POLITECNICO DI TORINO

Master's Degree in Automation and Intelligent Cyber-Physical Systems

Master's Degree Thesis

Precise landmark-based Navigation using a Landing Vision System supported by Radar and IMU

Supervisors

Prof. Carlo NOVARA

Ing. Paolo MARTELLA

Candidate

Alberto FOTI

April 2024

"Last time, I asked: 'What does **mathematics** *mean to you?', and some people answered: 'The manipulation of numbers, the manipulation of structures.' And if I had asked what* **music** *means to you, would you have answered: 'The manipulation of notes'?"*

- Serge Lang

Dedications

To my loving family, whose unwavering support and encouragement have been my rock throughout this journey.

Acknowledgments

I would like to express my deepest gratitude to my supervisors for their invaluable guidance, patience, and expertise throughout the course of this research. I am also thankful to my friends of a lifetime, and to my friends and colleagues in Team Diana and in class, who provided encouragement, good vibes and incredible experiences throughout these years.

Abstract

In the aerospace field there is a growing need for autonomous and precise landing capabilities, especially for the future colonization of Moon and Mars. Current operational landing systems don't have strict requirements in terms of landing zone radius. Since they were developed for scientific and exploration purposes, they didn't need precise landing requirements. With the ambitious goals to build martian and lunar bases in the near future, there is the growing need to precisely land human missions and cargo payloads, close to the base, able to be retrieved by a Rover. In addition, precision landing capabilities will be exploited more and more also to recover launchers primary stages, as achieved by SpaceX Falcon 9, with a dramatic positive impact on launch costs.

The Entry, Descent and Landing (EDL) phase of martian and lunar missions is a complex and critical endeavour. It involves navigating a spacecraft through the challenging transition from interplanetary space to the surface of these celestial bodies. Achieving precise landing in often hazardous terrain is vital for mission success.

This master's thesis focuses on the development and implementation of the Landing Vision System (LVS), tailored for lunar and Martian missions. The LVS's primary goal is to provide real-time corrections to the spacecraft's horizontal coordinate estimates. It works in combination with other essential sensors, such as the Inertial Measurement Unit (IMU), which measures accelerations, and the Radar Doppler and Altimeter, providing altitude and speed information in three dimensions. The LVS aims to reduce horizontal position estimate errors in order to achieve a precise pinpoint landing, with a landing precision of less than 100 meters.

This thesis explores the technical aspects of the Landing Vision System, including its underlying principles, design considerations, and practical implementation. It delves into the integration of image processing techniques and data fusion algorithms in the navigation part of the GNC loop, to enable real-time position corrections during the EDL phase.

In a time when martian and lunar exploration is becoming increasingly relevant, the Landing Vision System offers a pragmatic solution to enhance the accuracy and safety of planetary landings.

This thesis is built on the foundation laid by a study promoted by the Italian Space Agency (ASI) called Mars Moon Landing (MML) as part of a research effort on cutting edge technologies for the aerospace field.

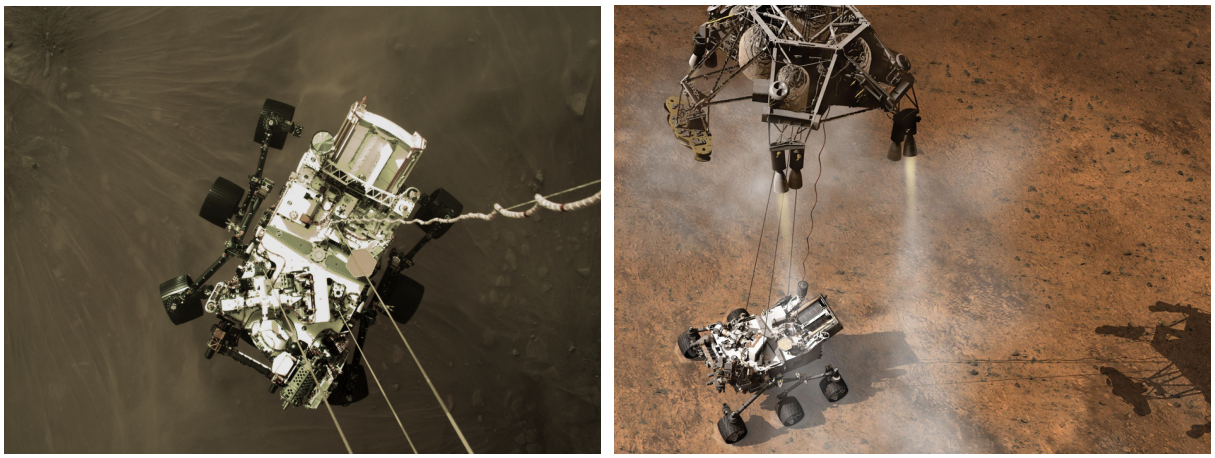


Figure 1: Mars Landing. Reference to MSL mission of JPL.

Contents

Glossary

ALT	Altimeter sensor.
CAT	Catalog for on-board processing and landmark based navigation.
EDL	Entry, Descent and Landing phases of a landing operation on a target planet.
EKF	Extended Kalman Filter.
ENU	East, North, Up variant of the LVLH (Local Vertical Local Horizonal) reference frame.
GNC	Guidance, Navigation and Control loop.
GUI	Guidance, a component of the GNC loop.
IMU	Inertial Measurement Unit sensor.
KF	Kalman Filter.
LVLH	Local Vertical Local Horizonal reference frame.
LVS	Landing Vision System, a camera landmark-based navigation architecture.
MA	Matching Algorithm.
NAV	Navigation, a component of the GNC loop.
RCS	Reaction Control System.
RDA	Radar Doppler Altimeter sensor.
TRN	Terrain Relative Navigation.

Notation and Conventions

Given the reference frames $\mathcal{A}$ and $\mathcal{B}$, then

$$q_B^A$$

is the quaternion representing the orientation of the reference system $\mathcal{B}$ w.r.t. $\mathcal{A}$.

In our context:

$$\mathcal{R}_M \quad \mathcal{R}_B \quad \mathcal{R}_C \quad \mathcal{R}_I$$

$\mathcal{R}_M$: is the Mars/Moon (target planet) ground reference frame (fixed).

$\mathcal{R}_B$: is the Lander body reference frame.

$\mathcal{R}_C$: is the camera reference frame (center in the pinhole point W).

$\mathcal{R}_I$: is the image reference frame (center in the camera center CC, at distance f along the z_c axis of the $\mathcal{R}_C$ reference frame.

The relative orientations and translations are described by the following quaternions/rotation matrices:

$$\mathbf{r}_{MB}^M, \quad \mathbf{q}_B^M, \mathbf{A}_B^M, \quad \mathbf{r}_{BC}^B, \quad \mathbf{q}_C^B, \mathbf{A}_C^B$$

$\mathbf{r}_{MB}^M$: is the position vector of the Body Ref. Frame origin w.r.t. the Mars Ref. frame origin, expressed in the Mars Ref. Frame.

$\mathbf{q}_B^M$, $\mathbf{A}_B^M$: is the orientation of the Body Ref. Frame w.r.t. the Mars Ref. Frame.

$\mathbf{r}_{BC}^B$: is the position vector of the Camera Ref. Frame origin w.r.t the origin of the Body Ref. Frame, expressed in the Body Ref. Frame.

$\mathbf{q}_C^B$, $\mathbf{A}_C^B$: is the orientation of the Camera Ref. Frame w.r.t. the Body Ref. Frame (fixed).

Rotation matrices are direct cosine matrices with unit vectors arranged in columns.

With this convention, to transform a vector expressed in a rotated frame in the original frame, the following transformation is needed:

$$\mathbf{r}^0 = \mathbf{A}_l^0 \mathbf{r}^l$$

LVLH/ENU Reference Frame

The common reference frame is the LVLH (Local Vertical Local Horizontal), in the ENU (East, North, Up) variant. The naming convention for the axis follow the standard for the general LVLH, therefore the following equivalences are valid

$$East\,(E) \quad \rightarrow \quad x$$

$$North\,(N) \quad \rightarrow \quad y$$

$$Up(U) \quad \rightarrow \quad z$$

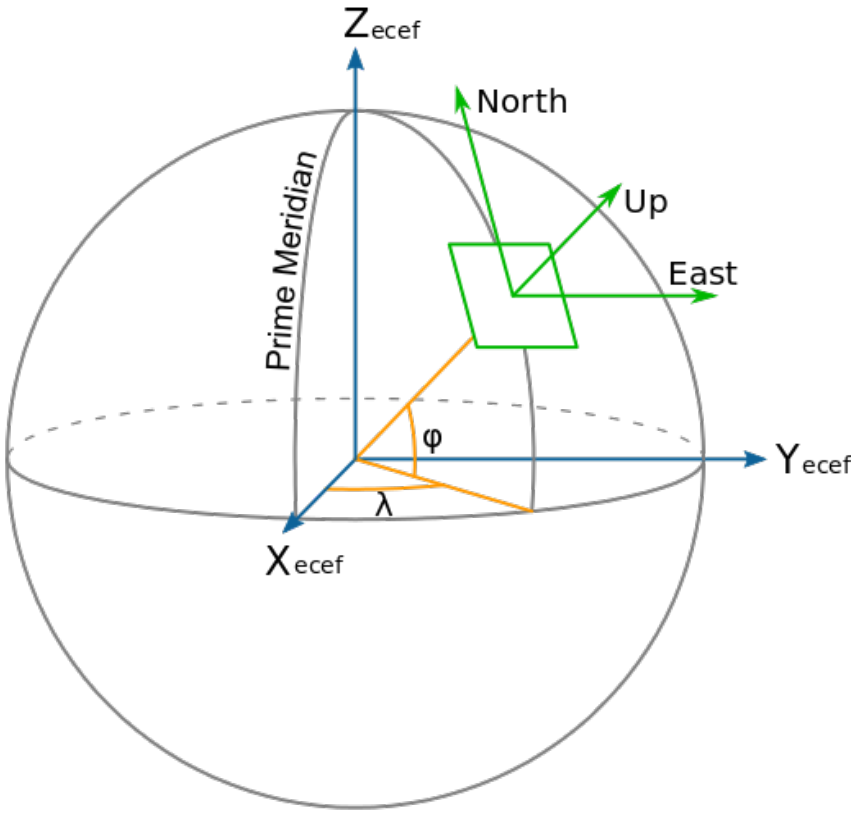


Figure 2: Local Vertical Local Horizontal (LVLH). East, North, Up (ENU) variant.

Further details on other reference frames used can be found in Appendix A.

Part I

Problem Domain Introduction

Chapter 1

Introduction

1.1 Precise Pinpoint Landing

In the context of planetary exploration, precise pinpoint landing refers to the capability of a spacecraft to touch down at a very specific and predefined location on the surface of a celestial body, such as the Moon or Mars, with an extremely high accuracy.

To break this down further, Precise pinpoint landing means that the spacecraft is not merely aiming for a general area or region on the celestial body's surface. Instead, it has a precise, predetermined target or landing site, which could be a scientific point of interest, a previously identified safe zone, or any location of strategic importance for the mission's objectives.

The term "precise" underscores the need for a high level of accuracy in reaching the target. The spacecraft must land as close as possible to the desired spot, minimizing the distance between its actual touchdown point and the intended target. This accuracy is often measured in meters.

Achieving precise pinpoint landing involves minimizing errors in the spacecraft's navigation and guidance systems. Errors can arise due to various factors, such as uncertainties in sensor measurements, variations in atmospheric conditions, or imprecise knowledge of the celestial body's terrain.

The primary motivation behind precise pinpoint landing is safety and mission success. Landing too far from the intended target can lead to a variety of challenges, including difficulties in reaching scientific objectives, increased risks to the spacecraft, and compromised mission goals. Landing with precision enhances the mission's chances of achieving its objectives and reduces the potential hazards associated with unknown or hazardous terrain.

Precise pinpoint landing is particularly valuable for scientific missions. It enables researchers to explore specific geological features, investigate areas of scientific interest, and collect data with precision. It also lays the foundation for human exploration by providing accurate landing sites for cargo and human mission for future lunar or martian bases.

1.2 Entry, Descent, Landing (EDL) phases of a lunar/martian mission

Entry, Descent, and Landing (EDL) phase of a mission [6] constitutes a complex and critical sequence of events that a spacecraft undergoes when arriving near the target planet. This can be broken down into several distinct stages, each serving a crucial purpose.

1.2.1 Entry Phase (if an atmosphere is present, as on Mars)

Atmospheric entry

The EDL phase begins when the spacecraft enters the target planet's atmosphere at a high velocity. At this point, the spacecraft is still in space and typically traveling at a speed of several kilometers per second.

Aerodynamic Deceleration

To slow down, the spacecraft uses the target planet's atmosphere for aerodynamic braking. The friction between the spacecraft and the atmosphere generates intense heat, causing the spacecraft's heat shield to heat up to thousands of degrees Celsius. This heat is absorbed and dissipated by ablative materials on the heat shield, preventing the spacecraft from burning up.

Controlled Entry

During the entry phase, the spacecraft is controlled to follow a specific trajectory, adjusting its orientation and attitude to remain on course. This phase ensures that the spacecraft descends at the correct angle and avoids skipping off the atmosphere.

Controlled Entry determines a modulation of the lift to drag effect, forcing the spacecraft to fly shorter than the natural behavior in case of shallow trajectories or longer than the natural behavior in case of steep trajectories.

To pursue the aim of pipoint landing the spacecraft must be controlled during the Entry phase. Indeed a ballistic entry would accrue an horizontal error w.r.t. the target in the order of decades of km, making it no more recoverable. This large error range is due to the extreme variability of the atmospheric conditions.

1.2.2 Descent Phase

Parachute Deployment and heat shield jettison (if an atmosphere is present, as on Mars)

After the entry phase, the spacecraft continues to slow down. At a lower altitude and velocity, a supersonic parachute is deployed. This parachute further reduces the spacecraft's speed, allowing for a controlled descent.

Once the spacecraft is safely under the parachute, the (front) heat shield is jettisoned, exposing the lander or rover beneath (this event is also referred to as Front Shield Release).

Terrain Relative Navigation (TRN) and Landing Vision System LVS activation

A radar altimeter is activated to measure the spacecraft's altitude above the planet surface. This data is crucial for timing subsequent landing events. TRN systems are used to autonomously compare surface features with pre-loaded maps to adjust the landing site and avoid hazards.

At the first stage, this matching allows to deliver a rough estimate of the initial localization of the spacecraft with respect to the target [2].

This is performed through a coarse matching, delivering a rough estimate of the initial conditions used later by the LVS, that is activated to work together with IMU and the RDA in the navigation.

This localization is fundamental to feed the Guidance module, in charge of the kinematic profile computation for reaching the target and the associated thrust reference command sequence. In detail, a specific task of the guidance module, that runs in "one shot" at this time, is the so-called "Divert Maneuver Planning" in order to find a suitable path to recover the horizontal distance without violating the predefined dynamic, kinematic and propellant consumption constraints.

Once the profile is defined, the fusion of data from the LVS, Radar Doppler Altimeter (RDA), and Inertial Measurement Unit (IMU) within a navigation filter facilitates the real-time determination of the state vector. This integrated approach allows for the continuous tracking of the predefined profile by the guidance module. Subsequently, the control module computes commands based on this updated state vector, ensuring precise and adaptive control throughout the navigation process.

1.2.3 Landing Phase

In this phase, the spacecraft performs a powered descent using rocket engines to further slow down and control its descent. This phase allows for fine adjustments to the landing site based on real-time data from sensors. The spacecraft then gently touches down on the planet surface. This is a critical moment, and the landing systems must ensure that the spacecraft lands safely without tipping over or damaging its payload.

1.3 The role of LVS in the EDL phase

The activation of the LVS is strategically timed within the EDL sequence at a specific altitude or time. It comes into play once a preliminary Terrain Relative Navigation (TRN [1]) algorithm has achieved an initial rough estimate of the spacecraft's position and orientation, at a relatively coarse level of precision. This initial estimate forms the foundation upon which the LVS can start its operations.

The LVS operates as a refinement mechanism, providing real-time data to complement and correct the TRN system's initial assessment. As the spacecraft descends at lower

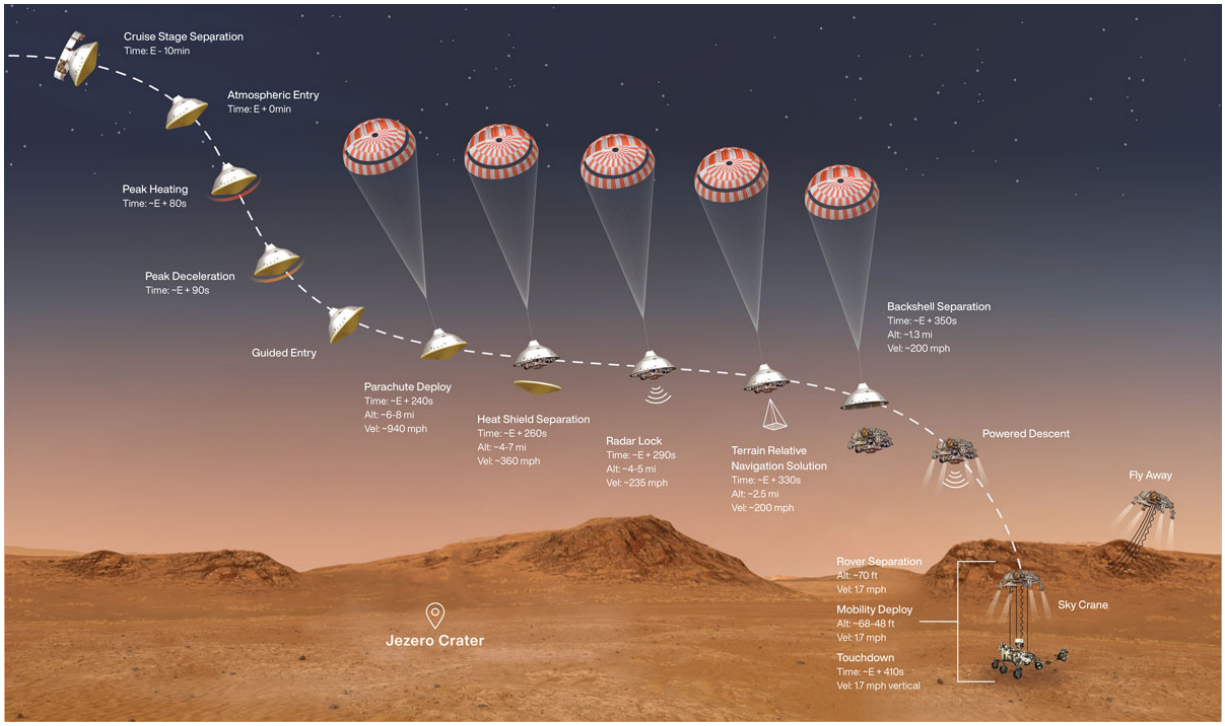


Figure 1.1: Mars EDL phase. Reference to MSL mission of JPL.

altitudes, the LVS leverages advanced computer vision techniques (machine learning algorithms, image processing, ...) to scrutinize and precisely analyze the terrain features below. It identifies key landmarks, evaluates the spacecraft's relative position (and possibly attitude) error, and cross-references this data with pre-loaded maps of the landing site.

1.3.1 Enhanced position estimation accuracy

The LVS fine-tunes the spacecraft's horizontal position estimate, focusing on the x and y coordinates. It identifies and corrects any discrepancies or drift that may have arisen during the descent, bringing the spacecraft closer to the intended landing target. This precision is vital for achieving the mission's scientific and operational objectives, especially in scenarios where the terrain is riddled with hazards or specific geological features of interest.

1.3.2 Terrain Avoidance and Hazard Mitigation

Beyond position correction, the LVS could play a role in identifying potential landing hazards or obstacles, such as boulders, cliffs, or slopes. By promptly detecting these threats, the LVS aids in the decision-making process, allowing the spacecraft to autonomously adjust its descent trajectory, ensuring a safe and secure landing.

This aspect was not analyzed during the development work for this thesis.

1.3.3 Key take away

The Landing Vision System represents a critical layer of intelligence in the EDL phase, acting as the spacecraft's "eyes" during its descent onto the Martian surface. By refining position estimates, facilitating precise pinpoint landings, and enhancing hazard avoidance, the LVS significantly contributes to the overall success and safety of Martian missions.

1.4 Objectives and contributions

The primary objective of this thesis is to address the critical challenges associated with precise pinpoint landing on Mars/Moon by developing a comprehensive Landing Vision System (LVS) that seamlessly integrates within the navigation architecture of the GNC during the Entry, Descent and Landing (EDL) phase. This endeavor comprises several key components:

1.4.1 Landmark Generation

Firstly, the development of a landmark generation tool capable of producing catalogs that emulate real data. These catalogs enable more precise simulations, and in a real mission are going to be developed starting from real target planet terrain data of the landing site, and loaded in the spacecraft's mainboard computer to run the LVS algorithm during the EDL phase.

1.4.2 Camera Model

A camera model based on the pinhole model is constructed within the simulation framework. This model provides landmarks in the Mars ground reference frame visible to the camera. This model was already implemented and was the starting point for understanding the LVS approach. It's adapted to be integrated better in the overall LVS architecture.

1.4.3 LVS Preprocessing

Next, a preprocessing algorithm is devised to match onboard memory landmarks with real-time sensor data captured by the camera. This algorithm will not only perform landmark matching but also enable precise position determination and correction, adding a new source of data for the navigation data fusion algorithm.

1.4.4 Navigation Data Fusion

Data coming from the LVS, IMU, and RDA (or simple altimeter) are fused together in the filtering stage of the navigation. A key focus is on the implementation of a Kalman filter to filter and perform data fusion from these diverse sensors. A particular emphasis is placed on addressing the challenges arising from different data rates among these sensors, ensuring a robust and reliable spacecraft position estimation.

Indeed time-switch criterion can determine the involvement of a second catalog at quite different altitudes in various mission realizations, as a consequence of the guidance profile variability during the divert maneuver phase.

Prior to this thesis work, a filter was already present, and integrated well the IMU, altimeter and RDA sensors for the rotational and translational dynamics estimation.

The kalman filter designed for this thesis replaces the translational filter, integrating the data coming from the LVS and the other sensors (IMU, altimeter, RDA).

The rotational filter, on the other hand, was not subject to any changes.

1.4.5 Results Analysis

Lastly, this thesis aims to conduct a comprehensive analysis of the results obtained. It will investigate whether the combination of the Radar Doppler Altimeter and the Landing Vision System is sufficient for achieving precise pinpoint landings on a target planet or whether the presence of a conventional radar system is necessary for additional support. This critical question will be addressed through experimentation, providing insights into the efficacy of the integrated system.

Part II

Landing Vision System (LVS) model

Chapter 2

Introduction to the LVS in the landing scenario

2.1 Introduction

The LVS uses on-board catalogs to match against real-time data coming from one or multiple cameras. Landmarks are identified and an algorithm is responsible to match input landmarks information and the landmarks data stored in the on-board computer memory.

From the matching procedure, a corrected estimate of position (x, y) is extrapolated. It's also possible to calculate a correction for the altitude (z) and the attitude estimate, particularly the "yaw" angle around the vertical z axis (in ground LVLH reference frame coordinates).

This information is then used in the following steps of the navigation architecture to gain more information about the overall system state, such as the horizontal velocity (in ground LVLH reference frame).

It is worth highlighting which are the implications required for the estimation of the various states. A camera placed above the site, and with a limited off-vertical displacement, is intuitively effective to estimate the horizontal coordinates such as the rotation around the local vertical direction but less effective in recovering the rotations around axes parallel to terrain. Analogously it is quite immediate to recognize the difficulty in using landmark information from a singular camera to estimate the altitude. For this scope, in particular, a stereoscopic system of at least 2 cameras would be more suitable to guarantee the needed signal-to-noise ratio.

2.2 General considerations

The following are a series of practical and technical considerations, working as requirements and premises for the proper operation of the LVS stack.

Catalogs

A catalog (or a series of catalogs) of the general landing area (5km) is needed for the LVS to work, since the operation of the LVS is based on the difference between the expected and the actual landmark pose.

Resolution range at varying altitudes

The LVS needs to be active from altitudes ranging from 3000m to 500m or less, during the landing phase. With such a wide range of altitudes, multiple catalogs at different resolutions are required for the on-board landmark matching algorithm (change of resolution), in order to achieve proper performance.

Real catalogs generation

Catalogs are taken for a particular range of hours during the day and visibility conditions, taking into account the estimated arrival time (shadows on craters, light glares, etc.. can affect LVS performances significantly) [7]. A significant drop in matching quality is expected if these assumptions are not met.

In the next few sections, I'm going to briefly describe the three main components of the LVS architecture. An in depth detailed dive into each one of these components and the work behind them will be dedicated in the next chapters.

2.3 Catalog generation

To setup the simulation infrastructure, a catalog generation step is necessary. In the real setting, catalogs are going to be created using real target planet data, mapping craters, significantly big rocks and other landmarks.

Camera Model is going to be fed with a full catalog (representing the real terrain) where landmarks at different resolutions (ranging from 200m radius to 10m radius) are mixed in the same big map.

This is just for emulation purposes, and in the real mission no catalog is going to be provided, as the camera is going to see and process the real target planet features, in order to extract landmark information.

On-board catalogs

The LVS Processing algorithm is going to be implemented in the on-board software, together with a set of catalogs (two was found to be the optimal sweet spot) with landmarks

at different resolutions. A catalog switch is performed during descent in the landing phase, to increase performance of the matching algorithm.

1. Close landmarks can affect the matching algorithm performances, so catalogs should not be too dense (less landmarks but well positioned is better than lots of landmarks really close or overlapping). In the simulation environment, the two catalogs going in the LVS main algorithm are taken from the full catalog in the following steps:
   (a) Extraction of the landmarks corresponding to the required resolution (150-70m radius landmarks for the first, and 40-10m radius landmarks for the second).
   (b) Filtering process, by not considering landmarks that are overlapping or too close to each other.
2. Catalogs are loaded in the on-board computer memory, max 500-600 landmarks per catalog, and maximum 2 catalogs.

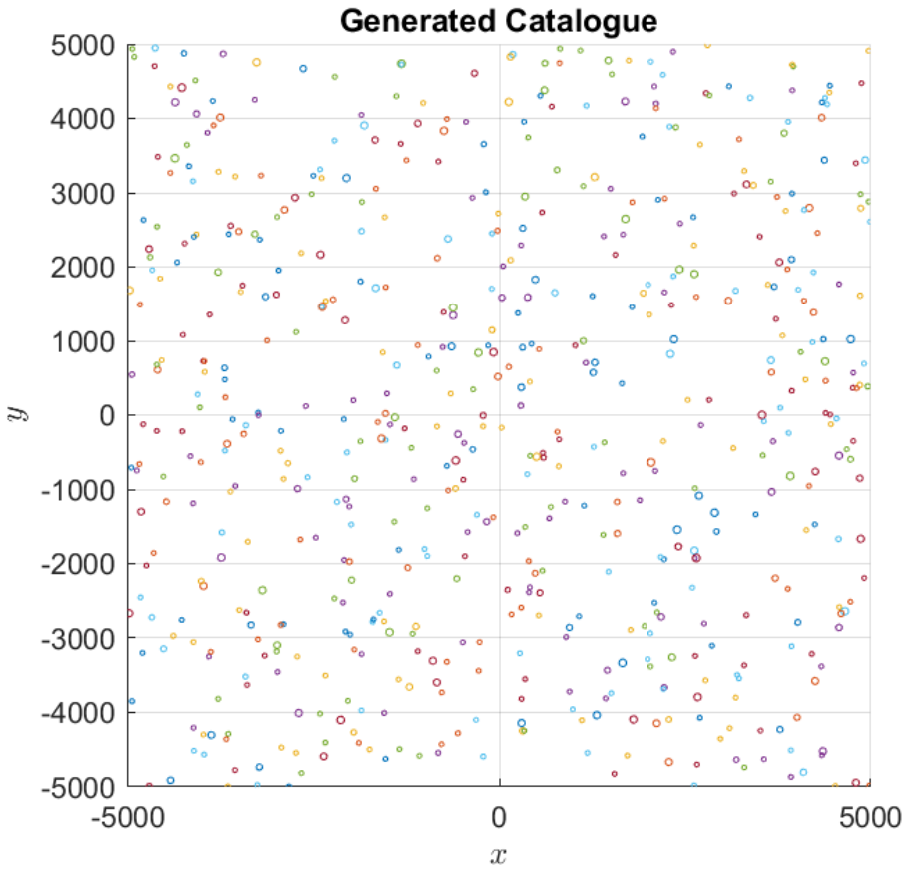


Figure 2.1: Example Catalog

2.4 Camera model

The camera model is the first component of the LVS architecture, leveraging the generated catalogs (look at the previous section) to emulate the real behaviour of a camera.

The model block is fed directly with the real full catalog (unfiltered) and the real position and attitude of the spacecraft during the simulations.

Camera settings, including positioning, orientation, focal aperture, sensor screen dimensions, and other parameters, are configurable within the model.

Furthermore, the camera model employs the pinhole model to emulate the input-output behavior of the actual camera.

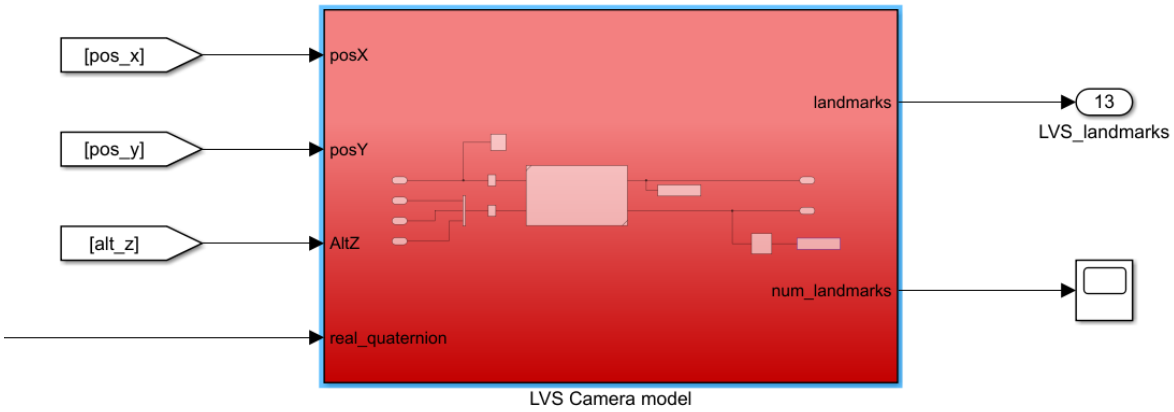


Figure 2.2: Camera Model block diagram (SIMULINK)

2.5 Processing LVS algorithm (embedded)

The landmark locations delivered by the upstream camera model are used to improve the state estimate (specifically x and y position coordinates, but more state variables can be estimated, with varying degrees of reliability and accuracy).

The inputs to the processing block are the on-board filtered catalogs (CAT1 and CAT2), the current state estimate coming from the navigation filter and the landmark data coming from the camera model.

The output of the LVS processing steps is the corrected state estimate $\tilde{x}$.

- Landmark matching is the first step, and is a procedure aiming at associating each landmark seen by the camera with a landmark present in the catalog.
- Position determination phase builds on top of the landmark matching step, in order to produce a corrected state estimate.

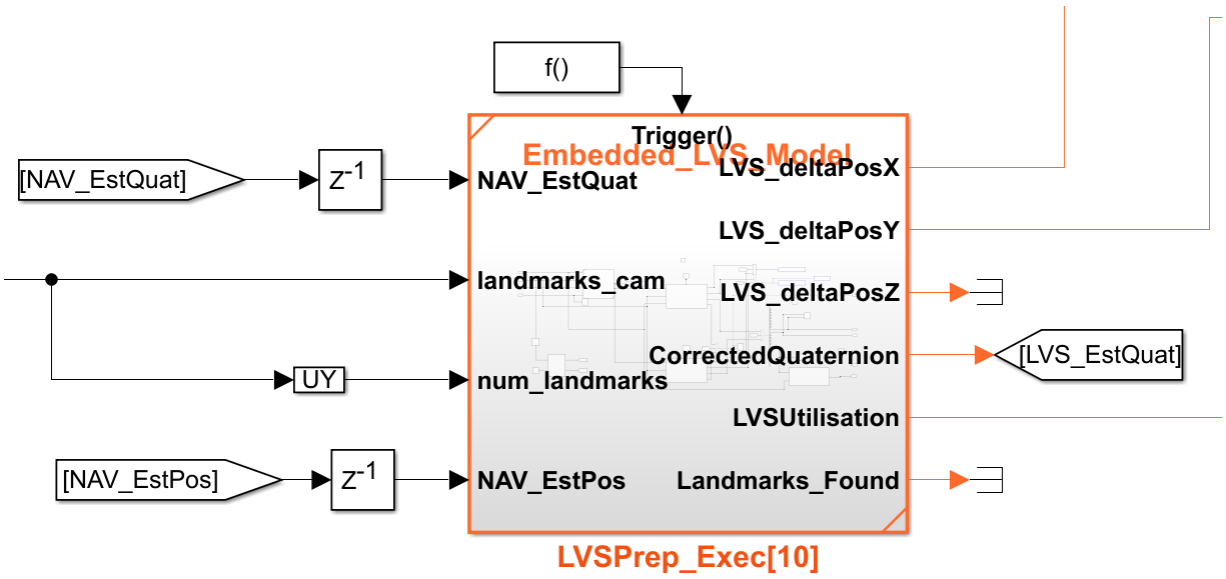


Figure 2.3: LVS preprocessing block diagram (SIMULINK)

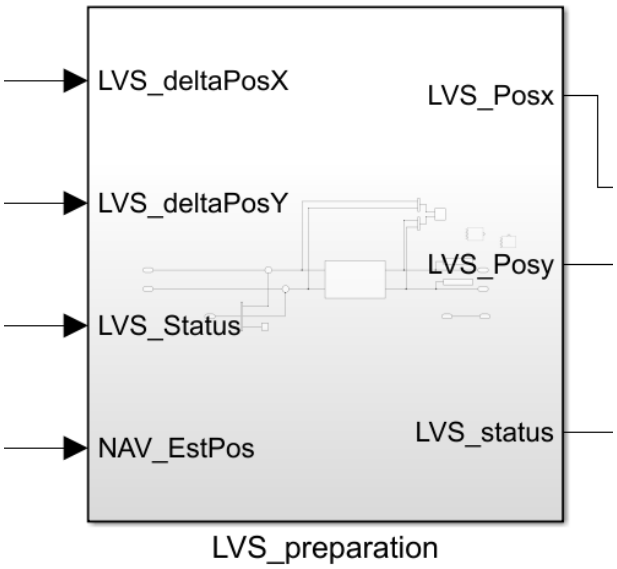


Figure 2.4: LVS preparation for navigation filter (SIMULINK)

Chapter 3

Landmark Catalog Processing

3.1 Introduction

In this section, the focus is on the Landmark Catalog generation and processing. This step is fundamental, since various components in the architecture need landmark data.

For instance, the Camera Model is a mathematical representation of the real Camera Sensor for the LVS, hence a full catalog is required, representing a realistic martian or lunar terrain.

To generate this catalog landmarks are taken at widely different dimensions (resolution), mixing landmarks with radii in the range 10-40m and landmarks with 70-150m.

The Camera Model is fed with this catalog and, depending on the altitude of the landing spacecraft, is going to be able to identify a subset of these landmarks.

The LVS embedded preprocessing module needs the landmark locations data to perform the Matching Algorithm (MA), therefore catalogs need to be stored on-board. It uses two other catalogs (CAT1 and CAT2) obtained from the full catalog by extracting landmarks with specific size. This was done to increase performance at specific altitude ranges (3000-1200m for CAT1, 1200-300m for CAT2).

After the extraction, these landmarks go through a filtering process to improve the performance of the matching algorithm. The performance of the landmark matching is hugely decreased if there are close or overlapping landmarks, so this filtering procedure is crucial. This is accomplished by ignoring overlapping or too close landmarks, thus collecting a sparse set of landmarks to be stored in the two on-board catalogs.

The filtering objective is to obtain catalogs:

1. with landmarks at reciprocal distances larger than the inherent error expected at the working altitude where they are applied

2. with a complete coverage of the area potentially interested by the landing such that, for whichever pose of the camera, a sufficient number of objects to be matched could be found.

3.2 Landmark Modeling

A landmark is a feature on the planet surface that can be a crater, a rock or other marker, depending on the camera image resolution.

3.2.1 Landmark on the Planet

Center and radius

Each landmark has a center (x, y, z) and a radius (r) forming a circle in the planet surface.

Normal inclination

Each landmark has a normal inclination (θ_x, θ_y) resulting in a tilt of the circle in the planet surface. This information is not used in this thesis but can be exploited to get a better position estimate.

3.2.2 Landmark in Camera view

Center and semi-axes major and minor

The tilted circle in the planet surface, through the pinhole model, in general, results in an ellipse in the camera sensor reference frame, given by its center, semi-axes minor and major $(c_c, c_y, \psi_1, \psi_2, \psi_3)$

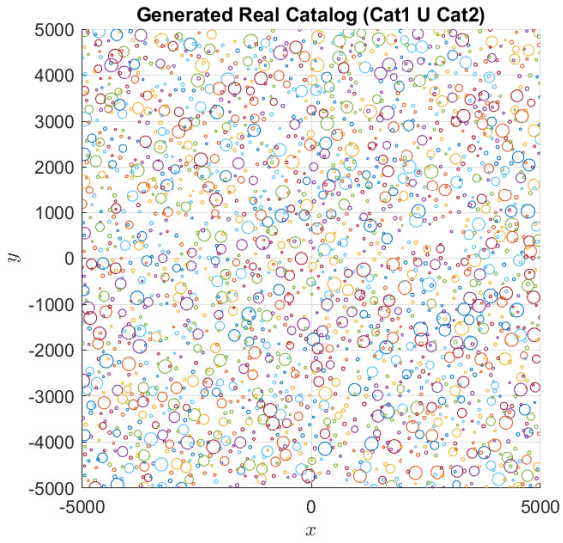


Figure 3.1: Generated real catalog (full version for camera model)

3.3 Catalog Characteristics

During descent, in the landing phase, the altitude changes rapidly and the range of altitudes is wide.

For this reason, multiple catalogs at different resolutions are needed. During the mission, when a certain altitude or time is reached, a switch of catalogs is performed in order to improve the LVS performance in the matching algorithm.

3.3.1 Time or Altitude for catalog switch trigger

The trigger for catalog switch can be a specific time (ex. 25 seconds after the beginning of the EDL phase), or a specific altitude (ex. catalog switch is triggered when the 1000m from ground altitude is reached).

A specific altitude is more precise, since the rationale behind catalog change is to adapt better to the change of landscape during the descent, and landmarks have a huge range of sizes.

3.3.2 Correct number of catalogs

By performing simulations, having two catalogs was found to be the optimal case, a trade-off between memory space and performance.

3.3.3 Camera Model Catalog

The camera model, for simulation purposes, is using a more complex catalog, obtained by the union of the two catalogs taken at different resolutions. This allows for a more realistic setting, in which the camera has a smooth transition instead of a perfect switch of catalogs in sync with the LVS main algorithm.

Note. The maximum number of landmarks found by the camera, in a real scenario, is set to 100 for computational complexity reasons. 100 landmarks are more than enough to perform the matching algorithm, adding more enters in a plateau of diminishing returns.

3.4 Catalog Generation

Catalogs generation is performed on two different maps, one for each catalog to be generated.

3.4.1 First Catalog (CAT1)

$$Number\ of\ landmarks\ :\ 600$$

$$Radius\ range\ :\ 70 - 150m$$

$$width\ =\ 5000m,\ height\ =\ 5000m$$

3.4.2 Second Catalog (CAT2)

$$Number\ of\ landmarks\ :\ 1200$$

$$Radius\ range\ :\ 10 - 40m$$

$$width\ =\ 5000m,\ height\ =\ 5000m$$

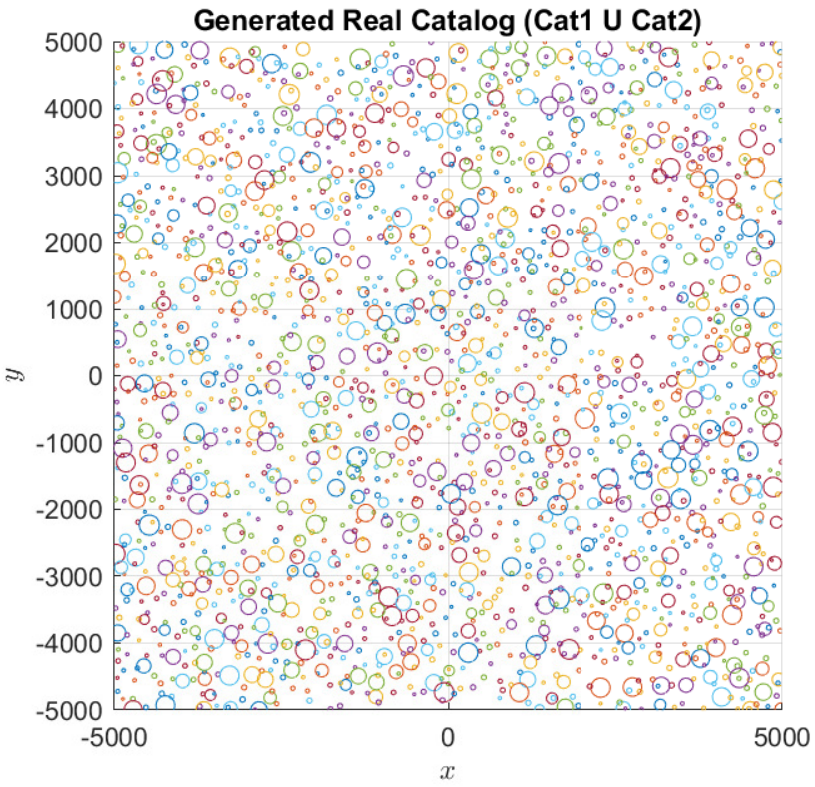
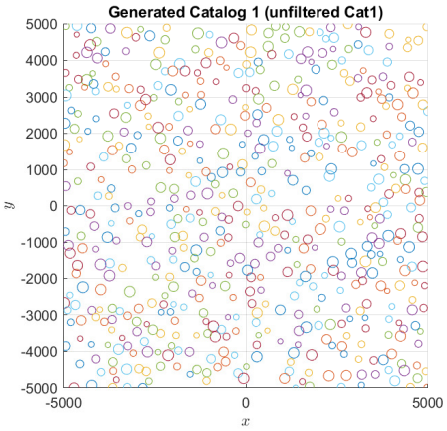


Figure 3.2: Generated real catalog (full version for camera model)

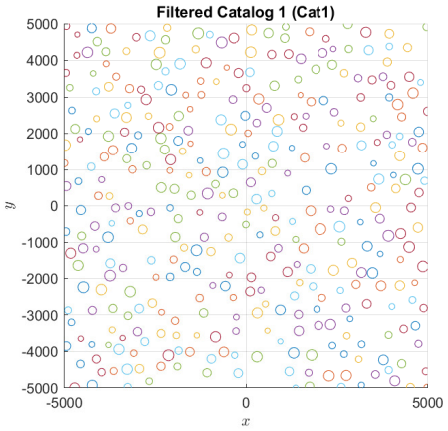
3.5 Catalog filtering

While the raw real catalog will be fed to the camera model, a filtering procedure is needed for the on-board catalogs. Filtering is performed on the raw catalogs, to be able to give a clean stable constellation of landmarks to the matching algorithm in the LVS embedded model.

These catalogs will be loaded in the on-board computer memory.

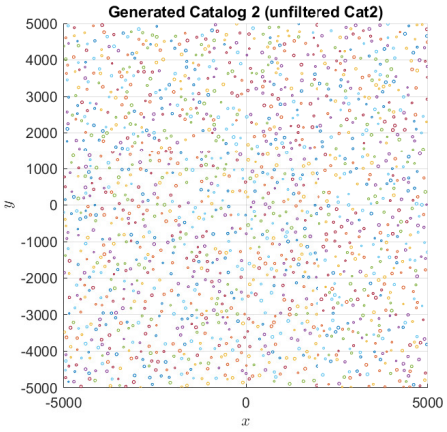


(a) CAT1 unfiltered

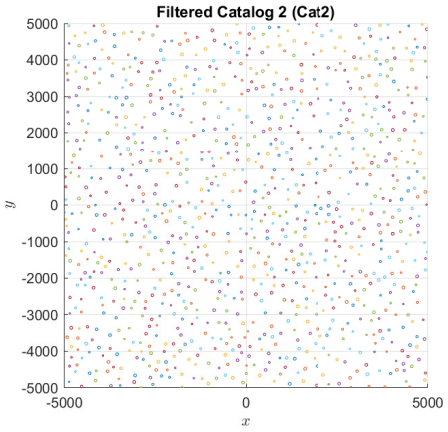


(b) CAT1 filtered

Figure 3.3: Catalog 1



(a) CAT2 unfiltered



(b) CAT2 filtered

Figure 3.4: Catalog 2

3.5.1 Minimum distance between landmark centers

Particular focus is on the minimum distance between landmarks, in order to avoid too close landmarks. If not properly tuned, can significantly affect algorithm performance.

A gridding analysis was performed on this distance value. It's important to note that this value is not crucial, but the relationship with the matching algorithm used in the LVS processing steps. Since different matching techniques are going to have different requirements for this value, the analysis is done in the chapter about LVS preprocessing.

3.5.2 Overlapping avoidance

Landmark radii are used in the filtering process to remove overlapping landmarks. The need for overlapping landmarks filtering is dependent on the requirements of the matching algorithm, as in the minimum distance between landmarks seen in the previous section.

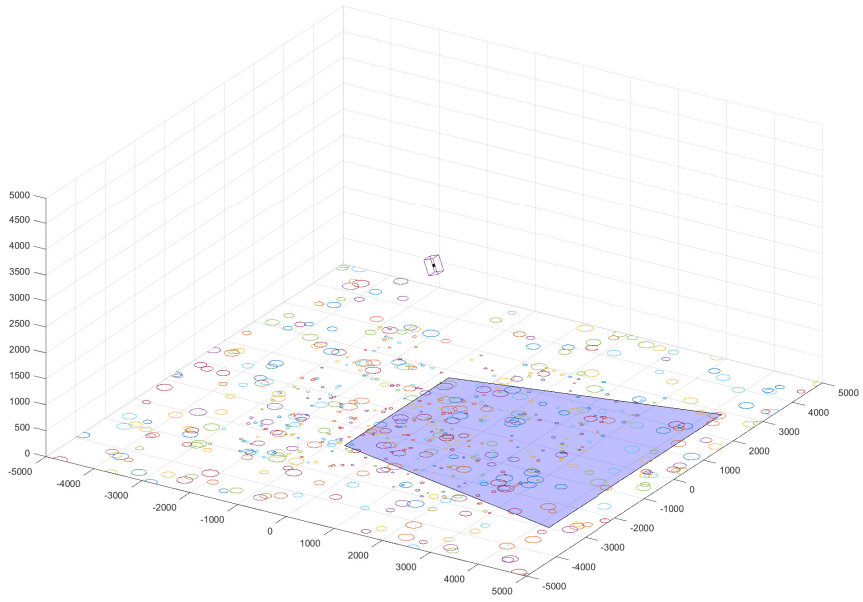


Figure 3.5: Landing terrain view. Camera visibility region is marked with a blue color area

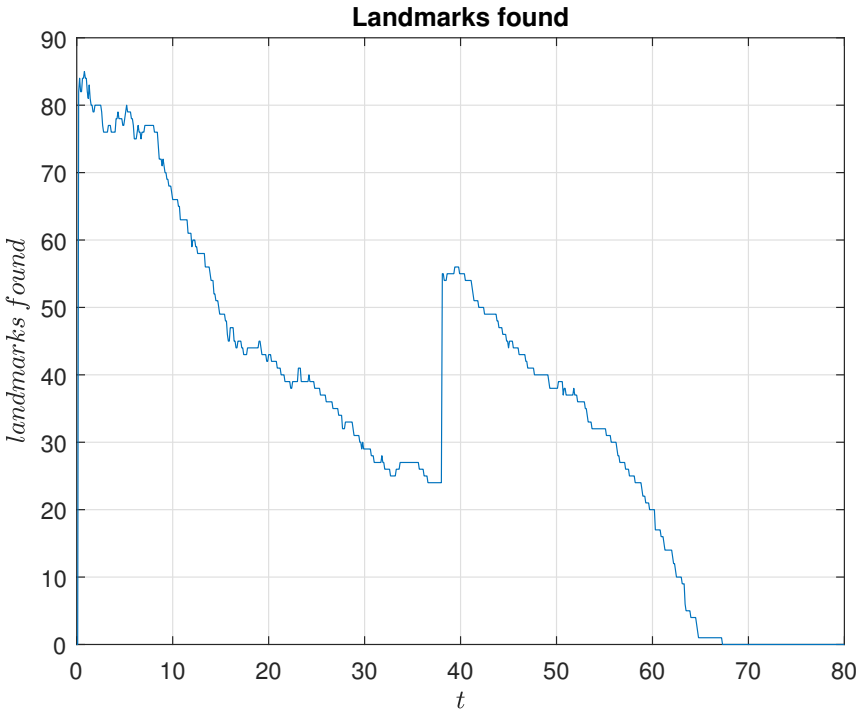


Figure 3.6: Landmarks found during a landing simulation (x axis: time, y axis: landmarks found count)

Chapter 4

LVS Camera Sensor Model

4.1 Objective

The objective of this component is to simulate the output that a real camera is able to provide during the landing phase.

A collection of landmark is extrapolated from the raw image and processed using image processing techniques of feature extraction and landmark identification.

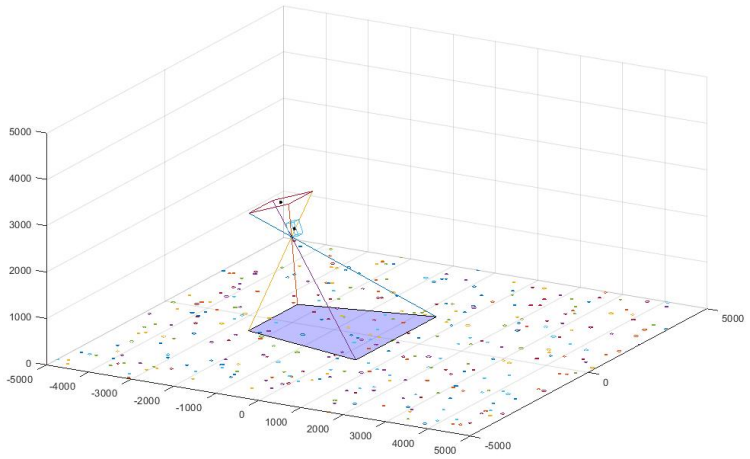
To simulate this behavior in the context of the design of an effective GNC loop, it's sufficient to emulate the input-output response desired from the camera.

In order to achieve this, a suitable realistic map of an hypothetical target planet terrain is provided to the camera (see previous chapter for more details on catalog generation), together with the real position and attitude coming from the simulated system.

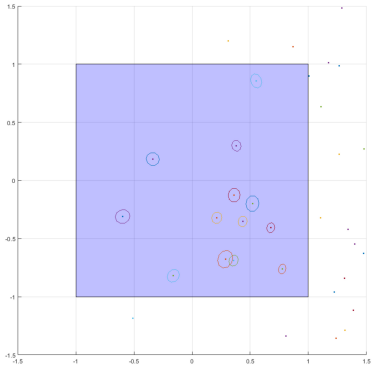
Using a pinhole model, it's possible to then retrieve the location of the landmarks in the image space during the landing phase, and feed this data to the downstream component of the LVS architecture for further processing.

4.2 Pinhole Model

The pinhole camera model describes the mathematical relationship between the coordinates of a point in three-dimensional space and its projection onto the image plane (camera sensor) of an ideal pinhole camera, where the camera aperture is described as a point and no lenses are used to focus light.



(a) Lander view of the target planet terrain



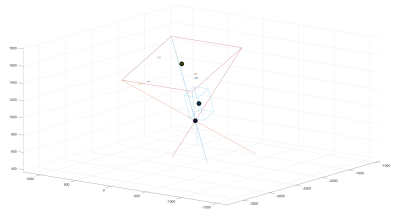
(b) Camera View. Projected visible landmarks are displayed as the camera sees them

4.3 Practical Considerations

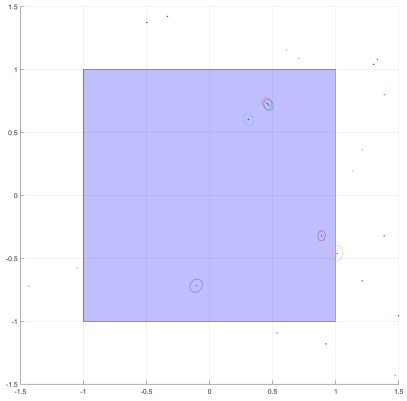
4.3.1 Non-idealities: Distortions, blurring

The pinhole model does not include, for example, geometric distortions or blurring of unfocused objects caused by lenses and finite sized apertures. It also does not take into account that most practical cameras have only discrete image coordinates. This means that the pinhole camera model can only be used as a first order approximation of the mapping from a 3D scene to a 2D image. Its validity depends on the quality of the camera and, in general, decreases from the center of the image to the edges as lens distortion effects increase.

Some of the effects that the pinhole camera model does not take into account can be compensated, for example by applying suitable coordinate transformations on the image coordinates; other effects are sufficiently small to be neglected if a high quality camera is used. This means that the pinhole camera model often can be used as a reasonable description of how a camera depicts a 3D scene.



(a) Pinhole model visualization (fake camera specifications are used for visualization purposes)



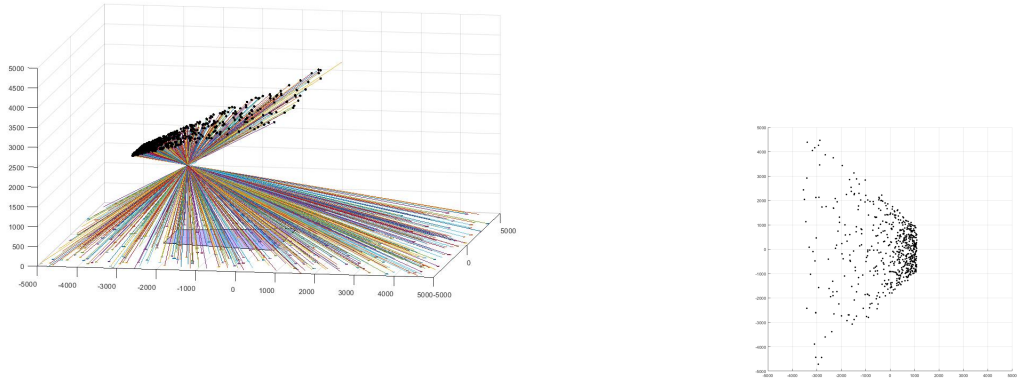
(b) Camera View. Projected visible landmarks are displayed as the camera sees them

4.4 Camera Characteristics

The useful camera characteristics for the pinhole model are the following

w: width of the camera display sensor (11mm)
h: height of the camera display sensor (11mm)
f: focal length (Pinhole center point to camera sensor) (8mm)

(i) Real camera specifications are going to vary, with a tolerance margin, w.r.t. the data presented here.



(a) Landmark Projection on the camera sensor plane

(b) Landmark Projections on the camera sensor plane, in image space ref.Frame

Figure 4.3: Landmark projection

4.5 Pinhole Model

The objective of the camera model is to associate, for each visible landmark (on the Mars ground), the corresponding projection (in the image reference frame) of the beam starting from the landmark on the ground and passing through the pinhole center.

If a landmark center, projected onto the image plane, is not inside the area determined by the sensor display's width and height, then it's not visible.

4.5.1 Geometric approach

The geometric approach uses a combination of Linear Algebra and geometric intuitions to approach the pinhole problem formulation. It's more intuitive and gives more flexibility in showing intermediate steps for simulation visualization.

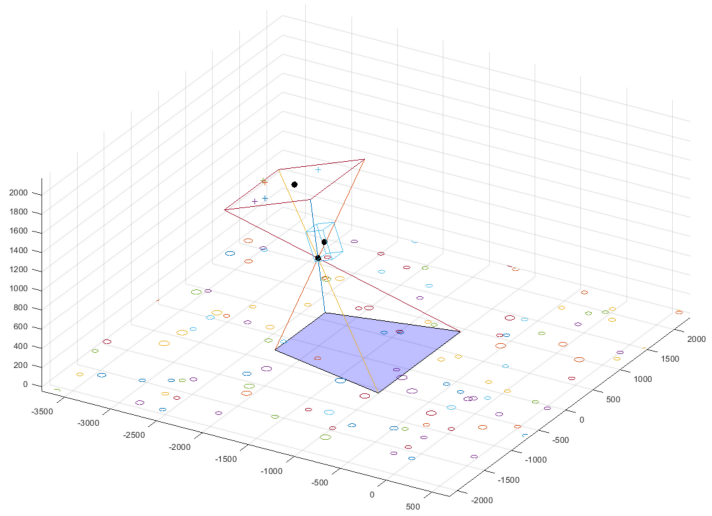


Figure 4.4: Landmark projection in camera lens visualization

Main pinhole parameters calculation

Starting by defining some components of the pinhole system:

W: Pinhole center point
CC: Camera (sensor) center point
$\mathbf{C}_{pinhole}$: 4 corner points of the CCD, in the Mars Ref. Frame

It's possible to retrieve this points thanks to the following transformations:

Given the vector $\mathbf{r}_{BC}^{B}$ going from the center of the body frame (origin of $\mathcal{R}_B$) to the camera pinhole center ($\mathbf{W}$ = origin of $\mathcal{R}_C$), expressed in $\mathcal{R}_B$, and the rotation matrix $\mathbf{A}_{B}^{M}$ from the planet fixed ref.Frame ($\mathcal{R}_M$) to $\mathcal{R}_B$, it's possible to retrieve the vector body-camera expressed in $\mathcal{R}_M$ as

$$\mathbf{r}_{BC}^{M} = \mathbf{A}_{B}^{M}\mathbf{r}_{BC}^{B}$$

From this relation it's easy to get the camera pinhole center $\mathbf{W}$ as

$$\mathbf{W} = \mathbf{r}_{BC}^{M} + \mathbf{r}_{MB}^{M}$$

Given $\mathbf{W}$, the rotation matrix $\mathbf{A}_{C}^{B}$ from $\mathcal{R}_B$ to the camera ref.Frame $\mathcal{R}_C$, and the vector $\mathbf{f} = \begin{bmatrix} 0 & 0 & f \end{bmatrix}^t$ going from $\mathbf{W}$ to the camera sensor center $\mathbf{CC}$ along the $\mathbf{z}$ axis of $\mathcal{R}_C$, it's possible to get the coordinates of the camera sensor center $\mathbf{CC}$ w.r.t the origin of $\mathcal{R}_M$, expressed in the $\mathcal{R}_M$ ref.Frame

$$\mathbf{CC} = \mathbf{A}_{C}^{M}\begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix} + \mathbf{W}$$

where $\mathbf{A}_C^M = \mathbf{A}_B^M \mathbf{A}_C^B$ is the orientation of $\mathcal{R}_C$ w.r.t. $\mathcal{R}_M$.

The 4 vertexes of the camera sensor are calculated in the following way

$$\begin{cases} C(1,:) = \mathbf{A}_C^M \begin{bmatrix} -w/2 & -h/2 & 0 \end{bmatrix}^t \\ C(2,:) = \mathbf{A}_C^M \begin{bmatrix} w/2 & -h/2 & 0 \end{bmatrix}^t \\ C(3,:) = \mathbf{A}_C^M \begin{bmatrix} w/2 & h/2 & 0 \end{bmatrix}^t \\ C(4,:) = \mathbf{A}_C^M \begin{bmatrix} -w/2 & h/2 & 0 \end{bmatrix}^t \end{cases}$$

```
% Camera configuration
w = 0.011;
h = 0.011;
f = 0.008;
cam_cfg = [w, h, f];

% PINHOLE function
[W, CC, C_ph] = pinhole(cam_cfg, r_MB_M, q_MB, r_BC_B, q_BC);
```

Listing 4.1: Pinhole camera model

Camera projections

Then, for each landmark in $\mathcal{R}_M$ coordinates, project it from the ground to $\mathcal{R}_C$ ref.Frame relative to Camera Center Point **CC**, in the image ref.Frame $\mathcal{R}_I$.

Given a landmark of coordinates **A** in $\mathcal{R}_M$

```
% CAMERA_INTERSECT function
p1 = camera_intersect(A, W, C_pinhole)
```

Listing 4.2: Camera screen sensor intersection of landmark A, through pinhole point W

where $\mathbf{p}_1$ is the landmark **A** projected onto the camera sensor, through the pinhole point **W**.

$$\mathbf{p}_2^M = \mathbf{p}_1^M - \mathbf{CC}$$

where $\mathbf{p}_2$ is the projection of the landmark **A** w.r.t. the camera sensor center **CC**, in $\mathcal{R}_M$ coordinates.
To get the coordinates in the $\mathcal{R}_I$ reference frame the following transformation is allied

$$\mathbf{p}_3^I = \mathbf{A}_M^C \mathbf{p}_2^M$$

where $\mathbf{A}_M^C$ is

$$\mathbf{A}_M^C = (\mathbf{A}_C^M)^{-1} = (\mathbf{A}_C^M)^t$$

```
% CAMERA_PROJECTION function
[p1, p3] = camera_projection(A', W, C_pinhole, CC, r_BM,
   q_BM, q_CB);
```

Listing 4.3: Camera screen sensor intersection of landmark A, through pinhole point W, in target planet ref.Frame (p_1) and in image plane ref.Frame (p_3)

Strategy to evaluate if a landmark is observable by the camera

The landmark projection in the camera in inside the camera sensor's display when considering the geometric intuition that if a point is internal, the sum of angles formed between pairs of corner points of the camera equals $2 \cdot pi$.

```
% IS_INSIDE_CAMERA_SENSOR function
function [res] = is_inside_camera_sensor(P, V)
    res = false;
    theta_tot = 0.0;

    X = V(1,:) - P;
    X = X / norm(X, 2);
    Y = V(2,:) - P;
    Y = Y / norm(Y, 2);
    theta_tot = theta_tot + acos(dot(X, Y));

    X = V(2,:) - P;
    X = X / norm(X, 2);
    Y = V(3,:) - P;
    Y = Y / norm(Y, 2);
    theta_tot = theta_tot + acos(dot(X, Y));

    X = V(3,:) - P;
    X = X / norm(X, 2);
    Y = V(4,:) - P;
    Y = Y / norm(Y, 2);
    theta_tot = theta_tot + acos(dot(X, Y));

    X = V(4,:) - P;
    X = X / norm(X, 2);
    Y = V(1,:) - P;
    Y = Y / norm(Y, 2);
    theta_tot = theta_tot + acos(dot(X, Y));

    if abs(theta_tot - 2*pi) < 0.1
        res = true;
    end
```

Listing 4.4: Procedure to check if a landmark is inside the camera sensor (hence, visible)

4.5.2 Implementation of the camera model

```
%%-------CAMERA MODEL-----------
%LVS (pinhole model)
[W, CC, C_pinhole, ~] = pinhole(w, h, actualf, a, r_MB_M,
   q_MB, r_BC_B, q_BC);
%Data
camera_center_landmarks = zeros(3, Cat.num_landmarks);

% Threshold for camera model landmark identification
cat_threshold = 50;

%For all landmarks
for j=1:Cat.num_landmarks
    A = [Cat.clandmarkX(j); Cat.clandmarkY(j);
       Cat.clandmarkZ(j)];

    %Landmark big enough to be seen at the current altitude
    if r_MB_M(3) > 1000
        if Cat.rlandmark(j) < cat_threshold
            continue;
        end
    else
        if Cat.rlandmark(j) > cat_threshold
            continue;
        end
    end

    %Landmark center projection in camera
    [k1, k] = camera_projection(A', W, C_pinhole, CC, r_MB_M,
       q_MB, q_BC);

    %Criterion of angles to decide if a landmark is seen of
       not.  If landmark is internal the sum of angles is 2*pi
    if is_inside_camera_lens(k1', C_pinhole)
        found_landmark = found_landmark + 1;

        camera_center_landmarks(:, found_landmark) = k';

        if found_landmark == max_landmark
            break;
        end

    end

end % closes the for on all landmarks
```

Listing 4.5: Camera Model implementation

Landmark resolution threshold

The camera model is fed directly with the full real catalog, and the following LVS stages use two separate catalogs for different altitude ranges (in order to improve reliability). The two catalogs are taken from the full catalog that is provided in input to the camera model, and the whole idea behind the multiple catalog configuration is that in a real setting the camera would be able to see only landmarks of a specific size range, depending on the current altitude.

Hence, to emulate this behavior, a threshold of 50m has been chosen, in order to separate what the camera is able to see at different stages during the EDL phase. When the altitude is above 1000m, the camera will be able to see landmarks with radii bigger than 50m (all landmarks from the unfiltered CAT1) and when the altitude is below 1000m, the camera will be able to see smaller landmarks (all landmarks from the unfiltered CAT2).

(i) A more refined, and accurate, solution would have been to have an altitude varying threshold for the camera landmark visibility

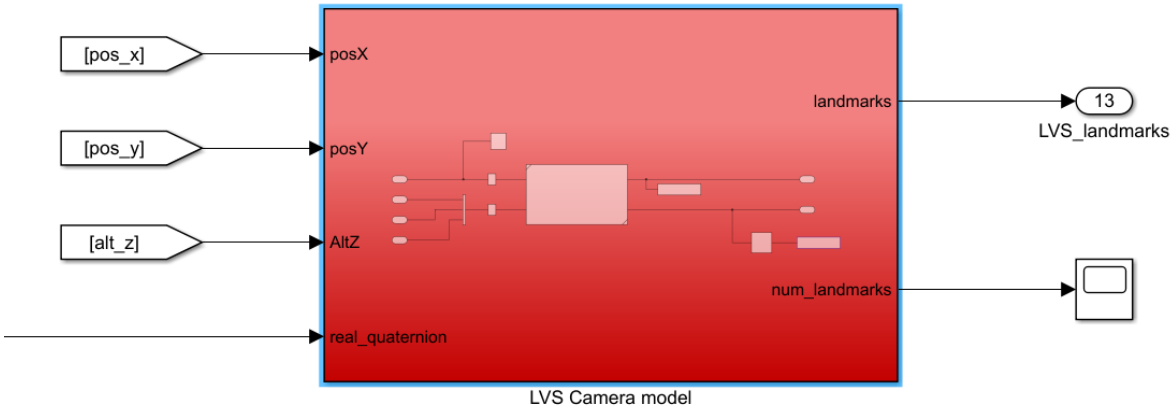


Figure 4.5: Camera Model block diagram (SIMULINK)

4.5.3 Formal approach

The formal approach to the pinhole model is the following [9].

Starting from a point p^M in target planet reference frame

$$p^M \rightarrow \tilde{p}^M = \begin{bmatrix} x_M \\ y_M \\ z_M \\ 1 \end{bmatrix}$$

Extracting the point in camera reference frame by using an homogeneous transformation matrix

$$\mathbf{T}_C^M = \begin{bmatrix} \mathbf{A}_C^M & \mathbf{r}_{MC}^M \\ \mathbf{0}_{1x3} & 1 \end{bmatrix}$$

$$\mathbf{p}^C \rightarrow \tilde{\mathbf{p}}^C = \mathbf{T}_C^M \tilde{\mathbf{p}}^M$$

$$\mathbf{p}^I = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{T}_C^M \tilde{\mathbf{p}}^M = \left[\begin{array}{ccc|c} & \mathbf{K} & & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{array}\right] \tilde{\mathbf{p}}^C \tag{4.1}$$

where $\mathbf{K}$ is the matrix of the intrinsic parameters.

Chapter 5

LVS Preprocessing

5.1 Introduction

Preprocessing, in the context of LVS, refers to the union of a matching algorithm (MA) and a pose (position/attitude) determination phase.

MA is responsible for finding landmark matches using data coming from the camera sensor and catalogs of the landing area stored in the on-board computer. The visible real landmarks data coming from the camera sensor is translated in its corresponding set of landmarks on the ground, via the inverse pinhole model. The MA is able to match this data against catalog landmarks, using various possible techniques.

A series of matched landmarks are then compared to the real landmarks data, and a correction is applied in order to correct the previously known pose estimate.

The problem of full attitude correction is significantly more complex. In this study only the horizontal position (x and y) is considered, as well as the angle error around the local vertical axis (z in LVLH, U in ENU).

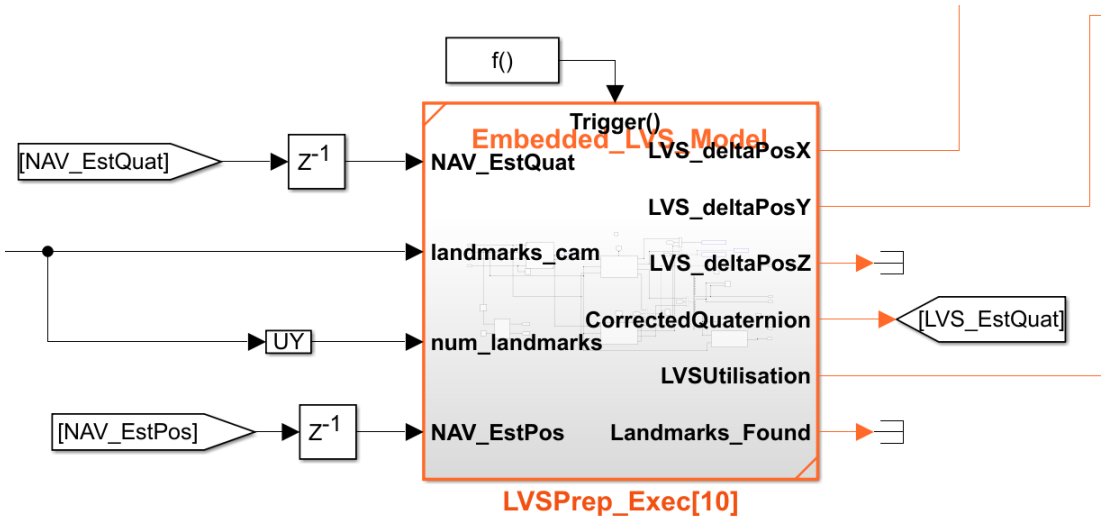


Figure 5.1: LVS preprocessing block diagram (SIMULINK)

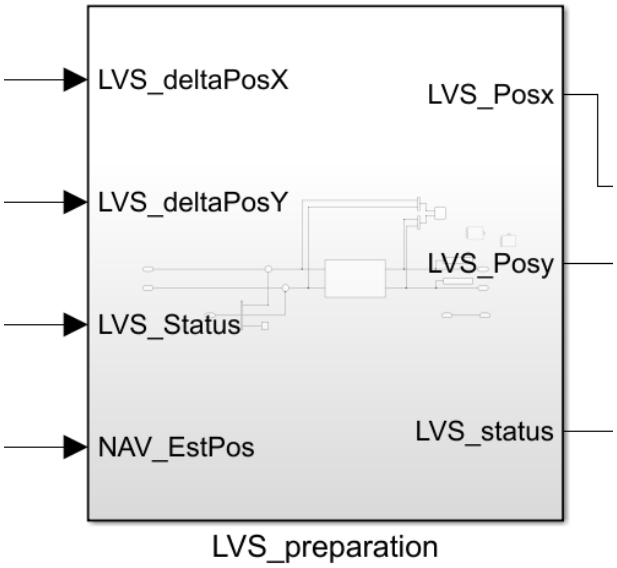


Figure 5.2: LVS preparation for navigation filter (SIMULINK)

5.2 Technical Considerations

5.2.1 Catalog Switching Time/Altitude

The catalog switch during the landing phase is crucial to LVS performance. It enables to optimize the matching procedure at varying altitudes, working in a sense as a change of resolution.

Three main approaches can be used.

The first is a catalog switch based on a specific timing (i.e. at 25s after the beginning of the EDL phase).

Indeed time-switch criterion can determine the involvement of a second catalog at quite different altitudes in various mission realizations, as a consequence of the guidance profile variability during the divert maneuver phase.

The second is a catalog switch that is activated when the number of matches is consistently below a specific threshold (i.e. 10 landmarks).

The third approach, and the one used in this implementation, is to activate the catalog switch at a specific altitude. An altitude of 1000m was found to be a sweet spot for LVS performance, but this value is highly dependent on the descent profile, the mission requirements and the target planet specific characteristics.

5.2.2 LVS deactivation

There must be a threshold for LVS deactivation, to notify the navigation filter that the current LVS output is not reliable and must not be considered (or used with an high associated variance/uncertainty).

One such condition can be that the number of matched landmarks is less than a minimum quantity (in our case is set to 5 matches). As the number of matches descreases, the quality of the corrected pose estimate significantly degrades.

Another condition for LVS deactivation is the altitude going lower than a certain critical altitude (in our case is set to 400m from the ground level).

Other possibles cases in which the LVS needs to be deactivated are hardware and software faults.

These last kinds of issues were not taken into account in this thesis.

5.3 Landmark data from image space to ground

The data coming from the camera is in the camera sensor image reference frame. By translating every landmark detected in the image space back to the planet using the reverse of the pinhole model, it's possible to then compare the landmark data from the on-board catalogs with the landmark data coming from the camera.

```
%% Landmark Data from Image space to Ground

%LVS (pinhole model)
[W, CC, ~, ~] = pinhole(w, h, actualf, a, r_MB_M_est, q_MB_est,
   r_BC_B, q_BC);

A_MB = quat2dcm(q_MB_est);
A_BC = quat2dcm(q_BC);
A_MC = A_MB*A_BC;

for i=1:num_landmarks
    cam_point = [landmarks_cam((i-1)*2+1);
       landmarks_cam((i-1)*2+2); 0];

    tmp = A_MC*cam_point + CC';
    k = ground_intersect(tmp', W);

    landmarks_ground(1, i) = k(1);
    landmarks_ground(2, i) = k(2);
end
```

Listing 5.1: Procedure to translate each landmark from the image space to the ground, using the inverse pinhole model

Where the *ground_intersect* function is responsible for finding the ground intersect point, passing through its two given input parameters (in this case the pinhole center W and the visible landmark expressed in the ground reference frame $\mathcal{R}_M$).

```
function [p] = ground_intersect(A, B)
    p = zeros(length(A(:,1)), 3);

    for i=1:length(A(:,1))
        % Line passing through two points
        x1 = A(i, 1);
        y1 = A(i, 2);
        z1 = A(i, 3);

        x2 = B(i, 1);
        y2 = B(i, 2);
        z2 = B(i, 3);

        % t calculation
        t = -z1/(z2-z1);

        x = x1 + t*(x2-x1);
        y = y1 + t*(y2-y1);
        z = 0;

        p(i,:) = [x y z];
    end
end
```

Listing 5.2: Implementation of the *ground_intersect* function

5.4 Algorithm for Landmark Matching

5.4.1 Initial Conditions

An initial pose estimate must be available, with a certain degree of accuracy. This can be done following a variety of algorithms known in literature [4]. techniques looking for constellations, triangles and clusters of landmarks in order to provide an initial pose estimate.

5.4.2 Threshold selection

A threshold value is needed to perform the matching. If the center point of the landmark observed by the camera and its nearest neighbor is less than the threshold value, than the matching is confirmed.

Therefore this value, tuned depending on the specific catalog granularity, is crucial for the MA performance. Its value depends also on the quality of the other sensors and the overall navigation filter characteristics.

A good performance can be obtained with the selection of a specific threshold value for each catalog. Additionally, if there is a need to increase performance, an altitude-varying

variable threshold can be chosen in order to avoid strong discontinuities in the matching errors.

In general, the matching errors need to be monitored during development and be relatively rare compared to the correct matches.

In our specific implementation, the following thresholds were chosen

$$threshold\,1 \;=\; 200m$$

$$threshold\,2 \;=\; 125m$$

5.4.3 MA implementation

```
matched_landmarks = 0;
landmarks_lvs = zeros(2, max_landmark);
landmarks_catalog = zeros(2, max_landmark);

%LVS (pinhole model)
[W, CC, C_pinhole, ~] = pinhole(w, h, actualf, a, r_MB_M_est,
   q_MB_est, r_BC_B, q_BC);

% for each CAM landmark
for j=1:Cam_found_landmarks
    match = false;

    % for all catalog landmarks
    for i=1:Cat.num_landmarks
        A = [Cat.clandmarkX(i); Cat.clandmarkY(i)];

        d = sqrt( (Cam_landmarks((j-1)*2+1)-A(1))^2 +
           (Cam_landmarks((j-1)*2+2)-A(2))^2 );

        % Threshold for matching
        if (d < matchingThreshold)
            match = true;
            matched_landmarks = matched_landmarks + 1;

            landmarks_catalog(1, matched_landmarks) = A(1);
            landmarks_catalog(2, matched_landmarks) = A(2);

            landmarks_lvs(1, matched_landmarks) =
               Cam_landmarks((j-1)*2+1);
            landmarks_lvs(2, matched_landmarks) =
               Cam_landmarks((j-1)*2+2);

            break;
        end
```

```
31      end
32
33      if matched_landmarks == max_landmark
34          break;
35      end
36  end
```

Listing 5.3: Procedure implementing the Matching Algorithm (MA)

5.4.4 Advanced techniques for MA

Other more refined techniques can be used if there is a need to improve the matching quality.

One of these possible ideas is to filter the matching results performing an outliers detection. This technique is given naturally if you use a least squares optimization technique in the Pose Determination phase, after the matching phase.

Additional data coming from the camera can be used, such as the radius or the parameters of the ellipse generated by the landamrk projection onto the camera sensor (semi axis major and minor, as well as inclination), can improve the matching quality.

Considering the closest neighbor might also be useful in reducing the possibility of a matching error, but this increases the computational complexity of the overall matching procedure, and provided that it utilizes a sufficiently filtered catalog it's not able to deliver additional tracking performance.

5.5 Optimization Algorithms for Pose Determination

An optimization algorithm for landmark matching can be employed, such as a least squares optimization [10], which is fundamentally taking into account the overall displacement error for x and y coordinates in LVLH reference frame, as well as any error in the z axis orientation ("yaw" angle).

In the following section a set of prerequisites are analyzed, as necessary conditions to apply the 2D least squares pose estimation algorithm, and the algorithm itself.

5.5.1 Prerequisites

Altitude (z) accuracy

The Altimeter for measuring the z position in LVLH reference frame is a crucial prerequisite for the position determination performance.

Known attitude quaternion ($\mathbf{q}_B^M$) accuracy

The attitude of the spacecraft w.r.t. the target planet reference frame must be known with an high confidence.

The LVS uses projections via the pinhole model, if the orientation estimate error is too big, this can lead to poor matching quality and therefore impact the overall system performance.

5.5.2 Least Squares 2D pose estimation

In order to get a 2D pose estimate, we start from a set of N two-dimensional landmarks observed by the camera sensor and matched by the MA.

Let's call $\mathbf{x}$ the observations vector

$$\mathbf{x}, \qquad \mathbf{x} = (\, x_1, x_2, ..., x_N \,)$$

And let's call $\mathbf{y}$ the landmark centers taken from the on-board catalogs that were a match for the N landmarks observations of the $\mathbf{x}$ vector

$$\mathbf{y}, \qquad \mathbf{y} = (\, y_1, y_2, ..., y_N \,)$$

In the ideal case, the observations relate to the real landmark positions such that

$$\mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{t}, \quad n = 1, ..., N$$

with $\mathbf{R}$ and $\mathbf{t}$ being the rotational and translational errors between reality and observations, giving a pose estimation error w.r.t the currently known state.

The real problem, with uncertainty and measurement errors, is the following minimization problem

$$\min \epsilon^2 \tag{5.1}$$

$$where\ \epsilon^2 = \sum_{n=1}^{N} w_n ||y_n - (Rx_n + t)||^2, \qquad w_n = \frac{1}{N} \tag{5.2}$$

which is a least square formulation. The solution, given by the translational error t and the rotation error R can be obtained by observing that

$$\bar{\mathbf{y}} = \mathbf{R}\bar{\mathbf{x}} + \mathbf{t} \tag{5.3}$$

$$where\ \bar{\mathbf{x}} = \sum_{n=1}^{N} w_n x_n, \qquad \bar{\mathbf{y}} = \sum_{n=1}^{N} w_n y_n \tag{5.4}$$

The translational error t is used to feed the Kalman filter, together with data coming from the RDA and the IMU. The same can be obtained for the orientation error estimation by observing the following relationships

$$\begin{cases} A = \sum_{n=1}^{N} w_n \left[(y_{n1} - \bar{y}_1)(x_{n1} - \bar{x}_1) + (y_{n2} - \bar{y}_2)(x_{n2} - \bar{x}_2) \right] \\ B = \sum_{n=1}^{N} w_n \left[(y_{n1} - \bar{y}_1)(x_{n2} - \bar{x}_2) + (y_{n2} - \bar{y}_2)(x_{n1} - \bar{x}_1) \right] \end{cases}$$

$$\begin{cases} cos\theta = \frac{-A}{\sqrt{A^2+B^2}} \\ sin\theta = \frac{B}{\sqrt{A^2+B^2}} \end{cases}$$

$$\mathbf{R} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \tag{5.5}$$

Where pedix 1 and 2 correspond to the horizontal components of each element x_n, y_n and the mean values $\bar{x}$ and $\bar{y}$.

5.5.3 Pose determination implementation

```
function [t, R, delta] = least_squares_2d(x, y, N)
 % Variables and parameters
 wn = 1/N;
 x_bar = zeros(2, 1);
 y_bar = zeros(2, 1);
 for i=1:N
    x_bar = x_bar + wn*x(:,i);
    y_bar = y_bar + wn*y(:,i);
 end

 % A, B calculation
 A = 0;
 B = 0;
 for i=1:N
     A = A + wn*((y(1,i) - y_bar(1))*(x(1,i) - x_bar(1)) +
        (y(2,i) - y_bar(2))+(x(2,i) - x_bar(2)));
     B = B + wn*((y(1,i) - y_bar(1))*(x(2,i) - x_bar(2)) -
        (y(2,i) - y_bar(2))+(x(1,i) - x_bar(1)));
 end

 % theta
 cos_theta = A/(sqrt(A^2+B^2));
 sin_theta = -B/(sqrt(A^2+B^2));

 % Results
 R = [
     cos_theta -sin_theta
     sin_theta cos_theta
 ];

 t = y_bar - x_bar;
 delta = y_bar -R*x_bar;
```

Listing 5.4: Procedure implementing a 2D least-squares optimization algorithm for pose determination

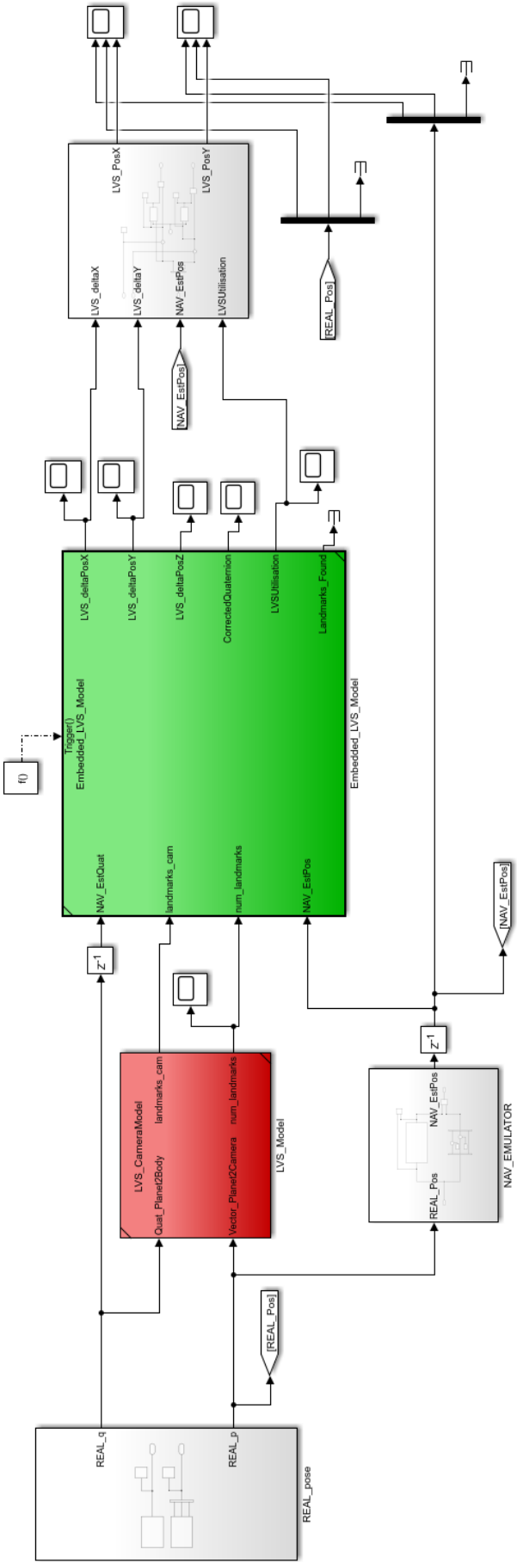


Figure 5.3: LVS architecture

Part III

Navigation

Chapter 6

Navigation Architecture

6.1 Introduction

The navigation architecture forms the backbone of any autonomous or semi-autonomous system, enabling precise state estimation.

This chapter aims to provide an overview of the navigation architecture and the GNC loop as a whole for the specific challenge that is the precise autonomous pinpoint landing.

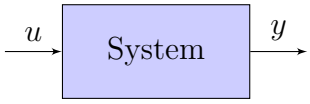
Furthermore, the role of on-board sensors is examined, including the Inertial Measurement Unit (IMU), Altimeter (ALT), Radar Doppler Altimeter (RDA), and Landing Vision System (LVS). These sensors serve as the eyes and ears of the navigation system, capturing critical data about the spacecraft's state. Each sensor provides different capabilities while also having its specific set of limitations.

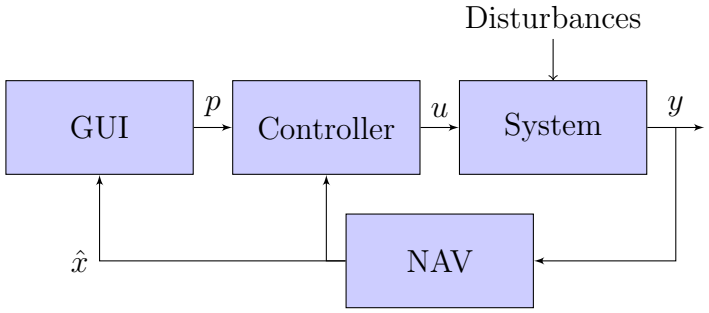
Finally, the role of the filtering process is examined, which plays a pivotal role in integrating heterogeneous sensor data, mitigating noise and uncertainty, and generating accurate estimates of the lander state.

6.2 Guidance, Navigation and Control (GNC)

The Guidance, Navigation, and Control (GNC) framework is a closed loop system designed to obtain a specific behavior from a system. It's composed by four main components:

- Plant/System (in our case the lander spacecraft)
- Navigation (NAV)
- Guidance (GUI)
- Control (CON)





Guidance

The guidance component of the GNC system is responsible for formulating high-level plans and trajectories to navigate the vehicle from its current state to a desired state. It involves decision-making algorithms that consider various factors such as mission objectives, environmental constraints, and vehicle dynamics.

Navigation

Navigation encompasses the process of determining the spacecraft's state. This involves the integration of sensor measurements, such as IMU, RDA and LVS, and taking into account the system dynamics, to estimate the lander state accurately. Through techniques like sensor fusion and probabilistic filtering, the navigation module tries to deliver the best description of the system state at any time. Navigation is the focus of this thesis.

Control

The control component of the GNC system is responsible for executing commands going to the RCS thrusters in order to follow the desired trajectory while maintaining stability and responsiveness. It involves feedback control algorithms that continuously adjust actuator inputs, such as throttle and steering, to regulate the vehicle's motion. The control module ensures precise tracking of the planned trajectory while compensating for disturbances and uncertainties in the environment.

6.3 On-board Sensors

The on-board sensors integrated into autonomous navigation systems play an important role in capturing critical data about the vehicle's surroundings, motion, and orientation. This section provides a description of the various on-board sensors, including their functionalities, advantages and limitations.

6.3.1 Inertial Measurement Unit (IMU)

The Inertial Measurement Unit (IMU) is a fundamental sensor in autonomous navigation systems, providing information about the vehicle's acceleration and angular velocity. Comprising accelerometers and gyroscopes, the IMU offers high-frequency measurements that are essential for precise localization and motion estimation.

In our context, the IMU works at a frequency of 100Hz, and is used in both the translational filter (x, y and z accelerometers) and orientation filter (gyroscopes).

Therefore, from the IMU, an estimate of the acceleration in x, y and z axis is expected, delivered at a frequency of 100 times per second.

$$acc_x,\ acc_y,\ acc_z, \qquad f = 100Hz$$

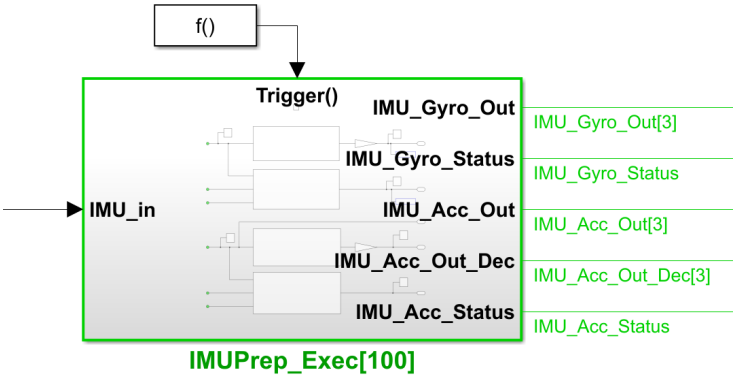


Figure 6.1: IMU sensor (SIMULINK)

6.3.2 Radar Altimeter or Radar Doppler Altimeter (RDA)

Radar Doppler Altimeter (RDA) sensor is able to measure velocity and altitude, while a simple Radar Altimeter only provides an altitude measurement. These sensors emit electromagnetic waves and analyze the reflected signals to determine the range, velocity relative to the LVLH ref.Frame. While traditional radar sensors provide range and bearing information, Doppler radar sensors offer additional capabilities for measuring relative velocities. Basic range functionality provides altimetric data (altitude).

$$pos_z, \qquad f = 20Hz$$

Doppler capability provides also

$$vel_x,\ vel_y,\ vel_z, \qquad f = 20Hz$$

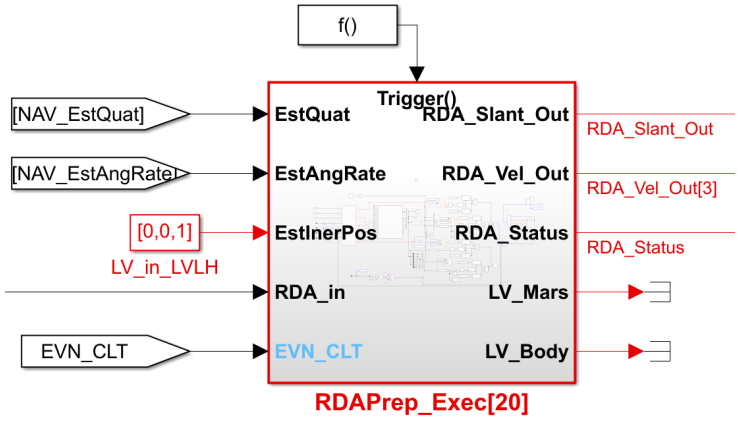


Figure 6.2: RDA sensor (SIMULINK)

Radar Doppler Altimeter sensor for precise pinpoint landing

In the results section, various simulations using Monte Carlo analysis are conducted to assess the criticality of RDA in achieving precise pinpoint landing.

6.3.3 Landing Vision System (LVS)

The LVS architecture was deeply analyzed in the previous part, and is a landmark-based pose estimation/correction sensor that uses a camera (or a collection of cameras). In this study the LVS is going to provide only the horizontal position components (in LVLH/ENU). A partial attitude correction is performed in the pose determination phase, but it's not used, since only the translational navigation filter was redesigned for this thesis, while the rotational/orientation dynamics observer was not modified.

$$pos_x,\ pos_y,\ q_{corrected} \qquad f = 10Hz$$

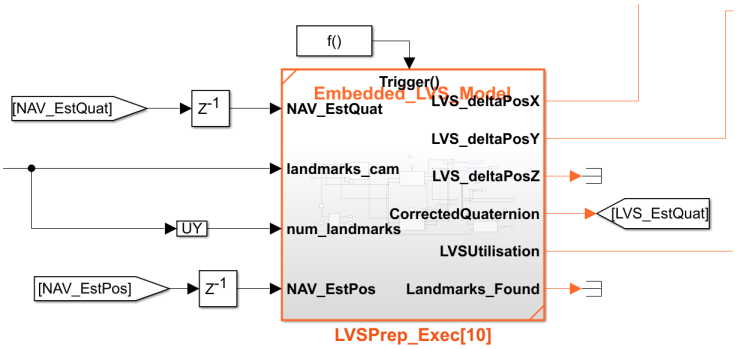


Figure 6.3: LVS sensor (SIMULINK)

6.4 Filter

The filter is in charge of merging together all data coming from the sensors (data fusion) at different data rates and with varying reliability and variance characteristics, as well as trying to estimate all state variables, even if not measured directly.

6.4.1 Orientation dynamics observer

The Orientation Dynamics Observer is tasked with estimating the orientation of the system based on sensor measurements and dynamics models. By integrating data from sensors such as gyroscopes and accelerometers, this observer provides a continuous estimate of the system's orientation.

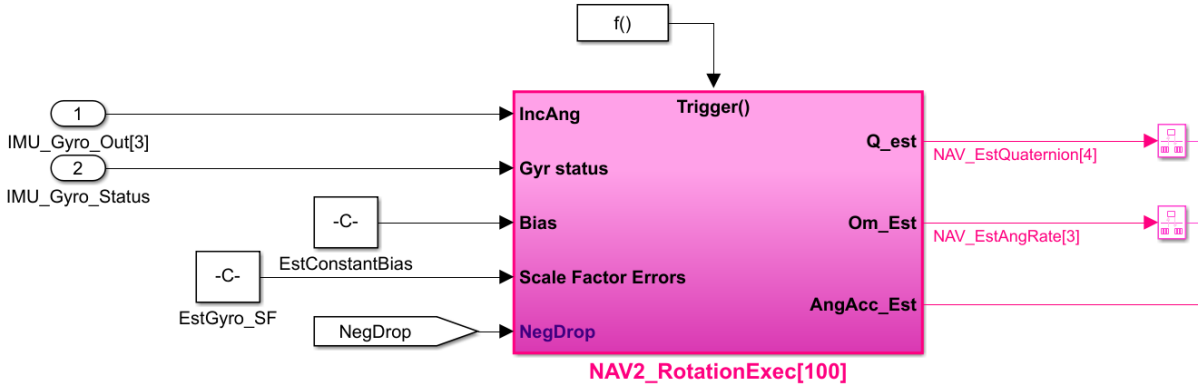


Figure 6.4: Orientation Observer (SIMULINK)

6.4.2 Translational dynamics observer

The Translational Dynamics Observer focuses on estimating the translational motion of the system, including position, velocity and acceleration, using sensor data. By incor-

porating measurements from sensors like accelerometers (in the IMU), RDA and LVS, this observer provides accurate estimates of the system's translational state, allowing for precise navigation and control.

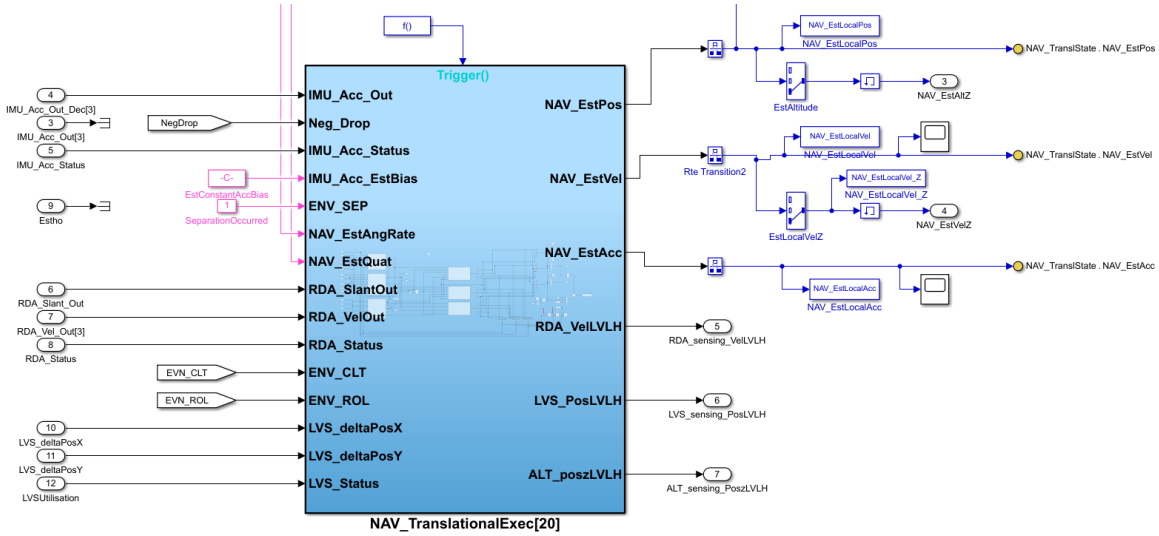


Figure 6.5: Translational Observer (SIMULINK)

6.5 Contributions

This thesis contributions to the filtering phase was the design of a Kalman Filter for the translational dynamics estimation.

In particular, all sensors characteristics were analyzed, such as the variance profile characterizations at different altitudes and during various stages during descent.

The filter takes all these sensor data as input and performs an estimation of the state vector.

Additional technical considerations were taken into account, such as different data rates among sensors, and delays in data processing, producing an output delayed in time.

Different sensor configurations were analyzed, and tracking performance is studied across all these different configurations of available sensors.

In particular, the configurations that doesn't use the RDA sensor but a simple altimeter is compared to the configuration with the RDA sensor available.

Chapter 7

Simulator Description

The simulator used during the development of this thesis includes a representation of the "Real World" in combination with the algorithmic suite of the GNC. The Real World, hereafter represented, includes the following elements:

7.0.1 RCS

Reaction Control System block is a simulacrum of the propulsion subsystem. It takes the commands generated by GNC and determines both the thrust level and the consumed propellant linked to the control activation. The model accounts the electrical and fluido-dynamic behavior of the propulsion system including the pressure decay from the tanks to the thruster inlet, the ambient pressure the rise and decay expected profiles of the thrusters such as the additive and multiplicative errors on the provided impulse.

7.0.2 Dynamic and Kinematics

Dynamics and Kinematics is fed by the forces and torques generated by the RCS. The model generates the rotational and dynamic nominal or "true" kinematics of the spacecraft

7.0.3 Gyroscopes and Accelerometers Packages

These models are the two portions of the Inertial Measurement Model. They determine the incremental angle (gyroscope segment) and incremental velocity (accelerometer segment) consistent with the input kinematics, including in addition the specified errors: biases, scale factors inaccuracy, random walk, quantization, misalignment. By accounting the integration time, incremental angle and velocity allow the computation of the measured angular rate and, respectively, acceleration.

7.0.4 RDA

The Radar Doppler Altimeter block determines the slant measurement and the Doppler measurement expected by the instrument. The slant measurement is the distance between the antenna and the terrain in the direction of a RDA beam. The doppler measurement is

the projection of the velocity in the direction of the beam. Both the slant and the doppler measurements are affected by errors depending on various error sources, in general, growing with the augmentation of the altitude. A peculiar error is represented by the avionic delay that has to take into account the age of each beam measurement when RDA data are combined. Given the RDA attitude, the slant of each beam permits the identification of the altitude, the Doppler measurements of at least three different beams allow the determination of the spacecraft velocity. For the specific study case "altimeter only", the Doppler segment is neglected in such manner to exercise the GNC and assess its capability in reconstructing, autonomously, the spacecraft velocity.

7.0.5 LVS

The Landing Vision System is a model fed by the attitude and position of the camera that provides in output the list of observed objects with the linked parameters. In the current application the list reports the position (2-D) of the detected landmark centers (in the target planet ground reference frame). An enhanced application of the model provides the amplitude of major and minor semiaxes (in the camera dimensions) and the orientation of the observed ellipse. The LVS includes a multiplicative error growing with the altitude from terrain. In addition provided LVS data are delayed by one working cycle to account the time needed for image elaboration.

7.1 Simulation Parameters

This section includes the most important parameters applied for both the model and the SW tuning.

7.1.1 Lander Configuration

M&I (Augmented EXM Lander)

Mass = 1500 [kg] EXM + 300 kg
Ixx = 680 $[kg \cdot m^2]$
Iyy = 1150 $[kg \cdot m^2]$
Izz = 1540 $[kg \cdot m^2]$
Note: Z_{BL} symmetry axis

7.1.2 Post-Divert Guidance

MinitPrev = 1873 Mass before the separation of the backshell
Mnominal = 1500 Mass of the lander after the separation of the backshell

$$IM_{orig} = \begin{bmatrix} 678.169 & 2.244 & 1.122 \\ 2.244 & 1144.990 & 3.366 \\ 1.122 & 3.366 & 1535.019 \end{bmatrix}$$

7.2 IMU

$$M_{IMU} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Nominal Mounting matrix of the IMU in the spacecraft.
$\sigma_{misgxy} = 1.e^{-4}$ Standard Deviation of the misalignment

Gyroscope Errors

$\sigma_{gyrR} = 2.99e^{-9}\,^\circ$ Rate Random Walk standard deviation
$\sigma_{gyrA} = 1.2e^{-4}\,^\circ$ Angular Random Walk standard deviation
$\sigma_{gyrN} = 1.e^{-3}\,^\circ$ Angular White Walk standard deviation
$\sigma_{biasGyr} = 2.78e^{-4}\,^\circ/s$ Constant Bias standard deviation
$\sigma_{ScaleFactorGyr} = 300ppm$ Scale factor standard deviation
$Quant_{gyr} = 4.354479e^{-4}\,^\circ/s$ Quantization level

Accelerometer Errors

$\sigma_{accN} = 1.1e^{-4}m/s$ Accelerometer White Noise
$\sigma_{biasAcc} = 300e^{-5}m/s^2$ Constant bias of the accelerometer
$\sigma_{ScaleFactorAcc} = 300ppm$ Constant bias standard deviation
$Quant_{acc} = 1.5e^{-4}m/s^2$ Quantization level

7.3 RDA

$RDA\_LowerLimitAltitude = 10m$ Minimum altitude for which RDA measurements are present
$Trd = 0.03s$ Delay of the RDA single measurement generation
$BeamSep = 20/180$ Angular Separation of the three lateral beams
Note: slant and doppler measurement from second, third and fourth beam in the data combination are delayed by respectively 50, 100 and 150 ms in addition

Position of the antenna centers in the lander reference frame

$$RDA0Pos = \begin{bmatrix} 0.5 & 0 & 0 \end{bmatrix}\ m$$
$$RDA1Pos = \begin{bmatrix} 0.5 \cdot cos(BeamSep) & 0.5 \cdot sin(BeamSep) & 0 \end{bmatrix}\ m$$
$$RDA2Pos = \begin{bmatrix} 0.5 \cdot cos(BeamSep) & -0.25 \cdot sin(BeamSep) & \sqrt{3}/4 \cdot sin(BeamSep) \end{bmatrix}\ m$$
$$RDA3Pos = \begin{bmatrix} 0.5 \cdot cos(BeamSep) & -0.25 \cdot sin(BeamSep) & -\sqrt{3}/4 \cdot sin(BeamSep) \end{bmatrix}\ m$$

Direction of the beams in the lander reference frame

$$RDA0Dir = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$
$$RDA1Dir = \begin{bmatrix} cos(BeamSep) & sin(BeamSep) & 0 \end{bmatrix}$$
$$RDA2Dir = \begin{bmatrix} cos(BeamSep) & -0.5 \cdot sin(BeamSep) & 0.5 \cdot \sqrt{3} \cdot sin(BeamSep) \end{bmatrix}$$

$RDA3Dir = \begin{bmatrix} cos(BeamSep) & -0.5 \cdot sin(BeamSep) & -0.5 \cdot \sqrt{3} \cdot sin(BeamSep) \end{bmatrix}$

7.4 LVS

Optical Errors of the camera 0.1%

$M_LVS = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ Mounting matrix of the LVS in the spacecraft

$Tld = 0.1s$ Delay of the LVS provided data

7.4.1 Initial Conditions

Initial vertical velocity from 80 to 100 m/s
Initial horizontal velocity ¡ 7 m/s
Initial offvertical angle ¡ 5°
Initial angular rate ¡ 4°/s
Initial Position knowledge 10 m per axis
Initial Velocity Knowledge 0.5 m/s per axis
Initial Attitude knowledge 1° per axis
Initial Lander Mass Knowledge 0.1%
Initial Diagonal Inertia knowledge 1%
Initial Knowledge of IMU biases and scale factors 50%

Chapter 8

Data Fusion: Kalman Filter

The translational dynamics estimation filter is the core of this thesis work, and is designed to deliver high quality full state vector estimation, integrating sensor measurements coming from IMU, RDA and also LVS.

Faults or deactivation of one or more of these sensors, although affecting performances, don't jeopardize the entire landing mission.

In the results section, different sensor configurations are tested in order to check landing success rate, vertical and horizontal landing speed.

8.1 Data Fusion

Data fusion is a fundamental process in modern engineering systems where information from multiple sources is combined to generate a more accurate and reliable estimate of the system state. In the context of navigation and control, data fusion is crucial for integrating measurements from various sensors to estimate the position, velocity, and orientation of a vehicle accurately.

8.1.1 Kalman Filter

The Kalman Filter (KF) [3] [5] is a widely used algorithm for data fusion in dynamic systems. It provides an optimal solution for estimating the state of a dynamic system when measurements are affected by noise. The KF minimizes the error of the estimate by recursively updating the state estimate based on incoming measurements and system dynamics.

The main advantages of the Kalman Filter include its simplicity, efficiency, and robustness in handling noisy measurements. It can effectively fuse data from multiple sensors with different characteristics and provide accurate state estimates even in the presence of uncertainties.

The KF has enhanced, at the maximum extent, features of agility and low computational burden, by identifying a linearized architecture not requiring periodic linearization and discretization. In a problem like the one under study, the basic source of non-linearity comes from the attitude. The basic guideline was therefore in the direction of decoupling the attitude and the translational navigation processes.

This is possible in consideration of the fact that the attitude estimated through the STR and, later on, integrated for few dozens of minutes through the IMU, has a precision hard to be improved through the outcome of the landmark navigation. On the other hand, if the attitude problem can be considered already solved, the known quaternion can be used to pre-process the slant and RDA velocity data such as the images of the LVS from the instrument (or lander) reference frame to the local vertical local horizontal reference frame. It's therefore possible to handle a simplified translational navigation architecture composed by three decoupled subsystems, each one of them fed by measurements of one axis derived from the IMU, LVS and RDA or Altimeter.

The inherent reduction of the sensor error features with the distance from the terrain is accounted through the dependence of the run-time computed gains of the variable covariance matrix R.

In the following sections, a deeper dive into the actual implementation details of the Kalman Filter for precise pinpoint landing will be undertaken.

8.2 Kalman Filter

8.2.1 State vector

The state vector for the control system is the following

$$\hat{\mathbf{x}}_n = \begin{bmatrix} \hat{x}_n \\ \dot{\hat{x}}_n \\ \ddot{\hat{x}}_n \\ \hat{y}_n \\ \dot{\hat{y}}_n \\ \ddot{\hat{y}}_n \\ \hat{z}_n \\ \dot{\hat{z}}_n \\ \ddot{\hat{z}}_n \end{bmatrix} \tag{8.1}$$

It includes the position, velocity and acceleration in LVLH reference frame for each axis, for a total of $n = 9$ components.

8.2.2 Initial Conditions

Initial State Vector

$$\hat{\mathbf{x}}_{00} = \begin{bmatrix} \hat{x}_0 \\ \hat{\dot{x}}_0 \\ \hat{\ddot{x}}_0 \\ \hat{y}_0 \\ \hat{\dot{y}}_0 \\ \hat{\ddot{y}}_0 \\ \hat{z}_0 \\ \hat{\dot{z}}_0 \\ \hat{\ddot{z}}_0 \end{bmatrix} \tag{8.2}$$

8.2.3 Process Prediction Equations

State Extrapolation Equation

State Predict process is modeled with the State Extrapolation equation

$$\hat{\mathbf{x}}_{n+1,n} = \mathbf{F}\hat{\mathbf{x}}_{n,n} + \mathbf{G}\mathbf{u}_n + \boldsymbol{\omega}_n \tag{8.3}$$

Where $\mathbf{x}$ is the state vector, $\mathbf{F}$ and $\mathbf{G}$ are the state transition matrix and the control matrix respectively, $\mathbf{u}$ is the input command and $\boldsymbol{\omega}_n$ is the process noise vector
In our case

$$\mathbf{u}_n = 0 \quad \rightarrow \quad \mathbf{G}\mathbf{u}_n = 0$$

$$\hat{\mathbf{x}}_{n+1,n} = \mathbf{F}\hat{\mathbf{x}}_{n,n} + \boldsymbol{\omega}_n \tag{8.4}$$

The discrete time constant of the filter is set to

$$\Delta t = 0.05s$$

Leading to the following state transition matrix

$$F = \begin{bmatrix} 1 & \Delta t & 0.5 \cdot \Delta t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & 0.5 \cdot \Delta t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0.5 \cdot \Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.5}$$

$$\begin{bmatrix} \hat{x}_{n+1,n} \\ \hat{\dot{x}}_{n+1,n} \\ \hat{\ddot{x}}_{n+1,n} \\ \hat{y}_{n+1,n} \\ \hat{\dot{y}}_{n+1,n} \\ \hat{\ddot{y}}_{n+1,n} \\ \hat{z}_{n+1,n} \\ \hat{\dot{z}}_{n+1,n} \\ \hat{\ddot{z}}_{n+1,n} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0.5 \cdot \Delta t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & 0.5 \cdot \Delta t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & 0.5 \cdot \Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{x}_{n,n} \\ \hat{\dot{x}}_{n,n} \\ \hat{\ddot{x}}_{n,n} \\ \hat{y}_{n,n} \\ \hat{\dot{y}}_{n,n} \\ \hat{\ddot{y}}_{n,n} \\ \hat{z}_{n,n} \\ \hat{\dot{z}}_{n,n} \\ \hat{\ddot{z}}_{n,n} \end{bmatrix} \tag{8.6}$$

Covariance Extrapolation Equation

$$\mathbf{P}_{n+1,n} = \mathbf{F}\mathbf{P}_{n,n}\mathbf{F}^t + \mathbf{Q} \tag{8.7}$$

Process Noise Matrix

$$\tilde{\mathbf{Q}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.8}$$

$$\mathbf{Q}_a = \tilde{\mathbf{Q}} \cdot \sigma_{acc}^2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{acc}^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{acc}^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{acc}^2 \end{bmatrix} \tag{8.9}$$

$$\mathbf{Q}_a = \tilde{\mathbf{Q}} \cdot \sigma_{acc}^2 \qquad \mathbf{Q} = \mathbf{F} \cdot \mathbf{Q}_a \cdot \mathbf{F}^t \tag{8.10}$$

The σ_{acc}^2 value depends on the system characteristics and model completeness. After simulations and fine-tuning, the value was set to

$$\sigma_{acc}^2 = 0.2$$

Measurement Equation

$$\mathbf{z}_n = \mathbf{H}\mathbf{x}_n + \mathbf{v}_n \tag{8.11}$$

Where $\mathbf{z}_n$ is the measurement vector, $\mathbf{x}_n$ is the hidden true state, $\mathbf{v}_n$ is a random noise vector and $\mathbf{H}$ is the observation matrix.

The observation matrix H serves the purpose of linearly transforming the system state into corresponding outputs.

Initial Process Covariance Matrix

The initial process covariance matrix indicated the trustworthiness of the state vector at $t = 0$, and assuming covariances between axis equal to zero, the initial matrix can be set as follows:

$$\mathbf{P}_{00} = \begin{bmatrix} \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 & 0 & 0 & 0 \\ \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 & 0 & 0 & 0 \\ \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{00} & \sigma_{00} & \sigma_{00} \end{bmatrix} \tag{8.12}$$

where σ_{00} is the initial state variance for each component of the state vector. It is initialized as

$$\sigma_{00} = \phi \quad \phi \in \mathcal{R},\ >> 0$$

in order to mark the initial state as highly unreliable.

Covariances across different axis are supposed null and are therefore set to zero.

Measurement state selection (H)

Observation matrices were developed to emulate different sensor configurations (depending on which sensors are active)

The observation matrix selects the estimated state variables to be compared with the sensor measurements.

$$\hat{\mathbf{y}}_{n,n} = \mathbf{H}\hat{\mathbf{x}}_{n,n-1} \tag{8.13}$$

$$Innovation = \mathbf{z}_n - \mathbf{H}\hat{\mathbf{x}}_{n,n-1} \tag{8.14}$$

The sensor measurements are ordered in the following way

$$\mathbf{z}_n = \begin{bmatrix} \text{LVS\_Posx} \\ \text{RDA\_Velx} \\ \text{IMU\_Accx} \\ \text{LVS\_Posy} \\ \text{RDA\_Vely} \\ \text{IMU\_Accy} \\ \text{ALT\_Posz} \\ \text{RDA\_Velz} \\ \text{IMU\_Accz} \end{bmatrix} \tag{8.15}$$

To compare these measurements with the previous state variables, the observation matrix is required.

Depending on the active sensors configuration, **H** matrices can be designed in the following ways.

H0: No sensors available

$$H_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{8.16}$$

H1: Only IMU is active

$$H_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.17}$$

H2: IMU, Altimeter and RDA available

$$H_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.18}$$

H3: IMU, Altimeter, RDA and LVS available

$$H_3 = \begin{bmatrix} 1 & k_1 & k_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & k_1 & k_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.19}$$

Where the k_1 and k_2 parameters are used to perform a delay compensation (see section 8.5.2), since the LVS preprocessing block produces an output that lags behind of a certain amount of time (in the order of 100ms).

H4: IMU, Altimeter and LVS available

$$H_4 = \begin{bmatrix} 1 & k_1 & k_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & k_1 & k_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.20}$$

Measurement Noise Covariance Matrix

$$\mathbf{R} = \begin{bmatrix} \text{LVS\_Posx\_variance} \\ \text{RDA\_Velx\_variance} \\ \text{IMU\_Accx\_variance} \\ \text{LVS\_Posy\_variance} \\ \text{RDA\_Vely\_variance} \\ \text{IMU\_Accy\_variance} \\ \text{ALT\_Posz\_variance} \\ \text{RDA\_Velz\_variance} \\ \text{IMU\_Accz\_variance} \end{bmatrix} \tag{8.21}$$

The measurement variance can be modeled as constant, or can vary across the landing phase. In section (7.3) variance is characterized for each sensor measurement.

8.2.4 State Update Equation

The overall formulation of the state update equation is the following

$$\hat{\mathbf{x}}_{n,n} = \hat{\mathbf{x}}_{n,n-1} + \mathbf{K}_n(\mathbf{z}_n - \mathbf{H}\hat{\mathbf{x}}_{n,n-1}) \tag{8.22}$$

where the innovation (equation 7.14) is weigthed by the Kalman Gains $\mathbf{K}_n$ and the result of this computation is used to update the previous state estimate. The equation to derive the Kalman Gains is at section (7.2.6)

8.2.5 Covariance Update Equation

$$\mathbf{P}_{n,n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H})\mathbf{P}_{n,n-1}(\mathbf{I} - \mathbf{K}_n\mathbf{H})^t + \mathbf{K}_n\mathbf{R}_n\mathbf{K}_n^t \tag{8.23}$$

8.2.6 Kalman Gain

$$\mathbf{K}_n = \mathbf{P}_{n,n-1}\mathbf{H}^t(\mathbf{H}\mathbf{P}_{n,n-1}\mathbf{H}^t + \mathbf{R}_n)^t \tag{8.24}$$

8.3 Variance characterization

Experimentally found parameters for variance of each sensor on specific axis is the following

$$\begin{cases} \text{IMU Accx variance} = 0.02; \\ \text{IMU Accy variance} = 0.02; \\ \text{IMU Accz variance} = 0.02; \\ \text{ALT Posz variance} = 100; \\ \text{RDA Velx variance} = 6.21 - \frac{4.24}{(1+(\frac{alt_z}{781.47})^{4.54})}; \\ \text{RDA Vely variance} = 6.21 - \frac{4.24}{(1+(\frac{alt_z}{781.47})^{4.54})}; \\ \text{RDA Velz variance} = 5.98 - \frac{4.24}{(1+(\frac{alt_z}{781.47})^{4.54})}; \\ \text{LVS Posx variance} = 200; \\ \text{LVS Posy variance} = 200; \end{cases} \tag{8.25}$$

8.3.1 IMU variance characterization

Inertial Measurement Unit has a fixed variance value, independent of the spacecraft altitude. This value was fine-tuned and values in the order of 0.01 up to 0.03 provide consistent tracking performance.

8.3.2 ALT variance characterization

Altimeter sensor, embedded in the RDA, provides the vertical position (altitude) measurement. Its measure quality depends on the spacecraft altitude, but a fixed value was still enough to guarantee vertical position and velocity tracking, so no further refinement was done.

8.3.3 RDA variance characterization

The RDA sensor is able to compute the horizontal and vertical velocities, and the variance of its measurements is highly dependent on the spacecract altitude. For this reason, an altitude-varying variance curve is designed to optimally characterize the output measeremement quality.

To be able to do this, a mobile variance measure is computed during landing simulations, and a curve able to envelope a good portion of variance profiles was chosen, using a fitting polinomial equation.

For the three axis, two curves has been designed, corresponding to the XY horizontal variances and the Z vertical velocity measurement variance.

$$\text{RDA Velxy variance} = 6.21 - \frac{4.24}{(1 + (\frac{alt_z}{781.47})^{4.54})}; \tag{8.26}$$

$$\text{RDA Velz variance} = 5.98 - \frac{4.24}{(1 + (\frac{alt_z}{781.47})^{4.54})}; \tag{8.27}$$

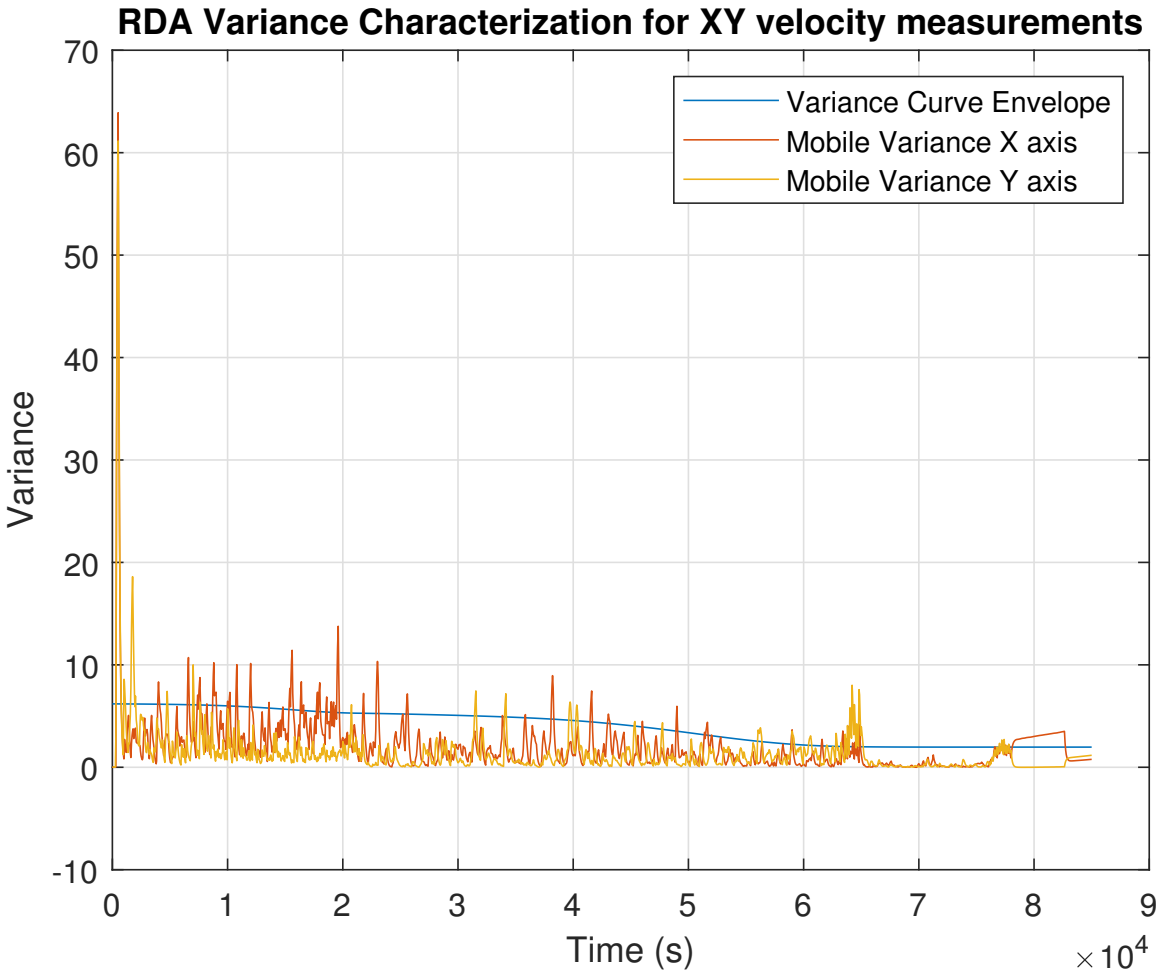


Figure 8.1: RDA Variance Characterization for X and Y velocity measurements

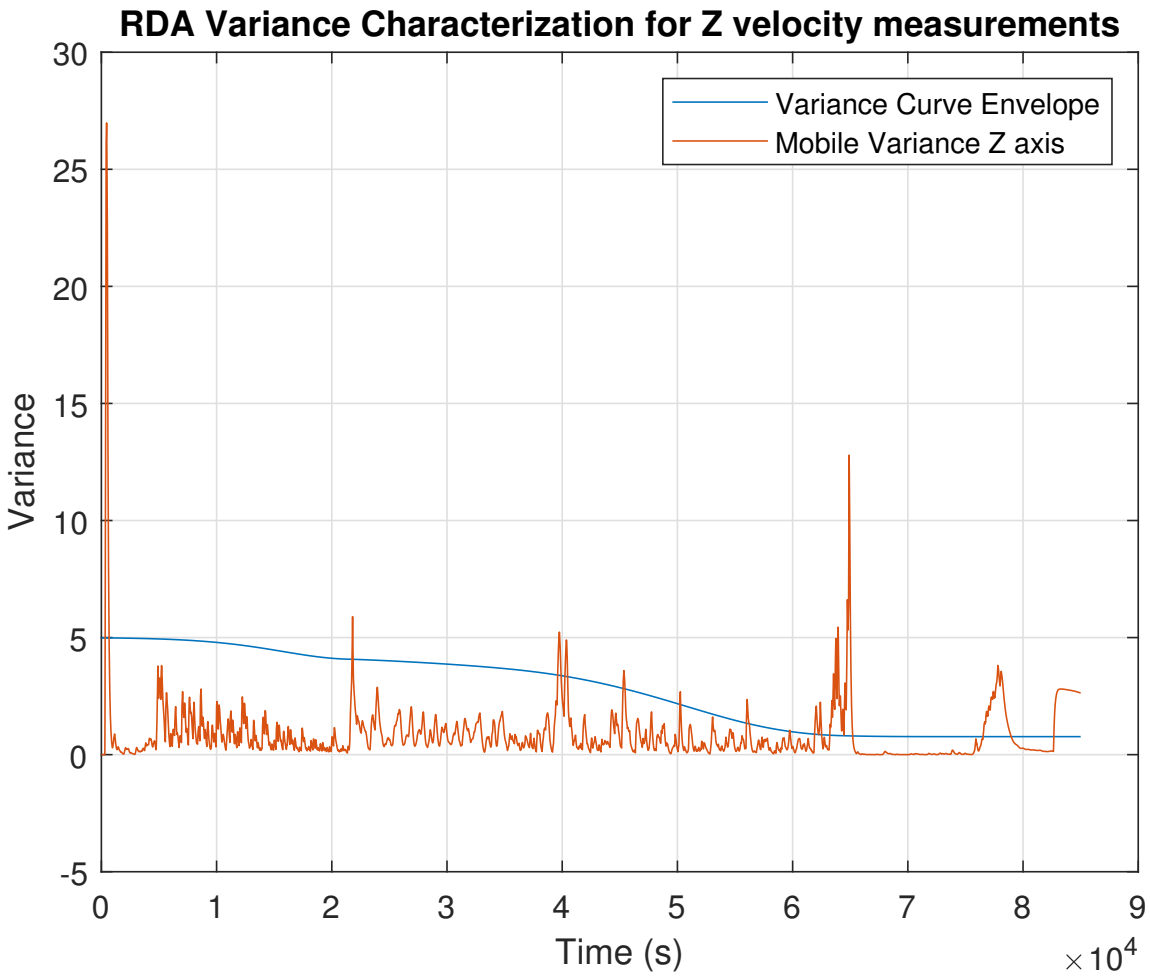


Figure 8.2: RDA Variance Characterization for Z velocity measurements

8.3.4 LVS variance characterization

The LVS, as the altimeter, have varying degrees of output measerement quality, in terms of variance. However, using a fixed variance value, provides sufficiently high tracking performance.

The LVS variance value is 200.

8.4 Implementation and integration

The overall translational dynamics filter block diagram is the following

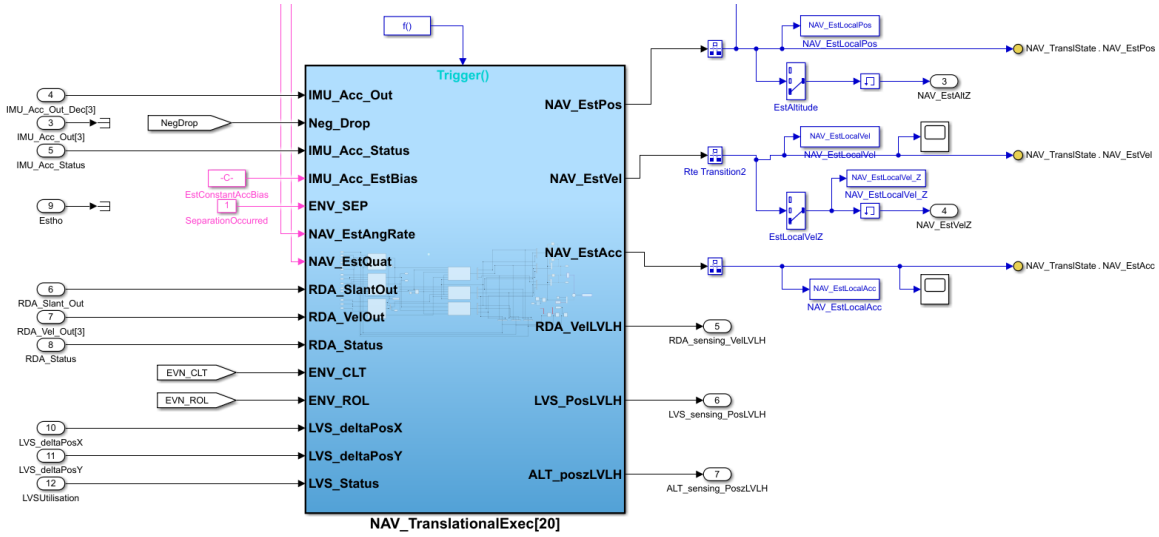
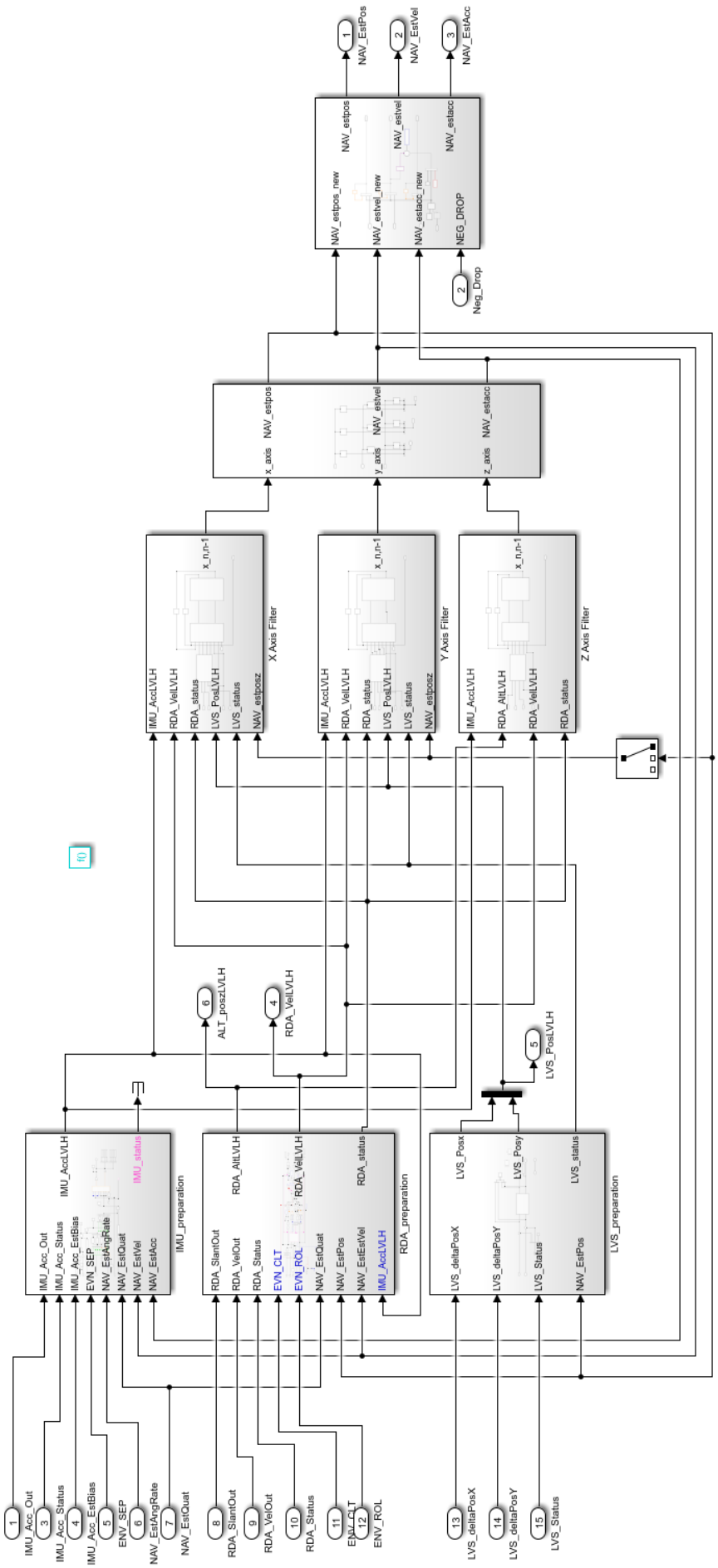


Figure 8.3: Translational dynamics navigation filter (SIMULINK)

8.4.1 Model recap

$$
\begin{cases}
\textit{initial conditions} \;\rightarrow\; \hat{\mathbf{x}}_{00},\ \mathbf{P}_{00} \\
\\
\textit{update model}: \\
\quad \textit{State Update Equation}: \;\hat{\mathbf{x}}_{n,n} = \hat{\mathbf{x}}_{n,n-1} + \mathbf{K}_n(\mathbf{z}_n - \mathbf{H}\hat{\mathbf{x}}_{n,n-1}) \\
\quad \textit{Covariance Update Equation}: \;\mathbf{P}_{n,n} = (\mathbf{I} - \mathbf{K}_n\mathbf{H})\mathbf{P}_{n,n-1}(\mathbf{I} - \mathbf{K}_n\mathbf{H})^t + \mathbf{K}_n\mathbf{R}_n\mathbf{K}_n^t \\
\quad \textit{Measurement Equation}: \;\mathbf{z}_n = \mathbf{H}\mathbf{x}_n + \mathbf{v}_n \\
\quad \textit{Kalman Gains}: \;\mathbf{K}_n = \mathbf{P}_{n,n-1}\mathbf{H}^t(\mathbf{H}\mathbf{P}_{n,n-1}\mathbf{H}^t + \mathbf{R}_n)^t \\
\quad \textit{Measurement State Selection}: \;\mathbf{H}_0,\ \mathbf{H}_1,\ \mathbf{H}_2,\ \mathbf{H}_3,\ \mathbf{H}_4 \\
\quad \textit{Measurement Noise Covariance Matrix}: \;\mathbf{R}_n(\hat{\mathbf{x}}) = E(\mathbf{v}_n\mathbf{v}_n^t) \\
\\
\textit{predict model}: \\
\quad \textit{State Extrapolation Equation}: \;\hat{\mathbf{x}}_{n+1,n} = \mathbf{F}\hat{\mathbf{x}}_{n,n} + \boldsymbol{\omega}_n \\
\quad \textit{Covariance Extrapolation Equation}: \;\mathbf{P}_{n+1,n} = \mathbf{F}\mathbf{P}_{n,n}\mathbf{F}^t + \mathbf{Q} \\
\quad \textit{Process Noise Matrix}: \;\mathbf{Q}_n = E(\boldsymbol{\omega}_n\boldsymbol{\omega}_n^t)
\end{cases}
\tag{8.28}
$$


NAV_EstPos
NAV_EstVel
NAV_EstAcc
NAV_estpos_new
NAV_estpos
NAV_estvel_new
NAV_estvel
NAV_estacc_new
NAV_estacc
NEG_DROP
Neg_Drop
x_axis
y_axis
z_axis
X Axis Filter
Y Axis Filter
Z Axis Filter
IMU_AccLVLH
RDA_VelLVLH
RDA_status
LVS_PosLVLH
LVS_status
NAV_estposz
RDA_AltLVLH
ALT_poszLVLH
IMU_preparation
RDA_preparation
LVS_preparation
IMU_Acc_Out
IMU_Acc_Status
IMU_Acc_EstBias
EVN_SEP
NAV_EstAngRate
NAV_EstQuat
NAV_EstVel
NAV_EstAcc
IMU_status
RDA_SlantOut
RDA_VelOut
RDA_Status
EVN_CLT
EVN_ROL
NAV_EstPos
NAV_EstEstVel
LVS_delPosX
LVS_delPosY
LVS_Posx
LVS_Posy
LVS_Status

8.5 Technical challenges

8.5.1 Non-homogeneous sensor data rates

Sensors provide output data at different frequency rates

$$\begin{cases} \text{IMU Task Rate} & = 0.010 \text{ (100Hz, 1ms)}; \\ \text{RDA Task Rate} & = 0.050 \text{ (20Hz, 50ms)}; \\ \text{LVS Task Rate} & = 0.100 \text{ (10Hz, 100ms)}; \end{cases} \tag{8.29}$$

In order to account for this, different techniques can be employed [8], and a series of design choices were made. Firstly, the overall filter step frequency was set equal to the RDA sensor rate.

$$\Delta t = 0.05$$

This allows to seamlessly integrate the RDA measerements and the IMU measurements (the IMU produced incremental data, therefore a frequency of 20Hz in the output can be obtained by simply reading the sensor every 50ms instead of 10ms). However, this generates issues in dealing with the slower rate of the LVS. To account for this, the LVS output data is processed to create a pseudo-measure that is at the same frequency of the RDA sensor. The LVS output measure has a frequency of 0.1s, providing one sample for every two RDA measures.

$$LVS(k) = LVS(k \cdot \Delta t), \quad \Delta t = 0.05$$

$$LVS(0) = LVS(0), \quad LVS(1) = LVS(1) = LVS(0), \quad LVS(2) = LVS(2)$$

$$\begin{cases} LVS(k) = 1.5 \cdot LVS(k) - 0.5 \cdot LVS(k-1), & k > 2 \, and \, k \leq 4, \; k\%2 == 1 \\ LVS(k) = LVS(k), & k > 2 \, and \, k \leq 4, \; k\%2 == 0 \end{cases} \tag{8.30}$$

$$\begin{cases} LVS(k) = LVS(k) + b \cdot 0.5 \cdot \Delta t + c \cdot (0.5 \cdot \Delta t)^2, & k > 4, \; k\%2 == 1 \\ LVS(k) = LVS(k), & k > 4, \; k\%2 == 0 \end{cases} \tag{8.31}$$

Where

$$\begin{cases} b = \frac{3 \cdot LVS(k) - 4 \cdot LVS(k-1) + LVS(k-2)}{2 \cdot \Delta t} \\ c = \frac{LVS(k) - 2 \cdot LVS(k-1) + LVS(k-2)}{2 \cdot \Delta t^2} \end{cases}$$

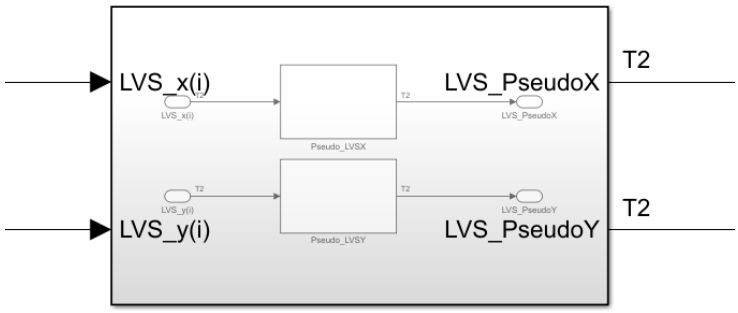


Figure 8.5: Pseudo Measure block (SIMULINK)

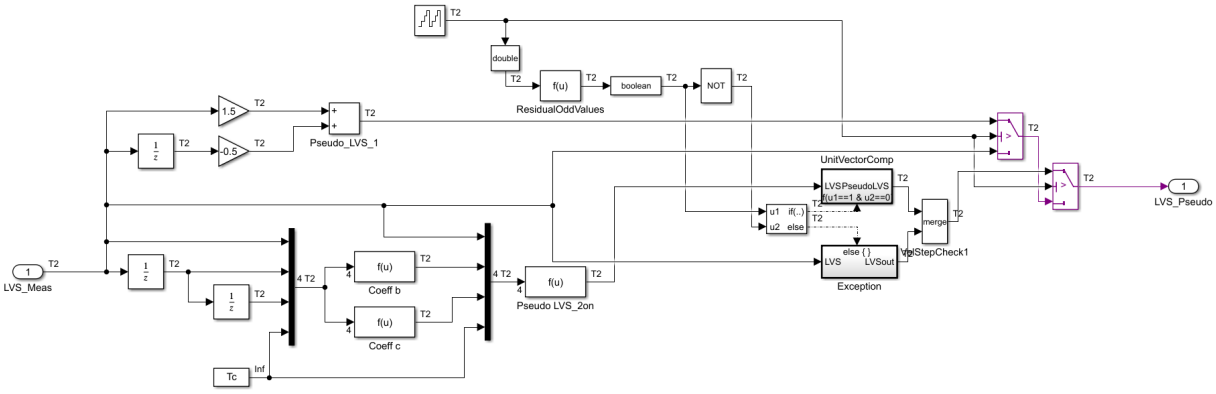


Figure 8.6: Pseudo Measure implementation (SIMULINK)

8.5.2 Sensor data delay compensation

Besides the different data rates, there is also a delay issue with the LVS measure. The data processing computations in the LVS architecture are particularly heavy, thus providing a delayed output measure. In order to compensate for this, appopriate coefficients k_1 and k_2 must be present in the state observation matrix $\mathbf{H}$ to be able to compensate the delay by using the velocity and acceleration estimates. These coefficients can be tuned in a real scenario and depend on the actual delay in the image processing and LVS preprocessing task.

The RDA delays doesn't appear in the matrix because the RDA measurements, supplied to the Kalman filter, are already compensated for delay through an anticipation of the data achieved in the RDA preprocessing block through the accelerometers.

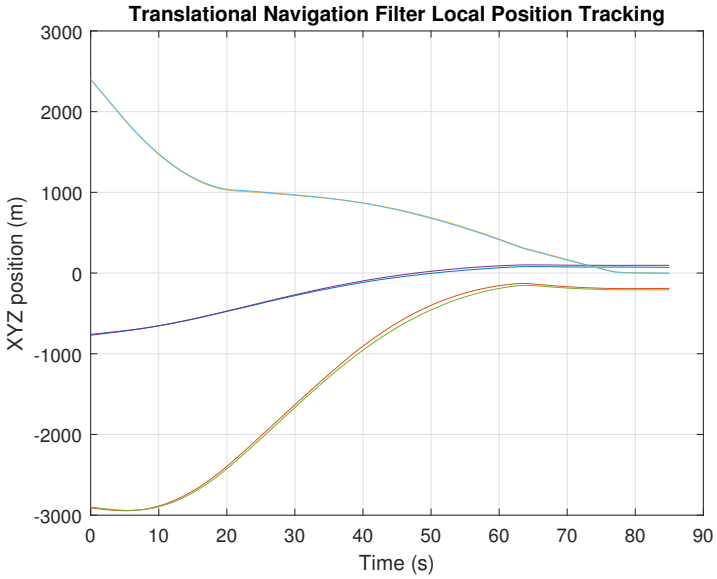


Figure 8.7: Translational dynamics navigation filter position tracking

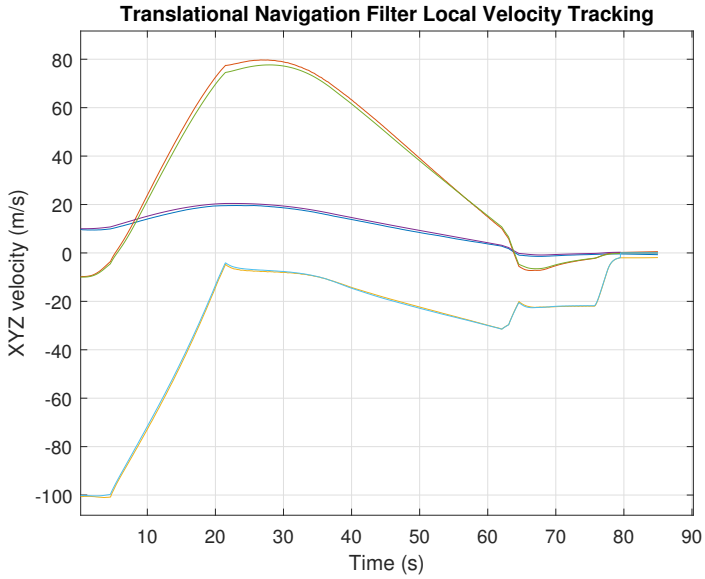


Figure 8.8: Translational dynamics navigation filter velocity tracking

Part IV

Simulation results

Chapter 9

Performance analysis

The following is a collection of the results of multiple simulations performed via Monte Carlo analysis.

The performance of the navigation with the LVS system supported by IMU and RDA sensors is tested, analyzing the success rate and other metrics such as horizontal and vertical touchdown speeds, as well

A total of 100 cases were used in the Monte Carlo simulation, corresponding to a hundred different initial conditions, leading to their corresponding guidance trajectory, which is run as one-shot at the beginning of the final landing phase of the mission, immediately before LVS activation.

After the initial condition assessment is performed by a TRN system, the guidance profile is calculated and the LVS sensor is activated, in order to work together with RDA and IMU and perform a precise landing.

In the following sections two main sensor configurations are tested, with and without the Radar Doppler sensor, letting the LVS horizontal position measure and the altimeter vertical position measure be indect information on speed, without the need of dedicated doppler functionality (direct speed measurements).

The filter tuning is performed on the LVS variance in the $\mathbf{R}$ matrix, by multiplying it for a coefficient ζ. Two different tuning coefficients are analyzed, with a value of 1 and 2

$$Base\ LVS\ Variance = 200$$

$$\zeta = 0.75 \quad \rightarrow \mathbf{R}_{LVS} = 200 \cdot 0.75 = 150$$

$$\zeta = 1.5 \quad \rightarrow \mathbf{R}_{LVS} = 200 \cdot 1.50 = 300$$

Where $\mathbf{R}_{LVS}$ is the value in the matrix $\mathbf{R}$ corresponding to the LVS measures of x and y position coordinates.

These two configurations are going to be called $NORDA_{150}$ and $NORDA_{300}$.

9.0.1 Requirements

Horizontal and Vertical Speed

The main requirement to be satisfied is the horizontal speed, since this is critical for mission success. A reasonable value for maximum horizontal speed can be set to

$$Max\ \ Horizontal\ \ Speed = 3m/s$$

In the simulation results different thresholds are taken into considerations

$$\begin{cases} Horizontal\ \ Speed < 3.0m/s \\ Horizontal\ \ Speed < 2.5m/s \\ Horizontal\ \ Speed < 2.0m/s \end{cases}$$

Horizontal speeds that exceed the 3m/s are always considered a landing failure.

For the vertical speed the requirement is to stay below the 4m/s threshold. Any vertical speed that crosses the 4m/s leads to a mission failure

$$Max\ \ Vertical\ \ Speed = 4m/s$$

For both requirements the reasons are related to mechanical stress on the landing spacecraft legs. For the horizontal touchdown speed constraint there is also the risk, if the speed is too high, for the spacecraft to topple over.

Touchdown distance to target

Precise landing capabilities are the frontier of the new landing missions on other planets. Cargo and human missions are going to need to land precisely in the order of 250m, to be retrieved by a rover or land close to a base on the target planet.

$$Touchdown\ \ Distance\ \ to\ \ Target < 250m$$

This is an hard constraint for mission success. Ideally, reaching distances to target lower than 100m is desirable.

9.1 Performance analysis in the presence of RDA sensor

In this section there are the results of the Monte Carlo simulation with all sensors active (IMU, RDA and LVS). When the results are expressed as a success rate, they follow the thresholds set in the requirements section at the beginning of this chapter.

Table 9.1: Simulation Sensor suite configuration and settings

Configuration in the presence of IMU, RDA and LVS	
Sensor	**Status setting**
IMU	*ACTIVE*
Altimeter	*ACTIVE*
Radar Doppler	*ACTIVE*
LVS	*ACTIVE*

Table 9.2: Simulation Results in the presence of RDA (success rate and mean distance to target)

Metric	**Result (out of 100 simulations)**
Success Rate	*100%*
mean distance to target at touchdown	*40.08m*

Table 9.3: Simulation Results in the presence of RDA (horizontal and vertical speeds)

Metric	**Result (out of 100 simulations)**
horizontal speed < 3.0m/s	*100%*
horizontal speed < 2.5m/s	*100%*
horizontal speed < 2.0m/s	*100%*
vertical speed < 4.0m/s	*100%*
vertical speed < 3.5m/s	*100%*
vertical speed < 3.0m/s	*98%*

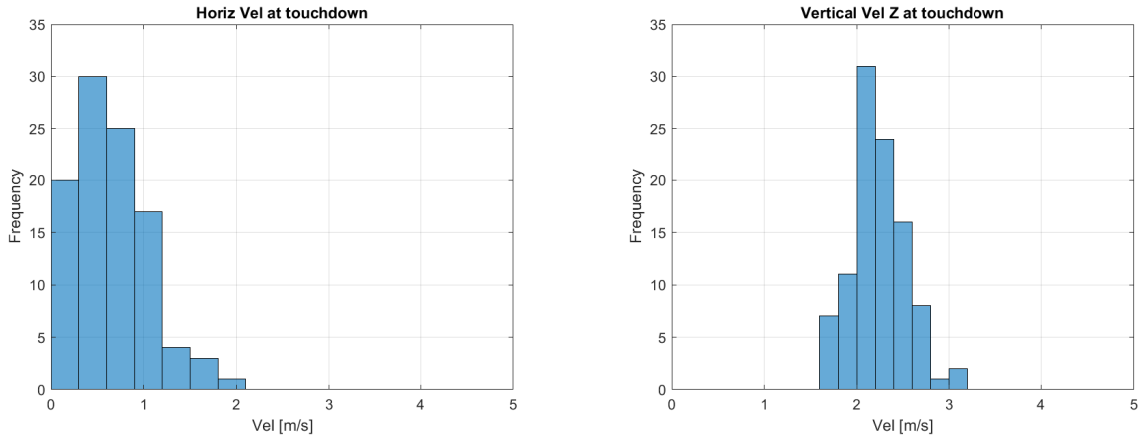


Figure 9.1: Horizontal and Vertical Speed at Touchdown

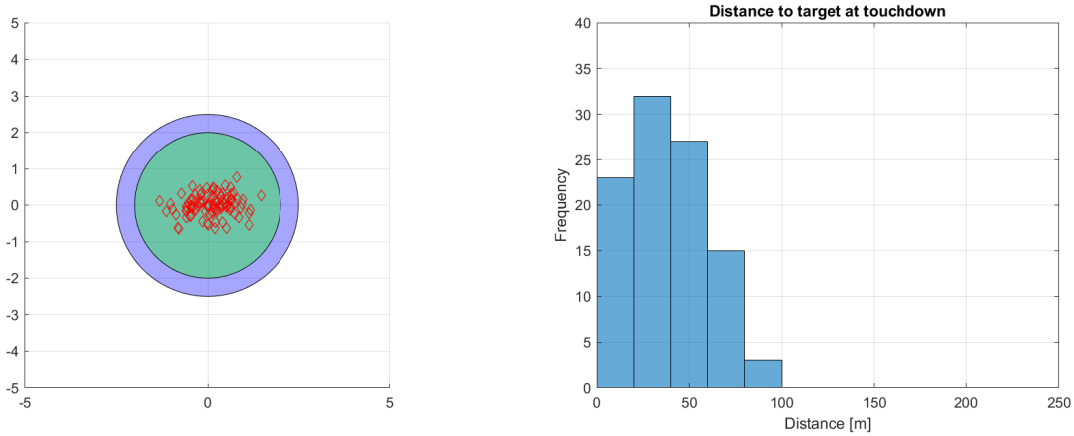


Figure 9.2: Horizontal and Vertical speeds (2m/s and 2.5m/s thresholds represented as circles) (left), Distance to target at Touchdown (right)

Detailed Performance (in the presence of RDA sensor)

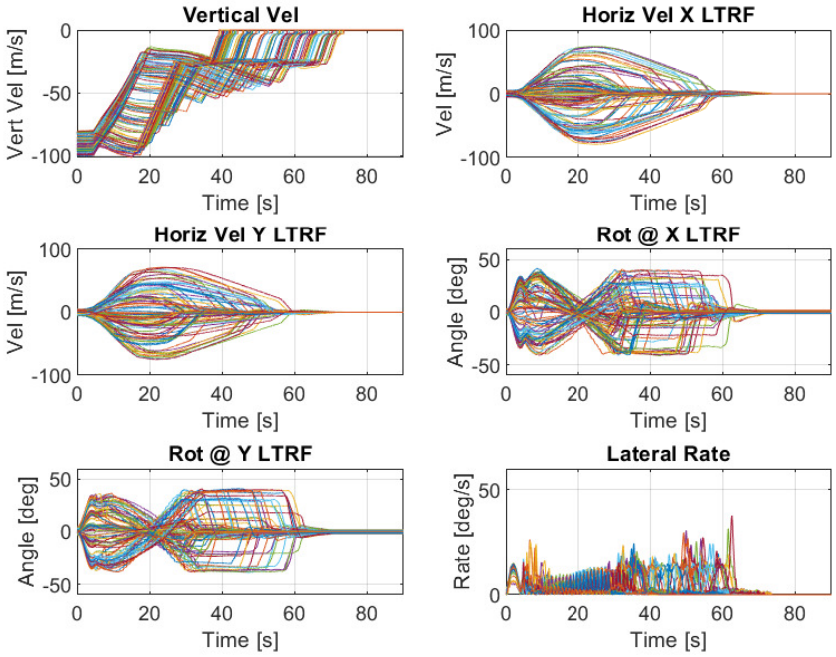


Figure 9.3: Velocities and Angular rates

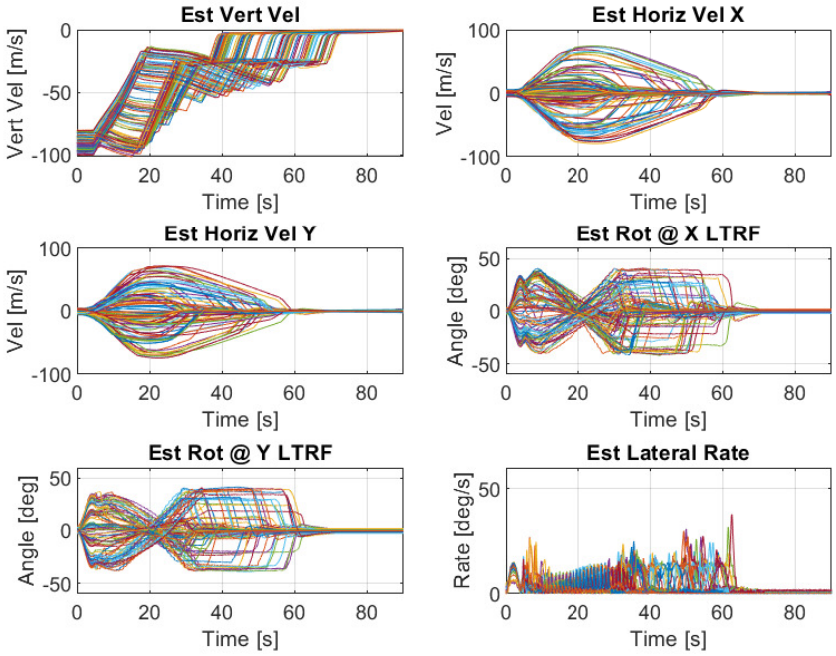


Figure 9.4: Estimated Velocities and Angular rates

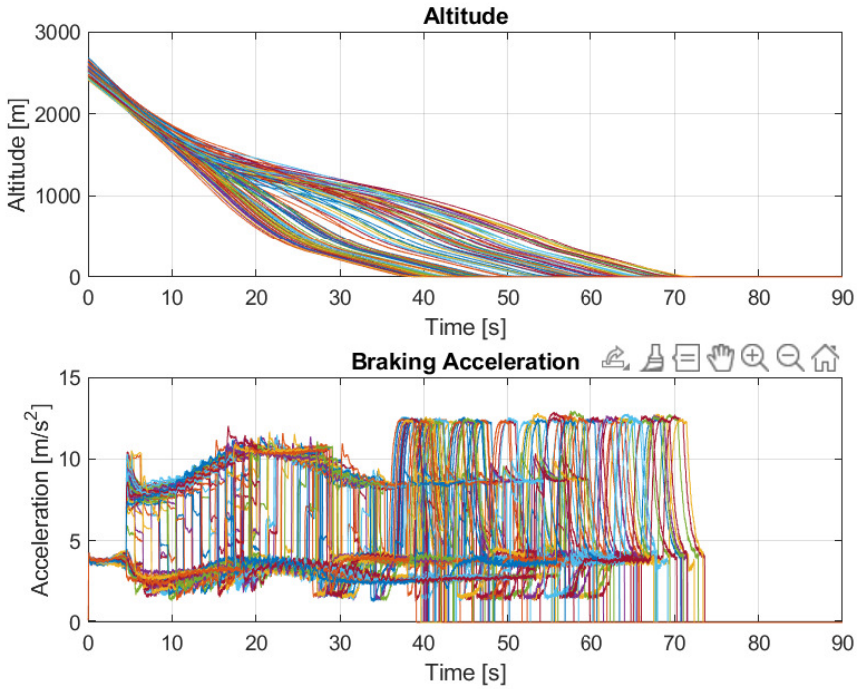


Figure 9.5: Altitude and Breaking acceleration

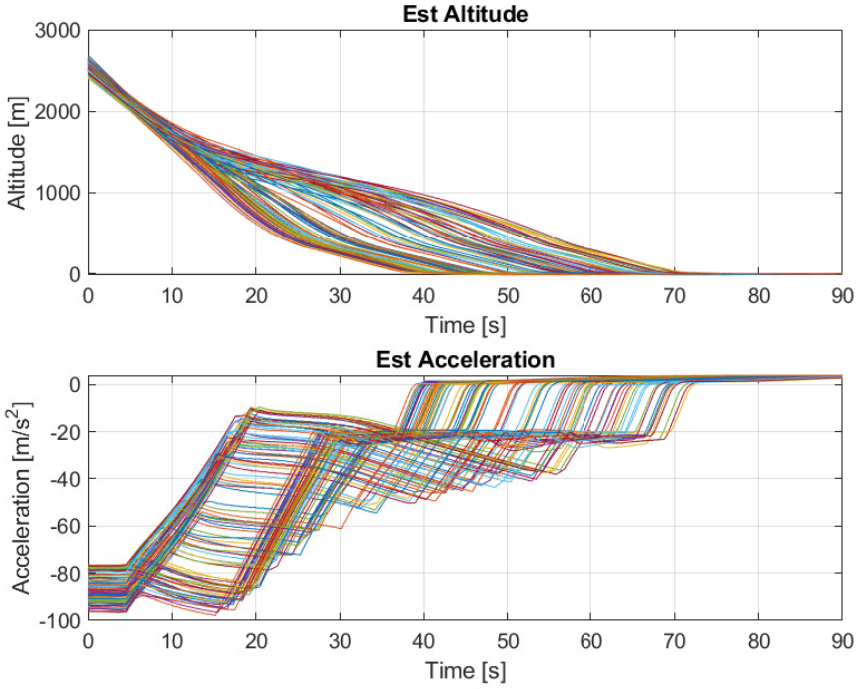


Figure 9.6: Estimated altitude and Estimated acceleration

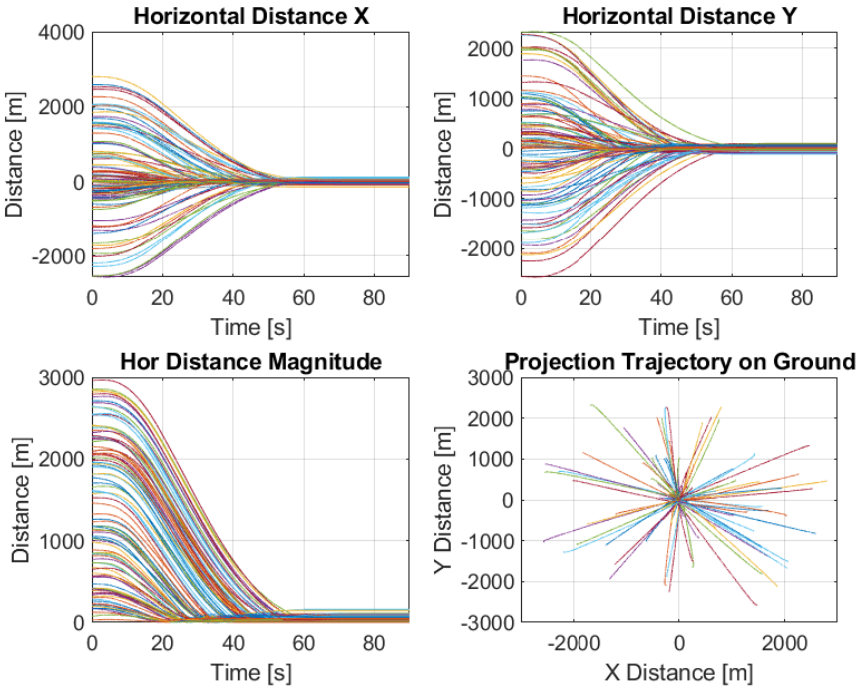


Figure 9.7: Horizontal distance to target

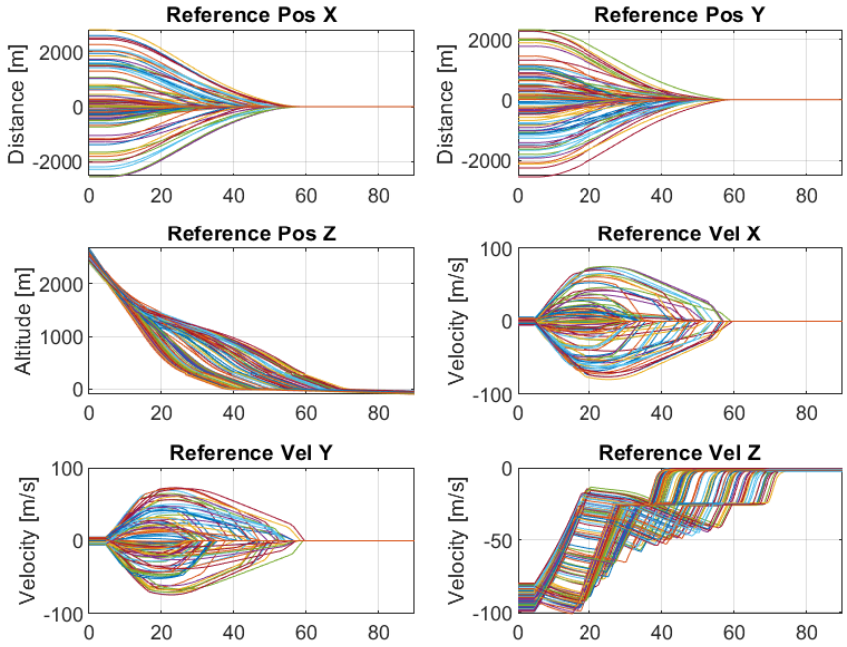


Figure 9.8: Guidance reference position and velocity

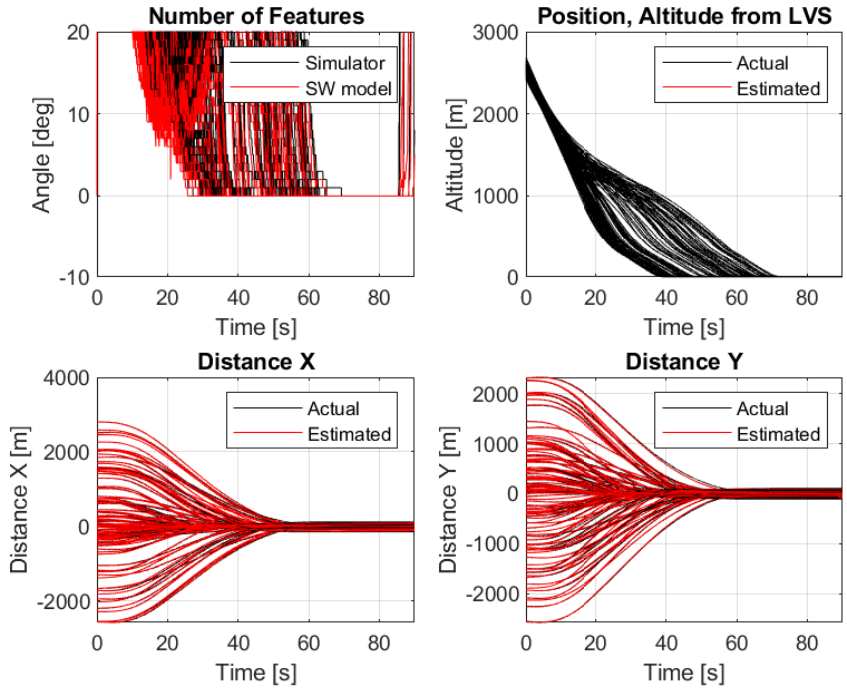


Figure 9.9: LVS number of features and altitude (top), Distance to target (bottom)

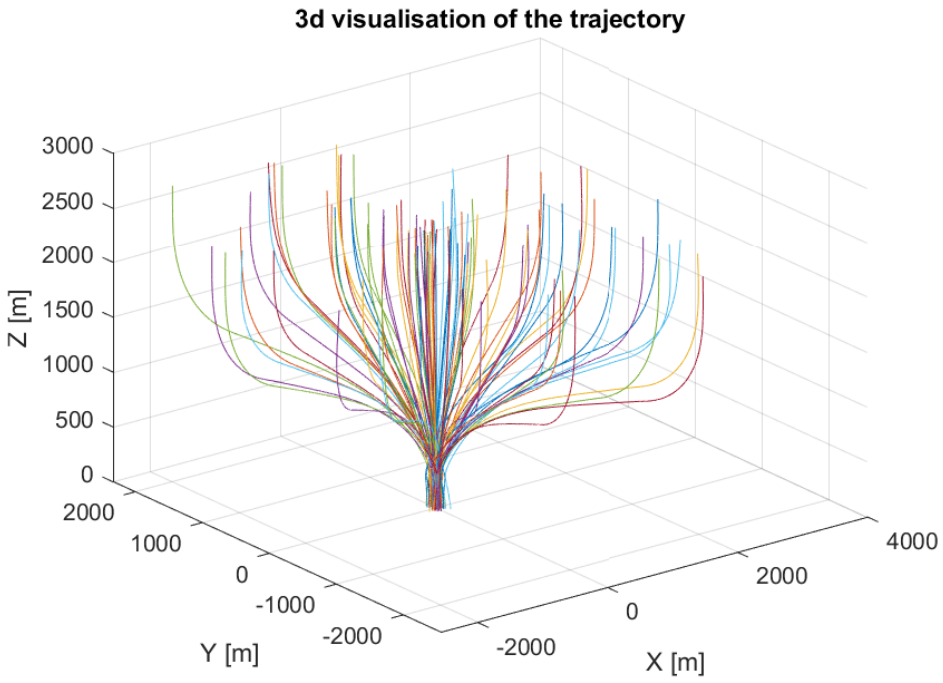


Figure 9.10: Trajectories visualization (3D)

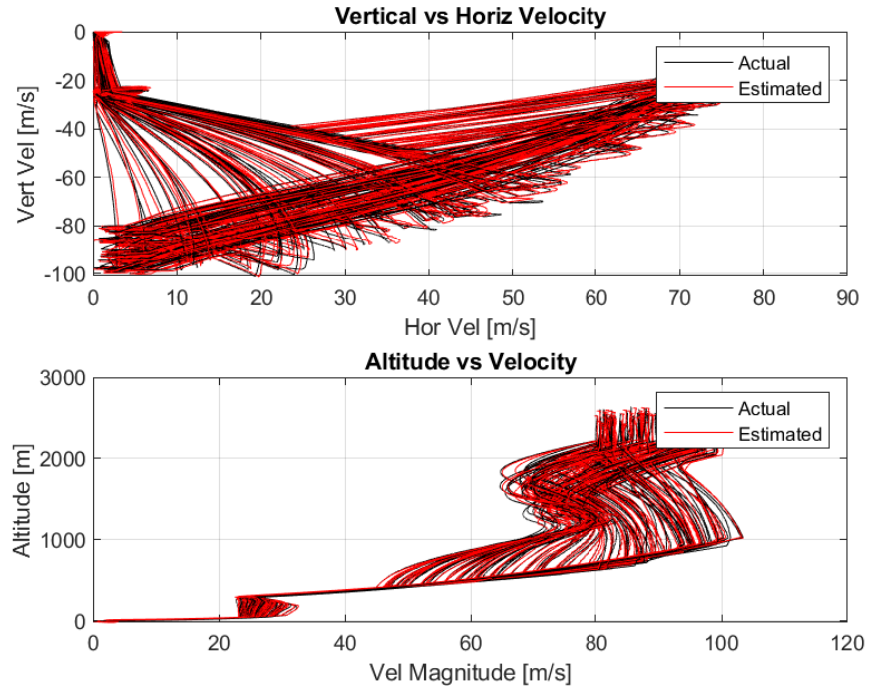


Figure 9.11: Vertical vs Horizontal velocity (top), Altitude vs Velocity (bottom)

9.2 Performance analysis in the absence of RDA sensor

The same as the previous section is done here, but analyzing the sensor configuration without the doppler functionalities of the radar (only the altimetric data is provided by the Radar).

Table 9.4: Simulation Sensor suite configuration and settings

Configuration in the presence of IMU, ALT and LVS	
Sensor	**Status setting**
IMU	*ACTIVE*
Altimeter	*ACTIVE*
Radar Doppler	*NO*
LVS	*ACTIVE*

Table 9.5: $NORDA_{150}$: Simulation Results in the absence of RDA

Metric	**Result (out of 100 simulations)**
horizontal speed $<$ 3.0m/s	*96%*
horizontal speed $<$ 2.5m/s	*95%*
horizontal speed $<$ 2.0m/s	*84%*
vertical speed $<$ 4.0m/s	*100%*
vertical speed $<$ 3.5m/s	*100%*
vertical speed $<$ 3.0m/s	*100%*
mean distance to target at touchdown	*70.28m*

Table 9.6: $NORDA_{300}$: Simulation Results in the absence of RDA

Metric	Result (out of 100 simulations)
horizontal speed < 3.0m/s	*94%*
horizontal speed < 2.5m/s	*87%*
horizontal speed < 2.0m/s	*74%*
vertical speed < 4.0m/s	*100%*
vertical speed < 3.5m/s	*100%*
vertical speed < 3.0m/s	*100%*
mean distance to target at touchdown	*75.36m*

In both configurations, the Radar provides just basic altimetric data, and no doppler capabilities are present. The $NORDA_{150}$ is significantly more precise and performs with an higher success rate, compared to the $NORDA_{300}$ configuration.

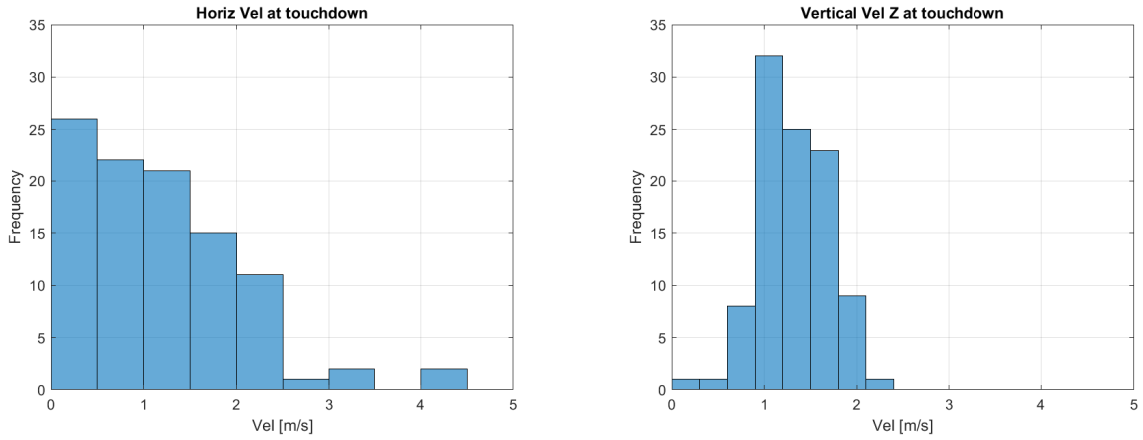


Figure 9.12: Horizontal and Vertical Speed at Touchdown

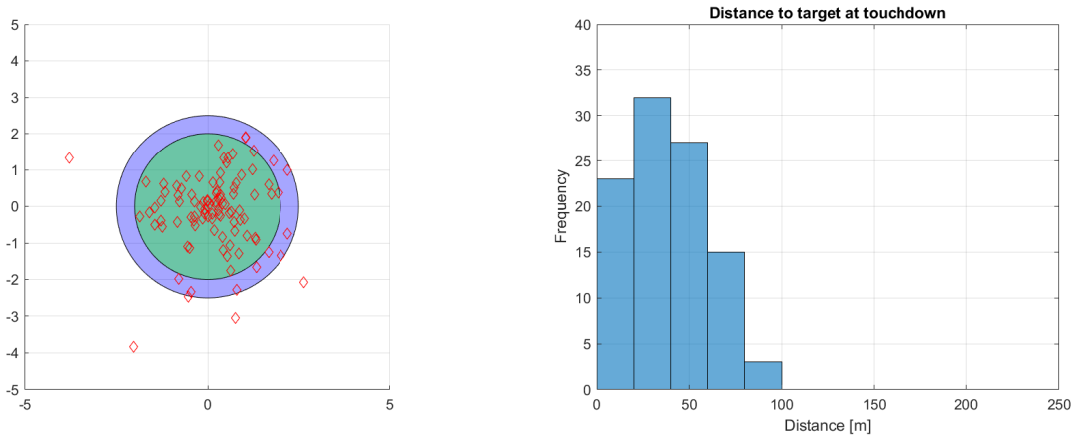


Figure 9.13: Horizontal and Vertical speeds (2m/s and 2.5m/s thresholds represented as circles) (left), Distance to target at Touchdown (right)

Detailed Performance (in the absence of RDA sensor)

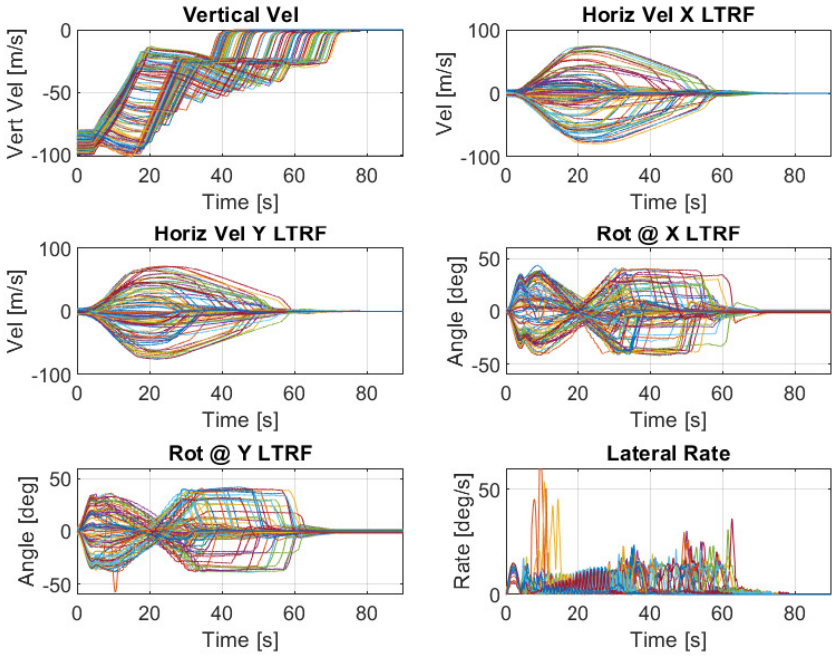


Figure 9.14: Velocities and Angular rates

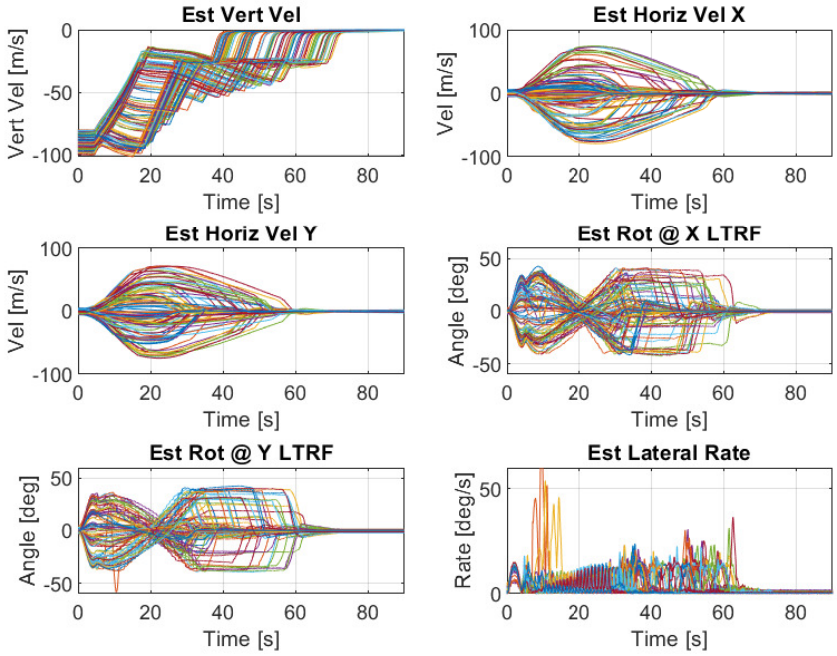


Figure 9.15: Estimated Velocities and Angular rates

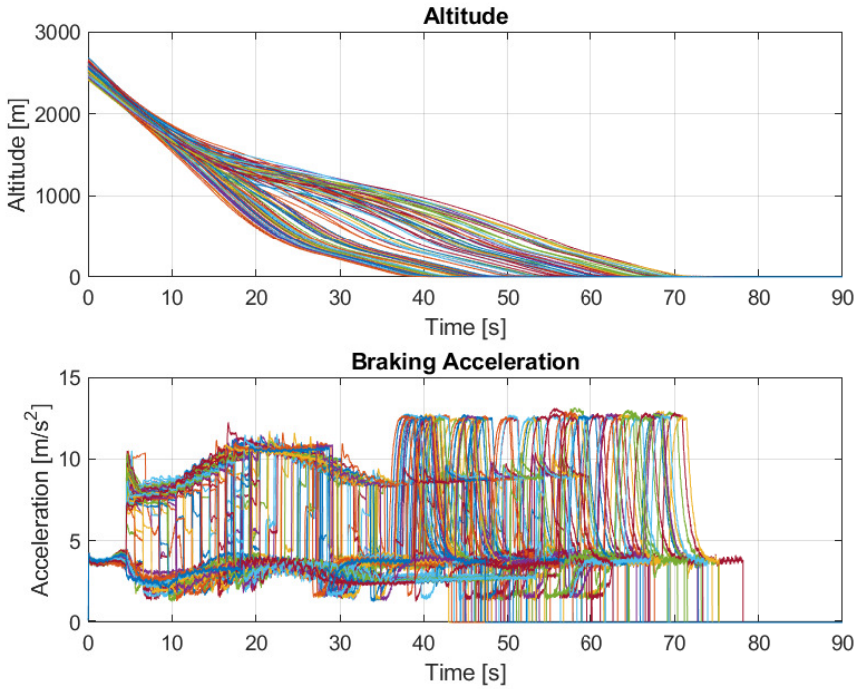


Figure 9.16: Altitude and Breaking acceleration

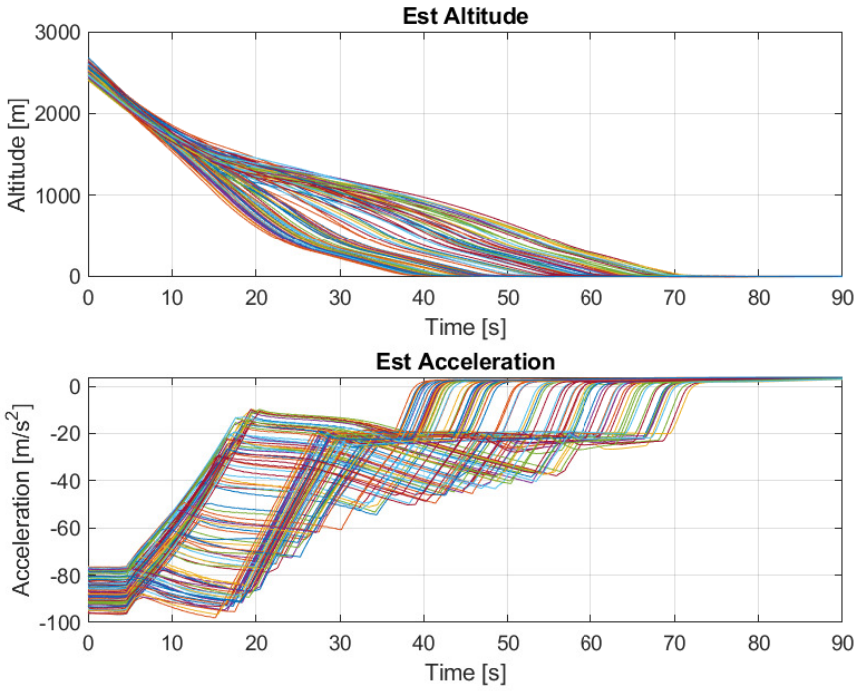


Figure 9.17: Estimated altitude and Estimated acceleration

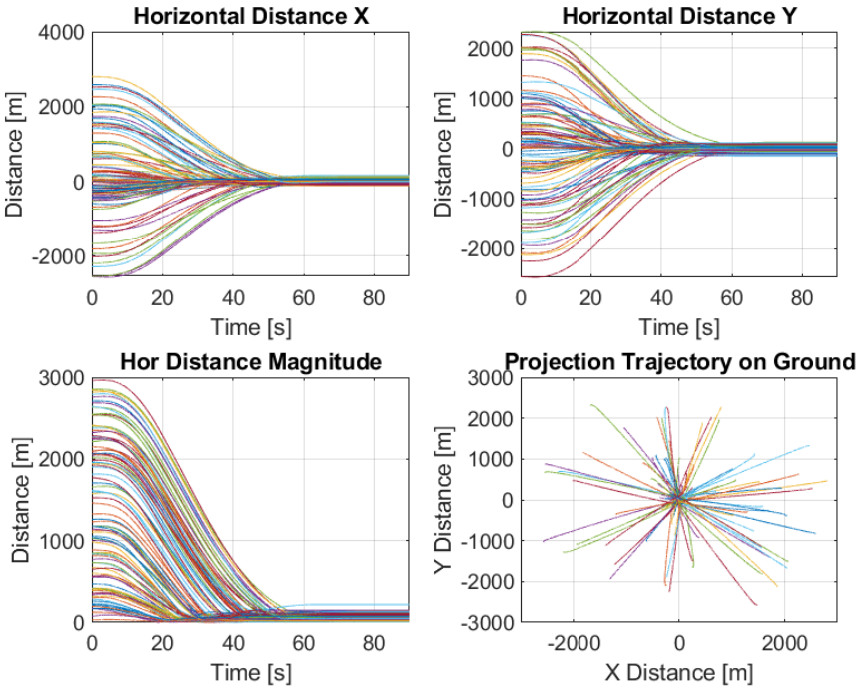


Figure 9.18: Horizontal distance to target

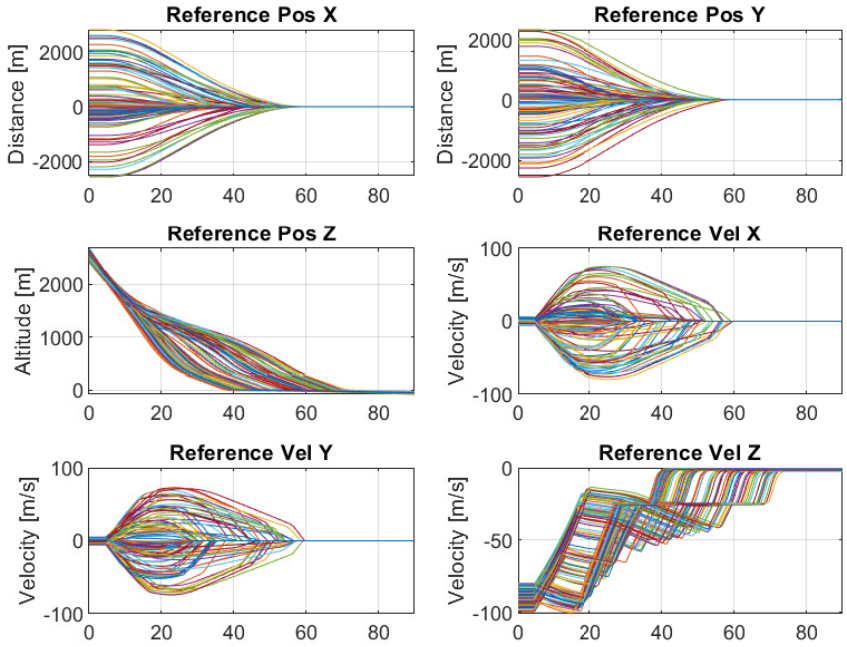


Figure 9.19: Guidance reference position and velocity

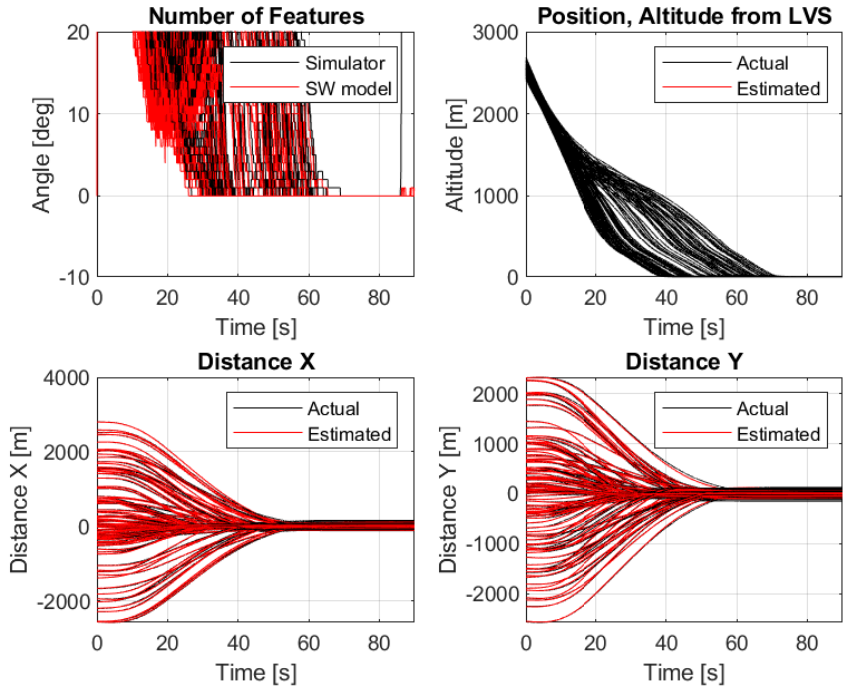


Figure 9.20: LVS number of features and altitude (top), Distance to target (bottom)

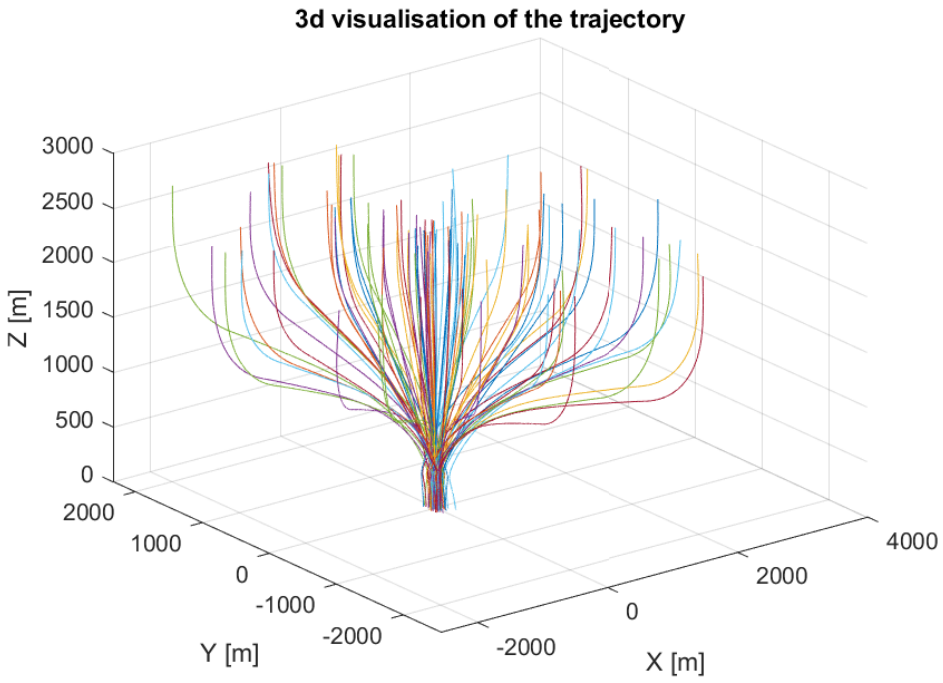


Figure 9.21: Trajectories visualization (3D)

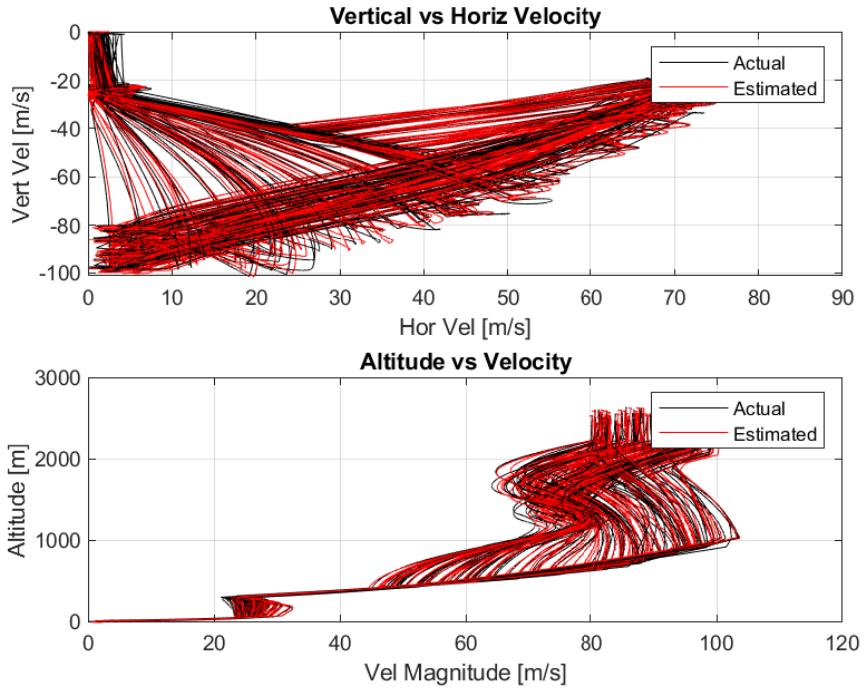


Figure 9.22: Vertical vs Horizontal velocity (top), Altitude vs Velocity (bottom)

9.2.1 Lateral rate comparison in the presence or absence of RDA sensor

Upon analyzing figures 9.14 and 9.15 (in the No-RDA configuration results), a noticeable lateral velocity, transversal to the longitudinal axis, exhibits a somewhat erratic behavior, particularly evident in the orange and yellow traces in the bottom-right subplot. This behavior starkly contrasts with the smoother trajectories depicted in Figures 9.2 and 9.3, corresponding to RDA-inclusive scenarios.

This observation underscores a significant aspect in the RDA versus no-RDA comparison: The presence of RDA, even mild gains are enough to guarantee good performance, ensuring a reduction in the likelihood of abrupt instabilities. Conversely, in the absence of RDA, the system necessitates potentially more precarious gains, posing risks of incipient instability.

Thus, this distinction highlights the important role of RDA in mitigating instability risks and promoting higher success rates.

Chapter 10

Conclusion and Future work

10.1 Conclusion

In this study, the feasibility of achieving precise pinpoint landing on target celestial bodies (particularly the Moon and Mars) is investigated, without the need of doppler functionalities, with a radar capable only of providing altimetric data. The approach is based on implementing a landmark-based Landing Vision System architecture for navigation and integrating a Kalman filter for data fusion and translational dynamics state estimation. Through development, experimentation and analysis, the following conclusions were found:

(i) High Mission Success Rate: Our findings demonstrate a 96% success rate in mission landings, even in the absence of radar Doppler functionalities. This success underscores the efficacy of the Kalman filter for precise landing.

(ii) Primary Failure Causes: Failures, although low in number, stemmed from horizontal velocities exceeding 3m/s, indicating a need for further refinement in horizontal position measurement quality from the LVS sensor or the introduction of a third catalog ad hoc for proximity conditions to terrain. The $NORDA_{150}$ performed significantly better than $NORDA_{300}$, while going lower than 150 for LVS tuned variance decreases the success rate. This means that if higher success rates are needed, the LVS output precision must be enhanced with advanced techniques such as using projected ellipses characteristics or taking into account the known and measured radius difference as additional metrics. However, it's notable that vertical velocity and landing precision requirements were consistently met.

(iii) Performance Reduction without RDA: Despite the overall success rate, the analysis reveals a performance reduction. The mean distance to the landing target at touchdown is approximately 70% higher in landing precision when excluding the radar Doppler, going from a mean distance to target or around 40m with RDA enabled, to around 70m. This emphasizes the importance of RDA functionalities in achieving optimal landing precision, albeit our approach still demonstrates significant capability even without it. Thus, the choice of using or not a Radar Doppler in the sensor

configuration for a real mission will depend on the specific requirements, objectives and needed margins of mission success.

It is also evident from the simulation results that utilizing Radar Doppler Altimetry (RDA) offers significant advantages over simple altimeters.

(i) Improved touchdown speed performance allows for lighter mechanical structures and legs, enhancing overall mission efficiency.

(ii) The ability to effectively integrate horizontal position data (even in temporary absence of LVS measurements) ensures navigation continuity and reliability, even under challenging conditions.

RDA facilitates easier tracking restoration by utilizing true velocity integration, mitigating the risk of substantial position errors compared to reconstructed velocity methods. This is crucial in scenarios where tracking is temporarily lost, as it minimizes the likelihood of significant discrepancies between known positions and actual target locations upon tracking reestablishment.

10.2 Future Work

Building upon the insights gained from this study, several avenues for future research and development emerge:

(i) LVS horizontal position output quality enhancement: Further investigation into techniques for more estimation of horizontal position, leading to better estimation of horizontal speeds during descent. This could mitigate the primary cause of mission failures identified in this study and may involve exploring advanced matching algorithms, pose-determination algorithms or refining the preprocessing phase as a whole.

(ii) LVS orientation estimation capability: Expanding the LVS range of capabilities to estimate the spacecraft orientation and merge this data with the orientation information provided by the laser-gyro in the on-board IMU. Precision obtained by the IMU should be higher and the LVS could not be able to increase the orientation dynamics estimation.

(iii) Advanced filtering: Although the kalman filter seems not to be the bottleneck of precise pinpoint landing performance, an upgrade to advanced non-linear filtering techniques can be beneficial. Extended Kalman Filters, Unscented Kalman Filters of Particle Filters are all good candidates for improvements in this part of the navigation.

By addressing these areas of future work, we can continue to advance the field of precise pinpoint landing on other planets and celestial bodies, leading to a new era of scientific discovery and planetary exploration.

Part V

Appendix

Appendix A

Reference Frames Conventions

The following reference frames convenctions and specifications were taken from the MML study on the Italian Space Agency.

A.1 Target Planet Reference Frame

Local Vertical Local Horizontal (LVLH)/ENU Reference Frame

The common reference frame is the LVLH (Local Vertical Local Horizontal), in the ENU (East, North, Up) variant.

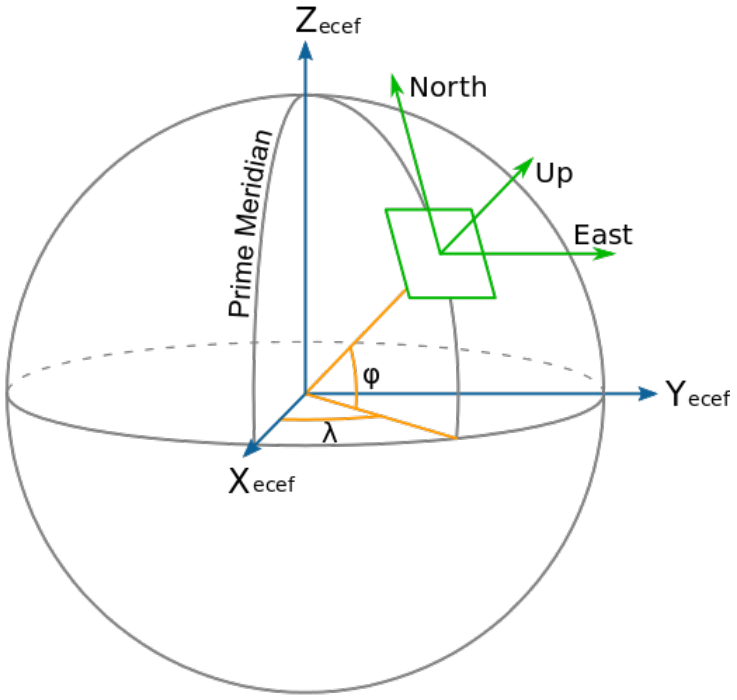


Figure A.1: Local Vertical Local Horizontal (LVLH). East, North, Up (ENU) variant.

A.2 Spacecraft Composite Reference System (SCRF)

The Spacecraft Composite Geometrical Reference Frame is used for defining the geometrical configuration of the composite; it is defined as follows:

1. O_{SC}: origin located on the Spacecraft Composite/Launcher separation plane at the centre of the Spacecraft interface ring
2. Z_{SC}: orthogonal to the Spacecraft/Launcher separation plane, pointing positively from the separation plane towards the Entry Vehicle nose
3. X_{SC}: in horizontal plane, positive in the direction from the Radar Doppler origin to the Robotic Egress Arm
4. Y_{SC}: completing the right handed coordinate system

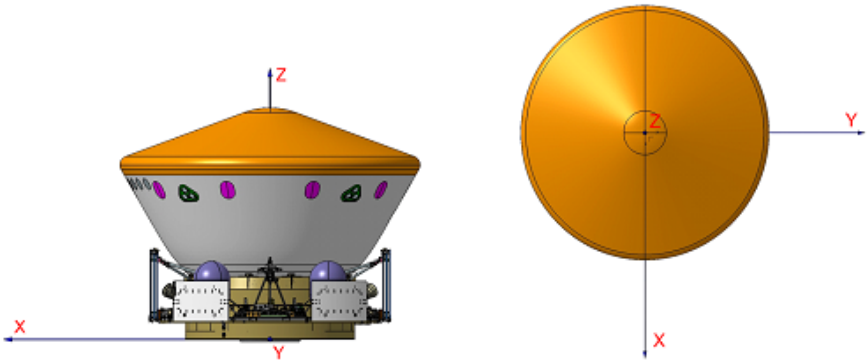


Figure A.2: S/C Geometrical Reference Frame

A.2.1 Spacecraft Composite Body Reference System (SCBRF)

This frame is parallel to the SCGRF and located in the center of mass of the composite.

A.3 Entry Vehicle Mechanical Coordinate Frame (EV, EVGRF)

The EV Geometrical Reference Frame (EVGRF) Frame is used for defining the geometrical configuration of the DM; it is defined as follows:

1. O_{EV}: the origin, is located on the EV/CS separation plane at the intersection with the revolution axis of the EV conical shape (i.e. located on the EV mounting plane coinciding with the geometrical center of circumference which goes through centers of fastening elements for the separation system

2. Z_{EV}: orthogonal to the EV mounting plane of the Separation Assembly, pointing positively toward the Heat Shield nose. Same orientation and direction of $+Z_{SC}$
3. X_{EV}: orthogonal to the $+Z_{EV}$ axis and parallel to the $+X_{SC}$ of the SCGRF
4. Y_{EV}: completing the right handed coordinate system

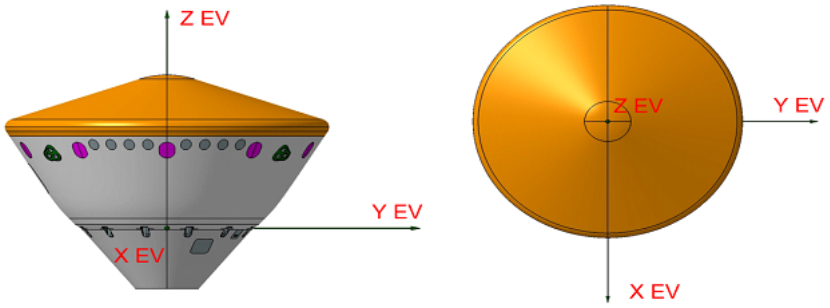


Figure A.3: Entry Vehicle Geometrical Reference Frame

A.3.1 Entry Vehicle Body Reference System (EVBRF)

This frame is parallel to the EVGRF and located in the center of mass of the composite.

A.4 Landing Platform Coordinate Frame (subscript LP)

The Landing Platform Mechanical Reference Coordinate (LP) shall be a right-handed, orthogonal coordinate system used for geometrical configuration, design drawings and dimensions, and defined as follows:

1. O_{LP}: the origin, is located on the Landing Platform/Backshell separation plane at the center of the top deck circular cut out.
2. Z_{LP}: axis is orthogonal to the upper plane of the Landing Platform, pointing positively toward the Front Shield nose. Same orientation and direction of $+Z_{EV}$
3. X_{LP}: orthogonal to the $+Z_{LP}$ axis with the same orientation and direction of $+X_{EV}$
4. Y_{LP}: completing the right handed coordinate system

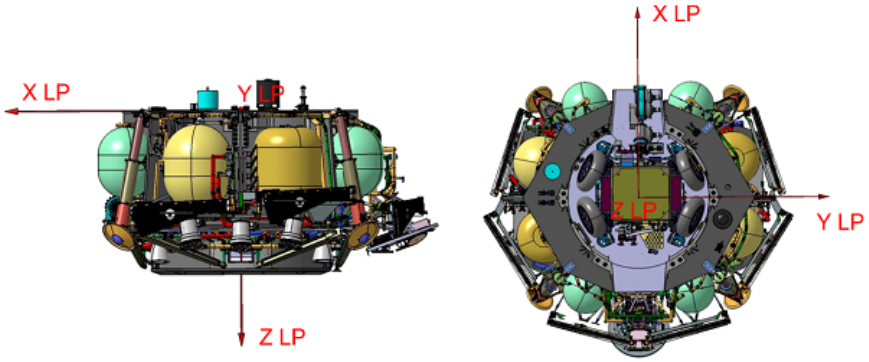


Figure A.4: Landing Platform Reference Frame

A.5 Other GNC Related Reference Frames

Besides the LVLH/ENU used for landing seen at A.1, there are other reference frames used by the GNC [11].

A.5.1 On orbit Local Vertical Local Horizontal Frame of the GNC (GNC_LOR)

This frame is devoted to the RDA-based navigation

1. O_{GNC_LOR}: the origin, is located at the center of mass of the DM. The axes are defined as hereafter described.
2. Z_{GNC_LOR}: in the orbital plane from the Mars center to the spacecraft CoG
3. Y_{GNC_LOR}: normal to the orbital plane with the direction of the orbital angular momentum vector. Its direction is defined by the extern product of the inertial position and velocity vectors
4. X_{GNC_LOR}: It is in the same half-plane of the inertial velocity vector of the spacecraft

A.5.2 Local Vertical Local Horizontal Frame on Ground at RIL (GNC_L)

The GNC_L frame is a frame defined like the GNC_LOR, but with origin translated down along the Z_{GNC_LOR} so that it remains located on the Mars surface in the instant, triggered by $Radar\_In\_The\_Loop$ (RIL) event, in which the Radar is included in the navigation loop (transition from the intermediate descent mode to the terminal descent mode). Axes are parallel to the ones of GNC_LOR frame, for the specified mentioned instant.

1. O_{GNC_L}: the origin, is located on the Mars surface. The axes are defined as hereafter described

2. Z_{GNC_L}: represent the Local Vertical unit vector at the location of the spacecraft CoG

3. Y_{GNC_L}: normal to the orbital plane with the direction of the orbital angular momentum vector. Its direction is defined by the extern product of the inertial position and velocity vectors

4. X_{GNC_L}: Completes the triad. It is in the same half-plane of the inertial velocity vector of the spacecraft and tangent to the surface under the hypothesis of spherical planet

A.5.3 Local Terrain Frame, East-North-Up at the Target location (GNC_ENU, LTF)

The GNC_ENU frame is a surface reference frame applicable for the Landing. It does correspond to ENU frame at the Target landing location.

1. O_{GNC_ENU}: the origin, is located at the landing target location

2. X_{GNC_ENU}: East oriented

3. Y_{GNC_ENU}: North oriented

4. Z_{GNC_LRF}: Up oriented (direction nadir-zenith)

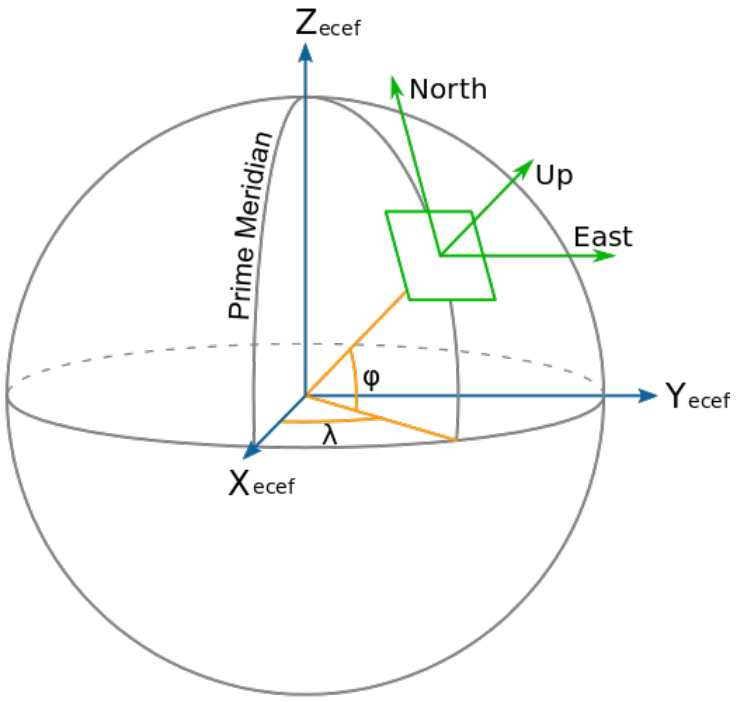


Figure A.5: Local Vertical Local Horizontal (LVLH). East, North, Up (ENU) variant.

A.5.4 Body Reference Frame of the GNC for Landing (subscript B_L)

The Body reference frame of the GNC landing is a reference frame with axes parallel to the EV/LP reference frame axes with the following characteristics:

1. O_{B_L}: the origin, is located at the EV/LP center of mass, assuming the best knowledge available. This location will be phase-depending based on the applicable configuration.
2. X_{B_L}
3. Y_{B_L}
4. Z_{B_L}

The transformation from the EV/LP to the Body landing reference frame is identified by a rotation of 180° around the YEV (or YLP) axis and by a translation along the symmetry axis. The amount of this translation is depending on the specific configuration (CoG location) of the EV and LP in the identified instant.

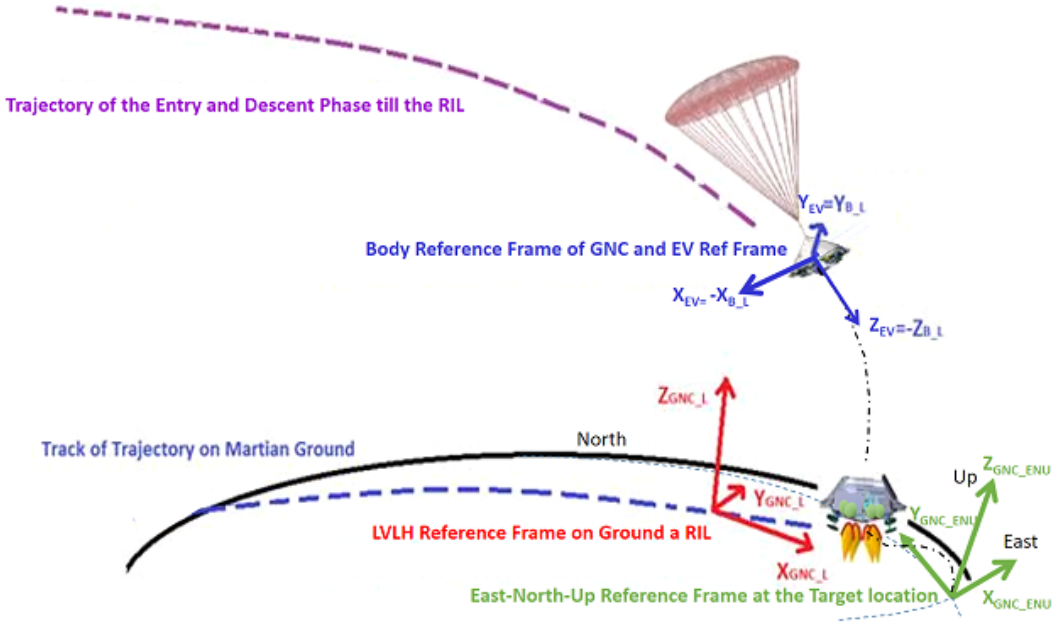


Figure A.6: EDL Reference Frames

Appendix B

Navigation Translational Filter blocks and models

B.1 Sensor data preparation

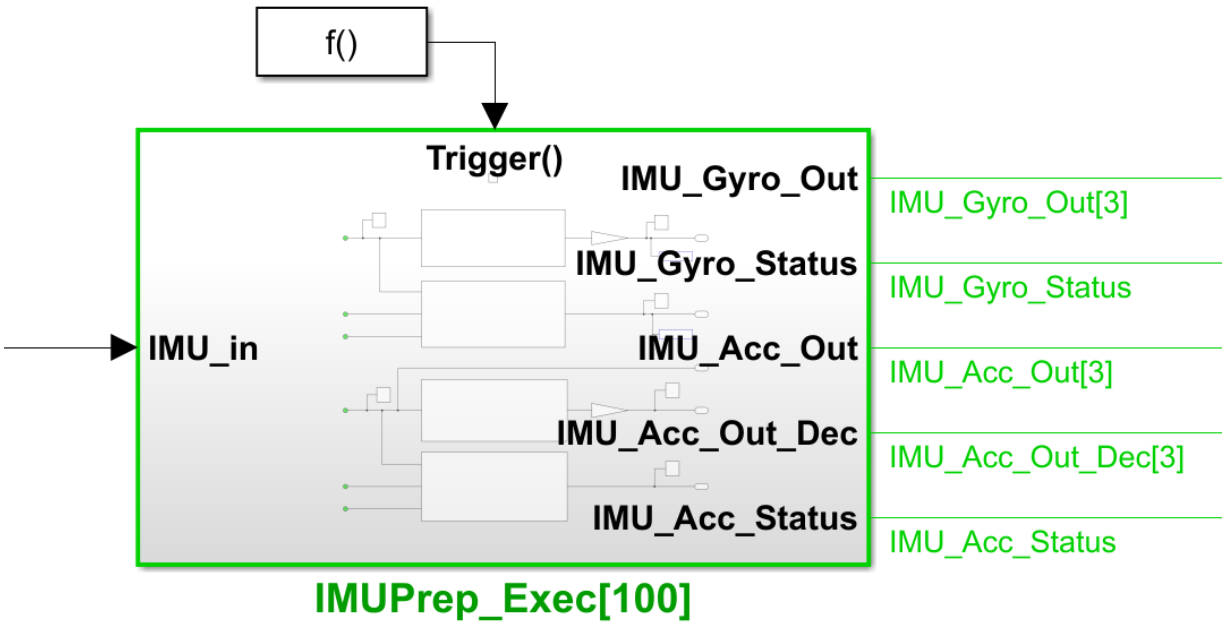


Figure B.1: IMU Data Preparation for KF

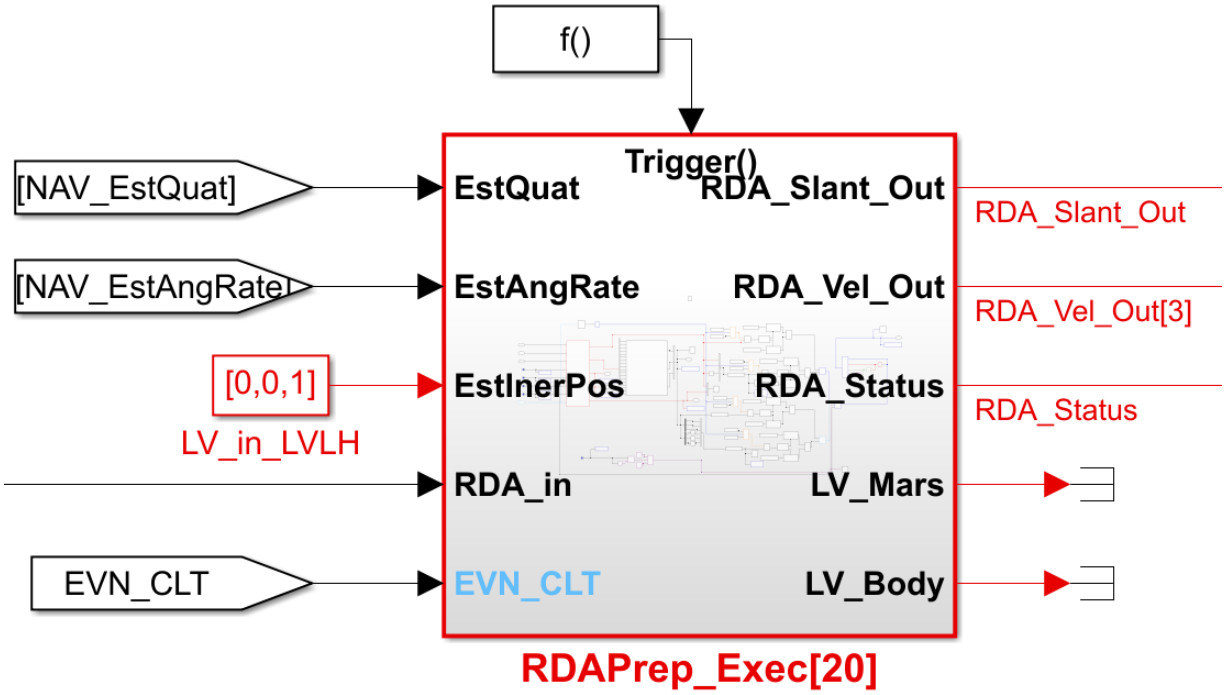


Figure B.2: RDA Data Preparation for KF

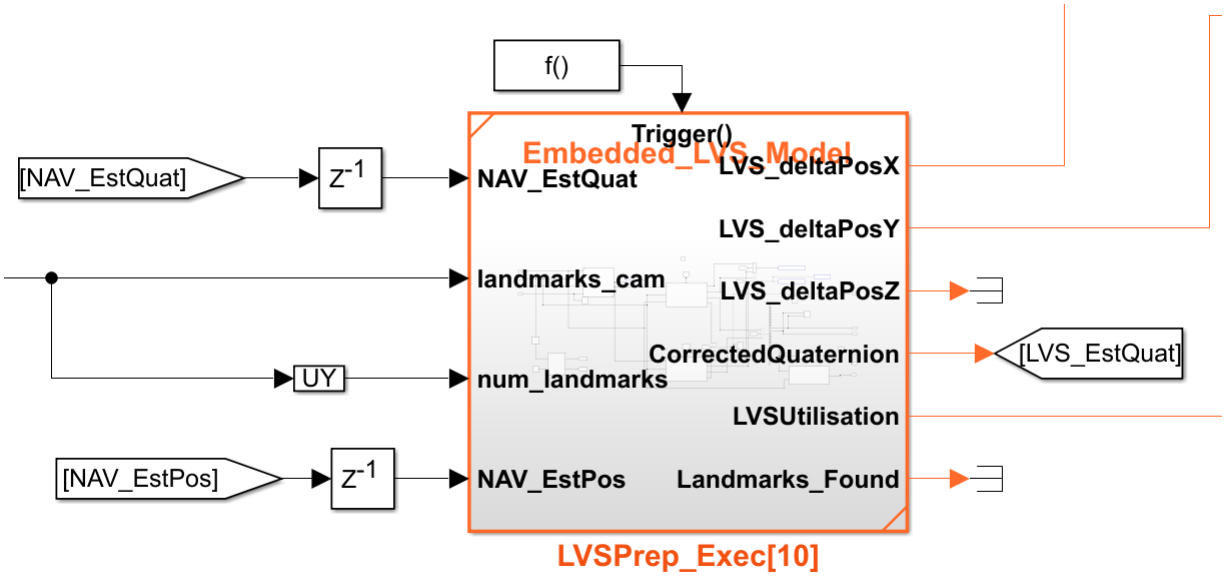


Figure B.3: LVS Data Preparation for KF

B.2 Axis Filters

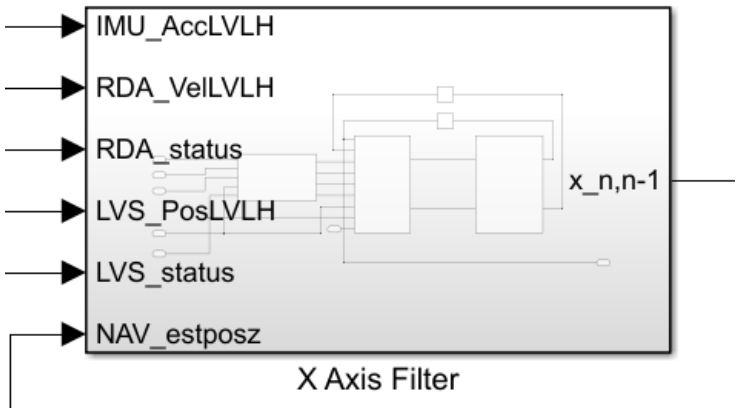


Figure B.4: X Axis Kalman Filter

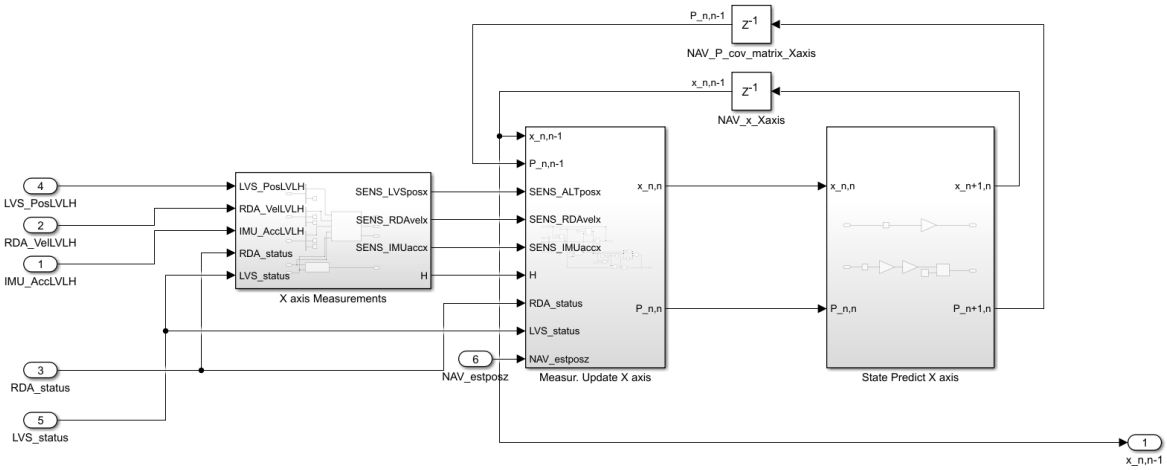


Figure B.5: X Axis Kalman Filter internals

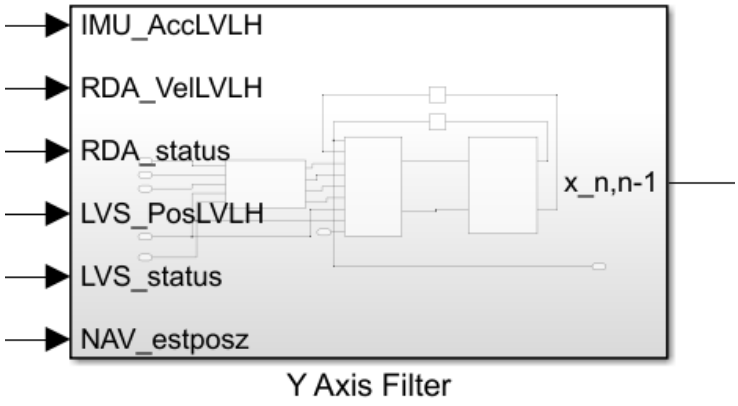


Figure B.6: Y Axis Kalman Filter

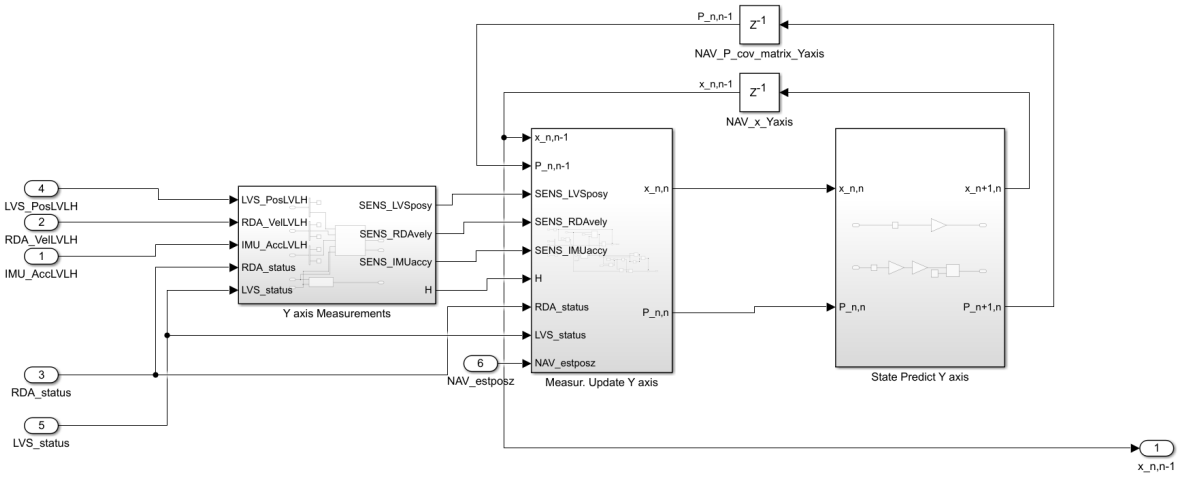


Figure B.7: Y Axis Kalman Filter internals

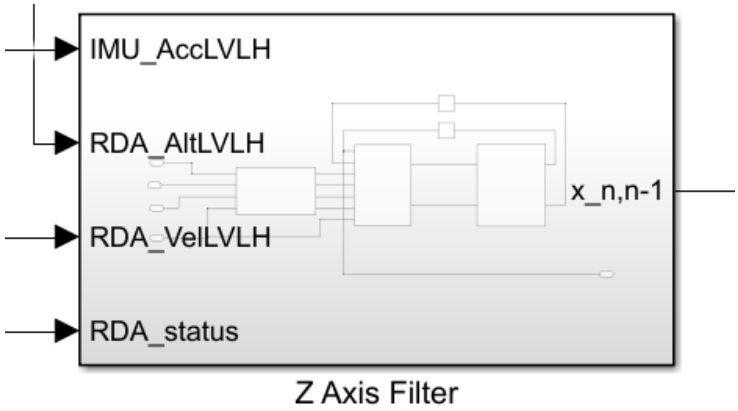


Figure B.8: Z Axis Kalman Filter

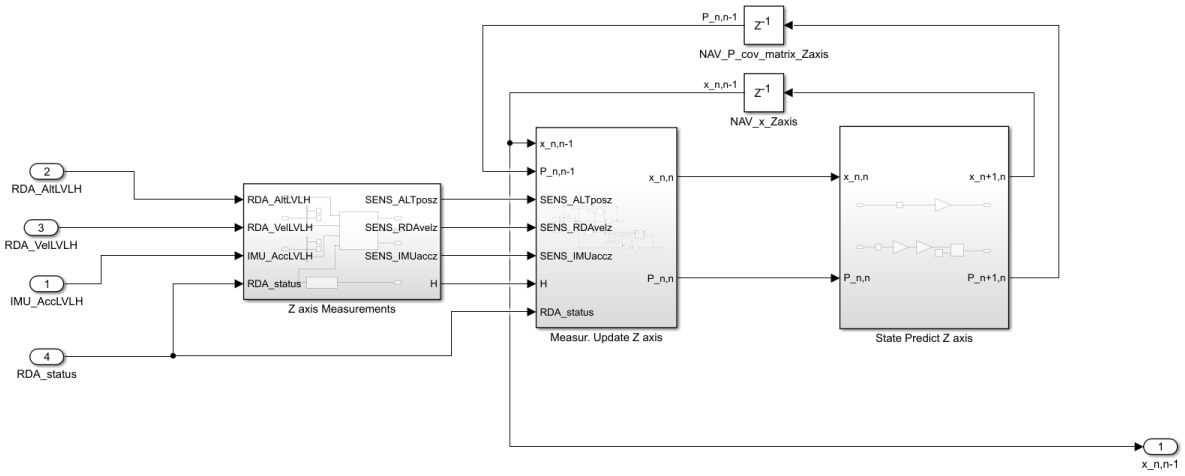


Figure B.9: Z Axis Kalman Filter internals

Appendix C

Monte Carlo Simulations results details

C.1 Sensor Configuration in the presence of RDA

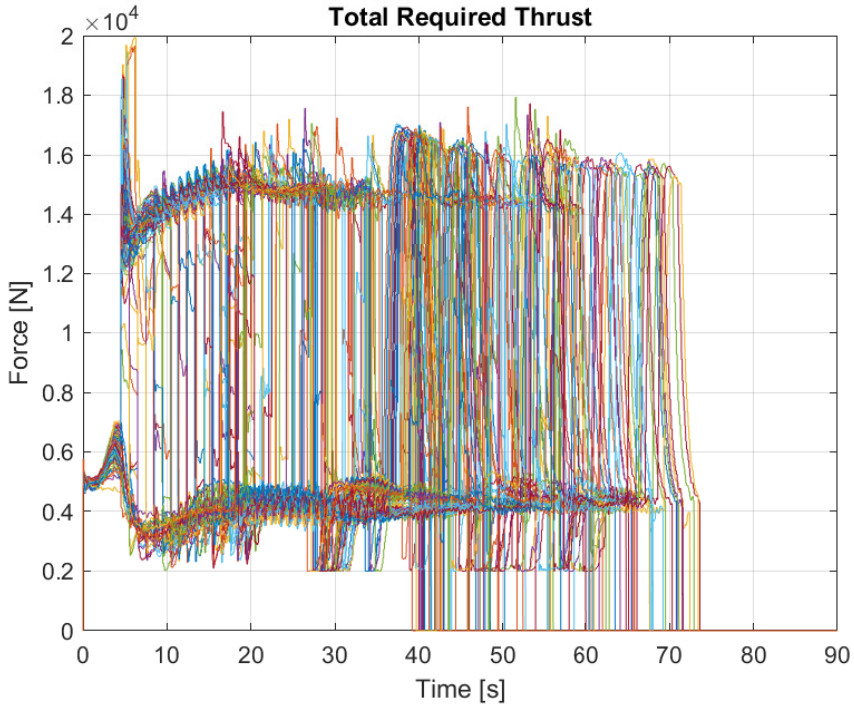


Figure C.1: Total required thrust

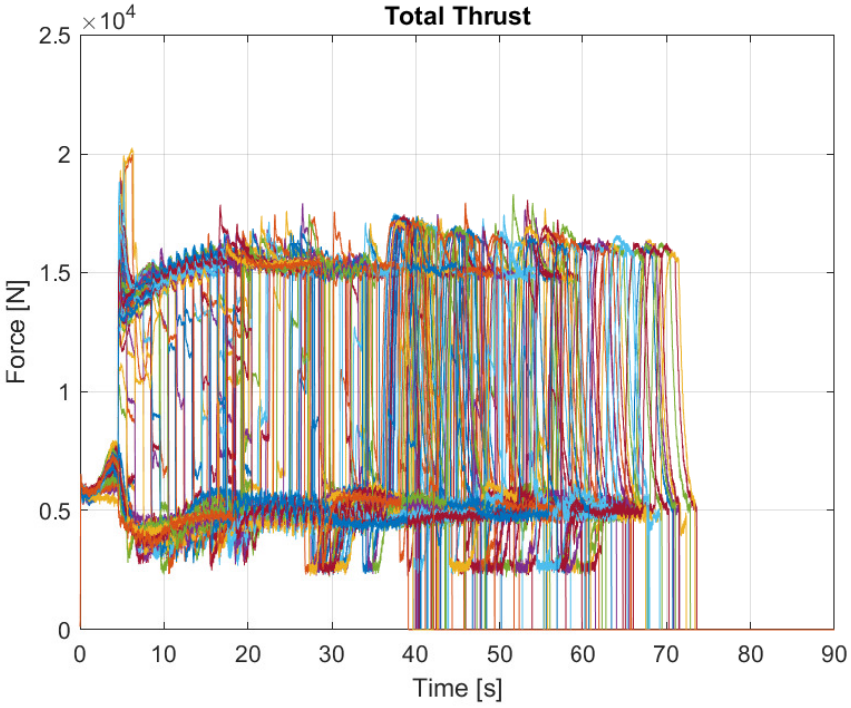


Figure C.2: Total thrust

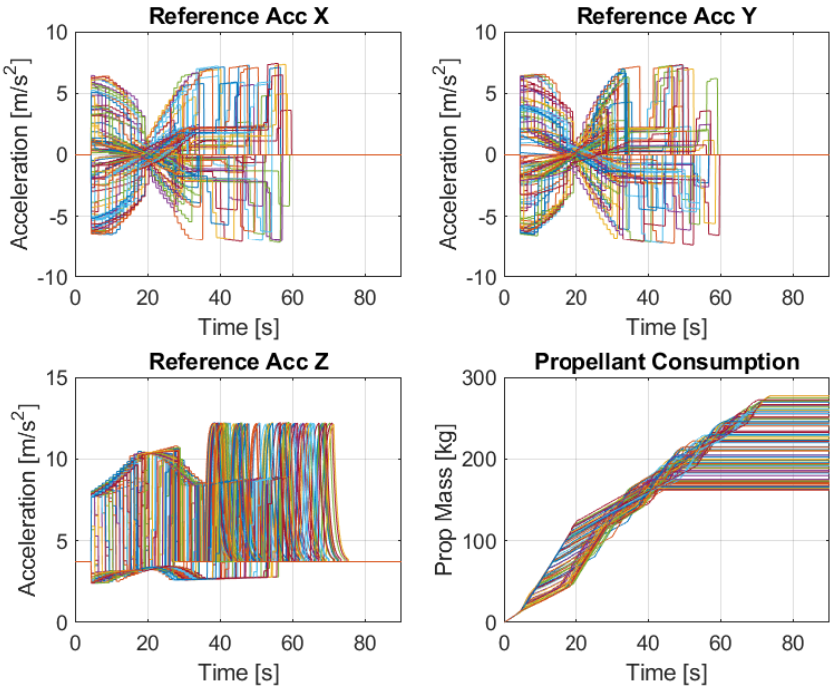


Figure C.3: Reference Acceleration and propellant consumption

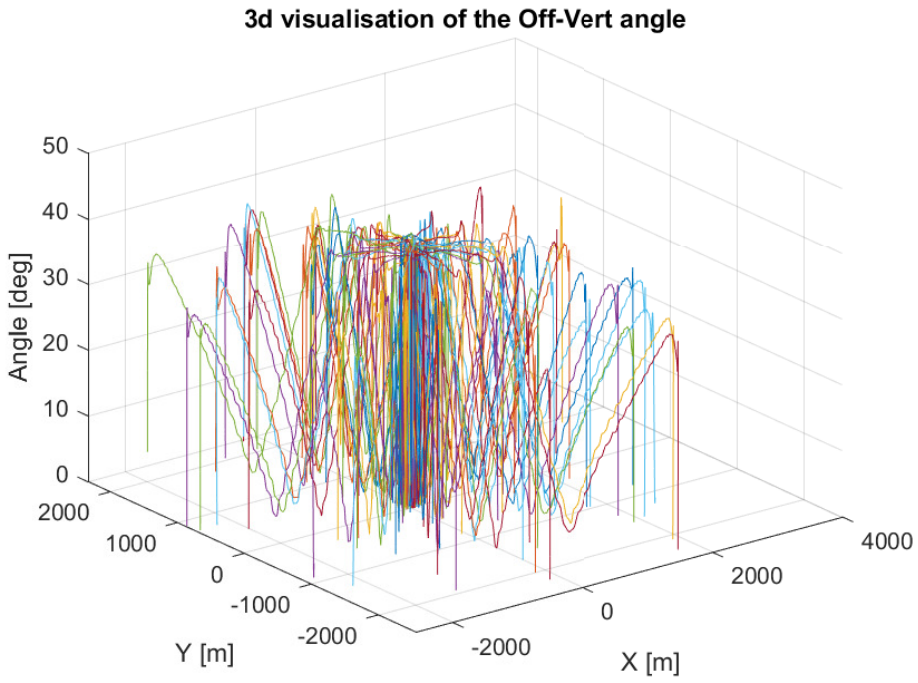
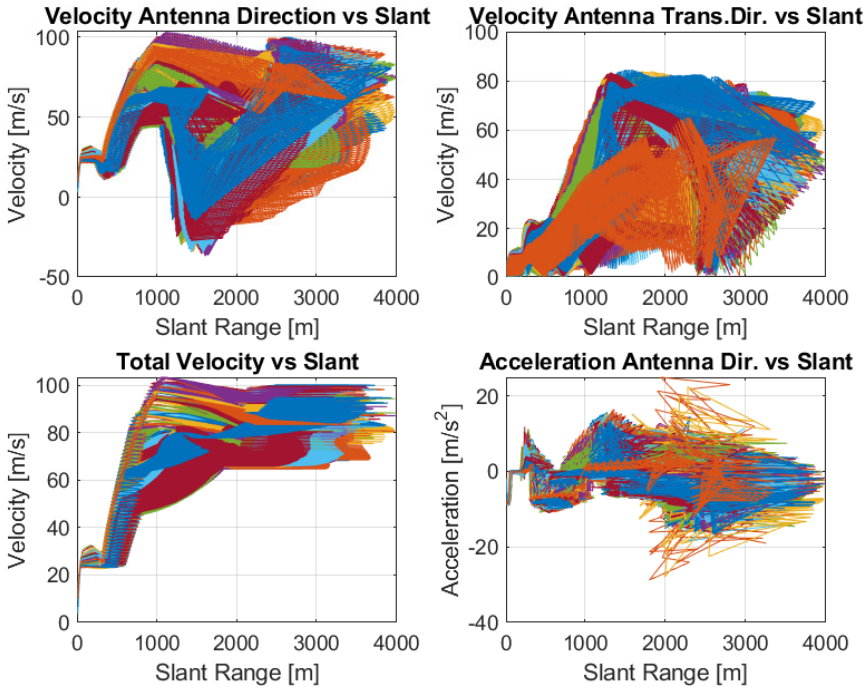
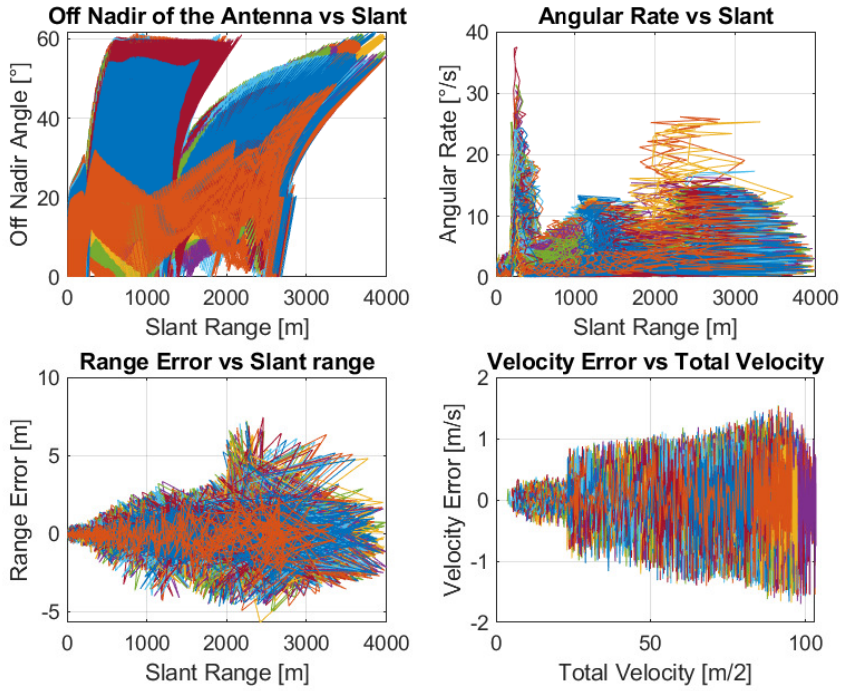


Figure C.4: Off-vertical angle vs horizontal xy position





C.2 Sensor Configuration in the absence of RDA

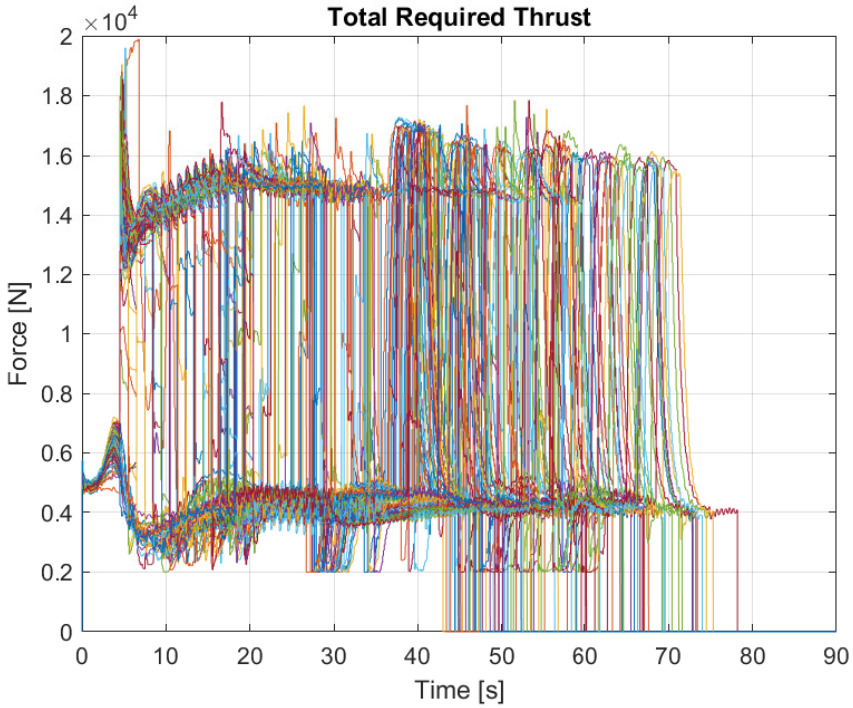


Figure C.5: Total required thrust

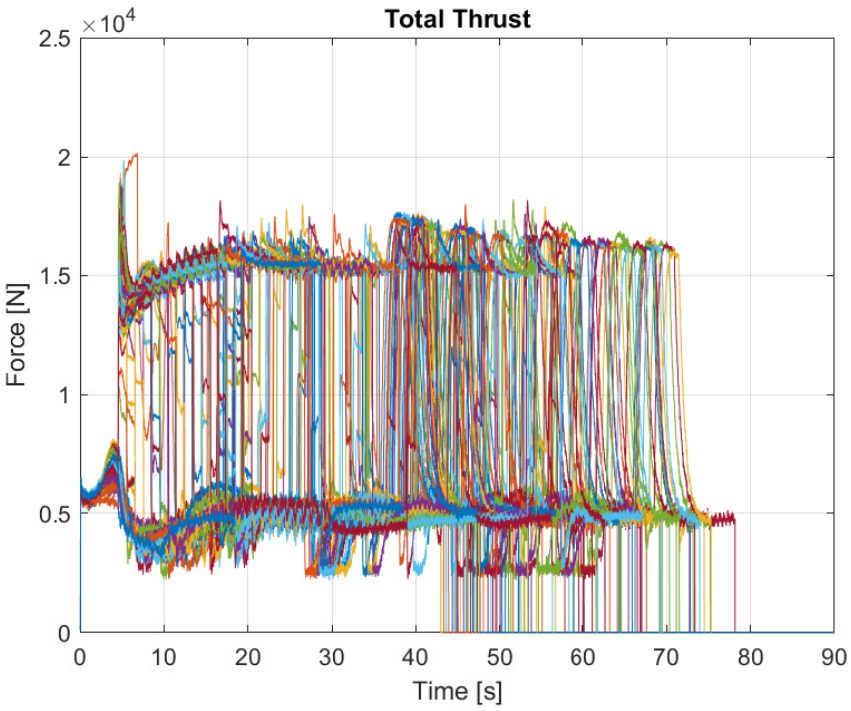


Figure C.6: Total thrust

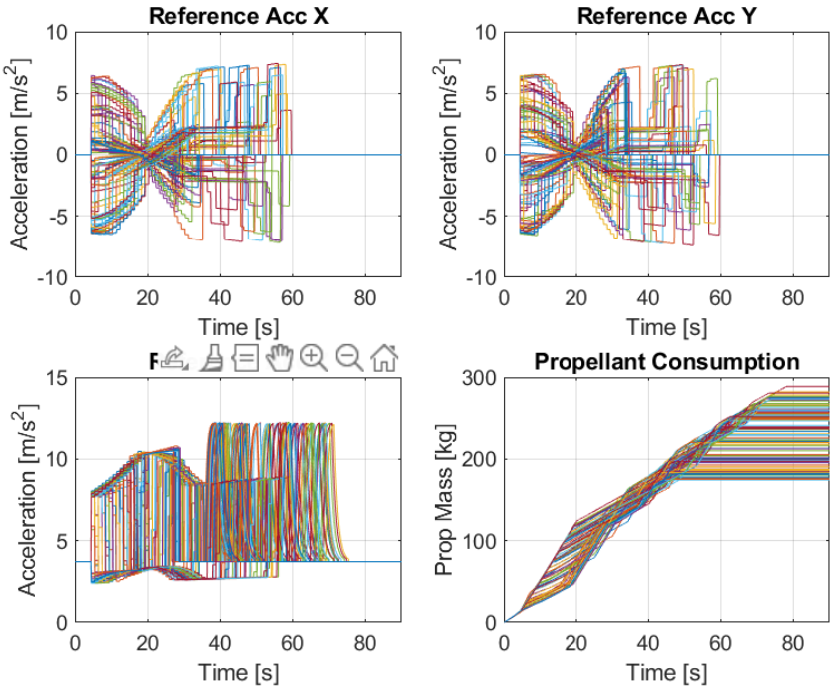


Figure C.7: Reference Acceleration and propellant consumption

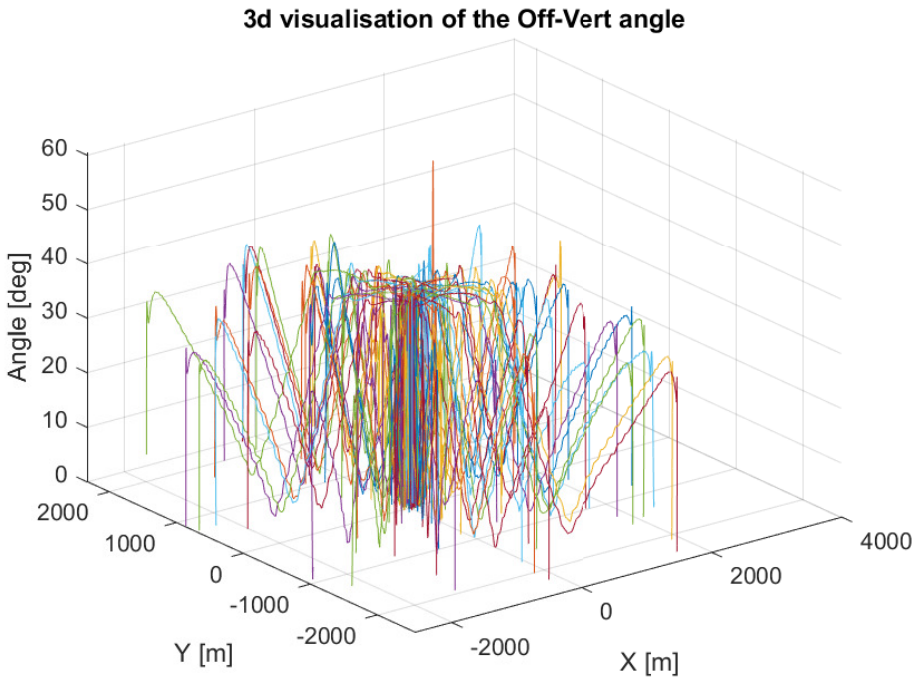
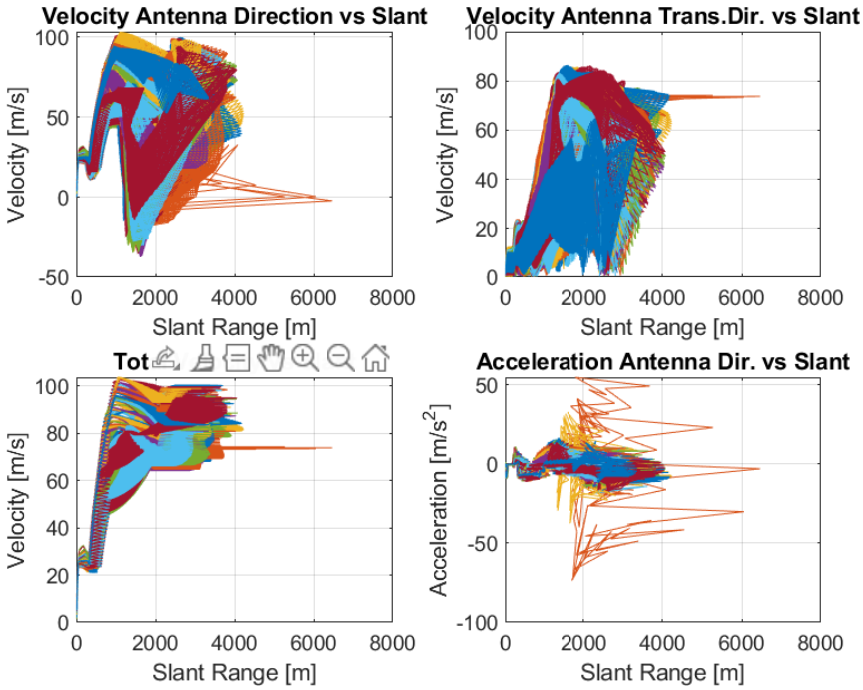
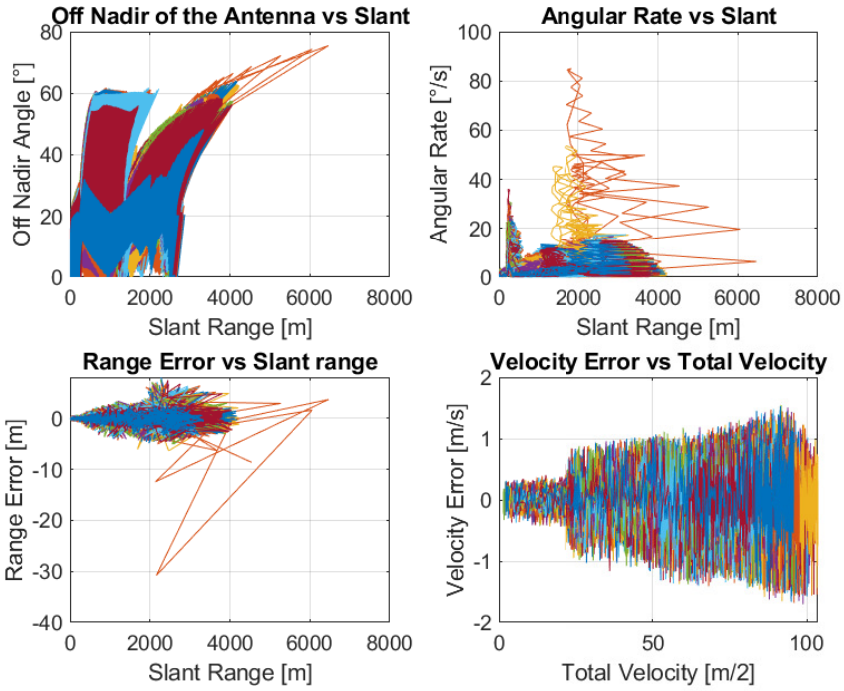


Figure C.8: Off-vertical angle vs horizontal xy position




Off Nadir of the Antenna vs Slant
Off Nadir Angle [°]
Slant Range [m]
Angular Rate vs Slant
Angular Rate [°/s]
Slant Range [m]
Range Error vs Slant range
Range Error [m]
Slant Range [m]
Velocity Error vs Total Velocity
Velocity Error [m/s]
Total Velocity [m/2]

List of Figures

List of Tables

Bibliography

[1] James F. Montgomery Andrew E. Johnson, Yang Cheng. Real-time terrain relative navigation test results from a relevant environment for mars landing. *AIAA Guidance, Navigation, and Control Conference*, 5(9), 2015.

[2] James F. Montgomery Andrew E. Johnson, Yang Cheng. Results from a relevant environment for mars landing. *AIAA Guidance, Navigation, and Control Conference*, 5(9), 2015.

[3] Alex Becker. *Kalman Filter from the Ground Up*. 2023.

[4] Willem M. Oliveira Bolko Maass, Svenja Woicke. Crater navigation system for autonomous precision landing on the moon. *Journal of Guidance, Control, and Dynamics*, 43(8):1414–1431, 2020.

[5] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[6] Ralph D. Lorenz. Planetary landings with terrain sensing and hazard avoidance: A review. *AIAA Guidance, Navigation, and Control Conference*, 71(1):1–15, 2023.

[7] Bolko Maass. Robust appoximation of image illumination direction in a segmentation-based crater detection algorithm for spacecraft navigation. *CEAS Space Journal*, 8:303–314, 2016.

[8] Behzad Moshiri Mojtaba Kordestani, Maryam Dehghani. A new fusion estimation method for multi-rate multi-sensor systems with missing measurements. *IEEE Access*, pages 47522–47532, 2020.

[9] Andrew Zisserman Richard Hartley. *Multiple View Geometry*. Cambridge University Press, 2003.

[10] Hyonam Joo Robert Martin Haralick. Pose estimation from corresponding point data. *IEEE Transactions on Systems Man and Cybernetics*, 19(6):1426–1446, 1989.

[11] Jürgen Müller Wolfgang Torge. *Geodesy*. DE GRUYTER, 2012.