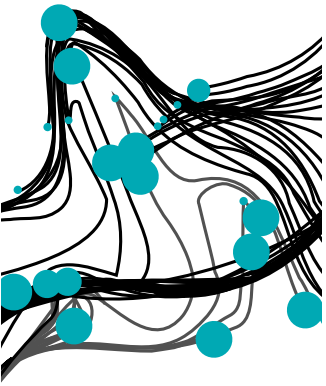
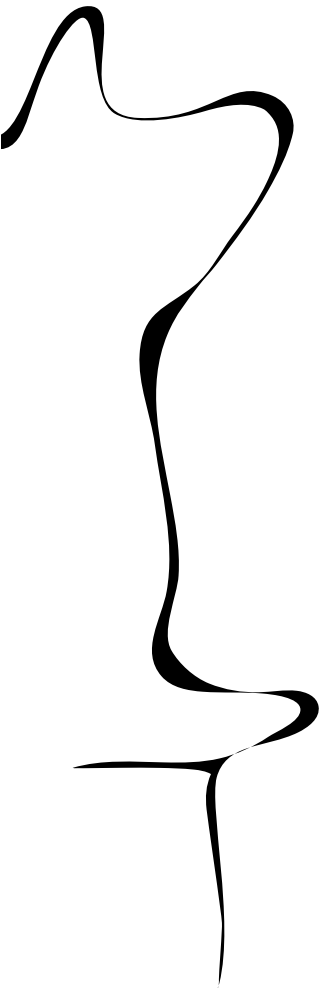


RAM ROBOTICS AND MECHATRONICS



ENHANCING LANDING ACCURACY THROUGH FIDUCIAL MARKER-BASED VISUAL NAVIGATION IN VTVL GNC SYSTEMS: A SIMULATION-BASED STUDY

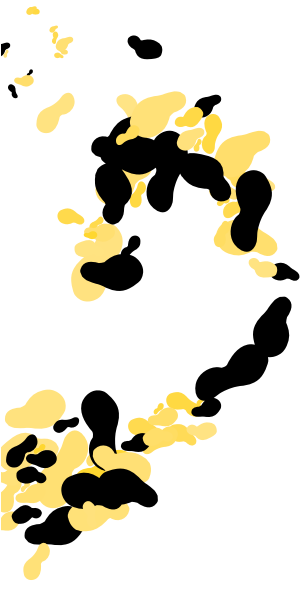
D. (Daniel) Ciulei

MSC ASSIGNMENT

Committee:
dr. ir. M. Abayazid
M.S. Selim, MSc
dr. V.V. Lehtola

January, 2025

001RaM2025
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands



UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Abstract

This study aims to develop and evaluate the Guidance, Navigation, and Control (GNC) system for a Vertical Takeoff and Vertical Landing (VTVL) - also called rocket hoppers - using the Alpha rocket designed by the UT/VU student team RISE. By integrating fiducial marker-based visual navigation, the GNC system seeks to enhance accuracy, precision, and robustness during the landing phase of a simulated hop test. A co-simulation environment enables the assessment of visual and inertial sensor fusion in state estimation, particularly in the final descent and landing stages. The system's performance is evaluated with and without visual navigation across multiple simulations under varying noise and disturbance conditions. Key metrics, including control stability, position estimation accuracy, landing precision and robustness, highlight the benefits of visual navigation integration. The results aim to establish a foundation for implementing the designed GNC subsystem on the physical Alpha rocket hopper, advancing its capabilities for accurate and robust autonomous landings.

Contents

1 Introduction

In recent years, the landing of rockets has emerged as a focal point within the aerospace industry, garnering significant attention from industry leaders [1], [2] and student teams alike [3]. The ability to achieve precise and reliable landings not only enhances the efficiency of space missions [4] but also plays a pivotal role in advancing technologies related to reusability and cost-effectiveness [5]. Amidst this backdrop, student-led rocketry teams face unique challenges, navigating budget constraints while striving to push the boundaries of innovation and exploration.

RISE was founded as a student team from both the Vrije Universiteit van Amsterdam (VU) and University of Twente UT) in September of 2021. The name RISE contains the core concepts “Rocketry Innovations” and “Space Engineering”. The idea behind “Rocketry Innovations” is to encourage innovations in the area of rocketry, and “Space Engineering” aims to focus on the design, development, production and testing of all types of space systems and components. The current focus of RISE, under the name Project Alpha, is the research and development of a rocket able to perform a propulsive rocket landing on a designated landing platform, and making the rocket reusable for multiple flights with minimal refurbishment required. It is this with this rocket (shown in Figure 1) that RISE intends competitions to showcase the capabilities of a reusable rocket used during student competitions.

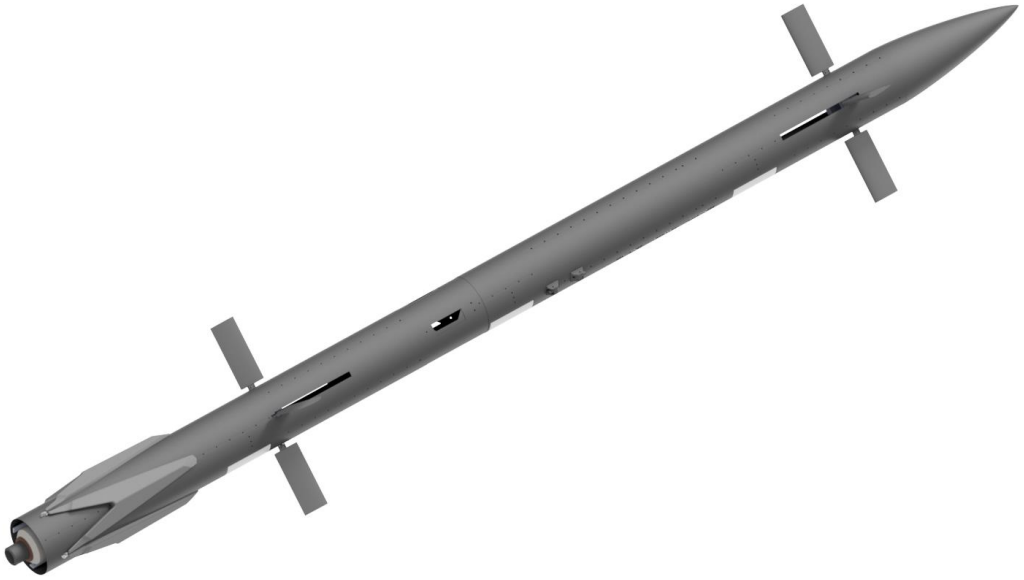


Figure 1: The Alpha rocket with stabilizing ascent support and descent support fins deployed

At the core of this endeavor lies a fundamental problem: How can a student-led rocketry team, operating within budget constraints, develop a robust Guidance, Navigation and Control (GNC) subsystem capable of achieving precise positioning and control during the ascent, descent, and landing phases of a rocket mission. Addressing this challenge necessitates a multifaceted approach, encompassing the integration of navigation technologies, control algorithms, and comprehensive simulation-based studies.

The objectives of the project are two-fold. Firstly, the development of a 6 Degrees of Freedom (6DoF) simulation environment serves as the cornerstone, providing a platform for iterative testing and refinement of the GNC subsystem. Secondly, developing a Navigation module to enhance the accuracy and robustness of the rocket’s navigation, thereby also designing the Guidance and Control modules, tailored to the Alpha’s unique mission requirements and constraints. By incorporating landing markers for visual navigation, the GNC subsystem seeks to navigate the Alpha rocket through key mission milestones further described in 1.4.

1.1 Literature Review

Recent advancements in rocket landing technologies have garnered significant attention, as evidenced by industry trends and student competitions focused on reusable launch vehicles. These endeavors highlight the growing emphasis on precision landing and reusability within the aerospace sector.

Research in this domain has explored various facets of GNC systems essential for achieving successful rocket landings.

In one study, [6] explores the potential of vision-based navigation and hazard detection systems in enhancing rocket flight safety and precision landing capabilities. By introducing the Terrain-Relative Navigation & Descent Imager (TRNDI) system, the study showcases the feasibility of real-time terrain mapping and hazard assessment using onboard optical payloads. The modular software framework enables seamless integration with existing autonomous GNC systems, paving the way for closed-loop navigation and trajectory adjustments based on visual feedback. Simulated results demonstrate the system's reliability in identifying suitable landing sites and assessing landing region safety, thus augmenting overall mission success rates.

Figure 2: Masten's Xombie Space-Access technology demonstrator used by NASA to develop the GENIE GNC system [7]

Another study [8] presents the design, conception, and testing of EAGLE, a platform for flight testing of GNC algorithms and systems. The research mentions that the platform is designed to test new and advanced GNC algorithms that employ a base set of sensors and actuators typically present on such vehicles. It also mentions that the platform should provide the option for an additional small payload, for example, enhancing the on-board avionics with different or more precise sensors. This could potentially include vision-based sensors.

[9] presents a comprehensive analysis of navigation and guidance algorithms deployed during a terrestrial suborbital rocket flight, offering valuable insights into real-world performance and applicability. The study highlights the successful integration of an extended Kalman Filter (EKF) navigation algorithm and a trajectory optimization framework, demonstrating the system's efficacy in achieving precision altitude control and landing dynamics simulation. The study also presents the practical relevance of such algorithms in extraterrestrial landing scenarios, providing invaluable data for mission planning and technology development.

Further exploration of GNC solutions for launcher return missions was conducted by [10]. Their work discusses the development of a GNC solution for the return mission of a launcher. The guidance strategy is based on direct optimal control methods via on-board optimization, which is necessary to satisfy the pinpoint landing requirement in a high uncertain dynamic system, such as a booster recovery mission. Online convex optimization and successive convexification are explored for the design of the guidance function.

[11] discusses the development and evaluation of a high-fidelity simulation platform for precision landing. NASA's science and exploration goals to return to the Moon and beyond will need to perform

precision landings to place humans and cargo supplies near places of scientific interest, surface resources, or pre-established base camps. With the maturation of new navigation technology, such as terrain relative navigation, precision landing is now feasible, enabling new exploration sites, such as the lunar poles.

The available literature presents possible opportunities for enhancing GNC development for unique mission objectives and requirements created by student teams. While existing studies have explored various sensor modalities and navigation algorithms for VTVL systems, the integration of landing markers as a means to augment traditional sensor suites has received limited attention, but it is extensively used in drone research [12] with the ArUco markers being a popular choice for aiding the landing and tracking of drones. Thus, visual navigation could potentially enhance the success rate of VTVL landing missions by fusing data retrieved from landing markers with existing sensor inputs.

In light of the insights gained from existing literature, the present study seeks to address this gap by exploring the feasibility and efficacy of integrating visual navigation in the form of fiducial marker detections into the GNC subsystem of the Alpha rocket.

1.2 Problem Definition and Objective

With Alpha, RISE wants to achieve apogee of 3 [km] and to perform a retro-propulsive landing, but before attempting such a feat initial low-altitude/low-velocity "hop" tests will be conducted. This a type of flight test where a rocket prototype performs a short, controlled flight, typically involving a vertical takeoff, a brief hover, and a vertical landing. These tests are crucial for evaluating the rocket's engines, control systems, and landing mechanisms [13]. The results from these tests help engineers to gather data and make necessary adjustments before attempting more complex and higher-altitude flights.

In preparation for the hop tests, RISE intends to develop a robust and adaptable GNC system to ensure the rocket's precise and safe operation throughout its flight phases. To facilitate this development and thoroughly evaluate the GNC system's performance, a comprehensive simulation environment is constructed by coupling Unreal Engine 5 and MATLAB's Simulink. This co-simulation environment serves as a virtual testbed, enabling iterative testing and refinement of the GNC algorithms in a controlled and realistic setting. By simulating the rocket's dynamics, sensor inputs, and control responses within this environment, the team can identify and address potential issues, optimize performance, and enhance the reliability of the GNC system before proceeding to actual flight tests.

Unreal Engine 5 has been chosen for its ability to create photorealistic environments with ray tracing, which is an important factor to consider for simulating the visual input the rocket would receive during the hop test. This simulation allows for the testing of a key aspect of the GNC system: the integration of ArUco marker information. By sending back camera images from the virtual rocket to the Simulink environment, image processing can be applied to extract valuable data from these markers. The processed data is then fused into an Extended Kalman Filter which is part of the Navigation module. The fusion of visual and inertial data is expected to enhance the GNC system's accuracy and robustness, ultimately contributing to the successful execution of the hop test and future flights. Thus, the following research question can be formulated: How can visual navigation, in the form of fiducial marker detections, be integrated into the GNC subsystem of a rocket to enhance landing accuracy and robustness?

1.3 The Alpha Rocket

It is important to note that the Alpha rocket, along with its rocket engine, is currently in the development phase. As such, the presented rocket design, including its control surfaces, onboard sensors, and other components, may undergo modifications and refinements as the project progresses. However, the core concept of the rocket, along with the GNC system developed in this research, will serve as a fundamental baseline for future iterations. This research aims to establish a solid foundation for the GNC system's functionality and integration with the rocket's hardware, ensuring a robust and adaptable framework that can accommodate potential design changes in the future. Thus, if otherwise stated, the rocket described in this section will be used for the co-simulation.

1.1.1 Rocket Airframe

The airframe is rocket's body which houses the propulsion system, avionics, and payload, while

providing structural integrity during flight. The airframe's design significantly influences the rocket's aerodynamic characteristics, stability, and overall performance. Presented in Table 1: Rocket airframe specification are the specifications of the rocket airframe, encompassing rocket height, engine, propellant for both wet mass (fully fueled) and dry mass (after propellant depletion) parameters.

Parameter	Value	Unit
Height	3.081	$[m]$
Wet Mass	20	$[kg]$
Dry Mass	15	$[kg]$

Table 1: Rocket airframe specification

Apart from these parameters it is important to know the wet and dry mass inertia parameters and where the Center of Gravity (CoG) which is the location about which the total forces and moments experienced by the rocket's body act about. Presented in Table 2 are the principal axes of inertia along with the CoG of the rocket's body.

Wet Mass Parameter	Value(s)	Unit
$\boldsymbol{I_{xx}}$	0.052	$[kg \cdot m^2]$
$\boldsymbol{I_{yy}}$	16.365	$[kg \cdot m^2]$
$\boldsymbol{I_{zz}}$	16.365	$[kg \cdot m^2]$
CoG	$(1.117, 0, 0)$	$[m]$

Table 2: Rocket Wet Mass specification

To create thrust, the rocket burns its propellant, as such the mass, inertia and the CoG of the rocket will decrease over the course of the trajectory. Presented in Table 3: Rocket Dry Mass specifications are the principal axes of inertia along with the CoG of the rocket's body.

Dry Mass Parameter	Value(s)	Unit
$\boldsymbol{I_{xx}}$	0.037	$[kg \cdot m^2]$
$\boldsymbol{I_{yy}}$	12.308	$[kg \cdot m^2]$
$\boldsymbol{I_{zz}}$	12.308	$[kg \cdot m^2]$
CoG	$(1.01, 0, 0)$	$[m]$

Table 3: Rocket Dry Mass specifications

1.1.2 TVC and Control Surfaces

The Alpha rocket is using a Hybrid Rocket Engine (HRE), nicknamed Green Phoenix, as its propulsion system, as one of the biggest reasons to focus on this type of engine is due to the fact that the Alpha rocket must be able to have a varying thrust output throughout its entire trajectory.

The engine is equipped with a Thrust Vectoring Control (TVC) system and Jet Vanes (JV) at the exit of the rocket engine's nozzle. With this system, the TVC allows Alpha to generate a pitching and yawing moment, while the JV can generate the rolling moments [14], all of which act about CoG of the rocket to propel it to the desired points of the trajectory.

Naturally, the rocket is also equipped with landing legs which are released before landing. Once released, the landing legs are being brought downwards by the force of gravity and are latched in place. Showcased in Figure 3 shows the landing being released.

Figure 3: Deploying of the landing legs

Although not used in the hop test nor in this project, but in order to explain more about the rocket seen in Figure 1, apart from the active controls described above, the rocket is also equipped with two sets of stabilizing fin groups, each group with 4 fins radially displaced 90° apart from each and placed at the top and bottom of the rocket. These fin groups are named Stabilizing Ascent Mechanism (SAM) and Stabilizing Descent Mechanism (SDM) which can be seen in Figure 4. The SAM is deployed during the ascent phase of Alpha to stabilize the rocket, and as the name suggests the SDM is deployed during the descent phase while the SAM is folded inside. This is done to stabilize the rocket during these different phases of the flight, as the rocket is only stable if the CoG is above the Center of Pressure (CoP) of the rocket [15].

Because the rocket will only ascend to an altitude of 30 [m] above the surface of the launch location, for the hop test the SMD and SAM will not be used. As the rocket will not be subjected to high aerodynamic forces, as in contrast to a high-altitude/high-velocity flight, these control surfaces and the resulting aerodynamic forces will not be considered in the co-simulation.

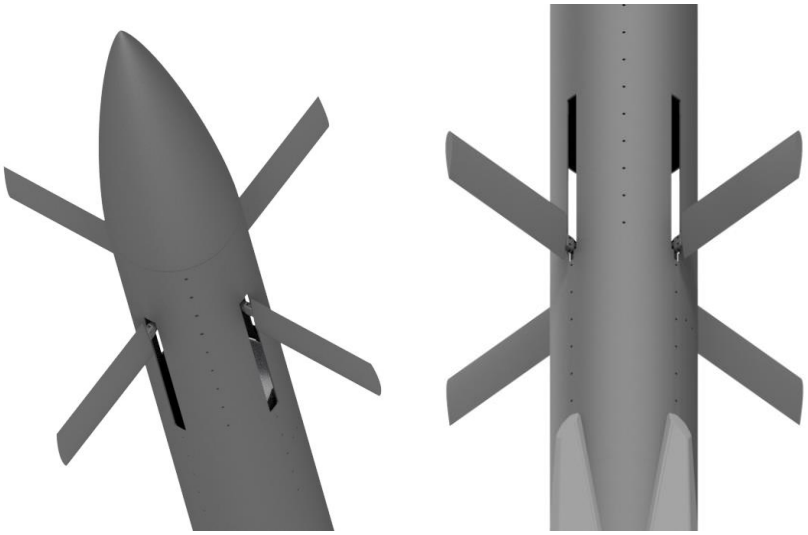


Figure 4: SDM and SAM deployed

1.1.3 Sensor suite

The MTi-7, a GNSS/INS module [16], will serve as the primary sensor module for the Alpha rocket. This module integrates a 3D accelerometer, a 3D gyroscope, and a magnetometer, along with the capability to incorporate data from an external GNSS receiver. The GNC system will leverage the accelerometer and gyroscope data for inertial measurements, while the GNSS receiver will provide global positioning information. Presented in Table 4 are the specifications for the sensors which will

be used to model the sensors inside the Simulink environment. Note that the MTi-7 is also equipped with a magnetometer, but it will not be used throughout this project.

MTi-7 Parameter	Value	Unit
Sampling Frequency	100	$[Hz]$
Accelerometer		
Full Range	± 16	$[g]$
In-Run Bias Stability	0.03	$[mg]$
Bandwidth (-3dB)	324 (Z: 262)	$[Hz]$
Noise Density	120	$\left[\frac{\mu g}{\sqrt{Hz}}\right]$
Gyroscope		
Full Range	± 2000	$\left[\frac{^\circ}{s}\right]$
In-Run Bias Stability	10	$\left[\frac{^\circ}{h}\right]$
Bandwidth (-3dB)	255	$[Hz]$
Noise Density	0.007	$\left[\frac{^\circ \sqrt{Hz}}{s}\right]$
Scale Factor Variation	0.5	%
GNSS		
Horizontal Position Accuracy	1	$[m]$
Vertical Position Accuracy	2	$[m]$
Velocity Accuracy	0.05	$\left[\frac{m}{s}\right]$

Table 4: MTi-7 Module Specifications

As seen in the table above the horizontal and vertical position accuracy of the GNSS can be considered too large to be used for accurate landings, this is due to the fact that the GNSS module of the MTI-7 does not use Real-time kinematic (RTK) positioning [17] which has a typical accuracy between 1 $[cm]$ and 5 $[cm]$ [18], but depending on the manufacturer it can cost 5 to 10 times as much as standard GNSS module.

To overcome such a price difference, RISE has decided to take a more cost-effective approach. Apart from the MTi-7, the rocket is also equipped with a Raspberry Pi Camera Module 3 Wide [19]. This is a versatile HD FPV camera designed for radio-controlled models such as drones. It combines a high-definition camera with a built-in recording module, allowing it to capture 720 $[px]$ videos at 60 $[fps]$ while providing a live feed for first-person view (FPV) applications. Its primary function will be to detect and track the landing marker, providing real-time visual feedback for enhancing landing accuracy during the rocket's descent phase.

1.4 VTVL Hop Flight Trajectory

For the hop test, RISE intends to attempt and complete the following mission objectives which are tied to the flight phases of the Alpha rocket:

- ***Flight Phase I (Vertical Ascent and Hover):*** The rocket will launch vertically from the launch pad and ascend to an altitude of 30 meters. Upon reaching this altitude, the rocket will maintain a stable hover for a predetermined duration. This phase will test the rocket's ability to achieve a controlled ascent and maintain stability at a specific altitude.
- ***Flight Phase II (Lateral Maneuver):*** While maintaining the 30-meter altitude, the rocket will perform a lateral maneuver, translating 20 meters horizontally. This phase will assess the rocket's maneuverability and control systems' ability to execute precise lateral movements.
- ***Flight Phase III (Descent and Landing):*** After completing the lateral maneuver, the rocket will initiate a controlled descent and touch down safely at the designated landing site, located 20 meters away from the launch pad. This phase will validate the functionality and accuracy of the rocket's landing mechanisms.

The hop test trajectory for the Alpha rocket, as depicted in Figure 5, outlines a controlled flight path designed to evaluate its ascent, maneuverability, and landing capabilities. The trajectory resembles a distinctive "Π" shape, encompassing the vertical ascent, horizontal translation, and vertical descent segments. It is important to note that this trajectory represents a simulation-based study and the actual flight test location will be determined based on safety regulations and operational constraints.

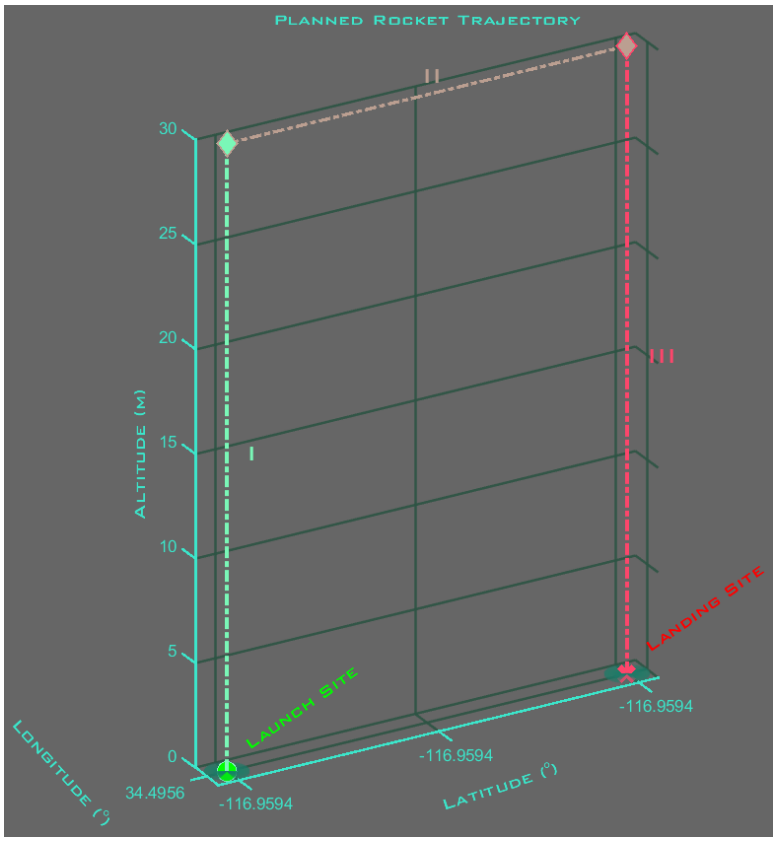


Figure 5: Proposed trajectory for the Alpha rocket hopper

2 Theoretical Background

Creating a co-simulation platform for the VTVL vehicle requires the definition of the reference frames and modelling of the flight dynamics of the rocket. Reference frames are essential for defining the rocket's position, orientation, and motion relative to its environment. The modeling of rocket dynamics provides the equations governing the rocket's forces, torques, and propulsion system, while the modeling of rocket kinematics describes the rocket's motion and state transitions over time. These elements collectively form the basis for constructing the co-simulation platform and optimizing the rocket's behavior throughout its flight phases.

2.1 Reference Frames

Different reference frames are essential for the GNC to accurately track and control the rocket's motion [20]. The rocket's body itself is a non-inertial reference frame, meaning it experiences accelerations and rotations. In this frame, the concept of position is relative and only meaningful when described with respect to an external inertial reference frame, thus it is essential to describe the reference frames which Alpha uses.

Figure 6 illustrates the coordinate transformations involved with the Navigation module of the co-simulation, while also bridging the virtual environment of Unreal Engine 5 (UE5) with the GNC implemented in Simulink.

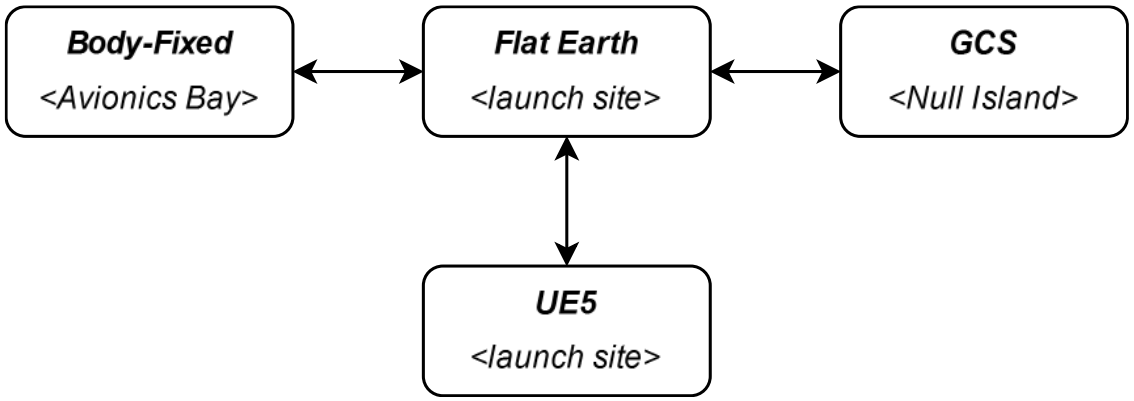


Figure 6: Coordinate system transformations

The reference frames illustrated above are used in the co-simulation, these are:

- ***Body-Fixed:*** this frame is rigidly attached to the rocket's CoG. Its axes move and rotate with the rocket, making it the most intuitive for understanding the rocket's orientation and attitude changes. Sensor data from the INS sensors are given in this frame.
- ***Flat-Earth:*** this is a local tangent plane approximation of the Earth's surface, centered at the launch site. It is assumed to be flat and non-rotating. This simplification is valid for short-range flights where the Earth's curvature has minimal impact. The body-fixed frame is transformed to this frame to understand the rocket's motion relative to the launch site.
- ***GCS*:** stands for Geodetic Coordinate System. Its origin is at Null Island, a location in the Gulf of Guinea used as a reference point for mapping. It uses *latitude*, *longitude* and *altitude* $LLA = (\phi_{LLA}, \lambda_{LLA}, h_{LLA})$ to represent the relative position of the rocket on the surface of the planet. GNSS data is received in this frame.
- ***UE5*:** simulates the rocket's motion inside the photorealistic environment using its own coordinate system [21]. The rocket's position and orientation in the Flat-Earth frame are sent from Simulink to the UE5 environment and computes the position of the rocket with respect to the launch site inside its environment.

2.1.1 Body-Fixed to Flat-Earth coordinates

Shown in Figure 7 is the rocket's Body-fixed reference frame. Here the X_b is aligned with the rocket's longitudinal axis, while the Y_b and Z_b form an orthogonal frame.

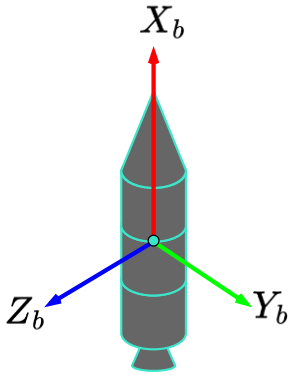


Figure 7: Body-fixed reference frame

The Flat-Earth is the inertial reference frame in which the rocket moves and rotates, this is illustrated in Figure 8. The frame's origin is the launch site, with the X_E pointing upwards with the remaining axes of Y_E and Z_E form the orthogonal frame.

The sequential rotation of a body frame relative to an Earth frame is achieved by utilizing three Euler angles (ϕ, θ, ψ) in the following order: $R(\phi, \theta, \psi) = R_z(\psi)R_y(\theta)R_x(\phi)$. In this sequence, ϕ represents the roll angle, which is the rotation of the body in the Flat-Earth frame. While the θ and ψ represents the pitch and yaw angles, of the y and z axes respectively.

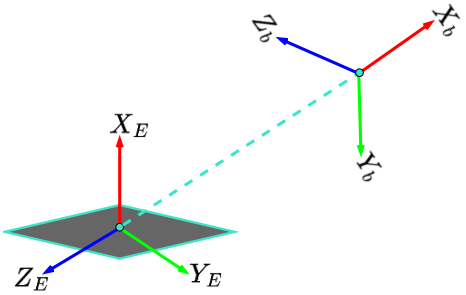


Figure 8: Flat-Earth reference frame

Presented in *Eq. 1* is the transformation matrix [22] associated with transforming the coordinates from the Body-fixed frame to the Flat-Earth frame, where the *s* and *c* letters are placeholders for the sin() and cos() trigonometric functions respectively.

$$R_E^b = \begin{bmatrix} c_\theta c_\psi & s_\phi c_\theta s_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \quad \textit{Eq. 1}$$

2.1.2 Flat-Earth to UE5 coordinates

As shown in Figure 6, from the Flat-Earth reference frame the Simulink environment transmits the position of the rocket to the UE5 environment.

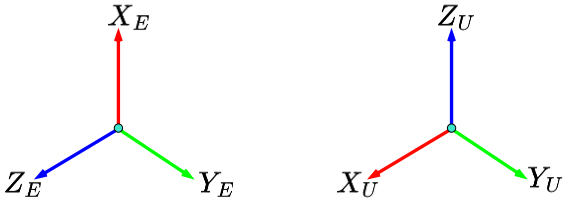


Figure 9: Difference between Flat-Earth and UE5 axes

Shown in Figure 9 is the difference between the axes of the coordinate systems. From this figure one can deduce that there is a need of two rotation to complete the transformation from the Flat-Earth frame to the UE5 frame, namely a first rotation of 90° around the y axis and secondly a rotation of 180° around the z axis, *Eq. 2* represents the resulting rotation matrix about these axes.

$$R_{UE4}^{E} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \qquad \textit{Eq. 2}$$

2.1.3 GCS to Flat-Earth coordinates

Geodetic coordinates allow for global navigation, enabling the rocket to pinpoint their exact position anywhere on the Earth's surface using LLA. The transformation between the two coordinate systems can be computed using the geographic coordinate conversion [23]. The transformation from LLA to the Flat-Earth reference frame involves a series of coordinate transformations that approximate the Earth as flat near a chosen reference point $LLA_{ref} = (\phi_{LLA_{ref}}, \lambda_{LLA_{ref}}, h_{LLA_{ref}})$, where $\phi_{LLA_{ref}}$ [°] is the reference latitude, $\lambda_{LLA_{ref}}$ [°] is the reference longitude and $h_{LLA_{ref}}$ $[m]$ is the reference altitude. The transformation converts the LLA coordinates into coordinates relative to the reference point. For small regions, this is a linear transformation. The transformation is defined as:

$$X_E = \begin{bmatrix} x_E \\ y_E \\ z_E \end{bmatrix} = T_E^{LLA} \cdot \begin{bmatrix} \phi - \phi_{ref} \\ \lambda - \lambda_{ref} \\ h - h_{ref} \end{bmatrix} \qquad \textit{Eq. 3}$$

where x_E, y_E and z_E are the coordinates in the Flat-Earth reference frame in meters. The transformation matrix which maps the LLA coordinates into the local Flat-Earth reference coordinate frame is defined as:

$$T_E^{LLA} = \begin{bmatrix} M_\phi & 0 & 0 \\ 0 & N_\lambda \cos\phi_{ref} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \textit{Eq. 4}$$

where M_ϕ $[m]$ is the Meridian radius and N_λ $[m]$ is the Prime vertical radius of curvature [24].

2.1.4 Camera Image frame to Flat-Earth

The ArUco pose estimation algorithm [25] is used to detect the fiducial marker near the landing site, providing the translation vector of the marker relative to the Camera Image frame. This translation vector represents the position of the marker in the camera's coordinate system. Shown in Figure 10 is overview of how the Camera Image frame is related to the Flat-Earth frame.

To localize the rocket in the Flat-Earth frame, the detected position of the marker within the Camera Image frame must first be transformed into the rocket's Body-fixed frame by applying the known extrinsic parameters between the camera and the rocket, this transformation is given by:

$${}^{b}t_{rocket} = R_C^{b^{-1}} \cdot {}^{c}t_{image} + {}^{b}t_{camera} \qquad \textit{Eq. 5}$$

where ${}^{b}t_{camera}$ translation vector which represents the camera offset from the center of the rocket's Body-fixed frame, the ${}^{c}t_{image}$ is the translation vector retrieved by the ArUco pose estimation algorithm, the R_C^b is the rotation matrix from the rocket's Body-fixed frame to the Camera Image frame.

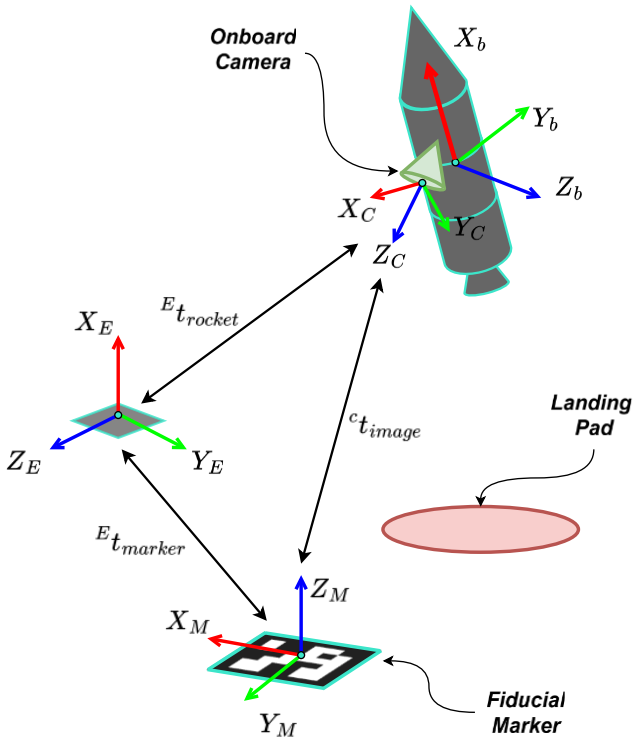


Figure 10: Camera Image frame relation to Flat-Earth frame

Once the marker's position is expressed in the rocket's Body-fixed frame, a sequential transformation is applied to convert it into the Flat-Earth frame using the following equation:

$$^{E}t_{rocket} = R_E^b \cdot {}^{b}t_{rocket} + {}^{E}t_{marker} \qquad Eq.\ 6$$

Where $^{E}t_{marker}$ is the known marker translation vector from the origin of the Flat-Earth reference frame and R_E^b is the rotation matrix from the rocket's Body-fixed frame to the Flat-Earth reference frame computed by using the information retrieved by the onboard gyroscope.

2.2 Modelling of Dynamics

Before creating the dynamic model of the rocket some assumptions are used to simplify the procedure. The following list contains assumptions which are used to derive the model:

- ***A. Rocket rigid body***: neglecting elastic behaviors. This is a valid assumption for control system design due to the smaller size of typical sounding rockets and the reduced impact of elastic behavior on overall dynamics.
- ***B. Axially symmetric mass allocation***: the principal inertia axes align with the body axes, the center of mass is on the longitudinal axis, and aerodynamic behavior is identical in both pitch and yaw planes.
- ***C. Optimal burn efficiency of propellants***: simultaneous depletion of both the solid fuel and liquid oxidizer which means that all energy released during combustion is converted into useful thrust.
- ***D. Ideal TVC system response:*** instantaneous response to control commands, which implies that the motors driving the TVCS actuators move directly to the desired position without any delay or overshoot.
- ***E. Ideal roll stabilization***: a roll control system is assumed to be present on the rocket, ensuring stable roll dynamics, this allows roll-induced effects, such as gyroscopic coupling or asymmetric aerodynamic forces, to be neglected in the dynamic model, simplifying the control system design and focusing on pitch and yaw plane dynamics.

2.2.1 Hybrid Rocket Propulsion

The amount of thrust generated by the propulsion model assumes ideal conditions [14]. Rocket thrust is directly influenced by the mass flow rate through the engine, the exhaust's exit velocity, and the pressure at the nozzle's exit:

$$^{b}F_t = \dot{m}_p V_e + (p_e - p_0)A_e \qquad \textit{Eq. 7}$$

where F_t represents thrust magnitude generated, $\dot{m}_p$ denotes the mass flow rate of the propellants, Ve signifies effective exhaust velocity, p_e stands for nozzle exit pressure, p_a represents the ambient atmospheric pressure, and A_e represents nozzle exit area.

Assuming that the ambient outside pressure of the nozzle, and the pressure generated by the exhaust is equal ($p_e = p_a$) Eq. 7 simplifies to:

$$^{b}F_t = \dot{m}_p V_e \qquad \textit{Eq. 8}$$

The ability to throttle the combustion process is given by the control valve found between the liquid oxidizer tank and the combustion chamber of the rocket. An oversimplified version of a HRE propulsion model is presented in Figure 11.

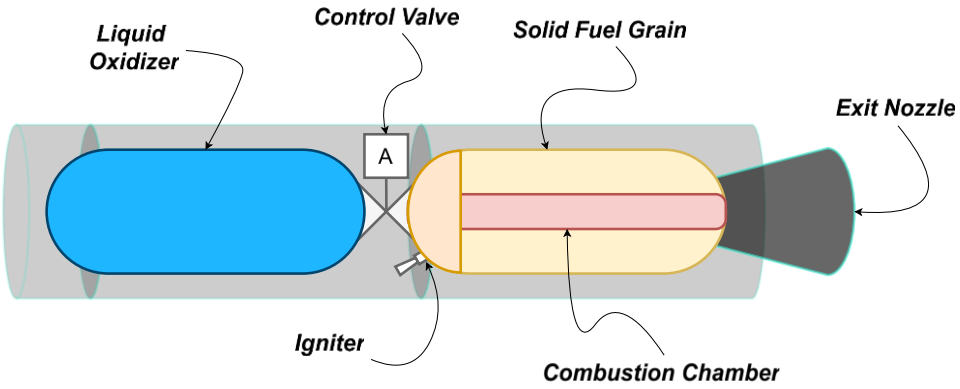


Figure 11: Hybrid rocket propulsion model

The thrust of a HRE is generated by combining a solid fuel grain with a liquid or gaseous oxidizer. The oxidizer is injected into the combustion chamber, where it reacts with the fuel grain's surface, causing it to burn and produce high-pressure gases. Table 5: Alpha HRE specifications presents the specifications of the HRE which Alpha will use to generate thrust.

Parameter	Value	Unit
Max. Mass Flow Rate	0.5	$\left[\frac{kg}{s}\right]$
Engine Exhaust Velocity	2000	$\left[\frac{m}{s}\right]$
Max. Thrust	1000	$[N]$
Oxidizer-to-fuel Ratio	2 : 1	N/A
Min. Throttling %	5	%
Max. Throttling %	100	%

Table 5: Alpha HRE specifications

In case of Alpha, the oxidizer is Nitrous oxide, while the solid fuel is Paraffin wax. As presented in Table 1 the propellant mass of the rocket is 5 $[kg]$, this means that the mass flow rate can be divided into two parts, namely the mass flow rate of the solid fuel and that of the oxidizer. The following equation describes this relationship while taking into account assumption ***C***:

$$\dot{m}_{p_s} = \frac{2}{3}\dot{m}_{max}$$
$$\dot{m}_{p_o} = \frac{1}{3}\dot{m}_{max} \qquad \text{Eq. 9}$$

where $\dot{m}_{p_s}$ is the mass flow rate of the solid fuel, while $\dot{m}_{p_o}$ is of the oxidizer and the $\dot{m}_{max}$ is the maximum flow rate of the HRE. Summing both of these terms and multiplying the result by the percentage of the control valve's opening δ_v gives the overall mass flow rate of the propellants:

$$\dot{m}_p = \left(\dot{m}_{o_s} + \dot{m}_{p_s}\right)\delta_v \qquad \text{Eq. 10}$$

2.2.2 Center of Gravity Shift and Inertia Decrease

As the rocket burns propellant its mass decreases, this results in the CoG of the rocket to shifts towards the remaining mass, which is typically at the bottom of the rocket due to the to the remaining propellants in the lower part of the tanks. This effect is illustrated Figure 12. As a result of this shift, the moment arm of any external force which acts about the CoG of the rocket, including those generated by the TVC system, is affected by this.

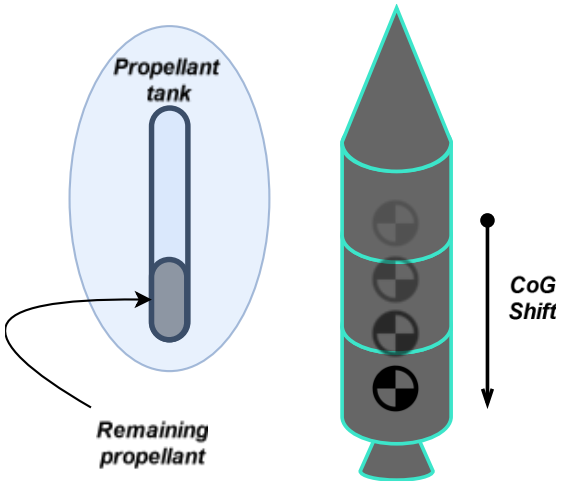


Figure 12: CoG shifts downwards to the bottom of the rocket

Because of the axially symmetric mass allocation assumption described in the beginning of this section, the dynamic moment arm length associated with moments created by the TVC can be calculated the following way:

$$l = l_{dry} + \frac{(l_{wet} - l_{dry}) \cdot m}{m_{wet}} \qquad \text{Eq. 11}$$

where l_{wet} and l_{dry} are the lengths of the moment arms when the rocket is fully fueled and completely depleted described in Table 2 and Table 3 respectively as the first value of the CoG vectors, m_{wet} is the wet mass of the rocket described in Table 1 and m is the actual mass of the rocket.

Another important aspect which needs to be taken into account while modelling the dynamics of the rocket is the change of the rocket's inertia matrix given by the following equation:

$$J = \frac{J_{wet} - J_{dry}}{m_{wet} - m_{dry}} \cdot \dot{m} \qquad \text{Eq. 12}$$

where J is the current inertia matrix of the rocket's body, while J_{wet} and J_{dry} compose the wet and dry inertia matrix of the rocket's body respectively. The values of these diagonal matrices are described in Table 2 and Table 3 as the principal axes of inertia of Alpha.

The principal axes of inertia of the rocket in the Body-fixed frame of the X, Y and Z axes are going to be referred as: I_x, I_y and I_z.

2.2.3 Thrust Vector Control

The TVC system enables the rocket to gimbal the direction of the rocket's thrust to generate torques which act about the CoG of the rocket, affecting its rotation in both pitch (up and down) and yaw (side to side) planes. Rotating the body enables the rocket to propel itself in the positive direction of the thrust vector, thus changing its direction during flight.

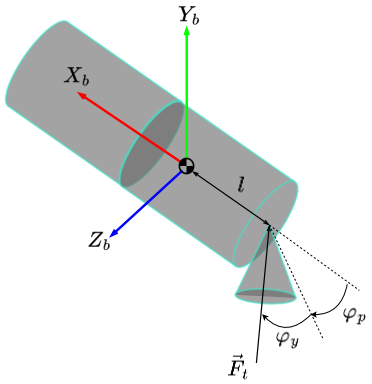


Figure 13: TVC force decomposition

Illustrated in Figure 13 is the thrust vector $\vec{F}_t$ generated by the HRE, where this force is broken down using angles φ_p (gimbal angle influencing pitch) and φ_y (gimbal angle influencing yaw). By gimbaling these angles the TVC produces the following forces in the Body-fixed frame of the rocket [26]:

$$ {}^{b}F_{TVC} = \begin{bmatrix} F_t c_{\varphi_p} c_{\varphi_y} \\ -F_t c_{\varphi_p} s_{\varphi_y} \\ -F_t s_{\varphi_p} \end{bmatrix} \qquad \textit{Eq. 13} $$

Naturally, these forces act about the CoG, thus generating moments in the Body-fixed frame described in Eq. 14.

$$ {}^{b}M_{TVC} = \begin{bmatrix} 0 \\ -F_t s_{\varphi_p} l \\ -F_t c_{\varphi_p} s_{\varphi_y} l \end{bmatrix} \qquad \textit{Eq. 14} $$

The forces and moments generated by the TVC system of the rocket in the Body-fixed frame of the X, Y and Z axes are going to be referred as: ${}^{b}F_{TVC_x}$, ${}^{b}F_{TVC_y}$, ${}^{b}F_{TVC_z}$ and ${}^{b}M_{TVC_x}$, ${}^{b}M_{TVC_y}$, ${}^{b}M_{TVC_z}$ respectively.

2.2.4 Earth's Gravity Model

For the modelling of the Earth's gravity the WGS-84 model [27] is used which assumes the Earth as a perfect sphere with a uniform mass distribution. This simplification allows for the assumption that the gravitational force acts towards the center of the Earth, aligning with the center of gravity of the rocket. Furthermore, since the gravitational force is considered to act through the center of gravity, it does not create any moments about this point.

As the body of the rocket rotates in flight, the force of gravity remains constant, acting through the rocket's CoG and always directed towards the center of the Earth (as shown in Figure 14). Thus, to express this force in the Body-fixed frame of the rocket, the following equation is used:

$$ {}^{b}F_g = R_E^b \cdot \begin{bmatrix} -mg \\ 0 \\ 0 \end{bmatrix} \qquad \textit{Eq. 15} $$

where m is the current body mass of the rocket, g is the varying gravitational acceleration on the surface of the Earth. This variable is computed as $g = g_0 \cdot \frac{R_e^2}{(R_e+h)^2}$, where g_0 is the constant gravitational acceleration, R_e is the mean Earth radius and h is the current altitude of the rocket.

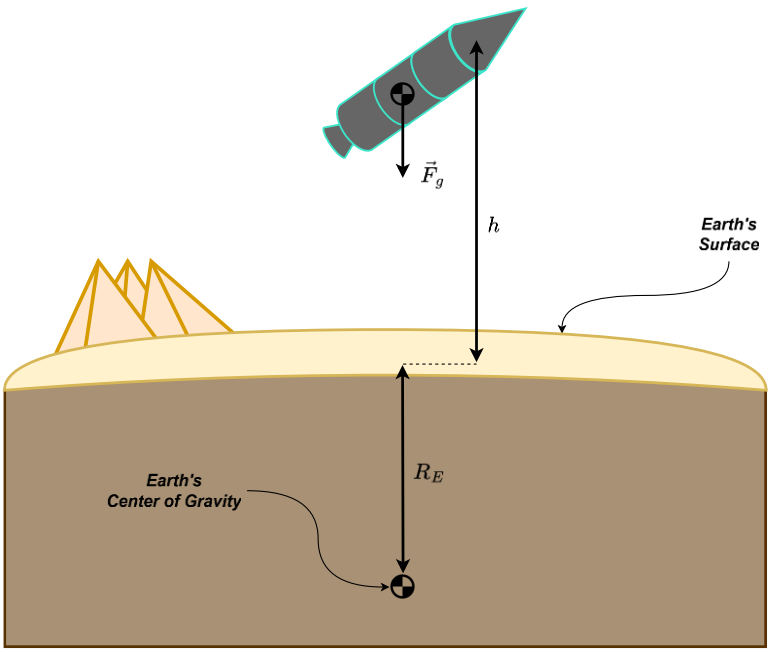


Figure 14: Gravity model

For the forces generated by the Earth's gravity in the Body-fixed frame of the X, Y and Z axes are going to be referred as: ${}^{b}F_{g_x}$, ${}^{b}F_{g_y}$ and ${}^{b}F_{g_z}$.

2.2.5 Aerodynamic Effects

A rocket experiences aerodynamic forces and moments [28] during flight due to its interaction with the surrounding atmosphere, as a result aerodynamic forces are generated in the Body-fixed frame and include axial, lateral, and normal components, represented as:

$$ {}^{b}F_A = \begin{bmatrix} -qC_A S \\ qC_Y S \\ -qC_N S \end{bmatrix} \qquad \textit{Eq. 16} $$

where C_A, C_Y and C_N are the axial, lateral, and normal aerodynamic force coefficients, respectively. The parameters q and S represent the dynamic pressure and a reference area, often corresponding to the fuselage's cross-sectional area.

These forces depend on the rocket's velocity relative to the atmosphere:

$$ \alpha = \tan^{-1}\frac{w_{rel}}{u_{rel}} \qquad \textit{Eq. 17} $$
$$ \beta = \sin^{-1}\frac{v_{rel}}{V_{re}} $$

where these are characterized by the angle of attack α and the sideslip angle β. These angles are defined by the components of the relative velocity vector w_{rel},v_{rel} and V_{rel} which also determines the coefficients $C_Y = C_Y\beta$ and $C_N = C_N\alpha$ through linear relations with the angles.

Similarly, aerodynamic moments, including rolling, pitching, and yawing moments, are represented in the Body-fixed frame as:

$$ {}^{b}M_A = \begin{bmatrix} qC_l Sd \\ qC_m Sd \\ qC_n Sd \end{bmatrix} \qquad \textit{Eq. 18} $$

where C_l, C_m and C_n are the aerodynamic rolling, pitching and yawing moment coefficients, and d is a reference length, often the fuselage diameter. These coefficients are influenced by the vehicle's static stability margin and damping characteristics, which are essential for controlling the rocket's attitude and maintaining stability during flight.

2.3 Modelling of Kinematics

The translational dynamics [28] of the rocket is given by applying Newton's second law on the forces acting on the rocket and including the cross-products created by the moments:

$$ {}^{b}\dot{v} = \frac{1}{m}\left({}^{B}F_{TVC} + {}^{B}F_{g}\right) - S({}^{b}\omega){}^{b}v \qquad \text{Eq. 19} $$

where $S(\cdot)$ is a skew-symmetric matrix, the ${}^{b}v = [u \quad v \quad w]^{T}$ is the velocity vector and ${}^{b}\omega = [p \quad q \quad r]^{T}$ is the angular velocity vector with the components of these vectors also illustrated in the Body-fixed frame in Figure 15.

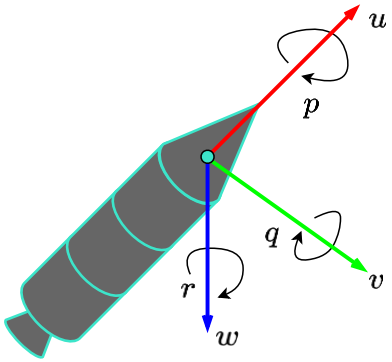


Figure 15: Decomposition of the translational and rotational vectors in the Body-fixed frame

As for the rotational dynamics of the rocket, Euler's equation for a rigid body produces the following equation:

$$ {}^{b}M_{TVC} = J{}^{b}\omega + {}^{b}\dot{\omega} \times J{}^{b}\omega \qquad \text{Eq. 20} $$

The rotation matrix R_{E}^{b} needs to be applied to ${}^{b}\dot{v}$ and ${}^{b}\dot{\omega}$ respectively, which will result in the acceleration ${}^{E}\dot{v}$ and the angular acceleration ${}^{E}\dot{\omega}$ of the rocket in the Flat-Earth reference frame. To find the position ${}^{E}x = [x \quad y \quad z]$ and the orientation $\alpha = [\phi \quad \theta \quad \psi]$ of the rocket in the Flat-Earth reference frame, the ${}^{E}\dot{v}$ and the ${}^{E}\dot{\omega}$ is numerically integrated two times using the discrete time step Δt.

2.4 VTVL System

Putting together the equations described in 2.2 and 2.3 the general 6DoF equations of motion [29] of the Alpha rocket can be constructed:

$$
\begin{aligned}
\dot{u} &= \frac{{}^{b}F_{A_x} + {}^{b}F_{TVC_x} + {}^{b}F_{g_x}}{m} - (qw - rv) \\
\dot{v} &= \frac{{}^{b}F_{A_y} + {}^{b}F_{TVC_y} + {}^{b}F_{g_y}}{m} - (ru - pw) \\
\dot{w} &= \frac{{}^{b}F_{A_z} + {}^{b}F_{TVC_z} + {}^{b}F_{g_z}}{m} - (pv - qu) \\
\dot{p} &= \frac{{}^{b}M_{A_x} + {}^{b}M_{TVC_x} - qr(I_z - I_y)}{I_x} \\
\dot{q} &= \frac{{}^{b}M_{A_y} + {}^{b}M_{TVC_y} - rp(I_x - I_z)}{I_y} \\
\dot{r} &= \frac{{}^{b}M_{A_z} + {}^{b}M_{TVC_z} - pq(I_y - I_x)}{I_z} \\
\dot{\phi} &= p + \left(qs_{\phi} + rc_{\phi}\right)\tan\theta \\
\dot{\theta} &= qc_{\phi} - rs_{\phi} \\
\dot{\psi} &= \frac{qs_{\phi} + rc_{\phi}}{c_{\theta}}
\end{aligned}
\qquad \text{Eq. 21}
$$

While the aerodynamic forces and moments have been described for the sake of completeness, their impact is significantly reduced in the context of the low-altitude/low-velocity hop test described in 1.4 and simulated for this study. Due to the relatively short duration and limited atmospheric interaction during the test, only the axial force ${}^{b}F_{A_x}$ acting on the hopper will be considered in the simulation. This simplification is justified as the lateral and normal aerodynamic forces, along with their associated moments, are negligible under the specific operating conditions of the test. Another important factor which will further simplify *Eq. 21* is due to the ***E*** assumption, which will result in the rolling angular velocity p, roll angle ϕ and their derivatives to become virtually 0.

It is also desirable to have the x_E, y_E and z_E position of the rocket in the Flat-Earth reference frame added into equations of motion, because it will further simplify and make the controller and sensor fusion design intelligible, as both of these require the position of the rocket. Thus, the motion of the Alpha VTVL will be defined by:

$$\begin{aligned}
\dot{x}_E &= u \\
\dot{y}_E &= v \\
\dot{z}_E &= w \\
\dot{u} &= \frac{{}^{b}F_{A_x} + {}^{b}F_{TVC_x} + {}^{b}F_{g_x}}{m} - (qw - rv) \\
\dot{v} &= \frac{{}^{b}F_{TVC_y} + {}^{b}F_{g_y}}{m} - ru \\
\dot{w} &= \frac{{}^{b}F_{TVC_z} + {}^{b}F_{g_z}}{m} + qu \\
\dot{q} &= \frac{{}^{b}M_{TVC_y}}{I_y} \\
\dot{r} &= \frac{{}^{b}M_{TVC_z}}{I_z} \\
\dot{\theta} &= qc_\phi - rs_\phi \\
\dot{\psi} &= \frac{qs_\phi + rc_\phi}{c_\theta}
\end{aligned} \qquad \textit{Eq. 22}$$

2.4.1 System Linearization

The states and the control inputs of the rocket can be contained in vectors of:

$$\begin{gathered}
\mathbf{x} = [x_E \quad y_E \quad z_E \quad u \quad v \quad w \quad q \quad r \quad \theta \quad \psi]^T \\
\mathbf{u} = [\mu_p \quad \mu_y \quad T]^T
\end{gathered} \qquad \textit{Eq. 23}$$

where **x** and **u** are the state and the control vectors representing the non-linear equation of motions of the hopper. However, for the design of a controller and sensor fusion algorithm, the system must be converted into a linear state-space representation. This is achieved by applying a first-order Taylor series expansion around a specific operating point, effectively linearizing the non-linear system. This is achieved by computing the Jacobian for both the state transition **A** and control-input **B** matrix.

$$\mathbf{J_A} = \frac{\delta \dot{\mathbf{x}}}{\delta \mathbf{x}} = \begin{bmatrix} \frac{\delta x_E}{\delta x_E} & \cdots & \frac{\delta x_E}{\delta \psi} \\ \vdots & \ddots & \vdots \\ \frac{\delta \psi}{\delta x_E} & \cdots & \frac{\delta \psi}{\delta \psi} \end{bmatrix} \quad \mathbf{J_B} = \frac{\delta \dot{\mathbf{x}}}{\delta \mathbf{u}} = \begin{bmatrix} \frac{\delta x_E}{\delta \mu_p} & \cdots & \frac{\delta x_E}{\delta T} \\ \vdots & \ddots & \vdots \\ \frac{\delta \psi}{\delta \mu_p} & \cdots & \frac{\delta \psi}{\delta T} \end{bmatrix} \qquad \textit{Eq. 24}$$

This will result in a 10×10 dimensional $\mathbf{J_A}(\cdot)$ and a 10×3 dimensional $\mathbf{J_B}(\cdot)$ matrices where each matrix component is the equation of motion for the derived Jacobian of the state derivatives $\dot{\mathbf{x}}$ with respect to the state vector **x** and control vector **u**.

These matrices are evaluated around a small disturbance from specific equilibrium points corresponding to the flight phases of the mission. For simplicity and proof-of-concept purposes, the system will be linearized around operating points $\mathbf{x_0}$, representing the end states of the three flight phases described in 1.4 with small disturbances to account for these expected effects which are captured in the **A** and **B** matrices.

The approach described earlier provides a straightforward and computationally efficient way to design

a gain-scheduled controller for the flight. However, it does come with drawbacks. Linearizing the system at only flight phase end points fails to capture the continuous variations in dynamics throughout the flight. This can lead to suboptimal control performance, particularly during transitions between phases or when the rocket operates far from the chosen linearization points. In a more robust and scalable design, the system should be linearized at multiple points along the trajectory, creating a time-varying linear model. This would allow for more accurate gain scheduling and better handling of dynamic changes during flight. Nonetheless, the current approach is sufficient to validate the concept and demonstrate the feasibility of the proposed GNC system.

2.4.2 State-space Representation

Substituting the vector of operating points $\mathbf{x_0}$ (presented in the Appendix A) of each flight phase of the hopper into $\mathbf{J_A}$ and $\mathbf{J_B}$ results in the state transition **A** and control-input **B** matrices. These matrices create a linearized version of the VTVL system around the operating points, thus the system can be written in state-space representation:

$$\begin{aligned}\dot{\mathbf{x}}(t)&=\mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)\\ \mathbf{y}(t)&=\mathbf{C}\mathbf{x}(t)\end{aligned} \qquad \textit{Eq. 25}$$

where **C** is the output identity matrix [30] of dimension 10×10, which essentially would make the VTVL system fully observable. This assumption is consistent with the described system as the sensor suite of the rocket presented in 1.1.3 provides every state measurement which is needed to make the system fully observable.

Considering that there are three flight phases in total, substituting $\mathbf{x_0}$ will result in a different **A** and **B** matrices for each flight phase. These matrices will be used to create the controllers for the Control module and Kalman filter for the Navigation module of the GNC subsystem, and considering the fact that the GNC designed in this project will be used onboard the flight computer of the real life rocket the state-space matrices are linearized with the Zero-order hold method [31] using highest sampling frequency from the sensor suite, that is 100 $[Hz]$.

2.5 Landing Legs Model

Lastly, an important aspect which needs to be modelled in order to simulate the landing of a rocket is the motion of the landing legs. As seen in Figure 3, initially the Landing Legs of Alpha are latched on the outside of the airframe and are released before landing. Illustrated in Figure 16 is one landing leg which is gravity-actuated, this means that once released the legs are pulled towards the bottom of the rocket by the force of gravity and are latched in place after a threshold angle is reached.

As such, a single landing leg can be modelled as a simple gravity pendulum [32] with the differential equation describing the motion of the landing leg as:

$$\ddot{\vartheta} + b\dot{\vartheta} + \frac{g}{L}\sin\vartheta \ = 0 \qquad \textit{Eq. 26}$$

where $\ddot{\vartheta}$ is the angular acceleration, $\dot{\vartheta}$ is the angular velocity, ϑ the angle with respect to the longitudinal axis of the rocket, b is the damping coefficient and L is the length of the landing leg.

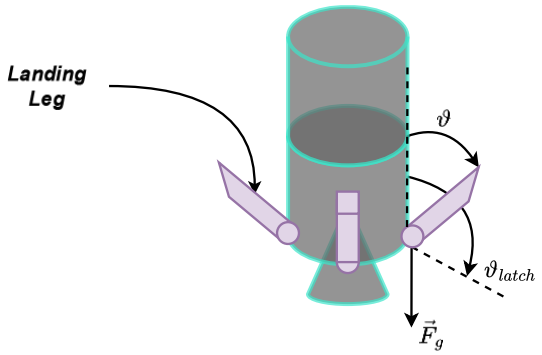


Figure 16: Gravity actuated landing legs

The described equation in *Eq. 26* can solved iteratively in the Simulink environment, thus it can be

broken down into the following set of equations which is solved at each k discrete time step and T_s sampling intervals:

$$\begin{cases} \dot{\vartheta}[k] = \dot{\vartheta}[k] - \left(\frac{g}{L}\sin\vartheta[k] + b\dot{\vartheta}[k]\right)T_s \\ \vartheta[k+1] = \vartheta[k] + \dot{\vartheta}[k-1]T_s \end{cases} \qquad \textit{Eq. 27}$$

In order to model the latching mechanism, and the inevitable bouncing of the landing leg just before correctly latching in place, a restitution coefficient is included into the model when $\vartheta[k+1] < \vartheta_{latch}$ and $\dot{\vartheta}[k+1] < 0$ condition is achieved:

$$\begin{cases} \vartheta[k+1] = \vartheta_{latch} \\ \dot{\vartheta}[k+1] = -e \cdot \dot{\vartheta}[k+1] \\ \dot{\vartheta}[k+1] < 0.1 \implies \dot{\vartheta}[k+1] = 0 \end{cases} \qquad \textit{Eq. 28}$$

3 Methods and Materials

To facilitate the development and evaluation of the GNC subsystem, the co-simulation platform will be constructed by coupling UE5 and MATLAB's Simulink. This co-simulation environment will leverage the Aerospace Blockset of Simulink, which provides a messaging interface between Simulink and UE5. This interface enables the exchange of data between the two environments, allowing for the simulation of the rocket's dynamics, sensor inputs, and control responses within the photorealistic virtual environment. This chapter describes the methods and system implemented within the two environments.

3.1 Co-Simulation Design

The co-simulation framework integrates MATLAB Simulink and UE5 to test the GNC subsystem in a realistic environment. Simulink models the rocket's dynamics, environmental conditions, sensor suite, and GNC algorithms, while UE5 provides a visual simulation for navigation and collision detection. Communication between the two environments is achieved via the UDP protocol, enabling the exchange of state information, commands, and visual data. Figure 17: Co-Simulation Framework illustrates the framework and its key components, which are detailed in the following subsections.

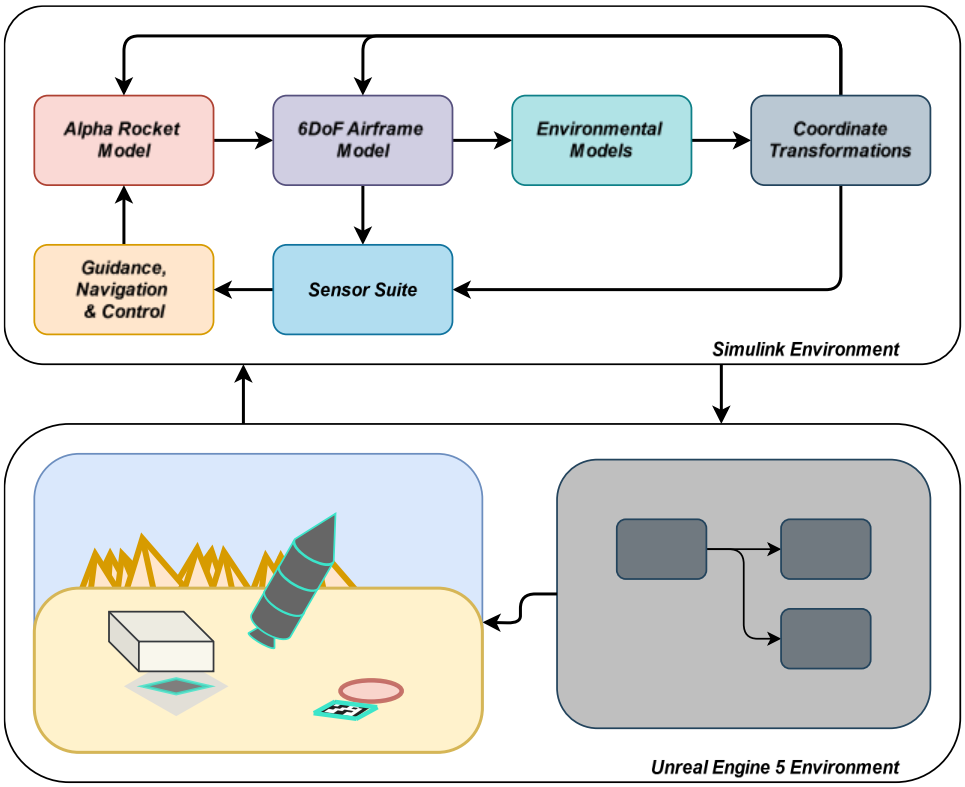


Figure 17: Co-Simulation Framework

3.1.1 Simulink Environment

The subsystems in the Simulink model (as shown in Figure 17: Co-Simulation Framework) and their functionalities are as follows:

- **Alpha Rocket Model**: represents the dynamics of the Alpha hopper, including rocket mass shift, a simplified hybrid propulsion model for thrust generation, forces and moments created by TVC and the aerodynamics of the airframe.
- **6-DoF Airframe Model**: simulates the 6DoF motion of the hopper, incorporating translational and rotational dynamics.
- **Environmental Models**: provides models of the environment which exert forces and moments on the hopper, such as a gravity model and an atmospheric model.

- **Coordinate Transformations**: computes the transformation of data between various reference frames (e.g., inertial, body, and navigation frames).
- **Sensor Suite**: The sensor suite simulates the outputs of onboard sensors, including an accelerometer, a gyroscope, a GNSS unit, and a camera.
- **Guidance, Navigation, and Control**: holds the model of the GNC, where each module.

3.1.1.1 Sensor Suite Design

To ensure the co-simulation closely resembles real-world conditions, the sensor models in Simulink are designed to replicate the performance of the actual sensors that will be onboard the rocket. Key specifications such as accuracy, resolution, and noise levels are modeled to reflect the behavior of these sensors under realistic operating conditions. This level of detail is essential to produce reliable data for testing the GNC system and validating its performance in a simulated environment. To this end, the sensors described in 1.1.3 are modelled using Simulink sensor blocks from the Aerospace Blockset and values provided in the datasheets [16], [19].

3.1.1.1.1 Accelerometer Modelling

To model the Three-axis Accelerometer [33] the following parameters need to be entered into the Simulink Block:

- Natural frequency $\left[\frac{Rad}{s}\right]$
- Damping ratio
- Scale factors and cross coupling
- Measurement bias $\left[\frac{m}{s^2}\right]$
- Update rate $[s]$
- Noise power $\left[\left(\frac{m}{s^2}\right)^2\right]$
- Lower and upper output limits $\left[\frac{m}{s^2}\right]$

The natural frequency of the three-axis accelerometer can be approximated based on its bandwidth. Since bandwidth is the frequency at which the response is reduced by 3 dB, the natural frequency can be estimated using the relationship:

$$\omega_0 = 2\pi \cdot B \qquad \textit{Eq. 29}$$

where ω_0 is the natural frequency and B is the bandwidth, and using the lowest bandwidth of the accelerometer found in Table 4, the natural frequency will be around 1646.2 $\left[\frac{Rad}{s}\right]$.

Typically, the damping ratio is around 0.707 for accelerometers to ensure a good trade-off between speed and stability. However, the damping ratio is not provided directly, so this is an assumed typical value [34].

The scale factor variation for the accelerometer is 0.5%, but since cross-coupling is not explicitly provided in the MTi-7 datasheet, a reasonable assumption for a well-calibrated sensor would be:

$$\begin{pmatrix} 1.005 & 0.001 & 0.001 \\ 0.001 & 1.005 & 0.001 \\ 0.001 & 0.001 & 1.005 \end{pmatrix}$$

The in-run bias stability for the accelerometer is $2.943 \cdot 10^{-5}$ $\left[\frac{m}{s^2}\right]$. Assuming this is uniform across all axes, the bias vector is $[2.943 \cdot 10^{-5} \quad 2.943 \cdot 10^{-5} \quad 2.943 \cdot 10^{-5}]$ $\left[\frac{m}{s^2}\right]$.

The MTi-7 datasheet specifies a maximum output data rate of 100 $[Hz]$ or $\frac{1}{100}$ $[s]$ for the

accelerometer.

Noise power is related to the sensor's noise density, thus for the accelerometer is $120\left[\frac{\mu g}{\sqrt{Hz}}\right]$ or $1.177 \cdot 10^{-3}\left[\frac{\frac{m}{s^2}}{\sqrt{Hz}}\right]$. The noise power over the sensor's bandwidth can be estimated as:

$$N_p = N_0 \cdot B \qquad Eq.\ 30$$

where N_p is the resultant noise power and the N_0 is the accelerometer's noise density? As the bandwidth for the Z axis differs from the X and Y axes of the noise power vector would be approximately $[4.48 \cdot 10^{-4} \quad 4.48 \cdot 10^{-4} \quad 3.62 \cdot 10^{-4}]\left[\left(\frac{m}{s^2}\right)^2\right]$.

The accelerometer's range is $\pm 16[g]$. Converting this to $\left[\frac{m}{s^2}\right]$ is $\pm 16[g] = \pm 16 \cdot 9.81 = \pm 156.96\left[\frac{m}{s^2}\right]$. Therefore, the output limits vector is $[-156.96 \quad -156.96 \quad -156.96 \quad 156.96 \quad 156.96 \quad 156.96]\left[\frac{m}{s^2}\right]$.

The three acceleration values of measured by the Three-axis Accelerometer in the Body-fixed frame of the X, Y and Z axes are going to be referred as: A_{meas_x}, A_{meas_y} and A_{meas_z}.

3.1.1.1.2 Gyroscope Modelling

To model the Three-axis Gyroscope [35] the following parameters need to be entered into the Simulink Block:

- Natural frequency $\left[\frac{Rad}{s}\right]$
- Damping ratio
- Scale factors and cross coupling
- Measurement bias $\left[\frac{Rad}{s}\right]$
- G-sensitive bias $\left[\frac{Rad \cdot g}{s}\right]$
- Update rate $[s]$
- Noise power $\left[\left(\frac{Rad}{s}\right)^2\right]$
- Lower and upper output limits $\left[\frac{Rad}{s}\right]$

Similarly, to the accelerometer, the gyroscope's natural frequency can be estimated using it's bandwidth of $255[Hz]$ and *Eq. 29*, which would result in a frequency of 1602.2 $\left[\frac{Rad}{s}\right]$.

The datasheet does not specify a damping ratio, thus the same value of 0.707 will be used for the gyroscope as well.

The scale factor variation for the gyroscope is the same as for the accelerometer: 0.5%. Thus, as above with accelerometer, the cross-coupling terms can be modelled similarly:

$$\begin{pmatrix} 1.005 & 0.001 & 0.001 \\ 0.001 & 1.005 & 0.001 \\ 0.001 & 0.001 & 1.005 \end{pmatrix}$$

The in-run bias stability for the gyroscope is $10\left[\frac{^\circ}{h}\right]$, converting to radians per second $10\left[\frac{^\circ}{h}\right] = \frac{10}{3600} \cdot \frac{\pi}{180} \approx 4.8481 \cdot 10^{-5}\left[\frac{Rad}{s}\right]$. Assuming uniform bias across the three axes of the gyroscope, the resultant bias vector is $[4.8481 \cdot 10^{-5} \quad 4.8481 \cdot 10^{-5} \quad 4.8481 \cdot 10^{-5}]\left[\frac{Rad}{s}\right]$.

The g-sensitivity of the gyroscope is $0.001\left[\frac{^\circ g}{s}\right]$. Converting this to radians per second per meter per second squared $0.001\left[\frac{^\circ g}{s}\right] = 0.001 \cdot \frac{\pi}{180} \approx 1.7453 \cdot 10^{-5}\left[\frac{Rad \cdot g}{s}\right]$.Assuming equal sensitivity across the axes of the gyroscope, the resultant g-sensitive bias vector would be

$[1.7453 \cdot 10^{-5} \quad 1.7453 \cdot 10^{-5} \quad 1.7453 \cdot 10^{-5}] \left[\frac{Rad \cdot g}{s}\right]$.

The gyroscope in the MTi-7 has the same update rate as the accelerometer, with a maximum output data rate of 100 $[Hz]$ or $\frac{1}{100}$ $[s]$.

The noise density for the gyroscope is 0.007 $\left[\frac{^\circ\sqrt{Hz}}{s}\right]$ which is $1.2217 \cdot 10^{-4}$ $\left[\frac{Rad \cdot \sqrt{Hz}}{s}\right]$. Using *Eq. 30* and a bandwidth of $255[Hz]$, assuming uniform noise power value across all three axes of the gyroscope, the noise power vector is $[3.80 \cdot 10^{-6} \quad 3.80 \cdot 10^{-6} \quad 3.62 \cdot 10^{-6}] \left[\left(\frac{Rad}{s}\right)^2\right]$.

The standard full range of the gyroscope is $\pm 2000 \left[\frac{^\circ}{s}\right]$ or $\pm 34.9066 \left[\frac{Rad}{s}\right]$. Therefore, the output limits vector is $[-34.9066 \quad -34.9066 \quad -34.9066 \quad 34.9066 \quad 34.9066 \quad 34.9066] \left[\frac{Rad}{s}\right]$.

The three angular velocity values of measured by the Three-axis Gyroscope in the Body-fixed frame of the X, Y and Z axes are going to be referred as: ω_{meas_x}, ω_{meas_y} and ω_{meas_z}.

3.1.1.1.3 GNSS Modelling

To model the GNSS [36] the following parameters need to be entered into the Simulink Block:

- Horizontal position accuracy
- Vertical position accuracy
- Velocity accuracy
- Decay factor

The horizontal position accuracy for the MTi-7 GNSS, using Satellite-Based Augmentation System (SBAS), is 1.0 $[m]$ (1σ STD).

The vertical position accuracy, using SBAS and a barometer, is 2.0 $[m]$ (1σ STD).

The velocity accuracy is 0.05 $\left[\frac{m}{s}\right]$ (1σ RMS).

The decay factor is a value between 0.0 (which generate purely white noise) and 1.0 (which would generate a random walk sequence). To generate sensor data which would exhibit behavior received by the sensor in the real-world, a decay factor between $0.1 - 0.3$ can be chosen. This range captures a mix of random errors and slow drift, reflecting the behavior of most GNSS systems under real-world conditions [37].

The vector containing the *latitude*, *longitude* and *altitude* values measured by the GNSS is going to be referred as $LLA_{meas} = (\phi_{LLA_{meas}}, \lambda_{LLA_{meas}}, h_{LLA_{meas}})$.

3.1.1.1.4 Camera Modelling

To model the Camera [38] the following parameters need to be entered into the Simulink Block:

- Image size
- Optical center
- Focal length $[px]$

The image size is $720\,x\,1280$ pixels, thus the optical center for the camera will be 640 and 360 respectively.

The focal length needs to be given in pixels and is calculated as:

$$f_x = l_F \cdot s_x$$
$$f_y = l_F \cdot s_y \qquad \textit{Eq. 31}$$

Where f_x, f_y are the focal lengths in the X and Y direction respectively of the camera in pixels, l_F is the focal length of the Camera Module 3 Wide which is 2.75 $[mm]$, while s_x, s_y are pixel density in the X and Y direction respectively. The Camera Module 3 Wide uses a Sony IMX708 image sensor with a

resolution of $4608\,x\,2592$ and a sensor size of $7.2\,x\,5.4\,[mm^2]$. Thus, the pixel density can be calculated as:

$$s_x = \frac{4608}{7.2} \approx 640$$
$$s_y = \frac{2592}{5.4} \approx 480 \qquad \textit{Eq. 32}$$

Computing the values for the focal length using *Eq. 31* results in the values of $[1760 \quad 1320][px]$.

3.1.2 Unreal Engine 5 Environment

This environment serves as the high-fidelity visualization and interaction platform for the co-simulation framework. It provides a photorealistic simulation of the hopper's operating environment, including the terrain, fiducial markers, and dynamic elements critical for visual navigation. The environment receives relevant data from the Simulink model via UDP, including the rocket's position, orientation, TVC pitch and yaw angles, and commands such as landing leg deployment and engine throttle exhaust. The UE5 environment is programmed using *Blueprints*, a visual scripting language that enables dynamic control of the rocket and its components in the virtual environment. This programming ensures that the rocket's pose and movements match the data received from the Simulink environment in real time, accurately representing the hopper's dynamics within the constructed scene. In return, the UE5 environment provides two critical outputs to the Simulink model. First, it streams images from the onboard rocket camera, simulating real-time visual data for use in state estimation. Second, it provides collision information during the landing phase, detecting interactions between the rocket and the terrain or landing pad. This bidirectional communication allows for a seamless integration of visual navigation and collision detection into the co-simulation framework, enabling comprehensive testing and evaluation of the Guidance, Navigation, and Control (GNC) system.

3.1.2.1 Launch site and landing site

For the UE5 simulated environment the Lucerne Valley ROC (Rocketry Organization of California) launch site was chosen (as shown in Figure 18). Cesium is used to enhance the photorealistic aspect of the UE5 environment [39]. The launch site is at an altitude of 870 meters above sea level and it is known for its wide-open space, making it ideal for rocketry activities.

Figure 18: Aerial view of the ROC launch site inside the UE5 environment

Inside the simulated environment, a launch and landing site is created which is not found in the real world. The launch pad is located at (34.495624°, −116.95782°, 868), these coordinates are used as reference coordinates $(\phi_{LLA_{ref}}, \lambda_{LLA_{ref}}, h_{LLA_{ref}})$ in the GCS reference frame to transform the LLA

coordinates received from the GNSS to Flat-Earth coordinates. The landing pad (34.495824°, −116.95782°, 868) is positioned 20 meters from the launch pad. Patches of rocks and fauna, which are usually found in these types of arid areas, are added to further enhance the photorealistic aspect of the project, this can be seen in Figure 19.

Figure 19: Constructed launch and landing site inside the UE5 Editor

The ArUco markers used for visual navigation measures $1 \times 1\,[m^2]$ and $0.5 \times 0.5\,[m^2]$ and are positioned at coordinates (0.0, -2.75, 20) and (0.0, -1.15, 20) relative to the launch site in the Flat-Earth reference frame, placing it 2.75 and 1.15 meters to the west of the landing pad, with the landing site being a concrete circle slab with a diameter of 10 meters.

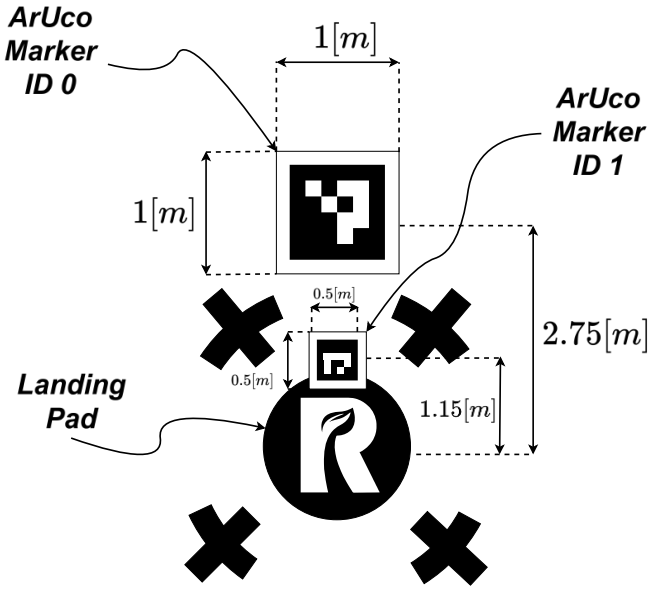


Figure 20: Landing Site ArUco Maker Size and Placement

To facilitate real-world mission replication by RISE, the marker and other landing site features are rendered in UE5 with attention to photo-realism. The marker is painted onto a concrete tiled surface using the black matte spray paint material available in the UE5 library, ensuring high visual fidelity under simulated lighting conditions. The selected markers are from the ArUco 4x4 dictionary with an

ID of 0 and 1. This specific choice ensures that the detection algorithm is as efficient as possible by minimizing the search space for the marker ID during processing. The detailed configuration and placement aim to replicate conditions for precise and robust visual navigation during the rocket's landing phase, both in simulation and potential real-world trials.

3.1.2.2 Rocket Engine Exhaust, Smoke and Dust

The exhaust plume of the RISE Green Phoenix engine exhibits a distinct orange-purplish hue. This characteristic coloring results from the combustion of the paraffin wax and nitrous oxide propellants, which achieve high efficiency and clean burning [40]. The bright orange core of the plume indicates a well-optimized temperature and combustion chamber pressure, while the purplish edges highlight the presence of certain ionized components within the exhaust. These colors were replicated in the co-simulation by studying similar characteristics observed in real-world hybrid rocket engines, such as those showcased in the image from Figure 21.

Figure 21: Paraffin Powered Rocket Engine developed by NASA at the Ames Research Center [41]

In addition to its vivid coloration, the smoke production from the Green Phoenix engine remains relatively low. This is a direct consequence of the clean-burning nature of the hybrid rocket design, where the oxidizer (nitrous oxide) and fuel (paraffin wax) react efficiently to produce minimal particulate matter. The combination of high combustion efficiency and reduced soot generation leads to a visually distinct, mostly transparent exhaust plume, this contrasts with traditional solid rocket engines, which often produce substantial smoke trails. This recreation can be seen in the UE5 environment showcased in Figure 22.

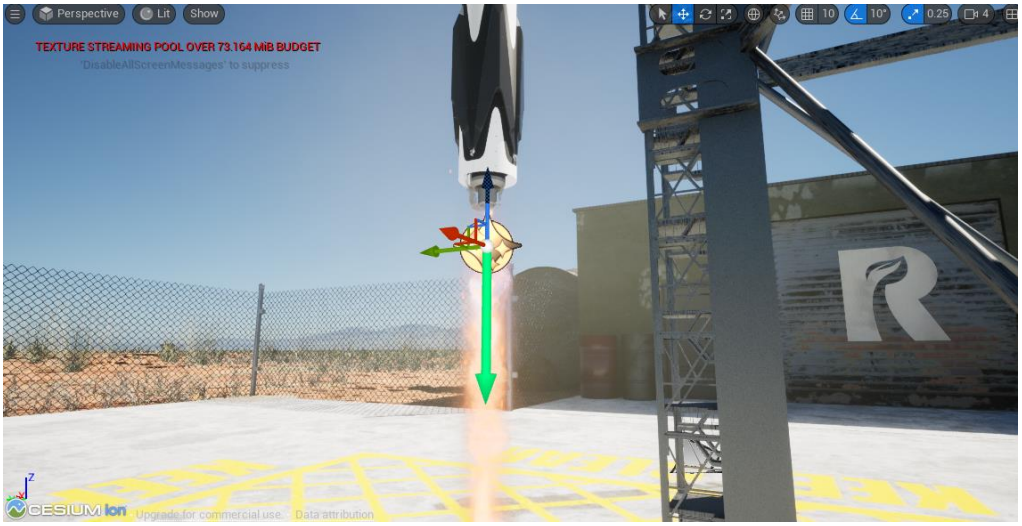


Figure 22: RISE's Paraffin Powered Green Phoenix Engine in the UE5 Editor

Smoke and dust generation during engine operation is primarily influenced by the interaction of the exhaust jet with the ground surface at liftoff or landing. Despite this, the relatively small scale of the Green Phoenix engine and the low particulate emissions ensure that dust clouds remain minimal compared to larger or less efficient propulsion systems. The co-simulation accounts for this to create a realistic visual representation of the rocket engine's behavior during hop tests.

3.2 GNC Subsystem Design

The GNC subsystem is responsible for maintaining stability, guiding the vehicle along a predefined trajectory, and ensuring precise landing of the VTVL hopper. The GNC subsystem comprises three primary modules: Guidance, Navigation, and Control. Presented in Figure 23 is the GNC subsystem along with subsystems witch which it interacts.

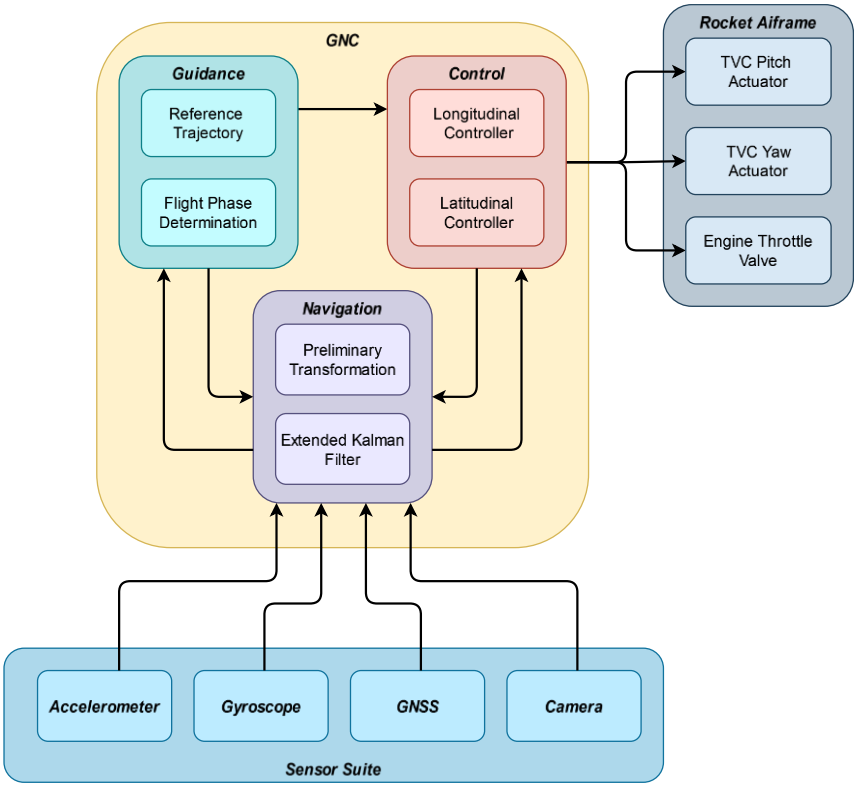


Figure 23: GNC subsystem and it's interaction with other relevant subsystems

The Guidance module holds the predefined reference trajectory and determines the current flight phase of the vehicle from the estimated states received from the Navigation module. The determined flight phase is then transmitted to the Control module to set the desired states and gains for the controllers (essentially acting as a gain scheduler) and to the Navigation module to correctly configure the matrices of the EKF for the given flight phase.

The Navigation module employs a Preliminary Transformation unit and an EKF to determine the vehicle's state. This module receives sensors data which includes readings from an accelerometer, gyroscope, GNSS module and a camera. The Preliminary Transformation unit computes the relative velocity and Euler angles of the rocket using a Complementary Filter and transforms the LLA coordinates into Flat-Earth Frame coordinates. The EKF fuses the data from the Preliminary Transformation unit and the measurements from the sensors to estimate position, velocity, angular velocity and orientation.

Finally, the Control module compromises of two controllers – a Longitudinal and a Latitudinal controller – to adjust the vehicle's orientation and thrust. These controllers receive input from both the Guidance and the Navigation modules. Both Longitudinal and Latitudinal controllers use a gain scheduled Linear Quadratic Regulator (LQR) with Integral Action design to compute the pitch, yaw and throttle percentage commands. These commands are then used to actuate the TVC's pitch and yaw actuators along with the engine's throttle valve.

3.2.1 Guidance Module Design

As mentioned above, this module of the GNC subsystem is responsible for determining the desired flight phase using the estimated states of the rocket and the predetermined flight trajectory which retains in its memory. Essentially, the Guidance module acts as a supervisory controller that ensures that the rocket follows the planned trajectory and achieves the mission goals with precision and accuracy.

3.2.1.1 Flight Phase Determination

The Flight Phase Determination is a simple algorithm which utilizes the rocket's position to ascertain its current flight phase. This determination is achieved through a two-step process. Depending on the flight phase, the position can be either the height of the rocket above the surface of the Earth or the 2D Euclidian distance from the launch site.

First, a moving average (MA) filter with a sliding window length of n_{MA} is applied to the position readings, smoothing out fluctuations and providing a more stable altitude estimate. Inside the Simulink environment this is implemented using a FIFO data structure and is computed as:

$$A = \frac{1}{n_{MA}} \sum_{i=1}^{n_{MA}} p_i = \frac{p_1 + \ldots + p_7}{n_{MA}} \qquad \textit{Eq. 33}$$

where p_i is the current position reading, p_7 is the newest position reading and p_1 is the oldest position reading.

Second, a moving standard deviation (MSTD) with a sliding window length of n_{MSTD} is calculated from the filtered height data. This MSTD serves as a stability indicator, where a low value signifies stable hovering, thereby enabling the transition to subsequent flight phases based on predefined thresholds and hysteresis checks.

$$P = \sqrt{\frac{1}{n_{MSTD} - 1} \sum_{i=1}^{n_{MSTD}} |A_i - \mu|^2} \qquad \textit{Eq. 34}$$

where μ is the mean of the current sliding window implemented as a FIFO made up of the MA values $[A_1 \quad A_2 \quad \cdots \quad A_{n_{MSTD}}]^T$ and where A_i is one of the values of the vector.

3.2.1.2 Terminal Landing Phase Command

The Apollo 11 lunar landing offers a valuable lesson for developing a Guidance module for a VTVL. During the final phase of the landing, Neil Armstrong could not find a suitable landing spot for the Lunar Excursion Module (LEM) which he operated, and as the LEM was hovering over the surface of the Moon its fuel levels were dwindling fast [47]. As everyone in Houston feared the worst, Gerard Elverum - the LEM descent engine's chief engineer, urgently advised Armstrong from the back of the mission control room "Find the damn rock and set it down" [46] . Prompted by the command Armstrong has done exactly that and the landed the LEM on the surface of the Moon.

Analog to this situation the rocket can find itself stuck in a similar problem. In the descent phase the ArUco marker of the landing site can disappear from the camera image, which will ultimately happen as the rocket descents further down and/or the marker is not in the camera's line of sight. Because of this, the Guidance module dynamically switches to an IMU-based positioning strategy. This approach involves integrating filtered velocity estimates over time and incorporating them into the Kalman filter for state estimation. When this happens the Control module will try to fly the rocket to the desired landing position, but if the descent velocity or the position error is too large, the Control module can either make the rocket hover until it runs out of fuel or fly to the desired landing position and tip the rocket over. To avoid such situations, the Guidance module prompts the Control module to urgently land, which is a valid redundancy objective as the rocket would already be above the landing site, making it a good area to land.

3.2.2 Navigation Module Design

To ensure the co-simulation accurately evaluates the GNC system's performance, the Navigation module is designed to replicate the state estimation process which will be used in the actual rocket. It

integrates sensor data, while accounting for real-world factors such as measurement noise and system dynamics.

3.2.2.1 Preliminary Transformation Unit

The Preliminary Transformation unit is responsible for pre-processing raw sensor data from the accelerometer, gyroscope, and GNSS to produce filtered and transformed outputs necessary for state estimation. Its task is to compute a velocity estimation from accelerometer data, Euler angle estimation through a Complementary Filter [44], and coordinate transformation for GNSS data.

The accelerometer measurements (A_{meas_x}, A_{meas_y} and A_{meas_z}) undergo low-pass filtering using Butterworth filter to remove high-frequency noise and vibrations that can corrupt the signal:

$$A_{filt}[k] = b_0 A_{filt}[k] + b_1 A_{filt}[k-1] + \cdots + a_1 A_{filt}[k-1] + \cdots \quad \textit{Eq. 35}$$

where b_0, b_1, ... and a_1, ... are the filter cofficients determined by the Butterworth filter design [45] for a 8th order with a cutoff frequency of 60 $[Hz]$ and a sampling frequency specified in Table 4. The cutoff frequency of 60 $[Hz]$ for the MTi-7's accelerometer was found through trial and error.

The filtered acceleration data is then integrated to obtain velocity estimates:

$$V_{filt}[k] = V_{filt}[k-1] + T_s A_{filt}[k] \quad \textit{Eq. 36}$$

where $V_{filt}[k-1]$ is the velocity from the previous time step. This integration assumes an initial velocity condition and is sensitive to bias and drift in the accelerometer data.

As stated in 3.2.1.2 when the ArUco markers are not visible in the camera frame and thus cannot estimate the rocket's position, the Navigation module will use the IMU-based positioning strategy. This is done by integrating the already filtered velocity V_{filt} vector to compute the position filtered position vector:

$$X_{filt}[k] = X_{filt}[k-1] + T_s V_{filt}[k] \quad \textit{Eq. 37}$$

where $X_{filt}[k-1]$ is the position from the previous time step, which in case of the switch from the ArUco-based positioning to the IMU-based positioning is the last known position of the rocket computed by the ArUco pose estimation algorithm.

Using a Complementary Filter, the system fuses these two sources of data to estimate the Euler angles (ϕ, θ, ψ), prioritizing gyroscope data for short-term changes and accelerometer data for long-term stability. The accelerometer provides orientation based on the gravity vector:

$$\begin{aligned} \phi_{acc}[k] &= \tan^{-1}\left(\frac{A_{meas_y}[k]}{A_{meas_z}[k]}\right) \\ \theta_{acc}[k] &= \tan^{-1}\left(-\frac{A_{meas_x}[k]}{\sqrt{A_{meas_y}[k]^2 + A_{meas_z}[k]^2}}\right) \end{aligned} \quad \textit{Eq. 38}$$

The gyroscope data provides angular rates which are integrated to estimate changes in orientation:

$$\begin{aligned} \phi_{gyro}[k] &= \phi_{gyro}[k-1] + T_s \omega_{meas_x}[k] \\ \theta_{gyro}[k] &= \theta_{gyro}[k-1] + T_s \omega_{meas_y}[k] \\ \psi_{gyro}[k] &= \psi_{gyro}[k-1] + T_s \omega_{meas_z}[k] \end{aligned} \quad \textit{Eq. 39}$$

The Complementary Filter combines these estimates to reduce drift and noise:

$$\begin{aligned} \phi_{filt}[k] &= \alpha \phi_{gyro}[k] + (1-\alpha)\phi_{acc}[k] \\ \theta_{filt}[k] &= \alpha \theta_{gyro}[k-1] + (1-\alpha)\theta_{acc}[k] \end{aligned} \quad \textit{Eq. 40}$$

As a result of the sensor suit not considering the usage of a magnetometer, the equivalent of $\psi_{filt}[k]$ is $\psi_{gyro}[k]$ as yaw estimation relies purely on gyroscope integration.

Finally, the GNSS measurements LLA_{meas} is converted to a Flat-Earth reference coordinate frame position $X_{E_{meas}}$ to localize the rocket using the transformation described in 2.1.3.

$$X_{E_{meas}} = \begin{bmatrix} x_{E_{meas}} \\ y_{E_{meas}} \\ z_{E_{meas}} \end{bmatrix} = T_E^{LLA} \cdot \begin{bmatrix} \phi_{LLA_{meas}} - \phi_{LLA_{ref}} \\ \lambda_{LLA_{meas}} - \lambda_{LLA_{ref}} \\ h_{LLA_{meas}} - h_{LLA_{ref}} \end{bmatrix} \quad \textit{Eq. 41}$$

3.2.2.2 Extended Kalman Filter

The Kalman filter [46] estimates the rocket's states described in *Eq. 23*, but to counteract the inevitable drift caused by the accumulation of sensor noise and biases, the state vector for the Kalman filter is augmented additionally by accelerometer (b_u, b_v, b_w) and gyroscope (b_q, b_r) biases to eliminate drift in the estimated velocities and Euler angles, resulting in a total of 15 states denoted by the augmented state vector:

$$\boldsymbol{\chi} = [\mathbf{x}^T \quad b_u \quad b_v \quad b_w \quad b_q \quad b_r]^T \quad \textit{Eq. 42}$$

This results in the need to redefine the state transition **A**, the control-input **B** for the Kalman filter to account for these added states, as the Jacobian for both **A** and **B** matrices need to be recomputed. Thus, to differentiate between these matrices, for the Kalman filter the state transition matrix and the control-input matrices will be referred as **F** and **P** respectively. The Jacobian for these matrices will be computed as:

$$\mathbf{J_F} = \frac{\delta \dot{\boldsymbol{\chi}}}{\delta \boldsymbol{\chi}} = \begin{bmatrix} \frac{\delta x}{\delta x} & \cdots & \frac{\delta x}{\delta b_r} \\ \vdots & \ddots & \vdots \\ \frac{\delta b_r}{\delta x} & \cdots & \frac{\delta b_r}{\delta b_r} \end{bmatrix} \quad \mathbf{J_P} = \frac{\delta \dot{\boldsymbol{\chi}}}{\delta \mathbf{u}} = \begin{bmatrix} \frac{\delta x}{\delta \mu_p} & \cdots & \frac{\delta x}{\delta T} \\ \vdots & \ddots & \vdots \\ \frac{\delta b_r}{\delta \mu_p} & \cdots & \frac{\delta b_r}{\delta T} \end{bmatrix} \quad \textit{Eq. 43}$$

This will result in a 15×15 dimensional $\mathbf{J_F}(\cdot)$ and a 15×3 dimensional $\mathbf{J_P}(\cdot)$ matrices where each matrix component is the equation of motion for the derived Jacobian of the state derivatives $\dot{\boldsymbol{\chi}}$ with respect to the augmented state vector $\boldsymbol{\chi}$ and control vector **u**. Substituting the operating points $\mathbf{x_0}$ of each flight phase described in Appendix A will result in three **F** and **P** matrices which are used throughout the hop, effectively creating an EKF.

Just as with **A** and **B** matrices, the output matrix also needs to be redefined, thus the measurement (ouput) matrix for the Kalman filter will be denoted as **H**. The bias states will be slowly changing over the course of the flight as these values will be recomputed into better estimates by the Kalman filter at each k time step. In order to remove the drift, the bias terms are subtracted from the accelerometer and gyroscope readings, by updating measurement equation. To achieve this, the **H** matrix is defined as:

$$\mathbf{H} = \begin{bmatrix} \mathbf{C}_{10} \begin{vmatrix} O_{3,5} \\ -I_5 \\ O_{2,5} \end{vmatrix} \end{bmatrix} \quad \textit{Eq. 44}$$

where $\mathbf{C}_{10}$ is the 10×10 output matrix defined in *Eq. 25*, $O_{3,5}$ and $O_{2,5}$ are zero matrices [47] of dimension 3×5 and 2×5 respectively, and I_5 is an 5×5 identity matrix while the negative sign indicates that the biases subtract from the raw measurements. The resulting discrete Kalman filter model [48] can be described by:

$$\begin{aligned} \boldsymbol{\chi}[k] &= \mathbf{F}\boldsymbol{\chi}[k-1] + \mathbf{P}\mathbf{u}[k] + \mathbf{w}[k] \\ \mathbf{z}[k] &= \mathbf{H}\boldsymbol{\chi}[k] + \mathbf{v}[k] \end{aligned} \quad \textit{Eq. 45}$$

where **w** is the process noise of the system assumed to have a zero mean multivariate normal distribution [49], **v** is the additive sensor noise assumed to be zero mean Gaussian white noise [50], while **z** is the measurement of the true state the $\boldsymbol{\chi}$.

In Flight Phases I and II, the Kalman filter relies on measurements from the GNSS, accelerometer, and gyroscope sensors to estimate the state of the rocket, but during Flight Phase III, the primary navigation sensor switches from the GNSS to the ArUco marker-based Camera position estimation, which provides higher accuracy for position measurements near the landing site. The covariance matrices for process noise Q, sensor R, and initial state error P_0 are configured to reflect the characteristics of these sensors and the dynamics of the system, these matrices are described in detail in Appendix B.

- P_0: Defines the initial uncertainty for the state vector, with smaller variances for position and velocity and higher variances for angular velocity and Euler angles. Biases are initialized with low uncertainty to reflect their assumed stability at the start of the flight.
- Q: Assumes small process noise for position, velocity, angular velocity, and Euler angles to model system dynamics accurately.
- R: Incorporates GNSS position variances, which dominate the measurement noise for position. Reflects the variances of velocity and angular velocity measurements derived from accelerometer and gyroscope specifications and their combined noise power reflected in the Complementary Filter. Euler angles are obtained from the complementary filter, with noise derived as a weighted combination of accelerometer and gyroscope noise.

3.2.3 Control Module Design

The Control module is designed to replicate the real-world behavior of the control algorithms. It models the decoupled Longitudinal and Latitudinal controllers that manage thrust vectoring and engine throttle adjustments, incorporating system dynamics and response delays. Using separate controllers for the Longitudinal and Latitudinal dynamics simplifies the control design by allowing each controller to focus on a specific aspect of the rocket's motion. This separation reduces complexity and enables independent tuning of each controller, making it easier to optimize performance while also enhancing robustness, as disturbances in one axis, such as lateral wind gusts, can be handled locally without affecting the control of other axes [51]. Additionally, splitting the control into two simpler systems is computationally efficient compared to implementing a single, large multi-input multi-output controller.

Decoupling the dynamics is justified by the weak coupling between the Longitudinal and Latitudinal motions under normal operating conditions. Linearizing the system around stable points, reveals that the coupling terms are small enough to be approximated as negligible. This allows each controller to operate independently without significant loss of accuracy. Furthermore, the rocket's physical design, with orthogonal alignment of forces and torques along principal axes, inherently supports this decoupling. The design parameters of the controllers and their stability are detailed in Appendix C.

3.2.3.1 Longitudinal Controller

The Longitudinal Controller for the rocket is implemented using LQR [52] with Integral Action [53] for trajectory tracking to stabilize and control the longitudinal dynamics by having the longitudinal states $\mathbf{x_{lon}}$ reach their setpoints by controlling their respective control-inputs $\mathbf{u_{lon}}$, these are defined as:

$$\mathbf{x_{lon}} = [x \quad z \quad u \quad w \quad q \quad \theta]^T \qquad \textit{Eq. 46}$$
$$\mathbf{u_{lon}} = [\mu_p \quad T]^T$$

The LQR design uses the linearized **A** and **B** matrices evaluated at each flight phase around operating points $\mathbf{x_0}$.

$$\begin{aligned} \mathbf{A_{lon}} &= \mathbf{A}([1,3,4,6,7,9],[1,3,4,6,7,9]) \\ \mathbf{B_{lon}} &= \mathbf{B}([1,3,4,6,7,9],[1,3]) \\ \mathbf{C_{lon}} &= \mathbf{C}([1,3,4,6,7,9],[1,3,4,6,7,9]) \end{aligned} \qquad \textit{Eq. 47}$$

where, the notation $\mathbf{A}(i,j)$ refers to selecting the rows i and columns j from **A** with the indices being determined based on the arrangement of states and inputs in the full system matrices to extract the $\mathbf{A_{lon}}$ matrix. The same procedure applies for the $\mathbf{B_{lon}}$ and $\mathbf{C_{lon}}$ using the indices provided for $\mathbf{B}(i,j)$ and $\mathbf{C}(i,j)$. The longitudinal state-space with the output $\mathbf{y_{lon}}$ is represented as:

$$\begin{aligned} \mathbf{x_{lon}}[k+1] &= \mathbf{A_{lon}}\mathbf{x_{lon}}[k] + \mathbf{B_{lon}}\mathbf{u_{lon}}[k] \\ \mathbf{y_{lon}}[k] &= \mathbf{C_{lon}}\mathbf{x_{lon}}[k] \end{aligned} \qquad \textit{Eq. 48}$$

To compute the $\mathbf{K_{lon}}$ discrete-time optimal LQR gain matrix for the Longitudinal Controller, integral states are introduced to track the cumulative error between the reference $\mathbf{r_{lon}}$ and the output $\mathbf{y_{lon}}$, the state-space model of *Eq. 48* is augmented to include the integral states resulting in the following augmented state-space model:

$$\underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lon}}[k+1] \\ \mathbf{y}_{\mathbf{lon}}[k] - \mathbf{r}_{\mathbf{lon}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lon}}[k+1]} = \underbrace{\begin{bmatrix} \mathbf{A}_{\mathbf{lon}} & O \\ -\mathbf{C}_{\mathbf{lon}} & I \end{bmatrix}}_{\tilde{\mathbf{A}}_{\mathbf{lon}}} \underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lon}}[k] \\ \mathbf{y}_{\mathbf{lon}}[k] - \mathbf{r}_{\mathbf{lon}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lon}}[k]} + \underbrace{\begin{bmatrix} \mathbf{B}_{\mathbf{lon}}[k] \\ O \end{bmatrix}}_{\tilde{\mathbf{B}}_{\mathbf{lon}}} \mathbf{u}_{\mathbf{lon}}[k]$$

$$\underbrace{\mathbf{y}_{\mathbf{lon}}[k] - \mathbf{r}_{\mathbf{lon}}[k]}_{\tilde{\mathbf{y}}_{\mathbf{lon}}[k]} = \underbrace{[-\mathbf{C}_{\mathbf{lon}} \quad I]}_{\tilde{\mathbf{C}}_{\mathbf{lon}}} \underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lon}}[k] \\ \mathbf{y}_{\mathbf{lon}}[k] - \mathbf{r}_{\mathbf{lon}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lon}}[k]} \quad \textit{Eq. 49}$$

The Longitudinal LQR controller matrix that minimizes the cost function:

$$\mathbf{J}_{\mathbf{lon}} = \sum_{k=0}^{\infty} \left(\tilde{\mathbf{x}}_{\mathbf{lon}}[k]^T \mathbf{Q}_{\mathbf{lon}} \tilde{\mathbf{x}}_{\mathbf{lon}}[k] + \mathbf{u}_{\mathbf{lon}}\ [k]^T \mathbf{R}_{\mathbf{lon}} \mathbf{u}_{\mathbf{lon}}[k] \right) \quad \textit{Eq. 50}$$

where $\mathbf{Q}_{\mathbf{lon}}$ is a matrix which penalizes state deviations from the desired trajectory and $\mathbf{R}_{\mathbf{lon}}$ is a matrix which penalizes excessive control effort. This const function is computed by solving the discrete-time Algebraic Riccati equation [54]. The optimal control law in discrete-time is:

$$\mathbf{u}_{\mathbf{lon}}\ [k] = -\mathbf{K}_{\mathbf{lon}} \tilde{\mathbf{x}}_{\mathbf{lon}}[k] \quad \textit{Eq. 51}$$

3.2.3.2 Latitudinal Controller

The design of the Latitudinal Controller is akin to that of the Longitudinal controller. The latitudinal states $\mathbf{x}_{\mathbf{lat}}$ and their control-inputs $\mathbf{u}_{\mathbf{lat}}$ are defined as:

$$\mathbf{x}_{\mathbf{lat}} = [y \quad v \quad r \quad \psi]^T \quad \textit{Eq. 52}$$
$$\mathbf{u}_{\mathbf{lat}} = [\mu_y \quad T]^T$$

The linearized latitudinal state transition matrix $\mathbf{A}_{\mathbf{lat}}$ and the control-inputs matrix $\mathbf{B}_{\mathbf{lat}}$ are extracted from the **A** and **B** matrices evaluated at each flight phase around operating points $\mathbf{x_0}$.

$$\mathbf{A}_{\mathbf{lat}} = \mathbf{A}([2,5,8,10],[2,5,8,10]) \quad \textit{Eq. 53}$$
$$\mathbf{B}_{\mathbf{lat}} = \mathbf{B}([2,5,8,10],[1,2])$$
$$\mathbf{C}_{\mathbf{lat}} = \mathbf{C}([2,5,8,10],[2,5,8,10])$$

The latitudinal state-space representation is defined as:

$$\mathbf{x}_{\mathbf{lat}}[k+1] = \mathbf{A}_{\mathbf{lat}} \mathbf{x}_{\mathbf{lat}}[k] + \mathbf{B}_{\mathbf{lat}} \mathbf{u}_{\mathbf{lat}}[k] \quad \textit{Eq. 54}$$
$$\mathbf{y}_{\mathbf{lat}}[k] = \mathbf{C}_{\mathbf{lat}} \mathbf{x}_{\mathbf{lat}}[k]$$

To create the LQR controller with Integral Action, the longitudinal state-space is augmented with the integral states resulting in the following state space model:

$$\underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lat}}[k+1] \\ \mathbf{y}_{\mathbf{lat}}[k] - \mathbf{r}_{\mathbf{lat}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lat}}[k+1]} = \underbrace{\begin{bmatrix} \mathbf{A}_{\mathbf{lat}} & O \\ -\mathbf{C}_{\mathbf{lat}} & I \end{bmatrix}}_{\tilde{\mathbf{A}}_{\mathbf{lat}}} \underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lat}}[k] \\ \mathbf{y}_{\mathbf{lat}}[k] - \mathbf{r}_{\mathbf{lat}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lat}}[k]} + \underbrace{\begin{bmatrix} \mathbf{B}_{\mathbf{lat}}[k] \\ O \end{bmatrix}}_{\tilde{\mathbf{B}}_{\mathbf{lat}}} \mathbf{u}_{\mathbf{lat}}[k]$$

$$\underbrace{\mathbf{y}_{\mathbf{lat}}[k] - \mathbf{r}_{\mathbf{lat}}[k]}_{\tilde{\mathbf{y}}_{\mathbf{lat}}[k]} = \underbrace{[-\mathbf{C}_{\mathbf{lat}} \quad I]}_{\tilde{\mathbf{C}}_{\mathbf{lat}}} \underbrace{\begin{bmatrix} \mathbf{x}_{\mathbf{lat}}[k] \\ \mathbf{y}_{\mathbf{lat}}[k] - \mathbf{r}_{\mathbf{lat}}[k] \end{bmatrix}}_{\tilde{\mathbf{x}}_{\mathbf{lat}}[k]} \quad \textit{Eq. 55}$$

The Latitudinal LQR controller matrix that minimizes the cost function:

$$\mathbf{J}_{\mathbf{lat}} = \sum_{k=0}^{\infty} \left(\tilde{\mathbf{x}}_{\mathbf{lat}}[k]^T \mathbf{Q}_{\mathbf{lat}} \tilde{\mathbf{x}}_{\mathbf{lat}}[k] + \mathbf{u}_{\mathbf{lat}}\ [k]^T \mathbf{R}_{\mathbf{lat}} \mathbf{u}_{\mathbf{lat}}[k] \right) \quad \textit{Eq. 56}$$

where $\mathbf{Q}_{\mathbf{lat}}$ is the latitudinal state cost matrix and $\mathbf{R}_{\mathbf{lat}}$ is the latitudinal control cost matrix. The optimal feedback law for the Latitudinal Controller is computed as:

$$\mathbf{u}_{\mathbf{lat}}\ [k] = -\mathbf{K}_{\mathbf{lat}} \mathbf{x}_{\mathbf{lat}}[k] \quad \textit{Eq. 57}$$

where $\mathbf{K}_{\mathbf{lat}}$ is the latitudinal feedback gain matrix.

4 Simulation Results

Simulation runs were conducted using the devleoped co-simulation environment. As a simplification, no wind gusts or external disturbances were introduced into the test runs apart from the inherent noise characteristics of the sensors and the added process noise affecting the states.

To address the research question, two distinct simulation runs were performed. In the first scenario, the GNC subsystem operated solely using data from the IMU and GNSS sensors. In the second scenario, the system was augmented with visual navigation data derived from the onboard camera detecting fiducial markers. For both scenarios the seed values for the noise level generators inside the Simulink environment were kept the same, thus ensuring the GNC would be tested against the same cases unbiased. Presented in Figure 24, is an image capture inside the UE5 environment showing the rocket in Flight Phase 3 with the GNC detecting the ArUco marker and heat from the rocket engine's exhaust disrupting the image view around the landing pad.

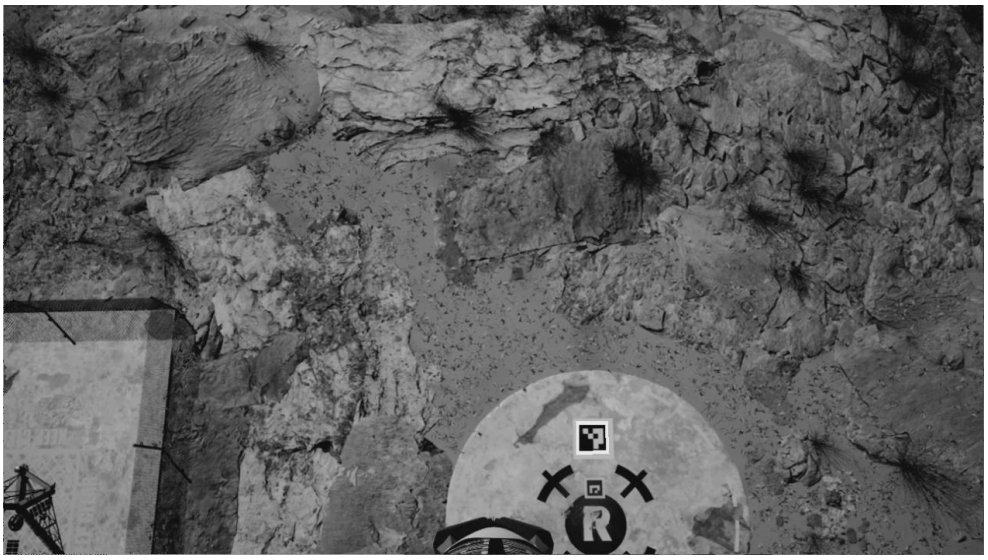


Figure 24: Image capture from the Camera inside the UE5

Results from init test runs are show in Figure 25 and Figure 26, where the true and estimated positions along the three axes of the rocket in both scenarios are presented, with the end of the flight phases, landing leg deployment, landing leg latching and engine cutoff marked along the vertical axis.

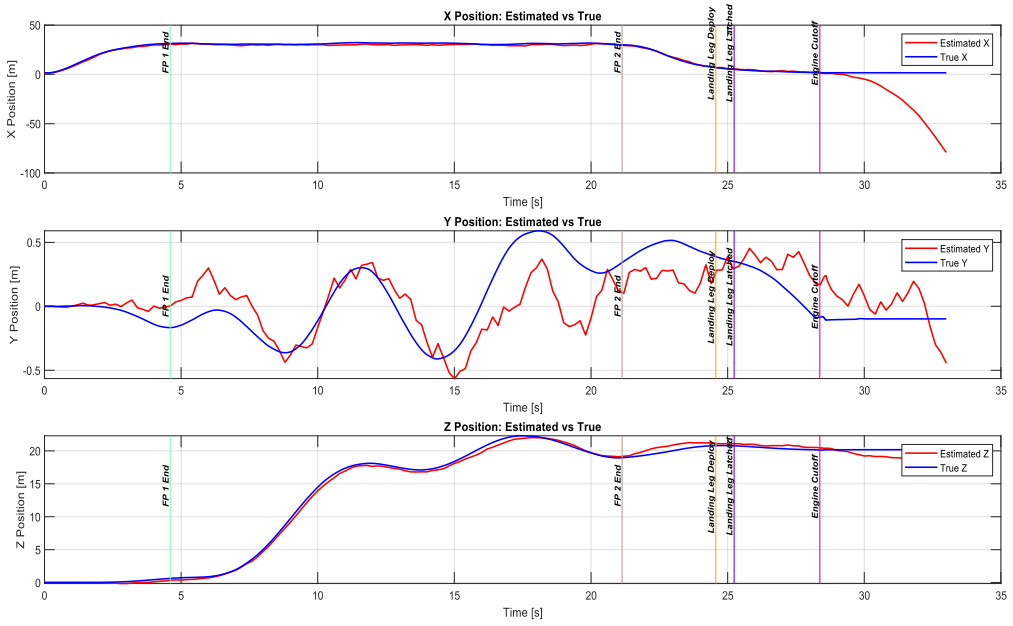


Figure 25: Estimated vs. True Position of the rocket using IMU+GNSS

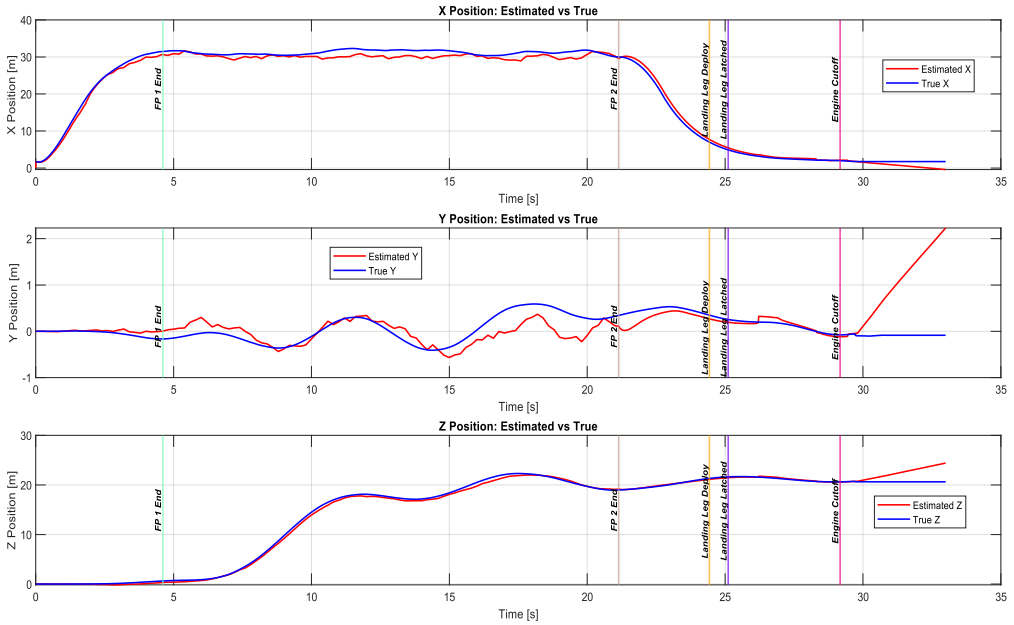


Figure 26: Estimated vs. True Position of the rocket using IMU+GNSS+Camera

As expected, the flight paths of the rocket up until the end of the Flight Phase 2 (marked as ***FP 2 End*** on the figures) are the same since the ArUco marker detection and position estimation starts from Flight Phase 3, until that point the Control module acts the same. Another observation which can be made is that the ***Engine Cutoff*** happens later, this implies that the Navigation module is much more confident where the rocket is and thus lands later for a controlled touch-down.

To have a better understanding of the results, Figure 27 presents only the flight path of the rocket from the beginning of Flight Phase 3 until touchdown. As it can be seen, the rocket is landing faster if the GNC is not using the Camera to get its position from the ArUco makers. Further analysis of the figure shows that the positioning with the Camera after 26.2 [s] the position estimation has an added bias, this can be clearly seen on the **Y Position: Camera vs GNSS** subplot. As described in 3.2.1.2, this is due to the fact that the ArUco marker disappear from the camera view at which point there is no positioning computed with it and thus the inertial navigation using only the IMU takes over to try and land the hopper at its designated position.

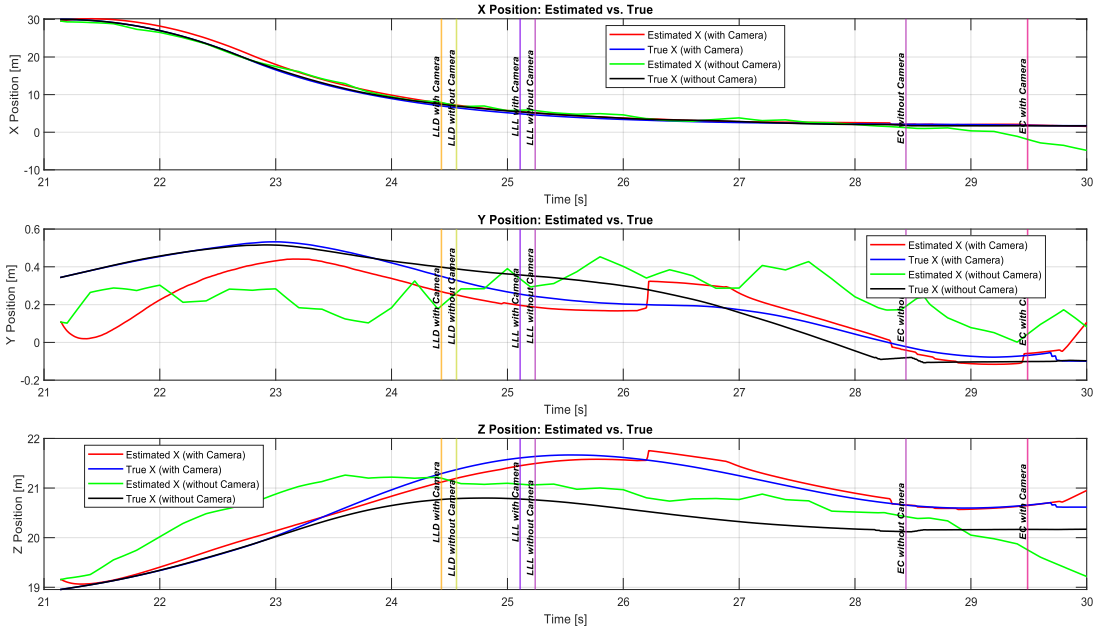


Figure 27: Flight Phase 3 Estimated vs. True Position of the rocket

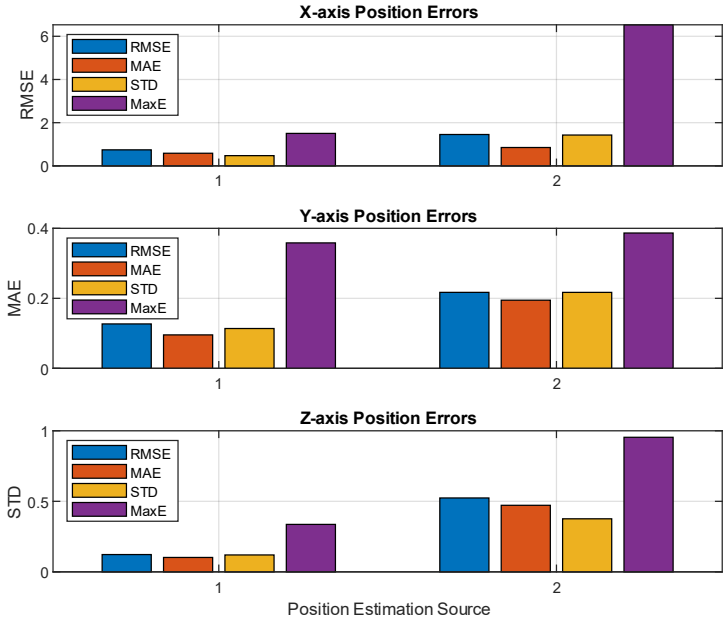


Figure 28: Comparison of RMSE, MAE and MaxAE between the GNC using the Camera (bar plot 1) and without using the Camera (bar plot 2)

To deduce if the Camera used during the final Flight Phase improves the position estimation and its overall controlled landing of the rocket, statistical measures are presented in Figure 28, such as the Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Maximum Absolute Error (MaxAE) for each axis can be used between the estimated and the true positions. These metrics quantify how closely the estimated positions match the true positions, as it can be seen from the figure, using the Camera slightly improves the position estimation, this is especially true because the vertical positioning accuracy (as described in Table 4) of the GNSS unit of the MTi-7 is greater than those of the other axes.

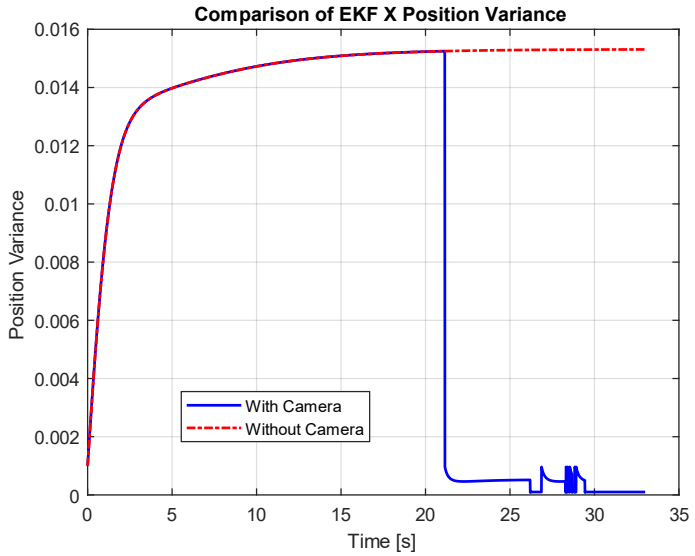


Figure 29: Comparison of EKF X Position Variance

As stated before, further investigating the EKF position variance confirms that when the Guidance module commands the Navigation module to start using the ArUco marker data, the confidence of the rocket's vertical position is increased (as shown in Figure 29) by the fast decrease in the position variance.

After the initial tests, a total of 100 Monte Carlo simulation runs were conducted to compare the performance of the GNC system in the two scenarios. This way the accuracy, precision and robustness of the landings are evaluated, this is done by changing the seed values for the noise generators of the sensor and process noises in the Simulink environment for each simulation run. Thus, a range of disturbances is simulated, allowing for a thorough assessment of the GNC's ability to ensure a stable and controlled landing under varying conditions.

4.1 GNC Accuracy and Precision

The accuracy and precision is evaluated based on the true y_E lateral and true z_E longitudinal positions recorded during the simulation runs. These measures were determined the following way:

- *Accuracy* was determined by calculating the closeness of the average landing position to the target landing pad position, this was quantified as the Euclidean distance between the average landing position across all runs and the target landing pad position.
- The *precision*, on the other hand, was calculated to measure the consistency of the landing positions across the simulation runs, this was evaluated as the standard deviation of the landing positions around the average landing position.

Presented in Figure 30 are two areas shaded in red for the landings performed without using the Camera and in blue where the GNC used the Camera.

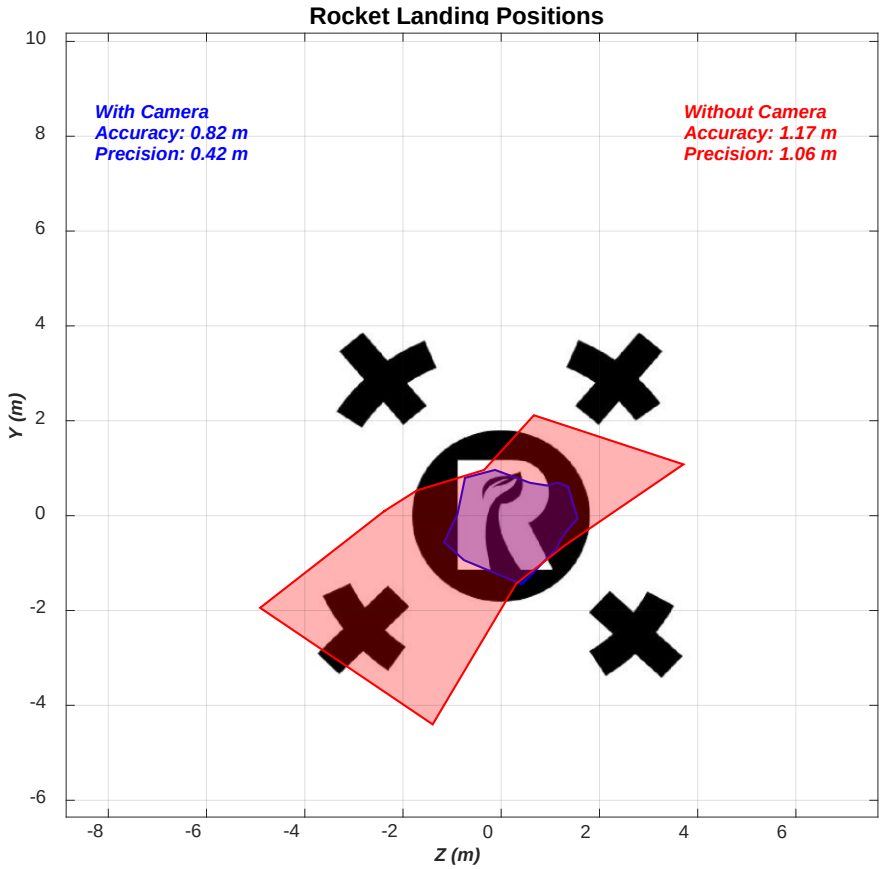


Figure 30: Rocket landing positions on the landing area

The shaded areas represent the most frequent areas where the rocket has landed, created by generating a convex hull from the recorded landing positions. As it can be seen on the figure above, the blue area has a smaller spread compared to the red area, indicating that the precision of the GNC, when it also uses the Camera, is greater as it resulted in precision of 0.42 $[m]$ compared to the 1.06 $[m]$ precision. As for the accuracy, the GNC performed yet again better when it had access to the Camera-based positioning, as it averaged a 0.82 $[m]$ accuracy compared to 1.17 $[m]$.

4.2 GNC Robustness

To evaluate the robustness, the landing outcomes of the rocket were categorized into three distinct categories: crash-landing, fallen over after landing, and successful landing. This classification provides a qualitative measure of how well the GNC handles various noise conditions introduced during the simulations. The landing categories are defined as follows:

- A *crash-landing* occurs when the rocket impacts the ground with insufficient control to decelerate and stabilize before touchdown. This outcome is identified if the rocket's true x_E position is below the height corresponding to the origin of the rocket plus the height gained from the fully deployed landing legs. Additionally, a crash-landing requires that the true u vertical velocity at impact to be less than $-1.0\ \left[\frac{m}{s}\right]$. If these conditions are met, the rocket is considered to have crash-landed.
- The *fallen over after landing* category represents a scenario where the rocket successfully lands but subsequently becomes unstable and tips over. This is determined by monitoring the θ and ψ rocket's Euler angles in the Flat-Earth reference frame at the end of the simulation. If the absolute value of either angle exceeds 7.5 [°], the rocket is classified as having fallen over, indicating that the rocket had deviated too much from the 0.0 [°] desired Euler angles at the end of the flight phase.
- A *successful landing* occurs when the rocket touches down, decelerates effectively, and remains upright throughout the simulation. In this scenario, the rocket satisfies the altitude and velocity thresholds for landing without tipping over, maintaining the θ and ψ Euler angles within ± 7.5 [°].

The gathered data for the analysis of the landing, post rocket engine cutoff, was done by letting the UE5 physics engine run for an additional 5 $[s]$ with the final states of the rocket transmitted from Simulink. During this time period where the Control module was completely shut-down, allowed the UE5 to simulate the post touch-down states of the rocket and transmit these states back to Simulink.

While there was not much of a difference between the landing accuracy, the robustness offers a better overview of the gained performance to use landing markers for rocket hoppers. The results of the 60 simulation runs (presented Figure 31) demonstrate the significant impact of incorporating visual navigation data from the camera on the robustness of the GNC system.

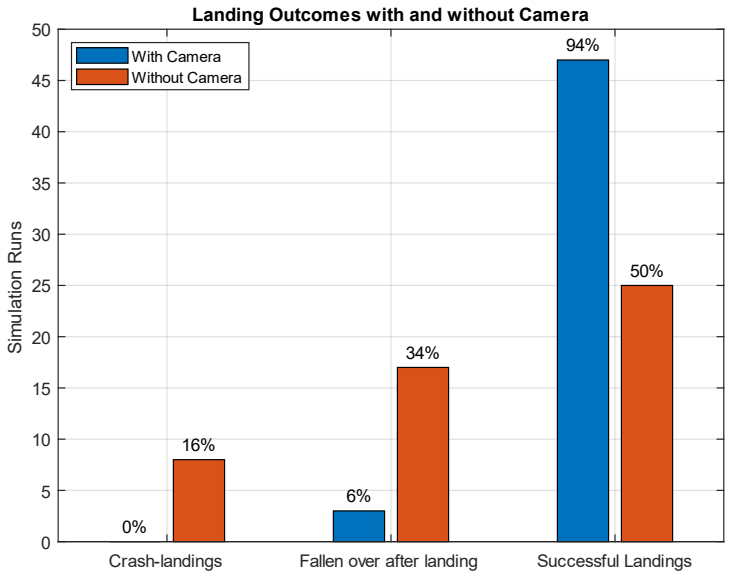


Figure 31: Post touchdown landing categories

With the inclusion of ArUco marker-based positioning during the third flight phase, the GNC system achieved a 94% success rate for stable landings, compared to only 50% in the scenario without camera data. Additionally, no crash-landings occurred in the camera-assisted scenario, while 8 were observed

when relying solely on IMU and GNSS measurements. Furthermore, the number of instances where the rocket fell over after landing was significantly reduced from 17 to 3 with the use of visual navigation. These results underscore the ability of fiducial marker-based navigation to enhance landing stability and accuracy, particularly during the critical descent phase.

	Average Remaining Propellant
Without Camera	1.86 $[kg]$
With Camera	1.77 $[kg]$

Table 6: Average Remaining Propellant

Interestingly enough, the added accuracy, precision and robustness does not come with much of a cost. As shown in Table 6, throughout the $50 - 50$ simulation runs the rocket had on an average around 1.80 $[kg]$ of propellant remaining out of the 5 $[kg]$ with the average propellant consumption when the GNC uses the Camera slightly higher than when it is not.

5 Conclusion

The results of the simulations provide initial insights into the impact of incorporating fiducial marker-based visual navigation on the GNC system's performance during the rocket's landing phase. The comparison between the two scenarios—using only IMU and GNSS data versus augmenting the system with camera-based position estimates—demonstrates significant improvements in accuracy, precision, and robustness when visual navigation is employed.

In the Camera-assisted scenario, the GNC system achieved a 94% success rate for stable landings, compared to only 50% when relying solely on IMU and GNSS data. Furthermore, the number of instances where the rocket fell over after landing was reduced from 17 to 3, and no crash-landings occurred with the camera-assisted configuration. These results emphasize the ability of fiducial marker-based navigation to enhance landing stability and reduce variability, particularly during the critical final moments of descent.

Metrics such as RMSE, MAE, and MaxAE also confirm that the addition of the camera improves position estimation accuracy, especially in the lateral and longitudinal axes. However, the results also highlight limitations in the current approach. When the ArUco marker disappears from the camera's field of view near the landing site, the system reverts to relying solely on the IMU data, leading to a loss of the additional accuracy gained from visual navigation.

6 Outlook

While the integration of fiducial marker-based visual navigation has demonstrated potential for improving the accuracy and robustness of the rocket's GNC subsystem, several areas warrant further development to enhance its overall performance. One of the key limitations observed in the current design is the lack of onboard or online computation of controller gains. The GNC relies on pre-computed gain schedules, which may not optimally handle scenarios involving heavier disturbances or unpredictable noise levels. Incorporating real-time gain adaptation, such as through adaptive control techniques or gain scheduling that continuously adjusts to changing flight dynamics, could significantly improve the system's ability to respond to unforeseen conditions, enhancing the rocket's robustness during all flight phases.

Additionally, the assumption that roll control is already implemented in the rocket introduces a simplification that may not hold for all designs or missions. While this assumption was valid for the scope of this thesis, addressing roll stability is crucial for a comprehensive GNC system. Future iterations of the rocket could explore implementing roll control mechanisms such as cold gas thrusters or jet vanes. Cold gas thrusters provide precise, rapid adjustments to stabilize roll but may add significant mass to the vehicle. Alternatively, jet vanes, which direct the exhaust flow for control authority, offer a lightweight solution but may require modifications to the engine design. The inclusion of a dedicated roll control subsystem would ensure full control over the rocket's six degrees of freedom, paving the way for more complex and dynamic mission profiles.

Finally, while this thesis focused on short, controlled flight tests with minimal external disturbances, the next steps should include testing the GNC system under more realistic environmental conditions. Introducing wind gusts, sensor drift over time, and varying levels of external noise in simulation or real-world testing would provide a clearer understanding of the system's robustness. Such testing, combined with addressing the aforementioned limitations, would move the GNC design closer to operational readiness, ensuring its effectiveness across a broader range of scenarios.

References

[1] "4 Trends Shaping The Global Space Sector In 2024." Accessed: Apr. 29, 2024. [Online]. Available: https://www.oliverwyman.com/our-expertise/insights/2024/jan/four-trends-shaping-the-space-sector.html

[2] "Riding the exponential growth in space," Deloitte Insights. Accessed: Apr. 29, 2024. [Online]. Available: https://www2.deloitte.com/us/en/insights/industry/aerospace-defense/future-of-space-economy.html

[3] "Lander Challenge." Accessed: Apr. 29, 2024. [Online]. Available: https://landerchallenge.space/

[4] L. Blackmore, "Autonomous precision landing of space rockets," vol. 46, pp. 15–20, Jan. 2016.

[5] E. S. Agency, "Prometheus Ignites: Future of Space Travel With Reusable Rockets," SciTechDaily. Accessed: Apr. 29, 2024. [Online]. Available: https://scitechdaily.com/prometheus-ignites-future-of-space-travel-with-reusable-rockets/

[6] T. J. Steiner and T. M. Brady, "Vision-based navigation and hazard detection for terrestrial rocket approach and landing," in *2014 IEEE Aerospace Conference*, Mar. 2014, pp. 1–8. doi: 10.1109/AERO.2014.6836216.

[7] "Masten's Xombie flight with Draper Lab's GENIE Controls - NASA." Accessed: Jan. 10, 2025. [Online]. Available: https://www.nasa.gov/image-article/mastens-xombie-flight-with-draper-labs-genie-controls/

[8] M. Dumke, G. F. Trigo, M. Sagliano, P. Saranrittichai, and S. Theil, "Design, development, and flight testing of the vertical take-off and landing GNC testbed EAGLE," *CEAS Space J*, vol. 12, no. 1, pp. 97–113, Jan. 2020, doi: 10.1007/s12567-019-00269-5.

[9] M. Fritz *et al.*, "Post-Flight Performance Analysis of Navigation and Advanced Guidance Algorithms on a Terrestrial Suborbital Rocket Flight," in *AIAA SCITECH 2022 Forum*, American Institute of Aeronautics and Astronautics, 2022. doi: 10.2514/6.2022-0765.

[10] A. Botelho, M. Martinez, C. Recupero, A. Fabrizi, and G. De Zaiacomo, "Design of the landing guidance for the retro-propulsive vertical landing of a reusable rocket stage," *CEAS Space J*, vol. 14, no. 3, pp. 551–564, Jul. 2022, doi: 10.1007/s12567-022-00423-6.

[11] J. Orphee *et al.*, "Evaluation of Precision Landing Performance Using A Generalized Aerospace Simulation in Simulink Framework," presented at the 45th Annual AAS Guidance, Navigation and Control (GN&C) Conference, Breckenridge, CO, 2023. Accessed: Apr. 29, 2024. [Online]. Available: https://ntrs.nasa.gov/citations/20230000011

[12] A. Marut, K. Wojtowicz, and K. Falkowski, "ArUco markers pose estimation in UAV landing aid system," in *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, Jun. 2019, pp. 261–266. doi: 10.1109/MetroAeroSpace.2019.8869572.

[13] "Reusable carrier rocket passes the 'hop test.'" Accessed: Jun. 29, 2024. [Online]. Available: https://mobile.chinadaily.com.cn/cn/html5/2024-06/25/content_005_6679e2d5ed50dde094234d70.htm

[14] G. P. Sutton and O. Biblarz, *Rocket Propulsion Elements*, 9th ed. John Wiley & Sons, 2001.

[15] "Rocket Stability Condition," Glenn Research Center | NASA. Accessed: Apr. 30, 2024. [Online]. Available: https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/conditions-for-rocket-stability/

[16] "MTi-1 Series Datasheet." XSense, Dec. 2019. Accessed: Jul. 08, 2024. [Online]. Available: https://www.xsens.com/hubfs/Downloads/Manuals/MTi-1-series-datasheet.pdf

[17] "Real-time kinematic positioning," *Wikipedia*. Sep. 30, 2024. Accessed: Jan. 10, 2025. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Real-time_kinematic_positioning&oldid=1248624457

[18] I. Safrel, E. N. Julianto, and N. Q. Usman, "Accuracy Comparison between GPS Real Time Kinematic (RTK) Method and Total Station to Determine The Coordinate of An Area," *JTSP*, vol. 20, no. 2, pp. 123–130, Nov. 2018, doi: 10.15294/jtsp.v20i2.16284.

[19] "Camera - Raspberry Pi Documentation." Accessed: Nov. 17, 2024. [Online]. Available: https://www.raspberrypi.com/documentation/accessories/camera.html

[20] P. T. Kabamba and A. R. Girard, *Fundamentals of Aerospace Navigation and Guidance*. in Cambridge Aerospace Series. Cambridge: Cambridge University Press, 2014. doi: 10.1017/CBO9781107741751.

[21] "Coordinate System and Spaces in Unreal Engine | Unreal Engine 5.4 Documentation | Epic Developer Community," Epic Games Developer. Accessed: Jun. 30, 2024. [Online]. Available: https://dev.epicgames.com/documentation/en-us/unreal-engine/coordinate-system-and-spaces-in-unreal-engine

[22] "Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems, 3rd Edition | Wiley," Wiley.com. Accessed: Jul. 01, 2024. [Online]. Available: https://www.wiley.com/en-us/Aircraft+Control+and+Simulation%3A+Dynamics%2C+Controls+Design%2C+and+Autonomous+Systems%2C+3rd+Edition-p-9781118870983

[23] G. Cai, B. M. Chen, and T. H. Lee, *Unmanned Rotorcraft Systems*. in Advances in Industrial Control. London: Springer, 2011. doi: 10.1007/978-0-85729-635-1.

[24] “Geographic coordinate conversion,” *Wikipedia*. Aug. 10, 2024. Accessed: Nov. 23, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Geographic_coordinate_conversion&oldid=1239592446
[25] OpenCV, “Detection of ArUco Markers,” Detection of ArUco Markers. Accessed: Oct. 08, 2024. [Online]. Available: https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html
[26] A. Tewari, *Advanced Control of Aircraft, Spacecraft and Rockets*. John Wiley & Sons, 2011.
[27] “Department of Defense World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems. Second Edition.” Accessed: Jul. 08, 2024. [Online]. Available: https://apps.dtic.mil/sti/citations/ADA280358
[28] P. dos Santos and P. Oliveira, “ADCS Design for a Sounding Rocket with Thrust Vectoring,” *Aerotec. Missili Spaz.*, vol. 102, no. 3, pp. 257–270, Sep. 2023, doi: 10.1007/s42496-023-00161-w.
[29] A. Bhar Kisabo and A. Funmilayo Adebimpe, “State-Space Modeling of a Rocket for Optimal Control System Design,” in *Ballistics*, C. Osheku, Ed., IntechOpen, 2019. doi: 10.5772/intechopen.82292.
[30] “Identity matrix,” *Wikipedia*. Oct. 17, 2024. Accessed: Nov. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Identity_matrix&oldid=1251662245
[31] “Zero-order hold,” *Wikipedia*. Jul. 19, 2024. Accessed: Nov. 23, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Zero-order_hold&oldid=1235522344
[32] C. Huygens, “Horologium Oscillatorium.” 1673. Accessed: Jul. 08, 2024. [Online]. Available: http://www.17centurymaths.com/contents/huygens/horologiumpart4a.pdf
[33] “Three-axis Accelerometer.” Accessed: Nov. 17, 2024. [Online]. Available: https://nl.mathworks.com/help/aeroblks/threeaxisaccelerometer.html
[34] X. Xu *et al.*, “Bandwidth Optimization of MEMS Accelerometers in Fluid Medium Environment,” *Sensors*, vol. 22, no. 24, p. 9855, Dec. 2022, doi: 10.3390/s22249855.
[35] “Three-axis Gyroscope.” Accessed: Nov. 17, 2024. [Online]. Available: https://nl.mathworks.com/help/aeroblks/threeaxisgyroscope.html
[36] “GPS.” Accessed: Nov. 17, 2024. [Online]. Available: https://nl.mathworks.com/help/fusion/ref/gps.html
[37] Y. Wu and S. Chen, “An enhanced stochastic error modeling using multi-Gauss–Markov processes for GNSS/INS integration system,” *Journal of Engineering and Applied Science*, vol. 71, no. 1, p. 186, Sep. 2024, doi: 10.1186/s44147-024-00520-9.
[38] “Simulation 3D Camera.” Accessed: Nov. 17, 2024. [Online]. Available: https://nl.mathworks.com/help/driving/ref/simulation3dcamera.html#mw_389e26ab-1e71-4eab-bffd-48f9a503fff0
[39] C. Beam, J. Zhang, N. Kakavitsas, C. Hague, A. Wolek, and A. Willis, “Cesium Tiles for High-Realism Simulation and Comparing SLAM Results in Corresponding Virtual and Real-World Environments,” in *SoutheastCon 2024*, Mar. 2024, pp. 95–100. doi: 10.1109/SoutheastCon52093.2024.10500076.
[40] S. Sisi and A. Gany, “Parametric investigation of a hybrid motor using paraffin and nitrous oxide,” *54th Israel Annual Conference on Aerospace Sciences 2014*, vol. 2, pp. 944–959, Jan. 2014.
[41] NASA’s Ames Research Center, *NASA Tests Rocket Powered By Paraffin Fuel*, (Apr. 26, 2017). Accessed: Jan. 11, 2025. [Online Video]. Available: https://www.youtube.com/watch?v=ZrswPmPQiy8
[42] S. C. Fisher and S. A. Rahman, “Remembering The Giants Apollo Rocket Propulsion Development,” *The NASA History Series*, p. 80, Dec. 2009.
[43] Banijay Science, *Engineering the Moon Landing - Engineering Space - S01 EP02 - Space Documentary*, (Oct. 23, 2023). Accessed: Nov. 30, 2024. [Online Video]. Available: https://www.youtube.com/watch?v=6d5cOb7r3j4
[44] R. G. Valenti, I. Dryanovski, and J. Xiao, “Keeping a Good Attitude: A Quaternion-Based Orientation Filter for IMUs and MARGs,” *Sensors*, vol. 15, no. 8, Art. no. 8, Aug. 2015, doi: 10.3390/s150819302.
[45] “Digital Butterworth Filter Calculator.” Accessed: Nov. 22, 2024. [Online]. Available: https://www.meme.net.au/butterworth.html
[46] “Kalman filter,” *Wikipedia*. Nov. 22, 2024. Accessed: Nov. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Kalman_filter&oldid=1258985233
[47] “Zero matrix,” *Wikipedia*. Sep. 20, 2024. Accessed: Nov. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Zero_matrix&oldid=1246697967
[48] “An introduction to Kalman filters.” Accessed: Nov. 23, 2024. [Online]. Available: https://cheever.domains.swarthmore.edu/Ref/Kalman/MatrixKalman.html
[49] “Multivariate normal distribution,” *Wikipedia*. Nov. 16, 2024. Accessed: Nov. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Multivariate_normal_distribution&oldid=1257828226
[50] “White noise,” *Wikipedia*. Nov. 10, 2024. Accessed: Nov. 24, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=White_noise&oldid=1256583744

[51] H. Xin, Q. Chen, P. Wang, Y. Wang, Z. Hou, and Y. Lu, “Research on Longitudinal Attitude and Velocity Coordinated Control Method for a Low Aspect Ratio Flying Wing Aircraft,” in *2023 Asia-Pacific International Symposium on Aerospace Technology (APISAT 2023) Proceedings*, S. Fu, Ed., Singapore: Springer Nature, 2024, pp. 225–238. doi: 10.1007/978-981-97-4010-9_17.

[52] L. Martins, C. Cardeira, and P. Oliveira, “Linear Quadratic Regulator for Trajectory Tracking of a Quadrotor,” *IFAC-PapersOnLine*, vol. 52, no. 12, pp. 176–181, Jan. 2019, doi: 10.1016/j.ifacol.2019.11.195.

[53] D. D. Ruscio, “Discrete LQ optimal control with integral action: A simple controller on incremental form for MIMO systems,” *MIC*, vol. 33, no. 2, pp. 35–44, 2012, doi: 10.4173/mic.2012.2.1.

[54] “Algebraic Riccati equation,” *Wikipedia*. Nov. 19, 2024. Accessed: Nov. 25, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Algebraic_Riccati_equation&oldid=1258405549#Context_of_the_discrete-time_algebraic_Riccati_equation

Appendix

A. Operating Points

To linearize the equations of motion, the system is evaluated at small disturbances around equilibrium points corresponding to the end states of each flight phase. The operating points resented in Table 7, denoted as $\mathbf{x_0}$, include the states and control inputs that define the rocket's behavior in hovering, lateral movement, and descent.

Flight Phase	***I***	***II***	***III***
$x_E\ [m]$	30	30	0
$y_E\ [m]$	0.1	0.1	0.1
$z_E\ [m]$	0.1	20	20
$u\ \left[\frac{m}{s}\right]$	0.5	0.5	0.5
$v\ \left[\frac{m}{s}\right]$	0.5	0.5	0.5
$w\ \left[\frac{m}{s}\right]$	0.5	0.5	0.5
$q\ \left[\frac{Rad}{s}\right]$	0.05	0.05	0.05
$r\ \left[\frac{Rad}{s}\right]$	0.05	0.05	0.05
$\theta\ [Rad]$	0.0175	0.0175	0.0175
$\psi\ [Rad]$	0.0175	0.0175	0.0175
$T\ [N]$	$m \cdot g$	$m \cdot g$	$m \cdot g$
$\mu_p\ [Rad]$	0.0175	0	0
$\mu_y\ [Rad]$	0.0175	0.0175	0.0175

Table 7: Flight phase equilibrium points

B. Kalman Filter Covariance Matrices and Parameters

The covariance matrices are constructed from the values presented in the tables below in a diagonal matrix, this essentially creates the assumption that the noises affecting the state variables are uncorrelated.

State	***Variance***
Position (x_E, y_E, z_E)	$10 \cdot 10^{-6}$
Velocity (u, v, w)	$10 \cdot 10^{-6}$
Angular Velocity (q, r)	$10 \cdot 10^{-6}$
Euler Angles (θ, ψ)	$10 \cdot 10^{-6}$
Accelerometer Biases (b_u, b_v, b_w)	$5.0 \cdot 10^{-2}$
Gyroscope Biases (b_q, b_r)	$5.0 \cdot 10^{-2}$

Table 8: Initial State Error Covariance Matrix P_0

State	***Variance***
Position (x_E, y_E, z_E)	$8.0 \cdot 10^{-2}$
Velocity (u, v, w)	$1.0 \cdot 10^{-4}$
Angular Velocity (q, r)	$2.0 \cdot 10^{-6}$
Euler Angles (θ, ψ)	$1.0 \cdot 10^{-6}$
Accelerometer Biases (b_u, b_v, b_w)	$1.0 \cdot 10^{-6}$
Gyroscope Biases (b_q, b_r)	$1.0 \cdot 10^{-5}$

Table 9: Process Noise Covariance Matrix Q

Measurement	***Axis/State***	***Variance***
Position (GNSS)	y, z	16
Position (GNSS)	x	4
Velocity (Accelerometer)	u	$1.310 \cdot 10^{-5}$
Velocity (Accelerometer)	v, w	$2.007 \cdot 10^{-5}$
Angular Velocity (Gyroscope)	q, r	$1.440 \cdot 10^{-5}$
Euler Angles (Complementary Filter)	θ, ψ	$0.98 \cdot 1.440 \cdot 10^{-5} + 2.007 \cdot 10^{-5} \cdot 0.0448$

Table 10: Sensor Noise Covariance R for Flight Phase I and II

Measurement	***Axis/State***	***Variance***
Position (Camera ArUco)	x, y, z	0.025
Velocity (Accelerometer)	u	$1.310 \cdot 10^{-5}$
Velocity (Accelerometer)	v, w	$2.007 \cdot 10^{-5}$
Angular Velocity (Gyroscope)	q, r	$1.440 \cdot 10^{-5}$
Euler Angles (Complementary Filter)	θ, ψ	$0.98 \cdot 1.440 \cdot 10^{-5} + 2.007 \cdot 10^{-5} \cdot 0.0448$

Table 11: Sensor Noise Covariance R for Flight Phase III Camera-based positioning

Measurement	***Axis/State***	***Variance***
Position (Camera ArUco)	x, y, z	$1 \cdot 10^{-3}$
Velocity (Accelerometer)	u	$1.310 \cdot 10^{-5}$
Velocity (Accelerometer)	v, w	$2.007 \cdot 10^{-5}$
Angular Velocity (Gyroscope)	q, r	$1.440 \cdot 10^{-5}$
Euler Angles (Complementary Filter)	θ, ψ	$0.98 \cdot 1.440 \cdot 10^{-5} + 2.007 \cdot 10^{-5} \cdot 0.0448$

Table 12: Sensor Noise Covariance R for Flight Phase III IMU-based positioning

C. Controller Design Parameters and Closed-loop Stability

The penalty matrices of the LQI design are constructed from the values presented in the tables below in a diagonal matrix.

As state variables have different units (e.g., $[m]$ for position and $\left[\frac{m}{s}\right]$ for velocity), controller tuning was done by normalizing these such that their contributions to the cost function are comparable. For example: weighting of Velocity u is done by setting the weight to the normalized maximum error which the controller should permit the state to deviate from, in case of Flight Phase I this is $0.6\ \left[\frac{m}{s}\right]$, thus the weight is set to $\frac{1}{0.6^2}$.

To ensure that the rocket has a soft touch-down in Flight Phase III, the weight for the Integral of Position x_E is set to a low value, this places less importance on the cumulative error correction, thus the controller prioritizes regulating faster changing states (e.g., velocities) more aggressively. This, in combination of setting the Thrust T to a lower value than the maximum thrust capable of the Alpha's HRE, makes the rocket hopper kill of its vertical velocity just before hitting the ground instead of constantly burning its propellant and climbing down at a slower rate. This technique is called retropropulsive landing, also known as suicide burn in the rocketry community.

Penalized State	***Weight Value***
Position x_E	1
Position z_E	1
Velocity u	$\frac{1}{0.6^2}$
Velocity w	$\frac{1}{0.6^2}$
Angular Velocity q	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle θ	$\frac{1}{\left(2 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position x_E	2.5
Integral of Position z_E	2.5

Table 13: Flight Phase I Longitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Pitch μ_p	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{600^2}$

Table 14: Flight Phase I Longitudinal Input Penalty Matrix R

Penalized State	***Weight Value***
Position y_E	$\frac{1}{0.6^2}$
Velocity v	$\frac{1}{0.1^2}$
Angular Velocity r	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle ψ	$\frac{1}{\left(2 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position y_I	$2 \cdot \frac{1}{0.6^2}$

Table 15: Flight Phase I Latitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Yaw μ_y	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{100^2}$

Table 16: Flight Phase I Latitudinal Input Penalty Matrix R

Penalized State	***Weight Value***
Position x_E	1
Position z_E	1
Velocity u	$\frac{1}{0.6^2}$
Velocity w	$\frac{1}{0.6^2}$
Angular Velocity q	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle θ	$\frac{1}{\left(2 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position x_E	2.5
Integral of Position z_E	2.5

Table 17: Flight Phase II Longitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Pitch μ_p	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{600^2}$

Table 18: Flight Phase II Longitudinal Input Penalty Matrix R

Penalized State	***Weight Value***
Position y_E	$\frac{1}{0.8^2}$
Velocity v	$\frac{1}{0.3^2}$
Angular Velocity r	$\frac{1}{\left(1 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle ψ	$\frac{1}{\left(10 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position y_I	$2 \cdot \frac{1}{0.8^2}$

Table 19: Flight Phase II Latitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Yaw μ_y	$\frac{1}{\left(1 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{100^2}$

Table 20: Flight Phase II Latitudinal Input Penalty Matrix *R*

Penalized State	***Weight Value***
Position x_E	1
Position z_E	1
Velocity u	$\frac{1}{0.7^2}$
Velocity w	$\frac{1}{0.5^2}$
Angular Velocity q	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle θ	$\frac{1}{\left(4 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position x_E	0.01
Integral of Position z_E	3

Table 21: Flight Phase III Longitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Pitch μ_p	$\frac{1}{\left(1 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{200^2}$

Table 22: Flight Phase III Longitudinal Input Penalty Matrix R

Penalized State	***Weight Value***
Position y_E	$\frac{1}{0.6^2}$
Velocity v	$\frac{1}{0.1^2}$
Angular Velocity r	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Euler Angle ψ	$\frac{1}{\left(2 \cdot \frac{\pi}{180}\right)^2}$
Integral of Position y_I	$2 \cdot \frac{1}{0.6^2}$

Table 23: Flight Phase III Latitudinal State Penalty Matrix Q

Penalized Input	***Weight Value***
TVC Yaw μ_y	$\frac{1}{\left(0.5 \cdot \frac{\pi}{180}\right)^2}$
Thrust T	$\frac{1}{100^2}$

Table 24: Flight Phase III Latitudinal Input Penalty Matrix *R*

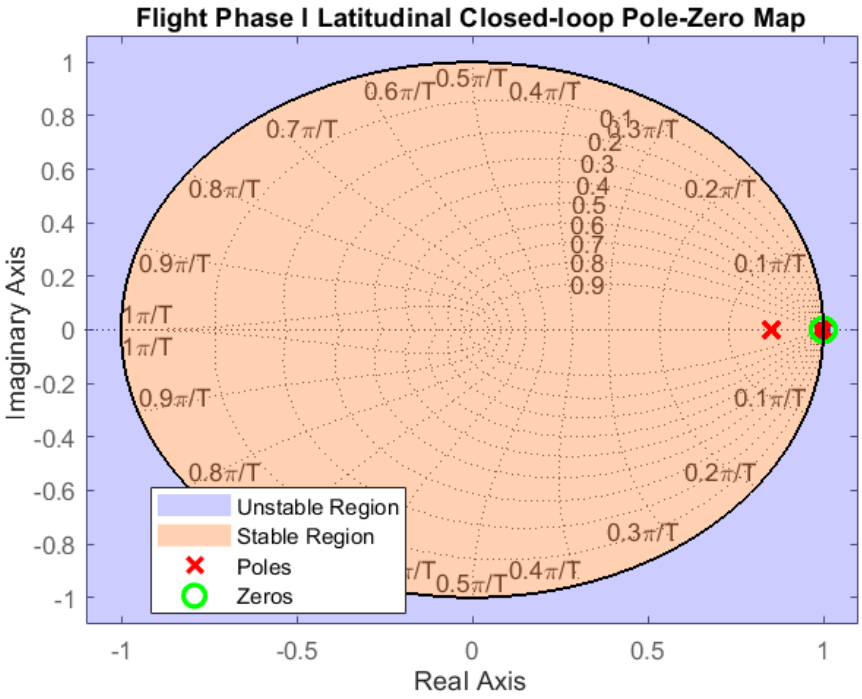


Figure 32: Flight Phase I Latitudinal Dynamics Closed-loop Stability

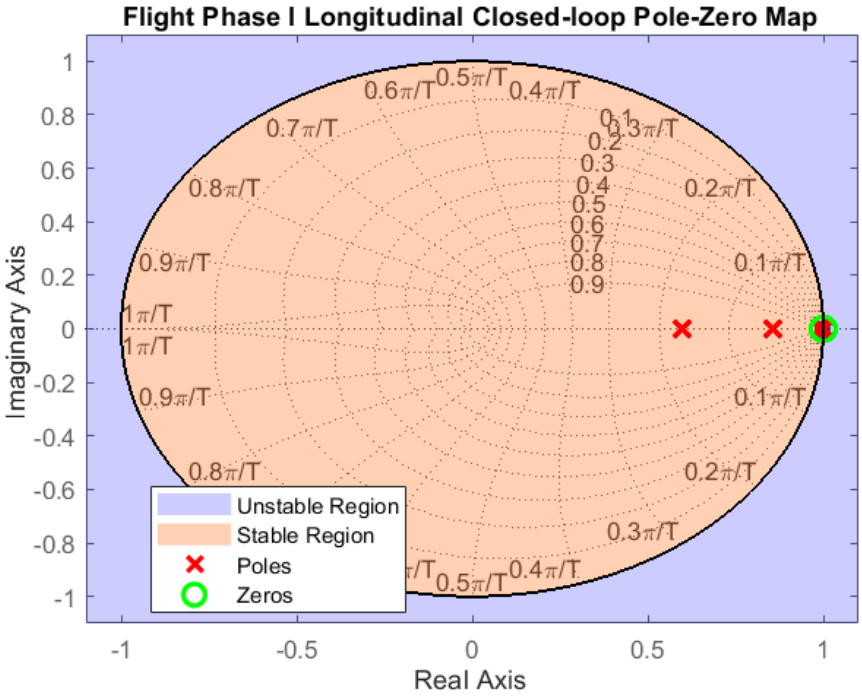


Figure 33: Flight Phase I Longitudinal Dynamics Closed-loop Stability

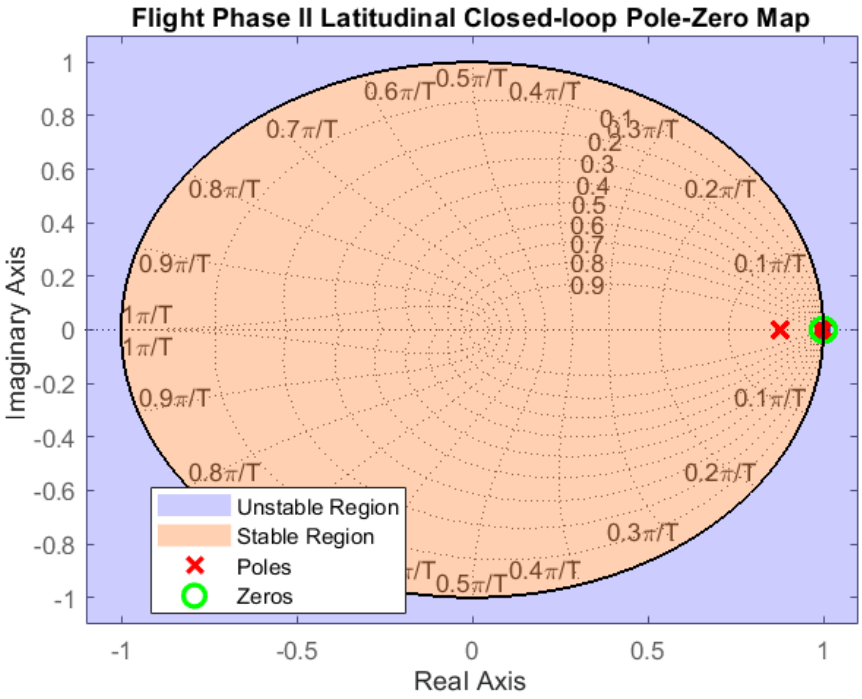


Figure 34: Flight Phase II Latitudinal Dynamics Closed-loop Stability

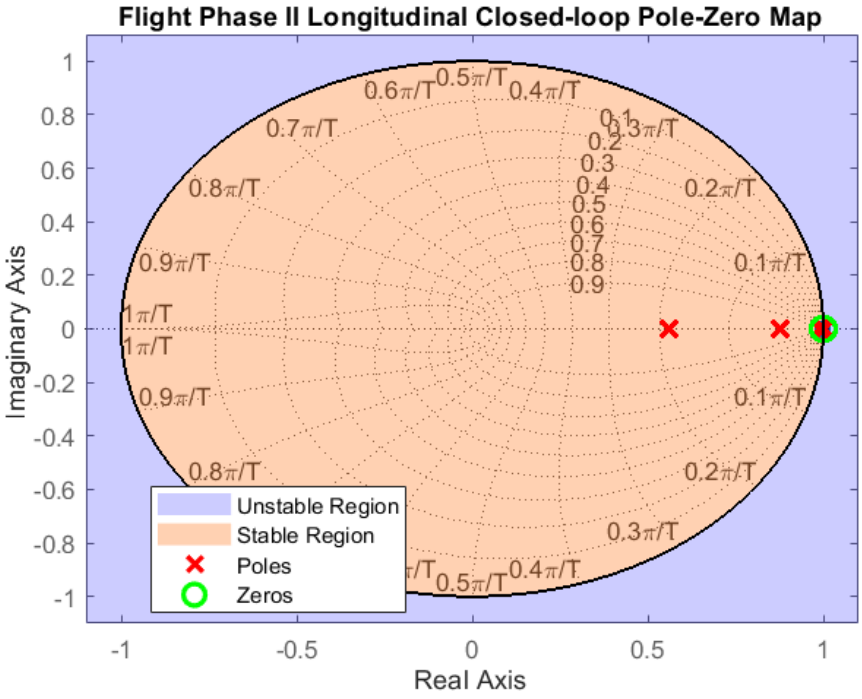


Figure 35: Flight Phase II Longitudinal Dynamics Closed-loop Stability

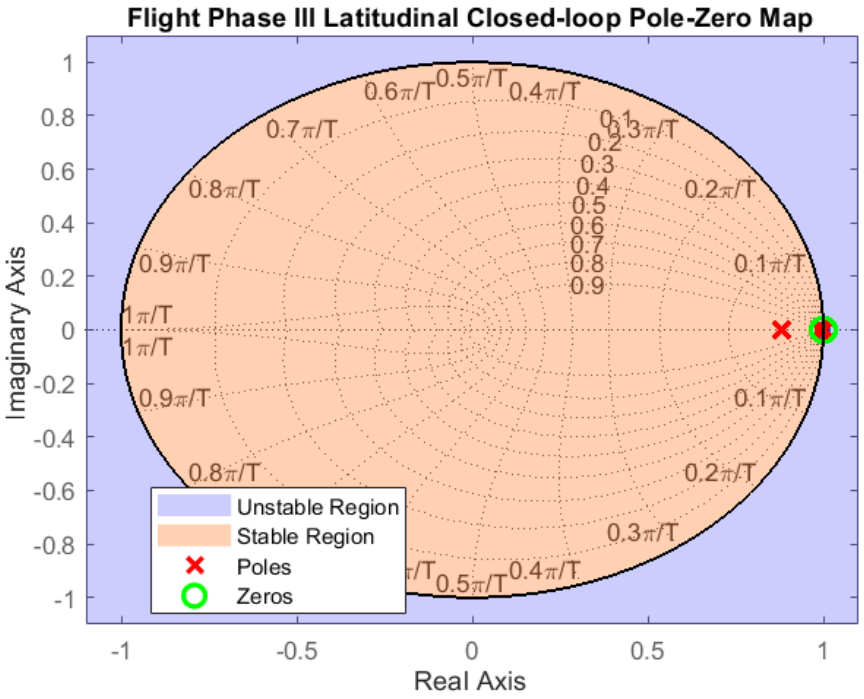


Figure 36: Flight Phase III Latitudinal Dynamics Closed-loop Stability

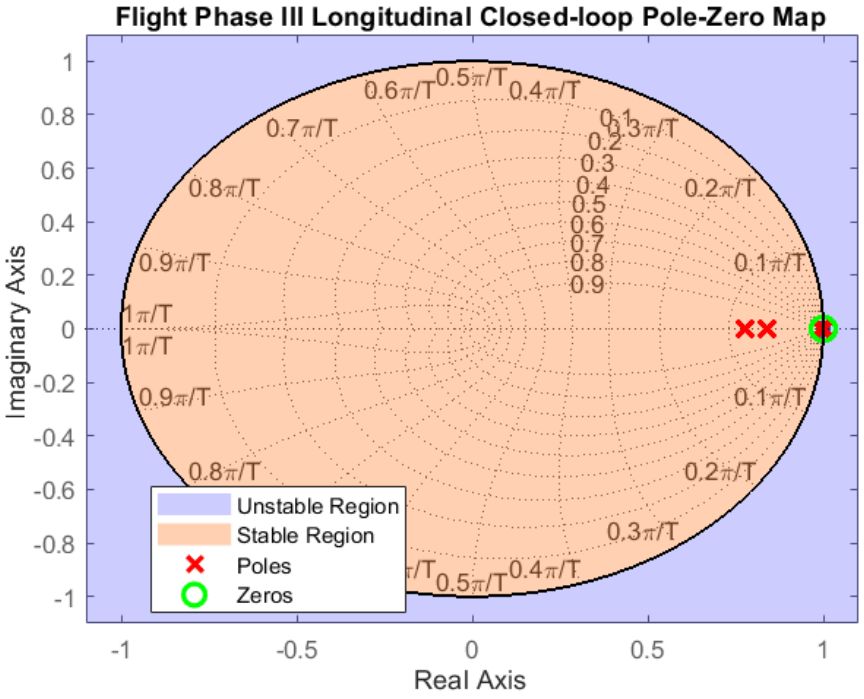


Figure 37: Flight Phase III Longitudinal Dynamics Closed-loop Stability