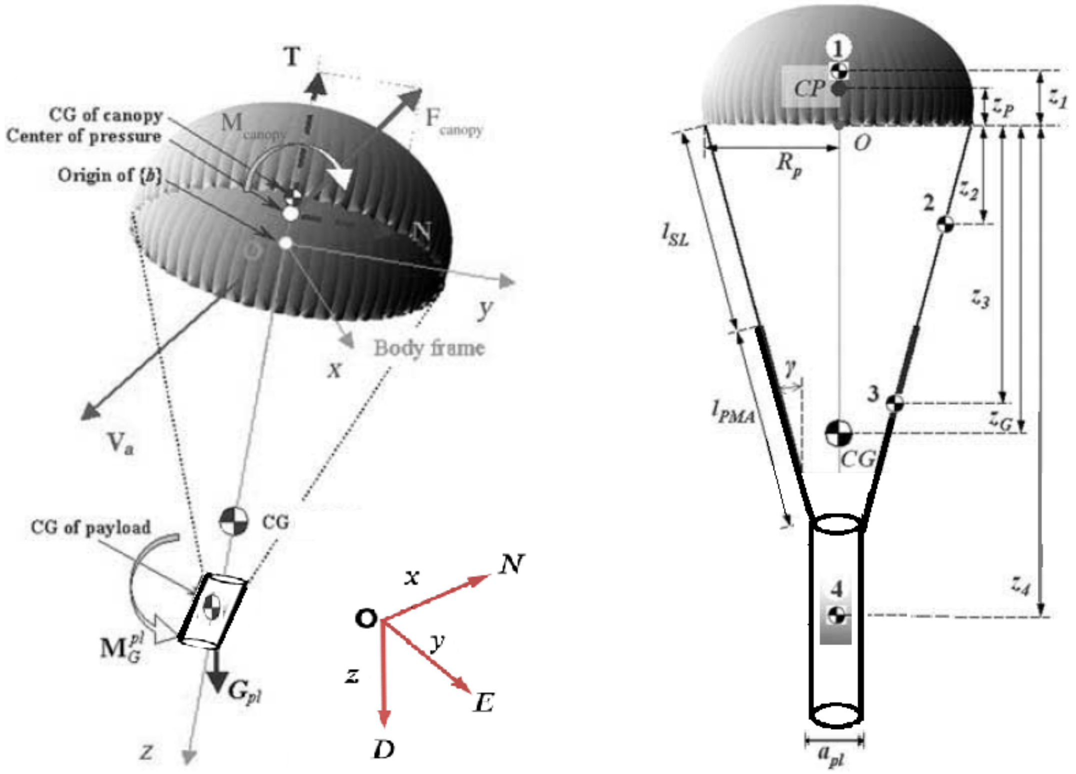
11

Descending Phase of the First Stage: Unguided Circular Parachute Recovery

According to Peter Beck, CEO of Rocket Lab, the cost of refurbishment after splashdown is comparable to mid-air recovery using a helicopter, making the economics neutral. As a result, the easier option of splashdown recovery is chosen [70]. This chapter presents the descending phase of the unguided circular parachute recovery of the first stage.

11.1. Reference Frames and Geometry

The reference frame defined in Section 8.2.1 is also used here, with a few exceptions. The body origin is positioned at the center of the open end of the canopy. The x and y axes lie within a plane parallel to the base of the canopy, while the z axis is aligned with an imaginary line extending toward the centroid of the payload as depicted in Figure 11.1.



(a) Reference Frame of the Circular Parachute

(b) Geometry of the Circular Parachute

Figure 11.1: Reference Frame and Geometry of the Circular Parachute (adapted from [53])

Furthermore, it is assumed that all variables and aerodynamic coefficients, where applicable, are defined in the body frame. Additionally, it is assumed that all aerodynamic forces and moments are applied about the center of pressure of the canopy. The undistorted canopy exhibits axial symmetry around the z_b-axis. In Figure 11.1b, the geometry of the parachute is shown. A set of numbers, from 1 to 4, corresponds to the parachute's canopy, suspension lines (SLs), actuators (PMAs), and payload (the first stage of the Electron rocket), respectively. The coordinates of the centroids Z_i are determined relative to the origin of the body frame. R_0 represents the nominal radius of the uninflated canopy, while R_p denotes the radius of the inflated parachute, which is given by $R_p = \frac{2}{3}R_0$ [53]. Additionally, l_{SL} is the length of the suspension lines, l_{PMA} is the nominal length of the PMAs, and a_{pl} is the diameter of the first stage. Rather than relying on complex mathematical models and detailed drawings to estimate the parachute's weight, the weight and dimensions are derived from existing data in the literature, such as that for the G-12 parachute, which is capable of carrying the first stage of the Electron rocket. The geometric and mass data for the circular parachute, based on the G-12 design, are provided in Tables 11.1, 11.2 and 11.3 [53].

Table 11.1: Geometric Data of G-12 Parachute [53]

Component	Value
Length of suspension lines l_{SL}	15.55 m
Length of pressurized PMA l_{PMA}	5.80 m
Length of vented PMA l^*_{PMA}	7.62 m
Radius of uninflated canopy R_0	9.75 m
Radius of inflated canopy $R_p = \frac{2}{3}R_0$	6.50 m
Canopy shape ratio ϵ	0.82
Cone half-angle γ	15.31°
Diameter of the payload α_{pl}	1.2 m
Length of the payload l_{pl}	12.1 m

Table 11.2: Center of Gravity Data of G-12 Parachute [53]

Z-Coordinate of each component	Value [m]
Centroid of hemispheroidal canopy z_1	-2.8
Centroid of suspension lines z_2	7.5
Centroid of PMAs z_3	17.8
Centroid of payload z_4	26.6
Static mass center of the system z_G	25.2
Center of pressure $z_p = \frac{3}{8}\epsilon R_p$	-2.0

Table 11.3: Mass data of G-12 Parachute [53]

i	Component	Mass [kg]
1	Canopy	22.8
2	Suspension lines	35.3
3	Four PMAs	13.4
4	Payload	950.0
Parachute-payload system (Total)		1021.5

$$z_G = \frac{\sum_{i=1}^{n} m_i \cdot z_i}{\sum_{i=1}^{n} m_i} \tag{11.1}$$

$$z_4 = \cos(\gamma) \cdot (l_{SL} + l_{PMA}) + \frac{l_{pl}}{2} \tag{11.2}$$

The z-coordinates of the center of gravity for each component, as labeled in Figure 11.1, are provided in Table 11.2. These values are based on the G-12 parachute, with the exception of the static mass center of the entire system, z_G, and the centroid of the payload, z_4, which are calculated using Equations 11.1 and 11.2, respectively. The parameters for both equations are in Tables 11.1, 11.2, and 11.3.

Although actuators are not required for the uncontrolled circular parachute and their mass could be neglected, it was decided to include them as part of the total mass to maintain the integrity of the parachute's characteristics, including its geometry, dimensions, aerodynamic coefficients (see Section 11.5), and moment of inertia data. Moreover, if any student intends to extend this tool further by incorporating GNC, the necessary modifications will be limited to the GNC model.

11.2. Environment

The environment model for the circular parachute is the same as that used for the parafoil-payload system in Section 8.4. The Simulink architecture of the environment model is presented in Figure 9.6.

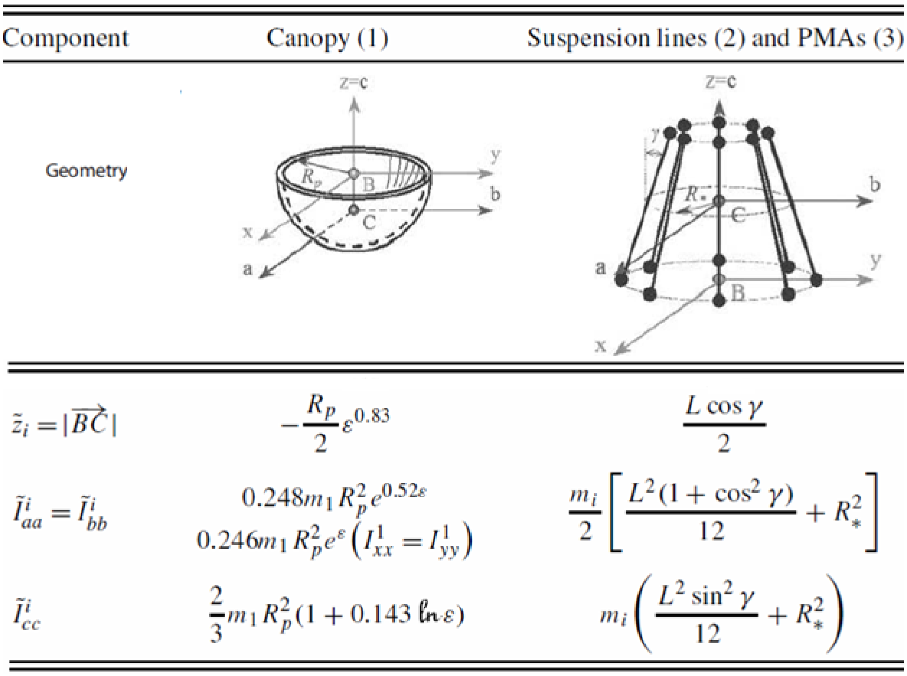
11.3. Mass Moments of Inertia of the Circular Parachute-Payload System

To calculate the mass moment of inertia of the circular parachute-payload system, the individual mass moments of inertia for each component are first computed and then transferred to the origin of the body frame using the parallel axis theorem, as shown in Equation 11.3 [53].

$$I^i_{jj} = \tilde{I}^i_{jj} + m_i \cdot z_i^2, \quad i = 1,\dots,4 \tag{11.3}$$

where I^i_{jj} represents the mass moment of inertia of the i-th component about the body frame axes, and $\tilde{I}^i_{jj}$ denotes the estimated central mass moment of inertia of the i-th component in its centroid-fixed axes, as illustrated in Figure 11.2. m_i is the mass of the i-th component, and z_i is the z-coordinate of the i-th component's centroid in the body frame. The equations required to calculate $\tilde{I}^i_{jj}$ for each component are provided in Figure 11.2. The values of m_i and z_i for each component are given in Tables 11.2 and 11.3, respectively. The mass moment of inertia of the payload is discussed in Section 8.5.4 and can be determined using Equation 8.42. Finally, the moments of inertia for the entire parachute-payload system are calculated as the sum of the inertias of the individual components, as shown in Equation 11.4 [69].

$$I_{jj} = \sum_{i=1}^{4} I^i_{jj} \tag{11.4}$$



Component	Canopy (1)	Suspension lines (2) and PMAs (3)
Geometry		
$\tilde{z}_i = \lvert\overrightarrow{BC}\rvert$	$-\frac{R_p}{2}\varepsilon^{0.83}$	$\frac{L\cos\gamma}{2}$
$\tilde{I}^i_{aa} = \tilde{I}^i_{bb}$	$0.248 m_1 R_p^2 e^{0.52\varepsilon}$ $0.246 m_1 R_p^2 e^{\varepsilon}\left(I^1_{xx} = I^1_{yy}\right)$	$\frac{m_i}{2}\left[\frac{L^2(1+\cos^2\gamma)}{12} + R_*^2\right]$
$\tilde{I}^i_{cc}$	$\frac{2}{3} m_1 R_p^2 (1 + 0.143 \ln\varepsilon)$	$m_i\left(\frac{L^2\sin^2\gamma}{12} + R_*^2\right)$

z=c
R_p
B
C
y
b
x
a
z=c
γ
R_*
C
b
a
y
B
x

Figure 11.2: Relevant Equations for the Mass Moments of Inertia of Parachute Components [53] [69]

In the above figure, R_* for SLs and PMAs can be calculated using 11.5 and **??**, respectively [69].

$$R_* = R_p - 0.5 \cdot l_{\mathrm{SL}} \cdot \sin(\gamma) \tag{11.5}$$

$$R_* = R_p - \left[\sin(\gamma) \cdot \left(l_{\mathrm{SL}} + \frac{l_{\mathrm{PMA}}}{2}\right)\right] \tag{11.6}$$

Fortunately, the mass moments of inertia for the canopy, SLS, and PMAs have already been calculated in the literature, leaving only the calculation for the payload to be addressed [69]. Since the centroid of the (cylindrical) payload is aligned along the Z-axis, the Parallel Axis Theorem simplifies to calculating the mass moment of inertia about the z-axis through the centroid and multiplying the payload's mass by the square of its distance from the z-axis, according to Equation 11.3. The results are presented in Table 11.4.

Table 11.4: Mass Moments of Inertia for the Circular Parachute-Payload System and Its Components $[kg \cdot m^2]$ [69]

i	Component	$\tilde{I}^i_{aa} = \tilde{I}^i_{bb}$	$I^i_{xx} = I^i_{yy}$	$I^i_{zz} = \tilde{I}^i_{cc}$
1	Canopy	365.93	539.16	623.98
2	SLs	1047.09	3032.22	770.76
3	PMAs	53.43	4296.81	36.98
4	Payload	339.16	672 521.16	339.16
Parachute-payload system (Total)			680 389.35	1770.88

The final form of the mass moment of inertia for the entire system is given by Equation 11.7, with its corresponding values listed in the last row of Table 11.4.

$$I_{\text{circular parachute}} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{11.7}$$

11.4. Apparent Mass and Inertia Tensors for the Circular Parachute

The apparent mass of a circular parachute depends on various factors, including the canopy's shape and configuration, spatial angle of attack, acceleration, and porosity. For example, increasing the canopy porosity by 40% can reduce the apparent mass by a factor of 20, while the apparent moment of inertia decreases by a factor of 2.75 [71]. Furthermore, the apparent mass can change by as much as five times its original value when subjected to steady acceleration and when the angle of attack is increased from 0° to 40°, the apparent mass coefficient (α_{33}) decreases by a factor of 4.5 [72]. This demonstrates that the apparent mass and apparent moment of inertia change significantly with the aforementioned parameters. Still, the most common method for estimating these quantities assumes they depend solely on air density, with other potential effects represented by constant multipliers, as provided below [53] [69].

$$\alpha_{11} = k_{11} \cdot m_a, \quad \alpha_{33} = k_{33} \cdot m_a, \quad \alpha_{44} = k_{44} \cdot \tilde{I}_{xxa}, \quad \alpha_{66} = k_{66} \cdot \tilde{I}_{zza}, \quad \alpha_{15} = k_{15} \cdot m_a \cdot z_P \tag{11.8}$$

Different studies report varying values for these multipliers; however, the values used in this thesis are based on the latest research and are given as $k_{11} = 0.5$, $k_{33} = 1.0$, $k_{44} = 0.24$, $k_{15} = 0.75$, and $k_{66} = 0$ [69] [53]. The term m_a represents the air mass trapped within a hemispheroid (added mass), which is half of a spheroid, and it can be calculated using Equation 11.9. Where ρ is the air density and ϵ is the canopy shape ratio (see Table 11.1).

$$m_a = \frac{1}{2} \cdot \frac{4}{3} \cdot \pi \cdot \rho \cdot R_p^3 \cdot \varepsilon \tag{11.9}$$

The mass moments of inertia corresponding to the displaced air by the body about the x-, y-, and z-axes can be computed using Equations 11.10 and 11.11, respectively [69].

$$\tilde{I}_{xxa} = I^{air}_{xx} = I^{air}_{yy} = \frac{1}{5} \cdot m_a \cdot R_p^2 \cdot \left(1 + \varepsilon^2\right) \tag{11.10}$$

$$\tilde{I}_{zza} = I^{air}_{zz} = \frac{2}{5} \cdot m_a \cdot R_p^2 \tag{11.11}$$

Due to the axial symmetry of the circular canopy, the following relationships hold: $\alpha_{22} = \alpha_{11}$, $\alpha_{55} = \alpha_{44}$, and $\alpha_{24} = 2\alpha_{15}$ [53]. The apparent mass and inertia tensors for the circular parachute can be determined using Equations 11.12 and 11.13, respectively [69].

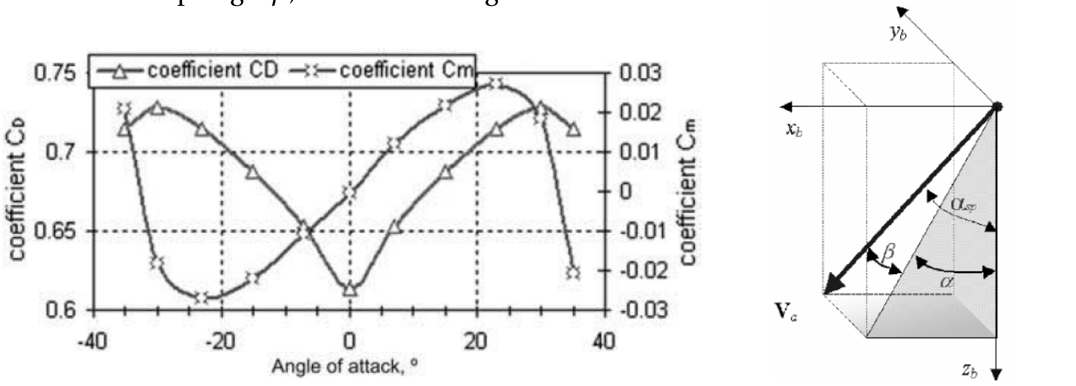
$$\boldsymbol{M}_{f_{\text{circular}}} = \begin{bmatrix} \alpha_{11} & 0 & 0 \\ 0 & \alpha_{22} & 0 \\ 0 & 0 & \alpha_{33} \end{bmatrix} \tag{11.12}$$

$$\boldsymbol{I}_{f_{\text{circular}}} = \begin{bmatrix} \alpha_{44} & 0 & 0 \\ 0 & \alpha_{55} & 0 \\ 0 & 0 & \alpha_{66} \end{bmatrix} \tag{11.13}$$

The diagonal elements in Equation 11.12 represent the apparent masses of the air, which remain virtually stagnant within and beneath the canopy, while the diagonal elements in Equation 11.13 represent the apparent mass moments of inertia of the displaced air by the body.

11.5. Aerodynamic Coefficients and Angles of the Circular Parachute

The dimensionless aerodynamic drag and moment coefficients are likewise obtained from the G-12 parachute data and are plotted against the angle of attack in Figure 11.3a [53]. They play a crucial role in calculating the aerodynamic forces and moments acting on the parachute. These aerodynamic coefficients are used to generate a lookup table in Simulink with Akima spline interpolation, similar to the lookup tables created for the ascent phase in Section 3.5. The airspeed vector V_a is defined in the same manner as in the parafoil-payload system and can be computed using Equation 8.56. However, the aerodynamic angles are defined slightly differently, as they are referenced from the z_b-axis for the circular parachute, while for the parafoil-payload system, they are referenced from the x_b-axis. The spatial angle of attack α_{sp} and its components, the angle of attack α and the sideslip angle β, are shown in Figure 11.3b.



(a) Aerodynamic Coefficients of the G-12 Parachute

(b) Aerodynamic Angles Illustration

Figure 11.3: G-12 Circular Parachute Aerodynamic Coefficients and Illustration of Aerodynamic Angles [53]

By using the components of the airspeed vector V_a, the spatial angle of attack α_{sp}, the angle of attack α, and the sideslip angle β can be calculated using Equations 11.14, 11.15, and 11.16, respectively. Note that V_a represents the airspeed vector, which is expressed in the body frame.

$$\alpha_{\text{sp}} = \cos^{-1}\left(\frac{w_a}{\sqrt{u_a^2 + v_a^2 + w_a^2}}\right) \tag{11.14}$$

$$\alpha = \tan^{-1}\left(\frac{u_a}{w_a}\right) \tag{11.15}$$

$$\beta = \tan^{-1}\left(\frac{v_a}{\sqrt{u_a^2 + w_a^2}}\right) \tag{11.16}$$

11.6. Equations of Motion for the Circular Parachute

The equations of motion for the circular parachute use the Simulink block *Custom Variable Mass 6DOF (Euler Angles)*, which implements the Euler angle representation of six-degree-of-freedom equations of motion for a custom variable mass (see Figure 11.4). The same Simulink block was used for the parafoil-payload system, which means the equations of motion remain unchanged, with the exception of those governing the moment of inertia (Equation 8.71), external forces and moments (Section 8.6.2), and the equations describing the effect of apparent mass and inertia. Consequently, Equations 8.65 through 8.73 remain the same. The new equations for inertia, external forces and moments, and those describing the effect of apparent mass and apparent inertia for the circular parachute are presented in this Section, as they differ from those of the parafoil-payload system.

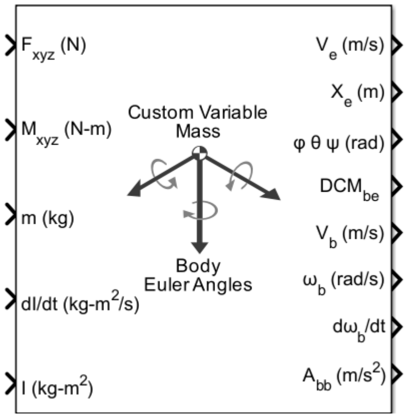


Figure 11.4: Custom Variable Mass 6-DOF (Euler Angles) [52]

11.6.1. External Forces and Moments

The total external forces and moments acting on the parachute, arising from aerodynamic effects and the weight of the system components, are expressed in the body frame and can be calculated using Equations 11.17 and 11.18, respectively [53].

$$F = F^{a/d} + G + F_{\text{buoyancy}} + F_{\text{apparent}} \tag{11.17}$$

$$M = M^{a/d} + \sum_i M_G^i + M_{\text{buoyancy}} + M_{\text{total, apparent}} \tag{11.18}$$

Where $F^{a/d}$ and $M^{a/d}$ represent the total aerodynamic forces and moments, respectively, and are determined using Equations 11.19 and 11.20. G represents the gravitational force vector resulting from Earth's gravity g, expressed in the body frame, and is determined using Equation 11.21. M_G^i denotes the moments generated by the weight of each component when translated to the origin of the body frame, and can be computed using Equation 11.22.

$$F^{a/d} = F_{\text{canopy}} + F_{\text{risers}} \tag{11.19}$$

$$M^{a/d} = M_{\text{canopy}} + M_{\text{risers}} \tag{11.20}$$

$$G = {}_b^nR^T \left[0, 0, mg\right]^T \tag{11.21}$$

$$M_G^i = P_i \times G^i \tag{11.22}$$

Here, m represents the mass of the entire system. F_{risers} and M_{risers} are the force and moment vectors caused by changes in riser length. These terms are relevant in the context of controlling non-gliding parachute systems. However, in the case of an unguided circular parachute, where control is not implemented, F_{risers} and M_{risers} are considered to be zero. The rotation matrix is orthogonal; therefore, its inverse equals its transpose, i.e., ${}^{n}_{b}R^{T} = {}^{b}_{n}R$. The matrix ${}^{b}_{n}R$ defines the transformation from the navigational reference frame (flat Earth) to the body reference frame, as given by Equation 8.8. P_i is the position vector of the i-th component relative to the origin of the body frame (see Tables 11.1, 11.2 and 11.3), and G^i is the gravitational force vector acting on the mass of the i-th component due to Earth's gravity, calculated using Equation 11.21. It is important to note that the mass of the i-th component is used in the calculation, rather than the total system mass. The aerodynamic force and moment vectors acting on the undisturbed canopy are computed using Equations 11.23 and 11.24, respectively.

$$F_{\text{canopy}} = C_D\left(\alpha_{sp}\right) \cdot Q \cdot S_0 \cdot \frac{V_a}{\|V_a\|} \tag{11.23}$$

$$M_{\text{canopy}} = 2 \cdot Q \cdot S_0 \cdot R_0 \cdot \begin{bmatrix} C_{\text{roll}} \\ C_{\text{pitch}} \\ C_n \end{bmatrix} \tag{11.24}$$

$$Q = \frac{1}{2} \cdot \rho \cdot \|V_a\|^2 \tag{11.25}$$

Where Q is the dynamic pressure, ρ is the air density, $S_0 = \pi \cdot R_0^2$ is the canopy's reference area, R_0 is the canopy's uninflated nominal radius and $\|\cdot\|$ represents the Euclidean norm of a vector. The moment coefficients in Equation 11.24 were initially formulated to have the form $\begin{bmatrix} C_l(\beta) & C_m(\alpha) & C_n \end{bmatrix}^T$, however, White and Wolf demonstrated that the longitudinal and lateral motions of a symmetric parachute are sufficiently decoupled, allowing them to be analyzed separately [73]. Therefore, this assumption implies that roll and pitch motions have identical moment characteristics, meaning $C_{\text{roll}} = C_m(\beta)$ and $C_{\text{pitch}} = C_m(\alpha)$. Both moment coefficients have identical values as $C_l(\beta) = C_m(\alpha)$ and their values are provided in Figure 11.3a. Furthermore, for a symmetric body, the yaw moment coefficient due to yaw rate, C_n, is equal to zero. Furthermore, in Equations 11.17 and 11.18, the terms F_{buoyancy} and M_{buoyancy} represent the forces and moments due to buoyancy, which are small upward forces generated by the added mass m_a, and can be calculated using Equations 11.26 and 11.27, respectively.

$$F_{\text{buoyancy}} = {}^{n}_{b}R^{T} \begin{bmatrix} 0 \\ 0 \\ -m_a \cdot g \end{bmatrix} \tag{11.26}$$

$$M_{\text{buoyancy}} = P_{canopy} \times F_{\text{buoyancy}} \tag{11.27}$$

where $P_{\text{canopy}} = [0, 0, z_1]^T$, with z_1 representing the centroid of the hemispheroidal canopy, as provided in Table 11.2. Additionally, m_a is calculated using Equation 11.9. With the apparent mass and inertia tensors determined using Equations 11.12 and 11.13, they must be incorporated into the equations of motion. However, the Simulink block *Custom Variable Mass 6-DOF (Euler Angles)*, which implements the 6-DOF equations of motion (see Figure 11.4), does not accept mass as a 3×1 matrix but only as a scalar. As a result, the apparent forces and moments cannot be directly included. Instead, they are computed separately and added as external inputs (forces and moments). To maintain mathematical correctness, their signs must be reversed, as they are effectively moved from inside Equations 8.65 and 8.70 to the outside, making them part of Equations 11.17 and 11.18. The apparent forces and apparent mass moment of inertia are computed using Equations 11.28 and 11.29, respectively.

$$F_{\text{apparent}} = -\left[M_{f_{\text{circular}}} \cdot \left(\dot{V}_b + \omega_b \times V_b\right)\right] \tag{11.28}$$

$$M_{\text{total, apparent}} = -\left[I_{f_{\text{circular}}} \cdot \dot{\omega}_b + \omega_b \times \left(I_{f_{\text{circular}}} \cdot \quad \omega_b\right) + \dot{I}_{f_{\text{circular}}} \cdot \omega_b\right] + M_{\text{offset}} \tag{11.29}$$

with

$$M_{\text{offset}} = P_{\text{CP}} \times F_{\text{apparent}} \tag{11.30}$$

where the offset moment M_{offset} arises because the center of pressure (CP) is not aligned with the origin of the body frame. Here, $P_{\text{CP}} = \left[0, 0, z_p\right]^T$, with z_p representing the z-coordinate of the center of pressure, as provided in Table 11.2. V_b and ω_{b} are the body's velocity and angular rates, respectively, expressed in the body reference frame. They are defined by Equations 8.66 and 8.67.

11.7. Verification and Validation

With the models now ready for simulation, the next step is verification and validation. As noted in Section 8.8, the recovery data for the Electron rocket is unfortunately not publicly available, preventing direct comparison of the simulation results for validation. The same limitation applies to the circular parachute as it does to the parafoil. Consequently, as discussed in Section 8.8, direct validation of the simulation results is not feasible. Although data is available for G-12 circular parachute drops, it primarily comes from tests with GNC models, whereas this thesis focuses on an unguided circular parachute. The only available data for an unguided parachute comes from a point mass model with a 3-DOF EOM, whereas this study uses 6-DOF EOM. The use of the 3-DOF EOM is due to the singularity-induced errors with their 6-DOF EOM [74]. More information on singularity errors in 6-DOF EOM can be found in Appendix E. Furthermore, several key pieces of information are missing in the data, making it unsuitable for validation. The wind values were measured, but they differ significantly from the wind models used in this study. Additionally, the wind direction is provided in a scattergram of wind vectors, making it extremely difficult to reconstruct the wind conditions and plot them against altitude. Furthermore, the 3-DOF point mass model drop test was conducted from an altitude of 2.5 km, whereas this study involves parachute deployment from an altitude of 6 km with specific initial conditions. The initial conditions of the drop test were unclear. Reconstructing the data to fit the models used in this study proved to be a challenging, if not impossible, task due to the limited information available. Given the substantial differences, the available data does not directly apply to this preliminary study. That said, the mathematical equations governing the aerodynamic forces and moments have been verified and match those in the literature [53] [69]. Therefore, in the absence of real flight data, this approach is considered the best available option and is assumed to be valid for the purposes of this study.

12

Simulink Architecture for the Circular Parachute-Payload System Descent Model

This chapter presents the Simulink architecture for the circular parachute-payload system descent trajectory model. The model comprises many subsystems, each of which will be presented in detail. The corresponding MATLAB code for each subsystem is provided in Appendix B.

Figure 12.1 presents an overview of the Simulink architecture for the parafoil-payload system descent model. Detailed overviews of each subsystem will also be presented.

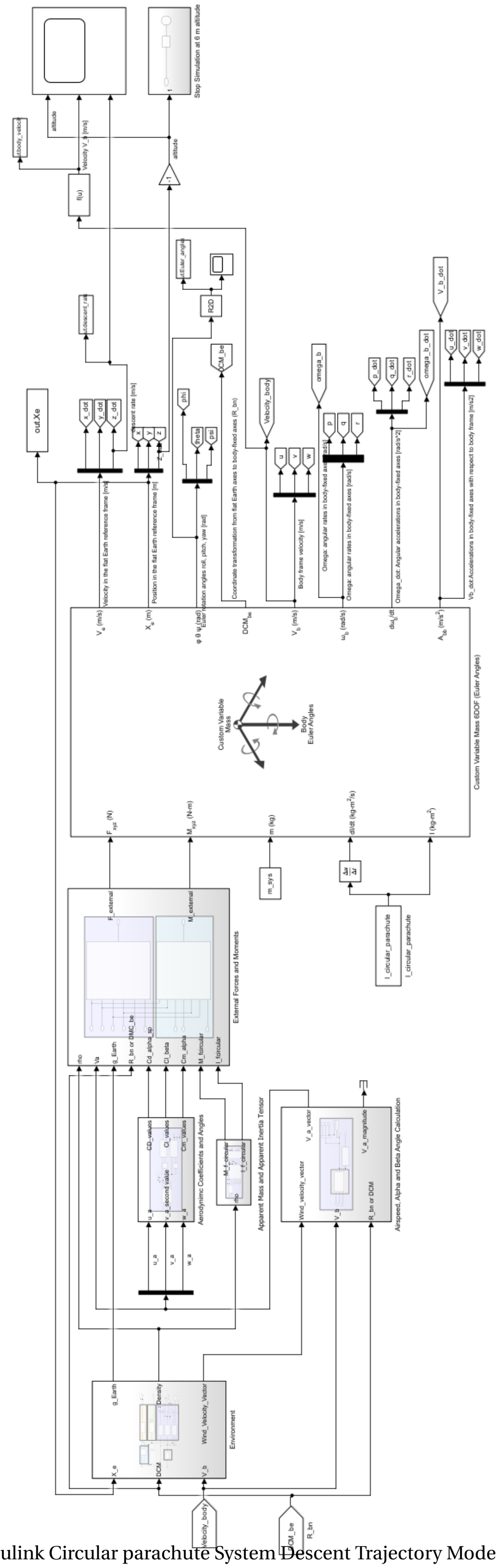


Figure 12.1: Simulink Circular parachute System Descent Trajectory Model Overview

Figure 12.2 presents the aerodynamic coefficient model

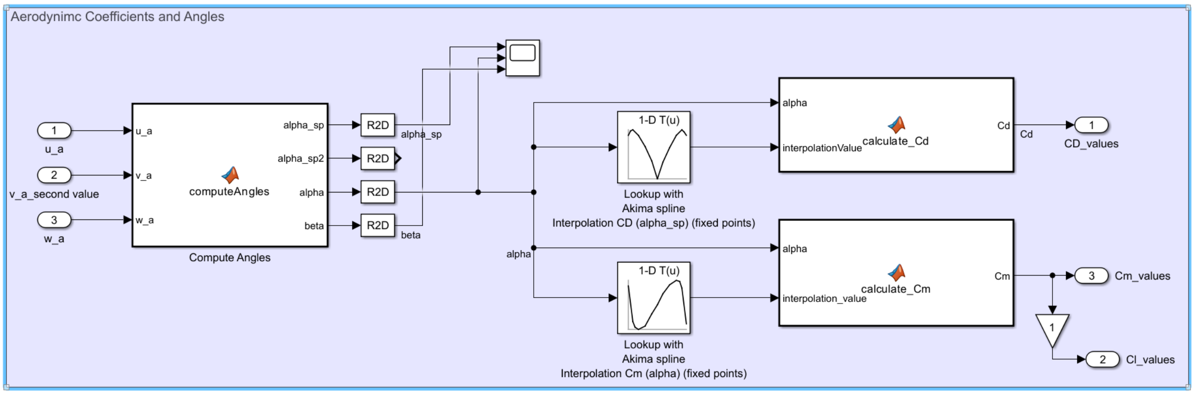


Figure 12.2: aerodynamic coefficient model

Figure 12.3 presents the apparent mass and inertia tensor model

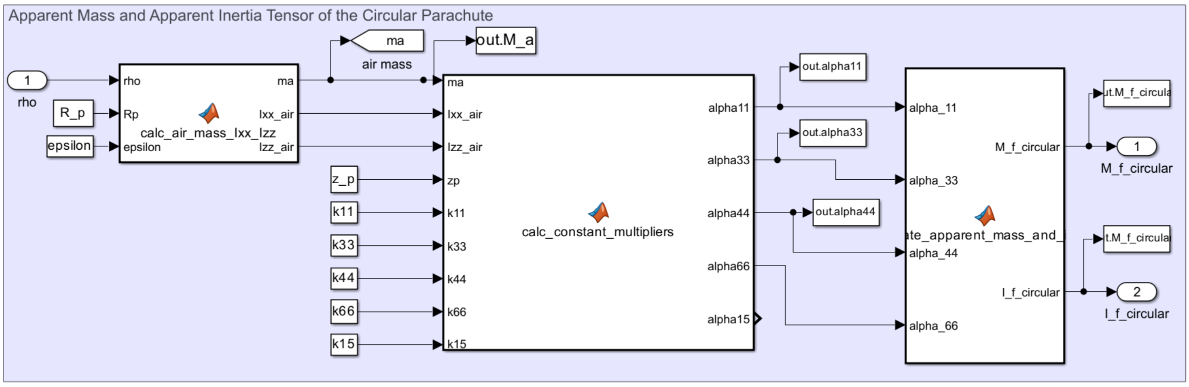


Figure 12.3: apparent mass and inertia tensor model

Figure 12.4 presents the airspeed and aerodynamic angle model

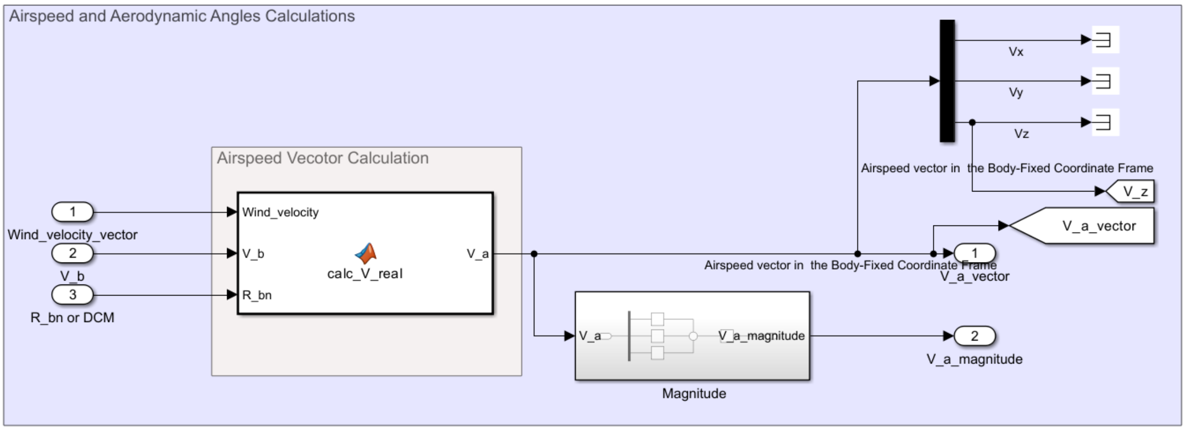

Airspeed and Aerodynamic Angles Calculations
Airspeed Vecotor Calculation
1
Wind_velocity_vector
2
V_b
3
R_bn or DCM
Wind_velocity
V_b
R_bn
calc_V_real
V_a
Vx
Vy
Vz
Airspeed vector in the Body-Fixed Coordinate Frame
V_z
V_a_vector
Airspeed vector in the Body-Fixed Coordinate Frame
1
V_a_vector
V_a
V_a_magnitude
Magnitude
2
V_a_magnitude

Figure 12.4: airspeed and aerodynamic angle model

Figure 12.5 presents the external forces and moments model

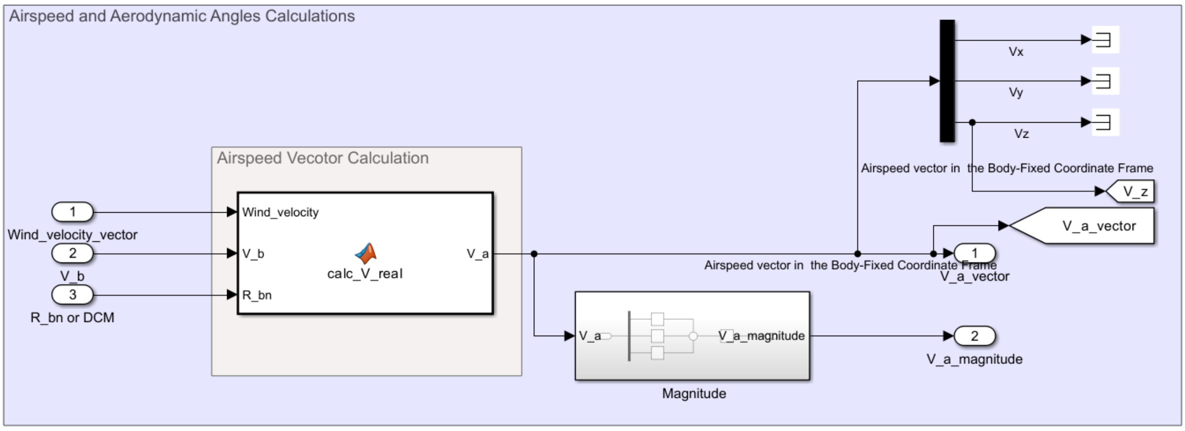

Airspeed and Aerodynamic Angles Calculations
Airspeed Vecotor Calculation
1
Wind_velocity_vector
2
V_b
3
R_bn or DCM
Wind_velocity
V_b
R_bn
calc_V_real
V_a
Vx
Vy
Vz
Airspeed vector in the Body-Fixed Coordinate Frame
V_z
V_a_vector
Airspeed vector in the Body-Fixed Coordinate Frame
1
V_a_vector
V_a
V_a_magnitude
Magnitude
2
V_a_magnitude

Figure 12.5: external forces and moments model

Figure 12.6 presents the external forces model

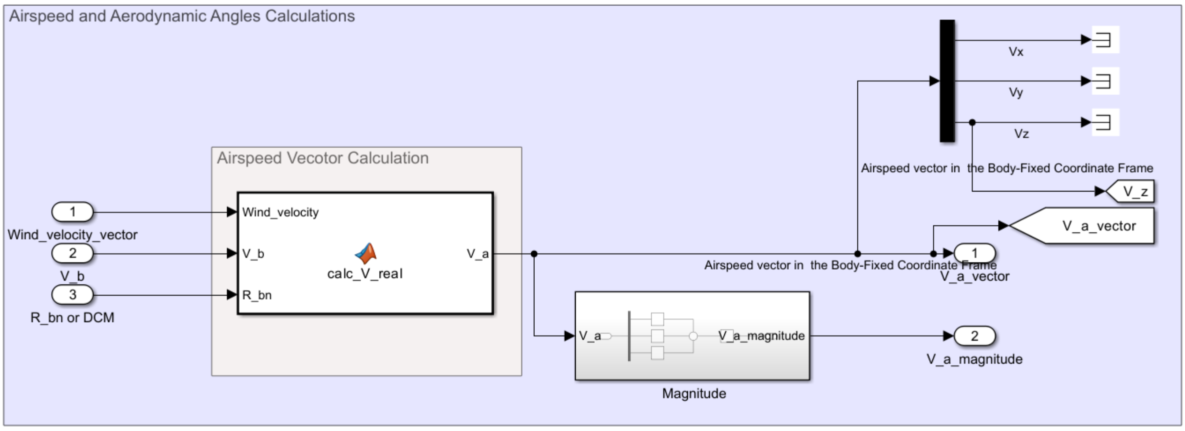

Airspeed and Aerodynamic Angles Calculations
Airspeed Vecotor Calculation
1
Wind_velocity_vector
2
V_b
3
R_bn or DCM
Wind_velocity
V_b
R_bn
calc_V_real
V_a
Vx
Vy
Vz
Airspeed vector in the Body-Fixed Coordinate Frame
V_z
V_a_vector
Airspeed vector in the Body-Fixed Coordinate Frame
1
V_a_vector
V_a
V_a_magnitude
Magnitude
2
V_a_magnitude

Figure 12.6: external forces model

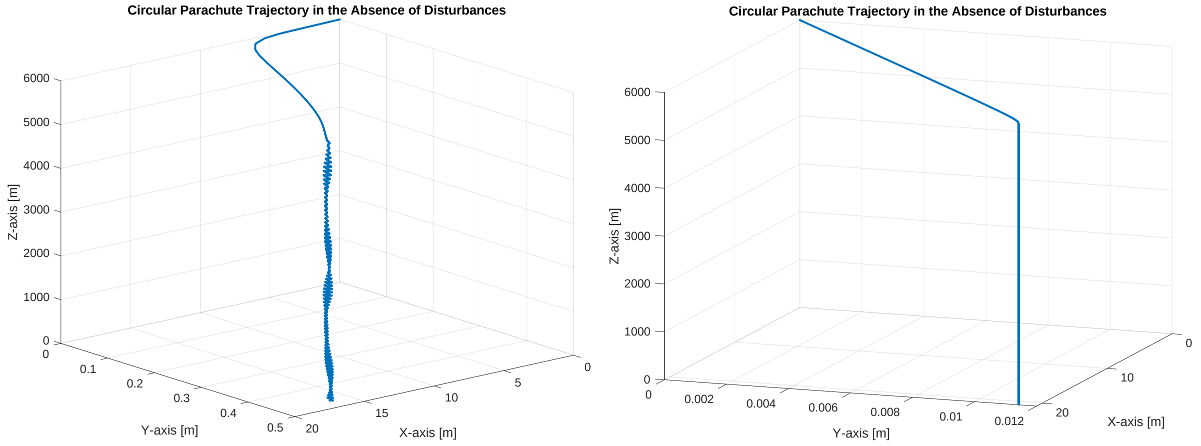
13

Results of the Descent Phase: First Stage Recovery with an Unguided Circular Parachute

This chapter presents the results and sensitivity analysis of the descent phase of the Electron rocket, focusing specifically on the unguided circular parachute based recovery system for the first stage. Table D.1 presents the initial conditions, which include position, velocity, Euler orientation, and angular velocity, derived from the descent trajectory of the first stage with the deployed ballute, just prior to circular parachute deployment, at an altitude of 6 km, as described in Section 8.1. The results are detailed in Section 13.1, while the sensitivity analysis is discussed in Section 13.2.

13.1. Results of the Uncontrolled Circular Parachute Descent Trajectory Model

First, the descent trajectory of the circular parachute, as shown in Figure 13.1a, is presented without considering disturbances such as wind velocity, wind gusts, and turbulence. The corresponding body velocity, descent rate and altitude over time plots are presented in Figure 13.2. The circular parachute stabilizes its trajectory by mitigating the effects of initial conditions and begins descending. While no external disturbances are applied, the influence of apparent mass, apparent mass moment of inertia, buoyancy forces and moments, and moments due to gravity cause the trajectory to oscillate side to side.

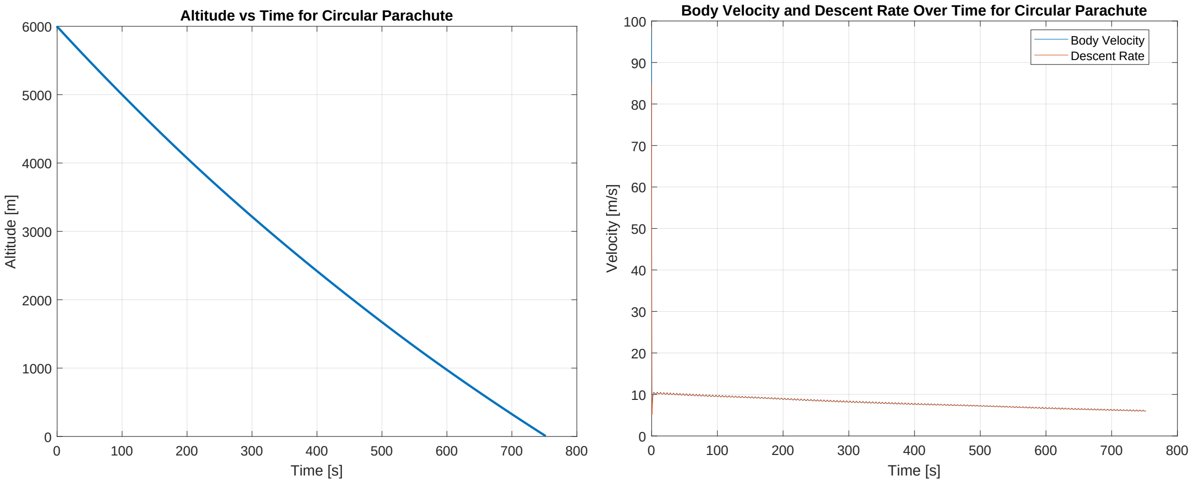


(a) Circular Parachute Descent Trajectory with

(b) Circular Parachute Descent Trajectory

Figure 13.1: Circular Parachute Descent Trajectory in the Absence of Disturbances

For comparison, Figure 13.1b illustrates the trajectory by neglecting the effects of apparent mass, apparent mass moment of inertia, buoyancy forces, and moments, and with all masses contributing only to moments due to gravity at the body's center of gravity. The trajectory exhibits no oscillation and descends along a straight path. The descent rate and body velocity in Figure 13.2 are slow enough for a safe splashdown, with both values approximately 6 m/s at the end of the simulation, occurring at 752 seconds (or 12.53 minutes).



(a) Circular Parachute Altitude Over Time

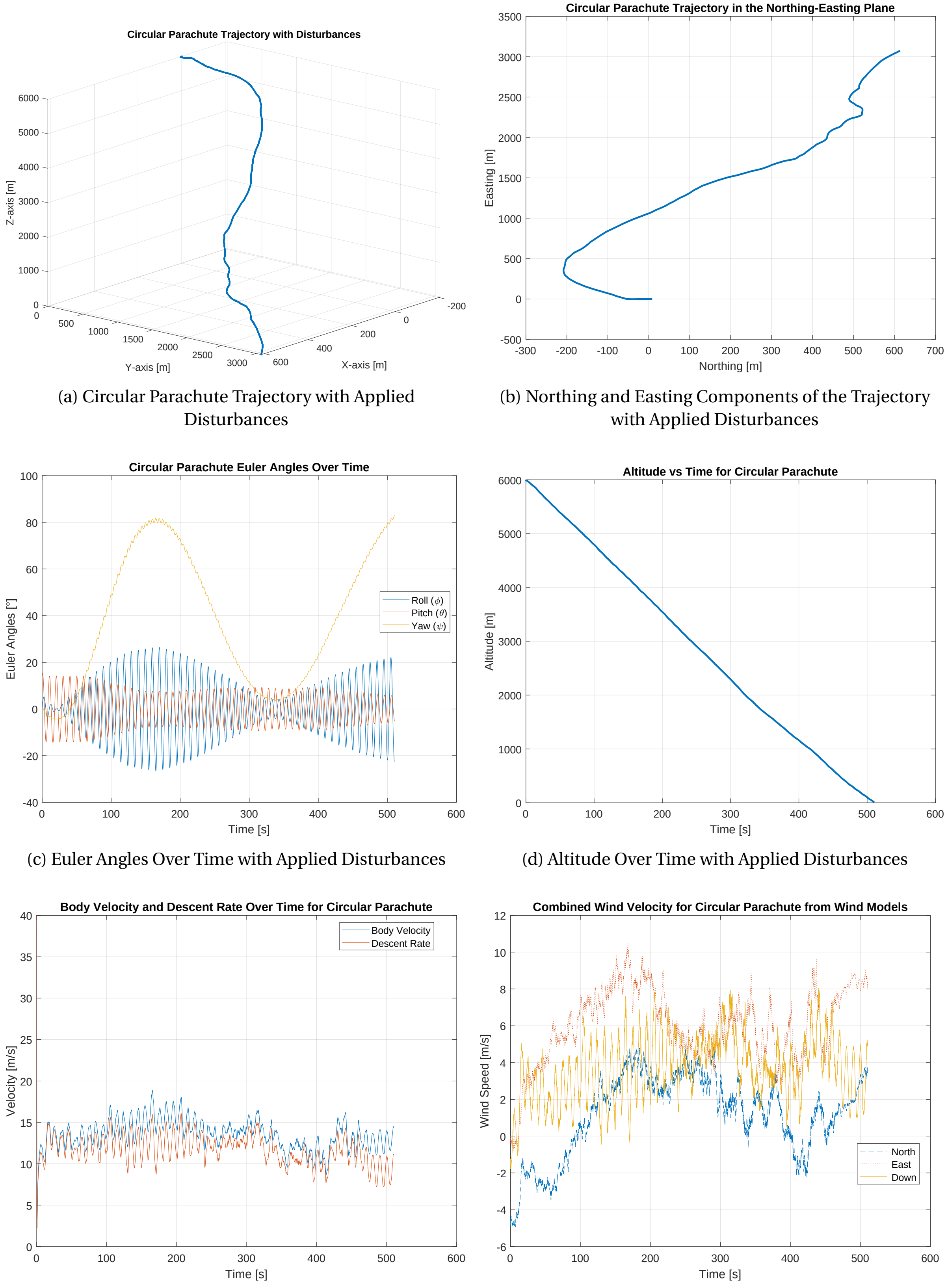
(b) Body Velocity and Descent Rate Over Time

Figure 13.2: Circular Parachute Altitude, Body Velocity and Descent Rate Over Time in the Absence of Disturbances

The approach outlined in Section 10.2 will be applied here, utilizing the same input parameters, including those for the *Horizontal Wind Model*, *Discrete Wind Gust Model*, and *Von Kármán Wind Turbulence Model*, as summarized in Table 10.1. The results of the trajectory with the applied disturbances, as shown in Figure 10.6, are presented in Figure 13.3. The combined wind velocity from different models, as detailed in Section 8.4, is shown in Figure 13.3f and is used in the circular parachute Simulink model. Although the wind model input parameters were identical for both the parafoil and circular parachute, the effect is more pronounced on the circular parachute, with the generated wind velocity components having approximately 1.5 times larger and has has more oscillations, as seen in Figure 10.6d.

The trajectory is shown in Figure 13.3a, while the Northing and Easting components of the trajectory are presented in Figure 13.3b. Initially, the parachute is pushed back in the negative North direction due to horizontal headwind. Under the influence of external forces, it then turns up to 80 °towards the positive East direction, as seen in Figures 13.3b and 13.3c, before the cycle reverses. Although the circular parachute lacks a wing-shaped canopy like the parafoil, wind still has a significant impact on its motion, causing it to travel approximately 3000 m in the east direction. This is notably less than the parafoil's travel distance of 4500 m against the headwind, as shown in Figure 10.7b.

The descent rate is around 10 m/s as seen in Figure 13.3e, which is also slow enough for a safe splashdown at the end of the simulation, occurring at 510 seconds (or 8.5 minutes).



(a) Circular Parachute Trajectory with Applied Disturbances

(b) Northing and Easting Components of the Trajectory with Applied Disturbances

(c) Euler Angles Over Time with Applied Disturbances

(d) Altitude Over Time with Applied Disturbances

(e) Circular Parachute Velocity and Descent Rate with Applied Disturbances

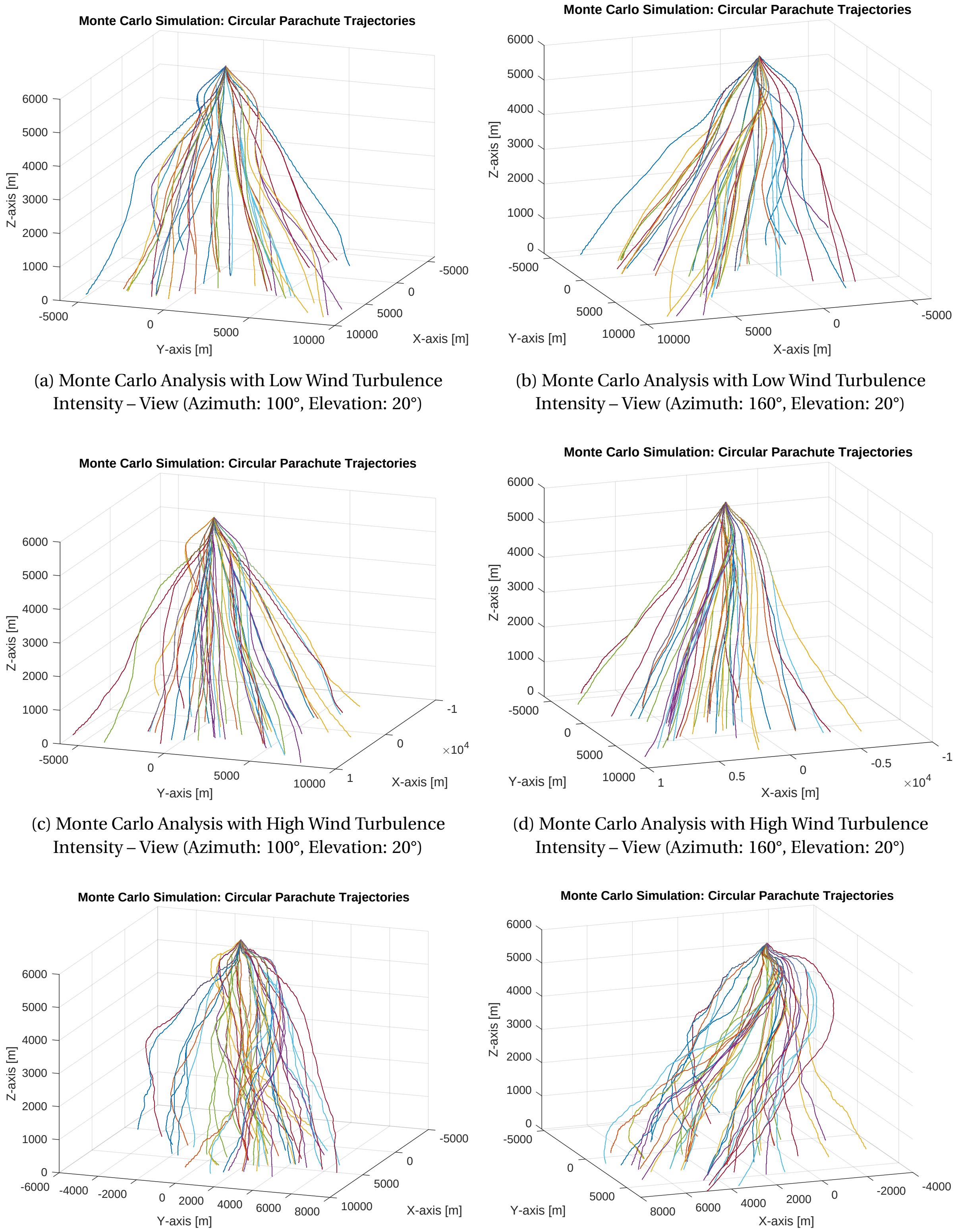
(f) Combined Wind Speed Components from All Disturbance Models

Figure 13.3: Trajectory of the Circular Parachute, Showing the Impact of Disturbances: (a) Trajectory Under Disturbances, (b) Northing and Easting components of Trajectory, (c) Euler Angles Over Time, (d) Altitude Over Time, (e) Body Velocity and Descent Rate Over Time, (f) Combined Wind Speed Components from All Disturbance Models

13.2. Sensitivity Analysis of Circular Parachute Performance

The same approach used in the parafoil sensitivity analysis in Section 10.3 will be applied here. In this Monte Carlo analysis, all parameters are varied simultaneously and randomly, with each parameter adjusted within the specified range outlined in Table 10.2. Again, parallel processing is used to speed up the simulation. It is observed that the simulation takes significantly less time than the one for the parafoil analysis. Figure 13.4 presents the Monte Carlo simulation results for the parameter variations specified in Table 10.2 with different wind turbulence intensities (Low, High, and Severe). Each subfigure corresponds to a different intensity level and is visualized from two distinct viewpoints: (Azimuth: 100°, Elevation: 20°) and (Azimuth: 160°, Elevation: 20°).

In general, trajectories with low turbulence intensities are less chaotic compared to those with high turbulence intensities. The dispersion from the deployment point reaches a maximum of 8000 meters, whether in the north or east direction. Interestingly, despite the higher turbulence intensities leading to more chaotic trajectories, the dispersion (i.e., the traveled distance in the north or east direction) is smaller compared to lower turbulence conditions. This can be attributed to the direction of the wind, whether it's a headwind, tailwind, or crosswind. A tailwind increases the travel distance, while a headwind decreases it, similar to the behavior observed in parafoil trajectories.



(a) Monte Carlo Analysis with Low Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(b) Monte Carlo Analysis with Low Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

(c) Monte Carlo Analysis with High Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(d) Monte Carlo Analysis with High Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

(e) Monte Carlo Analysis with Severe Wind Turbulence Intensity – View (Azimuth: 100°, Elevation: 20°)

(f) Monte Carlo Analysis with Severe Wind Turbulence Intensity – View (Azimuth: 160°, Elevation: 20°)

Figure 13.4: Monte Carlo Simulation Results of Circular Parachute for the Parameter Variations Specified in Table 10.2 with Different Wind Turbulence Intensities (Low, High, and Severe) and Viewpoints

14
Conclusion and Recommendations

The research objective of this thesis is to develop a simulation tool for analyzing the launch performance of the Electron rocket and its first-stage recovery, involving mid-air recovery using an air-ram parachute and splashdown recovery using an unguided circular parachute. For the ascent phase, a two-stage rocket Simulink model is developed that implements quaternion representation of six-degrees-of-freedom (6-DOF) equations of motion for custom variable mass in the Earth-centered Earth-fixed (ECEF) reference frame to simulate ascent trajectories. Another Simulink model is created to simulate the ballistic trajectory of the first stage after separation using quaternion representation of 6-DOF equations of motion for constant mass in the ECEF reference frame, particularly when first stage recovery is not implemented.
For the descent phase, two distinct Simulink models are developed. One implements Euler angle representation of 6-DOF equations of motion for custom variable mass with an unguided circular parachute to simulate the descent trajectory, while the other simulates the same equations for a steerable (guided) parafoil

The ascent trajectory model is validated by comparing simulation results with past mission data of Electron. The simulation results for the ascent trajectory show a close match to reference profiles for altitude and velocity, with small errors in the early stages (e.g., -4.3% at first stage burnout for altitude and 14% at 84 seconds for velocity), which decrease over time, confirming the model's reliability and validation. The simulation results show that all ascent trajectory constraints were met. The maximum dynamic pressure was 26.7 kPa, well below the 35 kPa limit. The maximum heat flux after fairing separation was 240 W/m^2, far within the 1135 W/m^2 constraint. The maximum axial acceleration reached 4.65g, which is under the 7.5g limit. The first stage burned out at 150.2 seconds, followed by a 5-second coasting phase, while the second stage operated for 406 seconds, with the vehicle's thrust ending after 561.2 seconds due to differences in engine configuration.

The simulation shows a velocity of 7.821 km/s at second stage burnout, a 0.4% error compared to orbital velocity, with altitude stabilizing at 200 km after a slight overshoot, and an initial orbit eccentricity of 0.0069, while also demonstrating the feasibility of north-to-south launches with yaw angle control. The Hohmann transfer orbit used a total ΔV of 171.9 m/s, requiring 13.32 kg of propellant.

The sensitivity analysis of the ascent trajectory model uses the one-at-a-time approach, varying payload mass while keeping other parameters constant. Results show that a 500 kg payload reaches a burnout velocity of 6226 m/s, insufficient to achieve a stable orbit. A 300 kg payload reaches 7070 m/s but still falls short of the required orbital velocity. A 200 kg payload reaches 7660 m/s at burnout but also fails to reach a stable orbit, potentially due to an unoptimized pitch profile.

The simulation results show that the apparent mass in the z-direction is significantly larger than in the x- and y-directions, exceeding the parafoil's mass of 73.5 kg, with the added mass ranging from 30 kg to 57 kg, which is 40% to 77% of the parafoil's mass. Thus including them is indeed required for better accuracy.

In general, wind direction significantly influences the trajectory. A tailwind increases the traveled distance along the flight path, while crosswinds cause lateral displacement. Additionally, the impact of wind gusts depends on their starting time, earlier gusts result in a more dispersed trajectory and greater travel distance along all axes. When wind turbulence is introduced, the trajectory becomes more irregular, exhibiting sharper deviations and unpredictable variations, making the flight path less stable. Increasing the rigging angle enhances travel distance, but angles below -10° introduce oscillations. It also raises body velocity, lowers descent rate, and extends flight time. Wind speed and direction significantly affect trajectory. When wind turbulence is introduced, the trajectory becomes more irregular, exhibiting additional edges and curves. The changes in travel distance are not simply an increase or decrease but occur randomly.
Increasing the rigging angle results in a greater traveled distance, but at very negative angles like -10°, oscillations in the trajectory may occur, complicating mid-air recovery. Therefore, it is advisable to keep the rigging angle above -10° for stability. A larger rigging angle leads to higher velocity and a lower descent rate, with oscillations in both velocity profiles appearing at -10°. A more negative rigging angle, such as -10°, extends the flight time due to increased velocity and greater travel distance, with a flight time of around 7.1 minutes. Additionally, wind speed and direction significantly affect the parafoil's trajectory, velocity, and descent rate, which can be observed by varying these factors while keeping other parameters constant. Although the circular parachute lacks a wing-shaped canopy like the parafoil, wind still has a significant impact on its motion, causing it to travel far distances.

The additional mass, which is required to transform the Electron's first stage into a reusable one is 100 kg, with the parafoil accounting for 74 kg, the ballute 9 kg, and the remaining mass attributed to modifications such as the parachute attachment and release mechanism and for the circular parachute 71.5 kg with 22.8 kg for canopy, 35.3 kg for suspension lines and 13.4 kg actuators. The parafoil provides superior control, whereas the circular parachute is lighter and more straightforward in design. Consequently, if refurbishment costs are identical for both, the circular parachute emerges as the more advantageous choice.

14.1. Recommendations

In this section, a brief recommendation is provided to improve the tool developed in this study and enhance its capability and accuracy.

- The first recommendation is to add a trajectory optimization model for the ascent phase of the Electron launch vehicle to obtain the trajectory that requires the least amount of fuel and/or delivers the maximum payload to a given orbit. For this purpose, a more robust GNC model is needed. Therefore, instead of relying solely on pitch angle data from past flight records, perhaps a Kalman filter can be used for state estimation. For example, to estimate the pitch angle and its rate of change over time for improved attitude control. Furthermore, a nonlinear control law should be designed that directly handles the nonlinear dynamics (e.g., sliding mode control, feedback linearization, backstepping, or model predictive control).
- To accurately replicate the physical system, the control input of the Electron launch vehicle must account for factors such as the movement of the engine gimbals and the moment arm,

effectively incorporating thrust vector control.

- Incorporate the dynamic center of gravity, moment coefficient, and center of pressure into the Electron launch vehicle model to achieve more accurate results and enhance thrust vectoring
- A GNC model should be added for the first-stage re-entry to control the re-entry attitude.
- Implementing parachute reefing, which allows for the gradual opening of the ballute, parafoil, and/or circular parachute canopy. It also helps restrain the canopy from fully inflating or over-inflating, thereby limiting opening forces and preventing potential damage to the system.
- To expand the capability of the developed tool, the GNC model for the parafoil should be enhanced to handle scenarios in which a predefined location is selected, requiring the parafoil to maneuver towards a specific landing zone. It should not only maintain a straight trajectory but also execute more complex maneuvers, as needed for aerial delivery systems.
- For the circular parachute, implement non-gliding parachute guidance through canopy distortion. This can be achieved by temporarily lengthening one or more adjacent risers to induce the desired canopy distortion. This will help reduce the dispersion of the landing zone (or splashdown zone).

A

Electron Rocket's Constructed Past Missions Data

Table A.1: The Owl's Night Begins 2020: Launch Complex 1, Mahia. 150 kg Payload to 500 km SSO Orbit with 97.3° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
01:15	Max Q	13.3	1711
02:27	MECO on Electron's first stage	64.6	7886
02:31	Stage 1 separation	69	8355
02:34	Electron's Stage 2 Rutherford engine ignites	72.2	8427
03:07	Fairing separation	104.3	8883
05:56	Battery hot-swap (jettison)	192.7	15336
08:33	Electron reaches orbit	202.6	26993
08:35	SECO on Electron's second stage	202.7	27298
08:40	Stage 2 separation from Kick Stage	203.3	28587
52:28	The Curie engine on the Kick Stage ignites	-	-
54:22	The Curie engine cuts off	-	-
60:00	Payloads deployed	500	-

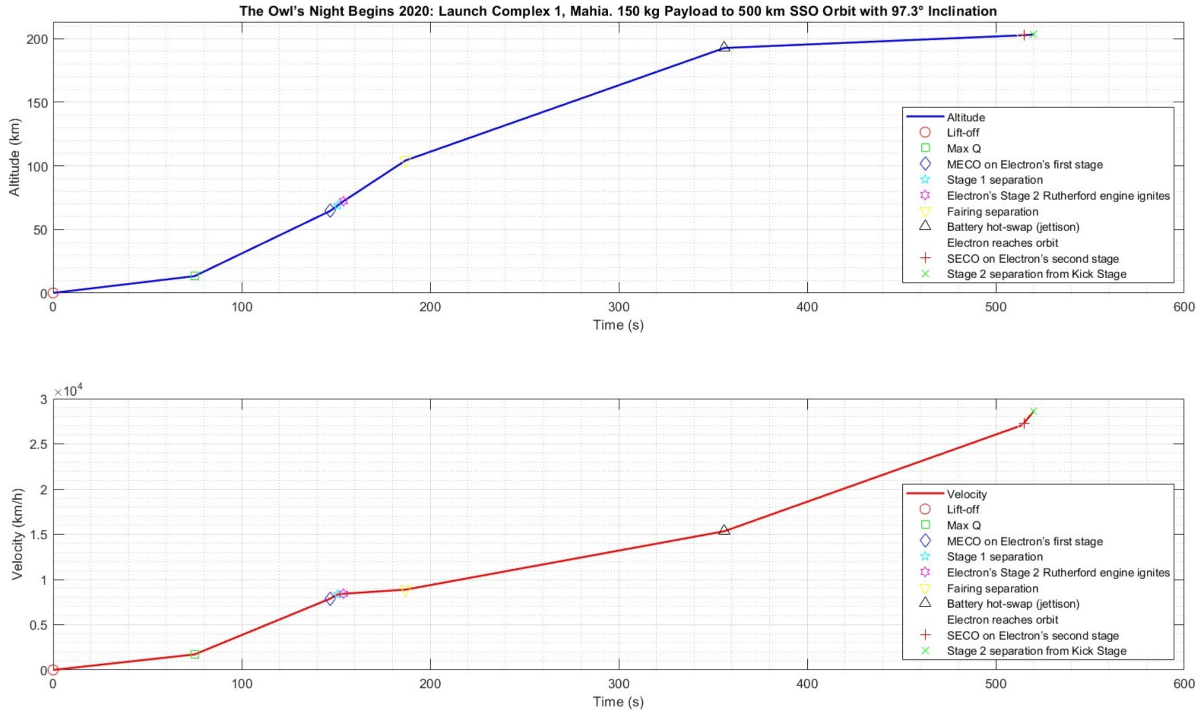


Figure A.1: The Owl's Night Begins Ascent Trajectory

Table A.2: The Owl's Night Continues 2022: Launch Complex 1, Mahia. 150 kg Payload to 561 km SSO Orbit with 97° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
01:10	Max Q	12.3	1626
02:23	MECO on Electron's first stage	67.9	7958
02:26	Stage 1 separation	71.6	7909
02:29	Electron's Stage 2 Rutherford engine ignites	75	7856
03:02	Fairing separation	109.4	8256
06:12	Battery hot-swap (jettison)	208.6	14743
09:12	Electron reaches orbit	205.2	28200
09:17	SECO on Electron's second stage	205.5	28650
09:20	Stage 2 separation from Kick Stage	205.7	28650
49:20	The Curie engine on the Kick Stage ignites	-	-
52:02	The Curie engine cuts off	-	-
53:03	Payloads deployed	561	-

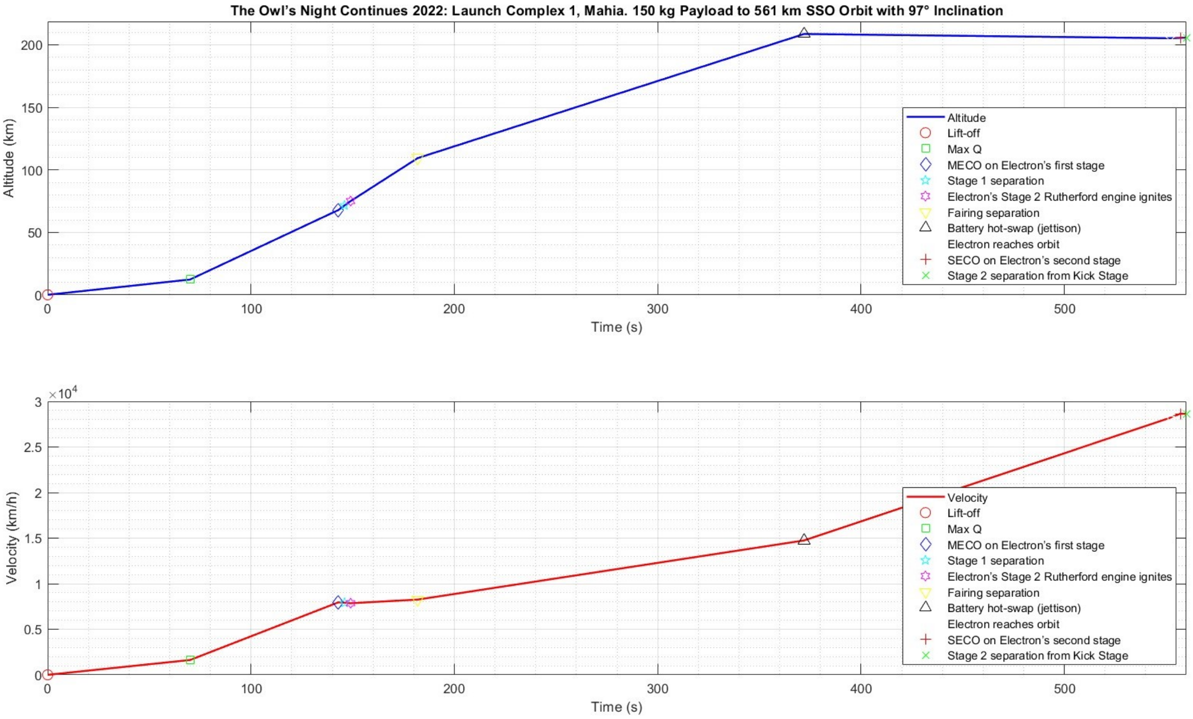


Figure A.2: The Owl's Night Continues Ascent Trajectory

Table A.3: There And Back Again 2022: Launch Complex 1, Mahia. 134 Pico Satellites as Payload to 520 km SSO Orbit with Approximately 97° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
01:14	Max Q	12.3	1599
02:29	MECO on Electron's first stage	69.5	7556
02:32	Stage 1 separation	73.2	7500
02:36	Electron's Stage 2 Rutherford engine ignites	77.6	7449
03:06	Fairing separation	109	7761
07:26	Battery hot-swap (jettison)	233.9	14863
10:15	Electron reaches orbit	216.5	28569
10:17	SECO on Electron's second stage	216.5	28569
10:23	Stage 2 separation from Kick Stage	216.5	28569
57:43	The Curie engine on the Kick Stage ignites	-	-
59:11	The Curie engine cuts off	-	-
60:00	All payloads deployed	520	-
04:36	Stage 1 Apogee	-	-
07:29	Stage 1 Drogue parachute deploys	13	-
08:12	Stage 1 main parachute deploys	6	36
18:12	Stage 1 captured	-	-

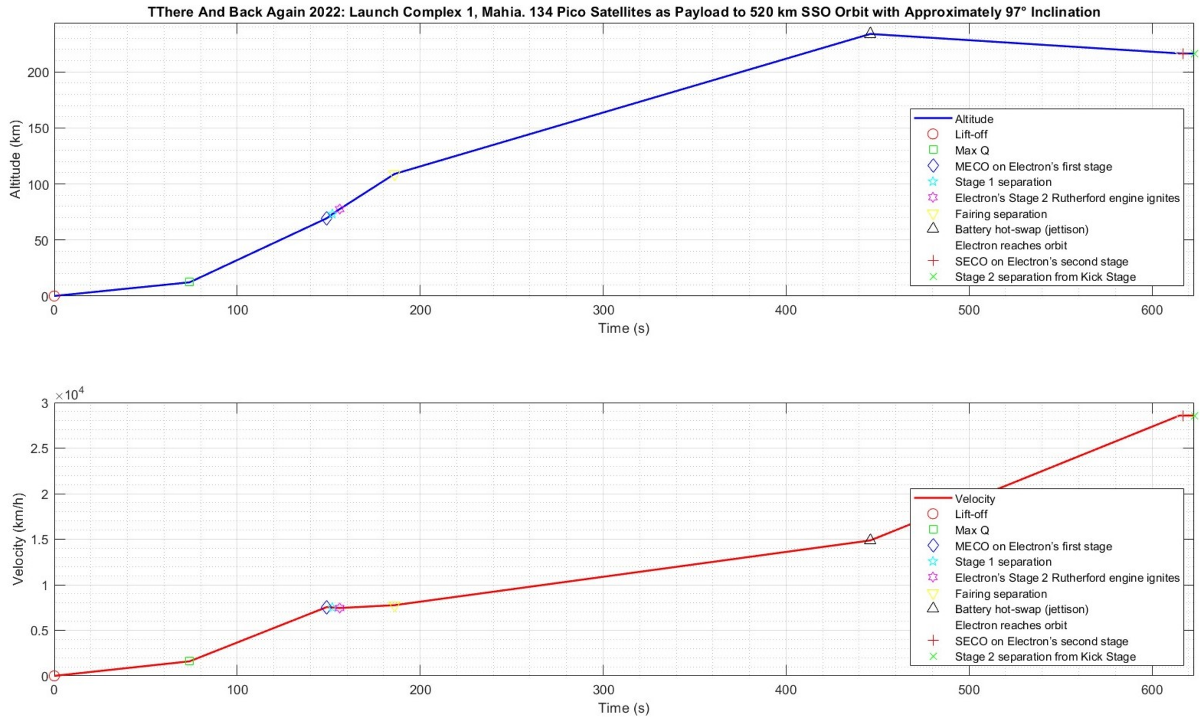


Figure A.3: There And Back Again Ascent Trajectory

Table A.4: The Owl Spreads Its Wings 2022: Launch Complex 1, Mahia. 100 kg Payload to 563 km SSO Orbit with 97° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
00:56	Vehicle supersonic	7.5	1151
01:08	Max Q	11.6	1565
02:26	MECO on Electron’s first stage	68.6	7977
02:29	Stage 1 separation	72	7926
02:32	Electron’s Stage 2 Rutherford engine ignites	75	7887
03:09	Fairing separation	110	8416
06:13	Battery hot-swap (jettison)	201.7	14733
09:16	Electron reaches orbit	204.4	28651
09:20	SECO on Electron’s second stage	204.7	28654
09:24	Stage 2 separation from Kick Stage	204.7	28654
50:35	The Curie engine on the Kick Stage ignites	-	-
53:18	The Curie engine cuts off	-	-
54:18	Payloads deployed	563	-

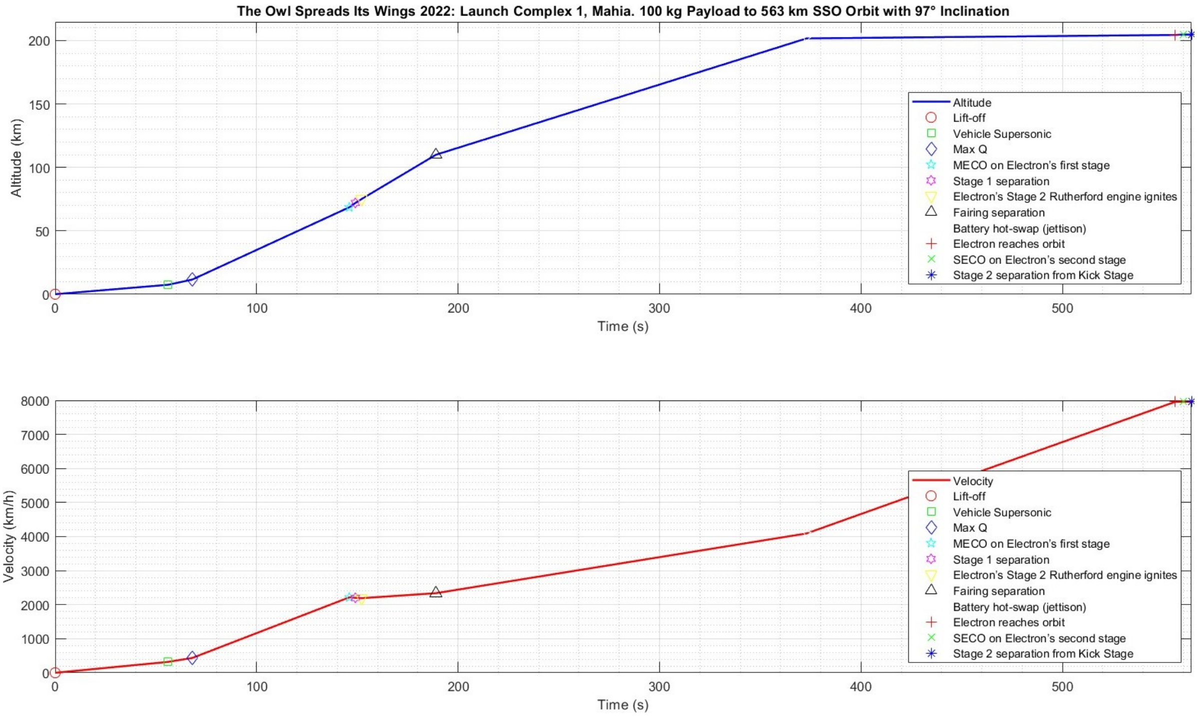


Figure A.4: The Owl Spreads Its Wings Ascent Trajectory

Table A.5: It Argos Up From Here 2022: Launch Complex 1, Mahia. 118 kg Payload to 750 km SSO Orbit with 98° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
00:56	Vehicle supersonic	7.8	1189
01:08	Max Q	12	1606
02:23	MECO on Electron's first stage	71.2	7966
02:26	Stage 1 separation	75	7908
02:29	Electron's Stage 2 Rutherford engine ignites	78.4	7861
02:57	Fairing separation	110.3	8164
06:11	Battery hot-swap (jettison)	262	14327
09:05	Electron reaches orbit	310.6	28521
09:08	SECO on Electron's second stage	310.7	28521
09:12	Stage 2 separation from Kick Stage	310.7	28521
51:26	The Curie engine on the Kick Stage ignites	-	-
53:58	The Curie engine cuts off	-	-
54:58	Payloads deployed	750	-

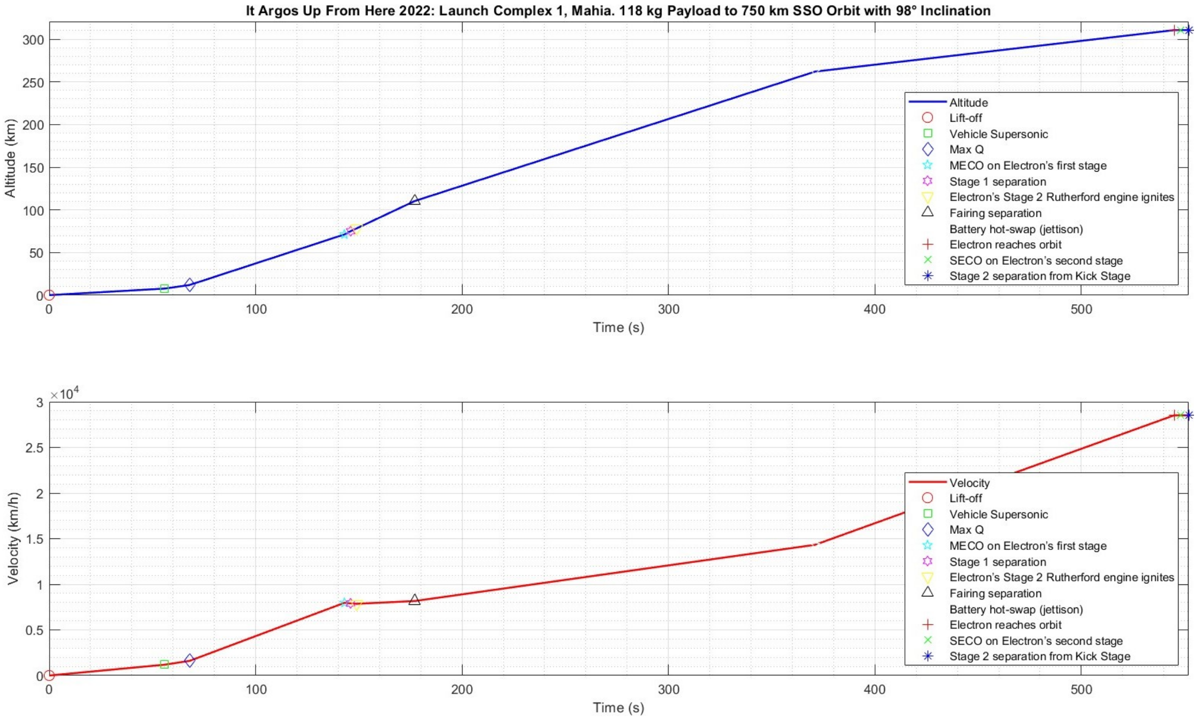


Figure A.5: It Argos Up From Here Ascent Trajectory

Table A.6: Catch Me If You Can 2022: Launch Complex 1, Mahia. 50 kg Payload to 585 km SSO Orbit with approximately 98° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
01:01	Vehicle supersonic	7.9	1156
01:14	Max Q	12.4	1621
02:29	MECO on Electron's first stage	69.2	7631
02:32	Stage 1 separation	72.6	7576
02:35	Electron's Stage 2 Rutherford engine ignites	76.1	7547
03:08	Fairing separation	109.2	7945
06:48	Battery hot-swap (jettison)	207.2	15967
08:30	Electron reaches orbit	203.5	24456
09:11	SECO on Electron's second stage	203.8	28677
09:12	Stage 2 separation from Kick Stage	203.9	28677
51:32	The Curie engine on the Kick Stage ignites	-	-
52:55	The Curie engine cuts off	-	-
60:00	Payloads deployed	585	-
04:35	Stage 1 apogee	142	-
07:20	Stage 1 drogue parachute deploys	-	-
08:09	Stage 1 main parachute deploys	-	-
18:44	Stage 1 captured	-	-

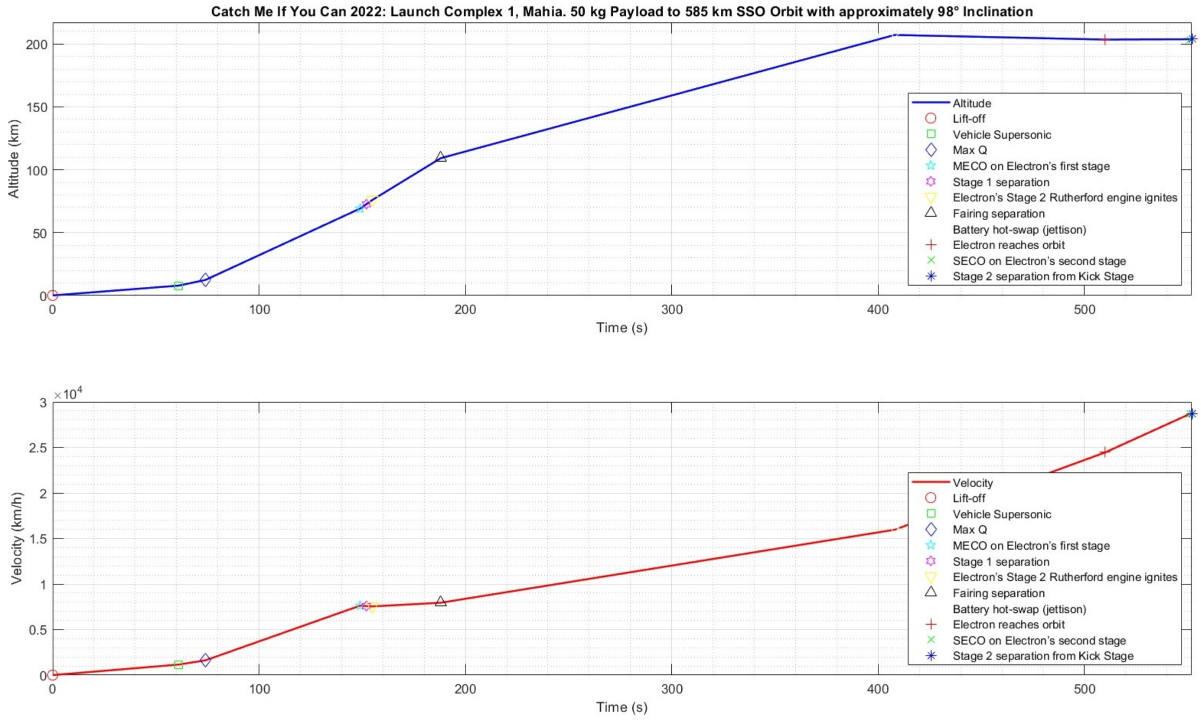


Figure A.6: Catch Me If You Can Ascent Trajectory

Table A.7: We Love The Nightlife 2023: Launch Complex 1, Mahia. 165 kg Payload to 640 km LEO Orbit with 53° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
00:54	Vehicle supersonic	6.5	1044
01:06	Max Q	10.3	1426
02:24	MECO on Electron's first stage	64.9	7445
02:27	Stage 1 separation	68.4	7593
02:31	Electron's Stage 2 Rutherford engine ignites	72.7	7524
03:07	Fairing separation	108.3	7953
06:09	Battery hot-swap (jettison)	202.2	14103
08:30	Electron reaches orbit	202.3	23269
09:07	SECO on Electron's second stage	200.2	27470
09:09	Stage 2 separation from Kick Stage	200	27471
54:22	The Curie engine on the Kick Stage ignites	-	-
57:34	The Curie engine cuts off	-	-
57:34	Payloads deployed	-	-
04:43	Stage 1 apogee	138	-
05:29	Stage 1	125	-
07:35	Stage 1 drogue parachute deploys	25 (approx.)	-
08:25	Stage 1 main parachute deploys	-	-
18:44	Stage 1 captured	-	-

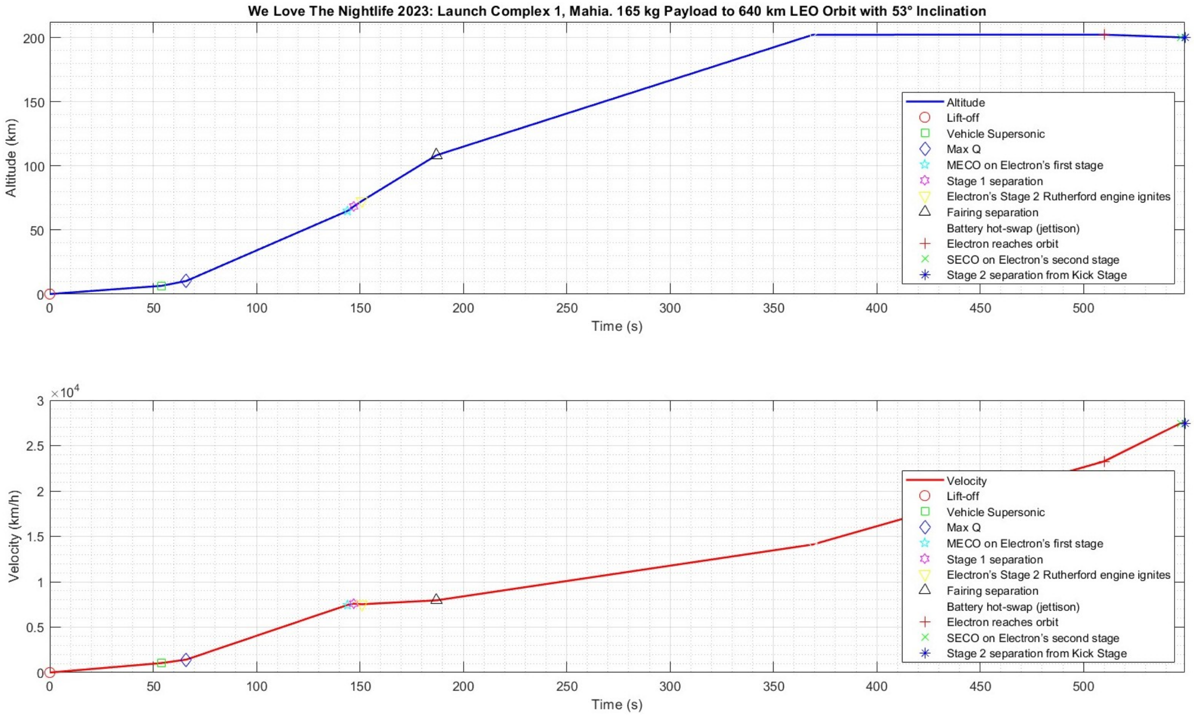


Figure A.7: We Love The Nightlife Ascent Trajectory

Table A.8: Return to Sender 2020: Launch Complex 1, Mahia. 200 kg Payload to 500 km SSO with 97.3° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.0	0
00:54	Vehicle Supersonic	-	-
01:14	Max Q	12.6	1627
02:33	MECO on Electron’s first stage	70.3	8253
02:36	Stage 1 separation	73.5	8215
02:39	Electron’s Stage 2 Rutherford engine ignites	76.8	8166
03:15	Fairing separation	110.2	8801
06:08	Battery hot-swap (jettison)	195.5	15609
08:46	Electron reaches orbit	203	28587
08:54	SECO on Electron’s second stage	203	28587
09:09	Stage 2 separation from Kick Stage	-	-
49:38	The Curie engine on the Kick Stage ignites	-	-
51:08	The Curie engine cuts off	-	-
60:00	Payloads deployed	-	-
04:43	Stage 1 apogee	-	-
05:29	Stage 1	-	-
07:38	Stage 1 Drogue parachute deploys	-	-
08:44	Stage 1 main parachute deploys	-	-
12:48	Stage 1 splashdown	-	-

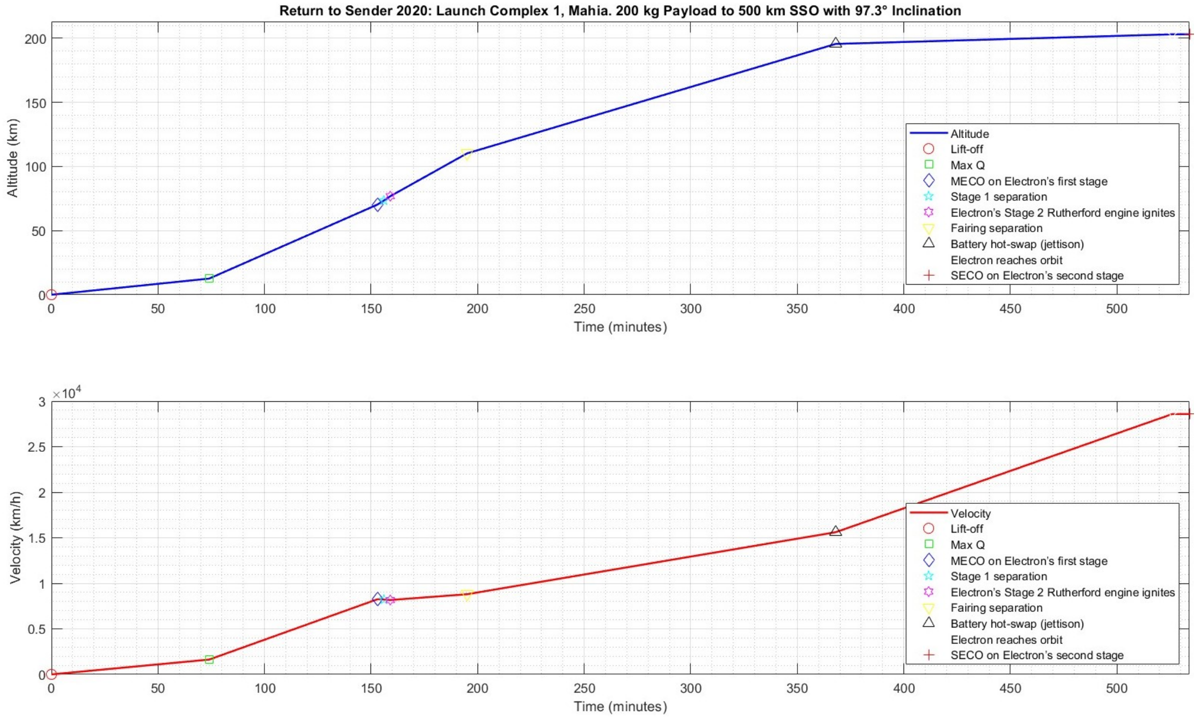


Figure A.8: Return to Sender Ascent Trajectory

Table A.9: Baby Come Back 2023: Launch Complex 1, Mahia. 86 kg Payload to 1000 km SSO with 99.45° Inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.1	0
1:00	Vehicle Supersonic	8.7	1269
1:11	Max Q	12.7	1676
02:24	MECO on Electron's first stage	67.9	7739
02:27	Stage 1 separation	71.1	7690
02:31	Electron's Stage 2 Rutherford engine ignites	75.8	7638
03:03	Fairing separation	109.4	8039
06:09	Battery hot-swap (jettison)	215.8	14498
08:30	Electron reaches orbit	227	25137
08:59	SECO on Electron's second stage	229.2	28629
09:09	Stage 2 separation from Kick Stage	-	-
09:09	The Curie engine on the Kick Stage ignites	-	-
09:09	The Curie engine cuts off	-	-
09:09	Payloads deployed	-	-
04:07	Stage 1 Apogee	Approx. 145	-
07:23	Stage 1 Drogue parachute deploys	-	-
07:38	Stage 1 is Subsonic	-	-
08:13	Stage 1 main parachute deploys	-	-
15:15-17:43	Stage 1 predicted splashdown time	-	

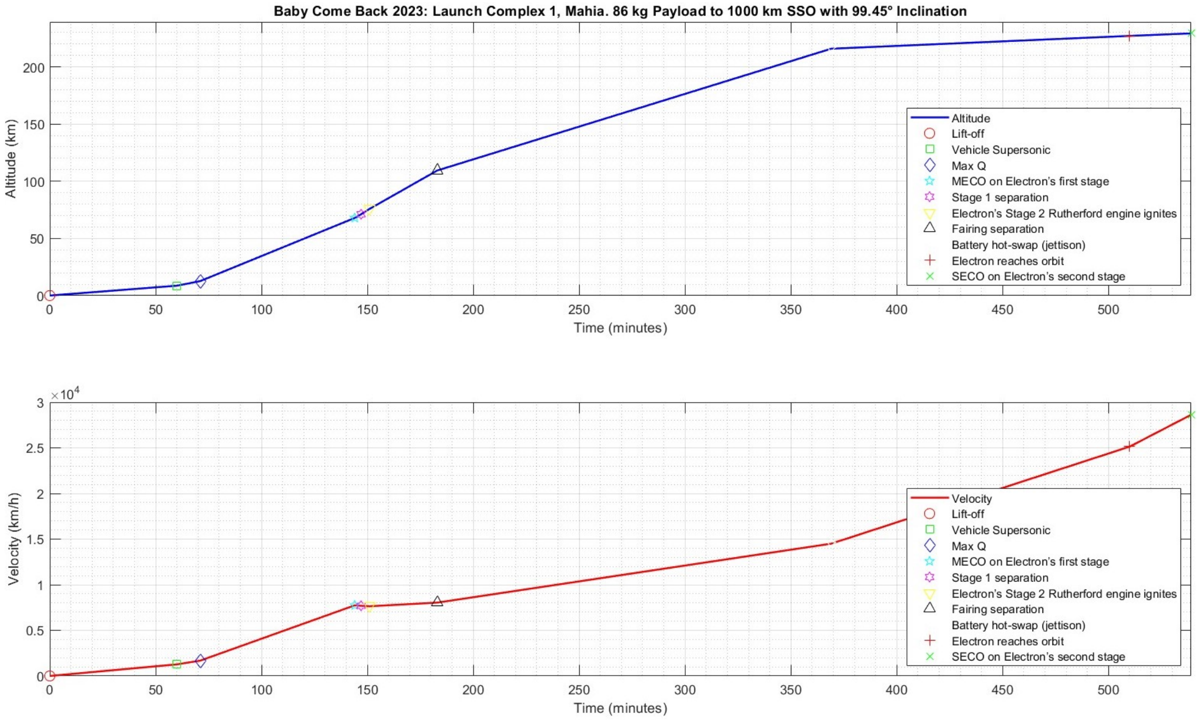


Figure A.9: Baby Come Back Ascent Trajectory

Table A.10: Virginia is for Launch Lovers 2023: Launch Complex 2, Wallops Island, Virginia, USA. 3 satellites, approx. 50 kg payload to 550 km circular orbit with 40.5° inclination

Time	Event	Altitude [km]	Velocity [km/h]
00:00	Lift-off	0.0	0
01:00	Vehicle Supersonic	7.4	1157
1:11	Max Q	11.1	1543
02:26	MECO on Electron's first stage	70	7866
02:29	Stage 1 separation	74	7804
02:32	Electron's Stage 2 Rutherford engine ignites	77.9	7746
02:56	Fairing separation	105.9	7947
06:45	Battery hot-swap (jettison)	292.2	14988
08:48	Electron reaches orbit	330.3	23950
09:14	SECO on Electron's second stage	331.1	26665
09:18	Stage 2 separation from Kick Stage	331.1	26665
55:10	The Curie engine on the Kick Stage ignites	-	-
57:18	The Curie engine cuts off	-	-
57:18	Payloads deployed	-	-

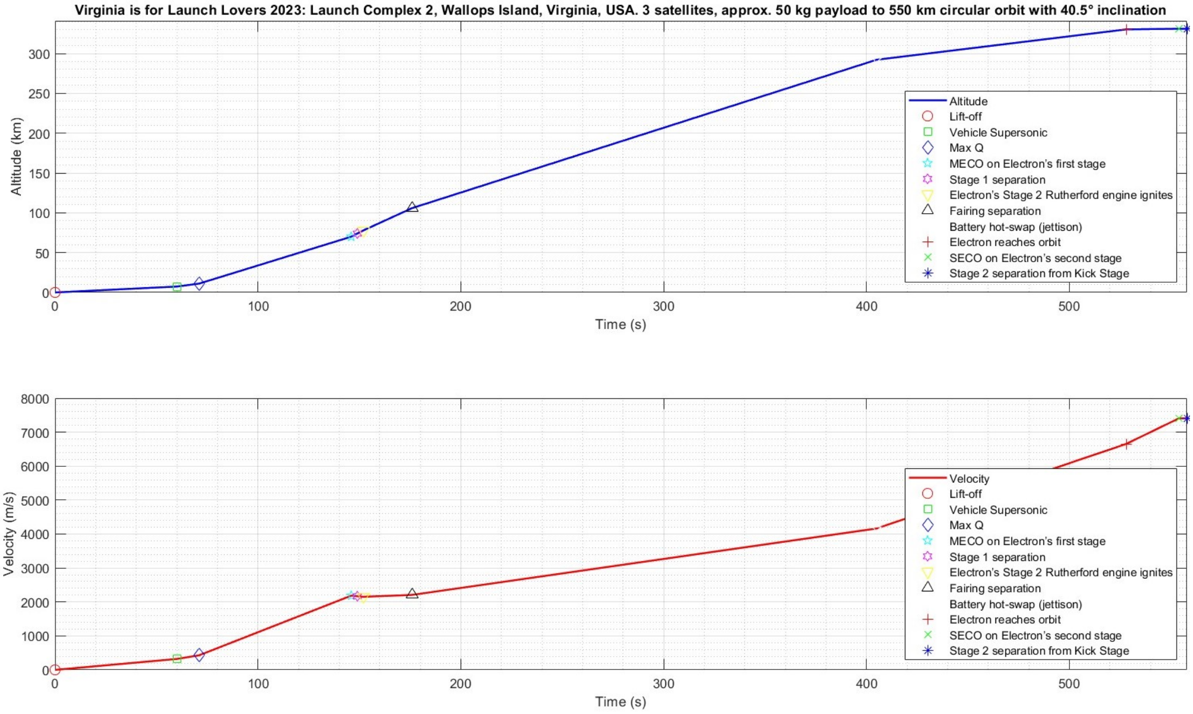
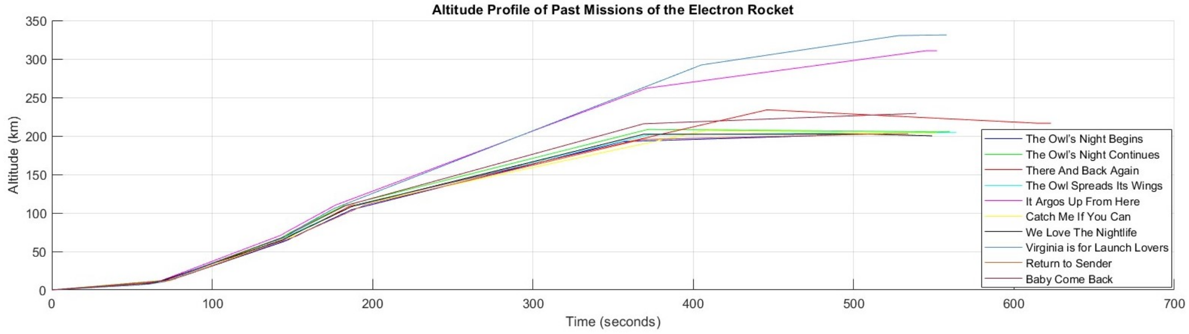


Figure A.10: Virginia is for Launch Lovers Ascent Trajectory



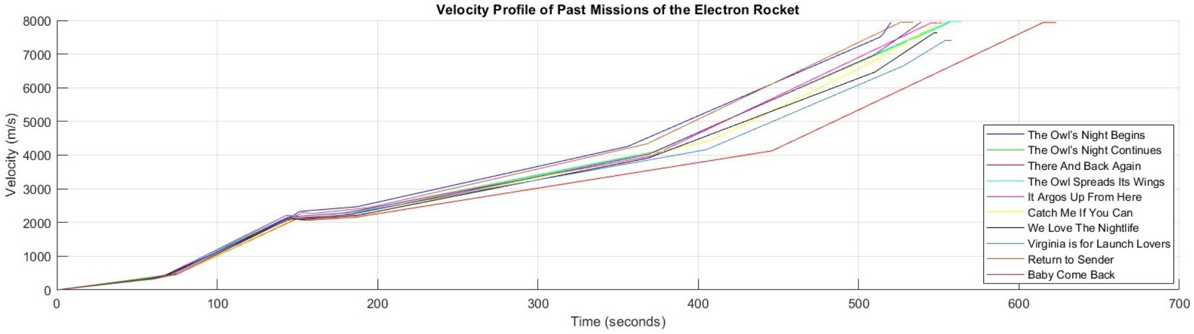


Figure A.11: The Altitude and Velocity Profiles of Past Missions of Electron Rocket up to SECO on Electron's Second Stage

B

MATLAB Functions and Scripts

```matlab
% Burnout Time Calculation for the First Stage
function burnout_time = calculateTimes(time, mass_flow)

     % Define a persistent variable
    % disp(mass_flow);
    % disp(time);
    persistent BurnTime;
    persistent LastMassFlow;
    if isempty(LastMassFlow)
        LastMassFlow=0;
    end

    %disp(BurnTime)
    % disp("==================")
    if isempty(BurnTime)
        % Initialize BurnTime if it is empty
        BurnTime = -1;
    elseif BurnTime>-1
        burnout_time=BurnTime;
        LastMassFlow=mass_flow;
        return
    end
    if mass_flow == 0 & LastMassFlow ~=0
        BurnTime=time;
        burnout_time=time;
        LastMassFlow=mass_flow;
        return
    else
        %Using realmax instead of BurnTime because the initial BurnTime
        %value is -1 and this causes error for the stage thrust selector of
        %second stage, which takes as input tb_1. realmax is choosen as
        %it is bigger than burn time of the first stage tb_1. this value
        %needs to be higher than tb_1. realmax is the highest available
        %intiger
        burnout_time=realmax;
        LastMassFlow=mass_flow;
        return
    end
    LastMassFlow=mass_flow;
end
```

Listing B.1: Burnout Time Calculation for the First Stage

```matlab
% Burnout Time Calculation for the Second Stage
function burnout_time = calculateTimes(time, mass_flow)

     % Define a persistent variable
    % disp(mass_flow);
    % disp(time);
    persistent BurnTime;
    persistent LastMassFlow;
    if isempty(LastMassFlow)
        LastMassFlow=0;
    end

    %disp(BurnTime)
    % disp("=================")
    if isempty(BurnTime)
        % Initialize BurnTime if it is empty
        BurnTime = -1;
    elseif BurnTime>-1
        burnout_time=BurnTime;
        LastMassFlow=mass_flow;
        return
    end
    if mass_flow == 0 & LastMassFlow ~=0
        BurnTime=time;
        burnout_time=time;
        LastMassFlow=mass_flow;
        return
    else
         %Using realmax instead of BurnTime because the initial BurnTime
        %value is -1 and this causes error for the stage thrust selector of
        %second stage, which takes as input tb_1. realmax is choosen as
        %it is bigger than burn time of the first stage tb_1. this value
        %needs to be higher than tb_1. realmax is the highest available
        %intiger
        burnout_time=realmax;
        LastMassFlow=mass_flow;
        return
    end
    LastMassFlow=mass_flow;
end
```

Listing B.2: Burnout Time Calculation for the Second Stage

```
function [T, m] = customFunction(t, t_coasting, T1, T2, t_b1, t_b2, m1, m2)
    %T is the thrust and m is the mass
    if t > 0 && t <= t_b1
        T = T1;
        m = m1;
    elseif t > t_b1 && t <= (t_b1 + t_coasting)
        T = [0; 0; 0];
        m = m2;
    elseif (t > (t_b1 + t_coasting)) && (t < (t_b1 + t_coasting + t_b2))
        T = T2;
        m = m2;
    else
        T = [0; 0; 0];
        m = m2;
    end
end

%% when coasting time is not taken into consideration
%function [T, m] = customFunction(t,t_coasting,T1, T2, t_b1, t_b2, m1, m2)
 %   if t > 0 && t <= t_b1
  %      T = T1;
   %     m = m1;
    %else
     %   T = T2;
      %  m = m2;
    %end
%end
```

Listing B.3: Stage Selector

```
function trust = Thrust_Activation(time, t_b1, t_b2, t_coasting, T2, T0, Thrust_correction)
```

```
    % Check if time is greater than tb_1 + t_coasting
    if (time > (t_b1 + t_coasting)) && (time < (t_b1 + t_coasting + t_b2))
        trust = T2 + Thrust_correction;
    else
        trust = T0;
    end
end
```

Listing B.4: Thrust Activation For The Second Stage

```
function trust = Thrust_termination(time, t_b1, T1, T0, Thrust_correction)
    % Thrust must be zero after first stage burnout
    if time < t_b1
        trust = T1 + Thrust_correction;
    else
        trust = T0;
    end
end
```

Listing B.5: Thrust Termination For The First Stage After Burnout

```
function updated_mass = adjustMass(altitude, current_mass, Fairing_mass, Battery_mass)
%Fairing seperation at 110 km altitude
%Battery Jettison at 195 km altitude
    if altitude >= 110000
        current_mass = current_mass - Fairing_mass;
    end

    if altitude >= 195000
        current_mass = current_mass - Battery_mass;
    end

    updated_mass = current_mass;
end
```

Listing B.6: Micro-Staging

```
function Thrust_correction = calculateCorrection(time, Correction, t_b1, t_b2, t_coasting)
    % Function to calculate the required correction thrust at a given time

    % Check if time is less than or equal to  seconds burnout time
    if time <= t_b1
        % Thrust is not zero
        Thrust_correction = Correction;
    elseif (time > t_b1) && (time < (t_b1 + t_coasting))
        % Thrust is  zero
        Thrust_correction = 0;
    elseif (time > (t_b1 + t_coasting)) && (time < (t_b1 + t_coasting + t_b2))
        Thrust_correction = Correction;
    else
        % Thrust is zero after second burnout time
        Thrust_correction = 0;
    end
end
```

Listing B.7: Thrust Modulation for Both Stages

```
function heat_flux = calculateHeatFlux(altitude, velocity, rho)
% Heat Flux Calculation for the Ascent Phase
    if altitude <= 111
        disp(’Altitude is below 110,000 meters. Heat flux cannot be calculated.’)
        heat_flux = NaN;
    else
        heat_flux = 0.5 * rho * velocity^3;
    end
end
```

Listing B.8: Heat Flux Calculation

```

function Cd_1  = customFunction(Mach, interpolation_function)
    if Mach > 5
        Cd_1 = 0.55038;
    else
         Cd_1 = interpolation_function;
    end
end
```

Listing B.9: Drag Coefficient For the First Stage

```

function Cd_2  = customFunction(Mach, interpolation_function)
    if Mach > 5
        Cd_2 = 0.55038;
    else
         Cd_2 = interpolation_function;
    end
end
```

Listing B.10: Drag Coefficient For the Second Stage

```
function Cd = calculateCd(time, tb_1, Cd_1, Cd_2)
    if time <= tb_1
        Cd = Cd_1;
    else
        Cd = Cd_2;
    end
end
% tb_1 is burn time first stage
% time is just simulation time
```

Listing B.11: Drag Coefficient Selection For Both Stage

```
function d = haversine_distance(lat1, lon1, lat2, lon2)
    % HAVERSINE_DISTANCE Calculate the great-circle distance between two points
    % on the Earth's surface using the Haversine formula.
    %
    %   d = HAVERSINE_DISTANCE(lat1, lon1, lat2, lon2, radius) returns the
    %   distance between the points specified by latitude and longitude
    %   in radians. The radius of the sphere (e.g., Earth) must be provided.
    %
    %   lat1, lon1: Latitude and Longitude of the first point (launch complex 1) in  radians
    %   lat2, lon2: Latitude and Longitude of the second point in radians
    %   radius: Radius of the sphere (e.g., Earth's radius in kilometers)
    %
    radius = 6371; % Earth's radius in kilometers
    % Differences in latitude and longitude
    delta_lat = lat2 - lat1;
    delta_lon = lon2 - lon1;
    % Haversine formula
    a = sin(delta_lat / 2)^2 + cos(lat1) * cos(lat2) * sin(delta_lon / 2)^2;
    c = 2 * atan2(sqrt(a), sqrt(1 - a));

    % Distance in km  (since radius is in km)
    d = radius * c;
end
```

Listing B.12: Downrange Distance Calculation

```

function Cd_1  = customFunction(Mach, interpolation_function)
    if Mach > 7.97;
        Cd_1 = 0.77;
    else
         Cd_1 = interpolation_function;
    end
```

```
end
```

Listing B.13: Drag Coefficient for the Aerodynamic Decelerator Ballute

```

function drag_output  = check_altitude_and_drag(ballute_drag, altitude)
    % Check if the altitude is 23 km or less
    if altitude <=23000
        % If altitude is 22 km or less, activate drag
       drag_output = ballute_drag;
    else
        % Otherwise, set drag output to zero
        drag_output = [0;0;0];
    end
end
```

Listing B.14: Ballute Deployment

```

function [volume, surfaceArea, mass] = balluteCalculations(radius, thickness, density)
    % This function calculates the volume, surface area, and mass of a ballute
    % given its radius (assuming a spherical shape), material thickness, and material density.
    %
    % Inputs:
    %   radius    - radius of the ballute (m)
    %   thickness - thickness of the material (m)
    %   density   - density of the material (kg/m^3)
    %
    % Outputs:
    %   volume      - volume of the ballute (m^3)
    %   surfaceArea - surface area of the ballute (m^2)
    %   mass        - mass of the ballute (kg)

    % Calculate volume of the sphere
    volume = (4/3) * pi * radius^3;

    % Calculate surface area of the sphere
    surfaceArea = 4 * pi * radius^2;

    % Calculate mass of the ballute
    mass = surfaceArea * thickness * density;
end
%note: for drag calculation cross-sectional area is used (pi*r^2)
```

Listing B.15: Ballute Volume, Surface Area and Mass Calculation

```
function s_omega = calc_skew_symmetric_matrix(p, q, r)
    % Calculate the skew-symmetric matrix s_omega based on p, q, and r
    s_omega = [0, -r, q;
               r, 0, -p;
               -q, p, 0];
end
```

Listing B.16: Skew-Symmetric Matrix $S(\omega)$ Calculation

```
    % This function computes the skew-symmetric matrix s_r_BS
    % based on the vector components x_BS, y_BS, and z_BS.
    y_BS = 0;
    x_BS = 0;
    % Compute the skew-symmetric matrix
    s_r_BS = [0, -z_BS, y_BS;
              z_BS, 0, -x_BS;
              -y_BS, x_BS, 0];
end
```

Listing B.17: Skew-Symmetric Matrix Calculation for $S(r_{BS})$

```
    % This function computes the skew-symmetric matrix s_r_BM
    % based on the vector components x_BM, y_BM, and z_BM.
    y_BM = 0;
    x_BM = 0;
    % Compute the skew-symmetric matrix
    s_r_BM = [0,     -z_BM,   y_BM;
              z_BM,    0,    -x_BM;
              -y_BM,   x_BM,   0];
end
```

Listing B.18: Skew-Symmetric Matrix $S(r_{BM})$ Calculation

```
function [z_BC, z_BS, z_BM, r_BM, r_BM2, r_BC] = calculate_z_BS(m_e, R_pb, l_z, l_c_gp, m_parafoil, m_payload)
    % Calculate z_BS: distance from payload center to combined system center of mass
    z_BS = l_z * (m_parafoil + m_e) / (m_parafoil + m_payload);

    % Calculate z_BC: distance from body center of mass to B and the confluence point C
    z_BC = l_z - l_c_gp - z_BS;

    % Define r_BC: vector from body center of mass to confluence point C
    r_BC = [0; 0; z_BC];

    % Calculate z_BM: distance from body center of mass to combined system center of mass
    z_BM = z_BS - l_z;

    % Define r_BM: vector from body center of mass to parafoil center of mass
    r_BM = [0; 0; z_BM];

    % Define r_BM2: vector from body center of mass to apparent mass center of gravity of the parafoil
    r_BM2 = R_pb' * [0; 0; z_BM];  % Transpose of R_pb used for vector transformation
end
```

Listing B.19: Calculation of z_{BS}

```

function m_e = calc_added_Mass(rho, b, c)
    % This function calculates the mass trapped inside the parafoil when inflated.
    m_e = 0.09 * b * c^2 * rho;
end
```

Listing B.20: Calculation of Added Mass m_e

```

function [M_f, A_x, B_y, C_z] = calc_ApparentMass_Tensor(rho, a_star, t, b, AR, t_star, c)
    % This function calculates the apparent mass tensor \( M_f \) in the x, y, and z directions.
    A_x = 0.666 * rho * (1 + 8/3 * a_star^2) * t^2 * b;
    B_y = 0.267 * rho * (1 + 2 * a_star^2 / t_star^2 * AR * (1 - t_star^2)) * t^2 * c;
    C_z = 0.785 * rho * sqrt(1 + 2 * a_star^2 * (1 - t_star^2)) * AR / (1 + AR) * c^2 * b;

    % Create the 3x3 matrix and define the diagonal values
    M_f = diag([A_x, B_y, C_z]);
end
```

Listing B.21: Calculation of Apparent Mass Tensor M_f

```

function [I_f, I_a, I_b, I_c] = calc_Apparent_Inertia_Tensor(rho, AR, c, b, a_star, t_star, t)
    % This function calculates the apparent inertia tensor \( I_f \) in the x, y, and z directions.
    I_a = 0.055 * rho * AR / (1 + AR) * c^2 * b^3;
    I_b = 0.0308 * rho * AR / (1 + AR) * (1 + pi/6 * (1 + AR) * AR * a_star^2 * t_star^2) * c^4 * b;
    I_c = 0.0555 * rho * (1 + 8 * a_star^2) * t^2 * b^3;

    % Create the 3x3 matrix and define the diagonal values for the apparent inertia tensor
    I_f = diag([I_a, I_b, I_c]);
end
```

Listing B.22: Calculation of Apparent Inertia Tensor I_f

```

function [I_parafoil, I_x_p, I_y_p, I_z_p] = calc_Inertia_Tensors(m_e, m_parafoil, b, c,
    h_mean)
    % Moments of inertia for the parafoil
    I_x_p = (m_parafoil + m_e) / 12 * (b^2 + h_mean^2);
    I_y_p = (m_parafoil + m_e) / 12 * (c^2 + h_mean^2);
    I_z_p = (m_parafoil + m_e) / 12 * (b^2 + c^2);

    % Inertia tensor for the parafoil
    I_parafoil = diag([I_x_p, I_y_p, I_z_p]);
end
```

Listing B.23: Calculation of Inertia Tensors for Parafoil I_{parafoil}

```

function I_f_prime = Calc_I_f_prime(I_f, R_pb)
    % Transpose of R_pb
    R_pb_transpose = R_pb';

    % Perform the matrix transformation
    % The parafoil apparent inertia tensor rotated by the rigging angle
    I_f_prime = R_pb_transpose * I_f * R_pb;
end
```

Listing B.24: Calculation of the Rotated Parafoil Apparent Inertia Tensor I'_f

```

function M_f_prime = Calc_M_f_prime(M_f, R_pb)
    % The parafoil apparent mass tensor rotated by the rigging angle
    % Transpose of R_pb
    R_pb_transpose = R_pb';

    % Perform the matrix transformation
    % The parafoil apparent mass tensor rotated by the rigging angle
    M_f_prime = R_pb_transpose * M_f * R_pb;
end
```

Listing B.25: Calculation of the Rotated Parafoil Apparent Mass Tensor M'_f

```

function [R_nw, R_nw2] = calculate_R_nw(chi_alpha, gamma_alpha)
    % Calculate the rotation matrix R_nw based on chi_alpha and gamma_alpha

    % Calculate the individual rotation matrices
    R_chi_alpha = [cos(chi_alpha), sin(chi_alpha), 0;
                   -sin(chi_alpha), cos(chi_alpha), 0;
                   0, 0, 1];

    R_gamma_alpha = [cos(gamma_alpha), 0, -sin(gamma_alpha);
                     0, 1, 0;
                     sin(gamma_alpha), 0, cos(gamma_alpha)];

    % R_nw2 is for verification purposes only
    R_nw2 = R_gamma_alpha * R_chi_alpha;

    R_nw = [cos(gamma_alpha)*cos(chi_alpha), cos(gamma_alpha)*sin(chi_alpha), -sin(
        gamma_alpha);
            -sin(chi_alpha), cos(chi_alpha), 0;
            sin(gamma_alpha)*cos(chi_alpha), sin(gamma_alpha)*sin(chi_alpha), cos(gamma_alpha
                )];

```

```
end
% Both R_nw and R_nw2 give the same results.
```

Listing B.26: Calculation of Rotation Matrix R_{nw}

```

function [R_psi, R_theta, R_phi] = calculate_rotation_matrices(psi, theta, phi)
    % Calculate the rotation matrices R_psi, R_theta, and R_phi
    % Calculate the rotation matrices
    R_psi = [cos(psi), sin(psi), 0;
             -sin(psi), cos(psi), 0;
             0, 0, 1];

    R_theta = [cos(theta), 0, -sin(theta);
               0, 1, 0;
               sin(theta), 0, cos(theta)];

    R_phi = [1, 0, 0;
             0, cos(phi), sin(phi);
             0, -sin(phi), cos(phi)];
end
```

Listing B.27: Calculation of Rotation Matrices $R_{\psi}, R_{\theta}, R_{\phi}$

```

function [R_bn, R_bn2] = calculate_R_bn(psi, theta, phi, R_psi, R_theta, R_phi)
    % Calculate the rotation matrix R_bn based on psi, theta, and phi

    % R_bn2 is for verification purposes only
    R_bn2 = R_phi * R_theta * R_psi;
    % Calculate the matrix R_bn
    R_bn = [cos(psi)*cos(theta),              sin(psi)*cos(theta),              -sin(theta);
            cos(psi)*sin(theta)*sin(phi) - sin(psi)*cos(phi), sin(psi)*sin(theta)*sin(phi) + cos(psi)*cos(phi), cos(theta)*sin(phi);
            cos(psi)*sin(theta)*cos(phi) + sin(psi)*sin(phi), sin(psi)*sin(theta)*cos(phi) - cos(psi)*sin(phi), cos(theta)*cos(phi)];
end

% R_bn and R_bn2 are the same, but due to rounding errors, some small discrepancies may appear in R_bn.
% When comparing it to DCM, they are effectively the same, so use DCM as R_bn.
```

Listing B.28: Calculation of Rotation Matrix R_{bn}

```

function R_pb = calculate_R_pb(mu)
    % Calculate the rotation matrix R_pb based on rigging angle mu

    R_pb = [cos(mu), 0, -sin(mu);
            0, 1, 0;
            sin(mu), 0, cos(mu)];
end
```

Listing B.29: Calculation of Rotation Matrix R_{pb}

```

function [I_x_pb, I_y_pb, I_z_pb, I_x_z_pb, I_z_x_pb] = calculate_transformed_inertia(mu, I_x_p, I_y_p, I_z_p)

    I_x_pb = I_x_p * cos(mu)^2 + I_z_p * sin(mu)^2;
    I_y_pb = I_y_p;
    I_z_pb = I_x_p * sin(mu)^2 + I_z_p * cos(mu)^2;
    I_x_z_pb = 1/2 * (I_x_p - I_z_p) * sin(2 * mu);
    I_z_x_pb = 1/2 * (I_z_p - I_x_p) * sin(2 * mu);

end
```

Listing B.30: Calculation of Transformation of Inertia Moment from Parafoil Reference Frame to the Body Frame

```
function R_pw = calculate_R_pw(alpha, beta)
    % Calculate the rotation matrix R_pw based on alpha and beta
    R_pw = [cos(alpha) * cos(beta), cos(alpha) * sin(beta), -sin(alpha);
            -sin(beta), cos(beta), 0;
            sin(alpha) * cos(beta), sin(alpha) * sin(beta), cos(alpha)];
end
```

Listing B.31: Calculation of Rotation Matrix R_{pw}

```
function R_w_S = computeRotationMatrix(alpha, beta, mu)
    % This function computes the rotation matrix R_w^S = R_w^p * (R_b^p)^T

    % Define the rotation matrix R_w^p
    R_w_p = [cos(alpha)*cos(beta), cos(alpha)*sin(beta), -sin(alpha);
             -sin(beta), cos(beta), 0;
             sin(alpha)*cos(beta), sin(alpha)*sin(beta), cos(alpha)];

    % Define the rotation matrix R_b^p
    R_b_p = [cos(mu), 0, -sin(mu);
             0, 1, 0;
             sin(mu), 0, cos(mu)];

    % Compute the transpose of R_b^p
    R_b_p_T = R_b_p’;
    I = eye(3);

    % Compute the final result R_w^S = R_w^p * (R_b^p)^T
    R_w_S = R_w_p * R_b_p_T * I;
end
```

Listing B.32: Calculation of Rotation Matrix R_w^S

```
function V_a = calc_V_real(Vb, Wind_velocity, R_bn)

    % Vb is ground velocity thus [u v w]^T in the body frame
    % Perform the operation: u - R_bn * Wind_velocity
    V_a = Vb - R_bn * Wind_velocity;
end
```

Listing B.33: Calculation of Airspeed Vecotor V_a

```
function [gamma_a, chi_a, v_x, v_y, v_z] = calc_gamma_a_and_chi_a(V_a_NED, V_a_magnitude)
    % Calculate flight path angle using arc sine based on Va
    % Extracting rows from V_a if V_a is [3x1]
    v_x = V_a_NED(1, :);  % First row
    v_y = V_a_NED(2, :);  % Second row
    v_z = V_a_NED(3, :);  % Third row
    % output gamma_a is in rad
    gamma_a = asin(v_z / V_a_magnitude);

    % Calculate heading angle using arctan
    % chi_a1 is for verification only
    chi_a = atan2(v_y, v_x);

end
```

Listing B.34: Flight Path Angle and Heading Angle Calculation

```
function [V_p, V_s] = calc_parfoil_velocity(V_a, R_pb, r_BM, s_omega, r_BC)
    % The airspeed vector of the canopy at the apparent mass center M can
    % be expressed in {p} as
    % V_a = V_b - R_bn * W therefore that part is omitted
    % note V_b is the ground velocity and its components in b frame are [u v w]

    % Calculate velocity vector of parafoil in p frame
    V_p = R_pb' * (V_a + s_omega * r_BM);

    % Calculate velocity vector of payload in S frame
    V_s = (V_a + s_omega * r_BC);
end
```

Listing B.35: Parafoil and Payload Velocity Calculation

```

function [CL, CD, Inlet_drag, Line_drag, Roughness_drag, CY] = calc_AeroCoefficients(alpha, beta, CL_0, CL_alpha, CL_delta_s, delta_s_bar, CD_0, CD_alpha_2, CD_delta_s, CY_beta, h, c)
    % Function to calculate aerodynamic coefficients

    % Lift coefficient (CL)
    CL = CL_0 + CL_alpha * alpha + CL_delta_s * delta_s_bar;

    % Drag coefficient (CD)
    CD = CD_0 + CD_alpha_2 * alpha^2 + CD_delta_s * delta_s_bar;

    % Additional Drag contributions
    % Inlet drag is typically 39% of the total drag
    Inlet_drag = 0.5 * h / c;

    % 30 % increase for line drag
    Line_drag = 30 * Inlet_drag / 39;

    % 2.2 % Roughness drag
    Roughness_drag = 2.2 * Inlet_drag / 39;

    % Side-force coefficient (CY)
    CY = CY_beta * beta;
end
```

Listing B.36: Aerodynamic Coefficients Calculation

```

function [Cl, Cm, Cn, Clp, Clr] = calc_AeroCoefficients_moments(alpha, beta, Cl_beta, b, V_ap_magnitude, Cl_p, Cl_r, Cl_delta_alpha, delta_a_bar, Cm_0, Cm_alpha, Cm_q, c, p, q, r, Cn_beta, Cn_p, Cn_r, Cn_delta_alpha)
    % Function to calculate aerodynamic coefficients for moments
    % Calculate the coefficients for the moments
    Clp = b / (2 * V_ap_magnitude) * Cl_p;
    Clr = b / (2 * V_ap_magnitude) * Cl_r;

    Cl = Cl_beta * beta + b / (2 * V_ap_magnitude) * Cl_p * p + b / (2 * V_ap_magnitude) * Cl_r * r + Cl_delta_alpha * delta_a_bar;
    Cm = Cm_0 + Cm_alpha * alpha + c / (2 * V_ap_magnitude) * Cm_q * q;
    Cn = Cn_beta * beta + b / (2 * V_ap_magnitude) * Cn_p * p + b / (2 * V_ap_magnitude) * Cn_r * r + Cn_delta_alpha * delta_a_bar;
end
```

Listing B.37: Aerodynamic Moment Coefficients Calculation

```
function I = calculate_inertia_matrix(m_e, z_BM, z_BS, I_x_pb, I_x_s, I_y_pb, I_y_s, I_z_pb, I_z_s, I_x_z_pb, I_z_x_pb, m_parafoil, m_payload)


    % Calculate the terms for I_x, I_y, and I_z
    I_x = I_x_pb + I_x_s + (m_parafoil + m_e) * z_BM^2 + m_payload * z_BS^2;
    I_y = I_y_pb + I_y_s + (m_parafoil + m_e) * z_BM^2 + m_payload * z_BS^2;
    I_z = I_z_pb + I_z_s;

    % Calculate the inertia matrix I of the entire system
    I = [I_x, 0, I_x_z_pb;
         0, I_y, 0;
```

```
        I_x_z_pb, 0, I_z];
end
```

Listing B.38: Calculation of the Inertia Tensor of the Entire System Combined

```
function F_aS = calc_aerodynamic_force(rho, V_as, S_s, R_sw, CD_payload)
    F_aS = 0.5 * rho * V_as^2 * S_s * R_sw * [-CD_payload; 0; 0];
end
```

Listing B.39: Calculation of Payload Aerodynamic Forces

```
function F_aP = calc_aerodynamic_force(rho, V_ap, S_p, R_pb, R_pw, CL, CY, CD, Vp_x, Vp_z)

    CX = (-CD * Vp_x + CL * Vp_z) / V_ap;
    CZ = (-CD * Vp_z - CL * Vp_x) / V_ap;

    % Final parafoil aerodynamic force vector in the transformed reference frame
    F_aP = 0.5 * rho * V_ap^2 * S_p * R_pb’ * R_pw * [CX; CY; CZ];
end
```

Listing B.40: Calculation of Parafoil Aerodynamic Forces

```
function F_bp = calc_buoyancy_weight(theta, phi, g_earth, m_e)
    % System weight force expressed in body frame {b}
    F_bp =  -m_e * g_earth * [-sin(theta); cos(theta)*sin(phi); cos(theta)*cos(phi)];
end
```

Listing B.41: Calculation of Buoyancy Forces Due to Added Mass

```
function F_g = calc_sys_weight(theta, phi, g_earth, m_e, m_parafoil, m_paylaod)
    % System weight force expressed in body frame {b}
    F_g = (m_parafoil + m_paylaod + m_e) * g_earth * [-sin(theta); cos(theta)*sin(phi); cos(
        theta)*cos(phi)];
end
```

Listing B.42: Calculation of System Weight Forces

```
function F_remaining_terms = calc_aerodynamic_force(m_t, V_b, s_r_BM, s_omega, omega_b,
    M_f_prime, Wind_Velocity_vector)

    % I didn’t use R_bn*W because W is already in the bn frame due to R_bn
    % or DCM used in the wind models which provides wind velocity in the bn
    % frame. The wind is already in the body frame. No need to use R_nb
    % (from navigation frame to body frame b) to bring wind in body frame.
    % It is already in the body frame.

    % Calculate the remaining aerodynamic force terms
    F_remaining_terms = -m_t * s_omega * V_b - s_omega * M_f_prime * (V_b - s_r_BM * omega_b)
         + s_omega * M_f_prime * Wind_Velocity_vector;

end
```

Listing B.43: Aerodynamic Force Calculation for Remaining Terms

```
function M_ap = calc_M_ap_moment(rho, V_ap_magnitude, S_p, R_pb, b, c, Cl, Cm, Cn)
    % Calculate the parafoil aerodynamic moment vector expressed as M_ap

    M_ap = 0.5 * rho * V_ap_magnitude^2 * S_p * R_pb’ * [b * Cl; c * Cm; b * Cn];
end
```

Listing B.44: Parafoil Aerodynamic Moment Calculation

```
function M_bp = calc_M_bp_moment(s_r_BM, F_bp)
    % Calculate the buoyancy force moment, M_bp
    M_bp = s_r_BM * F_bp;
end
```

Listing B.45: Calculation of Moments Due to Buoyancy Forces

```
function M_remaining_term = calc_M_remaining_term(s_r_BM, F_ap,S_r_BS, F_as,S_omega, I,
    omega_b, I_f_prime, M_f_prime, wind_velocity_vector, V_b)
    % Calculate the remaining terms in the moment equation

    M_remaining_term = s_r_BM * F_ap + S_r_BS*F_as - S_omega*I*omega_b - S_omega*I_f_prime*
        omega_b - s_r_BM*S_omega*M_f_prime*(V_b - s_r_BM*omega_b - wind_velocity_vector);
end
```

Listing B.46: Calculation of Moments Due to the Remaining Terms

C
Ascent Phase Results Data

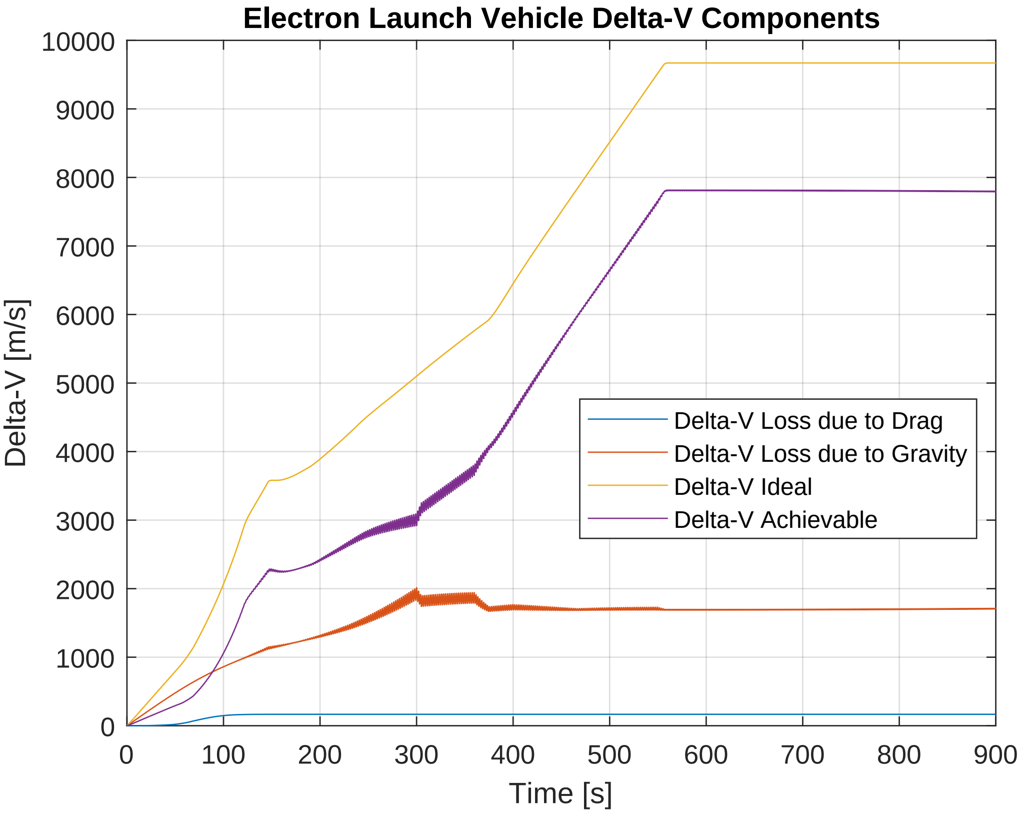


Figure C.1: ΔV Profile of Ascent Trajectory

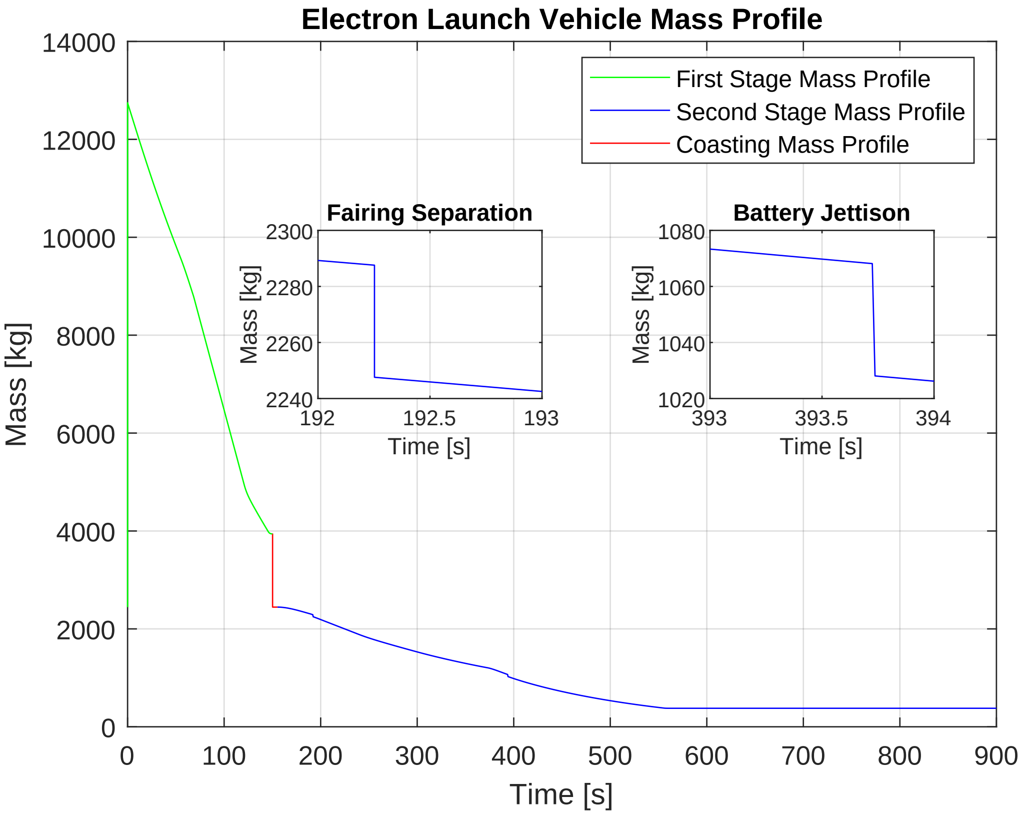


Figure C.2: Electron Mass Profiles of Ascent Trajectory

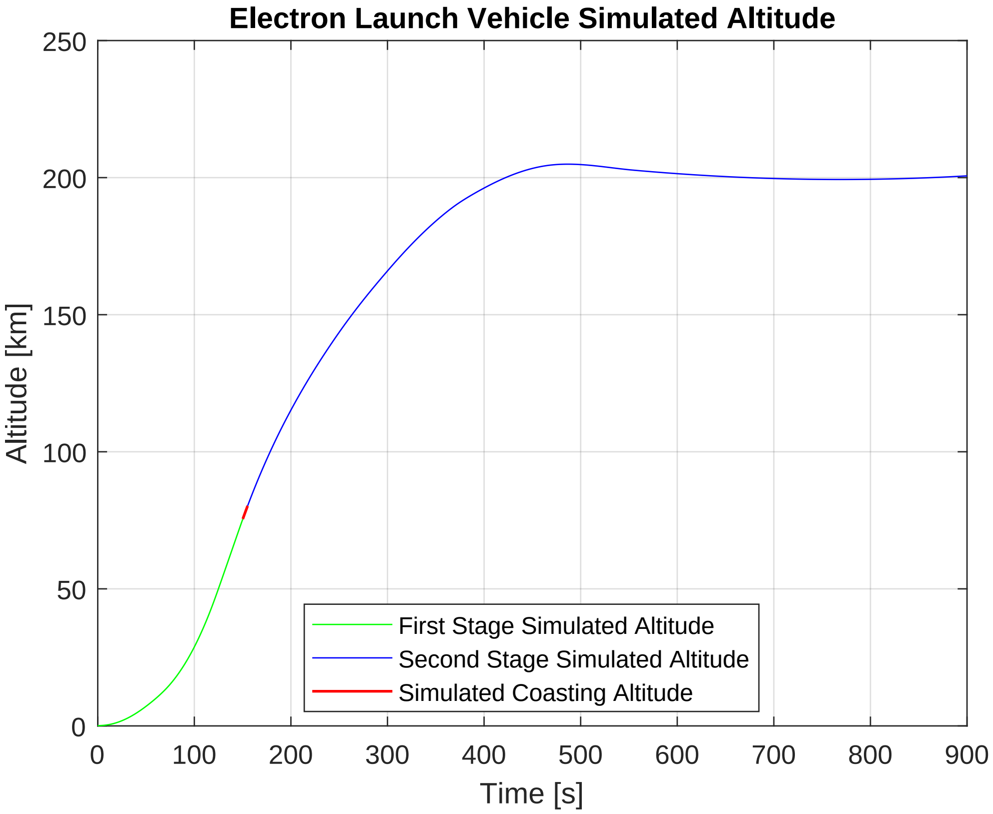


Figure C.3: Electron Altitude Profiles of Ascent Trajectory

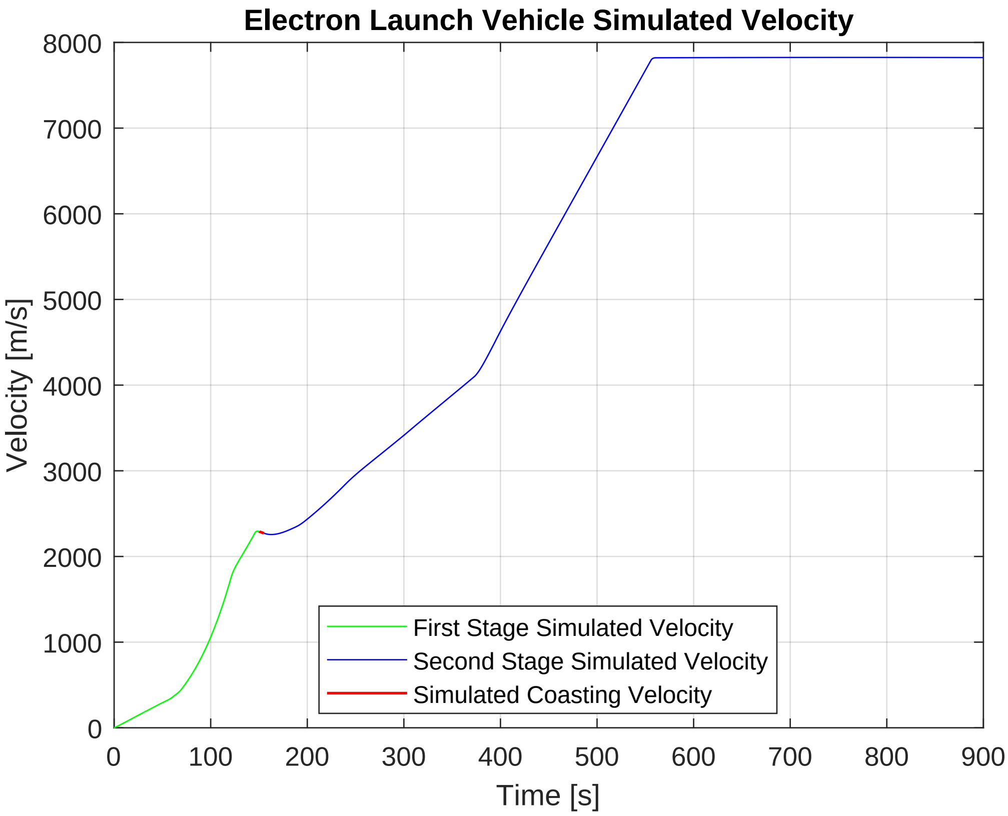


Figure C.4: Electron Velocity Profile of Ascent Trajectory

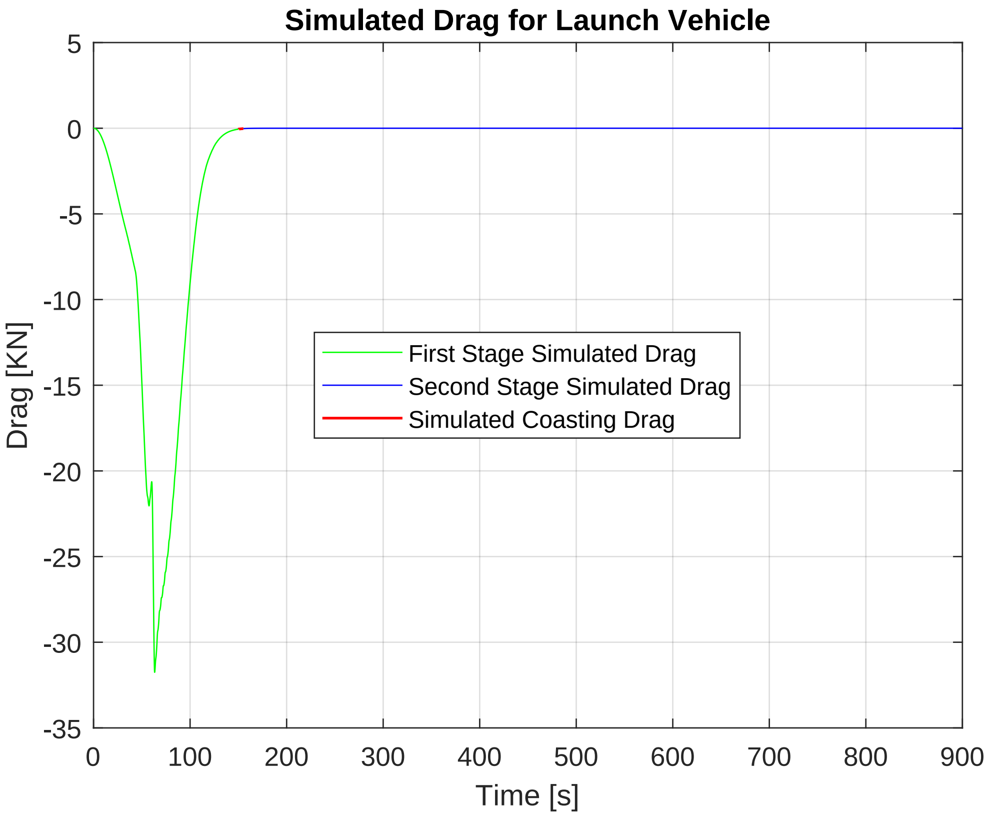


Figure C.5: Electron Drag Profile of Ascent Trajectory

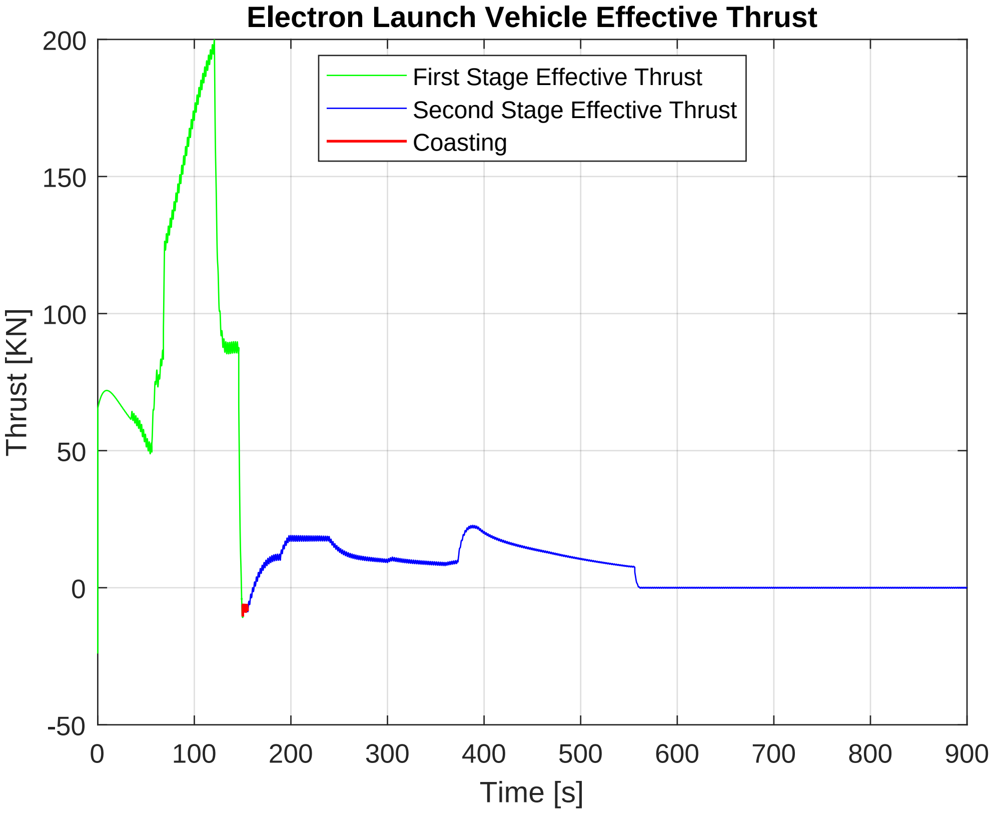


Figure C.6: Electron Effective Thrust Profile of Ascent Trajectory. Drag and Weight are Subtracted from the Thrust. Therefore, Effective Thrust

Table C.1: Second Stage Burnout ECEF Position and Velocity (Used for Hohmann Transfer Orbit Model)

Quantity	Value
ECEF Position [m]	$[-6.071299282808362 \times 10^{6}, 2.577416315657278 \times 10^{5}, -2.517249599841133 \times 10^{6}]$
ECEF Velocity [m/s]	$[-2.974740307269720 \times 10^{3}, 2.241249669341776 \times 10^{2}, 7.229858875041607 \times 10^{3}]$

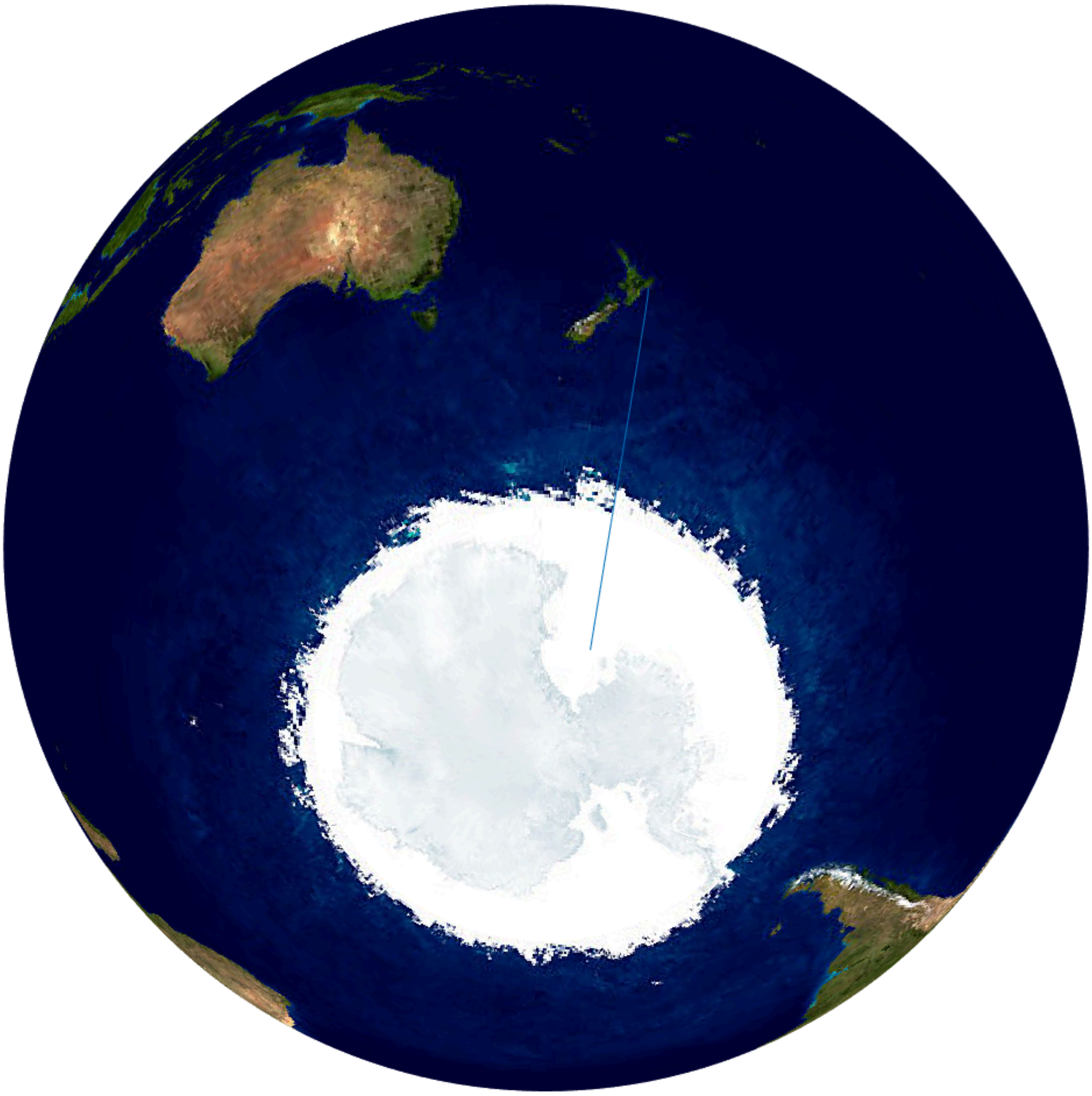


Figure C.7: Electron Launch Vehicle Ascent Trajectory from North to South Direction Launch Visualization

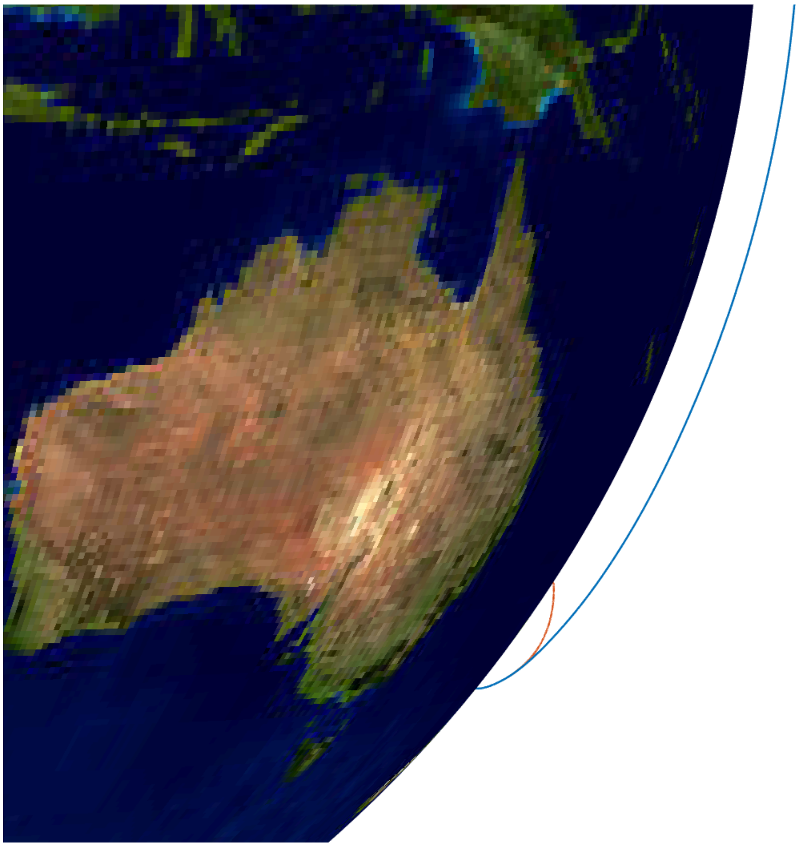


Figure C.8: Electron Launch Vehicle Ascent and First Stage Ballistic Trajectory Visualization in 3D

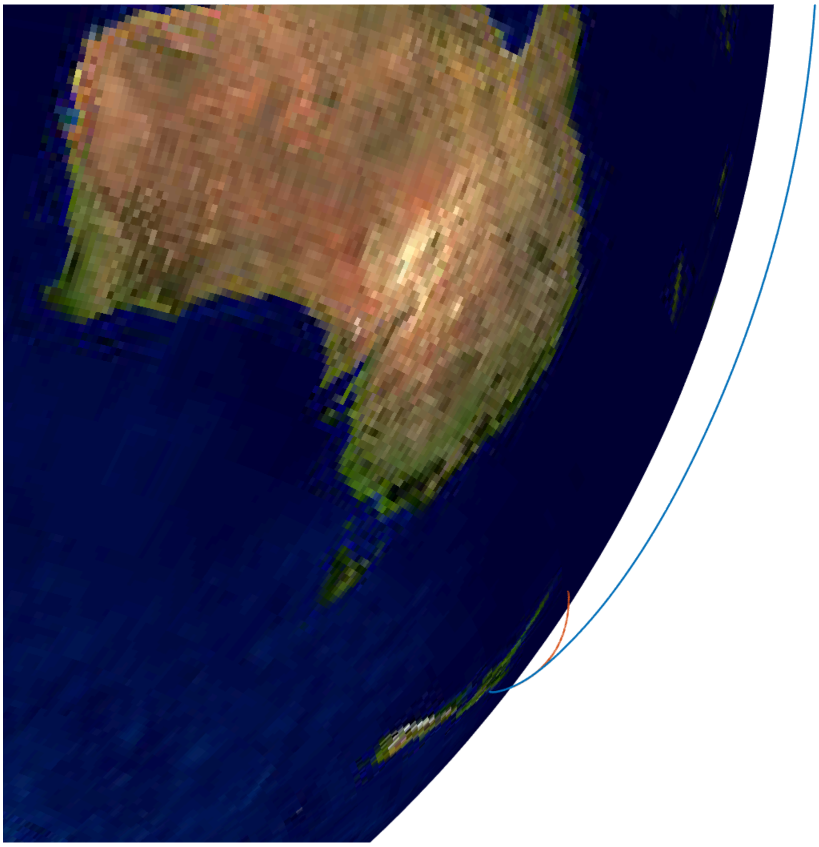


Figure C.9: Electron Launch Vehicle Ascent and First Stage Ballistic Trajectory Visualization in 3D

D

Recovery Phase

Table D.1: Final Re-entry Data to be Used as Input for the Main Parachute Simulink Model

Quantity	Value
ECEF Position [m]	$[-5.330152589214827 \times 10^6,\quad 2.072578939736822 \times 10^5,\quad -3.495916811194371 \times 10^6]$
ECEF Velocity [m/s]	[80.470882021928930, 3.247638146726317, 53.441968785947870]
LLA [deg, deg, m]	[33.416550787418930, 177.7732301389421, 6000]
Body Velocity [V_b] [m/s]	[23.637521364981720, 0.440793089671459, 93.718873871860150]
Euler Orientation [ϕ, θ, ψ] [rad]	[0.005959461892854, 0.249696920966741, 0.003102967475153]
Angular Velocity [ω_b] [rad/s]	$[4.010613057739172 \times 10^{-5},\quad -3.220465348970147 \times 10^{-4},\quad 5.455750594047451 \times 10^{-5}]$

E

Unsuccessful Initial Approach to the Equations of Motion

Before applying the equation of motion, as detailed in Section 11.6, various approaches were employed, based on the work of Yavuz [75]. The forces and moments acting on the system, as derived by Yavuz, are expressed in the following form:

$$\boldsymbol{F} = \begin{bmatrix} (m+\alpha_{11})(\dot{u}-vr)+(m+\alpha_{33})\,wq+(K+\alpha_{15})(\dot{q}+rp) \\ (m+\alpha_{11})(\dot{v}+ur)-(m+\alpha_{33})\,wp-(K+\alpha_{15})(\dot{p}-qr) \\ (m+\alpha_{33})\,\dot{w}-(m+\alpha_{11})(uq-vp)-(K+\alpha_{15})\left(p^2+q^2\right) \end{bmatrix} \tag{E.1}$$

$$\boldsymbol{M} = \begin{bmatrix} (I_{xx}+\alpha_{44})\,\dot{p}-(K+\alpha_{15})(\dot{v}-wp+ur)-\left(I_{yy}+\alpha_{44}-I_{zz}-\alpha_{66}\right)qr+(\alpha_{33}-\alpha_{11})\,vw \\ \left(I_{yy}+\alpha_{44}\right)\dot{q}+(K+\alpha_{15})(\dot{u}+wq-vr)+\left(I_{yy}+\alpha_{44}-I_{zz}-\alpha_{66}\right)pr-(\alpha_{33}-\alpha_{11})\,uw \\ (I_{zz}+\alpha_{66})\,\dot{r}+\left(I_{yy}-I_{xx}\right)pq \end{bmatrix} \tag{E.2}$$

The derivation of these equations is beyond the scope of this thesis; however, a detailed, step-by-step derivation can be found in the Journal of Aircraft article titled *Determining and Accounting for a Parachute Virtual Mass* [75]. While applying the rigid body assumption to calculate the velocity and trajectory using Equations E.1 and E.2, singularity-induced errors were observed. This numerical sensitivity and associated singularity-induced error were also identified by other researchers, as mentioned by Dellicker [74]. Their solution was to simplify the system by modeling it as a point mass with a 3-DOF EOM, rather than using the 6-DOF EOM. By adopting the approach of Oleg Yakimenko [69] and rewriting the aforementioned equations in vector form under the rigid body assumption, the following equations are derived [53]:

$$F = M_m\dot{V}+\Lambda_c\hat{M}_m, \quad M = I\dot{\Omega}+H_c\hat{I}+M_{\mathrm{cr}} \tag{E.3}$$

where,

$$\boldsymbol{M}_m = \operatorname{diag}\{m+\alpha_{11}, m+\alpha_{11}, m+\alpha_{33}\} \tag{E.4}$$

$$\boldsymbol{\Lambda}_c = \begin{bmatrix} -vr & wq & \dot{q}+rp \\ ur & -wp & qr-\dot{p} \\ vp-uq & 0 & -(p^2+q^2) \end{bmatrix} \tag{E.5}$$

$$\hat{\boldsymbol{M}}_m = [m+\alpha_{11}, m+\alpha_{33}, K+\alpha_{15}]^T \tag{E.6}$$

$$\boldsymbol{I} = \operatorname{diag}\left\{I_{xx}+\alpha_{44}, I_{yy}+\alpha_{44}, I_{zz}+\alpha_{66}\right\} \tag{E.7}$$

$$\boldsymbol{H}_c = \begin{bmatrix} wp - ur - \dot{v} & -qr & vw \\ wq - vr + \dot{u} & pr & -uw \\ 0 & 0 & 0 \end{bmatrix} \tag{E.8}$$

$$\hat{\boldsymbol{I}} = \left[K + \alpha_{15}, I_{yy} + \alpha_{44} - I_{zz} - \alpha_{66}, \alpha_{33} - \alpha_{11}\right]^T \tag{E.9}$$

$$\boldsymbol{M}_{\mathrm{cr}} = \left[0, 0, \left(I_{yy} - I_{xx}\right) pq\right]^T \tag{E.10}$$

Solving the equations with respect to body acceleration and angular acceleration yields [69]:

$$\dot{\boldsymbol{V}} = M_m^{-1}\left(\boldsymbol{F} - \boldsymbol{\Lambda}_c \hat{\boldsymbol{M}}_m\right), \quad \dot{\Omega} = I^{-1}\left(\boldsymbol{M} - \boldsymbol{H}_c \hat{I} - \boldsymbol{M}_{\mathrm{cr}}\right) \tag{E.11}$$

Although these equations could be used to obtain the velocity and trajectory of the circular parachute, the Simulink model based on them resulted in errors due to singularity. As a result, it was decided to proceed with the approach outlined in Section 11.6.

Bibliography

[1] Mack Crawford. *Render of Electron first stage descending under parachute.* https://www.nasaspaceflight.com/2020/01/rocket-lab-2020-dedicated-nro-mission/. Accessed: 12.8.2023.

[2] Nicholas Crisp, Katharine Smith, and Peter Hollingsworth. "Small Satellite Launch to LEO: A Review of Current and Future Launch Systems". In: *Transactions of the Japan Society for Aeronautical and Space Sciences* 12 (July 2014). DOI: 10.2322/tastj.12.Tf_39.

[3] Izzet Bayir. "A glimpse to future commercial spy satellite systems". In: *2009 4th International Conference on Recent Advances in Space Technologies.* 2009, pp. 370–375. DOI: 10.1109/RAST.2009.5158227.

[4] Rocket Lab. *launch: payload USER'S GUIDE.* Tech. rep. Rocket Lab, 2022.

[5] Madison Reidy. *It's do or die': Inside Rocket Lab's biggest mission yet.* https://www.youtube.com/watch?v=OCUlOiMKP6E&ab_channel=nzherald.co.nz. Accessed: 21.10.2023.

[6] Jack Beyer and Sawyer R. *Electrons Return to Flight, Neutron, and more - With Peter Beck - NSF Live.* https://www.youtube.com/watch?v=C3l87aPWUCk&ab_channel=NASASpaceflight. Accessed: 21.10.2023.

[7] Marco Lombardo and Andrea Togni. *Simulink Rocket Launch Simulation.* Accessed: 22.3.2023. MATLAB Ambassador-Italy. 2020. URL: https://www.youtube.com/watch?v=LrgYp7goMIE.

[8] Space Launch Report. *Rocket Lab Electron Data Sheet.* https://www.spacelaunchreport.com/electron.html#components. Accessed: 14.11.2023.

[9] Jacobus W Cornelisse, HFR Schoyer, and Karel F Wakker. "Rocket propulsion and spaceflight dynamics". In: *London: Pitman* (1979).

[10] Nick Lavars. *Rocket Lab to test advanced heat shield for Electron booster recovery.* https://newatlas.com/space/rocket-lab-heat-shield-upcoming-booster-recovery/. Accessed: 21.10.2023.

[11] Dean Jorgensen, Roy Haggard, and Glen Brown. "The past, present, and future of mid-air retrieval". In: *18th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar.* 2005.

[12] Adrien Finance et al. "In-Orbit Attitude Determination of the UVSQ-SAT CubeSat Using TRIAD and MEKF Methods". In: *Sensors* 21 (Nov. 2021), p. 7361. DOI: 10.3390/s21217361.

[13] Clayton Smith. "J2000. 0". In: *Highlights of Astronomy* 7 (1986), pp. 73–76.

[14] MH Xu, GL Wang, and M Zhao. "A new concept of the International Celestial Reference Frame: the epoch ICRF". In: *Monthly Notices of the Royal Astronomical Society* 430.4 (2013), pp. 2633–2637.

[15] Alexander V Nebylov and Joseph Watson. *Aerospace navigation systems.* John Wiley & Sons, 2016.

[16] MathWorks. *Comparison of 3-D Coordinate Systems.* https://nl.mathworks.com/help/map/choose-a-3-d-coordinate-system.html. Accessed: 25.10.2023.

[17] MathWorks. *Coordinate Systems for Modeling*. `https://nl.mathworks.com/help/aerotbx/ug/coordinate-systems-for-modeling.html`. Accessed: 20.10.2023.

[18] Erwin Mooij. *The motion of a vehicle in a planetary atmosphere*. Delft University Press, 1997.

[19] James R Wertz. *Spacecraft attitude determination and control*. Vol. 73. Springer Science & Business Media, 2012.

[20] Thanh Vu, Nicolas Rongione, and Amir Rahmani. "Minimalist Control Design of Canard-Actuated Micro-Projectiles". In: vol. 2015. Apr. 2015. DOI: `10.1109/SECON.2015.7133052`.

[21] Jack B Kuipers. "Quaternions and Rotation Sequences". In: (2002).

[22] Brian L. Stevens and Frank L. Lewis. *Aircraft Control and Simulation*. John Wiley & Sons, 1992.

[23] Peter H Zipfel. *Modeling and simulation of aerospace vehicle dynamics*. AIAA Education Series, 2000.

[24] American National Standards Institute. *Recommended Practice for Atmospheric and Space Flight Vehicle Coordinate Systems*. AIAA, 1992.

[25] Ralphm Toms. *An improved algorithm for geocentric to geodetic coordinate conversion*. Tech. rep. Lawrence Livermore National Lab, 1996.

[26] JM Picone et al. "NRLMSISE-00 empirical model of the atmosphere: Statistical comparisons and scientific issues". In: *Journal of Geophysical Research: Space Physics* 107.A12 (2002), SIA–15.

[27] US Standard Atmosphere. *U.S. standard atmosphere, 1976*. National Oceanic and Atmospheric Administration, 1976.

[28] Erwin Mooij. *Re-Entry Systems (AE4870B)-Lecture Notes*. Delft University of Technology, Faculty of Aerospace Engineering. 2017.

[29] Francisco Miranda. "Design Optimization of Ground and Air-Launched Hybrid Rockets". MA thesis. Delft University of Technology, 2015.

[30] M. D. Rozemeijer. "Launch Vehicle First Stage Reusability". MA thesis. Delft University of Technology, 2020.

[31] Jan Vandamme. "Assisted-Launch Performance Analysis Using Trajectory and Vehicle Optimization". MA thesis. Delft University of Technology, 2012.

[32] M.W. van Kesteren. "Air Launch versus Ground Launch: A Multidisciplinary Design Optimization Study of Expendable Launch Vehicles on Cost and Performance". MA thesis. Delft University of Technology, 2013.

[33] Stephane Contant. "Design and Optimization of a Small Reusable Launch Vehicle Using Vertical Landing Techniques". MA thesis. Delft University of Technology, 2019.

[34] Howard Curtis. *Orbital mechanics for engineering students*. Butterworth-Heinemann, 2013.

[35] T Mayer-Gürr et al. *The ITG-Grace 2010 gravity field model, Institute of Geodesy and Geoinformation, Bonn University, Bonn, Germany*. 2011.

[36] David Y Hsu. "Comparison of four gravity models". In: *Proceedings of Position, Location and Navigation Symposium-PLANS'96*. IEEE. 1996, pp. 631–635.

[37] N. Pavlis et al. "The Development and Evaluation of the Earth Gravitational Model 2008 (EGM2008) (vol 117, B04406, 2012)". In: *Journal of Geophysical Research* 118 (Apr. 2012).

[38] Francesco Castellini and Michele R Lavagna. "Comparative analysis of global techniques for performance and design optimization of launchers". In: *Journal of Spacecraft and Rockets* 49.2 (2012), pp. 274–285.

[39] Stijn Hoefsloot. “A feasibility study on the recovery of Electron’s first stage using a vertical landing approach”. MA thesis. Delft University of Technology, 2022.

[40] Huseyin Ozdemir. “Comparison of linear, cubic spline and akima interpolation methods”. In: *Hüseyin Özdemir* (2007).

[41] BTC Zandbergen. “AE4-S01 Thermal Rocket Propulsion”. In: *TUDelft Lecture Notes* (2010).

[42] MathWorks. *Custom Variable Mass 6DOF (ECEF) Quaternion.* `https://nl.mathworks.com/help/aeroblks/customvariablemass6dofecefquaternion.html`. Accessed: 15.4.2024.

[43] MathWorks. *Hohmann Transfer with the Spacecraft Dynamics Block.* `https://nl.mathworks.com/help/aeroblks/hohmann-transfer-with-the-spacecraft-dynamics-block.html`. Accessed: 10.3.2024.

[44] Boris Benedikter et al. “Convex optimization of launch vehicle ascent trajectories”. In: (2022).

[45] Stephen Clark. *Launch timeline for the Electron rocket’s second test flight.* 2017. URL: `https://spaceflightnow.com/2017/12/10/launch-timeline-for-the-electron-rockets-second-test-flight/` (visited on 06/05/2024).

[46] Ronald W. Schafer. “What Is a Savitzky-Golay Filter? [Lecture Notes]”. In: *IEEE Signal Processing Magazine* 28.4 (2011), pp. 111–117. DOI: `10.1109/MSP.2011.941097`.

[47] Matt Darley and Peter Beck. “Return to Sender: Lessons Learned from Rocket Lab’s First Recovery Mission”. In: (2021).

[48] Theo W Knacke. *Parachute recovery systems design manual.* Tech. rep. Naval Weapons Center China Lake CA, 1991.

[49] Reuben R Rohrschneider and Robert D Braun. “Survey of ballute technology for aerocapture”. In: *Journal of Spacecraft and Rockets* 44.1 (2007), pp. 10–23.

[50] Ian Clark et al. “An evaluation of ballute entry systems for lunar return missions”. In: *AIAA Atmospheric Flight Mechanics Conference and Exhibit.* 2006, p. 6276.

[51] SpaceNews. *Bringing a Rocket Back from Space: Rocket Lab to Recover Electron Booster on Next Mission.* `https://spacenews.com/bringing-a-rocket-back-from-space-rocket-lab-to-recover-electron-booster-on-next-mission/`. Accessed: 11.8.2024.

[52] MathWorks. *Custom Variable Mass 6DOF (Euler Angles).* `https://nl.mathworks.com/help/aeroblks/customvariablemass6dofeulerangles.html`. Accessed: 11.4.2024.

[53] Oleg A Yakimenko. *Precision aerial delivery systems: modeling, dynamics, and control.* American Institute of Aeronautics and Astronautics, Inc., 2015.

[54] Bernard Etkin. *Dynamics of atmospheric flight.* Courier Corporation, 2005.

[55] Jasmine Rimani. “High Lift Systems for Planetary Descent and Landing”. PhD thesis. Politecnico di Torino, 2018.

[56] United States. Defense Mapping Agency. *Department of Defense World Geodetic System 1984: its definition and relationships with local geodetic systems.* Vol. 8350. Defense Mapping Agency, 1987.

[57] MathWorks. *Horizontal Wind Model.* `https://nl.mathworks.com/help/aeroblks/horizontalwindmodel.html`. Accessed: 15.5.2024.

[58] *U.S. Military Specification MIL-F-8785C.* Tech. rep. U.S. Department of Defense, Nov. 1980.

[59] MathWorks. *Discrete Wind Gust Model.* `https://nl.mathworks.com/help/aeroblks/discretewindgustmodel.html`. Accessed: 16.5.2024.

[60] *U.S. Military Handbook MIL-HDBK-1797B*. Tech. rep. U.S. Department of Defense, Apr. 2012.

[61] *U.S. Military Handbook MIL-HDBK-1797*. Tech. rep. U.S. Department of Defense, Dec. 1997.

[62] MathWorks. *Von Karman Wind Turbulence Model*. `https://nl.mathworks.com/help/aeroblks/vonkarmanwindturbulencemodelcontinuous.html`. Accessed: 16.5.2024.

[63] Airborne Systems. *FIREFLY: guided precision areal delivery system*. `https://airborne-sys.com/product/firefly-military-cargo-delivery-parachute/`. Accessed: 26.2.2024.

[64] T.J. Becx. *Quantum Launch System*. Tech. rep. Delft University of Technology, 2016.

[65] Peter Lissaman and Glen Brown. "Apparent mass effects on parafoil dynamics". In: *Aerospace Design Conference*. 1993, p. 1236.

[66] Rayan Mazouz, Marco B Quadrelli, and Erwin Mooij. "Convex optimization guidance for precision landing on titan". In: *AIAA Scitech 2021 Forum*. 2021, p. 1345.

[67] RD Blevins. *Applied fluid dynamics handbook*. 1984.

[68] Marco B Quadrelli et al. "Aero maneuvering dynamics and control for precision landing on titan". In: *2019 IEEE Aerospace Conference*. IEEE. 2019, pp. 1–16.

[69] Vladimir N Dobrokhodov, Oleg A Yakimenko, and Christopher J Junge. "Six-degree-of-freedom model of a controlled circular parachute". In: *Journal of Aircraft* 40.3 (2003), pp. 482–493.

[70] NASASpaceflight. *Electrons Return to Flight, Neutron, and more - With Peter Beck - NSF Live*. `https://www.youtube.com/watch?v=C3l87aPWUCk`. Accessed: 12.12.2023.

[71] SK Ibrahim. *Experimental Determination of the Apparent Moment of Inertia of Parachutes. Air Force Flight Dynamics Lab, Wright-Patterson Air Force Base, Ohio*. Tech. rep. FDL-TDR-64-153, December, 1964.

[72] T Yavuz and D Cockrell. "Experimental determination of parachute apparent mass and its significance in predicting dynamic stability". In: *7th Aerodynamic Decelerator and Balloon Technology Conference*. 1981, p. 1920.

[73] Frank M. White and Dean F. Wolf. "A theory of three-dimensional parachute dynamic stability." In: *Journal of aircraft* 5.1 (1968), pp. 86–92.

[74] Scott Dellicker and Jim Bybee. "Low cost parachute guidance, navigation, and control". In: *15th Aerodynamic Decelerator Systems Technology Conference*. 1999, p. 1706.

[75] T Yavuz. "Determining and accounting for a parachute virtual mass". In: *Journal of Aircraft* 26.5 (1989), pp. 432–437.