POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Autonomous collision avoidance algorithm

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Author: **Jules Grauby**

Student ID: 986021
Advisor: Prof. Pierluigi Di Lizia
Academic Year: 2022-23

Abstract

The number of space objects in orbit keeps increasing and with it the frequency of collision alerts. Regarding this problem, the Flight Dynamics department of the CNES wishes to develop its already operational autonomous orbit control tools to incorporate risk management.

In that regard, ASTERIA, an autonomous Station-Keeping algorithm capable of detecting and avoiding risks, has been developed and tested in space. However, a collision avoidance is a compromise between mitigating the risk and minimizing its impact on the mission, measured by the satellite's deviation from its reference orbit. Moreover, the collision assessment implemented into ASTERIA was only able to handle one risk at a time and did not guarantee a maximum deviation from this reference. To solve this issue, CACAO, a multi-risk collision avoidance maneuver optimization algorithm capable of keeping the satellite inside a Station-Keeping box, has been developed during a R&T.

The first half of this internship focused on the addition of functionalities to this second algorithm to improve its efficiency and robustness. Each modification was tested and validated on real examples of conjunctions, mono-risk and multi-risk, whose data was provided by the CAESAR team of the CNES.

During the second half, this algorithm was then integrated into ASTERIA to replace the previous collision avoidance computation method. CACAO's ability to converge towards a solution in the context of a simulation was validated, though several functionalities can still be improved.

Keywords: Autonomous Orbit Control, Station-Keeping, Collision avoidance strategy.

Sommario

Il numero di oggetti spaziali in orbita é in continuo aumento e con esso la frequenza degli allarmi di collisione. Per rispondere a questo problema, il dipartimento di Dinamica del Volo del CNES desidera sviluppare i suoi strumenti di controllo autonomo dell'orbita già operativi per incorporare la gestione del rischio.

A questo proposito, è stato sviluppato e testato nello spazio ASTERIA, un algoritmo autonomo di station-keeping in grado di rilevare ed evitare i rischi di collisione. Tuttavia, la prevenzione delle collisioni è un compromesso tra la mitigazione del rischio e la minimizzazione del suo impatto sulla missione, misurato dalla deviazione del satellite rispetto alla sua orbita di riferimento. Inoltre, la valutazione delle collisioni implementata in ASTERIA era in grado di gestire solo un rischio alla volta e non garantiva una deviazione massima rispetto al riferimento. Per risolvere questo problema, nel corso di questo studio è stato sviluppato CACAO, un algoritmo di ottimizzazione delle manovre anti-collisione multirischio in grado di mantenere il satellite all'interno di un box di station-keeping.

La prima metà di questo studio si è concentrata sull'aggiunta di funzionalità a questo secondo algoritmo per migliorarne l'efficienza e la robustezza. Ogni modifica è stata testata e validata su esempi reali di congiunzioni, mono-rischio e multi-rischio, i cui dati sono stati forniti dal team CAESAR del CNES.

Nella seconda fase, questo algoritmo è stato integrato in ASTERIA per sostituire il precedente metodo di calcolo di evitamento delle collisioni. La capacità di CACAO di convergere verso una soluzione nel contesto di una simulazione è stata verificata, anche se diverse funzionalità possono ancora essere migliorate.

Parole chiave: Controllo autonomo dell'orbita, station-keeping, strategia di evitamento delle collisioni.

Contents

1 | State of the art - AOC and avoidance

1.1. Autonomous Orbit Control applied to SK

Autonomous Orbit Control or AOC is a satellite's ability to identify, plan and execute its corrective maneuvers without the Ground Control's intervention. It consists of on-board algorithms that aim to diminish the operational workload for ground teams.

This technology is only operational for Station-Keeping (SK), the maneuvers that prevent a satellite to stray too far from its reference orbit.

1.1.1. Working principle

Below is reported the working principle for an AOC algorithm applied to Station-Keeping :

- The satellite determines its own state vector through on board GNSS measurements and obtains its own orbital parameters. This determination usually takes place one time per orbit, at the ascending node.
- Depending of the value of these parameters and their evolution with respect to previous measurements, the satellite propagates them over a certain time horizon to detect whether one of these parameters exceeds a threshold. This corresponds to a situation where the satellite strays too far from its reference orbit.
- The satellite then plans one or several maneuvers during time slots allocated to Station-Keeping, then executes them.

All these steps do not require any space-to-ground link. The satellite determines itself when and how it must correct its trajectory and only warns the ground of its maneuver after it took place. As a consequence, the maximum tolerated deviation between the satellite's real state and its reference orbit must be much smaller than for non-autonomous Station Keeping. This is for reasons of pointing accuracy when communicating with a ground station. Since the satellite's SK maneuvers are not known and cannot be taken into account to predict the satellite's position in the sky, it must be kept very close to its reference orbit.

1.1.2. Maneuver slots

It has already been mentioned that the satellite only plans its corrective maneuvers within allocated slots. The notion of slots is indeed key, as the AOC guarantees total decorrelation between the mission and the SK.

This is because, from the satellite's mission point of view, the activity planning takes place without taking into account the needs of the SK. It must thus ensure that the satellite remains close to its reference orbit with a certain degree of accuracy, allowing the mission to be planned for the long term. A list of slots is therefore provided to the AOC at an earlier date, corresponding to the ones in which the satellite can perform corrective maneuvers without interrupting the mission.

Apart from the mission, these slots allocated to the AOC are also reduced due to other constraints linked to attitude control, propulsion and sensor glare.

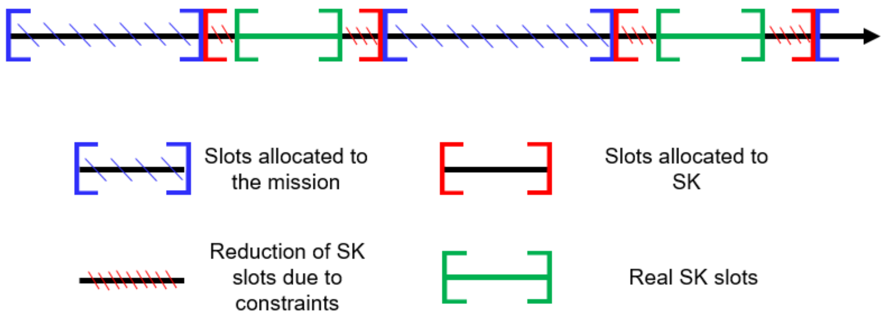

Slots allocated to the mission
Slots allocated to SK
Reduction of SK slots due to constraints
Real SK slots

Figure 1.1: Time slots allocated to the mission and to the SK

In short, the SK maneuvers planned by the satellite are not a priority in relation to the mission, and may take place during non-optimal slots.

1.1.3. Advantages

The advantages of such technology are numerous. The most obvious is the reduction in workload, and therefore operational costs. SK is no longer a problem managed by orbital maneuvers engineers.

Autonomy also makes the trajectory control more responsive to disturbances such as atmospheric friction. The satellite is in the best position to know its own state at a given time t, and can therefore adjust its maneuvers right up to the last moment. This eliminates the need to rely on station passes to correct a planned maneuver.

From the mission's point of view, as already mentioned, activity planning is simplified, as it is no longer based on the satellite's actual trajectory, but on its theoretical reference orbit, with the AOC guaranteeing concordance between the two. This also eliminates the need for any orbit determination calculations for pointing purposes during station passes.

AOC applied to SK is a technology that has already been operationally validated on board CNES's CSO satellites.

1.2. Collision avoidance

In recent years, the number of collision alerts received by satellite operators has steadily increased, driven by two phenomena :

- An increase in the number of objects in orbit (deployment of mega-constellations, end-of-life of satellites not complying with international recommendations, ground-to-space missiles aimed at destroying satellites).
- Improved tracking resources, thanks in particular to the European Union Space Surveillance and Tracking (EUSST) program and the US Army's Space Fence, which is gradually lowering its object detection threshold from 10 to 3 cm. Hundreds of debris will thus be added to the current catalog.

Managing the risk of collision, particularly in low-Earth orbit, is now a major issue for international space agencies and the private sector alike. The aim is to avoid at all costs what is known as the Kessler syndrome [8], i.e. a chain reaction scenario in which the debris population increases uncontrollably, making space activities impossible. Indeed, any collision generates a very large amount of debris. For example, the destruction of Fengyun-1 C by a Chinese anti-satellite missile in 2007 created over 2,700 pieces of catalogued debris [7]. Also, the collision between Iridium 33 and Cosmos 2251 in 2009 created over 1,600 pieces of debris [6]. On June 12 2023, the number of catalogued objects in orbit was 36,500 [5].

1.2.1. Geometry of a collision

By its very nature, a collision involves two objects :

- The primary, which corresponds to an operated satellite. Its role is to maneuver to avoid collision.
- The secondary, which may be either space debris or another satellite. In the latter case, agreement must be reached between the teams operating the two objects to avoid inconsistent maneuvers.

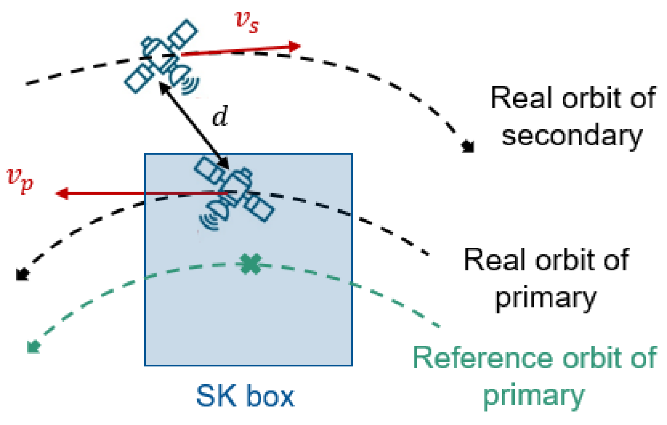


Figure 1.2: Geometry of a collision

The state of each of these objects is uncertain and thus modeled by a covariance assumed to be Gaussian, thus elliptical in shape. From a vocabulary point of view, the instant when the mean states of the two objects are closest is called the Time of Closest Approach or TCA.

The SK box

An important concept for the CACAO algorithm is the SK box. This is a rectangular parallelepiped defined in the local orbital frame QSW and centered on the satellite's reference orbit. Forcing the primary to remain in this box before, during and after the TCA minimizes the impact of avoidance maneuvers on the mission.

Short-term encounters

In the collision avoidance literature, a distinction is made between so-called "short-term" and "long-term" encounters, which are differentiated by the value of the relative velocity between the primary and secondary at the TCA. In terms of order of magnitude, a short-term encounter is generally characterized by a relative velocity of the order of km/s, while a long-term encounter is of the order of m/s [4].

However, the real difference between the two situations is that a short-term encounter allows for assumptions that simplify both the geometry of the problem and the estimation of collision risk. The CACAO algorithm applies these assumptions and therefore treats any encounter as fast, even when this is quantitatively false. This study will therefore only deal with such cases.

Hypotheses

Hypothesis 1 : Uniform rectilinear motion.
Due to the high relative velocity between the two objects, the relative motion is considered to be uniformly rectilinear over a short time interval centered on the TCA. Positional uncertainties are

considered constant, while velocity uncertainties are neglected.

Hypothesis 2 : Sphericity of the objects.
To simplify the model, the primary and secondary are considered spherical objects with radii R_p and R_s respectively. These radii are defined by the smallest spherical volume that encloses the object in question. Their sum defines the Hard Body Radius (HBR) such that HBR $= R_p + R_s$. By definition, a collision corresponds to a distance between the primary and secondary below this value.

Hypothesis 3 : Gaussian distribution.
As already mentioned, the uncertainties around the two objects are characterized by probability densities, ρ_p and ρ_s, assumed to be Gaussian throughout the encounter. The trajectory of each of the objects is thus defined by a mean state, $\boldsymbol{x_p}(t_{TCA})$ and $\boldsymbol{x_s}(t_{TCA})$, and a covariance matrix of dimension 6×6, C_p and C_s, such that :

$$C_i = \begin{bmatrix} C_i^{rr} & C_i^{rv} \\ C_i^{vr} & C_i^{vv} \end{bmatrix} \quad \forall i \in \{p, s\}$$

Hypothesis 4 : Independence.
The probability densities of the two objects are assumed to be independent of each other. Using these assumptions, the geometry is characterized by a sphere of radius HBR carried by the primary in uniform rectilinear translation in a local orbital frame centered on the secondary. A combined covariance, containing the uncertainties associated with both objects, is also centered on the secondary. By independence, this covariance is :

$$\Sigma = C_p + C_s \tag{1.1}$$

Collision plane

According to the above assumptions, the 3D problem of the encounter can be reduced to a 2D problem in a well-chosen space: the collision plane or BPlane. It's important to remember that this plane can only be defined in a very short time interval around the TCA, during which relative motion is assumed to be uniformly rectilinear.

This plane can be defined in several ways. We define here a basis $\mathcal{B}$ consisting of the vectors $(\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_2})$, centered on the secondary such that :

- The vector $\boldsymbol{e_2}$ is oriented along the relative velocity of the two objects. The collision plane is the plane orthogonal to this vector.
- The vector $\boldsymbol{e_1}$ belongs to the collision plane. It is orthogonal to the plane defined by the velocity of the primary $\boldsymbol{v_p}(t_{TCA})$ and the secondary $\boldsymbol{v_s}(t_{TCA})$ in an inertial reference frame.
- The vector $\boldsymbol{e_3}$ completes the direct trihedron.

i.e. :

$$\boldsymbol{e_1} = \frac{\boldsymbol{v_s}(t_{TCA}) \times \boldsymbol{v_p}(t_{TCA})}{||\boldsymbol{v_s}(t_{TCA}) \times \boldsymbol{v_p}(t_{TCA})||} \qquad \boldsymbol{e_2} = \frac{\boldsymbol{v}_r(t_{TCA})}{||\boldsymbol{v}_r(t_{TCA})||} \qquad \boldsymbol{e_3} = \boldsymbol{e_1} \times \boldsymbol{e_2} \tag{1.2}$$

The collision plane is therefore the plane defined by the vectors $\boldsymbol{e_1}$ and $\boldsymbol{e_3}$. The intersection of the primary with this plane corresponds to the TCA, the instant when the relative distance between the two objects is smallest.

1.2.2. Risk of collision calculation

Once the geometry of the encounter has been defined, it's important to establish a parameter that characterizes the risk of collision. This is generally either the probability of collision, or the distance separating the mean states of the two objects.

Probability of collision

The parameter most commonly used to assess the risk of collision is the PoC (*Probability of Collision*). This is the value used operationally by the ground segment to decide whether or not to maneuver. Moreover, a successful avoidance maneuver is characterized by a reduction in this value below a certain threshold. In terms of order of magnitude, avoidance takes place when this probability exceeds approximately 10^{-3}, and the aim of this avoidance is generally to bring this value down below 10^{-5}.

Using the geometry defined for the problem, this probability is the integral of the probability density over the volume V traversed by the sphere of radius HBR during the encounter, i.e. :

$$P_c = \frac{1}{\sqrt{(2\pi)^3|\Sigma|}} \iiint_V \exp\left(-\frac{1}{2}\boldsymbol{x_r}^T\Sigma^{-1}\boldsymbol{x_r}\right) dxdydz \tag{1.3}$$

With $\boldsymbol{x_r}$ the relative state vector such that $\boldsymbol{x_r} = \boldsymbol{x_p} - \boldsymbol{x_s}$. This formula is extracted from the technical report [11].

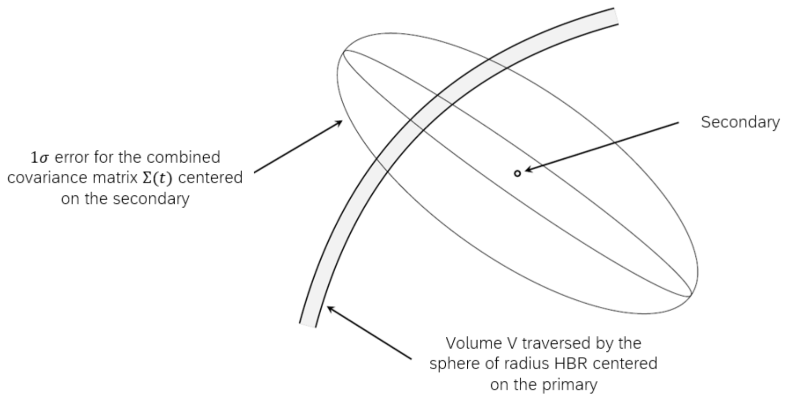

Secondary
1σ error for the combined covariance matrix $\Sigma(t)$ centered on the secondary
Volume V traversed by the sphere of radius HBR centered on the primary

Figure 1.3: PoC calculation

Increased probability of collision

In an operational context of collision risk assessment, it can happen that the covariances of objects, and particularly that of the secondary, are poorly defined. This can have a major impact on risk assessment, and hence on the decision to maneuver or not. It is therefore necessary to take these uncertainties on the covariances into account, and this is done by increasing the PoC. Instead of defining the combined covariance as $\Sigma = C_p + C_s$, we introduce homothetic factors k_p and k_s such that the previous equation becomes :

$$\Sigma = k_p C_p + k_s C_s \tag{1.4}$$

The value of these factors are then allowed to vary, generally between 0.25 and 4, to find the pair $[k_p, k_s]$ that maximizes the PoC. This new probability is called the COPoC for *CNES Operational Probability of Collision*, and is the value that serves as the reference for risk management.

Euclidean distance

Although PoC is the parameter most frequently used operationally, the CACAO algorithm uses distances to estimate the risk of collision. This is known as the miss distance, and is defined as the distance between the mean states of the two objects $\boldsymbol{r_p}(t_{TCA})$ and $\boldsymbol{r_s}(t_{TCA})$ at the TCA.

Indeed, the relative position vector between these two points has been defined as $\boldsymbol{r}_r(t_{TCA}) = \boldsymbol{r_p}(t_{TCA}) - \boldsymbol{r_s}(t_{TCA})$. The Euclidean distance is therefore :

$$d_{\text{Eucl}} = \sqrt{\boldsymbol{r}_r(t_{TCA})^T \boldsymbol{r}_r(t_{TCA})} \tag{1.5}$$

Assessing the risk in this way is interesting because it's a physical value that's easy to interpret. It can be compared with other distances, such as the dimensions of the SK box. For example, it is impossible to demand that the new miss distance after maneuvering be greater than the length between the center of the box and one of its vertices. A maximum avoidance criterion can then be associated with SK constraints.

Mahalanobis distance

However, reasoning in terms of Euclidean distance ignores the concept of uncertainty in object trajectories. Another quantity that the CACAO algorithm can manipulate is the Mahalanobis distance. This is a Euclidean distance weighted by the uncertainty associated with the two objects. Let Σ^{rr} be the combined positional covariance matrix defined by the equation (1.4), then this distance is :

$$d_{\text{Mah}} = \sqrt{\boldsymbol{r}_r(t_{TCA})^T {\Sigma^{rr}}^{-1} \boldsymbol{r}_r(t_{TCA})} \tag{1.6}$$

For example, a Mahalanobis distance of 3 means that the primary lies on the surface of the 3-σ ellipsoid defined by the combined covariance. Thus, a high degree of uncertainty about the position of the objects will decrease the Mahalanobis distance and thus increase our risk assessment.

This value will also favor certain directions during avoidance. For orbiting objects, uncertainties are greater in the tangential direction, aligned with velocity, than in radial or out-of-plane directions. So, for an equal Mahalanobis distance, avoidance in a tangential direction will correspond to a greater Euclidean distance than in other directions, and will therefore require a larger maneuver.

1.2.3. Operational risk management

Currently, even for satellites with an AOC applied to SK, any risk of collision is managed on the ground. This is organized in several stages :

- The cataloguing of orbiting objects and the detection of collision risks is largely carried out by the US Space Surveillance Network (SSN). ESA also carries out its own sky observation with numerous telescopes and radars installed in various countries.
  In particular, CNES owns the Télescopes à Action Rapide pour les Objets Transitoires (TAROT) and the Grand Réseau Adapté à la VEille Spatiale (GRAVES). All these networks work together to detect risks, with the aim of avoiding oversights.

- From CNES's point of view, these risks are analyzed by the Space Surveillance department, and particularly by the CAESAR team. This team then generates a Conjunction Data Message, an XML file containing collision-related information such as the primary and secondary state vectors at TCA, their covariances and the PoC.
  If this probability exceeds a certain threshold, CAESAR will warn the control center, i.e. the team operating the satellite, of the risk and transmit the associated CDM.

- It's up to the mission leader of this team to make the decision to maneuver. This decision is based on the PoC, but also on its evolution. It is CAESAR's role to update the covariances and therefore the probability on a daily basis by transmitting new CDMs until the TCA. An increase in the risk of collision despite a decrease in uncertainties is information that encourages to perform an avoidance.
  The mission is a factor taken into account when choosing the date for the maneuver. Although the satellite's survival remains a priority, avoidance risks damaging or even interrupting the mission for several orbits by disrupting its trajectory and attitude at key moments. For this reason, the maneuver is carried out as late as possible.

- If the decision is taken, the orbitography team associated with the satellite must calculate the maneuver to be performed, which must then be transmitted by telecommand during a station pass.

The problems of such collision risk management are, firstly, the heavy operational workload involved and, secondly, the dependence on station passes. The latter means that the avoidance maneuver is usually transmitted long before the collision, when the uncertainties surrounding the two objects are still high. All this can be resolved by autonomous collision risk management. The on-board-ground link would then be limited to the transmission of the CDMs to the satellite, which could, thanks to its precise knowledge of its own state, adjust the maneuver right up to the last moment, or even cancel it if it is no longer necessary.

1.3. Autonomous Orbit Control with avoidance

Autonomous management of collision avoidance therefore offers a number of advantages in terms of reduced operational load and improved satellite responsiveness to disturbances and changing risks. CNES, through its Autonomy research team within the Flight Dynamics department, is working on this subject with the aim of implementing this function on board the agency's future satellites.

1.3.1. ASTERIA

ASTERIA, or Autonomous Station-keeping Technology with Embedded collision RIsk Avoidance System, is an AOC algorithm developed by the CNES Autonomy research team to perform SK and collision avoidance autonomously. The general operation of this algorithm is described in detail in section 3.1.

Figure 1.4: Logos of ASTERIA and OPS-SAT

The ASTERIA algorithm had already been validated in a real-life environment using OPS-SAT, an ESA cubesat carrying several experimental flight software payload, the main one being ASTERIA, with the aim of running them in orbit. However, CNES had no knowledge of the real collision risks that the satellite might encounter, so artificially constructed CDMs were transmitted to the cubesat, which had no associated propulsion system. This meant that the algorithm studied collision risks and planned avoidance maneuvers, but OPS-SAT didn't actually carry them out. In spite of this, the experiment was conclusive about the algorithm's operability, particularly with regard to the throughput for uploading CDMs and the estimation of the computational load.

1.3.2. CACAO

Multi-risk avoidance

As mentioned earlier, the number of collision alerts that satellite operators have to manage is likely to rise sharply in the future. Not only will the number of avoidance situations increase, but more and more of these situations will be of a multi-risk nature.
Multi-risk avoidance can be defined as a succession of several collision risks within a reduced time interval, generally a few orbits of the primary satellite. There are two distinct cases :

- The first case involves one primary and several secondaries. This situation is likely to become increasingly frequent in LEO, where the concentration of debris and satellites is highest. The primary then faces a succession of conjunctions with several objects, each associated with a high PoC.

- The second case involves a primary and a single secondary. What differentiates this case from the usual mono-risk avoidance is that the trajectories of the two objects cross frequently, rather than in a one-off fashion. We can, for example, imagine two polar orbits that are identical except for their RAAN. With the right phasing, the two objects could collide twice per orbit. The most frequent situation, however, is when the two objects have very similar orbits. We then observe a succession of conjunctions at equal time intervals. The examples dealt with during the internship are of this type.

R&T of the EUSST

Faced with the expected increase in multi-risk situations, the EUSST commissioned a R&T involving CNES as technical expert, Thalès Services Numériques and the LAAS. The aim of this project was to design an algorithm capable of autonomously calculating avoidance maneuvers for mono and multi-risk situations.

LAAS carried out an extensive literature review on the subject of collision avoidance, noting the lack of sources and studies done on the issue of multiple risks [1]. Several theoretical algorithms were also studied and compared, with the aim of optimizing the performance of avoidance maneuvers while respecting SK and minimum distance constraints. Three algorithms were suggested by this study :

- The Branch & Bound (B&B).
- The Semi-Definite Random (SDR).
- The Lasserre Hierarchy.

Only the first two methods were implemented and tested by TSN, and the team in charge of this work came to the conclusion that the first option, the B&B algorithm, was the most efficient in terms of convergence and computational time [2].

However, the code delivered to CNES by TSN contained numerous problems and code errors, such as rotations from one frame of reference to another done in reverse, and incorrect implementation of the optimization algorithm. It was the job of a previous intern to correct these errors and test the algorithm on CDMs corresponding to real avoidance cases supplied by the CNES CAESAR team [9].

My work during the internship was therefore to familiarize myself with the algorithm and add features to make it more robust, and then to integrate it into ASTERIA to replace the existing mono-risk collision avoidance algorithm. I took the liberty of naming the algorithm to clarify the report. It's called CACAO for Code for Autonomous Collision Avoidance Optimization.

Figure 1.5: Logo of CACAO

2 | CACAO presentation and modifications

This chapter is devoted to presenting the CACAO algorithm, highlighting the modifications made during the internship. The first part (2.1) describes the overall principle and architecture of the algorithm, focusing on inputs and outputs and the construction of the optimization problem. The optimization algorithm itself is the subject of the following section (2.2). Finally, the last section (2.3) describes the main modifications and corrections made during the first half of the internship.

2.1. Architecture

2.1.1. Inputs and outputs

The algorithm's inputs can be classified into two categories :

1. Data specific to the objects involved in the collision. These are contained in CDMs and include :

   - The date of the TCA : t_{TCA}.
   - The state of the primary at TCA in the ITRF non-inertial reference frame : $\boldsymbol{x_s}(t_{TCA})$.
   - The state of the secondary at TCA in the ITRF reference frame : $\boldsymbol{x_s}(t_{TCA})$.
   - The relative state between the two objects centered on the primary in the local QSW reference frame at TCA : $\boldsymbol{x_r}(t_{TCA})$.
   - The primary and secondary covariance matrices at TCA in the ITRF frame : $C_p(t_{TCA})$ and $C_s(t_{TCA})$.

   Some of the information characterizing the risk can also be found in the CDM :

   - The PoC and the method used to calculate it.
   - The miss distance.
   - The relative speed between the two objects at the TCA.
   - The radii characterizing the size of each object : R_p and R_s.

2. Other data required to solve the avoidance problem. To function on its own, CACAO reads this data from an XML file known as the "Configuration file", which indicates, among other things:

   - The satellite's characteristics, such as its weight m or the specific impulse I_{sp} of its propulsion system.
   - Allowed maneuver slots. It's important to note that the algorithm assumes impulsive

maneuvers, which means that a slot is not a time interval but a date, specified in the number of orbits before the TCA.

- The dimensions of the SK box in the local reference frame QSW.
- The avoidance criterion, i.e. the minimum Euclidean distance, $\tilde{d}_{\text{Eucl}}^{\text{min}}$, or Mahalanobis distance, $\tilde{d}_{\text{Mah}}^{\text{min}}$, which must separate the two objects at TCA after maneuvering.
- Minimum ellipsoid volume. This is a criterion linked to the B&B (see section 2.2) which impacts the accuracy and computation time of the optimization.

The output of the algorithm is an avoidance strategy, i.e. one or more impulse maneuvers to be performed with their date, magnitude and direction.

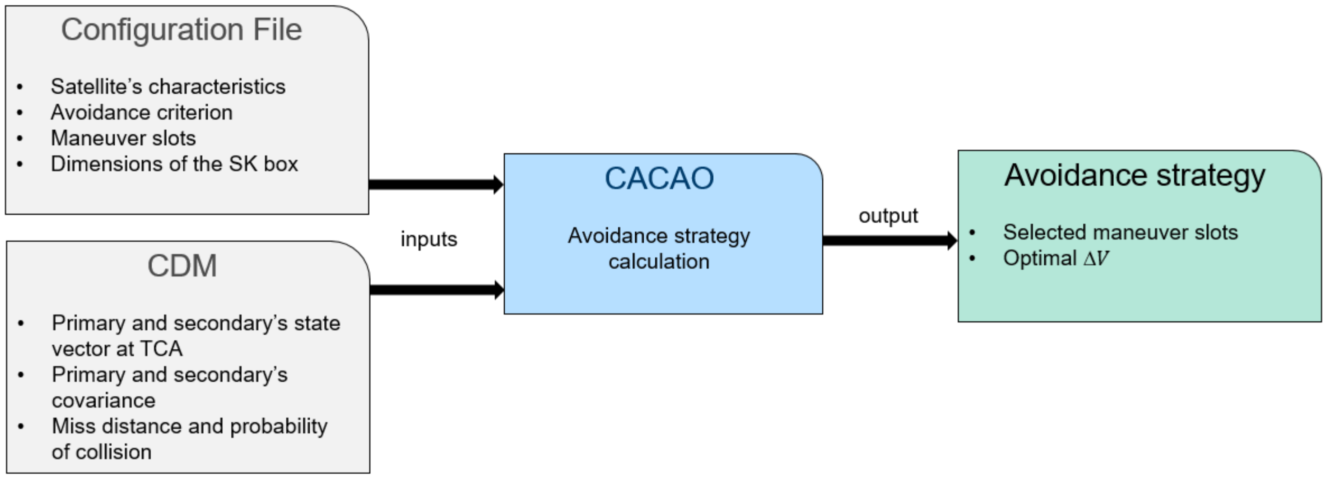

Configuration File
Satellite's characteristics
Avoidance criterion
Maneuver slots
Dimensions of the SK box
CDM
Primary and secondary's state vector at TCA
Primary and secondary's covariance
Miss distance and probability of collision
inputs
CACAO
Avoidance strategy calculation
output
Avoidance strategy
Selected maneuver slots
Optimal ΔV

Figure 2.1: Inputs and output of CACAO

2.1.2. Construction of the optimization problem

To go into more detail about the internal architecture of the code, the algorithm is divided into two parts: a Java overlay and an optimization algorithm coded in Python. The role of the Java part is first to construct the optimization problem to be solved by the B&B. To do this, input data are read and retrieved.

Linearization of the dynamics

The algorithm begins by defining an initial state for the primary $\boldsymbol{x_p}(t_0)$. In the absence of any data other than the CDMs, this state is obtained by back-propagating $\boldsymbol{x_p}(t_{TCA})$, the information on the primary contained in the CDMs, by 5 orbits after performing a reference frame change to the GCRF inertial reference frame. The duration of this back-propagation is arbitrary, and serves only to validate how CACAO works on its own.

In a multi-risk case, CACAO has to manage several CDMs and therefore several TCAs. To perform the above operation, the algorithm was modified during the internship to define a "main" TCA. This TCA simply corresponds to the encounter with the lowest distance miss indicated in its CDM. The initial state is therefore defined 5 orbits before this date. For the sake of generality, we now consider a multi-risk avoidance situation with N encounters at dates t^j_{TCA}, $j \in [1, N]$.

It's important to note that all of the numerical propagations and reference frame changes performed in this overlay use Patrius, the CNES Java library for orbital mechanics.

Once the initial state has been defined, the Java algorithm constructs the matrices that will be passed on to the B&B for the optimization. This optimization is based on a linearization of the Keplerian dynamics [3], so that the state of the primary at the j^{th} TCA is linked to the initial state by a transition matrix Φ :

$$\boldsymbol{x_p}(t^j_{TCA}) = \Phi(t^j_{TCA}, t_0)\boldsymbol{x_p}(t_0) \tag{2.1}$$

With Φ a 6×6 matrix such that :

$$\Phi(t^j_{TCA}, t_0) = \begin{bmatrix} \Phi_{rr}(t^j_{TCA}, t_0) & \Phi_{rv}(t^j_{TCA}, t_0) \\ \Phi_{vr}(t^j_{TCA}, t_0) & \Phi_{vv}(t^j_{TCA}, t_0) \end{bmatrix}$$

We now consider n impulsive maneuvers with respective unit vectors $\boldsymbol{\beta_i}$ in the GCRF frame and magnitudes Δv_i, each applied at date t_i belonging to the time interval between t_0 and t^j_{TCA}. Denoting $\boldsymbol{\tilde{r}_p}(t_{TCA})$ the position vector of the primary after these maneuvers, the linearization of the dynamics presented in equation (2.1) leads to the formula :

$$\boldsymbol{\tilde{r}_p}(t^j_{TCA}) = \boldsymbol{r_p}(t^j_{TCA}) + \sum_{i=1}^{n} \Delta v_i \Phi_{rv}(t_{i+1}, t_i)\boldsymbol{\beta_i} = \boldsymbol{r_p}(t^j_{TCA}) + \boldsymbol{\Phi^j}\boldsymbol{\Delta V} \tag{2.2}$$

Using the notation $t^j_{TCA} = t_{n+1}$ to simplify the previous formula. The matrix $\boldsymbol{\Phi^j}$ of dimensions $3 \times n$ is obtained by concatenating the following vectors :

$$\boldsymbol{\Phi^j} = \begin{bmatrix} \Phi_{rv}(t_2, t_1)\boldsymbol{\beta_1} & \ldots & \Phi_{rv}(t_{n+1}, t_n)\boldsymbol{\beta_n} \end{bmatrix}$$

And the vector $\boldsymbol{\Delta V}$ is defined by $\boldsymbol{\Delta V} = \begin{pmatrix} \Delta v_1 & \Delta v_2 & \ldots & \Delta v_n \end{pmatrix}^T$.

Cost function

The optimization algorithm, coded in Python, aims to minimize the total ΔV of the maneuvers while respecting certain constraints linked both to the minimum miss distance desired after maneuvers at TCA for each risk, and to the SK box. Since the direction of each maneuver is decided upstream of the B&B, only the magnitudes are optimized. The optimization variable is therefore $\boldsymbol{\Delta V}$.

The cost function is thus defined by :

$$f = \sum_{i=1}^{n} \Delta v_i = \boldsymbol{c}^T \boldsymbol{\Delta V} \tag{2.3}$$

Where $\boldsymbol{c}^T = \begin{pmatrix} 1 & 1 & ... & 1 \end{pmatrix}^T$. Thus, the number of maneuvers required of the algorithm will define the dimension of the problem, which has a strong impact on computational time, as will be studied in section 4.1.2.

Linear constraints

Linear constraints are twofold :

- The bounded nature of each maneuver. In fact, in the Configuration file given as input, each slot is associated with a $\Delta V_{\max}$ and a $\Delta V_{\min}$. Since the algorithm assumes impulsive maneuvers, it was decided that the time width of a slot would be measured by its $\Delta V_{\max}$, which characterizes the maximum thrust achievable during that slot. The $\Delta V_{\min}$, equal for all slots, characterizes the smallest thrust achievable by the satellite's propulsion system.
- The SK box constraint. The algorithm must check that the primary does not stray too far from its reference orbit at each TCA.

The Java overlay of CACAO must therefore construct the matrix A and the vector $\boldsymbol{b}$ such that :

$$A\boldsymbol{\Delta V} \leq \boldsymbol{b}$$

With A a matrix of dimension $2n + 6N \times n$ and $\boldsymbol{b}$ a column vector of dimension $2n + 6N$. The first 2n lines of this matrix inequality are such that :

$$\begin{cases} \Delta v_i \leq \Delta V^i_{\min} \\ -\Delta v_i \leq -\Delta V^i_{\max} \end{cases}$$

$$\forall i \in [1, m]$$

For SK constraints, let l_Q, l_S and l_W be the half-lengths of the box sides in the local QSW frame. Let $\boldsymbol{r_p}^{\text{ref}}(t^j_{TCA})$ be the position vector of the primary reference at the j^{th} TCA in the GCRF frame. This vector defines the center of the box.

Using the formula (2.2), we express the deviation in position between the actual position of the primary and its reference at this TCA as :

$$\boldsymbol{\Delta\tilde{r}_p}^{\text{ref}}(t^j_{TCA}) = \boldsymbol{r_p}^{\text{ref}}(t^j_{TCA}) - \boldsymbol{\tilde{r}_p}(t^j_{TCA}) = \boldsymbol{\Delta r_p^{\text{ref}}} - \boldsymbol{\Phi^j \Delta V}$$

With $\boldsymbol{\Delta r_p^{\text{ref}}}$ the distance between the reference and the primary before maneuvering. To compare the elements of this vector with the dimensions of the SK box, we need to rotate it to the QSW reference frame. Denoting M^j the rotation matrix from the GCRF frame to the local QSW frame centered on the reference at the j^{th} TCA and $\boldsymbol{q}$, $\boldsymbol{s}$ and $\boldsymbol{w}$ the canonical basis of this frame, the inequalities become :

$$\begin{cases} \boldsymbol{q}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_Q \\ -\boldsymbol{q}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_Q \\ \boldsymbol{s}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_S \\ -\boldsymbol{s}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_S \\ \boldsymbol{w}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_W \\ -\boldsymbol{w}^T M^j \boldsymbol{\Delta}\tilde{\boldsymbol{r}}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^j) \leq l_W \end{cases}$$
$$\forall j \in [1, N]$$

Thus, the matrix A and the vector $\boldsymbol{b}$ are defined by :

$$A = \begin{bmatrix} I_n \\ -I_n \\ -\boldsymbol{q}^T M^1 \boldsymbol{\Phi^1} \\ \vdots \\ \boldsymbol{w}^T M^1 \boldsymbol{\Phi^1} \\ \vdots \\ \vdots \\ -\boldsymbol{q}^T M^N \boldsymbol{\Phi^N} \\ \vdots \\ \boldsymbol{w}^T M^N \boldsymbol{\Phi^N} \end{bmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} \Delta V_{\text{min}}^1 \\ \vdots \\ \Delta V_{\text{min}}^n \\ -\Delta V_{\text{max}}^1 \\ \vdots \\ -\Delta V_{\text{max}}^n \\ l_Q - \boldsymbol{q}^T M^1 \boldsymbol{\Delta r}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^1) \\ \vdots \\ l_W + \boldsymbol{w}^T M^1 \boldsymbol{\Delta r}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^1) \\ \vdots \\ \vdots \\ l_Q - \boldsymbol{q}^T M^N \boldsymbol{\Delta r}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^N) \\ \vdots \\ l_W + \boldsymbol{w}^T M^N \boldsymbol{\Delta r}_{\boldsymbol{p}}^{\text{ref}}(t_{TCA}^N) \end{pmatrix} \tag{2.4}$$

Euclidean quadratic constraints

The quadratic constraints that must be satisfied by the output solution of the B&B algorithm ensure a minimum distance between the two objects at each TCA after maneuvering. This distance is assumed to be Euclidean. Let's denote $\boldsymbol{r_s}(t_{TCA}^j)$ the position vector of the secondary at the j^{th} TCA in the GCRF reference frame. This vector is obtained by rotating the reference frame of the vector state contained in the associated CDM. The position vector relative to the two objects after maneuvers is therefore defined by $\tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t_{TCA}^j) = \tilde{\boldsymbol{r}}_{\boldsymbol{p}}(t_{TCA}^j) - \boldsymbol{r_s}(t_{TCA}^j)$ in the GCRF reference frame.

The algorithm must therefore construct N scalar inequalities. The generic form of these inequalities is :

$$d^2 \leq p + 2\boldsymbol{q}^T \boldsymbol{\Delta V} + \boldsymbol{\Delta V}^T Q \boldsymbol{\Delta V} \tag{2.5}$$

With p a scalar, $\boldsymbol{q}$ a vector of dimension n and Q a matrix of dimension $n \times n$. In the Euclidean case, a collision plane must be defined for each encounter after maneuvering. Assuming uniform rectilinear relative motion for a very short time interval centered on t^j_{TCA}, the motion of the primary relative to the secondary is carried by the direction vector :

$$\tilde{\boldsymbol{e}}_{\boldsymbol{j}} = \frac{\tilde{\boldsymbol{v}}_{\boldsymbol{r}}(t^j_{TCA})}{||\tilde{\boldsymbol{v}}_{\boldsymbol{r}}(t^j_{TCA})||}$$

As the maneuvers carried out have modified the trajectory of the primary, the date t^j_{TCA} no longer corresponds to the minimum approach between the two objects. The new minimum distance is then calculated by the norm of the vector $\tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})$ projected onto the BPlane associated with this encounter, i.e. :

$$\tilde{\boldsymbol{r}}^{\min}_{\boldsymbol{r}} = (I_3 - \tilde{\boldsymbol{e}}_{\boldsymbol{j}} \tilde{\boldsymbol{e}}^T_{\boldsymbol{j}}) \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})$$

With $(I_3 - \tilde{\boldsymbol{e}}_{\boldsymbol{j}} \tilde{\boldsymbol{e}}^T_{\boldsymbol{j}})$ the symmetrical, idempotent projection matrix onto the j^{th} collision plane. In the following, this matrix will be denoted P^j. Taking the squared norm of the preceding vector, we obtain :

$$\tilde{d}^2_{\text{Eucl, j}} = \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})^T P^{jT} P^j \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA}) = \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})^T P^j \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA}) \tag{2.6}$$

Now, by linearizing the dynamics of the problem (equation (2.2)), the relative position vector is :

$$\tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA}) = \boldsymbol{r}_{\boldsymbol{p}}(t^j_{TCA}) - \boldsymbol{r}_{\boldsymbol{s}}(t^j_{TCA}) + \boldsymbol{\Phi^j \Delta V} = \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) + \boldsymbol{\Phi^j \Delta V} \tag{2.7}$$

The formula (2.6) thus becomes :

$$\tilde{d}^2_{\text{Eucl, j}} = \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA})^T P^j \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) + 2\boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA})^T P^j \boldsymbol{\Phi^j \Delta V} + \boldsymbol{\Delta V}^T \boldsymbol{\Phi}^{jT} P^j \boldsymbol{\Phi^j \Delta V}$$

So, for each encounter j, we can relate the terms of the equation (2.5) to those of the previous equation. The matrices that construct the inequality are therefore :

$$\begin{cases} Q_j = \boldsymbol{\Phi}^{jT} P^j \boldsymbol{\Phi^j} \\ q_j = \boldsymbol{\Phi}^{jT} P^j \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) \\ p_j = \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA})^T P^j \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) \end{cases}$$

A simplification is applied to this problem. Indeed, the vector $\tilde{\boldsymbol{e}}_{\boldsymbol{j}}$ depends on the optimization variable, as it is the direction of the relative speed after maneuvering. To keep the structure of the optimization problem simple and allow it to be solved by algorithms such as the B&B, CACAO replaces $\tilde{\boldsymbol{v}}_{\boldsymbol{r}}(t^j_{TCA})$ by $\boldsymbol{v}_{\boldsymbol{r}}(t^j_{TCA})$ in the expression of the preceding matrices.

Mahalanobis quadratic constraints

The previous formulation is only true if the avoidance criterion is Euclidean. If the avoidance criterion is a Mahalanobis distance, the construction of the quadratic constraints differs. First, we need to define the basis characterizing the collision plane :

$$\mathcal{B}_j = \left(\tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{1}}, \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{2}}, \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{3}}\right)$$

Using the construction defined in the formula (1.2) :

$$\tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{1}} = \frac{\boldsymbol{v}_{\boldsymbol{s}}(t^j_{TCA}) \times \boldsymbol{v}_{\boldsymbol{p}}(t^j_{TCA})}{||\boldsymbol{v}_{\boldsymbol{s}}(t^j_{TCA}) \times \boldsymbol{v}_{\boldsymbol{p}}(t^j_{TCA})||} \qquad \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{2}} = \tilde{\boldsymbol{e}}_{\boldsymbol{j}} \qquad \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{3}} = \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{1}} \times \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{2}}$$

The matrix $R_{\mathcal{B}_j} = [\tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{1}}\ \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{2}}\ \tilde{\boldsymbol{e}}^{\boldsymbol{j}}_{\boldsymbol{3}}]^T$ corresponds to the rotation matrix from the GCRF inertial frame to the BPlane frame. Considering $C^{rr}_p(t^j_{TCA})$ and $C^{rr}_s(t^j_{TCA})$ the primary and secondary positional covariance matrices given by the associated CDM, we obtain : $\Sigma^{rr}_{\mathcal{B}_j} = R_{\mathcal{B}_j}\Sigma^{rr}R_{\mathcal{B}_j} = R_{\mathcal{B}_j}(C^{rr}_p(t^j_{TCA}) + C^{rr}_s(t^j_{TCA}))R_{\mathcal{B}_j}$

$\Sigma^{rr}_{\mathcal{B}_j}$ corresponds to the combined positional covariance matrix in the BPlane reference frame. According to the definition of a Mahalanobis distance, we then have :

$$\tilde{d}^{\,2}_{\text{Mah, j}} = \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})^T P^j (\Sigma^{rr}_{\mathcal{B}_j})^{-1} P^j \tilde{\boldsymbol{r}}_{\boldsymbol{r}}(t^j_{TCA})$$

The matrices that construct the inequality are thus :

$$\begin{cases} Q_j = \boldsymbol{\Phi}^{jT} P^j (\Sigma^{rr}_{\mathcal{B}_j})^{-1} P^j \boldsymbol{\Phi}^{\boldsymbol{j}} \\ q_j = \boldsymbol{\Phi}^{jT} P^j (\Sigma^{rr}_{\mathcal{B}_j})^{-1} P^j \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) \\ p_j = \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA})^T P^j (\Sigma^{rr}_{\mathcal{B}_j})^{-1} P^j \boldsymbol{r}_{\boldsymbol{r}}(t^j_{TCA}) \end{cases}$$

2.1.3. Solution analysis

Once the problem has been constructed, the B&B algorithm calculates an optimal solution and communicates it to the Java part. The way the Branch & Bound functions is described in section 2.2.

It is then the role of the Java code to analyze the solution to ensure that it is realistic. Solving the optimization problem is based on a number of simplifying assumptions, such as a linearization of the problem's dynamics, or a rectilinear trajectory close to the TCA, which follows from the short-term encounter assumption. It is important to check by numerical propagation that the solution still respects the constraints imposed.

The first step is to recalculate the TCA. The maneuvers carried out by the primary will logically modify its trajectory, shifting the instant of minimum relative distance in time. In the case of a short-term encounter, the new TCA ($\tilde{t}_{TCA}$) is generally less than a second away from the one without maneuvers. However, for encounters with lower relative velocities, this time shift can be of the order of ten seconds.

The TCA is updated using a Brent's method root-finding algorithm, which finds the time t when the distance between the two objects is minimum in a time interval centered on the old TCA. This step therefore requires numerical propagation of the primary after maneuvers and of the secondary.

Consequently, the states $\tilde{\boldsymbol{r}}_{\boldsymbol{p}}(\tilde{t}_{TCA})$ and $\boldsymbol{r}_{\boldsymbol{s}}(\tilde{t}_{TCA})$ are obtained by numerical propagation and their relative distance, Euclidean or Mahalanobis, can thus be evaluated and compared with the desired avoidance criterion.

The position of the reference at the new TCA, $\boldsymbol{r}_{\boldsymbol{p}}^{\text{ref}}(\tilde{t}_{TCA})$ is also propagated to check the SK box constraint. If all constraints are verified, the algorithm returns the calculated avoidance strategy as output.

The figure below summarizes the internal architecture of the CACAO algorithm:

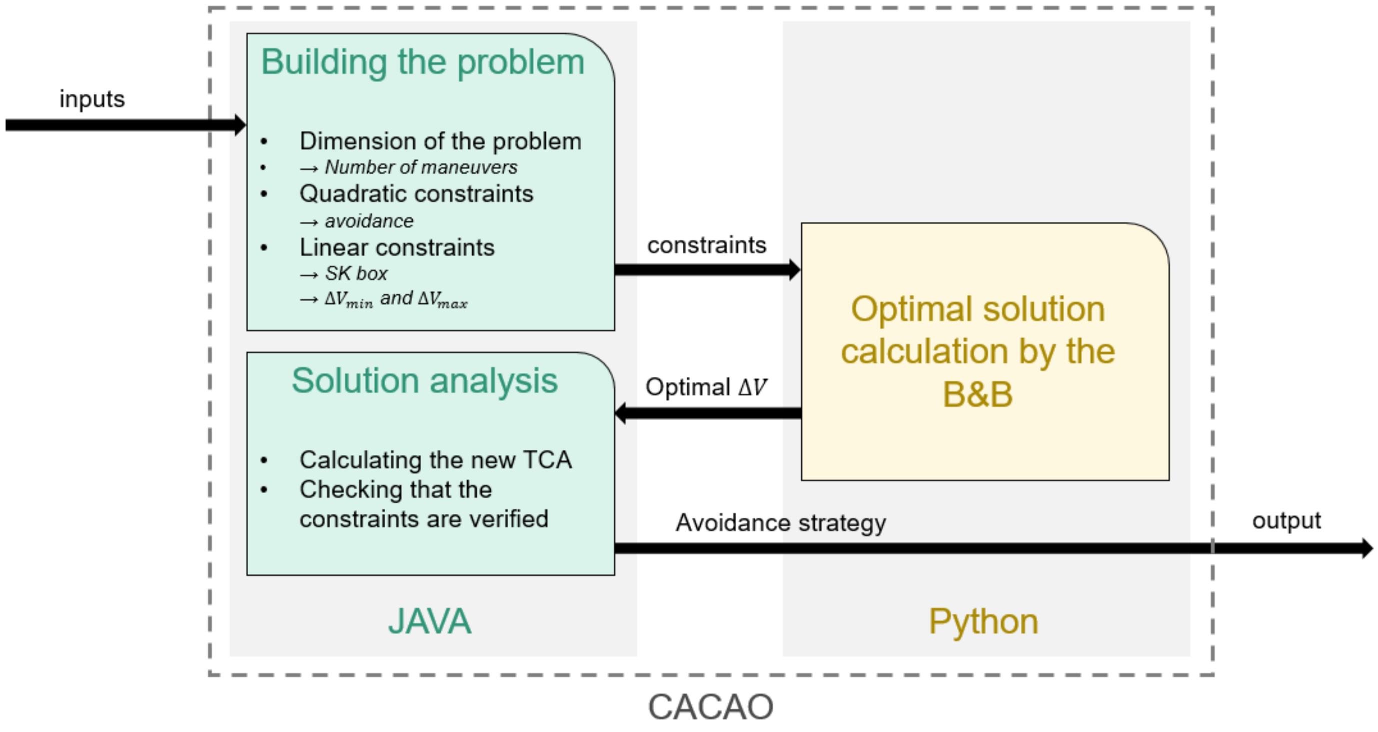


Figure 2.2: Architecture of CACAO

2.2. The Branch & Bound algorithm

To solve the constrained optimization problem created by the Java overlay, CACAO uses the Branch & Bound algorithm. This algorithm, implemented on Python, aims to minimize the sum of the ΔV of a succession of maneuvers while verifying linear and quadratic constraints. The dimension of the volume, n, in which the solution will be sought corresponds to the number of maneuvers. This type of algorithm was studied for the first time in [12].

2.2.1. Scaling the problem

The first task performed by the B&B is a scaling of the constraint matrices transmitted by the Java overlay. The main objective of this step is to ensure that all potential solutions lie within the $[-1, 1]^n$ hypercube.

This scaling method has been implemented in two ways in the Python code. The first corresponds to that used by the LAAS in their report [1] to test the B&B on theoretical examples, and which was implemented by Thalès Services Numériques. However, TSN has also implemented a second scaling method, the one selected by default in the algorithm.

Choosing one method or the other has no impact on the algorithm's working principle in theory, but the minimum volume of ellipsoids chosen as input will not have the same significance. Changing method without varying this value will therefore have an impact on computational time and result accuracy. Only the default method was used during the internship.

This scaling included a problem for avoidance situations with one maneuver, the correction for which is mentioned in 2.3.3.

2.2.2. The ellipsoid method

The LAAS report presents the general principle of the algorithm in the context of minimizing a convex function $f : \mathbb{R}^n \to \mathbb{R}$ not necessarily differentiable throughout its domain of definition, which corresponds to the optimization problem under study. The idea is to construct a sequence of $\mathbb{R}^n$ ellipsoids of decreasing size, all containing a minimizer of the function f. To do this, we use the fact that for a subgradient computed at a point, it is possible to identify or compute (using a secant plane) a half-space containing this point and containing no minimizer of f.

An ellipsoid $\mathcal{E}$ is characterized by a matrix P of dimension $n \times n$ which determines its shape and size, and by its center $\boldsymbol{o}$.

Let $\mathcal{E}(k)$ be the ellipsoid given at iteration k and containing a minimizer of f. Let $\partial f(k)$ be the subdifferential of f at iteration k, i.e. the set of subgradients of the function at the point $\boldsymbol{o}^{(k)}$, the center of $\mathcal{E}(k)$. The basic iteration of the ellipsoid algorithm consists of two steps, repeated until the end criterion, defined below, is reached.

1. Calculate a subgradient $\boldsymbol{h}(k) \in \partial f(k)$.

2. Calculate the ellipsoid $\mathcal{E}(k+1)$ of minimum volume containing the half-ellipsoid :

$$\mathcal{E}^{(k)} \bigcap \left\{ \boldsymbol{z} \mid \boldsymbol{h}^{(k)^T} (\boldsymbol{z} - \boldsymbol{o}^{(k)}) \leq 0 \right\}$$

which contains a minimizer of f.

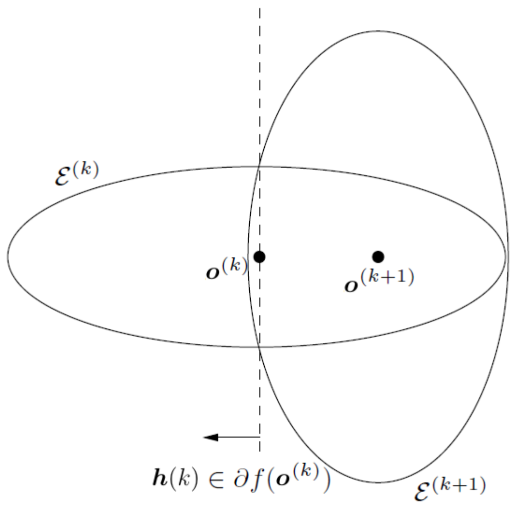


Figure 2.3: Iteration of the ellipsoid method

The first step is very straightforward in our case, since the objective function is linear and therefore differentiable, and calculating the subgradient will yield the vector $\boldsymbol{c}$ at each step. To clarify the second step, we need to show how to calculate the minimum volume ellipsoid containing the half-ellipsoid and including a minimizer of the function. An $\mathcal{E}$ ellipsoid with center $\boldsymbol{o}$ and matrix P is the set defined by :

$$\mathcal{E} = \left\{ \boldsymbol{z} \mid (\boldsymbol{z} - \boldsymbol{o})^T P^{-1} (\boldsymbol{z} - \boldsymbol{o}) \leq 1 \right\}$$

With P defined positive. The volume of the ellipsoid is calculated from the volume of the $\mathbb{R}^n$ unit ball such that :

$$\mathrm{vol}(\mathcal{E}) = \frac{\pi^{n/2}}{\Gamma(n/2+1)} \sqrt{\det(P)}$$

With Γ the Gamma function. The ellipsoid of minimal volume containing the half ellipsoid defined by :

$$\left\{ \boldsymbol{z} \mid (\boldsymbol{z} - \boldsymbol{o})^T P^{-1} (\boldsymbol{z} - \boldsymbol{o}) \leq 1,\ \boldsymbol{h}^T (\boldsymbol{z} - \boldsymbol{o}) \leq 0 \right\}$$

is given by :

$$\mathcal{E}_{\min} = \left\{ \boldsymbol{z} \mid (\boldsymbol{z} - \boldsymbol{o}^+)^T P^{+^{-1}} (\boldsymbol{z} - \boldsymbol{o}^+) \leq 1 \right\}$$

with :

$$\boldsymbol{o}^{+} = \boldsymbol{o} - \frac{1}{n+1} P \tilde{\boldsymbol{h}}$$

$$P^{+} = \frac{n^2}{n^2-1} \left(P - \frac{2}{n-1} P \tilde{\boldsymbol{h}} \tilde{\boldsymbol{h}}^T P \right)$$

and the normalized subgradient $\tilde{\boldsymbol{h}} = \frac{\boldsymbol{h}}{\sqrt{\boldsymbol{h}^T P \boldsymbol{h}}}$. The preceding equations are used to update the ellipsoids in the algorithm, and the equation of the updated center $\boldsymbol{o}^{+}$ corresponds to a step in the direction of the opposite of the subgradient in the coordinates defined by P.

2.2.3. B&B working principle

The following pseudo-code shows the complete B&B algorithm :

Algorithm 2.1 Branch & Bound (n, EllipsoidsList, LinearConstrList, c, $k_{\max}$, $V_{\min}$)

```
Require: Dimension $n$, non-convex quadratic constraints EllipsoidsList, linear constraints LinearConstrList, linear cost function $c$, loop parameters ($k_{\max}$, $V_{\min}$).
Ensure: Center of final ellipsoid and cost $(o_m, m)$.
  # Start with a sphere in which the hypercube $[-1,1]^n$ is contained
  $P_0 \leftarrow nI_n$
  $o_0 \leftarrow (0, ..., 0)$
  $k \leftarrow 1$
  $L[k] \leftarrow (P_0, o_0)$
  $m \leftarrow +\infty$
  while $k < k_{\max}$ and $k > 0$ do
    $(P, o) \leftarrow L[k]$
    if VolumeEllipsoid$(n, P) < V_{\min}$ then
      $L_{\text{keep}} \leftarrow (P, 0)$
      $k \leftarrow k - 1$
    else
      # Use the ellipsoid algorithm to reduce the current ellipsoid with respect to the linear constraints, until no reduction is possible: the center is a feasible point or the ellipsoid is completely unrealizable
      $(P, o) \leftarrow$ CutEllipsoidLinearConstraints $(n, P, o,$LinearConstrList$)$
      if $P = \emptyset$ then
        $k \leftarrow k - 1$
      end if
      # Test the feasibility of the current center with respect to the non-convex constraints
      if IsFeasible($o$,EllipsoidList) then
        # Update the current minimum
        if $m > min(m, c^T o)$ then
          $m \leftarrow min(m, c^T o)$
          $o_m \leftarrow o$
        end if
        # Reduce the objective by the unconstrained ellipsoid method
        $(P, o) \leftarrow$ CutEllipsoidObjective$(n, P, o, c)$
        $L[k] \leftarrow (P, o)$
      else
        # If the center of the current ellipsoid is not feasible for non-convex constraints, subdivide
        $(P_1, o_1, P_2, o_2) \leftarrow$ BisectEllipsoid$(n, P, o)$
        $L[k] \leftarrow (P_1, o_1)$
        $k \leftarrow k + 1$
        $L[k] \leftarrow (P_2, o_2)$
      end if
    end if
  end while
  return $(o_m, m)$
```

First, let's recall the inputs and outputs of the algorithm :

- As already mentioned, the dimension n of the problem corresponds to the number of maneuvers required to mitigate the risk.
- Non-convex quadratic constraints are linked to the miss distance.
- Linear constraints correspond to the fixed $\Delta V_{\min}$ and $\Delta V_{\max}$, as well as the SK box in which the primary must stay.
- $k_{\max}$ is the maximum number of unvisited ellipsoids that can be stored.
- Finally, $V_{\min}$ corresponds to the minimum volume an ellipsoid can reach.

The outputs are the center of the final ellipsoid and the associated cost.

First, the algorithm instantiates the variables P_0 and $\boldsymbol{o_0}$ defining the initial ellipsoid which includes the hypercube $[-1, 1]^n$. This ellipsoid is added to the list of ellipsoids to visit. The cost is initialized to $+\infty$.
The *while* loop is at the heart of the algorithm, and is its exit point. The algorithm terminates if the index k exceeds $k_{\max}$ (too many ellipsoids waiting to be processed) or if it reaches 0 (all ellipsoids with a volume greater than $V_{\min}$ have been visited). The first step in this loop is to retrieve the center and matrix of the ellipsoid in question, taken from the list of ellipsoids to be visited. Next, a volume check is performed (algorithm 2.2):

Algorithm 2.2 VolumeEllipsoid (n, P)

```
Require: Dimension $n$, Matrix $P$.
Ensure: Volume $v$ of the ellipsoid $X'PX \leq 1$.
  $v \leftarrow \dfrac{det(P)\pi^{n/2}}{\Gamma(n/2+1)}$
  return $v$
```

If this volume is less than the minimum volume defined in the input, the k index is decreased, which is equivalent to eliminating the current ellipsoid from the list, and the algorithm returns directly to the previous step. Otherwise, it continues and processes this ellipsoid.
Processing of the ellipsoid begins with a cut with respect to the linear constraint functions. This is done using the 2.3 algorithm :

Algorithm 2.3 CutEllipsoidLinearConstraints (n, P_0, o_0, LinearConstrList)

```
Require: Dimension $n$, $(P_0; o_0)$, LinearConstrList.
Ensure: $(P,o)$
  # Method that uses the ellipsoid algorithm to reduce the current ellipsoid with respect to the linear constraints, until no reduction is possible : the center is a feasible point or the ellipsoid is completely unrealizable.
  $k \leftarrow 0$
  while $o_k$ is not feasible do
    if $g_i(o_k) > 0$ pour $g_i \in$ LinearConstrList then
      # Compute the gradient $g^{(k)} = \partial g_i(o_k)$
      if $g_i(o_k) > \sqrt{g^{(k)T} P_k g^{(k)}}$ then
        # End the algorithm if all the points of the new ellipsoid violate the constraint i
        return $(\emptyset, \emptyset)$
      else
        $\tilde{g}^{(k)} \leftarrow \frac{g^{(k)}}{\sqrt{g^{(k)T} P_k g^{(k)}}}$
        $o_{k+1} \leftarrow o_k - \frac{1}{1+n} P_k \tilde{g}^{(k)}$
        $P_{k+1} \leftarrow \frac{n^2}{n^2-1} \left( P_k - \frac{2}{n+1} P_k \tilde{g}^{(k)} \tilde{g}^{(k)T} P_k \right)$
        $k \leftarrow k+1$
      end if
    end if
  end while
  return $(P_k, o_k)$
```

If the center of the ellipsoid already satisfies all the linear constraints, then nothing happens and the above algorithm returns the original ellipsoid. Otherwise, it is cut along the opposite gradient of each constraint not verified. If no point on the ellipsoid satisfies all the linear constraints, then an empty tuple is returned and k is decremented. Otherwise, the ellipsoid satisfying the linear constraints is returned. The calculations performed correspond to the cutting operations presented in the previous section.

Once the ellipsoid has been reduced so that its center respects the linear constraints, it is then tested against the quadratic constraints. If these are verified and the current center multiplied by the cost function is smaller than the current minimum, then the latter is updated. The current ellipsoid is then cut according to the constrained function (algorithm 2.4) :

Algorithm 2.4 CutEllipsoidObjective (n, P_0, o_0, c)

```
Require: Dimension $n$,$(P_0; o_0)$, linear objective $c^T X$.
Ensure: $(P$,$o)$.
  # Method to reduce the objective ellipsoid by the unconstrained ellipsoid method.
  # The gradient is simply $g = c$
  $\tilde{c} \leftarrow \frac{c}{\sqrt{c^T P_0 c}}$
  $o \leftarrow o_0 - \frac{1}{1+n} P_k \tilde{c}$
  $P \leftarrow \frac{n^2}{n^2-1} \left( P_0 - \frac{2}{n+1} P \tilde{c} \tilde{c}^T P \right)$
  return $(P_0, o_0)$
```

This new ellipsoid is then added to the list of ellipsoids to be visited if its center respects the quadratic constraints. Otherwise, a division algorithm is used (algorithm 2.5) :

Algorithm 2.5 BisectEllipsoid (n, P_0, o_0)

```
Require: Dimension $n$, $(P_0; o_0)$.
Ensure: Two ellipsoids $(P_1; o_1)$ et $(P_2; o_2)$ which contain the initial ellipsoid.
  # Method to cut the ellipsoid in half using the ellipsoid method based on normal to longest axis.
  $g \leftarrow$ the eigenvector of P0 corresponding to the greatest eigenvalue of P0
  $\tilde{g} \leftarrow \frac{g}{\sqrt{g^T P_0 g}}$
  $o_1 \leftarrow o_0 - \frac{1}{1+n} P_0 \tilde{g}$
  $P_1 \leftarrow \frac{n^2}{n^2-1} \left( P_0 - \frac{2}{n+1} P \tilde{g} \tilde{g}^T P \right)$
  $o_2 \leftarrow o_0 + \frac{1}{1+n} P_0 \tilde{g}$
  $P_2 \leftarrow P_1$
  return $(P_1, o_1)$, $(P_2, o_2)$
```

The ellipsoid is cut with respect to the eigenvector of the largest eigenvalue of the ellipsoid matrix. The two ellipsoids created have the same shape but a different center. They are added to the list of ellipsoids to visit.

The process continues until either :

- The list of ellipsoids to visit is empty.
- The maximum number of ellipsoids in this list has been reached.

In this case, the algorithm returns the minimum cost and the center of the corresponding ellipsoid.

2.3. Modifications of the algorithm

Much of the work carried out during the first half of the internship consisted of becoming familiar with the algorithm, adding new features and correcting errors. The aim was to make CACAO as robust and computationally efficient as possible.

In this section, we'll first look at the main changes that have been added before discussing a more minor but yet impactful error correction.

2.3.1. SK constraints

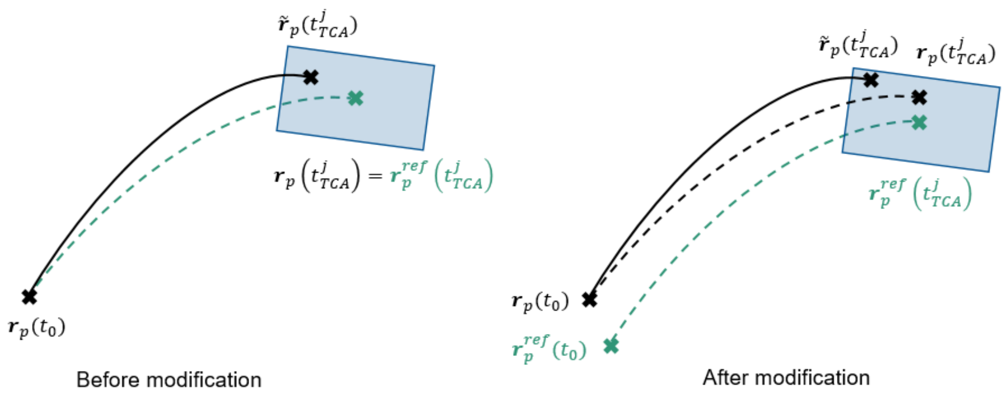
Modification of the reference

One of the features to be improved was the management of SK constraints. As explained in 2.1.2, these linear constraints ensure that the primary remains inside a box centered on the reference at each TCA.
However, in the initial version of the algorithm, this reference was simply defined by the position of the satellite without maneuvers at the TCA, i.e. the position indicated in the CDM. Using previous notations, this translates into :

$$\boldsymbol{r_p}^{\mathrm{ref}}(t^j_{TCA}) = \boldsymbol{r_p}(t^j_{TCA})$$

This means that the satellite was considered to be in the center of its SK box on every collision. However, in a real-life situation, this condition cannot be guaranteed, as the primary may be close to one of the edges, which could have an impact on the avoidance strategy to be undertaken.

The algorithm has therefore been modified so that an orbital state can be given as input and used to define the reference orbit on which the SK box is centered.



$\tilde{\boldsymbol{r}}_p(t^j_{TCA})$
$\boldsymbol{r}_p\left(t^j_{TCA}\right) = \boldsymbol{r}_p^{ref}\left(t^j_{TCA}\right)$
$\boldsymbol{r}_p(t_0)$
Before modification
$\tilde{\boldsymbol{r}}_p(t^j_{TCA})$
$\boldsymbol{r}_p(t^j_{TCA})$
$\boldsymbol{r}_p^{ref}\left(t^j_{TCA}\right)$
$\boldsymbol{r}_p(t_0)$
$\boldsymbol{r}_p^{ref}(t_0)$
After modification

Figure 2.4: SK box modification

Addition of control points

Another problem with this constraint was that the position of the primary in relation to its reference was only checked at each TCA. Retention in the box was not ensured over the entire time interval relevant to the risk.
The B&B algorithm is not designed to handle linear constraints that are continuous over time. It was therefore decided simply to increase the number of dates at which the constraint must be checked. These dates are referred to as control points in the rest of the report.

A first option considered was to add a very large number of control points at regular intervals in the period defined between the first maneuver and the last TCA. However, this method requires the orbits of the primary and its reference to be propagated to each of these points, which significantly increases computational time.

The solution finally adopted to solve this problem was to add control points on well-chosen dates to ensure at best as possible the SK. At present, these are located :

- At every TCA, as was already the case.
- A half-orbit after each maneuver, corresponding to a date when the radial shift between the primary before and after maneuvers is at its widest.
- A half-orbit after the last TCA. This is the date at which the satellite will generally maneuver to reposition itself on its reference orbit after the avoidance. The role of this control point is to ensure that the accumulated phase shift between the primary and its reference orbit since the first maneuver doesn't cause it to leave its box.

The number of linear constraints to be satisfied by the B&B solution is now $2n + 6(N + n + 1) = 8n + 6N + 6$. The dimensions of the matrix A and the vector $\boldsymbol{b}$ defined above have thus increased.

2.3.2. Multi-risk avoidance

A second important feature added to the algorithm during the internship was the ability to handle multi-risk avoidance situations. Indeed, the algorithm had been designed to handle such cases, but had not been tested for them, so several parts of the code had to be modified.

Modification of the numerical propagator

One problem encountered during the implementation of the multi-risk avoidance was the lack of precision of the propagator used in CACAO. Indeed, the algorithm had been implemented with a propagator that took no perturbations into account, so the dynamics of the problem were Keplerian. This problem was striking in a multi-risk case, as the primary's orbital state contained in the various CDMs, although generated on the same date, were completely incoherent. When these states were propagated to the same date, differences of several tens of km in position could be observed.

This problem was solved by adding perturbations to the propagator used by the Java overlay, and in particular by changing the Earth's geopotential model to include harmonics.

Some CDMs indicate which types of disturbance have been taken into account to generate it. In the examples presented in section 4.1, the geopotential model is always the same, EGM-96, with a number of harmonics that varies from case to case. The gravitational attraction of the Moon and Sun and the solar radiation pressure are also taken into account. Atmospheric friction, on the other hand, is only modelled for LEO cases.

The algorithm will then take this information into account to use the same geopotential model and the same number of harmonics, as well as other perturbations such as the gravitational influence of the Moon and Sun with the aim of increasing the coherence between CDMs.

On the other hand, certain forces such as solar radiation pressure or friction are not taken into account by CACAO, as they require satellite characteristics that are not indicated in the CDMs.

On the one hand, this modification has significantly increased the computational time required for the various propagations performed by the algorithm. On the other hand, the previously observed inconsistencies from one CDM to another have been reduced to the order of a meter when the states are propagated to the same date.

Multi-risk encounter example

Despite the increase of the number of debris and the need for satellites to be able to handle multi-risk avoidance situations, such encounters are still rare, and only two real-life examples were dealt with during the internship. Furthermore, these cases are very similar, both involving situations where the primary and the secondary have very similar orbits, resulting in a series of conjunctions. No examples involving multiple secondaries were available.

The first example concerns a series of encounters between two privately-owned satellites, ICEYE-X2 and CAPELLA-5, the former being the primary. Below is the distance between the two objects as a function of time, before and after maneuvers.

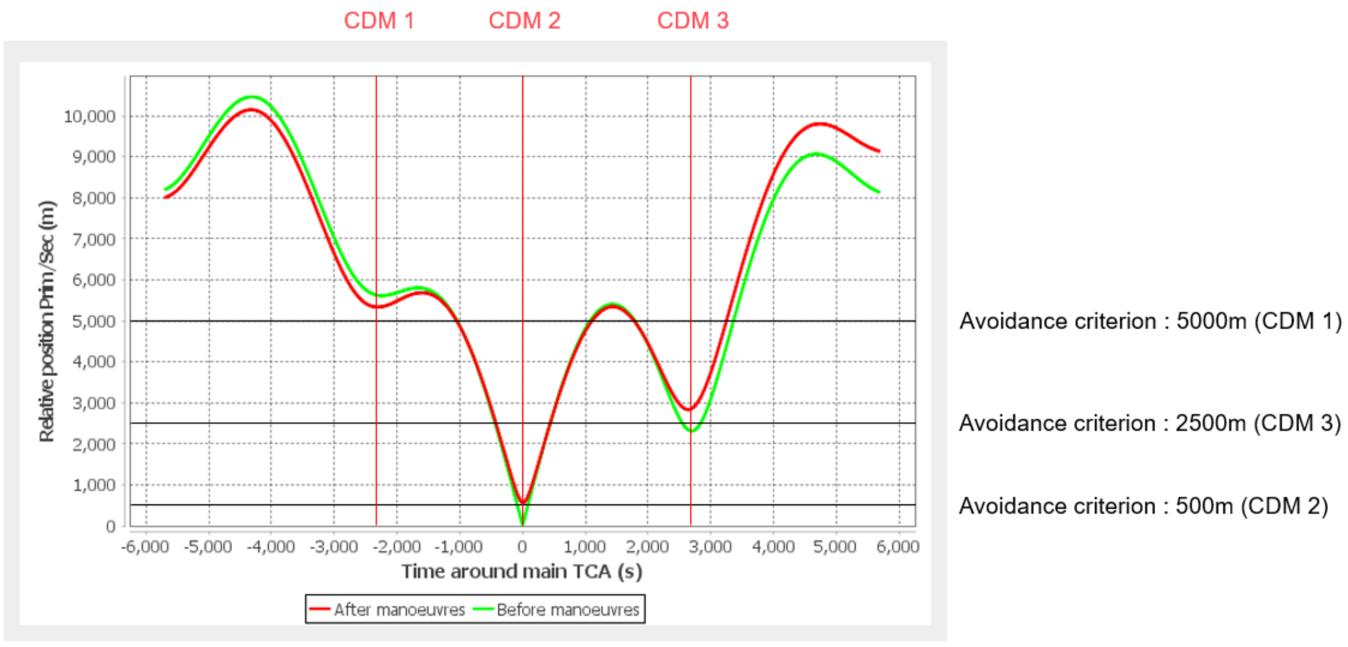


Figure 2.5: Relative position primary/secondary
ICEYE-X2 vs CAPELLA-5

We thus observe a succession of local minima in terms of miss distance, corresponding to conjunctions between the two objects. In all, 14 CDMs were generated regarding these two satellites, but only the 3 most dangerous encounters were taken into account in this case. The first and last, CDMs 1 and 3, correspond to miss distances of 5613 and 2314 m respectively. These distances are large, and such encounters would probably not require maneuvers on their own. However, CDM 2 corresponds to a miss distance of 35 m, a dangerous situation that calls for maneuvering.

A different Euclidean avoidance criterion has been imposed for each risk, namely 5000, 500 and 2500m. This is not necessary: if the two objects remain more than 500m apart for each TCA, the risks can be considered to have been sufficiently reduced. However, imposing miss distances after maneuvers that are greater than the initial miss distance for the last two TCAs shows that CACAO is capable of finding a solution that reduces the risk on two different encounters at the same time.

The example we're dealing with here is a special one, since only one encounter is risky. The aim of the algorithm is to perform the "main" avoidance, while ensuring that the risks associated with the previous and following conjunctions are not too great.

The second case is very similar. It involves a succession of conjunctions between the ICEYE-X13 satellite and a space debris.

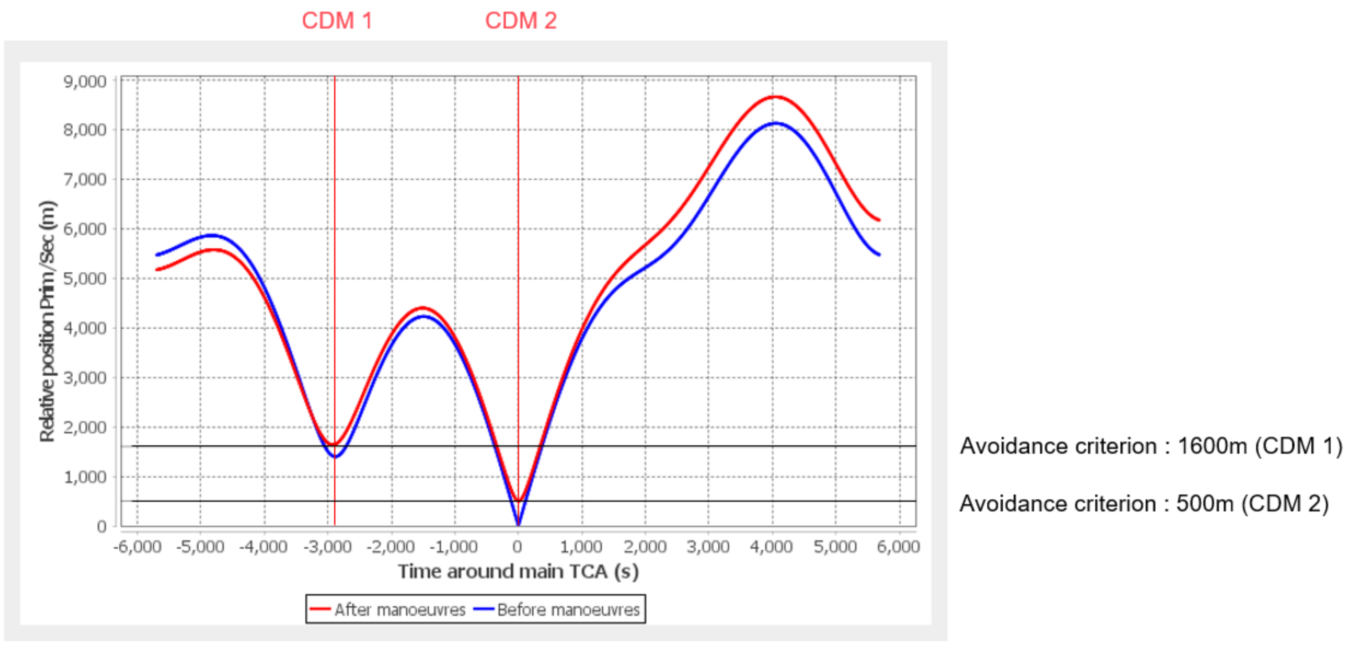


Figure 2.6: Relative position primary/secondary ICEYE-X13 vs debris

This time, only two CDMs were considered. The first is associated with a miss distance of 1401m and the second with 11m.

Slot pre-treatment

The third important new feature is a method for pre-treatment of the maneuver slots. In CACAO's initial version, the slots given as input were all used by the algorithm, i.e. giving 5 dates meant asking the satellite to thrust at each of them. However, since the dimension of the optimization problem depends on the number of slots, and this dimension has a very strong impact on computational time, it was almost impossible to use more than two slots.

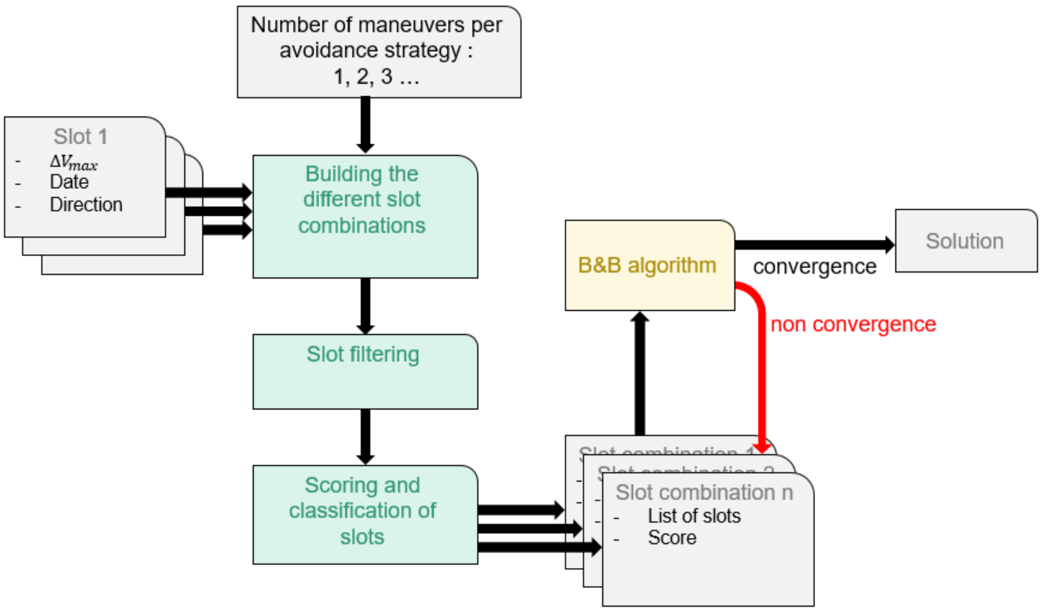


Figure 2.7: Steps for the slot pre-treatment

As such, the algorithm is organized as follows :

1. **Inputs of the algorithm** : The first input is list of n_c of slots. Each of these slots is associated with a date t_{Man}, a ΔV_{max} representing the maximum thrust achievable during this slot, and a direction along one of the axes of the TNW local reference frame, i.e. 6 different directions overall : {T, -T, N, -N, W, -W}.
   The second input is the number of slots per avoidance strategy. This value, which can be an integer or a set of integers, is explained with an example below.

2. **Building the combinations** : The algorithm builds combinations according to the number of slots desired per avoidance strategy. Let's take an example: three slots are given as input that are associated with the dates t^1_{Man}, t^2_{Man} and t^3_{Man}. If the number of slots per given strategy is 2, the algorithm will construct all unique combinations comprising 2 slots, i.e. $\{t^1_{\text{Man}}, t^2_{\text{Man}}\}$, $\{t^1_{\text{Man}}, t^3_{\text{Man}}\}$ and $\{t^2_{\text{Man}}, t^3_{\text{Man}}\}$.
   If this time the number of slots per avoidance strategy given as input is a set of integers such as [1, 2, 3], the algorithm will construct all unique combinations comprising 1, 2 or 3 slots, i.e. $\{t^1_{\text{Man}}\}$, $\{t^2_{\text{Man}}\}$, $\{t^3_{\text{Man}}\}$, $\{t^1_{\text{Man}}, t^2_{\text{Man}}\}$, $\{t^1_{\text{Man}}, t^3_{\text{Man}}\}$, $\{t^2_{\text{Man}}, t^3_{\text{Man}}\}$ and finally $\{t^1_{\text{Man}}, t^2_{\text{Man}}, t^3_{\text{Man}}\}$.

3. **Slot filtering** : Once these N_c combinations have been constructed, CACAO performs an initial filtering. Another input to the algorithm is a minimum slot spacing. In chemical propulsion, this spacing is linked to an attitude stabilization constraint in between maneuvers. In electric propulsion, it is linked to the time needed to recharge the batteries required for the thrust.
   If a combination contains two slots that are too close, it is removed from the list of avoidance strategies considered. This filtering does not apply to strategies involving a single slot. Furthermore, if two slots are given on the same date with different directions, this spacing constraint

doesn't apply. The algorithm then seeks to push in two different directions at the same time, which amounts to aligning the nozzles with an axis other than one of the main axes of the TNW frame.

4. **Slot scoring** : After this filtering, the number of combinations is $\tilde{N}_c$. The algorithm then computes a score for each slot to identify the most interesting ones. This ranking depends on 3 criteria. Details of this score are given below for a slot with a date t^i_{Man} :

$$\text{Score} = k_1 \frac{\Delta V^i_{\text{max}}}{\Delta V_{\text{MAX}}} + k_2 \frac{\Delta V_{\text{target}}(\Delta\alpha^{\text{opt}})}{\Delta V_{\text{target}}(\Delta\alpha^i)} + k_3 \frac{\Delta t^i_{\text{Man}}}{\Delta t_{\text{max}}} \tag{2.8}$$

With :

$$\Delta V_{\text{target}}(x) = \sqrt{\frac{\mu}{R}} \left(\sqrt{\frac{(R+\Delta R)(\cos(x)-1)}{(R+\Delta R)\cos(x) - R}} - 1 \right) \forall x \in [0, 2\pi] \tag{2.9}$$

The first term of this score favors slots with a high value of ΔV^i_{max}. ΔV_{MAX} corresponds to the maximum of this value among all slots, i.e. :

$$\Delta V_{\text{MAX}} = \max(\Delta V^i_{\text{max}}) \ \forall i \in [1, n_c]$$

The second term favors slots whose argument of latitude, α, is close to the one opposite to the risk. For each slot, the algorithm calculates the value $\Delta\alpha^i = \alpha^i - \alpha^{\text{TCA}} \in [0, 2\pi]$. The optimal $\Delta\alpha$ is thus $\Delta\alpha^{\text{opt}} = \pi$.

The function ΔV_{target} corresponds to the analytical value of the ΔV of a tangential maneuver required to create a radial deviation ΔR from a circular orbit of radius R, as a function of the relative argument of latitude between the maneuver and the point where the deviation is desired. In the algorithm, R is defined by the semi-major axis of the primary's orbit, μ corresponds to the Earth's gravitational parameter and finally ΔR is equal to the desired Euclidean avoidance criterion. If this criterion is Mahalanobis, a value of 500 m is taken by default.

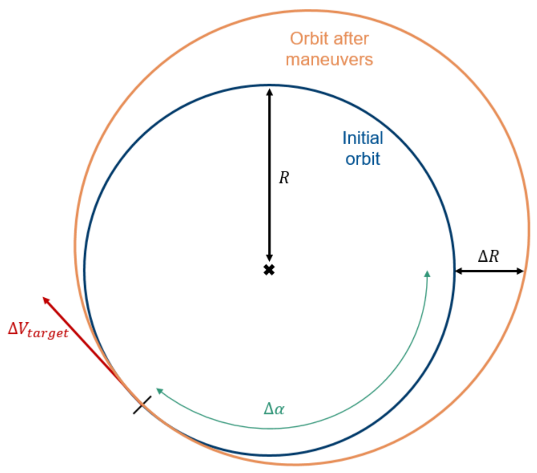


Figure 2.8: ΔV_{target} computation

This function admits a minimum when $x = \pi$ and diverges towards infinity at the edges of the interval $[0, 2\pi]$. This minimum is :

$$\Delta V_{\text{target}}(\Delta\alpha^{\text{opt}}) = \sqrt{\frac{\mu}{R}}\left(\sqrt{\frac{2(R+\Delta R)}{2R+\Delta R}} - 1\right)$$

The function f is defined by the formula :

$$f(x) = \frac{\Delta V_{\text{target}}(\Delta\alpha^{\text{opt}})}{\Delta V_{\text{target}}(x)}$$

Its plot between 0 and 1 for the values $R = 7000$ km, $\Delta R = 500$ m and $\mu = 398600$ km^3/s^2 is reported below :

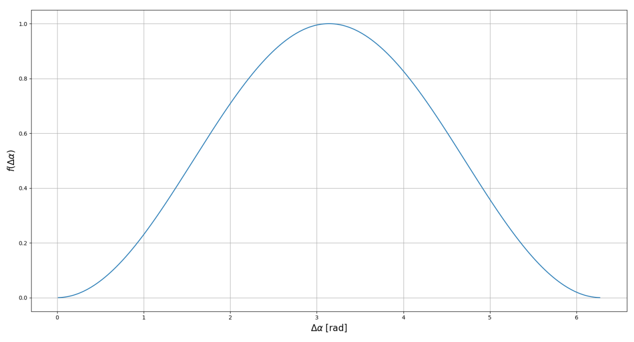


Figure 2.9: Plot of f

Finally, the last term favors slots whose date is close to the TCA. Maneuvering as early as possible will accumulate a phase shift between the primary and its reference, which will increase the time required for repositioning after the risk. Moreover, a late maneuver allows us to take into account the evolution of the risk and can be modified and canceled right up to the last moment, guaranteeing a better reactivity. The measured time interval is :

$$\Delta t^i_{\text{Man}} = t^N_{TCA} - t^i_{\text{Man}}$$

With t^N_{TCA} the latest TCA. This term is normalized by $\Delta t_{\max}$, the time difference between t^N_{TCA} and the initial date t_0.

Each of these terms is between 0 and 1 and the coefficients k_1, k_2 and k_3 can be adjusted to prioritize one term over the others.

5. **Solution computation** : The aim of the previous selection is to determine the most interesting combinations before any optimization. As the algorithm is intended to be flown on satellites with limited computational power, minimizing the computational time is a priority.

   The chosen option is to start with the combination with the best score and construct the optimization problem associated with this avoidance strategy. If the B&B algorithm does not converge, the solution with the second-highest score is then given as input to the optimization, and so on.

   There are two possible reasons for this non-convergence. Either it was impossible to achieve avoidance with the given slots, due to their dates and their associated maximum thrusts, or the algorithm exceeds the maximum convergence time allowed. Indeed, to avoid a situation where the B&B takes too long to converge, a *Time Out* criterion stops the optimization after a certain time. The value of this criterion must be carefully chosen according to the computational power available to avoid a situation where the algorithm never has time to arrive at a solution.

   However, if the B&B converges and the solution satisfies the required constraints, the process stops. This solution is given as output, and other slot combinations are forgotten.

2.3.3. B&B error correction

The modifications mentioned so far represent the most important additions to the algorithm. However, several minor modifications have been made to make the algorithm more robust and computationally efficient. One critical error correction is detailed below :

In its initial state, the algorithm was unable to converge when taking a single slot as input, which represents the majority of real debris avoidance cases. The problem lay not with the Java overlay, but with the Python implementation of the B&B. In fact, the problem stemmed from the 2.3 algorithm, in particular the ellipsoid division formula reported below:

$$P_{k+1} \leftarrow \frac{n^2}{n^2-1}\left(P_k - \frac{2}{n+1}P_k\tilde{g}^{(k)}\tilde{g}^{(k)T}P_k\right)$$

For one-slot avoidance situations, the problem dimension n is equal to 1. The problem associated with the first factor in the previous equation had been taken into account, the term $\frac{n^2}{n^2-1}$ being replaced by a unit factor in such a case.

However, another problem arose from the scaling of the problem mentioned above. Considering the initial ellipsoid, which corresponds to the interval $[-1, 1]$ in a 1-dimensional case, the terms P_k and $\tilde{g}^{(k)}$ are both equal to 1. This means that the term P_{k+1} characterizing the width of the two intervals obtained by cutting was zero. As their volume was thus less than the minimum volume criterion given as input, the algorithm stopped immediately, having studied only one solution. This was corrected by modifying the ellipsoid cut in a one-dimensional case so that :

$$o_{k+1} \leftarrow o_k \frac{1}{2} P_k \tilde{g}^{(k)}$$

$$P_{k+1} \leftarrow \frac{1}{2} P_k$$

This corresponds to a simple dichotomy, of which the ellipsoid method is the n-dimensional generalization.

3 | Integration of CACAO into ASTERIA

The second part of the internship involved integrating CACAO into ASTERIA, a flight-tested algorithm capable of predictive AOC with integrated collision avoidance. The aim was to replace the existing method of calculating avoidance maneuvers, since the one already implemented was mono-risk, had no imposed SK constraints and poorly minimized fuel costs.

First, the general principle of ASTERIA will be explained, focusing on risk management (part 3.1), then the modifications to the two algorithms that had to be made to make this integration work will be presented (part 3.2).

3.1. Presentation of ASTERIA

3.1.1. Global architecture

ASTERIA is made up of a simulator, which propagates the satellite over a certain period of time, taking into account the presence of maneuvers, and an AOC, which is activated at each ascending node and whose aim is to create a maneuver plan, i.e. to predict over a certain time horizon the thrusts that the satellite must perform to meet the objectives [10]. These objectives may be to avoid straying too far from the reference orbit, or to ensure that the risk of collision is mitigated below a certain threshold. Considering the i^{th} AOC activation, i.e. the i^{th} ascending node since the start of the simulation, ASTERIA is structured as follows :

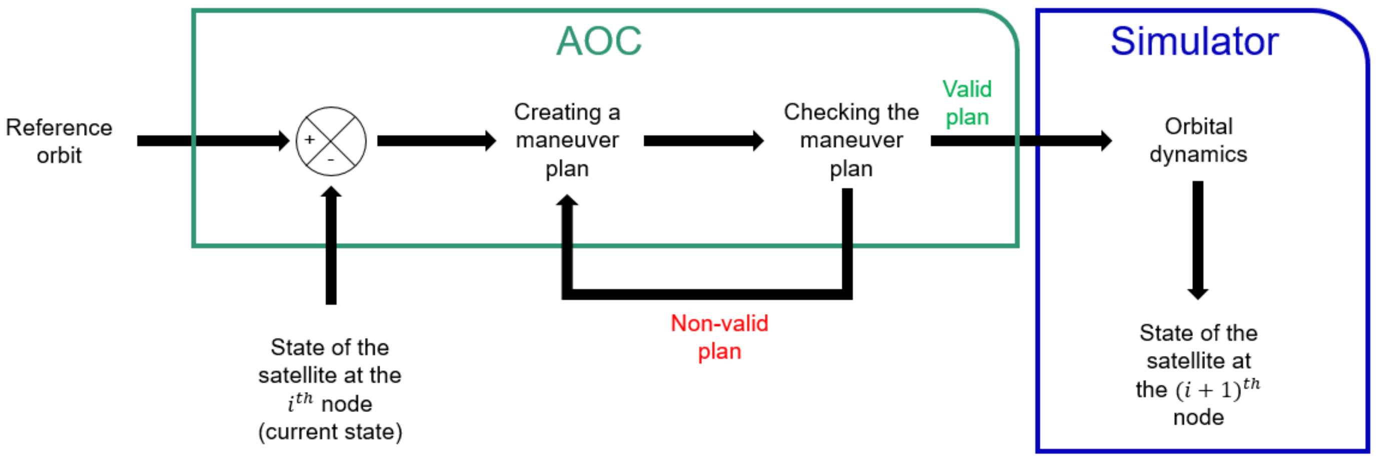

AOC
Simulator
Reference orbit
+
-
Creating a maneuver plan
Checking the maneuver plan
Valid plan
Orbital dynamics
State of the satellite at the i^{th} node (current state)
Non-valid plan
State of the satellite at the $(i+1)^{th}$ node

Figure 3.1: i^{th} activation of the AOC

1. **Calculating deviations** : At each activation, the AOC compares the reference orbit with the satellite's current state. The orbital parameters used are :

$$\begin{pmatrix} a \\ e_x = e\cos(\omega) \\ e_y = e\sin(\omega) \\ i \\ \Omega \\ \alpha = \omega + \nu \end{pmatrix}$$

   The state of the satellite at the node i is defined by the deviations Δe_x^i, Δe_y^i, $\Delta\Omega^i$ and $\Delta\alpha^i$, which measure the difference between the actual parameters and those of the reference. An threshold overrun occurs when one of these deviations exceeds a defined value, characterizing too great a distance from the reference. From the states of previous nodes stored in memory, the algorithm also estimates the derivative of these deviations, assuming the evolution of the parameters Δe_x, Δe_y and $\Delta\Omega$ to be linear, and that of $\Delta\alpha$ to be parabolic. These derivatives are used to propagate these parameters analytically.

2. **Creating the maneuver plan** : It's important to point out that ASTERIA is a predictive AOC, meaning that the maneuver plan is fixed over a certain future time horizon. This means that each time it is activated, the satellite knows which maneuvers it will carry out in the coming orbits, and these cannot be modified. The primary purpose of this "frozen" horizon is to better anticipate any risk of collision, since the satellite's trajectory is known until the end of the horizon.

   The consequence of a frozen horizon that is too small is that the satellite cannot predict its trajectory far into the future, meaning that a collision may be detected shortly before the TCA.

However, a fixed horizon that is too large has a negative impact on SK, as requirements are likely to change significantly between the calculation of the maneuver and its execution, particularly in a context of high solar activity. This leads to thrusts that are poorly-adapted to the real SK needs which leads to threshold overruns.

The solution adopted is to introduce a so-called "semi-frozen" horizon, in which the maneuvers already planned are modified in magnitude (but remain on the same date and in the same direction) each time the AOC is activated, to allow greater adaptability of the system. All this is summarized in the following figure :

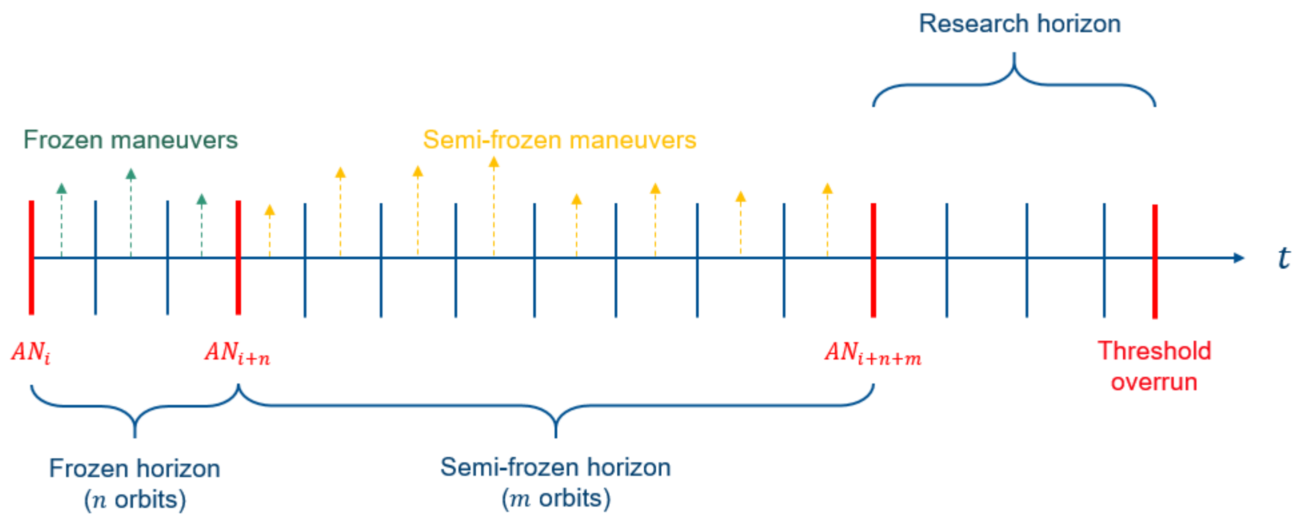


Figure 3.2: Predictive AOC

Therefore, to build its maneuver plan, ASTERIA will start by analytically propagating the deviation parameters introduced in step 1, taking maneuvers into account right up to the end of the frozen horizon. This analytical propagation then continues up to each maneuver scheduled in the semi-frozen horizon to check whether an threshold overrun occurs on that date. In this case, the magnitude of the maneuver is updated to prevent it, otherwise it remains unchanged.

Once the end of the semi-frozen horizon has been reached, this analytical propagation continues until either the end of the propagation horizon is reached, or an overrun is detected before that. In the latter case, the AOC searches for a slot in the research horizon between the end of the semi-frozen horizon and the date of the overrun. A maneuver, either in semi-major axis or in inclination depending on the type of threshold overrun, is then added to the plan.

3. **Checking the maneuver plan** : Once the maneuver plan has been decided, it is checked by CROCO, a risk management algorithm integrated into ASTERIA. At the start of the simulation, the AOC receives a list of CDMs. In a real-life situation, this list would be updated throughout the life of the satellite, and new CDMs would be regularly transmitted from the ground. The CDMs handled here differ from those used by CACAO in that they do not include information

on the primary, only on the secondary.

Each time CROCO is called, it will propagate the primary from the current state, as well as each secondary contained in the CDM list, and will check the PoC (and COPoC) evolution for each pair over a time horizon set to be larger than the semi-frozen horizon. If one or more dangerous encounters (assessed as such because the COPoC exceeds a given threshold) are detected within this horizon, the maneuver plan is considered invalid. CROCO returns the CDMs associated with these encounters, and ASTERIA exits its nominal mode to update its plan until it is valid. The different modes associated with risk management are detailed in the section 3.1.2.

4. **Simulation** : If no risk is detected, the AOC activation ends, and the simulator integrated into ASTERIA will then propagate the satellite state, taking into account any maneuvers contained in the time interval between the ascending nodes i and $i+1$. The state calculated at the end of this propagation will serve as input data for the next AOC activation. If the algorithm were actually integrated on board a satellite, this simulator would be useless, as the current state would be calculated by GNSS measurements at each node.

3.1.2. Avoidance modes

ASTERIA's AOC has different modes which define how the algorithm will create its maneuver plan and with what objective. The nominal SK mode, presented in the previous section, corresponds to the so-called **Free** mode. If CROCO detects a risk, the sequence of modes is described in the following figure :

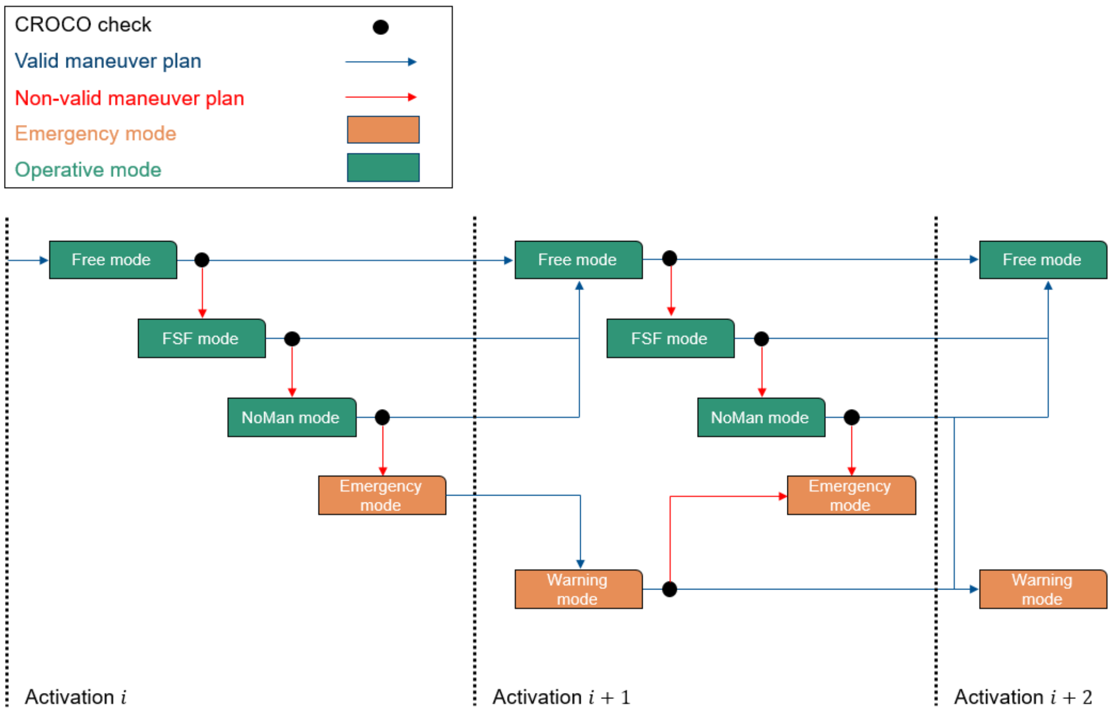


Figure 3.3: ASTERIA modes

As long as no risk is detected, the AOC remains in nominal mode. Otherwise, the maneuvering plan must be modified. A new risk can arise for three reasons :

- CROCO detects a conjunction because a new CDM has just been transmitted to the AOC. This could not happen with the examples performed during the internship, as new CDMs were not communicated to the satellite during the simulation.
- An already known risk enters the risk search horizon, i.e. the TCA is now close enough to the current date for the AOC to be impacted.
- The COPoC of an encounter already contained in the horizon exceeds the risk threshold due to a change in the planned maneuvers, impacting the future state of the primary.

The latter case is easier to deal with, and does not require dedicated avoidance maneuvers. For this reason, the first mode that attempts to reduce the risk is the so-called **FSF** or Frozen Semi-Frozen mode. This mode seeks to modify the plan by adjusting the semi-frozen maneuvers in magnitude so that the COPoC falls below the threshold while avoiding an overrun. This is a way of minimizing the impact of the avoidance on the mission, which still remains the priority.

If this modified plan is not sufficient to reduce the risk, the next mode is **NoMan**. All semi-frozen maneuvers are canceled, and the priority is now on avoidance. This mode simply verifies that this plan without any maneuvers is not sufficient to reduce the risk, which would be the most fuel-efficient method.

If the COPoC remains above the threshold, the **Emergency** mode is used. This is the last resort. ASTERIA will then try to calculate an avoidance maneuver plan. The details of this maneuver calculation are described in the next section. This is a mono-risk management of avoidance: if several risks are detected, only the first encounter (the earliest TCA) is taken into account. Once this plan has been decided, ASTERIA does not carry out a CROCO check, and the plan is assumed to be valid.

On the next activation of the AOC, after the Emergency mode, the algorithm starts in **Warning** mode. This is a hybrid mode, the aim of which is both to ensure that the previously calculated plan is still sufficient to reduce the risk, and also to plan the post-TCA maneuvers that will enable the satellite to return to its reference orbit. As long as CROCO detects no new risks, the AOC will remain in Warning mode on each activation until the date defined as the date of exit from this mode is included in the frozen horizon. Otherwise, the AOC returns to Emergency mode and computes a new avoidance maneuver plan. The definition of this exit date is a point of improvement for ASTERIA, mentioned in section5.2.1.

3.1.3. Avoidance maneuvers calculation

The method initially implemented in ASTERIA to establish the avoidance maneuver plan is a mono-risk analytical method. The avoidance criterion is calculated by CROCO, it is the minimum radial shift required between the primary before and after maneuvers at TCA for the COPoC to fall below the threshold. This method does not propagate the secondary, but simply attempts to maneuver the satellite until this radial shift is verified.

The method will first consider all the maneuver slots contained between the end of the frozen horizon and the TCA. They are then ranked, an advantageous slot being one that is as late as possible and whose argument of latitude is close to the optimal one, opposite to the risk. The algorithm will then go through this list, starting with the most interesting slot, and gradually increment the ΔV in the tangential direction until the radial shift between the position of the primary before and after maneuvers at the TCA is verified by analytical propagation. If the value of ΔV reaches its maximum for this slot, the algorithm moves on to the next without canceling the maneuvers already planned.

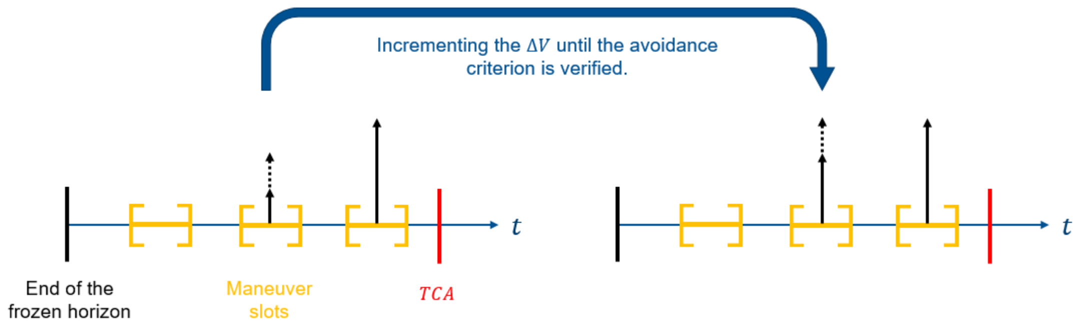

Incrementing the ΔV until the avoidance criterion is verified.
t
t
End of the frozen horizon
Maneuver slots
TCA

Figure 3.4: Avoidance method of ASTERIA

This method has several shortcomings. In addition to being a mono-risk method, it does not guarantee that the satellite will remain close to its reference orbit. In addition, it only seeks to increment the ΔV until the risk reduction is satisfactory, but does not seek to intelligently optimize the total fuel cost of maneuvers. The main aim of integrating CACAO was to replace this method.

3.2. Integration of CACAO

The aim of the second half of the internship was to integrate CACAO into ASTERIA to replace the already implemented method for calculating avoidance maneuvers. This involved several modifications to both algorithms, resulting in improved risk management on the part of ASTERIA. However, several improvements can still be made and are mentioned in chapter 5.

3.2.1. Modifications of ASTERIA

No fundamental changes have been made to the algorithm's architecture, the aim being only to replace the analytical method presented in the previous section with a call to CACAO.

Multi-risk situations management

As previously mentioned, ASTERIA managed collision risks one by one, the avoidance method being mono-risk. With the integration of CACAO, this risk management has been modified to retain all the

TCAs detected by CROCO in the risk search horizon, and not just the first one.

This ability of the algorithm to handle multi-risk situations has necessitated a slight modification to the sequence of modes presented above. To take an example, two risks are detected in the search horizon at dates t^1_{TCA} and t^2_{TCA}, and when exiting the Emergency mode, a maneuver plan is decided. At the next AOC activation, if the risk linked to date t^2_{TCA} is still considered dangerous, ASTERIA erases its maneuver plan and returns to Emergency mode, but does not consider the first risk in this new plan, only the second.

To avoid this problem, the AOC switches to the NoMan mode after exiting the Warning mode if the CROCO check fails. The new mode sequence thus becomes :

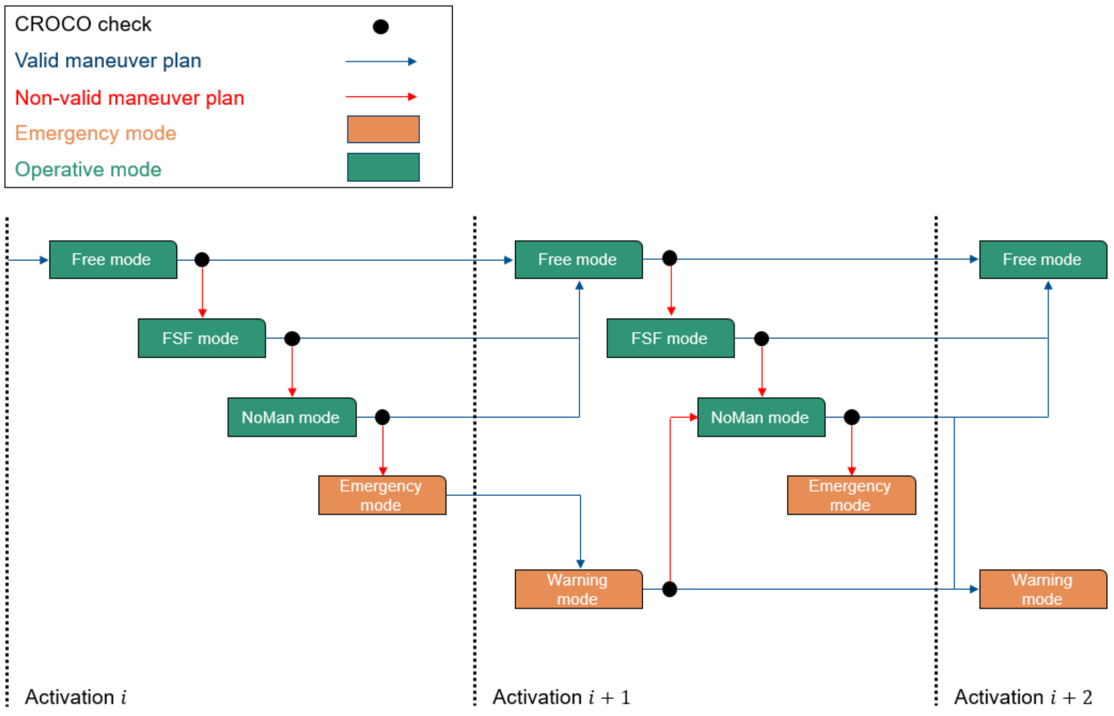


Figure 3.5: ASTERIA modes after modifications

Depending on the activation mode, the AOC defines a research horizon within which it builds the maneuver plan. For Emergency mode, this horizon is contained between the end of the frozen horizon and the TCA. For Warning mode, it is contained between the TCA and the end of the propagation horizon for adding SK maneuvers. If a CROCO check fails, the latest TCA is now identified and used to define these horizons.

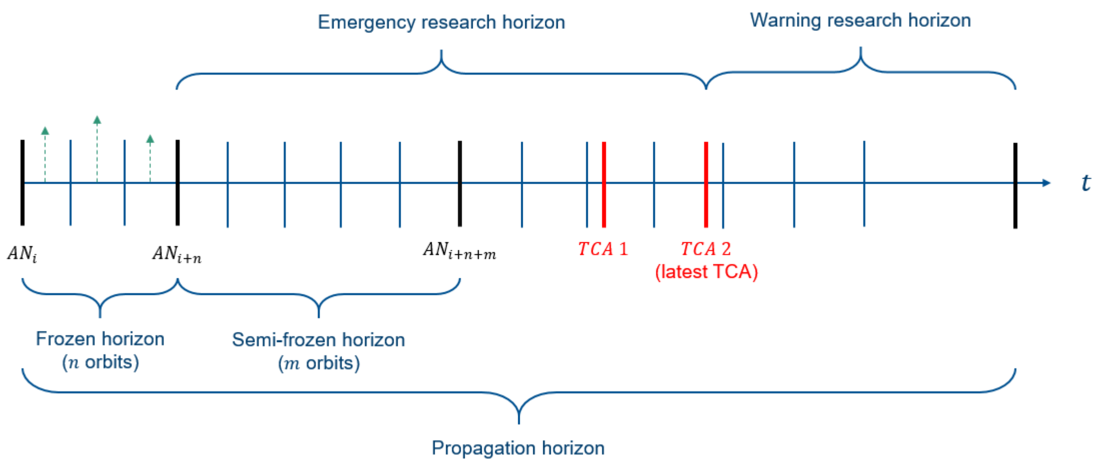


Figure 3.6: Research horizons in ASTERIA after modifications

Additional input data

ASTERIA retrieves all its input data in .properties files, which provide the necessary information about the satellite and the simulation.

A new file containing the data required to run CACAO is now necessary to run a simulation. This data contains, among other things :

- The Euclidean or Mahalanobis avoidance criterion, which remains the same whatever the risk.
- The SK box dimensions.
- The minimum volume of ellipsoids.
- The number of slots per avoidance strategy.
- The coefficients k_1, k_2 and k_3 used to rate the slots.

The fact that some of these data, in particular the avoidance criterion and the number of slots per strategy, are not adapted to each risk but remain constant during the simulation is an point of improvement in this integration.

Radial shift and COPoC computation

For each potentially dangerous encounter, CROCO calculates the PoC and COPoC, as well as the radial shift defined above and used as an avoidance criterion for the analytical method.

The possible values for k_p and k_s, the coefficients used to calculate the COPoC, were hard-coded. These values are now input data and have been modified, with the range for k_s being changed from [0.1, 10] to [0.25, 4] to prevent an encounter with a sufficiently large avoidance distance from being deemed dangerous due to too great an expansion of the secondary's covariance. These values are extracted from CROCO and given as input to CACAO to calculate the Mahalanobis distance.

A problem with the calculation of the radial shift has been corrected in ASTERIA. This value corresponds to the minimum radial shift between the primary before and after maneuvers so that the

COPoC falls below the threshold. However, it was calculated in relation to the PoC, which led to an underestimation of this shift and therefore to an avoidance criterion that was too low. This value is now measured in relation to the COPoC.

Although CACAO doesn't use this value, calculating it is still useful because the previous avoidance method can still be called up if CACAO doesn't converge. Indeed, the main advantage of this method is that it ensures that even if no "valid" maneuver plan (valid in the sense that it ensures the minimum required deviation) is found, the satellite will still perform thrusts. For this reason, it's worth keeping this method as a backup solution.

3.2.2. Modifications of CACAO

Similarly, CACAO's architecture was not altered in this integration, and the only modifications made were to ensure consistency between the data transmitted by ASTERIA and those manipulated by CACAO.

As mentioned above, the CDMs handled by ASTERIA are "pseudo-CDMs". They contain only information on the state and covariance of the secondary at the TCA. The state of the primary is obtained by propagating the current state, and its covariance is calculated in relation to the uncertainties of the maneuvers to be performed during this propagation. ASTERIA transmits a list of CDMs and the state of the primary at the initial date.

One of the problems of CACAO's initial state highlighted by the management of multi-risk situations was the consistency of the propagator with that used to generate the CDMs (see section 2.3.2).

In the context of this integration, this problem has been solved differently, as no details of the propagator are given in the CDMs transmitted by ASTERIA. CACAO is therefore built to use the same propagator as CROCO, i.e. the propagator used to generate the CDMs read by the algorithm. This ensures perfect consistency between the data transmitted by ASTERIA and those manipulated by CACAO.

3.2.3. Modified Emergency mode

At the start of a simulation, a *Cacao* class object is initialized with all constants related to the avoidance, such as the dimensions of the SK box or the avoidance criterion. If one or more risks are detected by CROCO and the FSF and NoMan modes fail to find a solution, a method called *AvoidanceSolver* of the *Cacao* class is called. This method takes as input the CDMs generated by CROCO, the list of authorized slots as well as the $\Delta V_{\min}$ and $\Delta V_{\max}$ associated with these slots, and finally the state and covariance of the primary at the initial date corresponding to the end of the frozen horizon.

CACAO then scores these slots and calls up the B&B algorithm until it converges. If the resulting solution is validated in terms of compliance with the constraints, the date and ΔV of the thrusts to be made is sent back to the AOC, which will incorporate them into its maneuver plan.

If none of the strategies built by CACAO converge (or take too long to converge) and the list returned is empty, ASTERIA will call up the analytical avoidance method to build its maneuver plan.

4 | Validation of the algorithms

This chapter is devoted to validating CACAO after the addition of the modifications mentioned in section 2.3, and then the integration of this algorithm into ASTERIA. The first step in this validation was to test CACAO on real examples, using as input the CDMs supplied by the CAESAR team (section 4.1). A second step consisted in cross-validating these results with those provided by JAC, a tool for calculating avoidance maneuvers used by the CNES Space Surveillance department (section 4.2). Finally, the results of ASTERIA's simulations carried out with CACAO integrated are presented (section 4.3).

4.1. CACAO tests on real conjunctions

The algorithm had already been tested on real cases [9], but modifications made during the first half of the internship meant that these tests had to be repeated. In addition, new CDMs corresponding to multi-risk avoidance situations, presented in part 2.3.2, were provided by the CAESAR team at the start of the internship.

4.1.1. Conjunction data

The examples covered correspond to real-life avoidance situations that required maneuvering. The various examples are presented in the following table :

Example	Primary	Secondary	TCA	Mono / Multi-risk	Miss distance [m]	Relative velocity [m/s]	Altitude [km]
01	SMOS	CBERS-1-DEB (debris)	2021-07-28T16:17	Mono-risk	87	13700	$\sim$ 750
02	SMOS	IRR-33-DEB (debris)	2021-03-04T13:00	Mono-risk	44	14931	$\sim$ 760
03	ICEYE-X2	CAPELLA-5	2021-09-22T14:37	Multi-risk	5613	5	$\sim$ 520
			2021-09-22T15:15		35	6	
			2021-09-22T16:00		2314	5	
04	Observation satellite	IRR-33-DEB (debris)	2021-05-27T08:58	Mono-risk	95	14786	$\sim$ 680
05	Observation satellite	LightSail	2021-02-22T13:08	Mono-risk	514	13095	$\sim$ 710
06	Galileo-23	Ariane 44LP RB (debris)	2021-03-07T02:48	Mono-risk	1004	3922	$\sim$ 23200
07	SMOS	Thoras Agena D DEB (debris)	2022-05-20T20:43	Mono-risk	57	4200	$\sim$ 760
08	GlobalStar M070	Cosmos 2328	2022-09-09T19:01	Mono-risk	83	12700	$\sim$ 1400
09	Inmarsat 4F3	GOES 6	2022-03-17T16:12	Mono-risk	5330	570	$\sim$ 35800
10	ICEYE-X13	Debris	2021-08-29T21:35	Multi-risk	4268	4	$\sim$ 550
			2021-08-29T23:04		1401	5	
			2021-08-29T23:52		11	4	

Table 4.1: Real conjunction data handled with CACAO

A total of 10 examples were covered: 8 mono-risk and 2 multi-risk. Most are in LEO, but example 06 is in MEO and example 09 in GEO. Although the AOC technology is designed for low-earth orbit satellites, it was important to check that the algorithm works in other cases. Some encounters, particularly the multi-risk situations, cannot be quantitatively characterized as short-term encounters, as the relative velocity is too low. It was also important to deal with such cases to observe the limits of the short-term encounter assumptions used to solve the optimization problem.

4.1.2. Results

This section presents examples of the results obtained by CACAO. The aim is to highlight the impact of the modifications made to the algorithm.

Control points study

The first study concerns control points, a modification made to the algorithm to ensure that the primary remains in its SK box throughout the avoidance time interval.

The first example treated is example 02, a conjunction between SMOS and an Iridium-33 debris dating from March 2021, and helps to justify the choice of control points defined above. The algorithm must therefore optimize the avoidance under the following constraints :

- The primary can only maneuver in two slots located 2.5 and 1.5 orbits before the TCA. Thrusts are tangential, and the value of the ΔV for each maneuver must be kept between 0 and 0.1 m/s.
- The avoidance criterion is a Mahalanobis distance with a value of 10. The coefficients k_p and k_s used to evaluate this distance are both unitary.
- The primary is initially at the center of its SK box, and its dimensions in the QSW frame are 1, 12 and 10 km respectively.

The algorithm's solution is shown in the following table :

Relative position primary/secondary before maneuvers				
Mahalanobis distance [-]	5.88			
Miss distance [m]	44.36			
Miss distance along Q [m]	16.81			
Miss distance along S [m]	1.79			
Miss distance along W [m]	-41.02			
Maneuvers				
Man 1 T [m/s] : [tca - 2.50 orb]	0.0005			
Man 2 T [m/s] : [tca - 1.50 orb]	0.0180			
Total ΔV [m/s]	0.0185			
Relative position primary/secondary after maneuvers				
Mahalanobis distance [-]	10.00			
Miss distance [m]	89.11			
Miss distance along Q [m]	87.98			
Miss distance along S [m]	-1.13			
Miss distance along W [m]	-14.05			
Relative position primary/reference after maneuvers at each control point				
Control point	TCA	0.5 orbit after Man 1	0.5 orbit after Man 2	0.5 orbit after TCA
Relative position along Q [m]	69.8	1.9	70.3	0.3
Relative position along S [m]	-507.3	-4.5	-175.2	-672.5
Relative position along W [m]	-0.1	0.0	0.0	0.1
Computational time				
B&B runtime [s]	2.336			
Total runtime [s]	14.220			

Table 4.2: Results of example 02 - Mahalanobis distance of 10

We can thus observe that the solution found respects all constraints. All computational times given in this report are averaged over 5 runs. The algorithm also displays the deviation of the primary from its reference as a function of time along the Q, S and W axes :

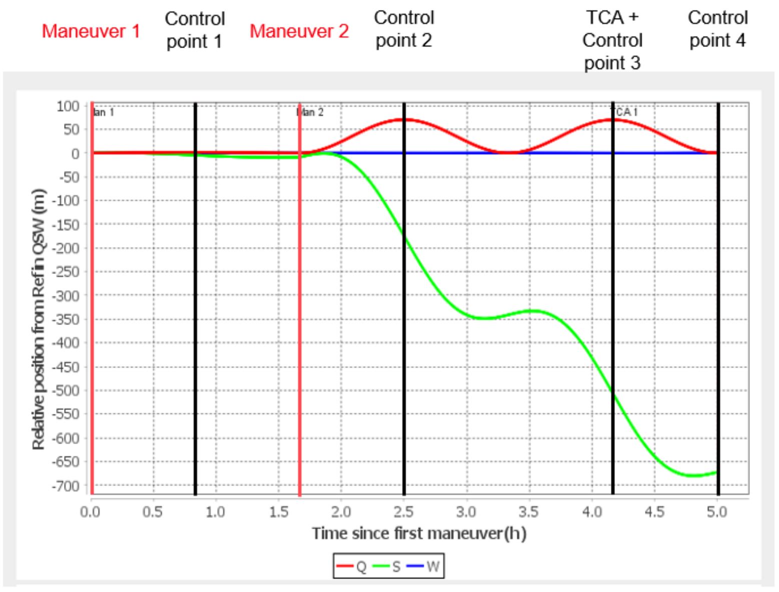


Figure 4.1: Control points of example 02

We can see that the satellite is initially at the center of its box, and then performs two maneuvers, indicated by vertical red lines. The first maneuver is very small, with almost no change in relative distances, but the second is a strong thrust in the tangential direction, creating a radial deviation and a phase shift that accumulates over time.

The control points are indicated by vertical black lines, and in this configuration there are 4 of them. They can thus be observed half an orbit after each maneuver, at the TCA and then half an orbit after the TCA. The first 2 are placed at dates when the deviation according to Q is at its maximum, and the last corresponds to a maximum for the deviation according to S.

To measure the impact of this modification on computational time, 3 other examples were treated with and without the control points. These examples and their results are presented below :

Example	01	09	10
Number of TCAs	1	1	2
Number of maneuvers	1	2	3
Avoidance criterion	Euclidean : 500 m	Mahalanobis : 15	Euclidean : 500 and 1600 m
Number of control points before modifications	1	1	2
Computational time before modifications [s]	6.753	15.192	13.780
Number of control points after modifications	3	6	6
Computational time after modifications [s]	8.243	27.040	16.595

Table 4.3: Impact of the control points on the computational time - examples 01, 09 and 10

The addition of control points is a necessary modification to ensure compliance with the SK box constraint, particularly after the last TCA. However, this modification increases the computational time by at most a factor of 2, which is due to all the additional propagations these checks require. Currently, each propagation to a control point is done from the initial state, which means that some calculations have to be repeated. This implementation can therefore be further improved to minimize the time spent propagating to these points.

Number of maneuvers study

A second study was carried out in relation to the number of maneuvers. As it stood at the start of the internship, strategies with a single slot did not work, and those with more than two did not converge after 15 minutes of calculation. After corrections, these computational times were reduced, but the number of maneuvers remains a determining factor whose impact has been studied with the following example.

The example treated here is the example 03, already presented in section 2.3.2, which corresponds to a succession of conjunctions between the ICEYE-X2 and CAPELLA-5 satellites in September 2021. The avoidance criterion is Euclidean this time, and is worth 5000, 500 and 2500 m respectively in the chronological order of the 3 encounters. The minimum and maximum values for thrust and the dimensions of the SK box remain the same as in the previous example. To study the impact of the number of maneuvers, different strategies are tested, the results of which are presented in the following table :

Miss distance before maneuvers [m]				
CDM 1	5610.78			
CDM 2	35.41			
CDM 3	2317.40			
Maneuvers				
Number of maneuvers	1	2	3	4
Man 1 T [m/s] : [tca - 0.50 orb]	0.0581	0.0209	0.0117	0.0093
Man 2 T [m/s] : [tca - 1.50 orb]	-	0.0001	0.0018	0.0015
Man 3 T [m/s] : [tca - 2.50 orb]	-	-	0.0003	0.0010
Man 4 T [m/s] : [tca - 3.50 orb]	-	-	-	0.0014
Total ΔV [m/s]	0.0581	0.0210	0.0138	0.0120
Miss distance after maneuvers [m]				
CDM 1	5636.67	5363.69	5310.43	5256.58
CDM 2	503.27	502.00	508.25	568.57
CDM 3	3005.66	2793.47	2755.06	2788.66
Computational time				
B&B runtime [s]	1.754	2.100	51.352	881.837
Total runtime [s]	9.631	11.948	65.877	900.322

Table 4.4: Results of example 03 - Euclidean distances of 5000, 500 and 2500 m

As the number of maneuvers increases, so does the computational time. Indeed, increasing the number of slots is linked to the dimension and hence complexity of the optimization problem, but also impacts the number of propagations required by the Java overlay. The latter must propagate the primary to the various control points in order to construct the optimization problem and check the validity of the output solution. A strategy including 5 slots was tested, but failed to converge after 30 minutes of computation. This divergence can be seen in the following graph :

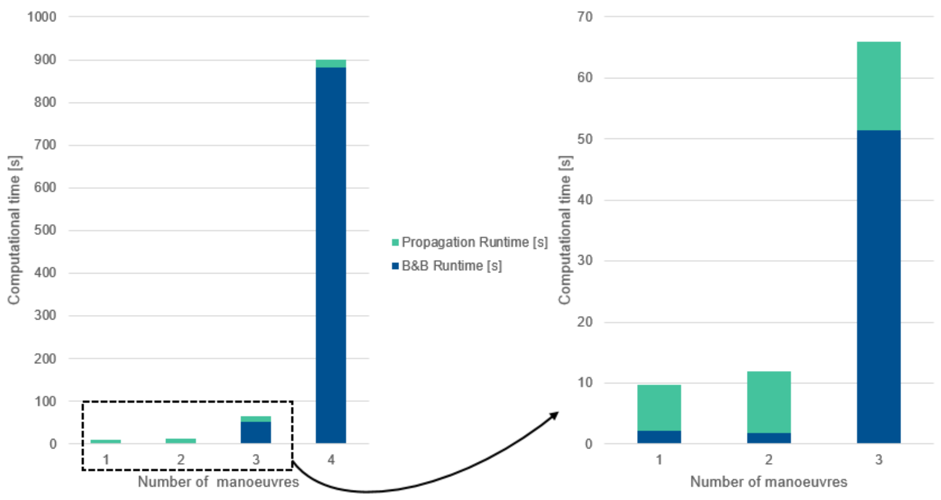


Figure 4.2: Example 03 - Computational time with respect to the number of maneuvers

In the current state of the algorithm, it is therefore recommended to use only strategies involving one or two slots, or to modify the value of other determining factors such as the minimum volume of ellipsoids if a greater number of maneuvers is required for the avoidance.

Minimum volume of ellipsoids study

The minimum volume of ellipsoids is an input parameter that directly impacts the number of ellipsoids visited by the B&B algorithm, and thus the computational time required to optimize and minimize the ΔV of the solution. It's important to note that modifying this value has no impact on compliance with the constraints and therefore the validity of the solution.

The first example treated here is the example 05, corresponding to a conjunction between an observation satellite and the LightSail satellite in February 2021. The imposed strategy consists of a tangential thrust half an orbit before the TCA, with the value of the ΔV bounded between 0 and 0.2 m/s. The avoidance criterion is Euclidean, with a value of 1 km.

The aim of this study is to vary the minimum volume between 10^{-6} and 10^{-1} and observe the effect on the computational time and the ΔV found by the B&B. The results are presented in the following graphs :

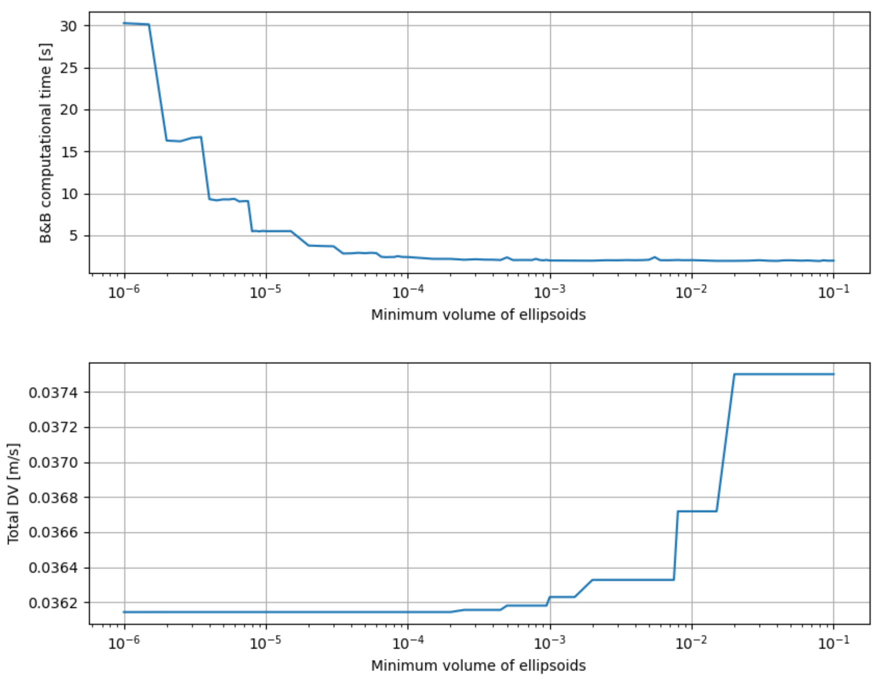


Figure 4.3: Example 05 - Computational time and total ΔV w.r.t. the minimum volume of the ellipsoids

As one might expect, increasing this minimum volume leads to a reduction in computational time and an increase in the ΔV of the solution. These variations occur in fits and starts, corresponding to a change in the best solution found and therefore in the number of iterations of the B&B. However, between the two extremes, the computational time is divided by a factor of 15, while the relative difference between the two solutions is only 3.8 %. Since the convergence of the algorithm is a more important criterion than minimizing the fuel cost of the avoidance, it is recommended to use a minimum volume between 10^{-3} and 10^{-2}.

A second example was treated, this time using a two-slot strategy to study the impact of the minimum volume value when the problem dimension changes. The example studied is example 07, which corresponds to a conjunction between SMOS and a debris from Thoras Agena in May 2022.

The imposed strategy consisted of two tangential maneuvers located 1.5 and 0.5 orbits before the TCA. The results are shown in the following graphs :

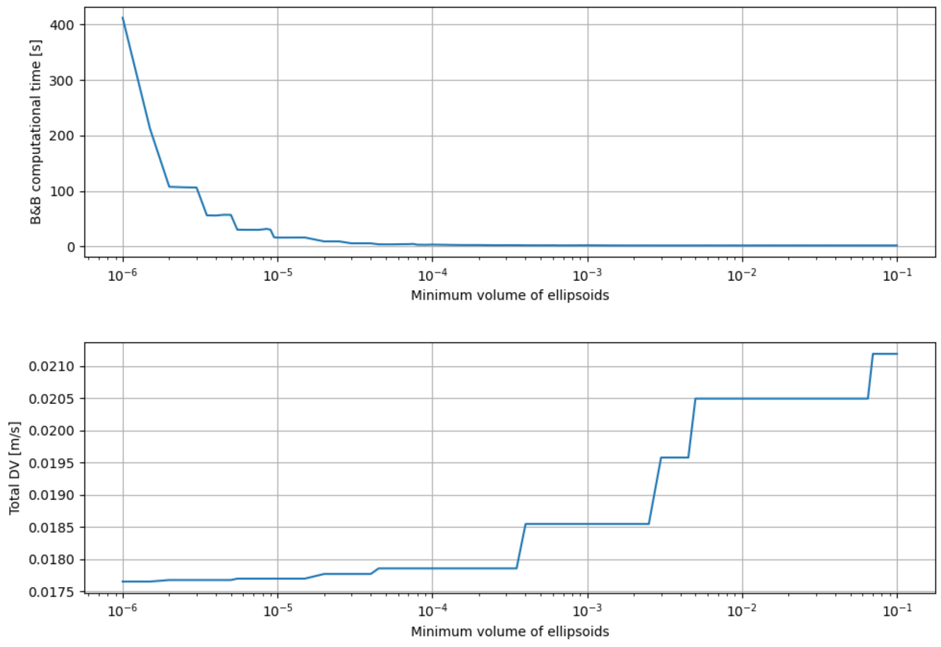


Figure 4.4: Example 07 - Computational time and total ΔV w.r.t. the minimum volume of the ellipsoids

Qualitatively, the impact of this value is similar to the previous case. However, the variations are much greater. Between the two extremes, calculation time is divided by a factor of 220, while the relative difference between the two solutions is 20.1 %. Choosing the right minimum volume is even more crucial as the number of maneuvers increases, to avoid a rapid increase in computational time or a degradation of the solution.

4.2. Validation with JAC

To validate the CACAO results, they were compared with those of JAC, an operational tool used by CAESAR.

4.2.1. Presentation of JAC

JAC or *Java for Assessment of Conjunction* is a CDM management and reading tool capable of analyzing risks and estimating the necessary avoidance maneuvers. The latter is an interesting feature, since the avoidance criterion given as input can be a PoC or a distance, which enables the maneuvers estimated by JAC to be compared with those calculated by CACAO.

Figure 4.5: Logo of JAC

Visualization of required maneuvers

JAC works as follows : CDMs are given as input and the software extracts the relevant information from each of these files, such as the TCA, PoC, miss distance or state vectors of the two objects. An estimate of the tangential maneuver required can then be requested, indicating the nature of the avoidance criterion, the maximum value of the ΔV and dates, for example those associated with the argument of latitude opposite to the risk preceding the TCA.

The results are then presented in the form of a table showing the value of the criterion (PoC or miss distance) as a function of the date of the maneuver and the value of the ΔV.

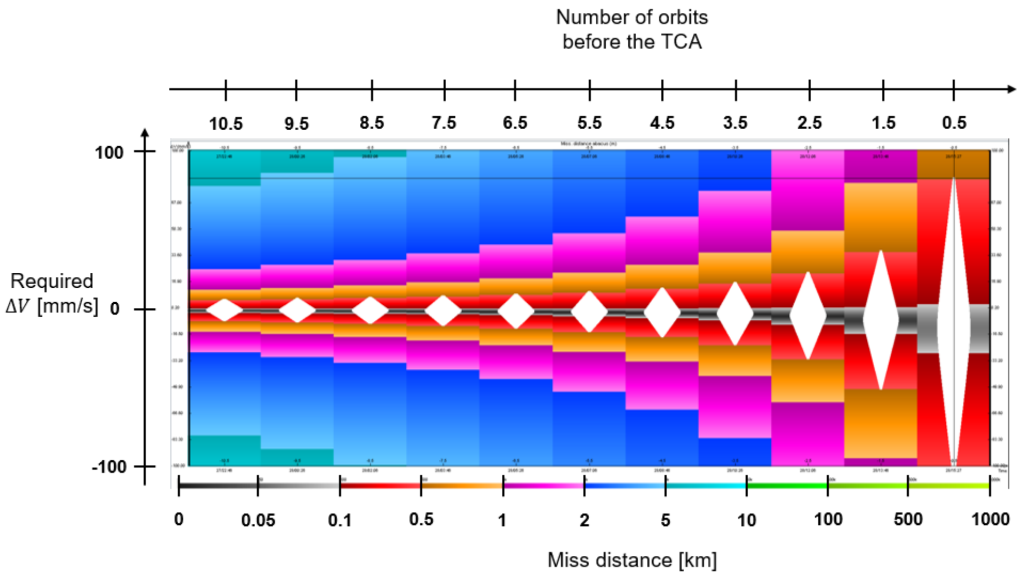


Figure 4.6: Required maneuvers recommended by JAC - example 01

In each column, which corresponds to a date preceding the TCA, the color code indicates the miss distance obtained by thrusting in the tangential direction with a certain ΔV.

Number of sigma

As an avoidance criterion, JAC lets one select an Euclidean distance or a distance related to the combined covariance of the two objects, the number of sigma, noted NoS. This dimensionless value, similar but not equal to the Mahalanobis distance, is calculated as follows :

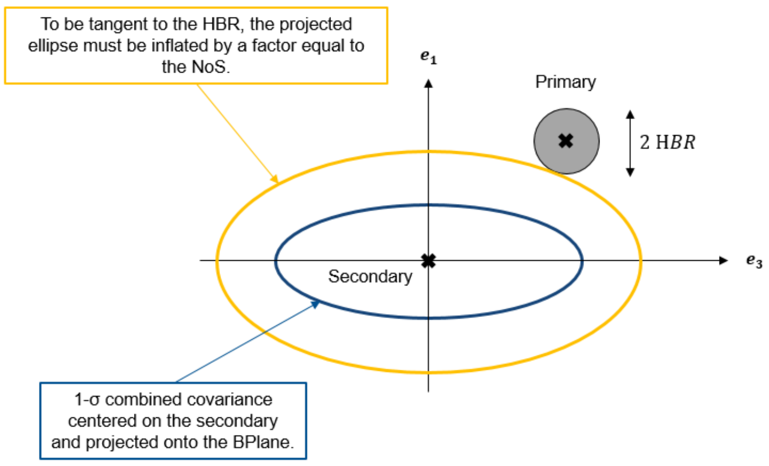

To be tangent to the HBR, the projected ellipse must be inflated by a factor equal to the NoS.
e_1
Primary
$2\,HBR$
e_3
Secondary
1-σ combined covariance centered on the secondary and projected onto the BPlane.

Figure 4.7: Definition of the NoS criterion

This value thus corresponds to the dilation factor of the ellipse, defined by the 1-σ combined covariance Σ projected onto the BPlane, so that it is tangent to the sphere of radius HBR centered on the primary. Although comparable to the Mahalanobis distance, this value is based on a 2D projection of the covariance onto the collision plane, and takes into account the HBR.

4.2.2. Results comparison

Example 01

The figure 4.6 corresponds to example 01, i.e. a conjunction between SMOS and space debris in July 2021. This is a typical geometry where the thrust required for avoidance decreases as the date of the maneuver moves away from the TCA since a phase shift will accumulate. Below we compare the ΔV calculated by JAC and CACAO to obtain the same avoidance criterion, as well as the ΔTCA, i.e. the difference between the TCA before and after the maneuver.

Avoidance criterion	**Date of maneuver**	**ΔV of JAC [cm/s]**	**ΔV of CACAO [cm/s]**	**ΔTCA of JAC [ms]**	**ΔTCA of CACAO [ms]**
Euclidean : 500 m	0.5 orbits before the TCA	8.23	8.24	49.3	49.2
Euclidean : 500 m	1.5 orbits before the TCA	3.62	3.62	65.2	65.2
Euclidean : 500 m	2.5 orbits before the TCA	2.26	2.26	67.9	67.4
Mahalanobis : 10 NoS : 10	0.5 orbits before the TCA	2.43	1.24	14.5	7.4
Mahalanobis : 10 NoS : 10	1.5 orbits before the TCA	1.72	0.45	31.0	8.1
Mahalanobis : 10 NoS : 10	2.5 orbits before the TCA	1.18	0.27	35.2	7.9

Table 4.5: Comparison of results from JAC and CACAO - example 01

This table shows a correspondence of the order of mm/s between JAC and CACAO when the avoidance criterion is Euclidean. However, the difference in definition between the NoS and the Mahalanobis distance makes it impossible to quantitatively compare the maneuvers calculated in relation to these criteria.

Example 02

A second case was treated, example 02, already presented in section 4.1.2. Below is reported the figure displayed by JAC when the avoidance criterion is in NoS :

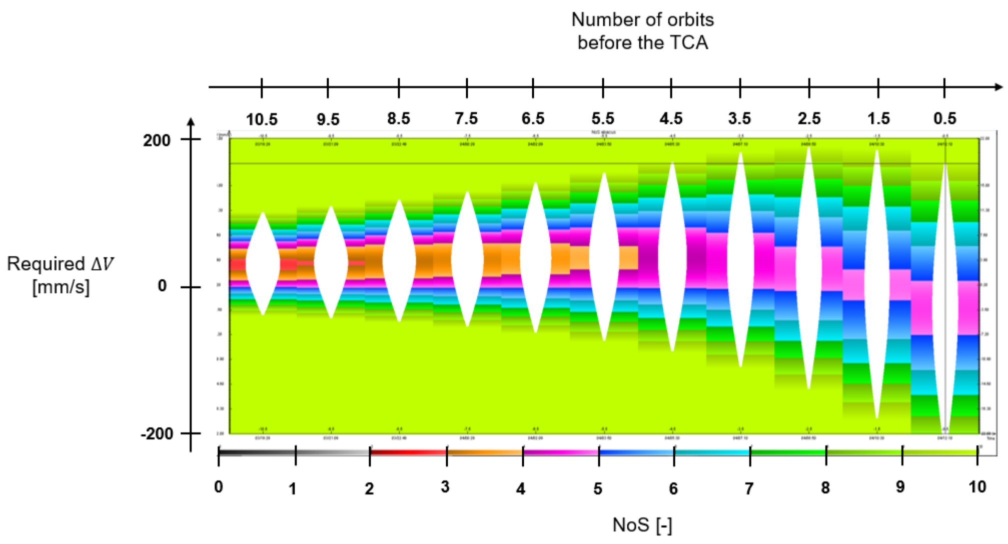


Figure 4.8: Required maneuvers recommended by JAC - example 02

This case has a particular encounter geometry, as we can observe a maximum for the ΔV required for the avoidance at 2.5 orbits before the TCA when this thrust is tangential in the positive direction and the avoidance criterion is in NoS.

Avoidance criterion	**Date of maneuver**	**ΔV of JAC [cm/s]**	**ΔV of CACAO [cm/s]**	**ΔTCA of JAC [ms]**	**ΔTCA of CACAO [ms]**
Euclidean : 500 m	0.5 orbits before the TCA	12.62	12.62	75.5	75.9
Euclidean : 500 m	1.5 orbits before the TCA	12.12	12.11	218.8	218.4
Euclidean : 500 m	2.5 orbits before the TCA	11.16	11.15	335.5	335.1
Euclidean : 500 m	3.5 orbits before the TCA	10.05	10.04	422.0	422.4
Mahalanobis : 10 NoS : 10	0.5 orbits before the TCA	1.85	1.61	10.8	9.7
Mahalanobis : 10 NoS : 10	1.5 orbits before the TCA	2.05	1.84	36.9	33.3
Mahalanobis : 10 NoS : 10	2.5 orbits before the TCA	2.08	1.92	62.7	57.9
Mahalanobis : 10 NoS : 10	3.5 orbits before the TCA	2.00	1.87	84.1	78.7

Table 4.6: Comparison of results from JAC and CACAO - example 02

When the criterion is Euclidean, we find a correspondence of the order of mm/s and we observe that, as in the previous case, the value of the ΔV decreases with distance from the TCA. When the criterion is in Mahalanobis distance for CACAO and in NoS for JAC, the values are still not quantitatively comparable, but in both cases a maximum is observed 3.5 orbits before the TCA.

In conclusion, this study validated the ΔV values calculated by CACAO when the criterion is Euclidean. For a Mahalanobis criterion, we can't validate these values quantitatively, but we can verify the consistency of their evolution in relation to the date of the maneuver.

4.3. ASTERIA simulations

A simulation involving avoidance maneuvers was carried out in order to validate CACAO's ability to converge on a valid solution using the data transmitted by ASTERIA.

4.3.1. Context for the simulations

Simulations were carried out using data from the OPS-SAT satellite. Its sun-synchronous orbit is described in the following table by its circular parameters :

a **[km]**	e_x **[-]**	e_y **[-]**	i **[°]**	Ω **[°]**
6901.143	4.723 e-4	1.247 e-3	97.468	-154.153

Table 4.7: OPS-SAT's orbit

The characteristics of the satellite's propulsion system are described below :

Propulsion type	I_{sp} **[s]**	ΔV_{min} **[m/s]**	ΔV_{max} **[m/s]**
Electric	2100	0.001	0.025

Table 4.8: Propulsion system data

Simulations last 3 days, starting at midnight on January 18, 2021. The satellite is initially aligned with its reference orbit. In terms of risk, the satellite has a list of CDMs at the start of the simulation, but none was associated with a COPoC above the 10^{-3} threshold.

It was therefore decided to create artificial risks by giving the simulation new CDMs. The state vector of the secondaries correspond to the positions at which OPS-SAT would pass on the dates indicated as TCAs in a risk-free simulation. Velocity vectors have been modified so that each risk remains punctual. The covariances in the QSW frame are taken from actual CDMs for debris in low-Earth orbit. The two risks are separated by one hour.

The first simulation is carried out using an Euclidean criterion, and takes only one of the two risks into account. The second is a multi-risk simulation, this time using a Mahalanobis criterion.

4.3.2. Mono-risk simulation

The input data associated with CACAO are as follows :

- The avoidance criterion is Euclidean and is worth 500 m.
- The MAP box is 20 km wide in 3 directions. This box has been enlarged compared with the real examples treated previously, as the primary may be several kilometers away from its initial reference.
- The algorithm can build avoidance strategies with two slots, associated with a minimum ellipsoid volume of 10^{-2}, or with one slot, associated with a minimum volume of 10^{-3}.
- The values of the coefficients k_1, k_2 and k_3 used in the slot scoring are 1, 5 and 0.2 respectively.

The sequence of modes during simulation is shown in the following figure :

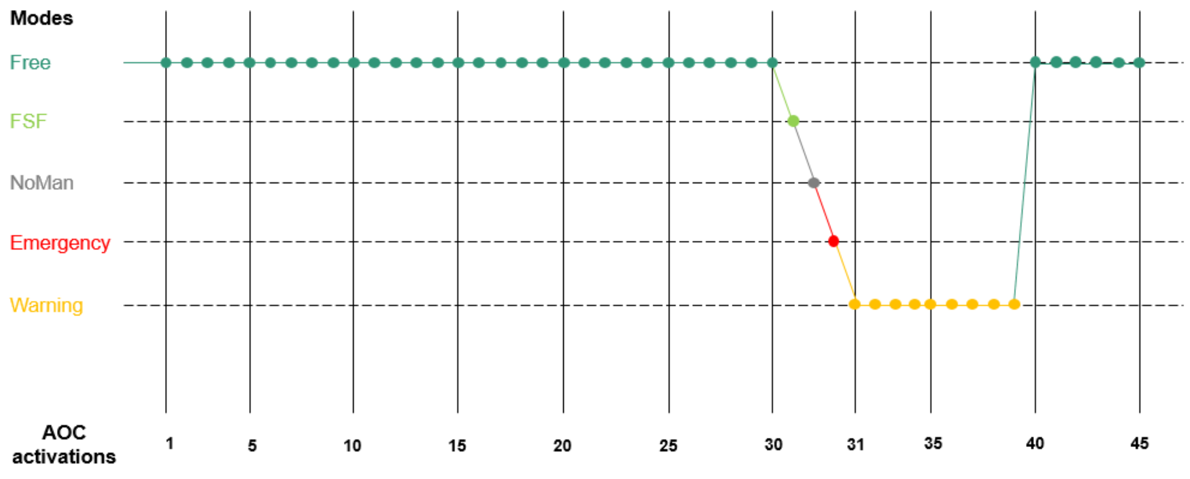


Figure 4.9: AOC modes during the simulation

The risk is detected at the 30th activation of the AOC with a COPoC of 0.002184 and an miss distance of 48.4 m. After the FSF and NoMan modes have failed to reduce the risk, the planned SK maneuvers are canceled, resulting in an miss distance of 18.2 m and a COPoC of 0.002201.

CACAO is called and 11 maneuvering slots are given as input. The calculated solution is reported in the following table :

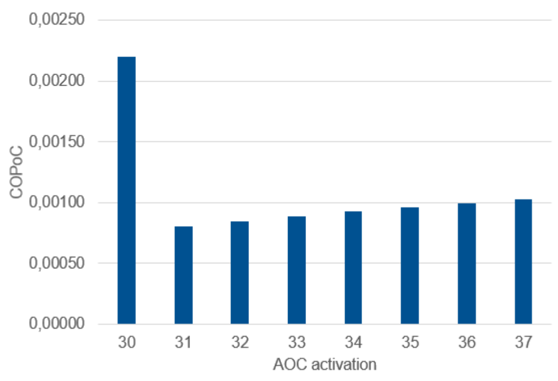
Relative position primary/secondary before maneuvers				
Miss distance [m]	18.23			
Miss distance along Q [m]	-7,87			
Miss distance along S [m]	7,30			
Miss distance along W [m]	-14,74			
Maneuvers				
Man 1 T [m/s] : [tca - 3,54 orb]	0.0189			
Man 2 T [m/s] : [tca - 2,54 orb]	0.0010			
Total ΔV [m/s]	0.0199			
Relative position primary/secondary after maneuvers				
Miss distance [m]	504.65			
Miss distance along Q [m]	149.16			
Miss distance along S [m]	-209.54			
Miss distance along en W [m]	434.19			
Relative position primary/reference after maneuvers at each control point				
Control point	TCA	0.5 orbit after Man 1	0.5 orbit after Man 2	0.5 orbit after TCA
Relative position along Q [m]	47.79	52.50	56.23	14.88
Relative position along S [m]	-6486.69	-5725.23	-6025.10	-6569.98
Relative position along W [m]	-5951.86	-6194.40	-6184.97	5963.61
Computational time				
B&B runtime [s]	2.660			
Total runtime [s]	4.788			

Table 4.9: Results of the avoidance - activation n°30 - simulation 1

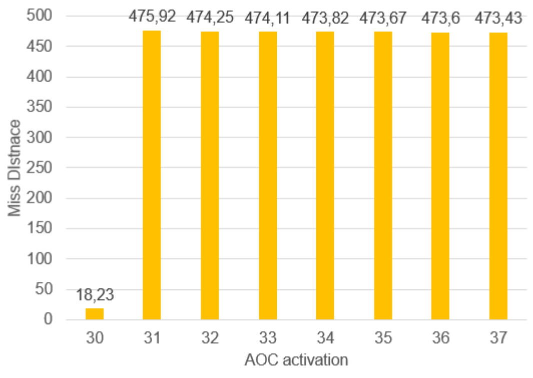
CACAO therefore manages to find a solution that verifies the constraints using two maneuvers. During subsequent activations, the strategy calculated by CACAO is sufficient to reduce the risk, and the AOC remains in Warning mode. The two decided maneuvers are frozen one after the other, then executed. At the 40th activation, the AOC decides to return to nominal mode, in which it remains until the end of the simulation.

4.3.3. COPoC of the risk

Although this simulation validates CACAO's ability to converge on a valid solution thanks to the data provided by ASTERIA, it highlights what appears to be a problem with CROCO. Below is reported the evolution of the COPoC and miss distance of the risk as a function of the number of activation :



(a) COPoC evolution



(b) Miss distance evolution

Figure 4.10: Evolution of the collision risk during the simulation

These values seem surprising. At the 30th activation, the COPoC is above the threshold and the miss distance very low. Following the development of the avoidance plan, whose maneuvers are now taken into account when studying the risk, the predicted miss distance stagnates at around 475 m for subsequent activations. However, the COPoC remains high at a value of 0.00081 at the 31th activation and increases as the risk approaches to the point of crossing the 10^{-3} threshold when the satellite is within one orbit of the TCA.

These probability values seem very high for such a miss distance. In an operational context, a radial gap of less than 100m is generally sufficient to greatly reduce the risk, and the maneuvers implemented here create a gap of almost 150m. Furthermore, the growth of this probability over time seems inconsistent. As we approach the TCA, the uncertainties around the primary decrease, which should reduce the COPoC when the objects pass at such a distance from each other. An in-depth study of the PoC calculation method implemented in CROCO deserves to be carried out to explain these results.

4.3.4. Multi-risk simulation

A second multi-risk simulation has been carried out in which the two artificial CDMs are given as input to the AOC. The CACAO data are identical to the previous simulation with the exception of the avoidance criterion, this time a Mahalanobis distance whose value is set to 10.

Since the two TCAs are located within one orbit of each other, the risks are detected at the same time and the sequence of modes is identical to the previous simulation, reported in figure 4.9. The AOC calls CACAO at the 30th activation and the second strategy studied converges on the solution detailed in the table 4.10.

In this solution, we can see that the first, more uncertain risk requires a large Euclidean distance to obtain a Mahalanobis distance equal to 10. For this reason, the maneuvers performed are greater than in the previous simulation. Risk reduction for both conjunctions and compliance with SK constraints are satisfactory, thus ASTERIA is capable of handling a multi-risk avoidance situation. However, the COPoC values for both risks are once again surprisingly high.

Relative position primary/secondary before maneuvers - CDM 1					
Mahalanobis distance [-]	0.13				
Miss distance [m]	4.32				
Miss distance along Q [m]	0.90				
Miss distance along S [m]	4.01				
Miss distance along W [m]	-1.35				
Relative position primary/secondary before maneuvers - CDM 2					
Mahalanobis distance [-]	0.51				
Miss distance [m]	18.23				
Miss distance along Q [m]	-7,87				
Miss distance along S [m]	7,30				
Miss distance along W [m]	-14,74				
Maneuvers					
Man 1 T [m/s] : [TCA 1 - 3,50 orb]	0.0248				
Man 2 T [m/s] : [TCA 2 - 2,50 orb]	0.0127				
Total ΔV [m/s]	0.0375				
Relative position primary/secondary after maneuvers - CDM 1					
Mahalanobis distance [-]	10.00				
Miss distance [m]	1971.51				
Miss distance along Q [m]	-272.64				
Miss distance along S [m]	-1950.32				
Miss distance along W [m]	-93.61				
Relative position primary/secondary after maneuvers - CDM 2					
Mahalanobis distance [-]	29.16				
Miss distance [m]	963.94				
Miss distance along Q [m]	180.04				
Miss distance along S [m]	-405.57				
Miss distance along W [m]	855.74				
Relative position primary/reference after maneuvers at each control point					
Control point	TCA 1	TCA 2	0.5 orbit after Man 1	0.5 orbit after Man 2	0.5 orbit after TCA 2
Relative position along Q [m]	131.2	-1.9	88.6	133.8	125.1
Relative position along S [m]	-7212.1	-7609.1	-5320.4	-5881.6	-8078.3
Relative position along W [m]	5251.9	-5952.1	5369.8	5333.2	5964.1
Computational time					
B&B runtime [s]	2.612				
Total runtime [s]	7.890				

Table 4.10: Results of the avoidance - activation n°30 - simulation 2

5 | Perspectives

This chapter presents ways of improving CACAO and ASTERIA's risk management.

5.1. CACAO improvement

5.1.1. Propagation of the control points

The study presented in part 4.1.2 shows that adding control points has an impact of up to a factor of 2 on the computational time. The reason for this is that adding these points lengthens the propagations that the Java overlay must perform. For example, to build the linear constraint matrices presented in section 2.1.2, the algorithm propagates from the initial date to the control point date for each of them.

CACAO's computational time can therefore be reduced if this list of dates is arranged in chronological order and any propagation that concerns them is carried out from one point to another without going back through t_0.

5.2. ASTERIA improvement

5.2.1. Warning mode exit date

Currently, when the AOC switches to the Warning mode, an exit date is set. Once this date is within the fixed horizon, the AOC returns to the Free mode on the next activation. As a reminder, in Warning mode, ASTERIA checks that the avoidance plan is still sufficient to reduce the risk and plans SK maneuvers after the TCA to return to the reference orbit. These maneuvers take advantage of extended slots, which degrade the mission, with the aim of returning as quickly as possible. At each activation, if the AOC detects an threshold overrun in the propagation horizon, this exit date is updated to correspond to the ascending node preceding this overrun.

The problem is that there is no guarantee that this mode will be exited. If the satellite deviates too regularly from its reference orbit, due to disturbances such as strong solar activity for example, this exit date will be constantly postponed, even if the TCA is in the past. However, this date can't simply be set at the TCA, which would solve this problem, as the satellite would no longer benefit from extended slots to return to its orbit after the risk.

One possible solution would be to define a guidance orbit that links the satellite's post-avoidance orbit to the reference orbit. Warning mode would perform SK around this orbit, and the exit date could be set at the moment when the guidance orbit crosses the reference orbit.

5.2.2. Avoidance criterion given to CACAO

The main problem with integrating CACAO into ASTERIA is the inconsistency of the avoidance criterion. ASTERIA considers that an encounter is no longer risky once the COPoC falls below a certain threshold. However, CACAO ensures that the primary passes at a certain distance from the secondary. There is therefore no guarantee that the maneuver plan calculated by CACAO will reduce this probability sufficiently.

Several solutions can be envisaged. Firstly, a minimum distance criterion could be introduced into ASTERIA, so that a meeting is no longer considered risky, even if the COPoC is above the threshold. Another solution would be to modify CACAO's avoidance criterion so that it is a radial distance, which would enable the minimum radial shift calculated by CROCO to be used to bring the COPoC back below the threshold. This would require modifying the quadratic constraints given to the B&B.

5.2.3. COPoC computation

Surprising results linked to the COPoC calculations, mentioned in section 4.3.3, were observed in ASTERIA simulations carried out at the end of the internship. Whatever the method used to draw up the avoidance plan, the analytical method or the call to CACAO, the COPoC values and their evolution over time appear to be abnormal. An in-depth study of their calculation method is required.

Conclusion

Autonomous collision risk management is a promising technology that could greatly simplify the operational workload for LEO satellites.

This internship therefore focused on methods for calculating autonomous avoidance maneuvers. Several modifications were made to CACAO to improve the verification of SK constraints, the management of multi-risk encounters, the selection of maneuver slots and, in general the robustness and performance of the algorithm. These additions were tested on real collision data supplied by the CNES CAESAR team, and the results of CACAO were compared with those of JAC, an operational collision management tool. These tests validated the algorithm's ability to converge towards a solution that satisfies the constraints given as input.

An integration was then carried out with the aim of replacing the analytical method for calculating avoidance maneuvers implemented in ASTERIA with a call to CACAO. Several simulations, both mono-risk and multi-risk, validated CACAO's ability to receive collision data and maneuver slots supplied by ASTERIA, and to converge, through strategy selection and optimization, on a solution that was satisfactory from the point of view of the constraints.

There are still a number of corrections and improvements to be made to this risk management system. The main task is to revise the PoC calculation method implemented in CROCO, the results of which appear to be inconsistent, thus distorting the risk assessment. A possible modification would also be to modify CACAO so that the avoidance criterion can be a radial shift, the minimum value of which sufficient to reduce the risk is already calculated by ASTERIA. Finally, the Warning mode needs to be revised, as its release date could theoretically be postponed indefinitely.

Bibliography

[1] D. Arzelier, M. Joldes, and J.B Lasserre. *Analyse et revue bibliographique de la littérature pour le calcul de manoeuvres impulsionnelles d'évitement de collision en orbite basse : rencontres simples et multiples*. Tech. rep. LAAS, 2021.

[2] *Automatisation et système bord autonome pour la gestion des risques de collision - Lot 2*. Tech. rep. DYNVOL-NT-SELRTD-00453-THA. Thalès Services Numériques, 2022.

[3] W.H. Clohessy and R.S. Wiltshire. "Terminal Guidance System for Satellite Rendezvous". In: *Journal of the Aerospace Sciences* 27.9 (1960), pp. 653–658. DOI: `10.2514/8.8704`. URL: `https://app.dimensions.ai/details/publication/pub.1019354263`.

[4] V. T. Coppola. "Evaluating the short encounter assumption of the probability of collision formula". In: *Advances in the Astronautical Sciences* 143 (2012).

[5] *ESA's annual space environment report*. Tech. rep. GEN-DB-LOG-00288-OPS-SD. Robert-Bosch-Strasse 5, D-64293 Darmstadt, Germany: ESA Space Debris Office, 2023.

[6] N.L. Johnson. "Orbital debris: the growing threat to space operations". In: *33rd Annual Guidance and Control Conference*. AAS 10-011. 2010.

[7] Nicholas L. Johnson et al. "The characteristics and consequences of the break-up of the Fengyun-1C spacecraft". In: *Acta Astronautica* 63.1 (2008), pp. 128–135.

[8] D.J. Kessler et al. "The Kessler Syndrome: Implications to Future Space operations". In: *Advances in the Astronautical Sciences* 137.8 (2010).

[9] Pierre Preault. "Autonomous collision avoidance algorithm". MA thesis. ISAE-Supaero, 2022.

[10] Chiara Rusconi. "Integration of Risk Collision Management in Autonomous Orbit Control". MA thesis. ISAE-Supaero, Politecnico di Milano, 2020.

[11] P. Seimandi. *Méthodes pour le calcul de la probabilité de collision*. Tech. rep. Centre National d'Etudes Spatiales, 2016.

[12] N.Z. Shor. "Cut-off method with space extension in convex programming problems". In: *Cybernetics* (1977).

List of Figures

List of Tables

Acronyms

EUSST *European Union Space Surveillance and Tracking*

CNES *Centre National d'Études Spatiales*

AOC *Autonomous Orbit Control*

ESA *European Space Agency*

GNSS *Global Navigation Satellite System*

LEO *Low Earth Orbit*

MEO *Medium Earth Orbit*

GEO *Geostationnary Earth Orbit*

CDM *Conjunction Data Message*

CAESAR *Conjunction Analysis and Evaluation, Assessment and Recommendations*

JAC *Java for Assessment of Conjunction*

TCA *Time of Closest Approach*

SK *Station Keeping*

BB *Branch and Bound*

PoC *Probability of Collision*

COPoC *CNES Operational Probability of Collision*

HBR *Hard Body Radius*

CACAO *Code for Autonomous Collision Avoidance Optimization*

ASTERIA *Autonomous Station-keeping Technology with Embedded collision RIsk Avoidance system*

CROCO *Collision Risk On board COmputation*

Acknowledgments

I would like to start by thanking Noé Charpigny who was the perfect supervisor throughout the internship, always available and ready to help. I would like also to thank Pierluigi DI LIZIA for supervising my internship from Milan.

I wish to thank Jérôme Thomassin for all the help and detailed explanations on ASTERIA as well as the opportunity for letting me help writing a scientific paper on the subject.

I would like to thank the whole CAESAR team and especially Christophe TAILLAN for giving helpful collision data and Pierre PREAULT for his advice on how to use JAC.

Finally, I wish to thank the entire Flight Dynamics department of the CNES for making me feel welcome during my internship and for all our discussions at coffee breaks and lunch at the delicious CNES canteen. I wish to thank especially all the other interns for the great atmosphere in the 208 room : Vincenzo CAVALIERI, Morgan DANIEL, Miguel DE ALMEIDA REBELO, Pau LAHOZ GAITX and Arnaud MULLER.