75th International Astronautical Congress (IAC), Milan, Italy, 14-18 October 2024.

IAC-24,D1,IP,7,x85237

Orbital Manoeuvring Optimization Techniques for Collision Avoidance through Decentralized Algorithms

Alessandro Tinucci Moñibas[1, *], **Vincenzo Messina**[1], **Alessandro Golkar**[1]

[1] *Chair of Spacecraft Systems, TUM School of Engineering and Design, Technical University of Munich, Caroline Herschel Straße 100/II, 85521 Ottobrunn, Germany*
[*] Corresponding Author, at.tinucci@tum.de

Abstract

Traditional centralized control schemes, while effective in less congested environments, are proving inadequate in today's increasingly dense and dynamic orbital landscapes, facing significant challenges in terms of scalability, flexibility, and responsiveness to the fast-changing conditions of space traffic. The goal of this paper is to present a novel framework for decentralized optimization of orbital manoeuvring techniques for cooperative collision avoidance, focusing on a scalable and adaptive approach that caters to the dynamic and heterogeneous nature of space traffic. Optimizing resources such as energy consumption, propellant usage, and data exchange through autonomous allocation and utilization of the satellites' subsystems.

By leveraging real-time data and predictive modeling, these algorithms enable satellites to make informed decisions about when and how to manoeuvre to avoid potential collisions. Not only improving the responsiveness of the system to unforeseen changes but also significantly reducing the communication overhead and delays associated with centralized control.

Results are presented on the benefits and drawbacks of such an approach under a varied range of operational conditions and complexities, facilitating efficient decision-making and resource allocation tailored to each unique situation. Through validation with conventional thrust-based manoeuvring techniques, comparisons are shown and an innovative procedure is proposed. Furthermore, this research aims to lay the groundwork for future advancements in spacecraft navigation, promising enhanced safety and efficiency.

1 Introduction

The increasing trend in the number of objects in orbit around Earth has meant an accompanying increase in the number of close encounters between such objects, and with it also an increased need for methodologies to deal with them.

As the number of objects in Earth's orbit continues to rise, so too does the complexity of space traffic management. This escalating scenario necessitates a significant paradigm shift, positioning enhanced operational safety as a critical aspect of future space missions.

The goal of this paper is that of proposing an alternative approach to satellite collision avoidance that, by leveraging satellites' autonomous capabilities, eases the efforts that are currently being carried out by ground stations for continuous monitoring, serving as a last safeguard in avoiding collisions while simultaneously optimizing the satellites' resource utilization.

Trough the use of inter-satellite communication and autonomous optimization, and based on the open-source Basilisk Astrodynamics Simulator[1] a framework has been developed in this paper for the simulation of this new methodology, where collision avoidance is carried out autonomously without external intervention.

Moreover, the different modules that make up the proposed framework have also been validated and a number of reference cases have been simulated in order to prove the feasibility and benefits of this new approach.

Following chapters will present first an overview of the current situation and the work that has, and is being continually developed in the frame of this particular problem, followed by an explanation of the work-flow that has been taken by the author for this paper and, finally, a display of the results obtained through a varied number of simulation runs where the effect of key parameters such as the level of adaptation of the novel scheme (measured by the number of satellites intervening in a single case) is studied.

2 Literature Review

Traditional approaches to satellite collision avoidance rely heavily on ground-based calculations and manual intervention, which are becoming increasingly unsustainable due to the growing number of objects in space. This risk, though, needs to be managed effectively to prevent the generation of space debris, which exacerbates the already cluttered orbital environment.

Indeed, any collision generates a very large amount of debris. The two biggest recent examples being the destruction of Fengyun-1 C by a Chinese anti-satellite missile in 2007, that created over 2,700 pieces of catalogued debris[2], and the collision between Iridium 33 and Cosmos 2251 in 2009, which created over 1,600 pieces of debris[3].

The need for more autonomous systems is evident given the saturation of ground-based monitoring systems. Slater et al.[4] discuss collision avoidance for satellites in formation flight, highlighting the importance of autonomous manoeuvres and the minimization of ΔV required for such manoeuvres. Also pointing towards the effectiveness of autonomous systems in managing the dynamics of satellite formations, reducing reliance on ground-based calculations.

Burgis et al.[5] focus on collaborative approaches within their simulation framework for satellite constellations, highlighting the potential for joint collision avoidance strategies.

Autonomous Orbit Control, or AOC, is a satellite's capability to identify, plan and execute corrective manoeuvres without any external intervention. Currently, this technology is operational only for Station-Keeping, preventing the satellite to drift from its reference object[6].

This is usually carried out in three steps: firstly the satellite determines its own state vector through on-board GNSS measurements, depending on this values it checks for conditions to determine if any manoeuvring is needed, and lastly it plans and executes such burns during the time slots pre-allocated for Station-Keeping.

Leveraging these existing capabilities, a novel approach to satellites' collision avoidance is possible, and is what this paper aims to present, through the utilisation of a cooperative and autonomous algorithm, with a particular focus on critical scenarios were the collision is detected during the same orbit it is expected to happen.

Additional attention to the versatility and modularity of the proposed algorithm enhance its applicability across a wide range of situations, making it a robust solution for dynamic space traffic management, with the aim of ensuring the long-term sustainability of space operations[7].

Mathematically, Patera[8] provides a general method for calculating satellite collision probabilities that integrate uncertainty factors and encounter dynamics. His work lays the foundation for many of the probabilistic models used today and is crucial for understanding how uncertainty in satellite positioning impacts collision risk.

Recent advancements also focus on incorporating real-time data and improving the accuracy of uncertainty models in collision probability calculations. Geul et al.[9] present methods for estimating uncertainty using robust weighted data fusion, enhancing the reliability of the collision probability assessments. Similarly, Coppola[10] includes velocity uncertainty in collision probability assessments, providing a more comprehensive understanding of encounter dynamics. Setting the basis for the mathematical models that are to be employed in this study

3 Methodology

In this section the structure of the proposed methodology will be laid out, starting by the simulator structure on which it was developed, and followed by the four main modules that compose it: collision generation, communication, collision detection and collision optimization.

Out of these four, two — collision generation and collision detection —- deal with the physical modelling of real case collision scenarios, and the other two, communication and collision optimization, are responsible for the particular methodology that is being developed.

3.1 Framework

All of the modules of the simulator have been developed in Python on top of the Basilisk Astrodynamics Simulation Framework[1] from the Autonomous Vehicle Systems (AVS) Laboratory of the University of Colorado Boulder.

The tool was chosen against similar software primarily because of its capabilities and the comprehensive package of tools it offers for astrodynamic simulations, its speed, and most importantly, its open-source nature.

As is usually the case, a big factor in enabling the simulation of multiple concurrent satellites is speed. In Basilisk even though the system is operated through a Python interface, the underlying simulation executes entirely in C/C++ which allows for maximum performance.

Being an open-source project, working with Basilisk enabled the complete customization of each module of the simulator to perfectly fit the needs of any particular case. The modules that were developed on top of it were all written in Python for easy and fast prototyping, but a transition to C++ is in the mind of the author in light of carrying out bigger simulations once a fixed collision avoidance algorithm has been selected.

Based on the messaging system structure of Basilisk[11], the overall collision avoidance and communication system for satellites was built on top following a general structure which can be seen in Figure 1.

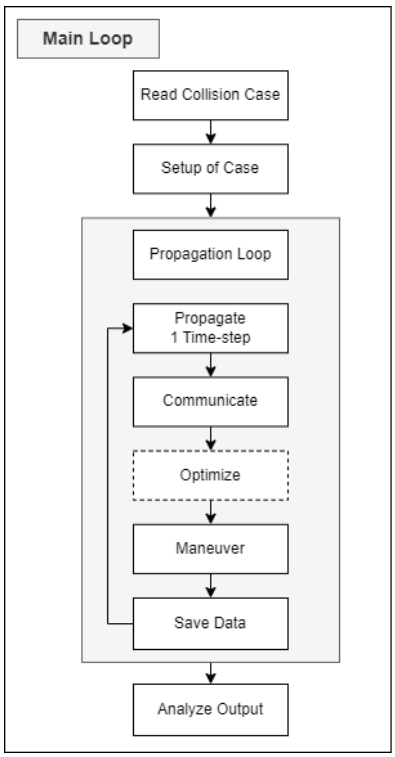

Figure 1: Structure of the main loop in the simulator.

As can be seen the main loop follows a structure in which, after the case parameters have been read and all parameters initialized, time is simulated forward one time-step at a time. Where at each time-step, all satellites do the following tasks if needed: communicate, optimize and manoeuvre. Next, the relevant data — such as current states and optimization results or manoeuvres — is saved. Finally after the simulation is finished, the data is analysed.

Regarding the overall subsystem and general modelling of the satellites, a simplified structure was opted for in order to better isolate the parameters relevant to collision avoidance phenomena.

Nevertheless, future work is planned in expanding the modelling of the different subsystems in order to allow for a more general optimization space, where different parameters — between different satellites — could be optimized for.

Satellites were therefore created with only basic parameters of mass and size and with a propulsion system that treats manoeuvres as instantaneous impulses with no regards for real propellant consumption.

3.1.1 Simulation Environment

The Basilisk environment in which all simulations are carried out consists of Earth — with J2 harmonics — as the only gravity body. The sun and other celestial bodies were not considered for the sake of simplicity, and neither were atmospheric nor solar drag.

The short time frames that are studied in this paper reduce the effect of such conditions to negligible levels, and negate the benefits obtained in light of the higher resulting computational costs.

The propagation of the satellites' states was done with an adaptive Runge-Kutta method of seventh order with eight order error estimation (RK78). Backwards propagation was done in the same fashion as normal propagation with opposite sign coefficients, making use of the deterministic nature of orbital mechanics.

The need for such high precision is a requirement that comes as a result of the high speeds and short encounter times that are usual in orbital collisions, necessitating elevated precision in order to estimate correctly the position at the time of closest approach (TCA) between the involved satellites.

With the same purpose in mind, an algorithm was developed to automatically adapt the Δt of the simulation at close distances between satellites. This was done through the use of a generalized logistic curve that returned lower values the closer the satellites were to each other, allowing for improved precision during collisions, at a reasonable computational cost.

3.2 Collision Generation

The first step in each simulation and a crucial part in the definition of the problem that is to be solved is the configuration and set-up of each collision case. As this defines the exact problem and the solution that is expected out of it.

Although for this first iteration of the simulator only simple cases on which to validate every model were needed, the versatility and variety of possible collisions was kept in mind from the very beginning.

With this in mind, the module that takes care of setting the initial conditions of the satellites was developed. Reading from a user-defined text file, it was developed to create collisions based on the following definitions:

- Definition 1. Given a point in space (x_0, y_0, z_0) and a time t_0, create n satellites (Sat_0, Sat_1, ... Sat_n) that will collide at that place and time.

- Definition 2. Given an already created satellite Sat_0, create another Sat_1 that will collide with it at a given time t_1.

- Definition 3. Given two already created satellites Sat_0 and Sat_1, and two times t_0 and t_1, create another satellite Sat_2 that will collide with them at the given times.

Definition 3, is famously known as Lambert's Problem, and through the years many algorithms to solve it have been proposed. According to [12] the most quick and accurate was presented by Izzo D.[13], and has been implemented in ESA's Pykep library [14]. This was the final algorithm implemented in the simulator.

Additionally, and because definitions 1 and 2 can leave free parameters, an additional section was added at the bottom of the text file were those values can be fixed by means of selecting a concrete orbit out of the infinitely possible ones.

With this done, any possible scenarios can be easily set up. A simple 3-satellite collision case is shown in Figure 2.

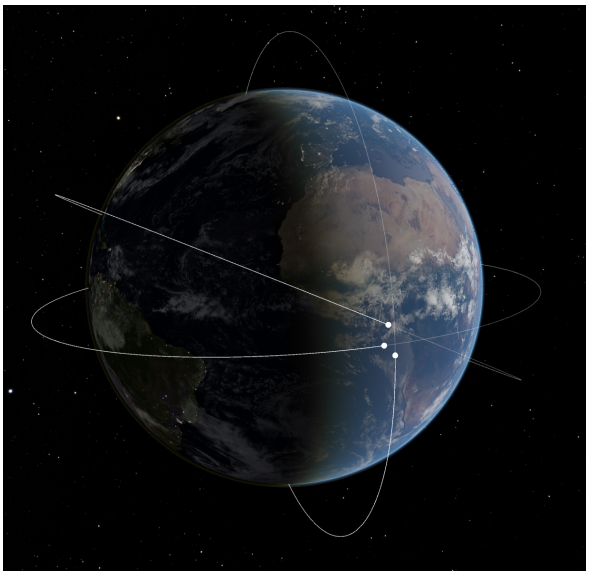

Figure 2: Example case of a 3-satellite collision case in LEO.

3.3 Communication

The first out of the simulated subsystems present in the simulator is the TT&C (Telemetry, Tracking and Command). Its functioning is fundamental for the correct exchange and propagation of the information regarding the detected collisions to the involved satellites, and key in allowing a satellite to manoeuvre as early as possible, which translates into a decrease of propellant use.

A Line of Sight model (LoS) was used for this, where satellites are allowed to communicate between them whenever they enter the cone of communication of each other. This fictional cone being given by the direction of the pointing antennas, their corresponding Field of View (FOV) and the range allowed by the signal strength and given by the corresponding link budgets of each satellite.

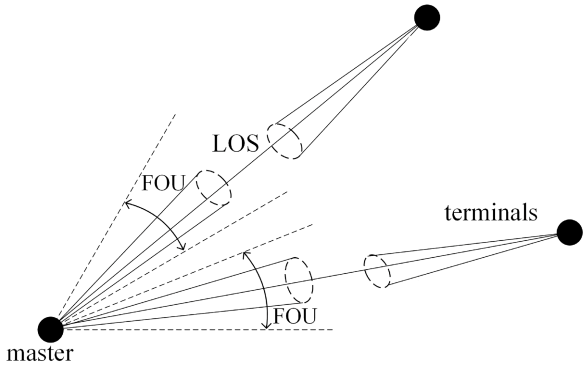

Figure 3: Visual representation of the LOS Communication model.

Some further assumptions taken in order to decrease the interference in the final results are:

- Each satellite is actively and constantly communicating within its range

- The communication between satellites has no delay, and is lossless.

An important aspect is that communication can be additionally established unidirectionally if one satellite has a high transmitting power but the second doesn't, making it so that only the second satellite receives the state information from the first satellite.

The logic the satellites follow for the communication can be better understood through the schematic on Figure 4.

This schematic consists of two main sections: the manoeuvre communication and the state communication. The first one is used to communicate the result of the optimization through satellites whenever a manoeuvre is calculated in order to have all satellites involved doing their respective burn. The state communication instead is used to transfer the state information among satellites.

Two important recalculation flags are changed throughout the simulation: one to control the recalculation of the

collision probability to reduce unnecessary calculations. The other, to check whether all satellites involved in the collision, if within communication range of each other, have the same planned manoeuvre.

3.4 Propulsion

As for propulsion, each satellite is allowed to have a variety systems on board with different values for thrust, isp and propellant mass, allowing for chemical and electrical propulsion.

With this capabilities, manoeuvres can be simulated both impulsively and non-impulsively.

The usual flow before performing a manoeuvre follows the following logic:

- A collision is first detected.
- Each satellite that has detected the collision tries to optimize the manoeuvre, considering the satellites that are capable of maneuvering based on the info he has available.
- After the optimization is done, the manoeuvre is added as a possible manoeuvre.
- The different manoeuvre are communicated among the involved satellites, until an optimal one is found. This is achieved when all satellites within communication range have confirmed the same manoeuvre among them.

3.5 Collision Detection

Naturally, one of the key aspects of a Collision Avoidance algorithm is that of calculating the actual probability that exists for any single encounter between satellites. This is a probabilistic exercise that depends on the state vectors of the satellites and on the errors (always present) of those values.

As explained by Geul [9] Two-line elements (TLEs) present the most comprehensive and up-to-date source of Earth-orbiting objects and are key in many monitoring and analysis activities. But despite their importance, they suffer from major drawbacks: they have limited accuracy, are mistagged, miss manoeuvres, and perhaps most importantly, lack covariance information. Although the TLEs are publicly available, their input observations and/or fit residuals are not. Thus, a way to calculate these values is needed

Through error propagation the probability of potential collisions can be calculated and a spread of impact locations and times anticipated. This, as he proposes, can be done through TLE differencing.

The particular method, shown in Figure 5, consists in applying a robust weighted differencing technique using least-squares regression to estimate uncertainties in satellite orbit data provided by two-line elements (TLEs). By propagating the TLEs to a common point in time, which is not necessarily the last state, accurate results can be obtained.

A similar approach is taken in the developed simulator, were additionally, because of the fact that satellites are artificially generated, real TLEs are not available.

For this, at the beginning of the simulation, satellites are propagated backwards in time for a given time, taking the state at a series of points equispaced in time as ground-truths and equivalent to the TLEs.

This yielded appropriately good results, as will be shown in the next section, that aligned with the reviewed literature.

Once the TLEs are calculated, the covariance matrix is obtained following Gaul's method:

$$C = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \tag{1}$$

Going back to the original purpose of calculating the probability of collision between two satellites at a given time, the probability can be obtained as:

$$P = \iint_{|\zeta| \leq R} N_2(\zeta, \mu_{\zeta_0}, P_c) d\zeta, \tag{2}$$

Meaning the integral of the probability density function over the hard-body radius centered at the secondary' body position at the time of closest approach in the VNB (Velocity, normal, bi-normal) reference frame with the origin around the primary body. The hard-body radius being the equivalent radius sum of the area-equivalent radiuses of each satellite. This can be visually seen in Figure 6.

Where ζ represents the position on which it is calculated and N_2 being the probability density function in the 2D case, with the general case given by:

$$\begin{aligned} \mathcal{N}_n(\xi, \eta, P) = & \frac{1}{\sqrt{(2\pi)^n}} \frac{1}{\sqrt{\det P}} \\ & \exp\left[-\frac{1}{2} (\xi - \eta)^T P^{-1} (\xi - \eta) \right] \end{aligned} \tag{3}$$

This being the function for the multivariate normal distribution where n is the dimension, P the covariance matrix that was calculated earlier, ξ the point at which it is calculated and η the initial condition, taken as the point (0, 0).

Numerically, the process of calculating the probability of collision in the simulator is carried out by each individual satellite and consists in the following process:

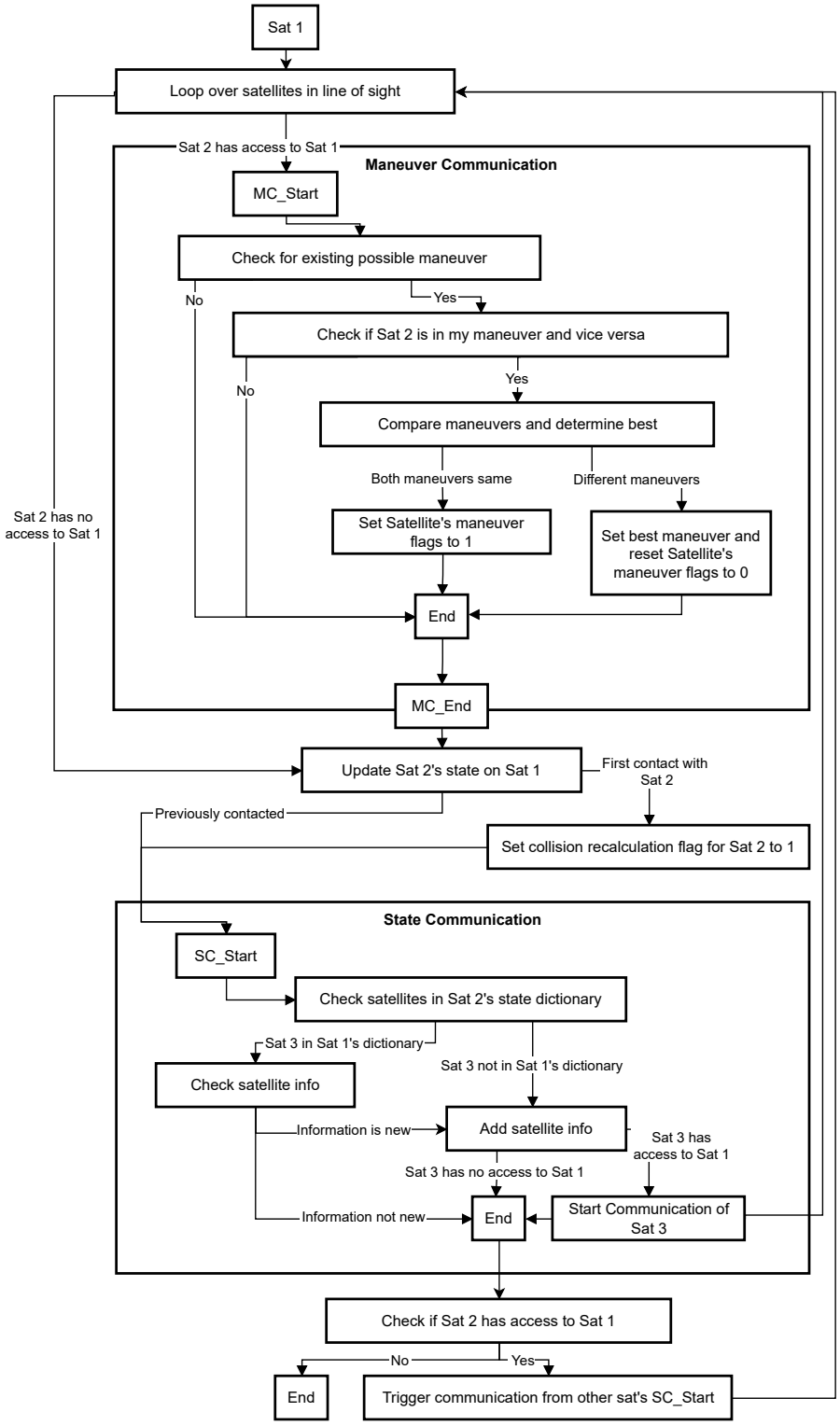

Figure 4: Schematic overview of the communication logic followed by the satellites.

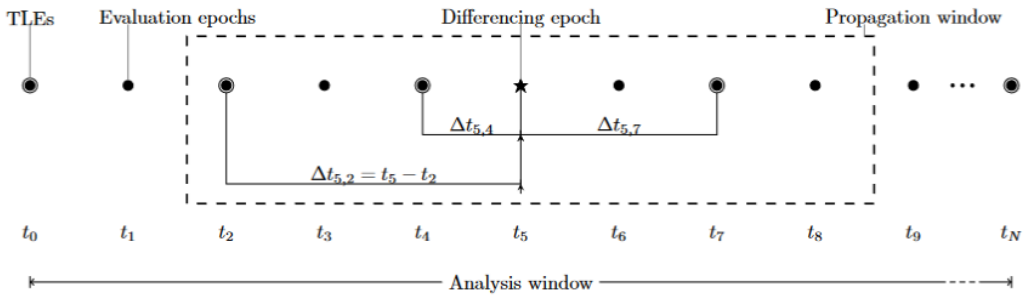

Figure 5: TLE differencing method for covariance matrix estimation in satellites.

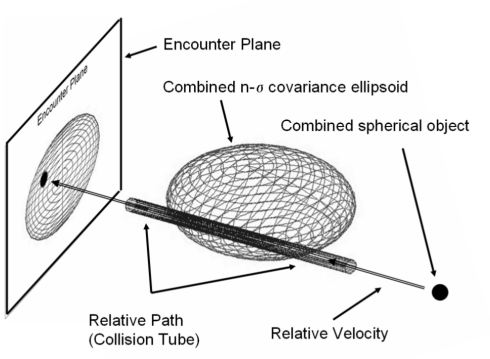

Figure 6: Collision encounter geometry from the primary body's frame of reference [8].

1. The covariance matrix is calculated once at the beginning of the simulation for each satellite and is considered constant throughout the whole simulation (assuming no more TLEs would be generated during the simulation).

2. Each satellite propagates his own and the other satellite's state to the time of closest approach (TCA).

3. The obtained states are translated into the own VNB reference frame.

4. Given the required data, the final probability of collision (PoC) is calculated.

One particularity of the implementation that was done in the code is the fact that in order to slightly speed-up the computation time, the minimum rectangle that encapsulates the hard-body circle, instead of the actual circle, was taken as the integration limit. In favour of the capabilities of the numerical libraries available in python.

This slight change is justified by the fact that the resulting effect is an increase in the probability of collision, and is could be understood as an artificial safety factor.

3.6 Collision Avoidance

Collision Avoidance is finally integrated into the simulator, and is done so following the main loop proposed in Figure 1.

Programatically, it has been integrated as a method of the satellite class that is called at each time-step after the communication loop has finished. The logic it follows is:

1. Loop over all other satellites of which the optimizing satellite has a state and where the recalculation flag is on

2. For each item the probability of collision is calculated and depending on its value being over a selected threshold:

   - If the value is over the threshold it calls the optimization method of the class to calculate the needed manoeuvres. A maneuver will be calculated and added to the list of manoeuvres to be carried out with a time at which to be done. The recalculation flag will be deactivated after the manoeuvre has been carried out.
   - It the threshold is under the limit, it deactivates the recalculation flag and continues without modification. This recalculation flag will get reset if there is an update of that particular satellite's state, if a manoeuvre for another collision case is done, or if some predetermined amount of time is elapsed.

3. Communication is established with the satellites with which it has active communication among the satel-

lites involved in the collision event in order to determine a common optimal manoeuvre

Through all simulations the value used as a threshold for considering a particular event a collision case was fixed at 10^{-6}, which corresponds to between 4-5 standard deviations from the mean. A similar value to what is used in real collision cases, which can vary between 3 to 6 standard deviations [6].

In the current iteration of the simulator no considerations were given to the satellites with regards to the available slots for manoeuvring and they were instead left free to manoeuvre at any time.

Multiple optimizers were developed for the simulations. On a first approach a gradient descent algorithm was developed. It was used for the validation of the simulations and because of its simplicity, considering only prograde manoeuvres for each satellite.

It's slowness and tendency to converge to local instead of global minima drove the development of a more versatile optimizer based on a Latin Hypercube Sampling method.

This quasi-Monte Carlo algorithm is a statistical method for generating a near-random sample of parameter values from a multidimensional distribution, giving a better coverage of the state space for a given number of samples.

The procedure employed by the optimizer is controlled by five parameters:

- ΔV Bounds: delimiting the maximum possible manoeuvre impulse
- Number of initial samples
- Loop iterations
- Domain reduction factor
- Samples reduction factor

And consists of the following steps:

- For each of the loop iterations
  - A number of samples is generated from the LHS algorithm
  - The collision probability is evaluated for each of these
  - The best manoeuvre is chosen according to the following criteria
    - If no manoeuvre produces a collision with a probability lower than the threshold, then choose the one with the lowest probability
    - Otherwise, choose the one which, producing a probability lower than the threshold, has the lowest propellant consumption
    - If, after this, there are more than one manoeuvre with the same propellant consumption, choose the one with the lowest produced ΔV.
  - The domain is re-centered around the best sample
  - The domain is reduced by the given factor
  - The number of samples to be generated is reduced by the given sample

In order to simplify posterior analysis and because of the fact that the current considerations taken in the simulator add any motivation to do so, each manoeuvre was planned for execution in the time-step after it was calculated.

Parameter		Range	Units
Orbit	SMA	6700 - 7500	km
	Eccentricity	0 - 0.3	-
	Inclination	0 - 360	º
Propulsive	Type	Impulsive / Finite	-
	Thrust	0 - 1	N
	Isp	0 - 3000	s
	Propellant Mass	0 - 10	kg
TT&C	Range	10^5 - 10^6	m
	FOV Angle	360	º
Structural	Frontal Area	0.01 - 5	m^2
	Mass	1 - 100	kg

Table 1: Main simulation parameter ranges used for the complete set of runs.

3.7 Simulation Parameters

For all simulation runs, the set of relevant parameters was chosen from inside a given range. In table 1 the main ones can be seen.

These values were chosen to be representative of the majority of satellites in LEO. Furthermore, relationships were established between the relevant parameters so as to avoid unreasonable combinations (vary low total mass with a high propellant mass).

4 Results

In this section the results obtained are displayed and explanations are given for each of the cases.

First basic results are shown: a validation of the covariance matrix estimation, a brief analysis of the performance

and execution time of the simulator and an analysis of the convergence profile of the gradient descent optimiser. In order to showcase the correct behaviour in the communication between satellites a simple 3-satellite collision is also shown. Finally an analysis of the communication range between satellites is shown, proving its benefits on reducing the needed ΔV.

4.1 Covariance and Probability Calculations

A key aspect in the performance and the requirements of the collision-avoidance maneuvers is the error on the position of each satellite, given by its covariance matrix. With the error being directly proportional to the required ΔV, a correct procedure to estimate it is crucial in obtaining realistic values for the re

Based on the values obtained by Geul et al.[9] — validated through the use of Global Positioning System (GPS) solutions for the GOCE satellite in Low-Earth Orbit (LEO), prior to its re-entry — the validity of the values obtained can be checked.

h_p [km]	e [-]		
	0 – 0.02	0.02 – 0.2	0.2 – 0.7
200 – 300	30.0	5.4	6.0
300 – 400	3.5	2.8	4.8
400 – 500	2.4	1.2	4.3
500 – 600	1.0	0.6	3.6

Table 2: Position errors in kilometres for one-day predictions as function of perigee height (hp) and eccentricity (e) identified by Wang et al. (2009).

These are shown on Table 2 where an estimation of the position error in kilometres is given based on the eccentricity and perigee height of the orbit of the object.

The simulation cases considered in this paper are LEO orbits of very low eccentricity. A range of the values obtained for them can be seen below in Table 3.

h_p [km]	e [0 - 0.02]
400 – 500	1.0 - 5.0
500 – 600	0.5 - 3.0

Table 3: Position errors' range in kilometres as a result of the artificially generated TLEs.

As can be seen, the values are in accordance to what is expected, with a variability that is a result of the three parameters that control the artificial method of developing the TLEs: the number of TLEs generated, the ΔT between them and the time-step used for the propagation.

Through these the behaviour can be fine-tuned to be within the desired range. For the particular cases tested, the following values were used.

- Number of TLEs: 3
- ΔT between TLEs: 12 hours
- Propagator ΔT: 1 second

Regarding the probability of collision (PoC) calculations, basic checks such as invariance in the reference of frame used — of the primary or secondary satellite — were carried out.

Particularly for the change of reference frame, small discrepancies on the order of 10^{-6} were observed, that although irrelevant for the actual PoC, ended up affecting the manoeuvre calculations, as will be shown.

4.2 Maneuver Optimizer Performance

Validation of the correct functioning of the optimizer is of utmost importance if the final results are to be studied. Figure 7 then represents what is the optimization of a two-satellite collision case.

In it, all of the points that were generated by the optimizer are shown along with their corresponding probability of collision. In this case, impulsive manoeuvres were considered and thus, all manoeuvres had the same equivalent propellant consumption, 0. In the x axis, the total ΔV is instead show. Finally, the area of $PoC \geq 10^{-6}$ is shown.

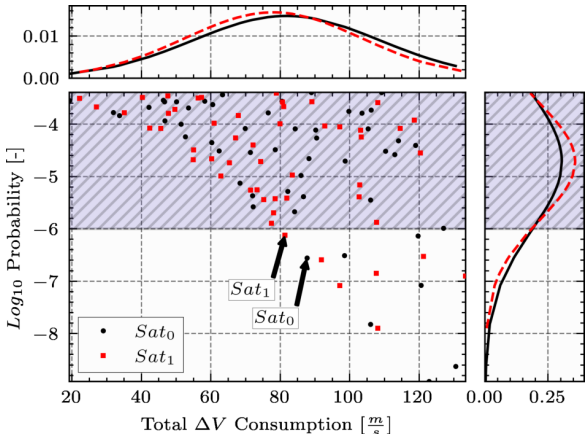

Figure 7: LHS Optimization showing the relation between the total ΔV requirements and the obtained collision probability for a simple 2-satellite scenario.

It is easy to verify that the chosen manoeuvres by the satellites are both under the critical zone, and among those,

correspond to the ones with lowest total ΔV. Validating the intended functioning of the algorithm.

A different visualization of a similar optimization is shown in Figure 8. Here each axis corresponds to the total ΔV to be carried out by each Satellite. On the background a contour is extrapolated from the probability values on a logarithmic scale.

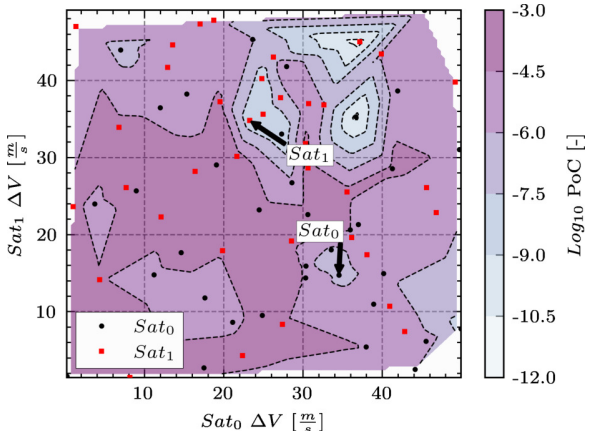

Figure 8: LHS Optimization showing the specific state-space analysed for each pair of satellites — only one in this case — in a 2-satellite scenario. $N = 37$

The lack of structure in the contour indicates though that the number of samples is not representative and has to be increased. After increasing the number of samples, some structure can start to be seen, as in Figure 9, as is to be expected.

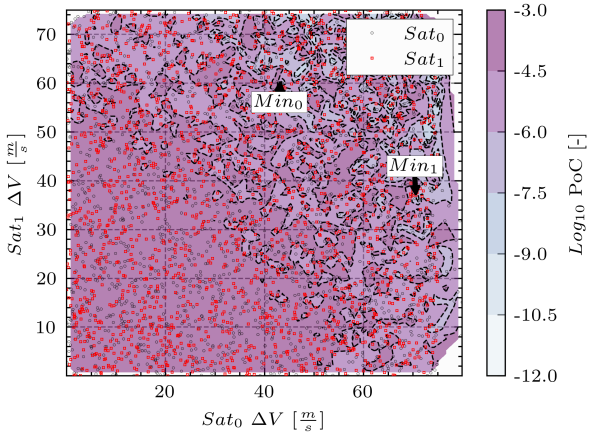

Figure 9: LHS Optimization showing the specific state-space analysed for each pair of satellites — only one in this case — in a 2-satellite scenario. $N = 1875$

One important aspect to be taken into account when studying the results obtained is the fact that even though convergence to a global best manoeuvre would be expected when increasing the number of samples used in the optimizer, the presence of symmetries between the distribution of the impulses among satellites meant an increase in the variability of results, that can vary even when repeating the same exact case several times due to the different trajectories that the satellites can follow afterwards. This symmetries are also the cause for the fact that the two satellites converge to different points in Figure 9.

4.3 Cooperative Satellite Network

The biggest and most important result was that of validating a simulation where satellites were made to autonomously detect, calculate and avoid manoeuvres without external intervention.

Furthermore the simulator was also used to show that an increase in the number of satellites involved in collission events could lower the utilization of resources.

This can be seen in Figure 10 were a series of simulations were made, randomly sampling the parameters of the collisions and of the involved satellites, and with a varying number of *communicator satellites*. Satellites that were not involved in the collisions and were only propagated considering communication with other satellites, effectively working as information relays

The rest of the parameters were sampled by an LHS algorithm again, and controlled:

- The geometry of the collision, by changing the number of satellites involved and their orbits.
- Effective communication range of the satellites, as a result of varying the transmitting and receiving powers of their TT&C Systems.
- Propulsive systems. With thrusts and isp values equivalent to systems ranging from mono-propellant to hall effect thrusters.
- Satellite size and mass.

While Figure 10 shows the total ΔV requirements and distribution for these simulations, the distribution for the mean time of the manoeuvres can be seen in Figure 11.

Although Figure 10 is more representative of the final value that is trying to be lowered, the slight inconvenience of using a statistical sampling method as optimizer results in many cases having a higher ΔV consumption due to the optimizer not sampling the same exact manoeuvre each single time.

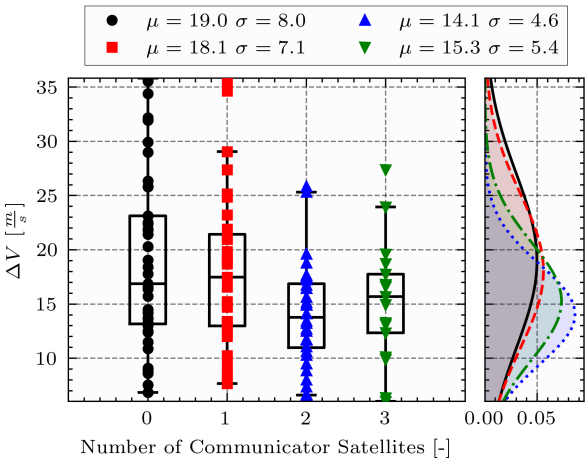

Figure 10: ΔV distribution for a set of randomized simulations with a varying number of relay satellites. N=125.

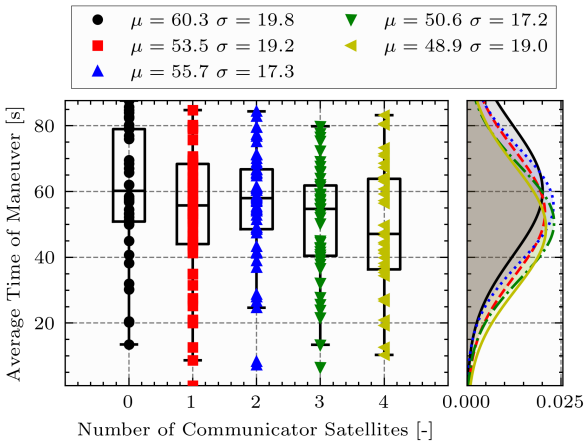

Figure 11: Mean time of manoeuvre distribution for a set of randomized simulations with a varying number of relay satellites. N=250.

This effect could effectively be lowered through the use of a higher number of samples. In this case, the particular parameters chosen for the optimizer, among which was the number of samples, was result of a trade-off study between speed and performance.

Several simulations were carried out with varying optimizer parameters for identical collision cases and the combination which produced consistently better manoeuvres was chosen.

The mean time of the manoeuvres becomes then the most useful of the parameters, showing the improved speed of propagation of the information among the satellites.

4.4 Communication Between Satellites

Some simulations were carried out afterwards to study the possibility and effect of allowing a unidirectional communication among satellites were a satellite can receive the state information from another satellite and calculate the manoeuvre based on it, even in cases where the second satellite cannot receive the first's information.

This allows for a faster transfer of information as only the condition of one satellite reaching the other needs to be achieved and there is no condition of bi-directionality.

As can be seen in both Figures 12 and 13, allowing for unidirectional communication may help lowering resource utilization further.

5 Conclusions

In response to the escalating challenges posed by the growing number of man-made objects in Earth's orbit and the associated increase in resource demands on ground stations, a novel approach has been proposed for collision detection and avoidance in this thesis.

This approach capitalizes on the autonomous capabilities of satellites to facilitate operational management and act as a crucial safeguard against imminent collision threats.

A modular and versatile simulator, underpinned by the principles of communication, cooperation, and autonomous optimization, has been developed and demonstrated.

Results have been shown, through multiple randomly sampled collision simulations where the number of involved satellites was increased, that a high adaptation of this novel approach is not only feasible, but also highly beneficial to the reduction of the satellites' resources exploitation.

A wide operational network of satellites operating according to the proposed paradigm could not only operate autonomously without the need for human intervention, but also considerably lower the resource utilization regarding the manoeuvres.

References

[1] P. W. Kenneally, S. Piggott, and H. Schaub, "Basilisk: A flexible, scalable and modular astrodynamics simulation framework," *Journal of Aerospace Information Systems*, vol. 17, no. 9, pp. 496–507, 2020.

[2] N. L. Johnson *et al.*, "The characteristics and consequences of the break-up of the fengyun-1c spacecraft," *Acta Astronautica*, vol. 63, no. 1, pp. 128–135, 2008.

[3] N. Johnson, "Orbital debris: the growing threat to

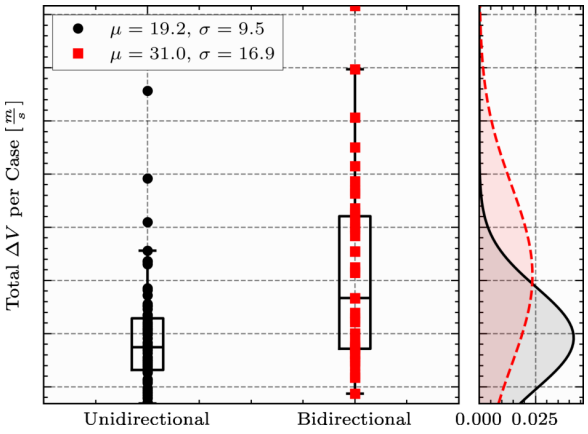

Figure 12: Total ΔV distribution comparison between a random set of cases simulated with unidirectional and bidirectional communication.

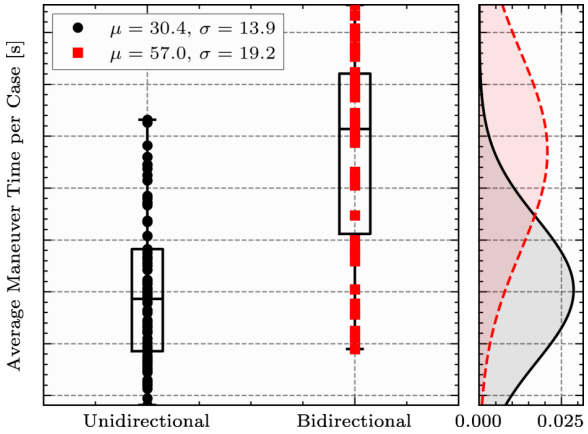

Figure 13: Mean manoeuvre time distribution comparison between a random set of cases simulated with unidirectional and bidirectional communication.

space operations," in *33rd Annual Guidance and Control Conference*, 2010. AAS 10-011.

[4] G. L. Slater, S. M. Byram, and T. W. Williams, "Collision avoidance for satellites in formation flight," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 5, pp. 1140–1146, 2006.

[5] S. Burgis, L. Rohrmüller, M. Michel, and R. Bertrand, "Simulation of satellites and constellations for the assessment of collision avoidance operations," *CEAS Space Journal*, vol. 15, no. 5, pp. 655–670, 2023.

[6] J. Grauby, "Autonomous collision avoidance algorithm," Master's thesis, Politecnico di Milano, 2023.

[7] National Aeronautics and Space Administration, *NASA Spacecraft Conjunction Assessment and Collision Avoidance Best Practices Handbook*, December 2020. NASA/SP-20205011318.

[8] R. P. Patera, "General method for calculating satellite collision probability," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 4, pp. 716–722, 2001.

[9] J. Geul, E. Mooij, and R. Noomen, "Tle uncertainty estimation using robust weighted differencing," *Advances in Space Research*, vol. 59, no. 10, pp. 2522–2535, 2017.

[10] V. T. Coppola, "Including velocity uncertainty in the probability of collision between space objects," *Proceedings of the AAS/AIAA Space Flight Mechanics Meeting*.

[11] S. J. Carnahan, S. Piggott, and H. Schaub, "A new messaging system for basilisk," in *Preprint AAS 20-134*, 2020.

[12] E. F. David de la Torre Sangrà, "Review of lambert's problem,"

[13] D. Izzo, "Revisiting lambert's problem," *Celestial Mechanics and Dynamical Astronomy*, vol. 121, pp. 1–15, Jan 2015.

[14] ESA, "Pykep library." `https://esa.github.io/pykep/`, 2020.