Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale

A.a. 2024/2025

Sessione di Laurea Marzo/Aprile 2025

Design of an optimization tool for Contamination and Collision Avoidance Manoeuvre of Vega-C launcher

Relatore Accademico:

Prof.ssa Elisa Capello

Relatori Aziendali:

Ing. Angelo Tomassini

Ing. Christophe Roux

Candidato:

Nikita Fusilli Grynchenko

Dedication

To my family, for your endless support and sacrifices you made to ensure I could focus on my studies without worry. Your belief in me has meant everything.

To my friends from Banda del Pelo, who have been there for me since day one and always made coming home for holidays so special.

To the amazing group from floor 89 in Residenza Borsellino, you made my time in Turin fun, intense, and full of life, with never a dull moment.

To Vincenzo, who helped me through countless long lecture days, always ready to support and share the burden with me.

To Veronica and Matteo, my wonderful housemates in Colleferro, who brought joy and positivity to even the toughest days.

To the GNC group at AVIO, who made me feel part of the team, helped me grow, and welcomed me into the engineering world with kindness and guidance.

And to the one who is no longer with us, but whose memory inspired me to start this journey and continues to be my guiding light. I would not be the person I am today without you.

Acknowledgements

I want to thank my tutor, Angelo Tomassini, for his guidance, advice, and support during my thesis work. His patience, kindness, and welcoming attitude made this process easier and more enjoyable. I am also grateful to Christophe Roux for sharing his knowledge and experience, which helped me learn and improve.

Thank you to Dr. Irene Cruciani for accepting my thesis request and entrusting me with a topic that was both challenging and inspiring.

I also want to thank my academic supervisor, Prof. Elisa Capello, for her dedication to providing her students with exceptional opportunities to work on formative thesis projects. Her efforts in facilitating the means and connections required for this work were invaluable, and her availability for feedback was greatly appreciated.

Abstract

Collision and Contamination Avoidance Maneuver (CCAM) is the flight phase of Vega-C missions directly after the Payload (PL) release, critical for the insurance of the integrity of the released PL. Its main objectives are to avoid the collision of the upper stage with the separated PL in a mid and long-term and to minimize the pollution of the PL by the plume of the launcher's motors (Main Engine and ACS Attitude Control System thrusters). Several aspects of the PL separation greatly affect this manoeuvre, above all the launch vehicle's attitude, PL release direction and the separation mechanism. These aspects, together with specific PL requirements and the possibility of multiple releases on the same orbit, make high variability one of the main characteristics of this flight phase, forcing the mission designer to look for an efficient solution in a custom-made way and making this mission design process highly time-consuming. This work is aimed at presenting a new tool for speeding up the mission design process and making it more efficient. The core of this tool is an open-loop simulator simplified with ad hoc models of the fourth stage of the Vega-C launcher and able to perform Monte Carlo-like campaigns that randomly generate a significant number of manoeuvres that are evaluated and fed to an evolutionary algorithm (Differential Evolution) for the research of one or more optimal solutions. After requesting a set of user-defined inputs and constraints, the tool integrates Hill's equations to simulate the PL trajectory in a local orbital frame, while the attitude of the launcher is computed through quaternion integration. The evaluation of the manoeuvres is based on the requirements of collision and contamination avoidance defined in a cost function. Once generated, the manoeuvres can be plotted and compared with each other. Furthermore, the user can save both input and output as well as the mission data to be used in the main simulator. Among other features, the tool can simulate multiple PL separations, both simultaneous and sequential, giving the user the freedom to define each release independently of the others.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Vega C (Vega Consolidation) is a multi-national European small-lift launch vehicle. It is developed and produced by Avio, which is also its lead manufacturer. Vega C is the evolution of Vega, delivering more performance, a doubled payload volume and better competitiveness. It is a single-body rocket composed of 3 solid-propellant stages, an upper stage equipped with a reignitable liquid engine, and a payload fairing.

Figure 1.1: Vega C on the ELV launch pad, ready for VV25 flight

Vega C is launched from the ELV (Ensemble de Lancement Vega) launch pad at the Guiana Space Centre in Kourou, French Guiana. It operates on polar Sun-synchronous orbits (SSO). Although the flight profile is optimised for each mission, it is possible to define the launch sequence of a typical mission.

Table 1.1: Typical Vega C flight sequence [1]

<table>
<tr><td>P120C ignition & lift-off</td><td rowspan="7">Solid stages ascent phase</td></tr>
<tr><td>P120C burn-out & separation</td></tr>
<tr><td>Zefiro-40 ignition</td></tr>
<tr><td>Zefiro-40 burn-out & separation</td></tr>
<tr><td>Zefiro-9 ignition</td></tr>
<tr><td>Fairing jettisoning</td></tr>
<tr><td>Zefiro-9 separation</td></tr>
<tr><td>1^{st} AVUM+ boost (transfer orbit injection)</td><td rowspan="13">Orbit insertion, PL release and deorbit</td></tr>
<tr><td>Coasting phase</td></tr>
<tr><td>2^{nd} AVUM+ boost (target orbit reached)</td></tr>
<tr><td>1^{st} payload separation</td></tr>
<tr><td>CCAM (Contamination and Collision Avoidance Manoeuvre)</td></tr>
<tr><td>3^{rd} and 4^{th} AVUM+ boosts to change SSO</td></tr>
<tr><td>Separation of a second batch of Small Satellites</td></tr>
<tr><td>CCAM</td></tr>
<tr><td>5^{th} and 6^{th} AVUM+ boosts to change SSO</td></tr>
<tr><td>CCAM</td></tr>
<tr><td>Separation of a third batch of Small Satellites</td></tr>
<tr><td>CCAM</td></tr>
<tr><td>Last AVUM+ boost for deorbiting</td></tr>
</table>

1.1 Objectives

The Contamination and Collision Avoidance Manoeuvre (CCAM) is the flight manoeuvre that directly follows the payload separation. It is designed to:

- Avoid the collision between AVUM+ and the released payload in the following manoeuvres by increasing their relative distance
- Avoid contamination of the separated payload caused by RACS (Roll and Attitude Control System) thrusters and AVUM+ Main Engine plumes
- Allow the launch vehicle to reach the required attitude to provide the necessary DV for a change of orbit or deorbiting.

The CCAM is a critical phase for the success of the mission. The collision between the payload and the fourth stage could cause damage leading to a partial or total loss of payload control and functionality. Moreover, the exhaust plumes, mainly the Main Engine one, could leave deposits derived from the condensation of the gaseous phase as well as from unburnt droplets of propellant and solid particles of Carbon and compounds. These deposits could obstruct sensors and lenses, limiting the payload functionalities, as well as solar panels, leading the satellite to an eventual power cutoff [2][3].

The Contamination and Collision Avoidance Manoeuvre (CCAM) is a highly variable type of manoeuvre. From payload release to AVUM+ boost, each of its initial conditions and constraints depend on the specific mission. Whether it is the required orbital altitude, the timing of the release, the separation attitude, or the necessary ΔV for deorbiting, the multitude of unique requirements makes it challenging to generalise the manoeuvre and find an analytical method for mission planning. Therefore, the mission design process for this particular phase has always been a very time-consuming trial-and-error activity, heavily reliant on the expertise of the mission designer and general guidelines derived from the experience of previous missions. While this design strategy was effective in the context of a few launches per year, the success of the VV25 mission and Avio's newfound independence from Arianespace in marketing and mission management are leading up to a busier launch schedule and, consequentially, shorter mission design times. Therefore, making the CCAM design more time-efficient and less demanding for the mission designer is an obvious improvement in this context. This work is a first attempt at generalizing the CCAM and building a support tool for CCAM design. The basic idea behind the tool is to develop a MATLAB program that is able to simulate exclusively the CCAM phase of a given mission and evaluate the manoeuvre effectiveness according to the requirements of contamination and collision avoidance. This fundamental core is then expanded to include an optimization algorithm aimed at searching for one or more potentially optimal solutions to the given problem. Despite the accuracy of the tool does not reach the level of the complete GNC simulator used for actual mission design, its potential uses are multiple. From a simple first-guess suggestion for the mission designer to the definition of the manoeuvre for the Preliminary Mission Analysis, this work is highly likely to be an impactful time-saver for Avio engineers. Furthermore, with the conversion of its optimized outputs into mission data to feed to the complete simulator, the possibility of designing the CCAM with a completely automatic procedure becomes ever more real.

Chapter 2

Environment and system definition

This chapter aims to define the environment and systems studied in this work. The hypotheses regarding the orbit, along with the establishment of reference frames, outline the domain of the CCAM. Additionally, the description of Vega-C's fourth stage, AVUM+, introduces the systems involved in the manoeuvre and sets the foundation for the simplifying models discussed in the following chapter.

2.1 Orbit and angular range

Vega-C's main target orbits are Sun-Synchronous Orbits (SSO). Thanks to AVUM+ reignition capabilities, several orbital altitudes can be reached during a single mission, leading to more flexibility in mission design. For each mission phase, an optimized trajectory is computed via Avio's 3-degrees-of-freedom simulator TRASIM. The results are then used as input and reference for mission design on the complete GNC simulator MORPHEUS. Despite each mission having different optimal orbits due to mission-specific mass and consumption budgets, payload requirements and deorbiting safety conditions, it is possible to define some common characteristics that can be useful for the definition of the environment in which the CCAM is going to be simulated. SSO are nearly polar orbits in which the spacecraft passes over any given point of the planet's surface at the same local mean solar time [4]. In particular, the SSO's at which payload separations usually happen in Vega C launches have such small eccentricities that they can be confidently assumed circular. This hypothesis allows the study of the relative dynamics between LV and released payload by means of well-defined and well-known sets of equations and reference frames, while also allowing the locating of the spacecraft in space without the need of integrating its position in an inertial reference frame. In fact, given a circular orbit, it follows that the orbital rate is constant:

$$\omega = \sqrt{\frac{\mu}{r^3}} = cost, \quad if \;\; r = a = cost \tag{2.1}$$

Given the initial position of the LV in the orbit, it is easy to evaluate its trajectory. In order to have information on the spacecraft's position while operating exclusively in a moving relative frame, it is possible to define an angle linked to the orbital rate. In Avio's case, the angular range is the measure used for setting up event triggers during in-orbit mission phases. It is defined as follows:

$$\phi_T = \omega_p + \nu_T \tag{2.2}$$

where ω_p is the argument of perigee of the target orbit and ν_T is the true anomaly. By this definition, the angular range ϕ_T has null value when the spacecraft is at the target orbit's ascending node where $\nu_T = -\omega_p$ and its value grows linearly with the orbital rate ω, making it simple to compute during the manoeuvre simulation.
In addition, no orbital disturbances are accounted for as the CCAM duration is usually short with respect to the orbital period.

2.2 Reference frames

In a typical mission, the Vega C launcher goes through several phases characterized by different altitudes, velocities and forces which need to be analysed and simulated by means of a range of phase-specific mathematical models and GNC algorithms. A wide set of reference frames is defined in order to make these mathematical models comprehensible and coherent with the needs of each phase. The reference frames relative to the CCAM manoeuvre are reported.

Orbital Basic Frame

The Orbital Basic Frame (OB) is a geocentric inertial frame. It is associated with the target orbit for payload separation and it is defined after the 2$^{\text{nd}}$ AVUM+ boost, after reaching the target orbit. It is the main reference frame used for the CCAM manoeuvre by the MORPHEUS simulator. Being an inertial frame, the orientation of the Launch Vehicle at payload release and during the CCAM manoeuvre are the same throughout the year and are not influenced by the launch date when referring to it. In the case of payload release requirements regarding the Sun aspect angle, using such a reference frame makes the separation phase design much simpler, allowing the mission designer to verify exclusively the worst case of Sun aspect angle as the rest is verified by design choices [5].

Orbital Basic Frame ($O_E, \xi_0, \eta_0, \zeta_0$)

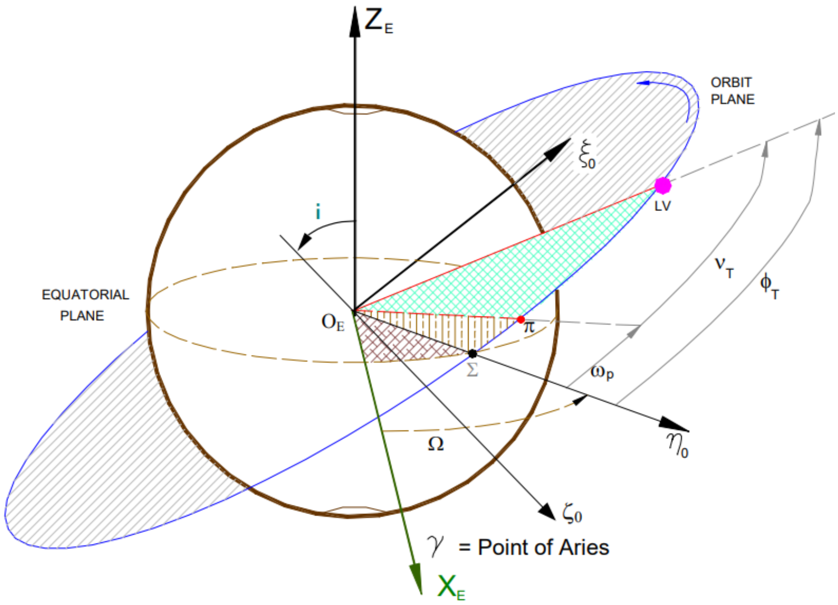


Figure 2.1: Orbital Basic Frame

$\boldsymbol{O_E}$: origin at the centre of the Earth

$\boldsymbol{\xi_0}$: on the orbital plane, completes the right-hand system

$\boldsymbol{\eta_0}$: the intersection between the orbital plane and the equatorial plane, positive toward the ascending node of the orbit

$\boldsymbol{\zeta_0}$: perpendicular to the orbital plane, positive towards the $\vec{v} \times \vec{r}$ vector

Local Orbital Frame (LVLH)

The Local Orbital Frame, also known as the Local Vertical Local Horizontal (LVLH) Frame, is a coordinate frame used to describe motions with respect to a body orbiting around the Earth. It is commonly used for the analysis of the relative motion of a chaser spacecraft with respect to a target one in rendezvous missions. Thinking of the CCAM as the polar opposite of a rendezvous makes the LVLH frame the perfect candidate for the study of the relative dynamics between the AVUM+ stage and the separated payload.

Local Orbital Frame (LVLH) ($\boldsymbol{O_{lo}}, \boldsymbol{V}$-$\boldsymbol{bar}, \boldsymbol{R}$-$\boldsymbol{bar}, \boldsymbol{H}$-$\boldsymbol{bar}$)

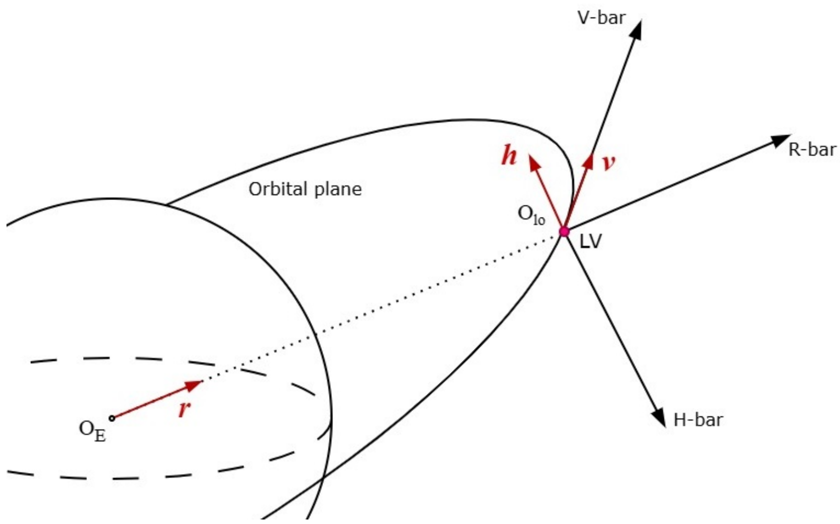


Figure 2.2: Local Orbital Frame (LVLH)

$\boldsymbol{O_{lo}}$: origin at the COG of the spacecraft

$\boldsymbol{V}$-$\boldsymbol{bar}$: on the orbital plane, in the direction of the velocity of the spacecraft

$\boldsymbol{R}$-$\boldsymbol{bar}$: on the orbital plane, in the zenith direction

$\boldsymbol{H}$-$\boldsymbol{bar}$: perpendicular to the orbital frame, in the opposite direction of the orbit's angular momentum vector

As it can be noted, this definition of the LVLH frame is different from the one commonly used in rendezvous missions and described in [6]. In particular, while the V-bar and H-bar axis are defined as in [6], R-bar is oriented in the opposite direction. In order to keep it a right-handed system, R-bar and H-bar are switched in the axes triplet. This definition choice has major implications for the definition of the relative dynamics equations, which will be discussed in the following chapter. Hereafter, $LVLH$ refers to the reference system defined above, while $LVLH_{Fehse}$ refers to the Local Orbital reference frame defined in [6].

Launch Vehicle Geometric Frame

The LV Geometric Frame represents the body axes of the whole launcher. Also called STA-1, it is used as reference for the definition of the position of several points and components involved in the studied manoeuvre.

LV Geometric Frame (O_L, X_L, Y_L, Z_L)

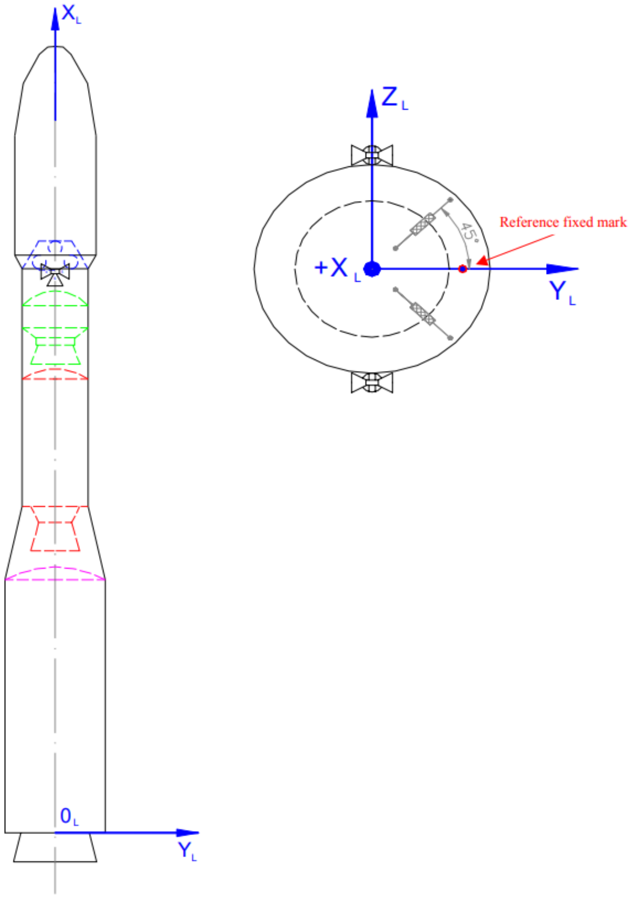


Figure 2.3: LV Geometric Frame

$\boldsymbol{O_L}$: origin at the geometric centre of the bottom flange diameter of Interstage 0/1, at the interface plane between the Launch Vehicle and the Launch Pad

$\boldsymbol{X_L}$: perpendicular to the interface plane between the LV and the Launch Pad, positive towards the nose of the LV

$\boldsymbol{Y_L}$: perpendicular to the X_L axis, at 45° with respect to the actuators, positive towards a fixed reference mark

$\boldsymbol{Z_L}$: completes the right-handed system, in the plane of the reaction control thrusters

The LV Geometric Frame will be referred to as STA-1 from now on.

Body-fixed Frame

The Body-fixed Frame represents the body axes of the fourth stage. It is a fundamental reference frame for the definition of the LV attitude and for the modelization of the payload separation system, the thrusters' configuration and the forces and torques acting on the AVUM+ stage.

AVUM+ Body-fixed Frame

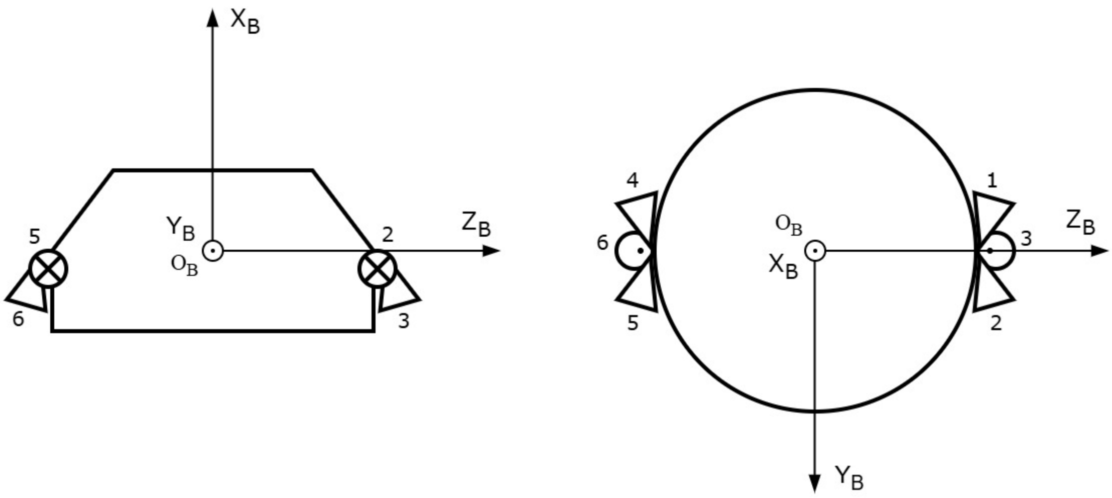


Figure 2.4: AVUM+ Body-fixed Frame

- $\boldsymbol{O_B}$: origin at the COG of AVUM+
- $\boldsymbol{X_B}$: parallel to STA-1's X_L axis, positive towards the nose of the LV
- $\boldsymbol{Y_B}$: same direction and pointing of STA-1's Y_L axis
- $\boldsymbol{Z_B}$: same direction and pointing of STA-1's Z_L axis

The AVUM+ Body-fixed Frame is effectively equivalent to STA-1 shifted in the positive X_L direction if no offset of the COG from the geometric centre exists. From now on, the AVUM+ Body-fixed Frame will be referred to as the Body Frame.

2.3 Attitude Vernier Upper Module

As previously introduced, the Attitude Vernier Upper Module AVUM+ is the upper stage of Vega C. It is part of the fourth assembly of the Vega C launcher, Assy 4, which also includes the Interstage 3/AVUM+. The AVUM+ structure can be divided into 2 parts:

- the AVUM External Structure (AES), which includes the Interstage 3/AVUM+ and all the mechanical supports for the equipment;
- the AVUM Internal Structure (AIS), where most of the Propulsion Module is integrated.

The main systems of the AVUM+ stage:

- the Liquid Propulsion Sub-system (LPS+)
- the Roll & Attitude Control Sub-system (RACS)
- the fourth stage Thrust Vector Control (TVC) Sub-system
- the third stage Safeguard Sub-system
- the Assembly Thermal Protection
- the AVUM Avionic Module (AAM)
- the Telemetry Sub-system
- the Autonomous Localization System [7]

Only the systems and components directly involved in the CCAM manoeuvre are described and analysed further, namely the LPS+ and the RACS. The separation system and the payload adapters are also briefly examined.

Liquid Propulsion Sub-system

The LPS+ is a bipropellant system utilizing unsymmetrical dimethyl hydrazine (UDMH) as a fuel and nitrogen tetroxide as an oxidizer (NTO). The two liquids are pressurized by means of Helium gas, stored in a high-pressure gas vessel. The propulsion is performed by a high thrust bipropellant engine which is fed directly by the propellant tanks. Four tanks for both UDMH (2 items) and NTO (2 items) will be used to store and expel the propellants of LPS+.
The Main Engine assembly is composed of the LPS+, a gimbal system for TVC and structural components for transmitting the engine force to the AVUM structure. It provides 2445 N of nominal thrust with a nominal specific impulse of 315.9 s and it is able to be restarted multiple times, for a maximum of 8 boosts. The TVC provides up to a 9.4° commanded deflection angle [7].

Roll & Attitude Control Sub-system

The Roll & Attitude Control Sub-system is part of the AVUM+ propulsion system and delivers forces and torques to the Attitude and Vernier Upper Module for:

- roll control of the launcher during boost phases
- 3-axis control of the launcher during coasting phases
- longitudinal boost during the first AVUM boost and deorbiting
- payload separation manoeuvres
- CCAM
- Roll control during deorbiting
- Deplation of the remaining propellant to safe levels

The Roll & Attitude Control System is a monopropellant (N2H4) system with two clusters of three thrusters operating in blow down, each having a Begin of Life thrust of 210 N and End of Life thrust of 90 N. The RACS is fed by a single vessel which contains both the propellant (N2H4) and the pressuring gas (GN2).

The RACS thrusters configuration allows 3-axis control of the AVUM+ by being able to generate torques in all body axes [7]. For each cluster, 2 thrusters are canted at 15° with respect to the Y_B axis and their thrust vector is located in the (Y_B, Z_B) plane, while the third thruster is canted at 10° with respect to the X_B axis and lies in the (X_B, Z_B) plane. Each thruster is fixed with no TVC and its position is defined with respect to the STA-1 frame (table 2.1). Table 2.2 shows

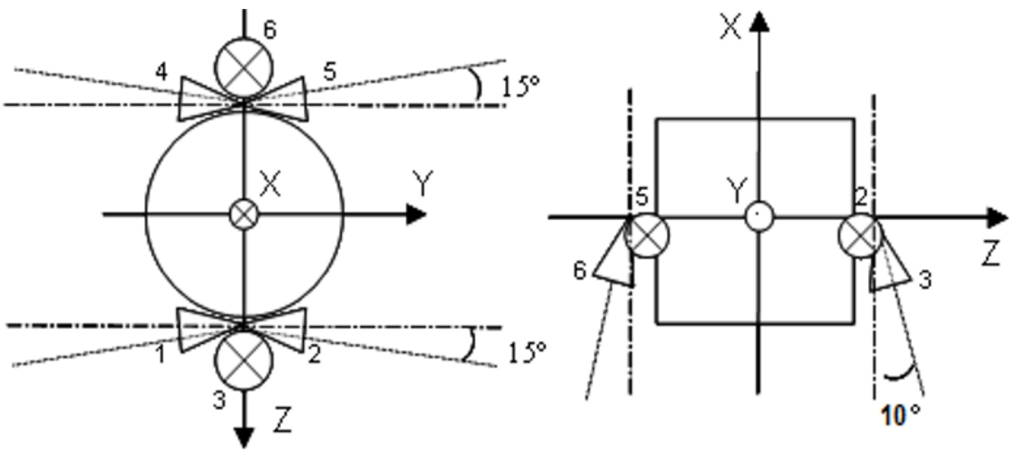


Figure 2.5: RACS thrusters configuration

Table 2.1: RACS thrusters positioning in the STA-1 frame

	X_L [mm]	Y_L [mm]	Z_L [mm]
Thruster 1	25045.5	85.7	1070.6
Thruster 2	25045.5	85.7	1070.6
Thruster 3	25107.9	0	1115.3
Thruster 4	25045.5	85.7	-1070.6
Thruster 5	25045.5	85.7	-1070.6
Thruster 6	25107.9	0	-1115.3

Table 2.2: Torque-Thrusters relationship

	$+T_X$	$-T_X$	$+T_Y$	$-T_Y$	$+T_Z$	$-T_Z$
Thruster 1		×		×		
Thruster 2	×				×	
Thruster 3			×			
Thruster 4						×
Thruster 5		×			×	
Thruster 6	×			×		

the combinations of RACS thrusters to be activated in order to provide torques in the 3 body axes. However, this number of fixed thrusters in such a configuration cannot eliminate force-moment coupling for pitch and yaw slew manoeuvres and is only capable of controlling AVUM roll without causing translational accelerations. For boost manoeuvres, thrusters 3 and 6 are activated, providing pure longitudinal acceleration along the X_B axis, although this is only true in the ideal case of a perfectly centred centre of gravity.

Payload adapters

A critical aspect of the CCAM manoeuvre is the payload separation. It is dependent on the mission which, in turn, depends on the launcher's payload-carrying capabilities. The number and dimensions of satellites as well as the separation directions and velocities depend on the payload adapters and the separation systems. In the case of the Vega C launcher, the separation system is based on springs mounted on different adapters.

Vampire 1194

The Vampire 1194 configuration is used for single payload mission with a mass between 1000 kg and 2500 kg. The number of springs can be selected depending on the needs from a minimum of 4 springs to a maximum of 12. The springs can be adjustable in terms of stroke due to the use of a stroke limiter, in a range of spring energy from 20 J to 61 J. Vampire 1194 bolted flange configuration is used when a secondary payload adapter is installed on it [8].

Vampire 937

The Vampire 937 can be used for single payload missions in its standard configuration and for multi payload missions in a configuration with 6 additional secondary towers. The mass of carried satellites can vary, however in the multi payload configuration, the main payload located on the top has to be the heaviest. The number of springs is fixed at 4 and it can be tuned at two different values of energy [8].

VESPA

The VESPA adapter allows to carry a passenger on the top (mass up to 1000 kg) and several passengers inside its cavity for a maximum total mass of 600 kg [8].

SSMS

The Small Satellite Mission Service (SSMS) is a modular dispenser that allows to allocate: CubeSats (from 1 kg), NanoSats (100 kg) and MiniSats (up to 400 kg). Different configurations are possible in order to optimize the accommodation of satellites and the separation systems depend on the type of satellites carried.

Chapter 3

Contamination and Collision Avoidance Manoeuvre definition

This chapter lays the foundation for the tool's functionality by introducing the key parameters and models it operates on. The parameters are detailed through an in-depth definition of the CCAM manoeuvre. Based on the described manoeuvre environment and the involved systems, this chapter also develops the models adopted for spacecraft dynamics, control systems, and payload contamination.

3.1 CCAM structure

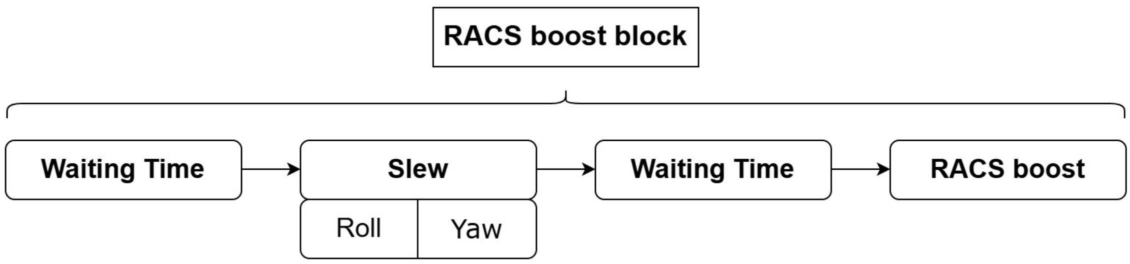
Due to the numerous variable parameters and requirements upon which the CCAM manoeuvre depends, defining a general CCAM structure that applies to all missions is no simple task. However, it is possible to identify common patterns and similarities in missions without unique constraints and requirements. From these observations, manoeuvre-specific blocks and phases can be derived, which are present in most previously designed missions.

Every CCAM begins with the separation of one or more satellites. While critical for achieving proper distancing and relative dynamics, the payload separation phase is typically constrained by mission-specific requirements. Consequently, AVUM's angular range and attitude at release, as well as the payload's release direction and velocity relative to the body frame, are predetermined and serve as initial conditions for designing the CCAM. Therefore, the initial time t_0 in the manoeuvre definition corresponds to the moment the first payload to be released receives the separation springs' impulse. Each CCAM is directly followed by a Main Engine boost for de-orbiting or orbit variations. Although often considered a separate mission phase, this final boost represents a major source of contamination for the released payload

and must therefore be simulated to ensure an accurate evaluation of the manoeuvre. In the scope of this work, the Main Engine boost is defined as the final phase of the CCAM.
What remains to be detailed is the sequence of events between the payload release and the AVUM boost, that is, the actual manoeuvre itself. Analysing multiple missions and their CCAMs has revealed distinct phases that can be grouped into four primary blocks. These blocks are common to most missions, although their number, sequence, and the duration of their phases may vary significantly. Nonetheless, each block generally consists of slew phases, boost phases, and waiting periods that separate the phases and blocks.

RACS thrusters boost

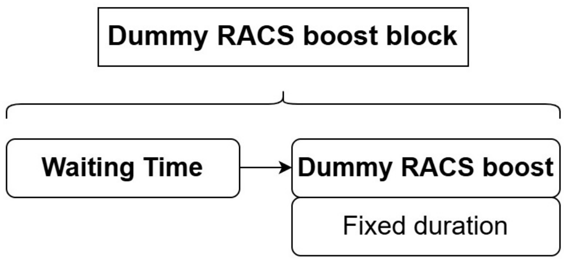
The RACS thrusters boost, or more simply RACS boost, is the main block of the CCAM. It fundamentally consists of a slew phase followed by a RACS thrusters boost and it is by means of this block that the AVUM+ stage is oriented in the desired attitude and distanced from the separated payload. While a single RACS boost is enough for some missions, usually, several of these blocks are designed to move the AVUM+ stage away from the released payload and reach the required attitude for the Main Engine ignition.



The slew phase is divided in two sub-phases. According to the flight phase and to mission requirements, Avio engineers use different types of slew manoeuvres in the mission design process. In the CCAM scenario, the most commonly used type of slew manoeuvre is the Yaw-Short slew, consisting of a first roll phase followed by a yaw rotation to reach the desired attitude. This type of slew is used in this part of the mission because it is the one that causes less RACS pollution, as only one thruster is active during the yaw manoeuvre (see table 2.2). The Yaw-Short slew differs from the Yaw-Long type by the angle covered to reach the final attitude, which is the smaller one. Other possible types of slews used in Vega-C mission design are Pitch-Short/Long where the yaw slew is replaced by a pitch slew and Euler-Short/Long which is a one-phase slew around the Euler axis allowing to reach

the desired attitude in a single manoeuvre. However, since the Yaw-Short slew is by far the most convenient for contamination avoidance and it is the one used in the great majority of missions, it is chosen as the default slew type for the definition of the RACS boost block. Furthermore, the two-phase slews (Yaw-Short/Long and Pitch-Short/Long) can also have an additional third roll phase, for instance in the case of the slew preceding the payload separation. Nevertheless, the instant the first payload is released is defined as the initial time of the CCAM, therefore this particular case is of no interest in the context of this work as it happens before that starting moment.
The Waiting Time phase before the slew is of particular importance, as the RACS boost block always follows the payload release to meet the distancing requirements. Referring specifically to this initial RACS boost block, the first Waiting Time, called the Post-Release Waiting Time, is longer than a usual one. In fact, assuming infinite time, the simplest and most efficient CCAM is a basic free drift with no slews or boosts, relying solely on the impulse from the separation system springs to distance the satellites. While infinite time is not feasible, setting a longer initial Waiting Time proves to be an effective strategy for achieving a successful CCAM

Dummy RACS thrusters boost

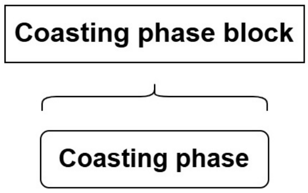


This block is a specific case of RACS thrusters boost without a slew phase, consisting of a single waiting time and a fixed-duration RACS boost.

It is usually placed before the Main Engine boost for safety reasons, serving as a final check of payload separation. While AVUM+ attitude control is activated during RACS boosts to compensate for the COG offset effect, it is deactivated for dummy boosts, as these typically occur far from the separated payload and have no actual manoeuvring purposes. Moreover, this block is always included in CCAM manoeuvrers, which is why it is defined separately from standard RACS boosts.

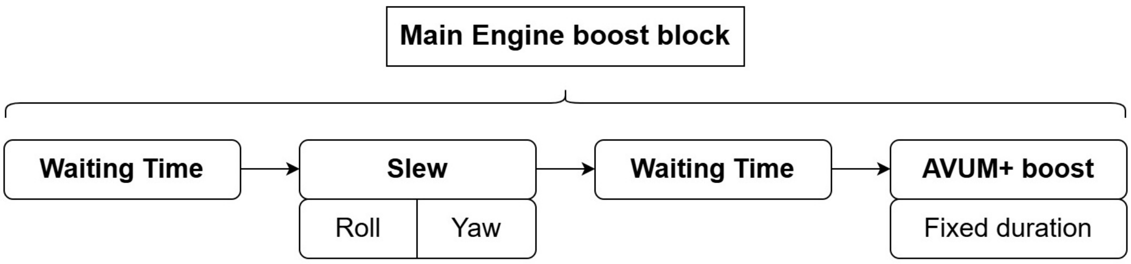
Coasting Phase

The Coasting Phase block is used when the first distancing by RACS boost has already happened, but the spacecraft is still far away from the required angular range for deorbiting or orbit variation.



It consists of a unique phase of coasting with no slews preceding it and no boosts. Attitude control is active in this phase.

Main Engine boost



The Main Engine boost is the final block of the CCAM manoeuvre as defined in this work. It has the same structure as the RACS boost block with the RACS boost replaced by the Main Engine boost.
Since the contamination caused by the AVUM+ Main Engine is much greater than the RACS thrusters', this boost can only start when the payload is distant enough and the spacecraft is oriented in a direction that would not cause the exhaust plume gases to hit the separated satellite. Moreover, the AVUM+ boost is used for deorbiting or orbit variation so its ΔV and duration are fixed values derived from the study of the optimal orbit and, therefore, constraints for the CCAM manoeuvre.

Reference missions

Among the missions analysed for the definition of the CCAM manoeuvre blocks, two were chosen as reference for the development of the tool and the generalisation of the problem.

Sentinel Z40RP

Sentinel Z40RP is a non-final version of the VV25 mission that launched the Sentinel-1C Earth Observation Satellite for the European Commission Copernicus Program. In this version, the payload is released 1 hour and 44 minutes after lift-off in an SSO orbit at 700 km of altitude. This mission was chosen as a reference because it represents the most simple and general case of CCAM.

RACS boost	**Dummy RACS boost**	**Main Engine boost**

VC04 Biomass

VC04 Biomass is planned to launch in April 2025 to inject the ESA Earth Observation Satellite Biomass in a 666 km SSO. While this mission is also carrying a single payload, its CCAM is much longer than the Sentinel one as it features an additional RACS boost block and a coasting phase before deorbiting.

RACS boost 1	**RACS boost 2**	**Coasting Phase**	...

...	**Dummy RACS boost**	**Main Engine boost**

3.2 Hypotheses and Models

In this section, the hypotheses and the models adopted in the tool are presented and discussed. The simplification of the problem is necessary to allow the tool to compute a sufficient number of manoeuvres to allow the research of optimized solutions in an acceptable amount of time.

Orbit and angular range

As previously discussed, the CCAM takes place in SSO circular orbits, where the angular range increases linearly over time due to the constant orbital rate. An additional assumption must be made regarding this aspect. One might consider that CCAM boosts by the RACS thrusters or the Main Engine could alter the orbital altitude and eccentricity, resulting in changes to the orbital rate and thus disrupting the linearity of the angular range. This would compromise the usefulness of this measure, requiring the integration of the Launch Vehicle's orbital position to determine its location in space. However, this is not the case for the CCAM

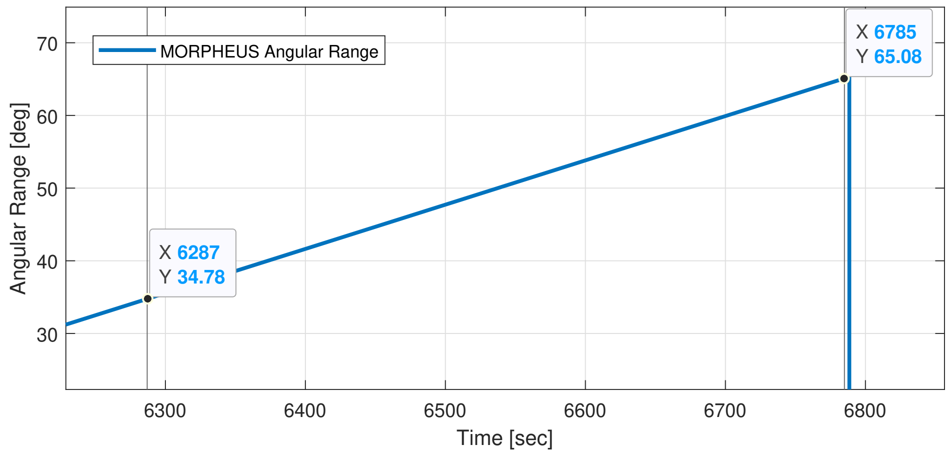


Figure 3.1: Sentinel Z40RP angular range vs time (Morpheus output)

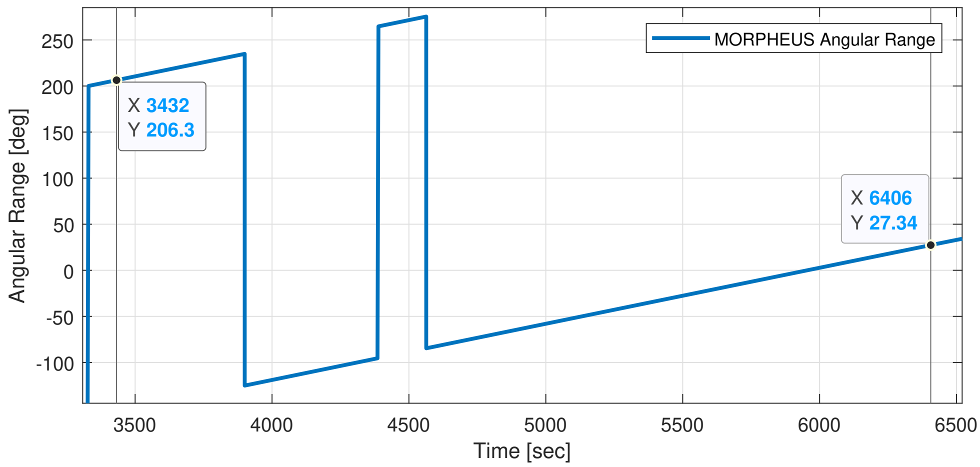


Figure 3.2: VC04 Biomass angular range vs time (Morpheus output)

manoeuvrer. It can be assumed that RACS thruster boosts have a negligible effect on the orbital rate and, consequently, on the angular range, whereas Main Engine boosts, with their much greater accelerations over longer durations, do have an impact. Nevertheless, this limitation does not affect the practicality of the angular range measurement, as the Main Engine boost phase is fixed by the optimal orbit, and by that stage, the CCAM is essentially complete. In figures 3.1 and 3.2 the plots of the angular range variation during the CCAM of the reference missions obtained from the output of the Morpheus simulator prove this hypothesis valid. Notice that the dips in angular range in figure 3.2 are due to the way different angle conventions are used during the various manoeuvre phases and to the simulator trying to avoid the measure to saturate to 360° by periodically subtracting 360° from the value of angular range at specific moment.

Boosts

The RACS thrusters operate in blow down with a Begin of Life thrust of 210 N and End of Life thrust of 90 N. The RACS boosts of the CCAM are not, however, the only sources of fuel consumption and thrust lowering.

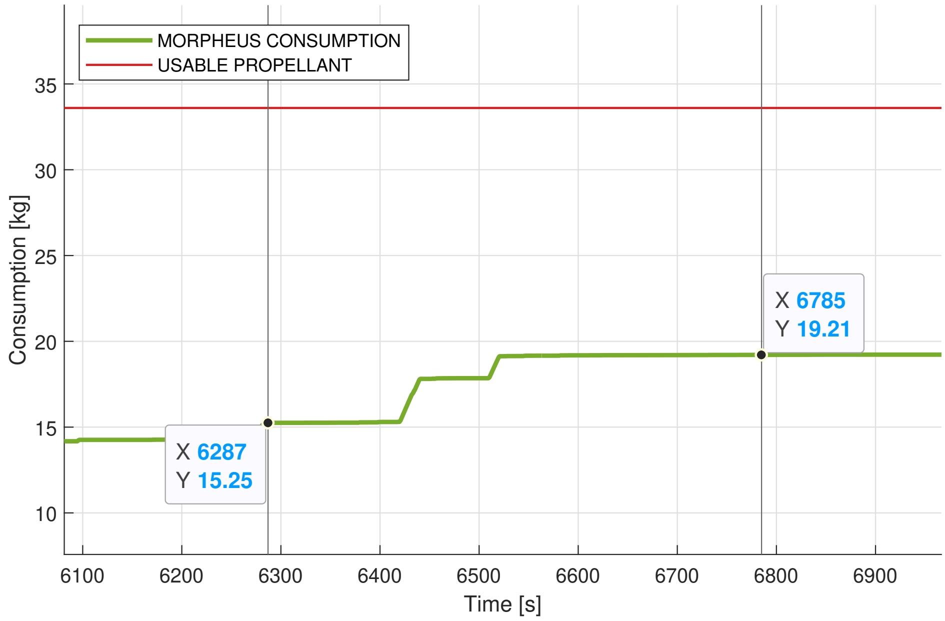


Figure 3.3: Sentinel Z40RP RACS consumption (Morpheus output)

As visible from figures 3.3 and 3.4, the whole CCAM of the reference missions accounts for about 20% of the total RACS consumption in Sentinel Z40RP and 46% for VC04 Biomass, with both the percentages almost halved when referring to the total usable propellant, about 11% for Sentinel Z40RP and 25% for VC04 Biomass. Therefore, assuming the thrust of the RACS thrusters constant and referring to the value at the beginning of the CCAM, can be an acceptable hypothesis if taking into consideration that higher simulation accuracy for the first phases is most valuable in the evaluation of contamination and distancing. A similar argument can be made for the Main Engine thrust for which the hypothesis of constant thrust is closer to reality since the AVUM+ Main Engine does not operate in blow down. Besides, errors in the simulation of the final boost have fewer repercussions on the evaluation of the manoeuvre as the payload has already reached acceptable distances for which collision is less likely to happen and contamination depends more on the nozzle orientation with respect to the satellites than on the level of thrust. In addition, the thrust vector of the Main Engine is supposed to be aligned with the X_B axis and passing through the geometric centre of the AVUM+ stage at all times. This

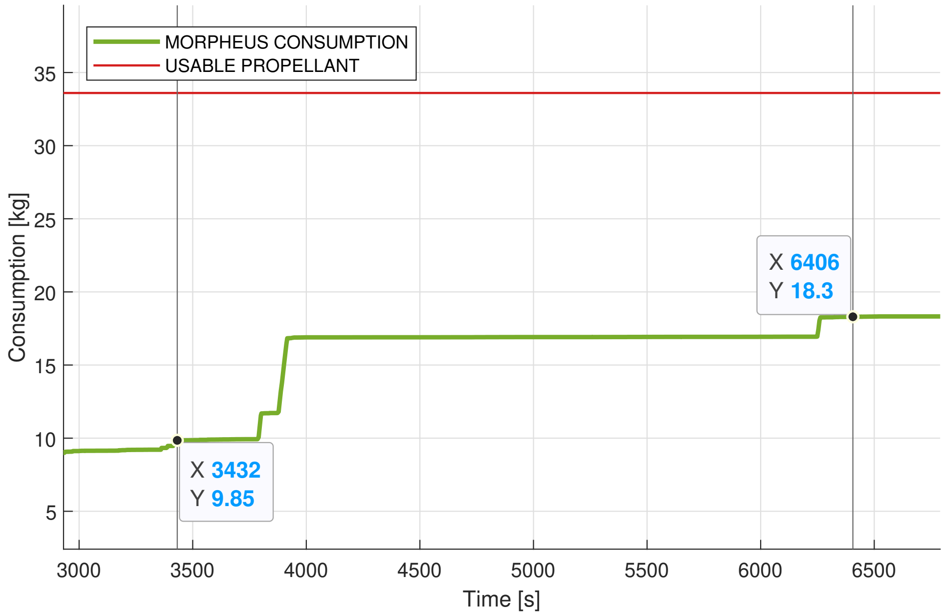


Figure 3.4: VC04 Biomass RACS consumption (Morpheus output)

assumption is justifiable by the TVC that, among its funtions, compensates for the offset of the centre of gravity.
If the hypothesis of constant thrust is combined with the additional hypothesis of constant mass throughout the whole CCAM, it implies that the provided acceleration to the AVUM+ stage is always constant too. This assumption is coherent with the hypothesis of constant RACS thrust, as fuel consumption is neglected, and it is acceptable in the context of the Main Engine boost as the same argument on the accuracy of this phase used to justify constant thrust applies here too.

Slews

The RACS thrusters' configuration allows a complete 3-axis control of the attitude of the AVUM+ stage. In order to carry out the slew manoeuvres, the RACS thrusters are activated to provide angular velocities by quasi-impulsive boosts, and once the programmed angle is covered, an opposite angular impulse cancels the angular velocity. This control logic minimizes RACS activation and is beneficial for fuel consumption and payload contamination. In the scope of this work, this strategy reveals an additional benefit, as it allows for the simple modelization of slew manoeuvres. In fact the angular velocities involved in the slews resemble a step function during their manoeuvring time, as noticeable in figure 3.5 where a first roll slew is followed by a yaw slew. It is possible to find minor peaks and dips in the

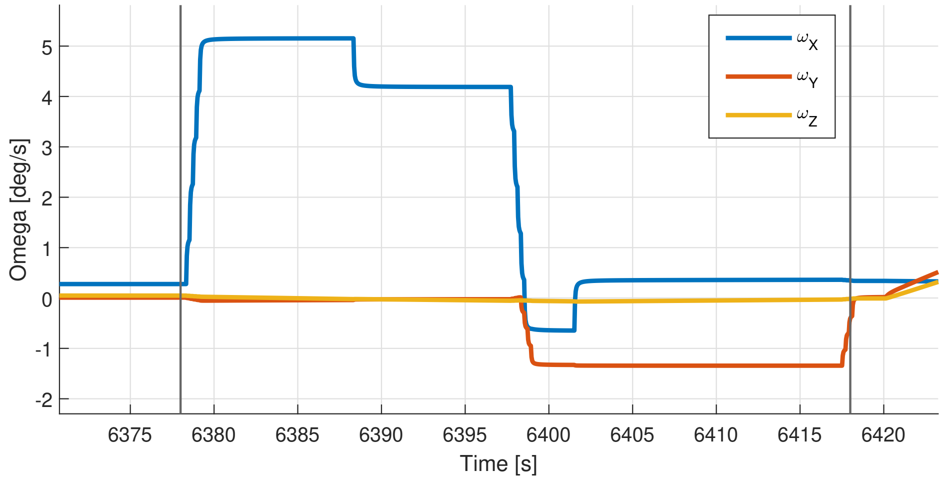


Figure 3.5: Sentinel Z40RP RACS boost slew phase

step-like plots of the angular velocities obtained from the Morpheus output due to the simulated automatic control, which can also lead to angular velocities approaching a null value without actually reaching it. Using a pure step-like model for slew manoeuvres can be a good trade-off between simplicity and simulation accuracy. The acceleration phases can be thought of as an additional source of error for such a model, however, their impact is so minor that in the Morpehus simulator itself their duration is fixed independently of the manoeuvring angular velocity to reach. Furthermore, the error they cause is mostly cancelled by the absence of angular velocity dips and peaks in the model adopted for the tool.

Contamination model

The definition of a reliable contamination model is a critical aspect for the utility and reliability of the developed tool. Where possible and computationally efficient, the same model used in the Morpheus simulator is adopted as in the case of Main Engine contamination. As regards the RACS thrusters' contamination, it is necessary to define a conservative method of evaluation partially derived from the main simulator.

Main Engine

The gas at the nozzle exit of the Main Engine is mainly composed of H_2, H_2O, N_2, CO and CO_2. In order to estimate the contamination by the exhaust plume, the following relations are used to calculate the LPS plume mass flow $[mg/m^2/s]$:

$$
\begin{aligned}
T &= \frac{T_0}{1-0.5(\gamma-1)M^2}, && \text{Plume Temperature}\\
\rho &= \frac{\rho_0}{[1-0.5(\gamma-1)M^2]^{\frac{\gamma}{\gamma-1}}}, && \text{Plume Density}\\
p &= \frac{p_0}{[1-0.5(\gamma-1)M^2]^{\frac{\gamma}{\gamma-1}}}, && \text{Plume Pressure}\\
V &= M\sqrt{\gamma R_{gas} T}, && \text{Gas Speed}\\
q &= p + \frac{\rho V^2}{2}, && \text{Dynamic Pressure}\\
\dot{m} &= \sqrt{2q\rho}, && \text{Mass Flow}
\end{aligned}
$$

where T_0 is the chamber temperature, ρ_0 is the chamber density, p_0 is the chamber pressure, R_{gas} is the gas constant, γ is the plume specific heat ratio and M is the Mach number. M and V are obtained from near flow field CFD analyses as functions of polar coordinates $f(R, \phi)$ where R is the distance from the nozzle exit and ϕ is the angle with respect to the nozzle axis. Mach and velocity values are computed

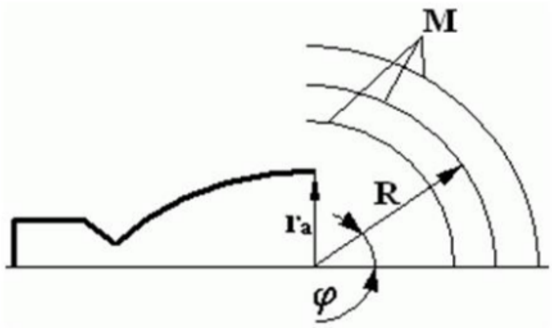


Figure 3.6: ME contamination polar coordinates

up to $R = 300m$ and $\phi = 80°$ and interpolated with a 4^{th} order polynomial in order to obtain:

$$M = f(R)|_{\phi=0} \quad M = f(R)|_{\phi=\phi_1} \quad M = f(R)|_{\phi=\phi_2} \quad M = f(R)|_{\phi=\phi_3}$$

For data up to distances greater than $300m$ it is sufficient to extend the the interpolating polynomials. For $R = \overline{R}$, $M(\overline{R}, \phi)$ can be evaluated by the 4^{th} power polynomial fitting the values of M obtained from the previous relations.
By following this procedure of interpolation and extrapolation, one gets a set of mass flow values as functions of distance and angle with respect to the nozzle exit, figure 3.7. In particular, for distances greater than $10000m$, the pollution level is so low that it is safe to consider it null regardless of the ϕ angle. For this reason

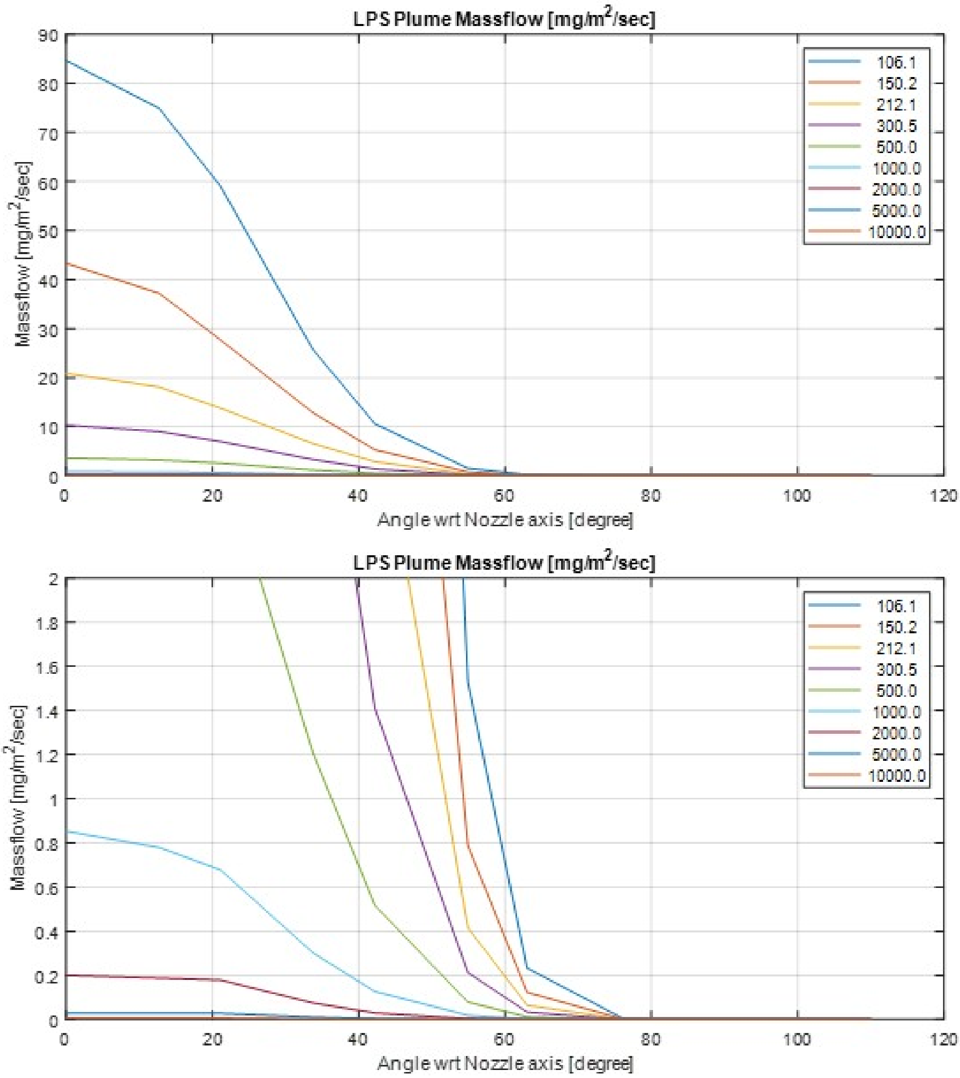


Figure 3.7: Extrapolated LPS Plume Massflow

the Main Engine contamination model of Morpheus evaluates mass flow values for distances up to 10000m and angles up to 110°. This values are then integrated with respect to time to compute a conservative value of expected contaminant deposit $[mg/m^2]$.

RACS thrusters

In the case of the RACS thrusters' exhaust, the gaseous phase consists of H_2, N_2, and NH_3. A similar modelling approach to that of the Main Engine plume can be applied; however, due to the lower thrust capabilities compared to the LPS, the RACS exhaust plume is less dispersed and reaches shorter distances. By extrapolating the near flow field computed through CFD analyses, as done for the Main Engine, a set of correlation relations is derived to compute the mass flow density

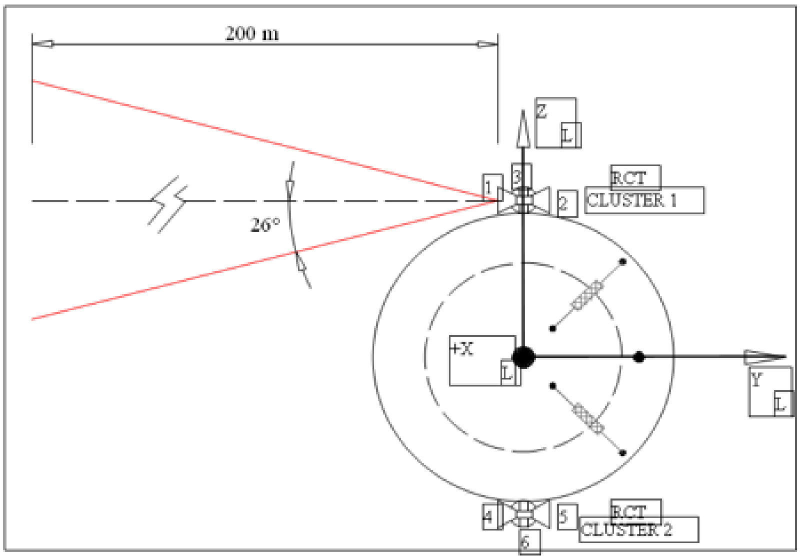


Figure 3.8: RACS contamination cone

as a function of distance and angle. This model is implemented in the Morpheus simulator but remains unusable within the scope of this work. Unlike the complete simulator, the simplified tool developed here cannot simulate the closed-loop active attitude control by RACS thrusters. Assuming that RACS thrusters are only activated during longitudinal boosts and at the beginning and end of slews may lead to a potential underestimation of the actual contamination, as control activations are not modelled. On the other hand, assuming that RACS thrusters remain constantly active during slew manoeuvres, or even throughout the initial CCAM phases, would result in an overestimation of the pollution level by several orders of magnitude. As with the LPS plume, a maximum distance can be established, beyond which it is safe to assume no contamination occurs regardless of the aspect angle between the payload and the thruster. Given the less dispersed nature of the RACS thrusters' plume, a maximum contamination angle is also defined, leading to the formation of a RACS contamination cone originating from each thruster's nozzle exit.
The contamination cones extend up to $200m$ in length and have half angles of 26°, covering a significant volume around the AVUM+ stage (figures 3.8 and 3.9). This danger zone model enables the definition of a conservative yet reasonable contamination model that aims to avoid the passage of the separated payload through these areas or, at the very least, minimize the time spent inside any cone. This approach is consistent with the practices followed by Avio engineers during CCAM design, as RACS contamination at short relative distances is typically unacceptable, and control boosts are usually hard to predict.

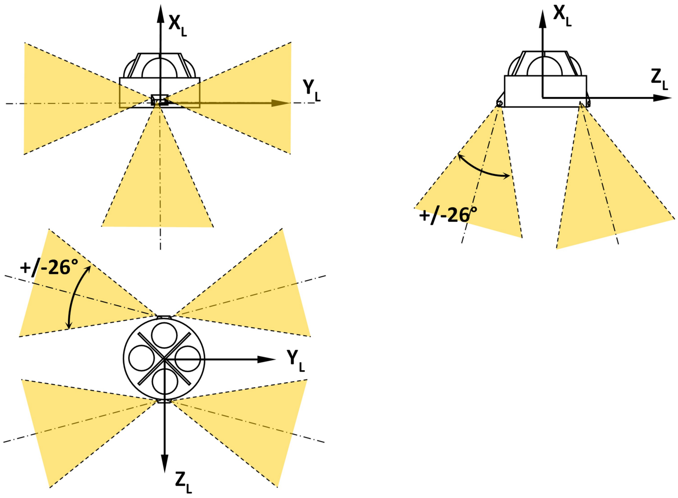


Figure 3.9: RACS contamination cones coverage

Force-Torque coupling

The assumption of constant mass and a centred, fixed centre of gravity for the AVUM+ stage can be considered acceptable in the context of Preliminary Mission Analysis and the initial estimation of the CCAM manoeuvre. This hypothesis simplifies the simulation as it eliminates Force-Torque coupling during longitudinal RACS boosts and roll slews. However, this assumption does not hold for yaw and pitch slews, as can be inferred from the RACS thrusters' configuration. To account for all cases, including scenarios with an offset COG position, a Torque-Force coupling model is introduced. This model allows the estimation of both the translational acceleration generated during slew manoeuvres and the angular acceleration produced during longitudinal RACS boosts. For the Main Engine, the thrust vector is assumed to always pass through the Launch Vehicle's COG, even with COG offsets, due to the TVC hypothesis.

Slew manoeuvre

During yaw slew manoeuvres, only one RACS thruster is active at a time, specifically either Thruster 3 or Thruster 6, the longitudinal ones. Without an equal and opposite force acting on the spacecraft, the translational effect of the thrust cannot be cancelled, as shown in figure 3.10. The same is true for pitch manoeuvres, where two thrusters are active simultaneously, namely T-1 and T-4 or T-2 and T-5. In this case, the likelihood of contamination increases significantly, not only because the number of active thrusters is doubled but also due to the much shorter effective lever arm vector compared to the yaw slew. This results in considerably

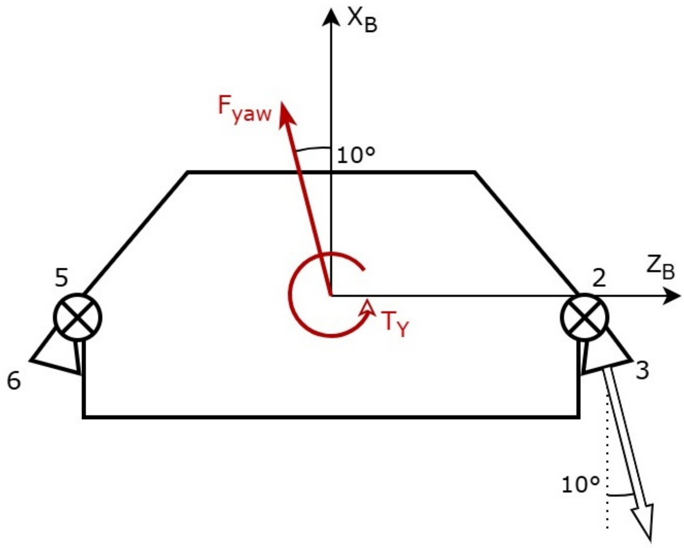


Figure 3.10: Force-Torque coupling in positive yaw slew

longer firing times to achieve the required angular velocity. Since this work focuses exclusively on yaw slews, the pitch case is neither represented nor analysed further. Regarding roll control, no significant translational acceleration component is generated, as opposite tangential thrusters are activated to produce torque. To model this coupling effect during yaw slews, an additional acceleration vector is applied to the Launch Vehicle's COG based on the canting of the active thruster. The activation time corresponds to the acceleration time specified for slew manoeuvres in the Morpheus simulator, equal to 12 Major Cycles. This measure is tied to the clock frequency of the onboard computer interface with the Launch Vehicle, where 1 $mjC = 0.004997253 \times 8s \approx 0.04s$, making 12 $mjC \approx 0.5s$.

RACS boost

Both longitudinal RACS thrusters, T-3 and T-6, are activated simultaneously to provide the required ΔV during the RACS boost phase. In an ideal scenario with a perfectly centred COG, no torque components are generated by this manoeuvre, as both thrusters are aligned with the Z_B axis. In reality, however, a small offset of the COG with respect to the geometric centre is always present in all flight phases. Even if minor, this offset can noticeably affect simulation accuracy, particularly during the initial and most critical distancing manoeuvres.

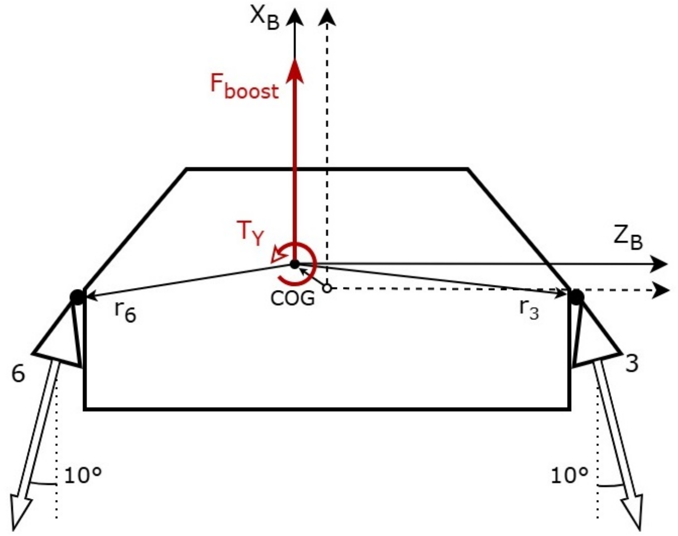


Figure 3.11: Force-Torque coupling in RACS boost, (X_B, Z_B) plane

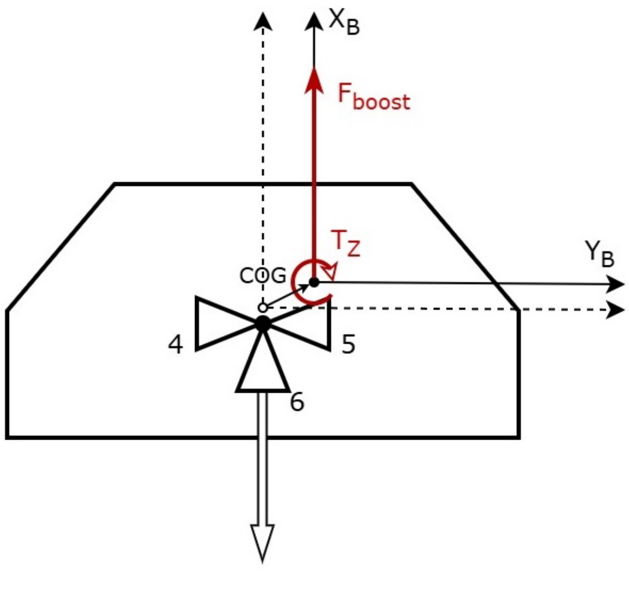


Figure 3.12: Force-Torque coupling in RACS boost, (X_B, Y_B) plane

Figures 3.11 and 3.12 show that two different torque components are generated by the COG offset. In the (X_B, Z_B) plane the difference of the arm lever vectors between T-3 and T-6 generates a net yaw torque, while in the (X_B, Y_B) plane the COG offset creates a misalignment with the thrust vector generating a pitch torque. The pitch component is also the most influential as both thrusters contribute to its generation. This phenomenon can be an issue for the mission designer trying to predict

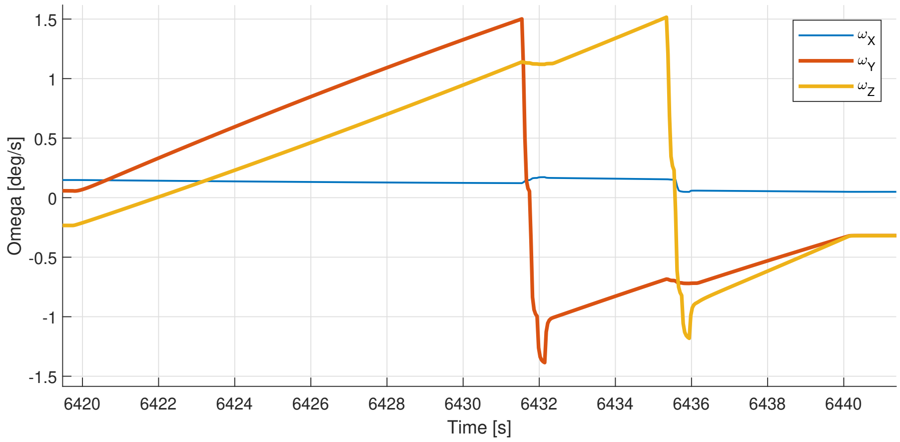


Figure 3.13: RACS boost coupling control effect on angular velocity

the AVUM+ stage trajectory during the RACS boost phase since the attitude of the spacecraft is continuously changed by the boost itself. An active control strategy is defined in the GNC software to reduce this negative effect. When either a component of angular velocity or the attitude error exceeds a certain threshold during the RACS boost and the control gain results in a command that is higher than the minimum thrust provided by RACS, the control is activated, effectively inverting the value of angular velocity to regain the required attitude and compensate for the coupled torque effect of the rest of the boost. As visible in figure 3.13, the control does not activate immediately after the angular velocity threshold (circa $1.5rad/s$) is surpassed as the control gain depends on both the angular velocity and the attitude error, leading to activations of the RACS boost control at slightly different angular velocities. Since the tool developed in this work relies on a complete open-loop logic, this control strategy is simplified based on the observations of the angular velocity responses. During RACS boosts, the torque due to the COG offset is computed and integrated to obtain the angular velocity generated, and once any of its components reaches a predefined threshold value, the value of all the angular velocity components is instantly inverted, mimicking the active control to a satisfactory level. The limiting factor of this model is linked to the hypothesis of a fixed COG, that, in the real scenario, changes its position in time, changing in turn the effect of the force-torque coupling on the AVUM+ stage. The choice of an appropriate value of COG offset is fundamental for the reliability of the simulation, especially for the first phases of the CCAM manoeuvre. Hence, the choice of defining a centred COG for Preliminary Mission Design and first-guess estimations, while more accurate simulations are planned for the later stages of mission design, once estimations of the COG offset are available.

3.3 Parameters

After the definition of the CCAM manoeuvre structure and the adopted models, it is necessary to define the set of parameters the developed tool uses for simulation and optimization before proceeding with the in-depth description of the program. The CCAM is a manoeuvre with an outstanding number of degrees of freedom, ranging from the attitudes at different phases to the ΔV provided by the thrusters. The waiting times between each of the phases are also variable and influential on the outcome of the CCAM, which, as already discussed, is highly variable in its structure too. This characteristic of the manoeuvre is the leading reason for the block definition of its structure and of its parameters as well. A different set of parameters is defined for each block of the manoeuvre, allowing for a dynamic definition of the parameter matrix according to the structure of the CCAM analysed. In general, the parameters can be of two types: durations and angular velocities. The first group, which is also the majority, describes the duration of each phase, including the boost phases. Because of the hypothesis of constant thrust of both RACS thrusters and Main Engine, the boost durations actually represent the ΔV provided to the AVUM+ stage. However, due to simplicity and coherence with the rest of the parameters, it was chosen to keep these parameters as time measures. The second group of parameters describes the slew angular velocities which, together with the slew durations, define the angles to cover in order to achieve the intermediate and final manoeuvre attitudes.

RACS boost

ID	Name	Unit
1	Waiting time	[s]
2	Roll slew dt	[s]
3	Yaw slew dt	[s]
4	Waiting time	[s]
5	RACS boost dt	[s]
6	Roll ω	[rad/s]
7	Yaw ω	[rad/s]

AVUM boost

ID	Name	Unit
1	Waiting time	[s]
2	Roll slew dt	[s]
3	Yaw slew dt	[s]
4	Waiting time	[s]
5	Main Engine boost dt	[s]
6	Roll ω	[rad/s]
7	Yaw ω	[rad/s]

Dummy RACS boost

ID	Name	Unit
1	Waiting time	[s]
2	Dummy boost dt	[s]

Coasting Phase

ID	Name	Unit
1	Coasting time	[s]

According to the above-defined block scheme for parametrization of the CCAM manoeuvre, the reference missions feature the following sets of parameters

Sentinel Z40RP

The Sentinel Z40RP CCAM consists of three blocks: RACS boost, dummy RACS boost, and Main Engine boost. As a result, it is characterized by a total of $7+2+7 = 16$ parameters that define it starting from its initial conditions, of which 12 are phase durations and 4 are slew angular velocities.

VC04 Biomass

The CCAM of VC04 Biomass includes an additional RACS boost and coasting phase compared to the Sentinel Z40RP mission. Therefore, it is described by $7 + 7 + 1 + 2 + 7 = 24$ parameters, of which 18 are phase durations and 6 are angular velocities.

Chapter 4

Tool

The tool developed in this work is based on an open-loop simulator of relative orbital dynamics which integrates both the LV-payload relative position and the attitude of the AVUM+ stage. After requesting a set of information on initial conditions as user inputs, it performs Monte Carlo-like campaigns to generate a user-defined number of random CCAMs. This first set of manoeuvres is evaluated by a cost function based on the contamination and collision avoidance requirements and each solution is ranked accordingly. While this process could be enough for a first analysis of the domain of potentially optimal solutions, the following optimization procedure refines the best-generated CCAMs, leading towards one or more local optima. The tool is suited for multi-payload simulations too, allowing the definition of contemporaneous and sequential releases with different separation conditions. The tool features extensive post-processing capabilities, enabling the user to plot any manoeuvre from the randomly generated pool and the optimized set, compare any number of manoeuvres on the same plots and study the best solutions. The plotting section is well suited for multi-payload missions too, giving the user the choice to analyse all payloads simultaneously or any arbitrary number of them. To speed up the initialization process of already simulated missions, a dedicated section of data saving lets the user store the input data of the current mission to use in future runs. Additionally, the output of the optimizer, as well as the randomly generated results, can be saved for future plotting and analysis.

4.1 Conversions

Before beginning the description of the tool and its functionalities, it is necessary to illustrate the conversions used in this work, as the functions that define them are used throughout the whole program. Both reference frame conversions and conversions between different attitude representations are included in this section.

Orbital Basic to Local Orbital LVLH

The peculiar definition of the LVLV frame adopted in this work appears to be quite convenient for its conversion with the Orbital Basic frame. Indeed, given any value of angular range ϕ_T, it is sufficient to rotate the Orbital Basic frame of this angle around its ζ axis clockwise (effectively rotating of $-\phi_T$) to reach the LVLH frame. It can be noted that these frames overlap once per orbital period when $\phi_T = 0$.

$$R_{OB \rightarrow LVLH} = \begin{bmatrix} cos(\phi_T) & -sin(\phi_T) & 0 \\ sin(\phi_T) & cos(\phi_T) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Euler Angles to Rotation Matrix

The Euler angles convention used by Avio for the definition of the launch vehicle attitude in the Mission Data is the ZYX convention, namely the Pitch-Yaw-Roll rotation order. Given the Euler angles (ϕ, θ, ψ) ordered according to the adopted convention, with ϕ describing the Pitch rotation and θ and ψ the Yaw and Roll ones respectively, it is easy to define their rotation matrices $(Z_\phi, Y_\theta, X_\psi)$ and multiply them to obtain the conversion matrix:

$$R(\phi, \theta, \psi) = \begin{bmatrix} c(\phi)c(\theta) & c(\phi)s(\theta)s(\psi) - c(\psi)s(\phi) & s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta) \\ c(\theta)s(\phi) & c(\phi)c(\psi) + s(\phi)s(\theta)s(\psi) & c(\psi)s(\phi)s(\theta) - c(\phi)s(\psi) \\ -s(\theta) & c(\theta)s(\psi) & c(\theta)c(\psi) \end{bmatrix} \tag{4.2}$$

where $s(\cdot) = sin(\cdot)$ and $c(\cdot) = cos(\cdot)$ for notation clarity.

Rotation Matrix to Quaternion

The method featured in the Matlab Navigation Toolbox and Robotic Systems Toolbox function rotm2quat is chosen to define the conversion between the Rotation Matrix and Quaternion representation. Given an orthogonal matrix $D \in \mathbb{R}^{3\times3}$, $K \in \mathbb{R}^{4\times4}$ is formed:

$$K = \frac{1}{3} \begin{bmatrix} d_{11} - d_{22} - d_{33} & d_{21} + d_{12} & d_{31} + d_{13} & d_{23} - d_{32} \\ d_{21} + d_{12} & d_{22} - d_{11} - d_{33} & d_{32} + d_{23} & d_{31} - d_{13} \\ d_{31} + d_{13} & d_{32} + d_{23} & d_{33} - d_{11} - d_{22} & d_{12} - d_{21} \\ d_{23} - d_{32} & d_{31} - d_{13} & d_{12} - d_{21} & d_{11} + d_{22} + d_{33} \end{bmatrix} \tag{4.3}$$

The corresponding eigenvalues and eigenvectors are computed. The eigenvector corresponding to the largest eigenvalues is the unit quaternion representing the rotation [9]. The scalar part is kept positive by convention, as $q = -q$ represents the same rotation. The resulting quaternion is in the form $q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$ however, the convention used in this work has the vector part preceding the scalar part:

$$q = a\mathbf{i} + b\mathbf{j} + c\mathbf{k} + d \tag{4.4}$$

therefore a final rearrangement of the quaternion components is needed to complete the conversion.

Quaternion to Rotation Matrix

Given a unit quaternion, it is possible to derive the orthogonal matrix corresponding to its rotation when post-multiplying with a column vector [10]:

$$R(q) = \begin{bmatrix} a^2 - b^2 - c^2 + d^2 & 2ab - 2cd & 2ac + 2bd \\ 2ab + 2cd & -a^2 + b^2 - c^2 + d^2 & 2bc - 2ad \\ 2ac - 2bd & 2bc + 2ad & -a^2 - b^2 + c^2 + d^2 \end{bmatrix} \tag{4.5}$$

This conversion is also applied in the vector rotation function, which takes a quaternion and a vector as inputs and post-multiplies the rotation matrix derived from the quaternion with the vector.

Quaternion to Euler Angles

While direct conversion formulations exist for the ZYX convention case, in this work the previously defined Quaternion to Rotation Matrix conversion is used. The quaternion is transformed into its corresponding rotation matrix, from which the Euler Angles are then derived. From $R = Z_\phi Y_\theta X_\psi$ it is possible to obtain:

$$\phi = arctan\left(\frac{R_{21}}{R_{11}}\right), \quad \theta = arcsin(-R_{31}), \quad \psi = arctan\left(\frac{R_{32}}{R_{33}}\right) \tag{4.6}$$

4.2 Main

The description of the tool will follow the flow chart of its 'Main' script, which calls and runs the principal subsections (figure 4.1). Each block is discussed and described in detail, including its functions and data handling. To run the different scripts and functions, the Main script adds the folders containing them to the current

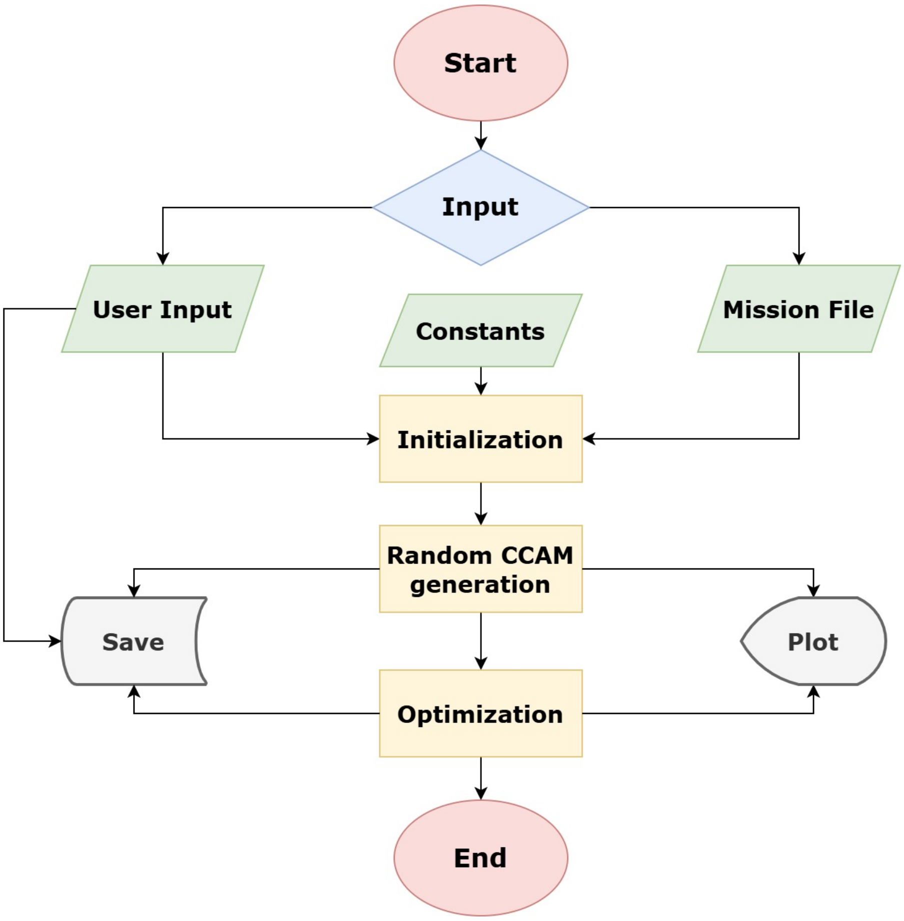


Figure 4.1: Main flow chart

matlabpath. The folders containing the tool's scripts and functions are helpful in the management of a large number of files. The list of folders the tool is divided into:

- conversions
- input requests
- integrator
- main
- models
- optimization
- quaternion algebra
- utilities

4.3 Input

The first task the tool performs once started is to ask a series of questions the user can answer by typing in the Matlab command window. This allows the user to define the initial conditions and the constraints of the mission for which they want to design the CCAM. Alternatively, it is possible not to define a new mission and load a previously simulated one whose input data has been saved.

Altitude definition

This script asks the user to input the orbit altitude at which the CCAM is simulated and computes the orbital rate as in equation 2.1.

Payload definition

The user is asked to decide if the mission being designed is a single-payload or multi-payload one. In the case of a single-payload mission, the separation time is set at the beginning of the simulation. For multi-payload missions, the separation time for each satellite must be defined, with the first separation always set at $t_0 = 0$. The user is then required to set the initial position of each satellite in the STA-1 reference frame to better simulate the adapters' position for a more accurate separation phase.

Release definition

The script asks to define the separation direction for each payload in the Body Frame. This release direction is represented as a unit vector and is multiplied by the separation velocity magnitude, requested as the next input, to compute the effective ΔV provided by the separation system.

Initial attitude definition

The AVUM+ initial attitude is requested in the format used by Avio in the Mission Data as input for the Morpheus simulator, specifically Euler Angles in the Orbital Basic reference frame using the ZYX (Pitch-Yaw-Roll) convention. This script also includes a request for the angular range at separation. In the case of delayed payload releases, the initial angular range is linked to the first separation.

Manoeuvre definition

The tool requests the definition of the CCAM structure as a vector of integers, where each integer corresponds to a maneuver block. The user is free to decide both the order and the number of blocks, and repeated blocks are permitted. The only requirement is that the Main Engine boost must be set as the final block. The block IDs are:

- RACS boost ID: 1

- Dummy RACS boost ID: 2
- Coasting Phase ID: 3
- Main Engine boost ID: 4

so in the case of the Sentinel Z40RP mission the structure of the manoeuvre is defined by the vector $(1, 2, 4)$, while for VC04 Biomass it is $(1, 1, 3, 2, 4)$.

Main Engine boost definition

This script allows the user to define the constraints regarding the Main Engine boost at the end of the CCAM. The user is asked to constrain the manoeuvre duration by choosing the desired angular range to be reached by the end of the slew phase preceding the AVUM+ boost. Concerning that same instant, the tool requires the definition of the attitude of the launch vehicle in the form of Euler Angles with the same reference and convention of the initial attitude input. This allows to set the direction of the boost, while its duration, or rather the provided ΔV, is requested afterwards.

MCI definition

The user is asked to define the Mass, Centre of Gravity and Inertia matrix of the AVUM+ stage. Before doing that, the tool gives the option of deactivating the Force-Torque coupling model for RACS thruster boosts which skips the Inertia matrix definition as it would be unused in that case.

RACS boost thrust definition

The resultant thrust provided by the RACS thrusters during the RACS boost phase is required as an input. This particular quantity is chosen as an input instead of the single RACS thrust as it is obtainable from the LV Body Force Plot of the Morpheus simulator.

$$T_{RB} = 2 \cdot T_{RACS} \cdot cos(10°) \tag{4.7}$$

While the constant thrust hypothesis is based on the idea that the RACS fuel consumption during the CCAM is only a fraction of the usable propellant, the value of the RACS thrust is required as an input since it depends on the previous mission phases that are unknown to the tool.

Runs definition

The number of randomly generated manoeuvres is required as an input, as well as the number of agents to optimize (the population) which cannot be lower than that of the random runs. As discussed in the dedicated section, because of the optimization algorithm chosen, at least 4 agents, and thus at least 4 random runs, must be selected.

The data entered by the user in this section is stored in a dedicated structure array called IN. In addition, the choice to optimize or not the RACS consumption is presented to the user. The details and the implications of this choice are discussed in the section dedicated to the cost function however, it is necessary to mention it as part of the input process despite not being part of the input data.

4.4 Initialization

After getting the required input data from the user, the tool proceeds with the initialization of the matrices dedicated to parameter handling and output storing while also computing values required by other sections of the tool, like the initial state vector among others. Additionally, in this section, the user is asked for a last input definition concerning the boundaries of the manoeuvre parameters. Afterwards, the tool computes the time constraint from the angular range inputs and checks if the inserted parameter boundaries allow to generate manoeuvres that respect that constraint.

Initialization and bounding

This is the main script of this section. Beyond the initialization of the output matrices, it generates parameter matrices, initializes the state vector, computes the value of several simulation variables and lets the user define the parameter bounds. To do so, it uses the IN structure array and the simulation constants as inputs.

Parameter matrices

Before the initialization of the output matrices, the script defines the matrices for parameter storing and handling and the boundaries of each parameter. Since the manoeuvre is defined dynamically for each mission, the parameter set is variable in quantity and order. For each run, a parameter vector is created with its components arranged in the same order as the manoeuvre blocks. For instance, the

Sentinel Z40RP mission features parameter vectors of 16 elements. In general, the

WT	Slew 1	Slew 2	WT	RB	ω_{Roll}	ω_{Yaw}	WT	...

...	RB_{dummy}	WT	Slew 1	Slew 2	WT	ME	ω_{Roll}	ω_{Yaw}

RACS boost
Dummy RACS boost
Main Engine boost

parameter matrix is a $N_{runs} \times N_p$ matrix where N_{runs} is the number of manoeuvres generated and N_p is the number of parameters of that CCAM. A parameter identification matrix is defined to identify each parameter vector component for the following computations. It is a $3 \times N_p$ matrix with each row describing the parameters differently.

- Row 1: Block ID
- Row 2: Parameter ID
- Row 3: Parameter type (1 - duration, 2 - angular velocity)

Referring again to the Sentinel Z40RP mission one gets the following parameter identification matrix:

1	1	1	1	1	1	1	2	2	4	4	4	4	4	4	4
1	2	3	4	5	6	7	1	2	1	2	3	4	5	6	7
1	1	1	1	1	2	2	1	1	1	1	1	1	1	2	2

As regards the parameter bounds, a $2 \times N_p$ matrix is defined with the lower and upper bounds of the parameters in each row. The only constrained parameters are the duration of the Main Engine boost, which is chosen and fixed during the input process, the Dummy RACS boost duration as it is the same for each CCAM and the angular velocities, whose upper and lower bounds are set to respectively 5°/s and −5°/s, which are the slew speed limits imposed for the CCAM manoeuvre.

Output matrices initialization

Once the definition of the parameter bounds is completed, the script computes the maximum and the minimum potential duration of the generated CCAMs performing the sum of the upper and lower time parameter bounds respectively. The maximum time is then used to obtain the maximum number of time steps of a simulation n_{max}. As analysed in the dedicated section, the simulator integrates the manoeuvre with a fixed time step defined in the simulation constants, with few exceptions. In the initialization of the output matrices, n_{max} defines the column dimension while the number of rows is again the number of generated manoeuvres N_{runs}. The output of the random CCAMs is stored in the OUT structure array. Furthermore, for each payload, a nested structure array is created inside OUT to separate the result matrices relative to each satellite. An in-depth analysis of the generated output is carried out in the section dedicated to the cost function.

State vector initialization

After the initialization of the output matrices, the tool proceeds with the definition of the initial state vector. This vector contains information on the attitude of the AVUM+ stage and the positions and velocities of all the separated satellites in the LVLH frame. This implies that the dimension of the state vector varies for each mission since it depends on the number of payloads. Being S the state vector and N_{PL} the number of satellites:

$$S \in \mathbb{R}^n, \quad n = 4 + 6 \times N_{PL} \tag{4.8}$$

The first 4 components of S are the elements of the quaternion describing the attitude of the launch vehicle. Given the Euler angles of the initial attitude in the Orbital Basic frame, the Euler Angles to Rotation Matrix conversion followed by the Orbital Basic to LVLH rotation yields the rotation matrix describing the launch vehicle orientation in the Local Orbital Frame. A final Rotation Matrix to Quaternion conversion gives the quaternion to be saved in the initial state vector. The initial position and velocity of each satellite are computed. Both are relative to the Launch Vehicle's COG, so the position of the COG in the STA-1 frame is subtracted from each satellite's coordinates in that same reference system to obtain the position in the Body Frame. The vector rotation function is then used to convert the position into the LVLH frame knowing the initial attitude quaternion. A similar approach is adopted for the initial velocity, with the release direction converted into the LVLH frame and the resulting unit vector multiplied by the separation ΔV. The possibility of delayed separations requires an additional computation step due to the rotation of the LVLH frame. When a satellite is released at a time offset Δt from the start of the simulation, its attitude in the Local Orbital frame at that moment differs from its attitude at t_0. This is because the Launch Vehicle while maintaining the same orientation in the Orbital Basic frame due to the absence of an active control system during separation, has traversed an angular range of $\omega \cdot \Delta t$. Small errors in the simulation of the first phases of the CCMA can lead to wrong evaluations of the manoeuvre, so it is necessary to update the initial state vector at each separation, making the initialization process more complex. However, it is possible to bypass this updating issue thanks to the angular range hypothesis and the knowledge of each satellite's release time. It is sufficient to rotate the initial attitude quaternion of $\omega \cdot \Delta t$ after converting it into a rotation matrix and then convert it back to a quaternion for each payload separated after t_0, keeping the rest of the initialization process unchanged.

Simulation variables

After completing the initialization of the state vector, additional variables required for the simulation are computed, namely the accelerations provided during RACS boosts and Main Engine boost, and the time at which the Active Control System (ACS) is reactivated. The former are obtained trivially because of the constant thrust and mass assumptions. The latter is defined as the time of the last payload separation with an additional delay set to allow the satellites to distance themselves from the AVUM+ stage and avoid collisions. This reactivation delay is usually set to 10 s. The set of simulation variables including the constants and the values computed during the initialization process are stored in the SIM structure array.

Time constraint

The initialization block includes the computation of the ideal CCAM duration for the given initial and final angular ranges which impose a time constraint on the manoeuvre generation. In particular, once the ideal time of manoeuvre is computed by

$$\begin{cases} t_{ideal} = \frac{\phi_{T_D} - \phi_{T_0}}{\omega}, & \phi_{T_D} > \phi_{T_0} \\ t_{ideal} = \frac{2\pi + \phi_{T_D} - \phi_{T_0}}{\omega}, & \phi_{T_D} < \phi_{T_0} \end{cases} \tag{4.9}$$

a generous margin of ± 60 s is added to allow a faster and more variable random generation. Then the maximum and minimum manoeuvre times computed after the parameter bounds definition are compared to the ideal time to check if ideally long runs can be generated. If not the user is asked to repeat the definition of the parameter bound matrix.

4.5 Random CCAM generation

The Monte Carlo-like generation of random CCAMs is one of the core blocks of this work. After all the required information regarding the mission and the CCAM initial conditions is acquired and the initialization process is completed, the tool generates random parameters limited by the user-defined upper and lower bounds and simulates the corresponding manoeuvre. After storing the output in the dedicated structure array, it evaluates the generated CCAMs through a cost function.

Parameter randomization

The tool operates at 10 Hz, that is to say, with a time step of 0.1 s. Because of this, the duration of each phase is defined up to the decimal value and otherwise approximated before integration to avoid issues due to the accumulation of rounding errors. The tool generates pseudorandom values from a discrete distribution bounded by the upper and lower limits defined for each parameter. Once a run is generated, the total manoeuvre time is computed and compared to the ideal duration. If the CCAM does not respect the time constraint with a margin, it is overwritten by a newly generated manoeuvre. This check is necessary to avoid wasting computation time for manoeuvres which would be unacceptable. Once this check is passed, the total number of time steps of the generated run is computed and the tool proceeds with the simulation.

Integrator

The integrator block performs the simulation of the manoeuvre. This tool section takes up the most computational effort and includes most of the models defined previously. The computation of the relative motion of the separated payload with respect to the Launch Vehicle is entrusted to Hill's equations:

$$\begin{aligned} \ddot{x} - 2\omega\dot{z} &= \frac{1}{m_c}F_x \\ \ddot{y} + \omega^2 y &= \frac{1}{m_c}F_y \\ \ddot{z} + 2\omega\dot{x} - 3\omega^2 z &= \frac{1}{m_c}F_z \end{aligned} \tag{4.10}$$

These equations are used for the trajectory analysis of rendezvous missions when the chaser vehicle is in close proximity to the target [6]. The hypotheses for their derivation are that the spacecraft are in circular orbits and that disturbances are neglected. These assumptions are compatible with the problem analysed in this work however, some modifications must be applied to the formulation in (4.10). Firstly, the CCAM does not include a chaser satellite. Instead, the controlled spacecraft is the Launch Vehicle, which remains fixed at the origin of the LVLH system. For this reason, the chaser mass m_c is replaced by the Launch Vehicle mass m_{LV}. However, the most significant difference with the formulation provided in [6] is the definition of the Local Orbital reference frame already discussed in the dedicated section. The LVLH system defined in this work is effectively a rotated version of the $LVLV_{Fehse}$ with *R-bar* $=$ $-$*R-bar*$_{Fehse}$ and the axes *R-bar* and *H-bar* switched in the triplet. This implies that when derived with the current LVLV frame Hill's equations take the form:

$$\begin{aligned}\ddot{x} + 2\omega\dot{y} &= \frac{1}{m_{LV}}F_x \\ \ddot{y} - 2\omega\dot{x} - 3\omega^2 y &= \frac{1}{m_{LV}}F_y \\ \ddot{z} + \omega^2 z &= \frac{1}{m_{LV}}F_z\end{aligned} \tag{4.11}$$

since $(x, y, z) = (x, -z, y)_{Fehse}$. With regard to attitude integration, the tool leverages the simplicity of quaternion dynamics, which follow a differential equation governed by the body angular velocity $\Omega = (P, Q, R)$. It is possible to represent Ω as a quaternion with null scalar part $W = (P, Q, R, 0)$ and perform a quaternion product to compute the time derivative of the spacecraft attitude quaternion:

$$\dot{q} = \frac{1}{2}W \otimes q \tag{4.12}$$

where the operator $\otimes$ is defined as:

$$q \otimes p = \begin{bmatrix} q4p1 + q3p2 - q2p3 + q1p4 \\ -q3p1 + q4p2 + q1p3 + q2p4 \\ q2p1 - q1p2 + q4p3 + q3p4 \\ -q1p1 - q2p2 - q3p3 + q4p4 \end{bmatrix} \tag{4.13}$$

These equations are integrated over time using the 4^{th}-order Runge-Kutta method (RK4). Given an initial value problem:

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0$$

and a time step $h > 0$ one defines:

$$\begin{aligned}y_{n+1} &= y_n + \frac{6}{h}(F_1 + F_2 + F_3 + F_4) \\ t_{n+1} &= t_n + h\end{aligned}$$

using

$$
\begin{aligned}
F_1 &= f(t_n, y_n)\\
F_2 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}F_1)\\
F_3 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}F_2)\\
F_4 &= f(t_n + h, y_n + hF_3)
\end{aligned}
$$

In the case studied, the dynamics functions are time-independent, simplifying the integration process.
The RK4 script is designed with a block structure that reflects the layout of the CCAM. A main script is responsible for initializing temporary vectors and matrices, and, depending on the simulated block, it invokes the appropriate integration script. Four distinct integration scripts are implemented, each tailored to the specific models corresponding to the respective CCAM blocks. After the initialization of the TEMP structure array for temporary output storage, the main script sets the simulation time at $t = 0$ and starts to simulate each block of the CCAM in the order it is defined.

RACS boost integrator

The RACS boost constitutes the primary and typically the initial block of the CCAM. When invoked by the main RK4 script, it retrieves the parameters of the manoeuvre block being simulated from the parameters matrix. When cycling through the time steps, it checks if any payload is released and updates the state of the separated ones. These first operations are shared between all the integration blocks. In each cycle the simulated time is increased by the time step value and, according to the phase simulated, the input vector U is assigned different values.

$$
U \in \mathbb{R}^n, \quad n = 4 + N_{PL} \tag{4.14}
$$

The first 3 components of U are the values of the angular velocities (P, Q, R) while the fourth component is the magnitude of the translational acceleration. The remaining components serve as flags for the separation of each payload. The Force-Moment coupling model is implemented for yaw slews and RACS boost phases. Once the U vector has been defined, the actual RK4 integration takes place by plugging the angular velocity and acceleration components into the dynamics equations. The state vector S is then updated and the cycle starts over. During each step the aspect angles between each RACS thruster axis and the separated satellites are computed according to the RACS contamination model. Given that the positions of

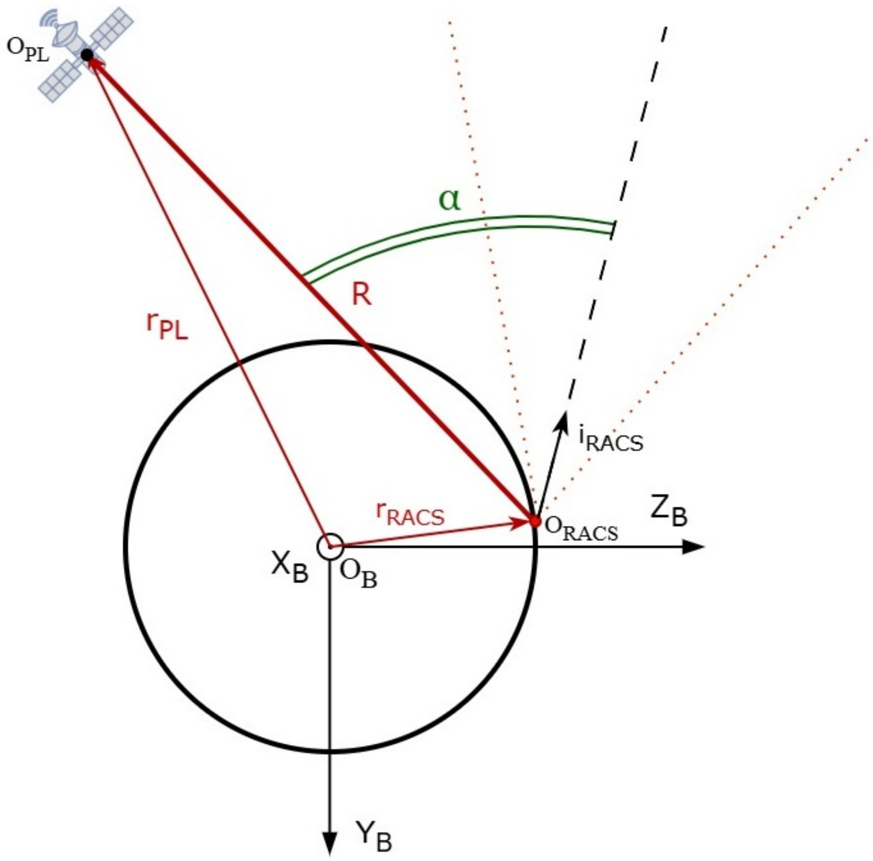


Figure 4.2: RACS thruster angle with separated payload (2D representation)

the thrusters are defined by the architecture of the AVUM+ stage and the relative positions of the separated satellites are calculated at each step, the relative position of the payload with respect to each thruster can be easily determined through vector subtraction.

$$\vec{R} = \vec{r}_{PL} - \vec{r}_{RACS} \tag{4.15}$$

The thrust direction of each RACS thruster is also defined and fixed allowing the computation of the aspect angle α by

$$\alpha = cos^{-1}\left(\frac{\vec{R}}{||R||} \cdot \vec{i}_{RACS}\right) \tag{4.16}$$

Dummy RACS boost integrator

This script integrates the Dummy RACS boost and follows the same structure as the RACS boost integrator script, with two key differences: no slew phases are simulated, and while the Force-Moment coupling model is applied during the boosting phase, attitude control is not implemented during the RACS boost.

Coasting Phase integrator

During the coasting phase, the Launch Vehicle remains in a free-drift state, performing neither slew nor boost manoeuvres. To achieve the required angular range for the subsequent Main Engine activation, this phase can extend for thousands of seconds, significantly increasing the simulation duration. The necessity of the tool to simulate tens of thousands of manoeuvres creates the need to increase the integration time step to fix this issue. The tool adjusts the step length based on the duration of the coasting phase and applies linear interpolation to populate the output matrices, whose dimensions are pre-established during initialization. This approach significantly reduces simulation time while maintaining a straightforward output matrix allocation process. As in all the integrator blocks, the angles between the RACS thrusters and the separated payloads are computed in each cycle.

Main Engine boost integrator

The Main Engine boost block is the last block of the manoeuvre. It has a similar structure to the RACS boost block as it features both slew and boost phases, however it also computes the contamination by the AVUM+ exhaust plume. The Force-Moment coupling model is applied exclusively to the yaw slew, under the assumption that the AVUM thrust vector is parallel to the X_B axis and passes through the COG. The attitude at the conclusion of the slew phase is a mission-critical constraint, and the likelihood of randomly generated slew parameters enabling the Launch Vehicle to achieve this orientation is negligible. To address this, a dedicated algorithm is employed at the start of the slew phase to compute the required roll and yaw angles needed for the spacecraft to attain the desired attitude. The core idea involves determining the matrix that defines the rotation necessary to achieve the target attitude relative to the Body Frame's orientation at the beginning of the slew. Firstly, the desired Euler Angles and the current quaternion are converted into rotation matrices.

$$EUL_{des}^{OB} \xrightarrow{EUL \rightarrow ROTM} R_{des}^{OB}$$
$$q^{LVLH} \xrightarrow{QUAT \rightarrow ROTM} R^{LVLH}$$

Then, the current attitude matrix is rotated into the Orbital Basic frame. The matrix that allows the conversion from the Orbital Basic frame into the Body Frame can be computed because the orientation relative to the Body Frame is defined as the identity.

$$R^{OB} = R_{LVLV \to OB} R^{LVLH} \tag{4.17}$$

$$R^{BODY} = I_3 \tag{4.18}$$

$$R_{OB \to BODY} = R^{BODY} (R^{OB})^{-1} \tag{4.19}$$

Now the desired attitude can be expressed in the current Body Frame. This orientation can be represented as a combination of three rotations. By defining it as a Roll-Yaw-Roll (XYX) manoeuvre, the roll (ϕ) and yaw (θ) angles needed for the Launch Vehicle to achieve the desired pointing for the Main Engine boost can be determined.

$$R_{des}^{BODY} = R_{OB \to BODY} R_{des}^{OB} \tag{4.20}$$

$$R_{des}^{BODY} = X_\phi Y_\theta X_\psi \tag{4.21}$$

$$R_{des}^{BODY} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{4.22}$$

$$\phi = atan2(r_{21}, -r_{31)} \tag{4.23}$$

$$\theta = cos^{-1}(r_{11}) \tag{4.24}$$

$$\psi = atan2(r_{12}, r_{13)} \tag{4.25}$$

Notice that the Launch Vehicle will reach the programmed attitude with a random roll orientation, as the third ψ rotation is not performed in the slew manoeuvre. This does not pose any issues, as the Launch Vehicle is guaranteed to be outside the RACS contamination zone. The only critical requirement is the Main Engine thrust vector's direction, which is specified by the pitch and yaw Euler Angles (ZYX convention). Typically, in defining the desired attitude for the Main Engine boost, the roll Euler Angle is either left undefined or assigned a default value, as it does not factor into verifying the final orientation. Particular attention must be paid to the $R_{OB \to BODY}$ matrix in equation (4.17). One could think of using the value of the Angular Range at the beginning of the slew phase to define this conversion matrix. While counter-intuitive, the conversion in (4.17) is carried out with the desired Angular Range as it can be deduced from (4.26).

$$\begin{gathered} R^{OB} = f(\phi_T) \implies R_{OB \to BODY} = f(\phi_T) \\ R_{des}^{BODY} = R_{OB \to BODY} R_{des}^{OB} = f(\phi_T) \\ (\phi, \theta, \psi) = f(\phi_T) \implies \phi_T = \phi_{Tdes} \end{gathered} \tag{4.26}$$

Once the slew angles are determined, the CCAM parameters are updated accordingly. First, the slew durations are set, and the necessary angular velocities are calculated. If these exceed the maximum slew speed, the tool attempts to adjust the slew times. The subsequent waiting time is then reduced to ensure the total CCAM duration remains unchanged. If the waiting time is insufficient to extend the slew phase and achieve the required angles at maximum angular velocity, the integration terminates, and the run is discarded. If the slew computation succeeds, the integration block operates similarly to the RACS boost integrator while also computing the AVUM+ contamination during the boost phase.

Output storage

After the manoeuvre is simulated, the output data needed for evaluation and post-processing is stored in the dedicated OUT structure array. Following is the list of saved output. For each payload:

- Relative position in the LVLH Frame
- Relative velocity in the LVLH Frame
- RACS thrusters angle
- Main Engine contamination level

and for the whole mission:

- Parameters
- Total CCAM duration
- Launch Vehicle attitude

Throughout this process, the output data is utilized to extract the information required by the cost function to evaluate the manoeuvre.

Cost function

The definition of the cost function is fundamental for the evaluation of the simulated manoeuvres and for directing the optimization process toward specific goals. The primary objectives of the CCAM are to increase the relative distance between the AVUM+ stage and the payload to prevent collisions during subsequent manoeuvres, to position the Launch Vehicle in the optimal attitude for de-orbiting, and to minimize payload contamination from RACS thrusters and Main Engine plumes. While

the first two goals are straightforward and impose no significant constraints on the mission designer, contamination avoidance is defined as follows: *"The launch vehicle and the launch pad shall not generate organic deposits exceeding 4 mg/m^2 on the spacecraft from the beginning of its encapsulation up to its separation from the launcher, and the following collision and contamination avoidance manoeuvre"*. The requirement is supposed to be allocated half to the ground phases, and the other half to the flight phases, effectively making the contamination limit considered in this work equal to 2 mg/m^2.
The cost function is defined by 3 evaluation criteria deriving from the objectives of the CCAM and that are applied to each payload:

- Distancing evaluation
- RACS angle evaluation
- Main Engine contamination evaluation

and by 2 additional criteria that apply to the manoeuvre as a whole:

- CCAM duration evaluation
- RACS boost consumption evaluation

For each criterion, the cost function assigns a score ranging from 0 for the worst case to 5 for the ideal case. The evaluation method for each criterion is distinct and tailored to its specific requirements. A final score is calculated by weighting each criterion according to mission priorities. In multi-payload missions, each payload is assessed individually, and an overall score for the entire manoeuvre is determined through a dynamic weighting process. Lower scores are given greater weight, influencing the final evaluation more significantly than higher scores, encouraging the optimizer to focus on improving weaker aspects.

Distancing evaluation

After storing the relative position of each payload with respect to the Launch Vehicle into the OUT structure, the relative distance is computed and used to check whether the separated satellites tend to re-approach the AVUM+ stage at any point during the CCAM. A dedicated script takes the relative distance vector and saves the value of the absolute minimum re-approach distance. The maximum score of 5 is given to manoeuvres in which the distance is monotonously increasing or the re-approach is at distances $> 10^4$ m while the minimum score of 0 is assigned in case a re-approach reaches a distance lower than 200 m. The 200 threshold is chosen because it is the extension of the RACS thrusters cones, so a re-approach at these distances increases the chances of contamination. This case includes both re-approaches that begin inside the danger zone prolonging the time spent inside it, and re-approaches that start outside the 200 m mark with the Launch Vehicle re-entering the contamination area after successfully evading it. A score of 4 is assigned if the re-approach distance is greater than 500 m and for values between 200 − 500 m the score is derived by linear interpolation.

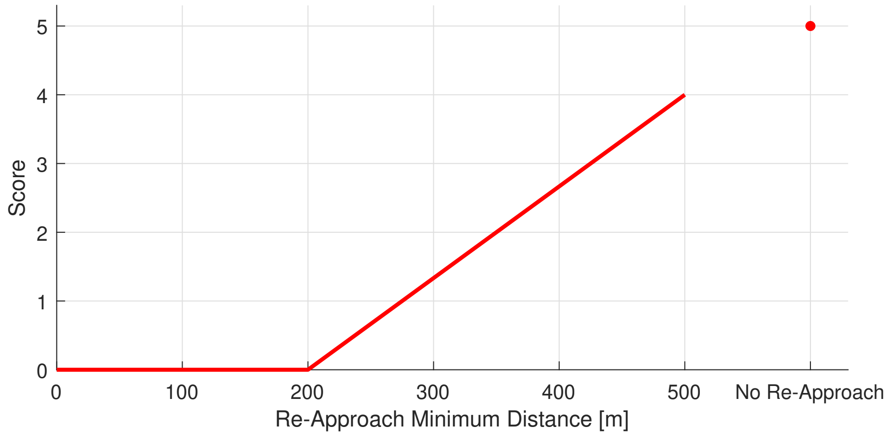


Figure 4.3: Re-approach evaluation

RACS angle evaluation

The models used in this work are unable to reliably estimate the level of contamination caused by the RACS thruster plumes. To assess this aspect of the CCAM, the aspect angles between each thruster and the released satellites are calculated at every time step and compared to the cone half-angle of the RACS contamination zone. For each thruster, the minimum RACS angles observed when the payload is within a relative distance of < 200 m are identified, and the smallest among them is used for evaluation. The goal is to guide the tool to generate CCAMs where the payload never enters any contamination cone, making the absolute minimum RACS angle within the 200 m threshold an ideal indicator for this condition. However, multi-payload missions make the research for a manoeuvre that satisfies this requirement quite challenging and, for some release directions, basically impossible. To prevent hindering the subsequent optimization process, it is essential to incorporate a more flexible evaluation criterion. Passages inside the contamination areas are therefore allowed, as long as they are brief. In the case of RACS angles lower than the cone half angle, the time spent inside any danger zone is computed for each RACS and their sum is used for evaluation. The complete criterion features a first grading of the time spent inside the RACS cones, assigning a score of 4 if the payload does not cross any cone. The score lowers with the increase of the time measure following a hyperbolic law:

$$score_{time} = \frac{12}{t_{RACS} + 3} \tag{4.27}$$

that degrades the evaluation rapidly with increasing times, reaching the value of 1.5 for $t_{RACS} = 5$ s.

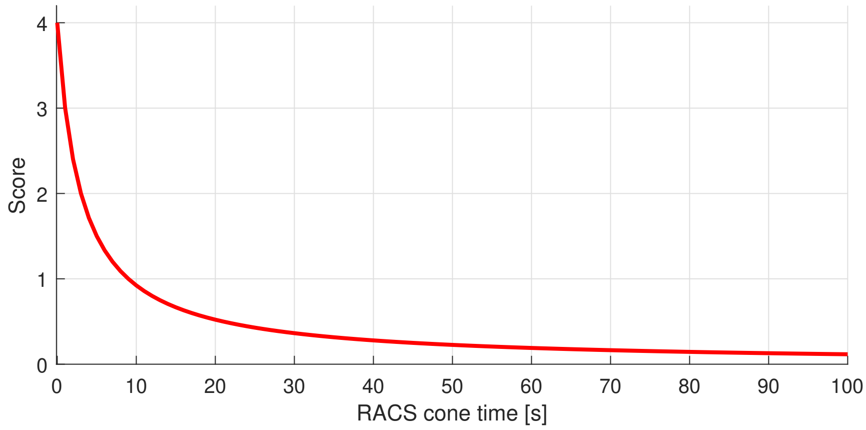


Figure 4.4: RACS cone time score chart

For cases with null RACS cone times, the minimum angle is evaluated, adding 1 point to the score. A maximum score of 5 is achieved for minimum angles of 90°, with lower angles reducing the score linearly to 4 at a minimum angle of 26°

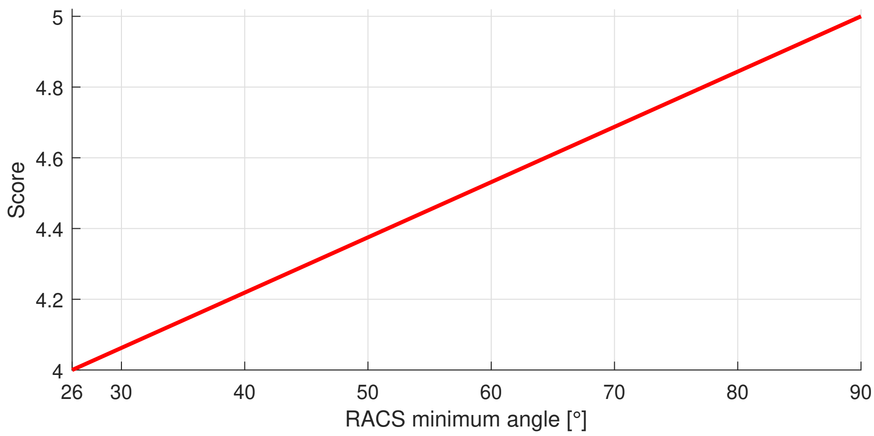


Figure 4.5: RACS minimum angle score chart

Main Engine contamination evaluation

The Main Engine contamination is evaluated based on the organic deposit limit of 2 mg/m^2 during the flight phase. The mission designer aims to minimize this value as much as possible, ideally approaching zero. For this reason, the score for the total contamination of the payload is determined using a hyperbolic scaling law:

$$\begin{aligned} score_{ME} &= \frac{1}{a \cdot cont + b} + c \\ a &= \frac{250}{399},\ b = \frac{10}{57},\ c = -\frac{7}{10} \end{aligned} \tag{4.28}$$

assigning the maximum value of 5 for no contamination and the minimum 0 for values greater or equal to 2 mg/m^2. The score decreases rapidly, emphasizing the importance of achieving lower pollution levels. For instance, a contamination level of 0.1 mg/m^2 is assigned a score of 3.5.

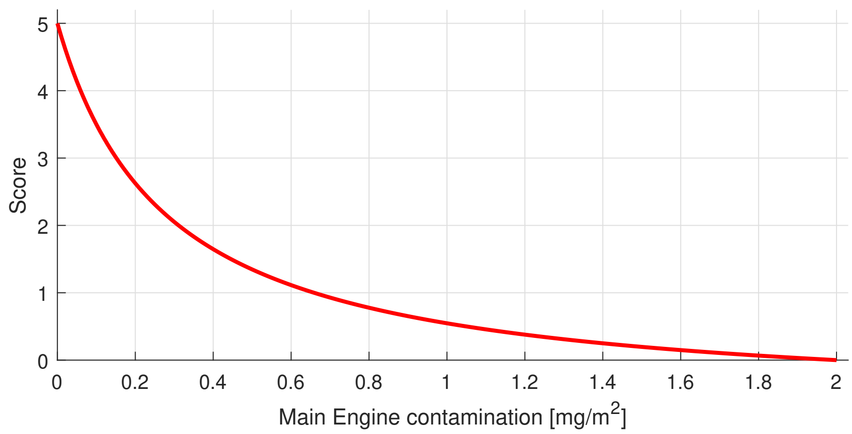


Figure 4.6: Main Engine contamination score chart

CCAM duration evaluation

The total duration of the CCAM is compared to the ideal time and evaluated. The time margin used for the manoeuvre generation is set as the lower bound, corresponding to a score of 0, while the ideal time gets the best score of 5. The ranking law is linear.

RACS boost consumption evaluation

This is the only optional criterion, which the user can disable during the input process. RACS consumption is evaluated exclusively during the boost phases. Due to the adopted constant acceleration model, the total boost time directly corresponds

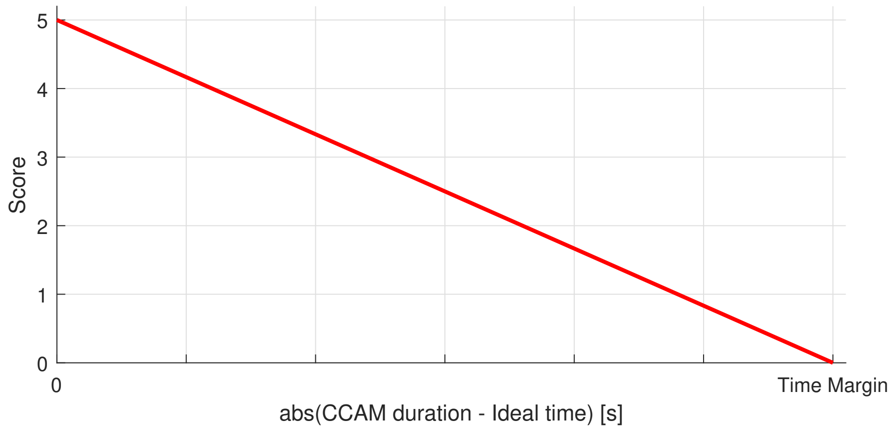


Figure 4.7: CCAM duration score chart

to the total ΔV. The parameter matrices are utilized to calculate the minimum and maximum RACS ΔV achievable within the defined parameter bounds, and the ΔV of the evaluated run is compared to these limits. The scoring follows a linear scale, assigning the lowest score to the maximum ΔV and the highest score to the minimum ΔV.

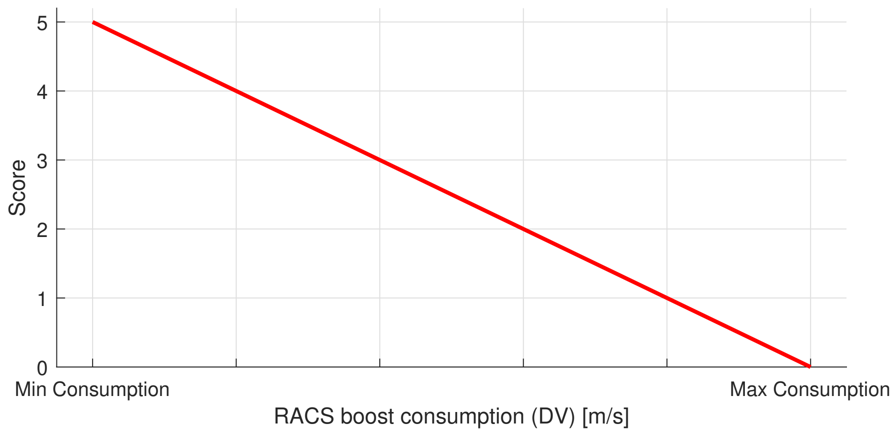


Figure 4.8: RACS boost consumption score chart

Score weighing

The scores are weighted in two stages. First, they are weighted across the payloads to produce a single score vector representing the entire manoeuvre. Second, they are weighted across the different criteria to calculate the final grade. To ensure that lower payload scores have a greater impact on the general score vector, a dynamic weighting system is applied. Perfect scores of 5 are given the lowest weight coefficient of 1, while scores of 0 are assigned the highest weight coefficient of 10, with intermediate scores following a linear scale. The final score values are then averaged across all satellites based on the assigned weight factors. An example of a multi-payload mission with 3 satellites:

	Distancing	RACS angle	ME contam.
PL_1	5	3.2	4.8
WC_1	1	4.24	1.36
PL_2	5	0.8	5
WC_2	1	8.56	1
PL_3	5	5	4
WC_3	1	1	2.8
Result	5	1.84	4.40

$$Result = \frac{\sum_{n=1}^{N_{PL}} PL_n \times WC_n}{\sum_{n=1}^{N_{PL}} WC_n}$$

It is evident that the low RACS angle score of Payload 2 significantly affects the final score of the manoeuvre. Once the final score vector is compiled, incorporating the evaluations of CCAM duration and RACS ΔV, the second weighing step is performed to calculate the final score for manoeuvre ranking. In this step, fixed weighting coefficients are applied to average the various scores into a single final value.

Table 4.1: CCAM weight vector (second row: suppressed RACS ΔV evaluation case)

Distancing	RACS angle	ME cont.	Duration	RACS ΔV
0.20	0.35	0.25	0.10	0.10
0.23	0.40	0.27	0.10	suppressed

4.6 Optimization

The problem of optimization consists of maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function [11]. In the context of this work, the input values are represented by the CCAM parameters and the real function to optimize is the cost function that evaluates the manoeuvre goodness. Due to the scoring strategy adopted, the optimization of the CCAM consists of finding the maximum of the cost function output.

Optimization algorithm choice

The CCAM optimization problem can be considered a multi-objective optimization. At least 3 different objectives to optimize are defined: distancing, RACS angles and Main Engine contamination. If one considers the manoeuvre duration an optimization objective and if RACS consumption optimization is turned on, then the number is even higher. Additionally, it is impossible to prove that these goals do not conflict with each other as the payload configurations and the manoeuvre structures differ considerably between each mission. Furthermore, the existence of an absolute optimal solution is not assured, especially for simpler single payload missions where the available degrees of freedom are exceptionally numerous and multiple acceptable solutions from the point of view of contamination and collision avoidance are available. For these reasons, the optimization section of the tool is entrusted to an Evolutionary Algorithm which, by reproducing the essential elements of biological evolution, searches for maxima, even though mostly local, of the cost function output. Generally, an evolutionary algorithm features the following steps:

1. Random generation of the first population of individuals
2. Evaluation of the fitness of each individual
3. Selection of the parents, usually the fittest individuals
4. Offspring production
5. Application of mutations on the offspring
6. Replacement of the less fit individuals with new individuals
7. Iteration from point 2

The ability to find multiple maxima in large spaces of candidate solutions is an additional benefit of evolutionary algorithms, as providing the mission designer with several optimal CCAM proposals allows them to choose the most suited one for the specific case.

Differential Evolution

The evolutionary algorithm chosen for the optimization section of the tool is the Differential Evolution (DE) algorithm. This algorithm does not require the problem to be differentiable as it does not use the gradient of the problem to find solutions, so it is suitable for problems that are not even continuous. The problem is effectively

treated as a black box that provides a measure of the fitness of candidate solutions. There exist several variants of DE algorithms that differ in the way the candidate solutions are moved around in the space of solutions by the combination of the existing agents of the current population.
Given:

- $f : \mathbb{R}^n \to \mathbb{R}$ the cost function to maximise
- $\mathbf{x} \in \mathbb{R}^n$ a candidate solution (agent)
- $NP \geq 4$ the population size
- $CR \in [0, 1]$ the crossover probability
- $F \in [0, 2]$ the differential weight

The basic DE algorithm:

1. Initializes all agents with random positions in the solution space
2. For each agent $\mathbf{x}$ picks 3 agents $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ distinct from each other and from $\mathbf{x}$ ($\mathbf{a}$ is the base vector)
3. Picks a random index $R \in (1, ..., n)$
4. For each $i \in \{1, ..., n\}$ picks a uniformly distributed random number $r_i \sim U(0, 1)$
5. Being y_i a component of the agent's potentially new position $\mathbf{y}$, if $r_i < CR$ or $i = R$ then defines $y_i = a_i + F \cdot (b_i - c_i)$ otherwise $y_i = x_i$.
6. If $f(\mathbf{y}) > f(\mathbf{x})$ then replaces the agent $\mathbf{x}$ with the candidate solution $\mathbf{y}$
7. Iterates from step 2 until a termination criterion is met [12]

CCAM optimization

After the randomly generated CCAMs are simulated and evaluated, the tool selects the NP best runs to serve as the initial generation for the DE loop. The algorithm then mutates the entire population, with each new agent being simulated and assessed using the cost function. Improved maneuvers replace their outdated parents, and the process continues until either a predetermined maximum number of generations is reached or the score difference between the best and worst agents in the same generation falls below the tolerance coefficient ϵ. Each generation is processed

using the same script that integrates the random CCAMs and evaluates them with the cost function. Upon exiting the loop, the final generation undergoes one last simulation, and its output matrices are saved in the OPT structure array, mirroring the entries in the OUT structure. Additionally, the vector containing the best score of each generation is saved to check how much improvement was brought by each iteration.
The DE parameters chosen for CCAM optimization are:

$$CR = 0.8 \quad F = 0.1 \quad NP = 100 \tag{4.29}$$

The termination criterion uses a tolerance coefficient of $\epsilon = 0.001$ and a maximum generation count of $GEN_{max} = 100$. This setup balances computation time with the number of iterations, allowing the DE loop to simulate up to $GEN_{max} \times NP = 10^4$ CCAMs if the tolerance criterion is unmet. This number of runs remains manageable within a reasonable computation timeframe.

4.7 Post-Processing

The post-processing section of the tool includes plotting and data saving.

Plot

The plotting section is essential for visualizing and analyzing the tool's output. While the scoring system provides a detailed evaluation of the generated manoeuvres, a final review by the mission designer is necessary to ensure the quality of the solutions. The tool supports plotting both randomly generated and optimized CCAM sets. Additionally, users can upload and visualize the saved outputs of previously simulated missions. For multi-payload missions, users can select the number of payloads to display on a single figure. After making these selections, users can choose to plot a specific CCAM, compare multiple manoeuvres on the same graphs, or display the N_{best} best-evaluated runs. For clarity, when comparing multiple runs in multi-payload missions, each payload is plotted on separate figures.
The figures plotted for a single CCAM (multiple payloads on the same graph):

- Distance vs Time
- RACS minimum angle vs Thrusters
- RACS angle vs Time

- Main Engine contamination vs Time
- R-bar vs V-bar
- H-bar vs V-bar
- 3D LVLH trajectory

The figures plotted for multiple CCAMs, either the N_{best} or any user-picked runs (each payload on a separate figure):

- Distance vs Time
- RACS minimum angle vs Thrusters
- Total Main Engine contamination vs Runs
- Total manoeuvre time vs Runs
- R-bar vs V-bar
- H-bar vs V-bar
- 3D LVLH trajectory

Save

After the optimization process, users have the option to save various data sets for different purposes. To bypass the input phase for a previously defined mission, the IN structure array can be saved in an input file. This saves all user-provided information from the mission definition process into a .m file, allowing users to skip this lengthy phase in future simulations. Additionally, output data can be stored for later analysis or visualization. Both randomly generated and optimized CCAMs can be saved in .mat files and reloaded during the plotting phase. A key feature of the saving script is its ability to convert the output into mission data suitable for the Morpheus simulator, enabling the reproduction of the computed CCAM. This mission data includes the Euler Angles relative to the Orbital Basic frame at the end of the slew phases for each manoeuvre block, as well as the duration of each phase. Users can choose a single run from either the random or optimized CCAMs and save this data in a text file. This file can then be used by mission designers to define the manoeuvre in the main simulator

Chapter 5

Results

This chapter presents detailed representations of the results achieved through simulations carried out using the developed tool. It includes a set of candidate optimized manoeuvres for the reference missions, which are compared to the output produced by the Morpheus simulator when supplied with the mission data generated by the tool corresponding to those CCAMs. Additionally, the results for a multi-payload mission are also discussed.

5.1 Single-payload results

The initial conditions and constraints for the reference missions Sentinel Z40RP and VC04 Biomass are provided as inputs for identifying candidate optimal CCAMs. The tool generates 1000 random maneuvers, selecting the best 100 runs to feed into the Differential Evolution algorithm for optimization over a maximum of 100 generations. In all reported cases, the ϵ tolerance termination condition is not met, leading the tool to perform the maximum number of iterations. The Sentinel Z40RP mission involves a short and simple CCAM with a single RACS boost. Consequently, the RACS consumption optimization criterion is disabled in this analysis, as differences in RACS ΔV would be negligible for such a maneuver. This approach, however, is not applicable to the VC04 Biomass CCAM, which features a more complex maneuver with two RACS boosts and an extended coasting phase. In this case, the ΔV-optimized scenario is analyzed to evaluate its effectiveness in minimizing RACS consumption.

Sentinel Z40RP

The payload of the Sentinel Z40RP mission is deployed longitudinally with a ΔV of 0.9 m/s provided by the separation system. At the time of separation, the AVUM+

orientation is specified by the Euler Angles relative to the Orbital Basic frame $\{\psi_0, \theta_0, \phi_0\} = \{-0.2915, 0.0801, -1.7540\}$ rad, and its position in the orbit corresponds to the angular range $\phi_{T0} = 34.75°$. The Main Engine boost for de-orbiting the Launch Vehicle is planned to occur at $\phi_{Tf} = 65°$, with the spacecraft oriented in the direction $\{\psi_f, \theta_f, \phi_f\} = \{0, 0, 1.92\}$ for a 67 s burn. The satellite's COG is aligned with that of the Launch Vehicle, with an offset of 1.36 m along the X_B axis. Following payload separation, the AVUM+ mass amounts to 791.4 kg, and during boost phases, the RACS thrusters provide a total thrust of 300 N. The candidate solution found by the Differential Evolution loop is defined by the following parameters:

RACS boost						
WT	Roll dt	Yaw dt	WT	RB	Roll ω	Yaw ω
115.5	39.5	24.8	35.2	59.1	2.5	3.5

Dummy RACS boost	
WT	Dummy RB
33.6	10

Main Engine boost						
WT	Roll dt	Yaw dt	WT	ME	Roll ω	Yaw ω
7.7	31.7	24.4	116.5	67	-1.3	4.9

which yield the following scores:

Distancing	**RACS angle**	**ME cont.**	**Duration**	**RACS ΔV**
5	4.7658	4.9928	4.9935	suppressed

CCAM total score
4.9073

The proposed CCAM is highly effective based on the evaluation criteria for distancing and contamination, and it demonstrates excellent precision in reaching the desired angular range for the final Main Engine boost. The total manoeuvring time is 565 s, matching the ideal value of 565.077 s with only a negligible error attributable to the tool's time sensitivity. The CCAM's output is analysed and compared with the results from the Morpheus simulator, where the manoeuvre is replicated using the corresponding mission data.

Starting with the distancing evaluation, as shown in figure 5.1, the relative distance of the separated payload increases consistently throughout the simulation, ensuring that the separated satellite does not re-approach the AVUM+ stage at any point. Regarding the RACS angle evaluation, the tool aims to maximize the minimum

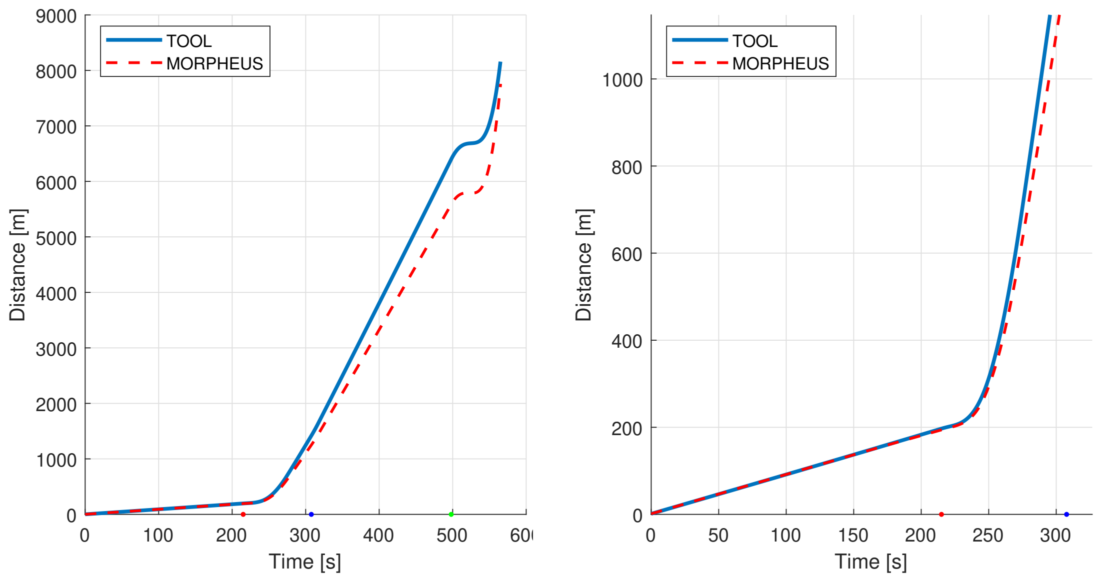


Figure 5.1: Sentinel Z40RP relative distance

angle achieved while the Launch Vehicle remains within the contamination zone. This objective is simplified by the presence of a single payload separation. The minimum RACS angle, recorded at 75°, is achieved by Thruster 1 precisely at the 200 m threshold distance. With a 49° margin from the RACS cone half-angle, this solution more than satisfies the RACS contamination requirement.

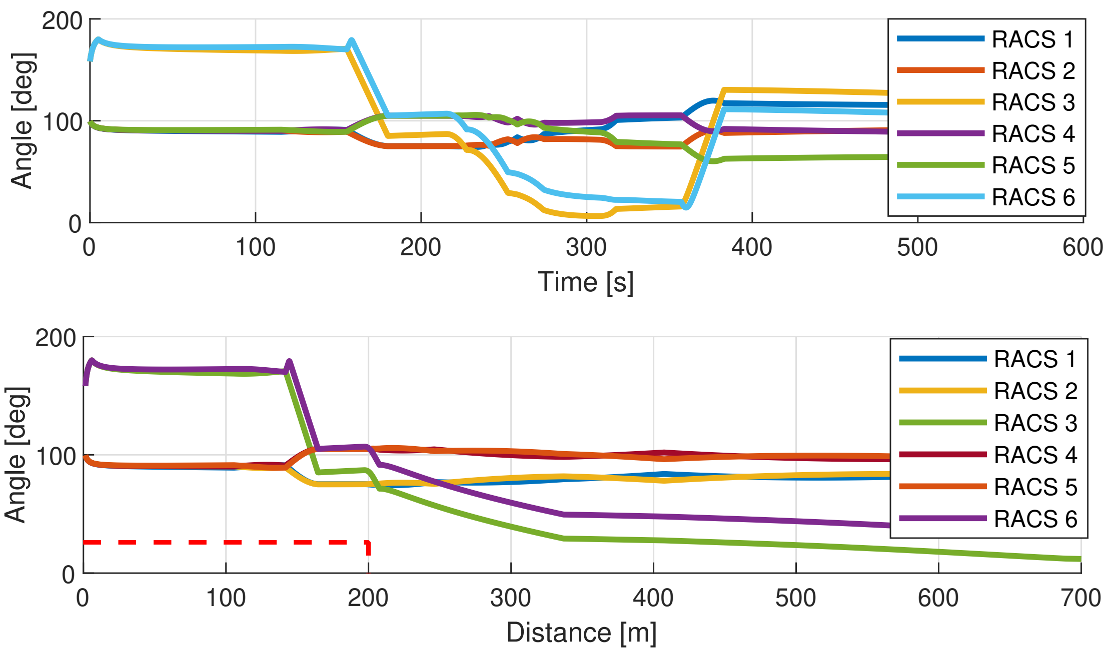


Figure 5.2: Sentinel Z40RP RACS angles (tool output)

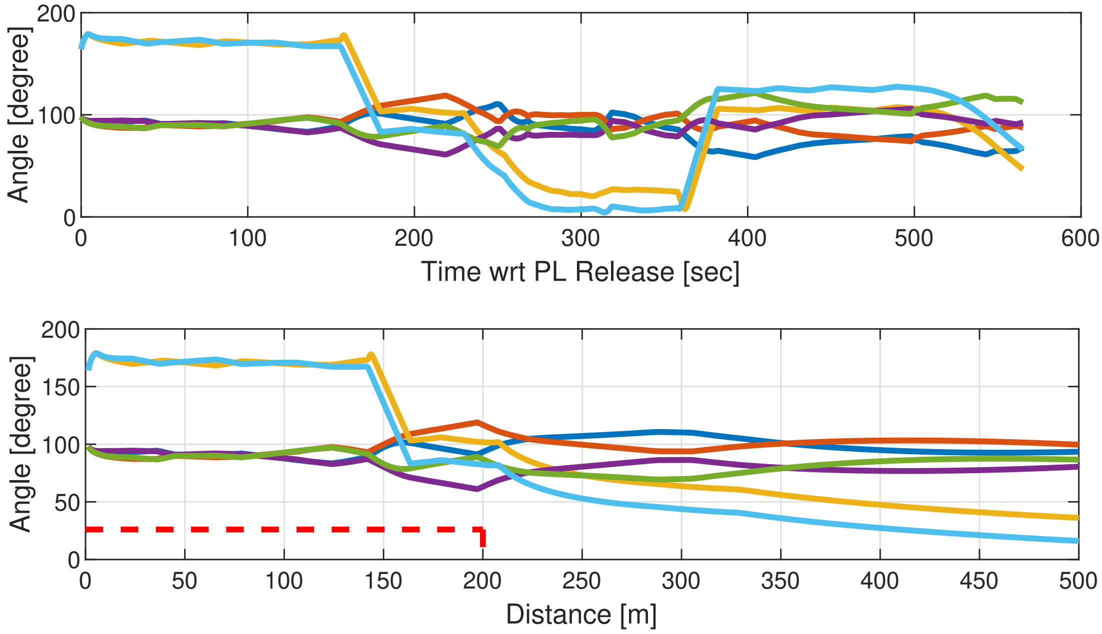


Figure 5.3: Sentinel Z40RP RACS angles (Morpheus output)

From the relative distance graph in figure 5.1, it can be observed that the satellite does not exceed the 10^4 m threshold, which ensures no Main Engine contamination. Despite this, the proposed CCAM generates minimal levels of deposits on the payload surface, as shown in the plots of figure 5.4.

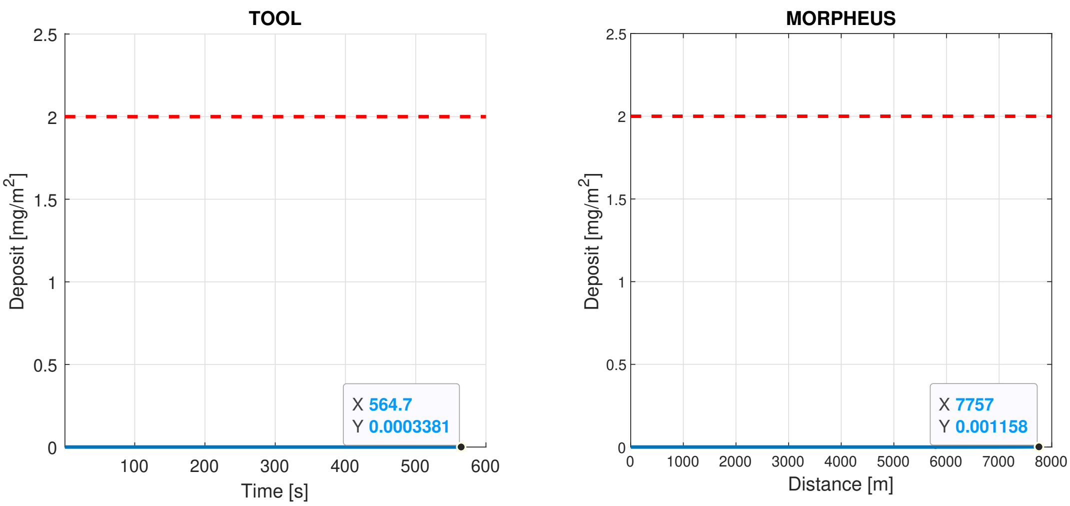


Figure 5.4: Sentinel Z40RP Main Engine contamination

To conclude this first analysis, the trajectory in the LVLH frame of the separated satellite, relative to the AVUM+ stage fixed at the origin of the reference system, is plotted and compared to that of the same manoeuvre reproduced on the Morpheus simulator. As visible in figure 5.5, the trajectories are very similar within the orbital plane during the initial part of the manoeuvre and tend to diverge over time. In particular, the motion in the (V-bar, H-bar) plane shows noticeable differences between the two plots. The origin of this divergence lies in the assumption of constant mass, thrust, and COG offset, which simplifies the Force-Moment coupling model. The Morpheus simulator accounts for the variation of these parameters, which can influence the dynamics of the spacecraft. Despite this simplification, the measures related to the CCAM requirements remain highly reliable, as they are primarily influenced by the simulation's accuracy during the initial phases and the attitude during the Main Engine boost. An additional source of error in the simulation arises from inaccuracies in the initial conditions, as Sentinel Z40RP is a non-final version of the VV25 mission. Consequently, the payload position, separation velocity, and Launch Vehicle attitude are approximated using the Morpheus simulator's output from the standard mission.

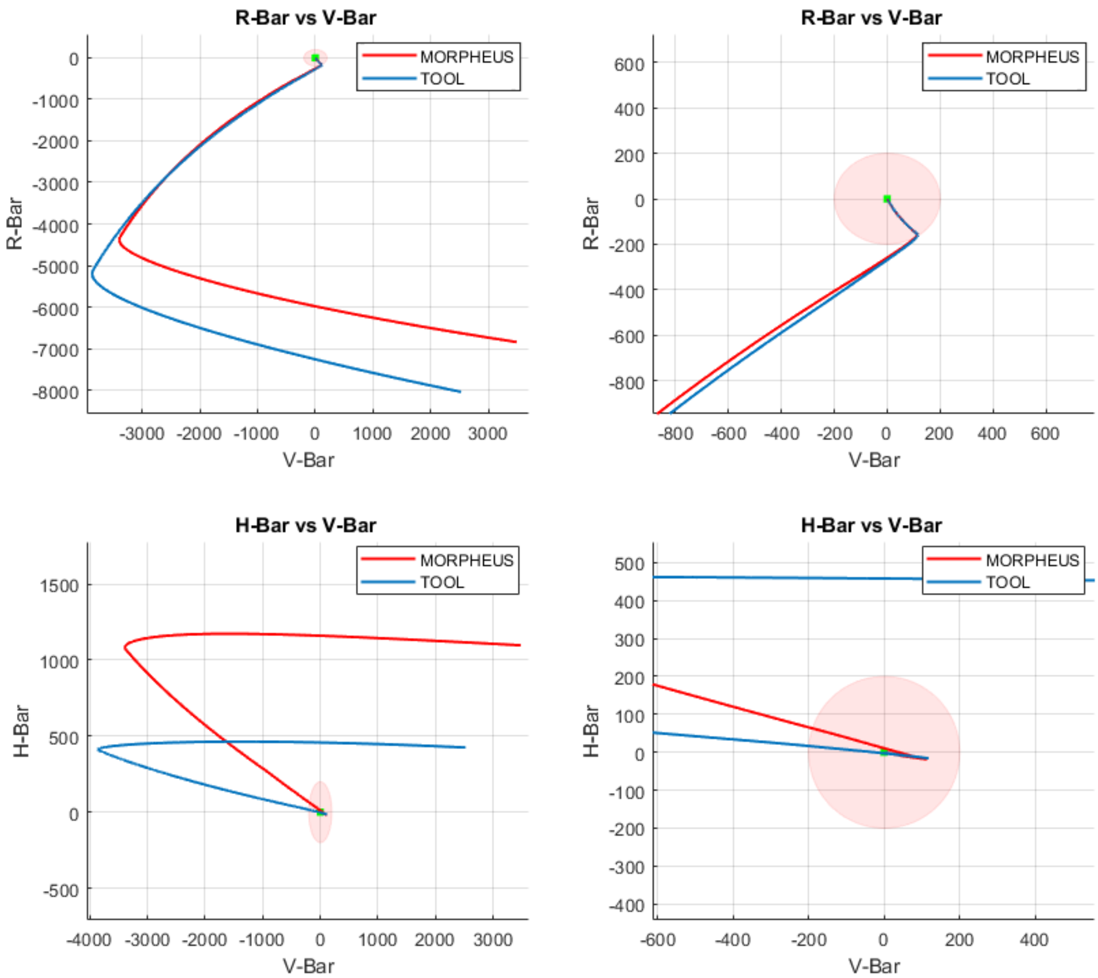


Figure 5.5: Sentinel Z40RP LVLV trajectory

VC04 Biomass

The Biomass satellite is released longitudinally with a $\Delta V = 0.54$ m/s provided by the separation springs. The Launch Vehicle orientation at the moment of separation is expressed in the Orbital Basic frame by the Euler Angles $\{\psi_0, \theta_0, \phi_0\} = \{-1.3880, -0.0395, -1.1660\}$ rad. At that instant the payload is in an SSO circular orbit at the angular range $\phi_{T0} = 206.3°$. The final attitude for the Main Engine boost is set at $\{\psi_f, \theta_f, \phi_f\} = \{-0.1287, 0.0120, 1.6790\}$ rad when the AVUM+ stage is in $\phi_{Tf} = 27.44°$ and the burn lasts for 125 s. The satellite COG is aligned with the Launch Vehicle's COG with an offset of 1.36 m in the X_B direction. The Launch Vehicle's mass is 950 kg after the payload is separated and the thrust provided during RACS boosts is equal to 350 N. The candidate optimised solution is described by the following parameters:

RACS boost 1						
WT	Roll dt	Yaw dt	WT	RB	Roll ω	Yaw ω
87.6	4.7	32.6	58.2	4.8	1.9	-2.8

RACS boost 2						
WT	Roll dt	Yaw dt	WT	RB	Roll ω	Yaw ω
7.1	6.7	26.4	24.3	2.5	-0.8	1.4

Coasting phase
Coasting time
2526.5

Dummy RACS boost	
WT	Dummy RB
20	10

Main Engine boost						
WT	Roll dt	Yaw dt	WT	ME	Roll ω	Yaw ω
33.5	25.9	27	84.8	125	0.2	2.9

The evaluation vector given by the score function:

Distancing	**RACS angle**	**ME cont.**	**Duration**	**RACS ΔV**
5	4.6970	5	4.9948	4.6958

CCAM total score
4.8630

As observable from the values of the score vector, the analysed CCAM fully respects the distancing and contamination requirements while also having low RACS thrusters boost times. Despite the additional RACS boost block, the proposed CCAM has a total RB time of 17.3 s, much lower than the 69.1 s of the Sentinel Z40RP which features a single RACS boost block but is not optimized for consumption. This CCAM lasts for 3107.6 s, reaching the ideal manoeuvring time of 3107.537 s with an error < 0.1 s.

The Biomass satellite respects the distancing criterion, with a single re-approach at a distance greater than the 10^4 m limit.

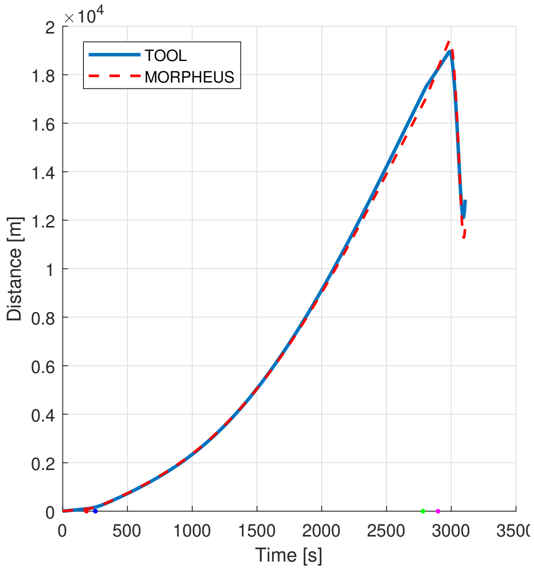


Figure 5.6: VC04 Biomass relative distance

The minimum RACS angle is reached by Thruster 4 with a value of 70.61°, ensuring a solid safety margin with respect to the 26° threshold.

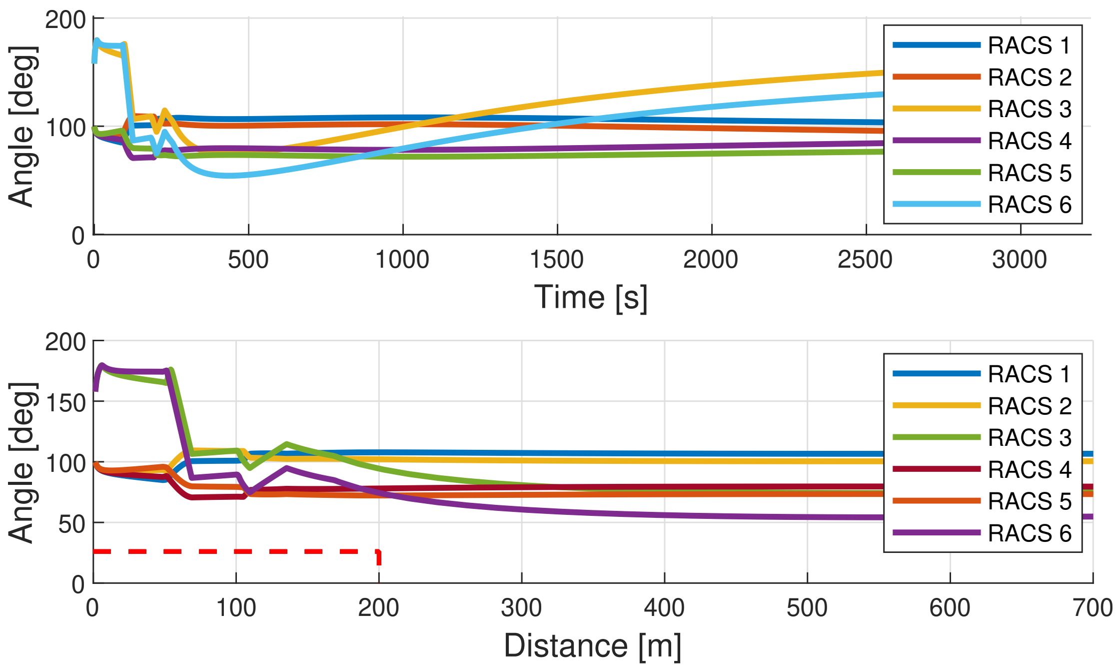


Figure 5.7: VC04 Biomass RACS angles (tool output)

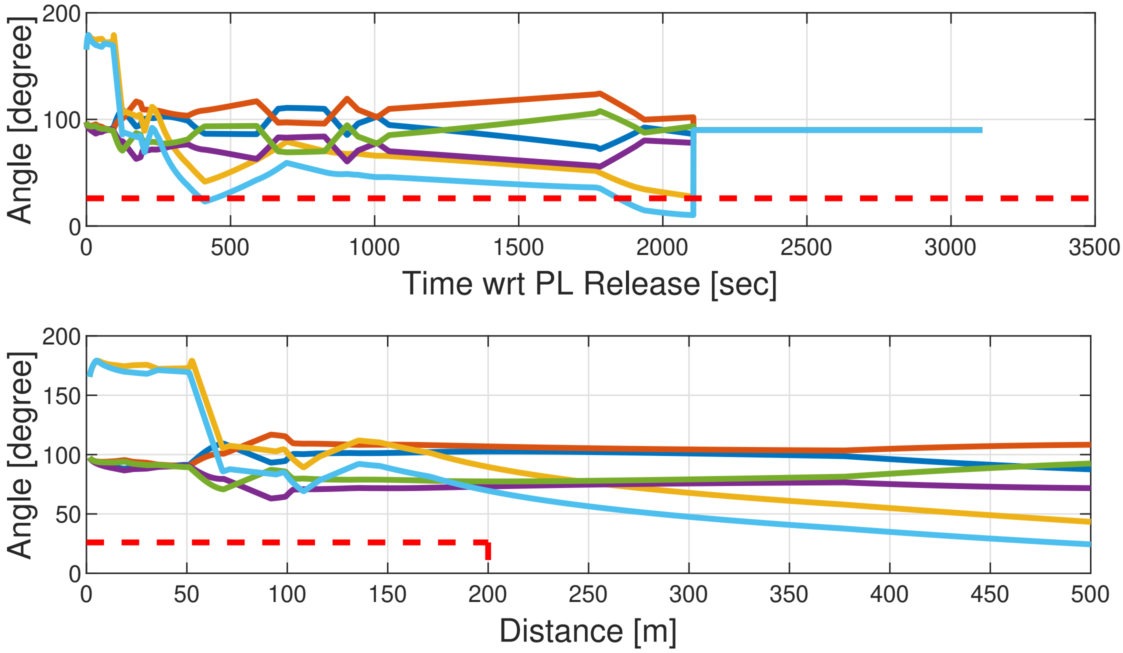


Figure 5.8: VC04 Biomass RACS angles (Morheus output)

The AVUM+ stage is in the ideal condition of a relative distance greater than 10^4 m from the separated payload at the ignition of the Main Engine for the de-orbiting phase. For this reason the deposit level is null in the proposed CCAM.

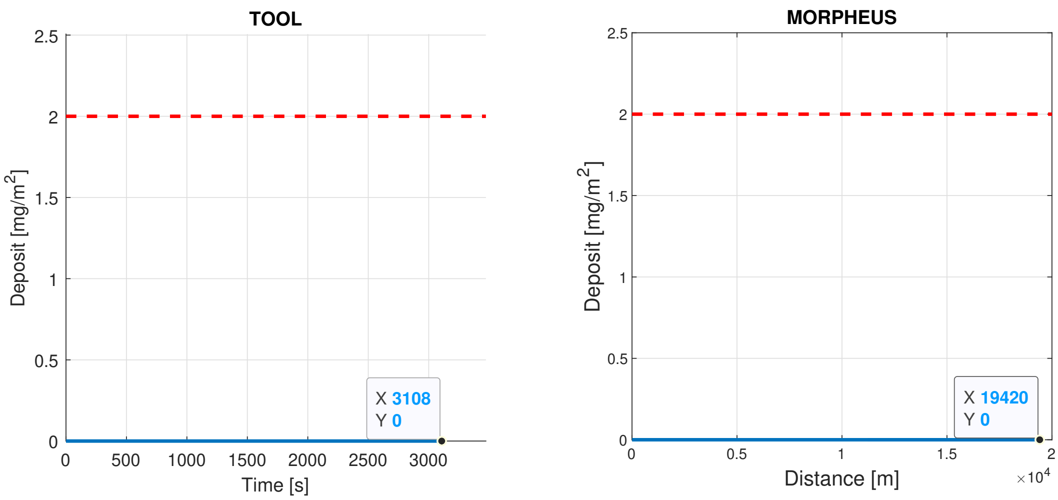


Figure 5.9: VC04 Biomass Main Engine contamination

In conclusion, the trajectory in the LVLH frame of the CCAM of the VC04 Biomass separation is showed in figure 5.10. The manoeuvre reproduced on the Morpheus simulator corresponds almost perfectly in the first phases and starts diverging from the tool result after the second RACS boost and during the coasting phase. The justification to this divergence is the same as in the Sentinel Z40RP case however, in this case, having more precise measures of the CCAM initial conditions, the error becomes noticeable after the critical phases of the evasion manoeuvre have already been executed.

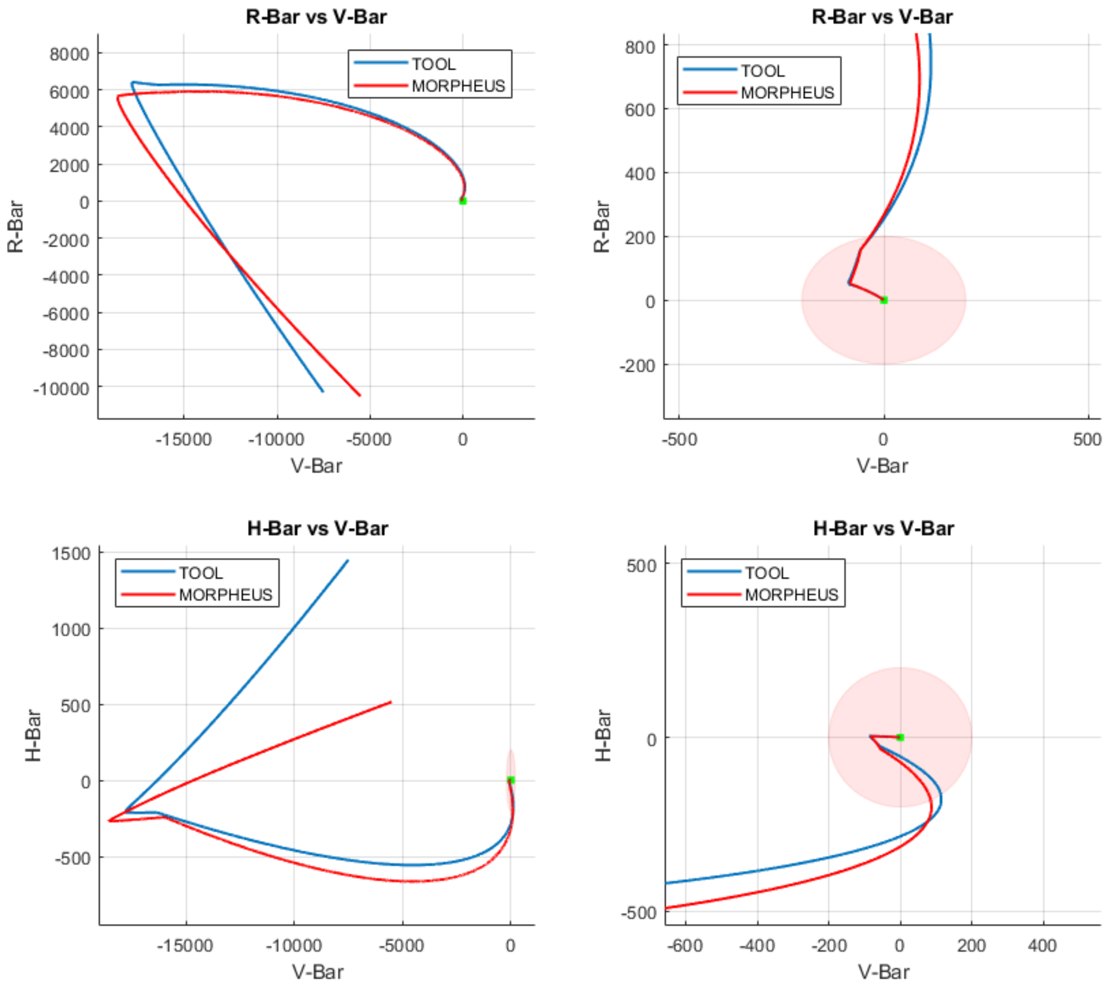


Figure 5.10: VC04 Biomass LVLV trajectory

5.2 Multi-payload results

The reference missions discussed in this work involve the release of a single payload. However, the true utility of the developed tool becomes apparent when applied to more complex missions, where the mission designer cannot easily define a CCAM strategy and verify its compliance with contamination and collision avoidance requirements. To address this, an example of a multi-payload CCAM is provided. The reference mission for this case is the IRIDE mission, which, at the time of writing, is in the preliminary stages of its mission analysis. Avio plans to deploy 5 NIMBUS-

SAR satellites into the same SSO orbit at an altitude of 562.32 km before adjusting the orbit to release its secondary payload. The CCAM following the main payload separation is calculated and optimized using the developed tool.
The position of each satellite is defined together with their release time, direction and ΔV.

Table 5.1: Payload COG position [m] in the Body Frame and separation time [s]

	$\boldsymbol{X_B}$	$\boldsymbol{Y_B}$	$\boldsymbol{Z_B}$	$\boldsymbol{t_0}$
PL1	5.0920	0.0089	0.0167	0
PL2	3.7970	0.6530	0.6950	5
PL3	3.7970	-0.6690	-0.7030	5
PL4	3.7970	-0.7070	0.6570	10
PL5	3.7970	0.6910	-0.6650	10

Table 5.2: Payload direction of separation and spring ΔV [m/s]

	$\boldsymbol{X_B}$	$\boldsymbol{Y_B}$	$\boldsymbol{Z_B}$	ΔV
PL1	1	0	0	0.4053
PL2	0.9869	0.1143	0.1143	0.4077
PL3	0.9869	-0.1143	-0.1143	0.2904
PL4	0.9869	-0.1143	0.1143	0.4146
PL5	0.9869	0.1143	-0.1143	0.297

The payload separation starts at $\phi_{T0} = 64.1°$ angular range. There are no specific attitude requirements for this phase, therefore the initial Euler Angles are chosen by the mission analyst. The AVUM+ stage pointing is fixed towards the $-H$-*bar* direction with $\{\psi_0, \theta_0, \phi_0\} = \{0, \pi/2, -\phi_{T0}\}$ rad. The Main Engine boost begins at $\phi_{Tf} = 85.14°$ and lasts for 7.7 s. The Launch Vehicle mass after all the satellites are released is 832 kg and the provided RACS boost thrust is assumed equal to the standard 350 N value. The CCAM structure chosen for this analysis features 2 RACS boosts followed by a Dummy RACS boost and the Main Engine boost, for a total of 23 manoeuvre parameters. This structure was chosen over a simpler single-RB one to allow the AVUM+ stage to have more manoeuvring freedom in trying to evade the multiple satellites. The candidate solution found by the tool has the following parameter values:

RACS boost 1						
WT	Roll dt	Yaw dt	WT	RB	Roll ω	Yaw ω
28.8	25.9	23.3	36.5	5.6	2.9	-3.7

RACS boost 2						
WT	Roll dt	Yaw dt	WT	RB	Roll ω	Yaw ω
0.3	0	13.8	36	1.5	-5	4.9

Dummy RACS boost	
WT	Dummy RB
30	10

Main Engine boost						
WT	Roll dt	Yaw dt	WT	ME	Roll ω	Yaw ω
14.9	28.9	34.1	45.9	7.7	2.1	4.6

The RACS optimization criterion is suppressed as the preliminary mission analysis focuses on the feasibility of the manoeuvre and the consumption requirements are not yet defined at this time. The evaluation of this CCAM:

Distancing	**RACS angle**	**ME cont.**	**Duration**	**RACS ΔV**
5	4.6560	5	4.9863	suppressed

CCAM total score
4.8610

The total manoeuvring time of 343.3 s reaches the ideal value 343.36 s with an error $<$ 0.1 s. The requirements of distancing and contamination avoidance are respected with good margins for all the separated satellites as visible by the high scores obtained for all criteria.

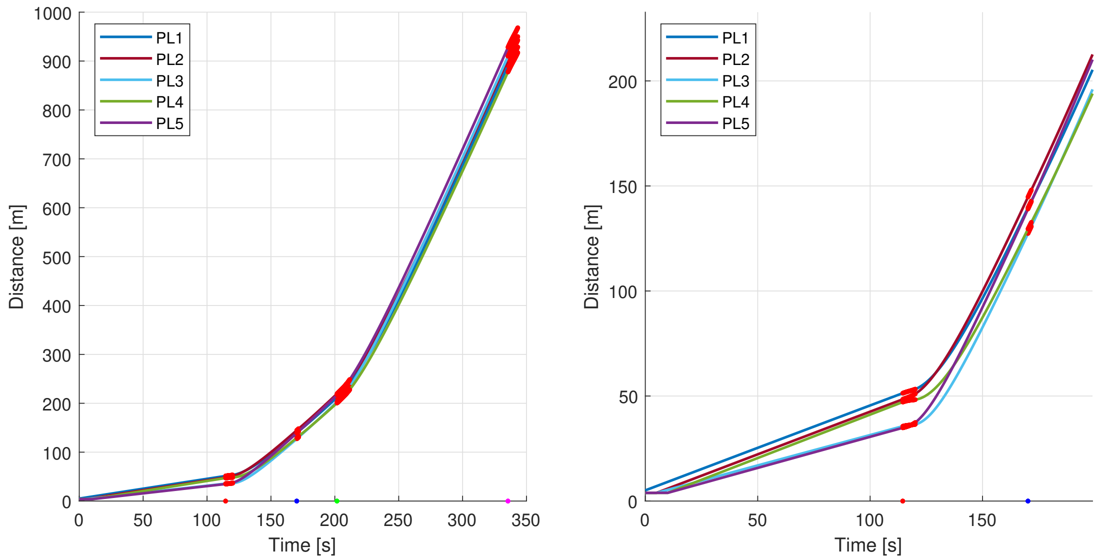


Figure 5.11: IRIDE relative distance

In particular, no payload ever re-approaches the AVUM+ stage after their release as seen in figure 5.11, where the boost phases are highlighted in red.

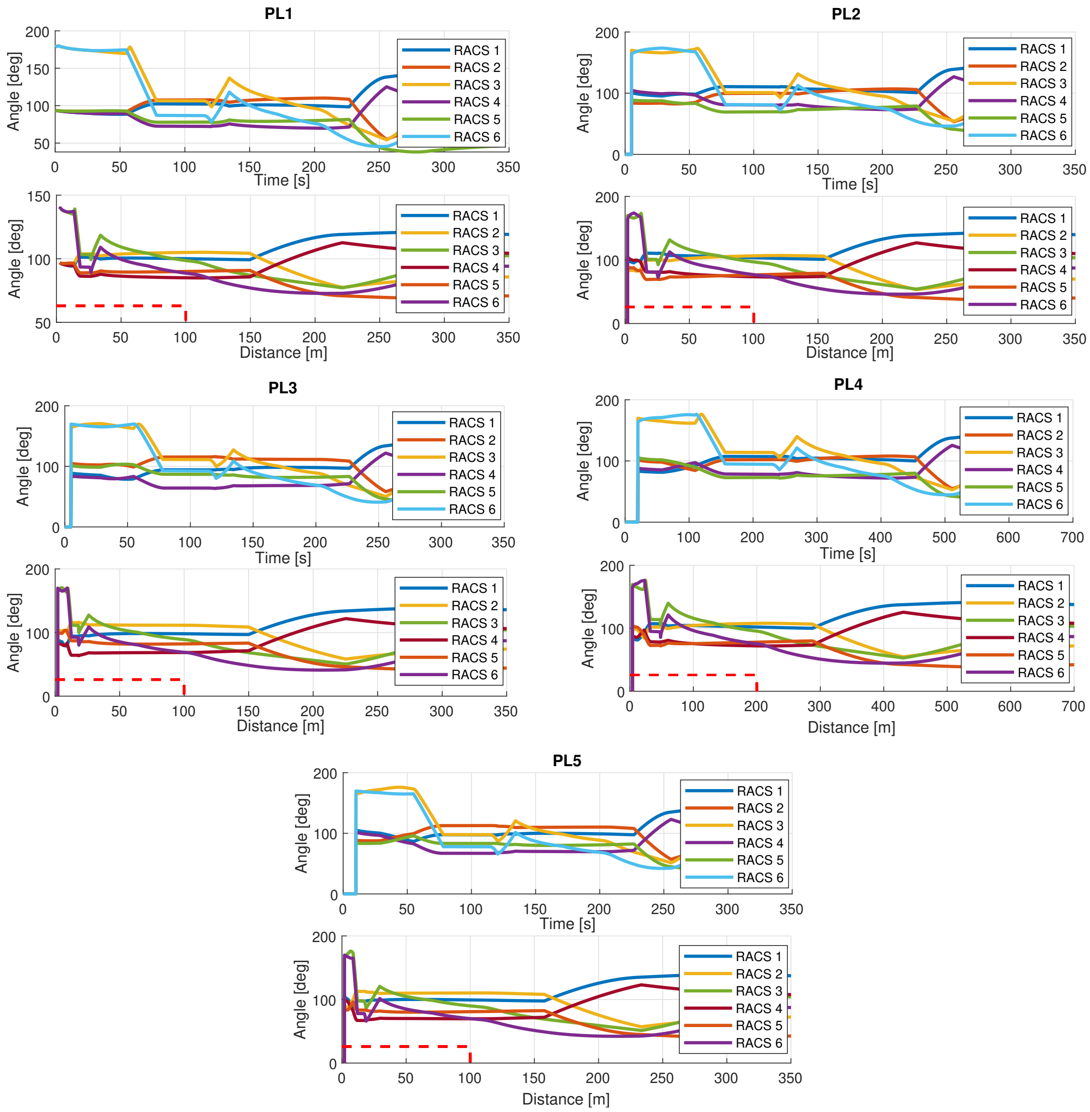


Figure 5.12: IRIDE RACS angles

No satellite falls within the RACS contamination cones and the minimum angles between each payload and the RACS thrusters are high enough to ensure a wide safety margin, as visible in the graphs of figure 5.12. The absolute minimum angle inside the RACS contamination range of 200 m is reached by PL3 with respect to RACS Thruster 4 with a value of 63.54°.

The Main Engine contamination levels are negligible for all payloads, figure 5.13. Despite the Main Engine is ignited at distances lower than 1000 m from the separated satellites, the AVUM+ stage is pointed away from the payloads, causing minimal deposits. Finally in figure 5.14 the CCAM trajectory in the LVLH reference frame is represented.

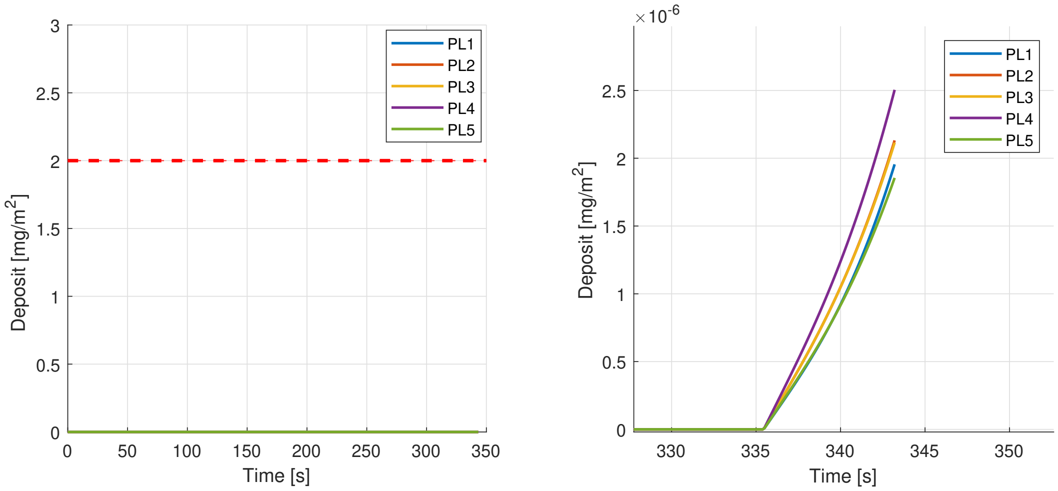


Figure 5.13: IRIDE Main Engine contamination

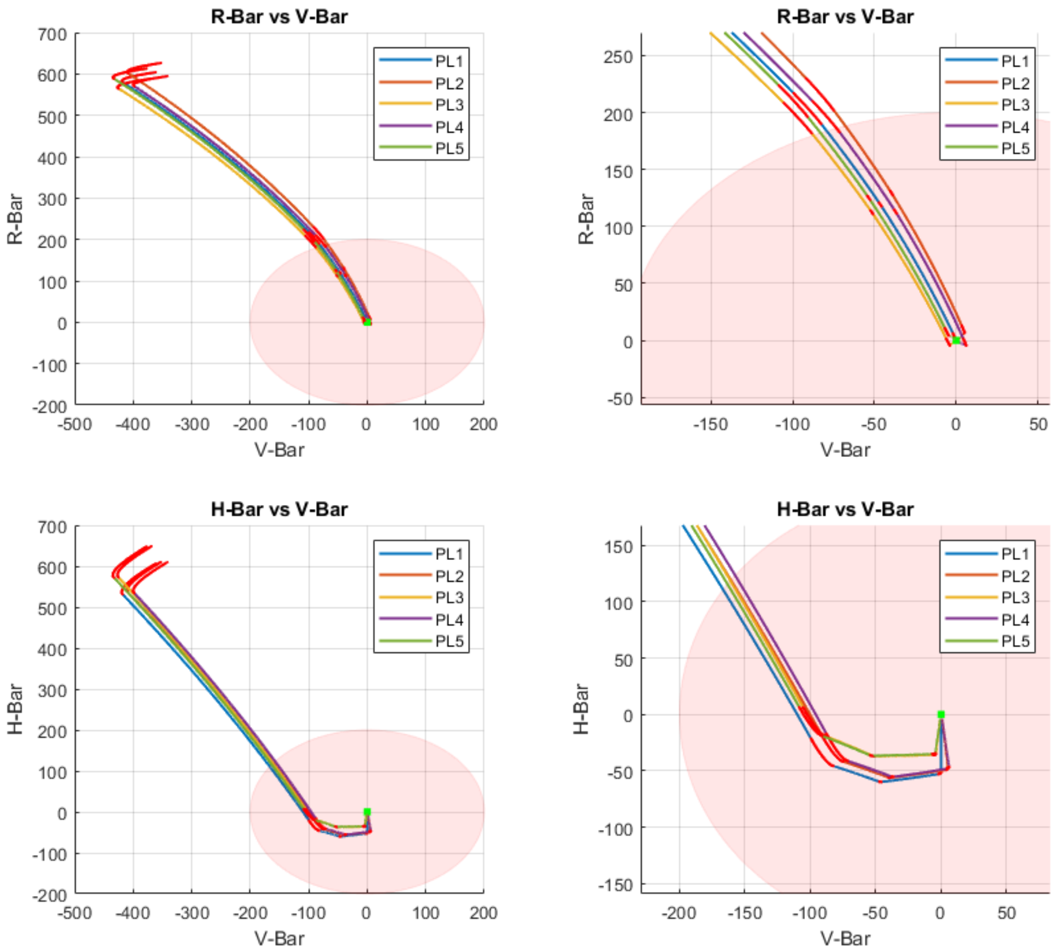


Figure 5.14: IRIDE LVLH trajectory

Chapter 6

Conclusion

The utility of the developed tool is undeniable, as it is currently being utilized by Avio's GNC engineers for feasibility analyses and first-guess estimates within the scope of the IRIDE mission's preliminary mission analysis. This tool not only saves significant time during the design of a demanding and critical mission phase but also empowers mission designers to select from a range of proposed optimal solutions, diverging from the conventional manually generated CCAMs used in past missions. Nonetheless, there is room for improvement. Adding a Graphical User Interface (GUI) would simplify the mission definition phase and make it more intuitive. This improvement could also be applied to the post-processing phase, making it easier to visualize plots and reducing the chance of losing data after long optimization runs by ensuring safer storage of the result. Regarding the more technical aspects, despite the efforts to minimize simulation time by keeping the code simple and efficient, there is undoubtedly room for further optimization. These enhancements could lead to faster execution, either facilitating quicker identification of refined solutions or enabling additional differential evolution iterations to achieve better convergence in the results. Some improvements can also be made to the already implemented features. While the dynamic definition of the CCAM provides users with significant flexibility in simulating a broad range of possible mission configurations, it does have certain limitations. Specifically, separations involving reorientation or boost manoeuvrers between consecutive short-term payload separations cannot be fully defined. Although clever approaches to defining the Launch Vehicle attitude and payload separation data can address the first issue, the tool's environment does not currently support RACS boosts between two releases. Expanding the CCAM definition section to accommodate these uncommon cases could further enhance its overall usefulness. The assumption of a fixed COG is the main source of discrepancies between the trajectories generated by the tool and those reproduced in Morpheus.

While locking the MCI values at the start of the manoeuvre offers a reliable basis for estimating distancing and contamination, this approach can be improved to achieve even greater accuracy, particularly in the final phases of the CCAM. An iterative process to fine-tune the MCI values by comparing the tool's output with that of Morpheus could resolve this issue. After identifying the MCI values that minimize divergence, the tool could then proceed with its usual optimization workflow. To conclude, this work offers a first attempt at defining the CCAM in a general way and provides a solid, functional tool that, despite everything, produces results that surpass the expectations set by the premises of this thesis.

Bibliography

[1] Roland Lagier. *SSMS Vega-C User's Manual*. Issue 1 - Revision 0. Arianespace. Sept. 2020.

[2] P.Bini and G.Martens. *CCAM - Deorbiting and passivation phase analisys report*. AVIO SpA, Vega program.

[3] ELV team. *Payload Contamination Analisys Report*. AVIO SpA, Vega CSI programme.

[4] *Sun-synchronous orbit*. URL: `https://en.wikipedia.org/wiki/Sun-synchronous_orbit`.

[5] G.Martens et al. *VV22 - VC02 - Mission VHR2020-B2 STACK Mission Design and 3DoFs verification*. AVIO SpA, Vega-C Program.

[6] Fehse W. *Automated Rendezvous and Docking of Spacecraft*. Cambridge University Press, 2003.

[7] G. Fabbi. *Vega C launch vehicle functional file*. AVIO SpA, Vecep Program.

[8] M. Cremaschini. *TRANSIENT PHASES THEORY AND APPLICATION*. AVIO SpA.

[9] I.Y. Bar-Itzhack. "New Method for Extracting the Quaternion from a Rotation Matrix". In: *Journal of Guidance, Control, and Dynamics* 23.6 (2000), pp. 1085–1087.

[10] *Quaternions and spatial rotation*. URL: `https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation`.

[11] *Mathematical optimization*. URL: `https://en.wikipedia.org/wiki/Mathematical_optimization`.

[12] *Differential evolution*. URL: `https://en.wikipedia.org/wiki/Differential_evolution`.