AUTOCA

Autonomous Collision Avoidance System

I. F. Stroe[(1)], A. D. Stanculescu[(1)], P. B. Ilioaica[(1)], C. F. Blaj[(1)] , M. A. Nita[(1)], A. F. Butu[(1)], D. Escobar[(2)], J. Tirado[(2)], B. Bija[(3)] , D. Saez[(3)]

[(1)] GMV, SkyTower, 32nd floor 246C Calea Floreasca, Sector 1, Bucharest, Romania, email: istroe@gmv.com , andrei.stanculescu@gmv.com, paul.ilioaica@gmv.com, cblaj@gmv.com, anita@gmv.com, abutu@gmv.com

[(2)]GMV, Isaac Newton 11, P.T.M. Tres Cantos, Madrid, Spain, email: descobar@gmv.com, jtirado@gmv.com

[(3)] GMV, Harwell Innovation Centre,Building 173, Curie Avenue, Harwell Science and Innovation Campus, Didcot, Oxfordshire OX11 0QG, UK, email: bbija@gmv.com , dsaez@gmvnsl.com

ABSTRACT

Due to the intensive activities in space during the last half century, the population of man-made space objects is playing an increasingly relevant role in the space environment. ESA´s Master Model predicts that there are around 600000 objects bigger than 1 cm orbiting the Earth.

In light of recent events, SpaceX has launched several Starlink satellites in the near Earth region. While taking into account the increase in the number of space debris and the deployment of mega-constellations, an increase of the number of collision avoidance warnings is inevitable. Furthermore, this increase in the number of high-risk conjunctions will led to an increase in the Collision Assessment workload, making difficult and almost impossible for the collision avoidance decision to be taken manually.

This paper provides an in-depth description of the Autonomous Collision Avoidance System (***autoca***) which has the objective to develop mechanisms that allow the automation of all the processes and procedures related to decision-making. Furthermore, the current study is focused on the collision risk assessment and on the operational implementation of collision avoidance manoeuvres, as well as on the use of existing GMV software [2] that provides implementation for orbital mechanics, collision risk and avoidance manoeuvre algorithms. Having the goal to recommend accurate manoeuvre decisions, the study innovates with the use of deep learning algorithms that are used to discover patterns within the historical data.

1. INTRODUCTION

The increasing density of satellites in Low Earth Orbit causes multiple concerns about the safety of future space missions. When a collision occurs, the consequences depend on the size and velocity of the debris. To determine the risk of collision for the active satellites, conjunction assessments are continuously performed for all trackable objects in LEO and GEO regions. The problem becomes more complex for the case of missions with electric propulsion. While taking into consideration the longer reaction time, the decision must be performed well in advance. Because of the high uncertainties in orbit determination and orbit propagation, the decision making process tends to become a highly difficult task.

GMV is currently developing an Autonomous Collision Avoidance System (***autoca***) together with EUTELSAT and ESA, within the ARTES program. This system is based on the use of machine learning techniques, and aims at its use for large fleets (e.g. large operators in GEO and future mega-constellations in LEO and MEO) and also orbit raising scenarios with full-electric satellites (e.g. orbit transfer from LEO to Upper-LEO for deployment of a large constellation or from LEO/GTO to GEO for a large telecom satellite).

GMV has extensive experience in flight dynamics systems and space debris, and has its own software solution for Collision Assessment: ***closeap [2]***, a tool designed to support the detection of close conjunctions, the collision risk computation and the optimization of collision avoidance manoeuvres. ***closeap*** has also a solid foundation provided by the conjunction detection and collision probability algorithms inherited from ***CRASS***. It also counts on the special algorithms for orbit determination in degraded tracking situations from ***ODIN*** and on ESA's ***NAPEOS*** library. The ESA's Database and Information System Characterizing Objects in Space (***DISCOS***) is used operationally together with ***CRASS*** and ***ODIN*** being also developed and supported by GMV.

The innovation is brought in this project by the use of machine learning techniques to generate models that are trained on the past data, in the form of augmented conjunction data messages (**CDMs**), in order to decide

whether a collision avoidance manoeuvre should be considered or not to minimise the collision risk.

2. AUTOCA OVERVIEW

The purpose of the ***autoca*** software is to take autonomous decisions on the need to perform a manoeuvre after a collision risk assessment has been performed and a high-risk conjunction has been detected and to recommend the optimal collision avoidance manoeuvre when needed. This section aims to outline the software architecture in terms of its components, inputs, outputs and operation states.

The Conjunction Data Messages (CDMs) are the main input of the software and they are used to decide if a collision avoidance manoeuvre is needed and to recommend an optimized solution. The information provided in CDMs is complemented with the operator orbit, which includes not only a more reliable source of information for the primary satellite orbit but also includes the planned manoeuvre in the future. The already propagated ephemerides are used as a trigger for a CAM analysis in case of receiving an update not necessarily linked to a new CDM. Furthermore, some external auxiliary data such as EOPs, leap-seconds and space weather information is needed. Finally, a configuration file received is used to configure the operational constraints such as the CAM strategy to be applied, the maximum delta-V allowed by the propulsion system of the satellite and others.

autoca is composed of the following main components:

- autoca-core: the core component that groups all the subcomponents providing the functionality of the software.
- CDM-reader: it is an adapter to read the CDM from different sources. It defines a common internal interface with autoca-core and different implementation strategies can be developed to read the CDMs from different sources.
- Datastore: it is an internal database to store the CDMs, configuration, inputs and outputs or any other data that the software needs to operate.
- CAM library: it encapsulates all the business logic related to the conjunction analysis and the generation of the possible collision avoidance manoeuvres that could be applied in any case according the operational constraints, making use of ***closeap*** library.

The dynamic architecture of the software presented in Figure 1 indicates the 4 different operation states:

1. Data Initialisation: The first time the ***autoca*** is executed, it needs to be initialised with historical records of collision warnings in the form of CDMs coming from the SSA provider. Those CDMs are read and stored in the internal database so it can be used in the next stage. The data initialisation can be seen as an initial execution of the data ingestion state that is performed right after installing the software.
2. Machine Learning Training: Once the database has been populated with CDMs, they are used to train the machine learning models in order to predict the evolution of the secondary object's covariance.
3. Data Ingestion: When a new CDM arrives, it needs to be parsed and stored into the internal database. It also correlates the new CDM with the already existent ones to identify conjunction events (series of CDMs referring to the same event).
4. CAM Decision-making: the software autonomously decides if a CAM is to be implemented and which one is recommended. It is made by assessing the criticality of the event, using the predicted covariance evolution and analysing the possible manoeuvres that fits the operational constraints in case of high-risk events.

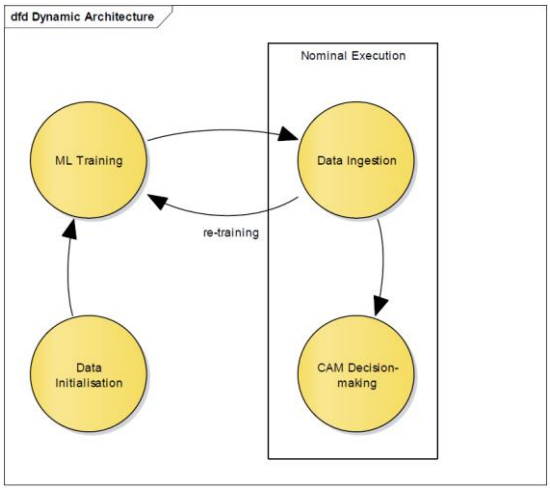


Figure 1. AUTOCA state diagram

Apart from this workflow, the user can decide, at any moment, to re-train the machine learning models to fit the new data that has been ingested since the last time they were trained. It allows to improve the score of the machine learning models and, consequently, improves the CAM decisions.

The main outputs obtained as a result of the ***autoca*** execution are the CDMs re-processed by the software and augmented with some more information such as the probability of collision and criticality assessment and the orbital information of the primary object containing the manoeuvre proposed, as well as the graphical visualisation of the conjunction parameters (and their covariance) and the proposed manoeuvres to better understand the nature of the problem and the calculations performed.

3. DATA PREPARATION

autoca software takes as input dataset for the machine learning models training a set of historical CDMs for different conjunction events. The raw CDMs are processed and stored in the software internal database before using them. The accepted formats are KVN, XML and CSV.

3.1. Data pre-processing

3.1.1. Data filtering

After assuring unified datatypes for each set of data and initial cleaning of missing values or misspelled entries, fields of the dataset are renamed and normalized to allow easier data handling. Additional useful fields are calculated, e.g. time to TCA for each CDM (based on CDM creation date) or time since last observation used for the OD process. State vectors and covariance matrix elements for primary and secondary objects, as well as relative state vector may be registered in different references frames, thus all the relevant values are transformed to common inertial system EME2000, allowing proper and consistent computations. For a portion of CDMs the information of the probability of collision is missing, thus for consistency it is recalculated for all CDMs according to [7].

In order to identify the actual updates of orbital information of the secondary object, which is the one which covariance evolution needs to be predicted, a set of filters are defined to discard the updates in which the secondary object information has not been updated, based on the information provided in the CDM. Therefore, events with a single update, covariance outliers, updates that have last observations start and end equal to the previous, updates with observation used or available equal to the previous CDM and Updates with state vector distances within 10^-6 compared to the previous CDMs are disregarded.

In order to train the Machine Learning model, the dataset is divided within the following way: 80% for training, 10% for testing and 10% for validation.

3.1.2. Scaling the data

The range of feature values varies widely, thus a transformation is applied aiming proper scaling of data used as input for machine learning algorithm.

Quantile Transformer is used in order to transform features according to quantile information [9]. Its default behaviour is to map the original values to a uniform distribution; however, it is possible to obtain a normal distribution as well. The mapping is done by using the cumulative distribution function F of the feature and the quantile function of the desired distribution G as such: $G^{-1}(F(X))$; this formula is based on the fact that the cumulative distribution function of a random variable X is uniformly distributed on $[0, 1]$, and that $G^{-1}(U)$ has distribution G, where U is a random variable uniformly distributed in the range $[0, 1]$ as well. According to the documentation, the quantile transform, by performing a rank transform, smooths out "unusual" distributions and is more robust to outliers than regular scaling methods. Correlations and distances within/across features are, however, affected (distorted).

The ranking of the values of each feature is preserved when using quantile and power transforms, since they are based on monotonic transformations [10]. Power transformations aim to achieve an output distribution as close to a normal distribution as possible, with the intent of minimising skewness and to stabilise variance. In the Python Scikit-Learn library [8], there are two means of computing power transformations: the Yeo-Johnson and Box-Cox transforms. Since the Box-Cox transform can be applied only to strictly positive data, the Yeo-Johnson method is used in the scope of the prototypes described herein. The Eq. (1) details the Yeo-Johnson transform, parametrized by λ (determined through maximum likelihood estimation for minimising skewness and stabilising the variance):

$$x_i^{\lambda} = \begin{cases} \dfrac{[(x_i+1)^{\lambda}-1]}{\lambda}, & if\ \lambda \neq 0, x_i \geq 0 \\ ln(x_i+1), & if\ \lambda = 0, x_i \geq 0 \\ -\dfrac{[(-x_i+1)^{2-\lambda}-1]}{2-\lambda}, & if\ \lambda \neq 2, x_i < 0 \\ -ln(-x_i+1), & if \lambda = 2, x_i < 0 \end{cases} \quad (1)$$

4. COLLISION RISK EVALUATION AND MITIGATION STRATEGY

4.1. Conjunction assessment

When a collision occurs, the consequences depend on the size and velocity of the debris. To determine the risk of collision for the active satellites, conjunction assessments are continuously performed for all trackable objects in LEO and GEO.

The miss distance and the probability of collision are the main parameters broadly considered in the conjunction assessment, especially in LEO regime. The probability of collision depends both on the minimum distance between the first and second flying object at TCA as well as on their size. Besides the position and velocity vectors, the covariance matrix (size and orientation in the relative frame, B-plane) and the size of the objects have a significant role in the calculation of the collision probability as presented in Eq. (2) thus the accurate

prediction of these parameters is the basis of the validity of the collision risk assessment.

$$PoC = \frac{1}{2\pi|Det(C)|^{\frac{1}{2}}} \iint_{x^2+y^2 \le d^2} \exp\left(-\frac{1}{2}(r - r_{S/P})^T C^{-1} (r - r_{S/P})\right) dxdy \quad (2)$$

Where C is the $2x2$ projection of the combined $3x3$ covariance at TCA onto the conjunction plane, d is the sum of the two object sizes, $r = (x, y)^T$ is any point in the collision plane such that $x^2 + y^2 \le d^2$ and $r_{S/P} = (r_{S/P}, 0)^T$ is the position of the secondary relative to the primary along the x-axis in the collision plane, as described in [1].

The GEO orbital regime is much less populated than the LEO regime, between one and two orders of magnitude less populated. This implies that high risk events in GEO occur with lower frequency than in LEO. Because of this, GEO operators do not use probability of collision as a driving parameter to monitor the risk of collision and evaluate the need for collision avoidance manoeuvres. Instead, GEO operators monitor the miss distance of secondary objects, and in particular, the radial miss distance. GEO operators perform station-keeping manoeuvres around every two-weeks in order to keep the satellites in the operational orbit. Moreover, GEO dynamics occur at a larger time scale than in LEO, being in the order of days in GEO (e.g., orbital period in GEO is 1 day) and in the order of hours in LEO (e.g. orbital period in LEO is between 1-2 hours normally). GEO dynamics occur at a very different time scale than in LEO, allowing to predict potential collisions much more time in advance (up to 10 to 14 days) than in LEO (about 7 days).

4.2. Collision risk computation

An important contribution in the collision risk assessment is the evaluation of the probability of collision. A new methodology into this filed is worth considering for ***autoca*** and it is the so-called scaled probability (scaled *PoC*). The idea behind this concept is that covariance information linked to the orbital information used for the computation of the probability of collision may not be very accurate. Covariance values in current objects catalogues can be in many cases poorly estimated by the orbit determination processes used to estimate the orbit and its associated uncertainty for both objects involved in the conjunction.

This methodology tries to enhance the level of realism of the covariance matrices involved in a conjunction event in order to achieve a more reliable estimate of the actual risk of collision.

A qualitative characterization of covariance, presented in Figure 2, can be described generating a diagram of the orbital positions and successive covariance in the local orbital system referred to the object in its most recent occurrence, as highlighted in [3].

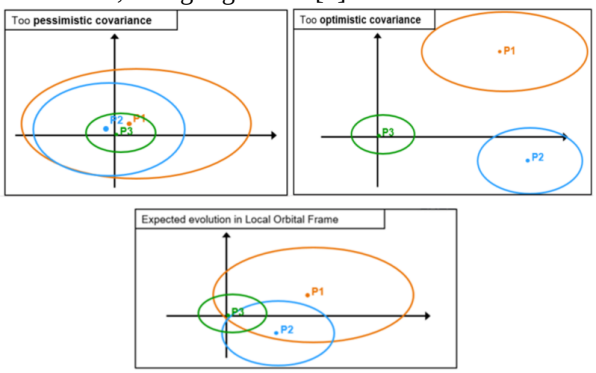


Figure 2. Visualization of orbital positions and succesive covariances

In order to obtain a reliable value of the collision probability, scaling factors for the primary and secondary covariance matrices need to be applied. The scaling factors are determined assuming that the most recent observation (CDM) is the most accurate and the Mahalanobis distance for each orbital information can be calculated with respect to it.

The uncertainties in position follow a Gaussian distribution, so the square of the Mahalanobis distance for each observation in an event must follow a Chi2 distribution, with three degrees of freedom. Thus, an observed cumulative distribution function (CDF) can be obtained by sorting in ascending order the Mahalanobis distances of the latest update with respect to the previous ones as shown in Figure 3.

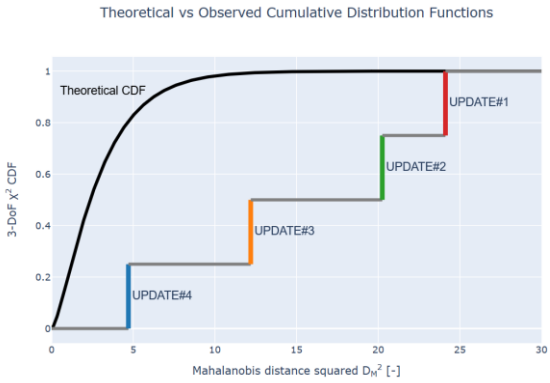


Figure 3. Theoretical and observed CDFs

The following methods are proposed in [3] for adjusting the scaling factor values, so that the observed distribution fits the theoretical distribution expected:

- K-Point coefficient
- K-Interval coefficient
- Kolmogorov-Smirnov method

In k-point method each observed point should correspond exactly to its theoretical position from the cumulative distribution function. In the "K-interval" method, each element of the sample is corrected with a different factor so that each observed point should be in its theoretical interval. The maximum difference in the probability recorded between the distribution function and the data set corrected by a certain scaling factor K corresponds to the Kolmogorov Smirnov method. The KS test measures the compatibility of the data sample with a theoretical probability distribution function. In other words, the purpose of using this method is that the Mahalanobis distances belonging to an observation must follow a theoretical cumulative distribution function.

The general method consists in performing a statistical test that measures the distance between the theoretical distribution and data. Then the two distributions are projected together and the greatest vertical distance between the two is measured using Eq. (3):

$$D_{ks} = \sup\left(F_t(x) - F_{data}(x)\right) \qquad (3)$$

Where $F_t(x)$ is the CDF of the theoretical distribution and $F_{data}(x)$ is the empirical distribution function of the data.

Then the probability of obtaining data that have a value greater than the observed value is calculated, assuming that the theoretical hypothesis is true. With the D_{KS} value and the size of the Mahalanobis distance sample (number of observations), the probability (p-value) of getting a D_{KS} value greater than the computed one if the null hypothesis is true can be obtained. This p-value is compared against the level of significance α chosen when performing the test (a typical value is α = 5%) and, if the p-value is less than α, there is evidence enough to reject the null hypothesis with a confidence level of (1−α) %, as presented in Figure 4.

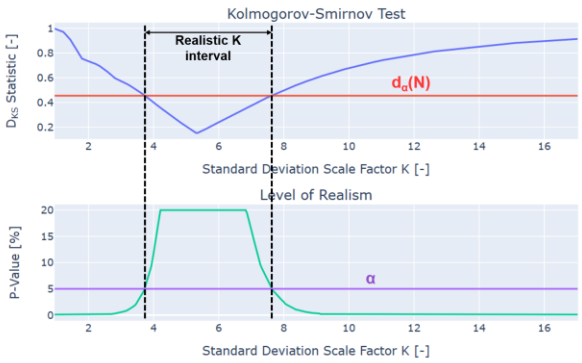


Figure 4. Kolmogorov-Smirnov Test

It shall be noted that, since the value K=1 may not be included in the realistic interval that arises from the Kolmogorov-Smirnov test, a good practice is to set as final solution the union of the ranges obtained from the K-Interval method and the Kolmogorov-Smirnov test. In this way, the nominal data of the latest update of the object under analysis is guaranteed to be considered when computing the scaled probability of collision.

Another condition that must be taken into account for the scaling factors obtained from the Kolmogorov-Smirnov method to be reliable is the number of observations used. In order to achieve enough statistical power it is good practice to use as many observations as possible, so that a minimum threshold of the previous independent updates must be set (typically 3). If the amount of information in a conjunction event does not exceed this threshold, a fixed range *is* considered instead.

Following the three methods described above it is possible to obtain the range of the scaling factors for which the validity of the relationship between the sample corrected by the K factor and the theoretical distribution is statistically confirmed

Withal the scaling factors for primary and secondary objects, K_p and K_s, needs to be used and applied to the primary and secondary covariance matrices. Both factors are squared because the convention followed here considers that K_p and K_s, are scale factors applied to the standard deviations. Therefore, the corresponding scale factors for the elements of the covariance matrix should be K_p^2 and K_s^2. Thus the corrected level of realism of the combined position covariance matrix is obtained by applying separate scale factors both for the position covariance matrices of the primary and secondary objects as expressed in Eq. (4) :

$$C = K_p^2 C_p + K_s^2 C_s \qquad (4)$$

4.3. Predicted probability of collision

The **next-day conjunction analysis** is the methodology proposed in [4] to predict the evolution of the probability of collision over time. Anticipating the evolution of the collision probability in the near future aims to ease the work of the satellite operators regarding the decision to perform the avoidance manoeuvre if the collision risk stays high or if it is expected to dissipate.

For predicting the evolution of the PoC, a prediction of how the covariance of primary and secondary objects are expected to evolve is needed first. The approach used for this consists in the use of Machine learning models which allow predicting the evolution of position error of the objects during a conjunction event. To this end, the previous CDMs are used to predict the covariance and hence the collision risk between objects. Different time horizons for the predictions (24h, 48h, 72h before TCA, and TCA) are used in ***autoca***.

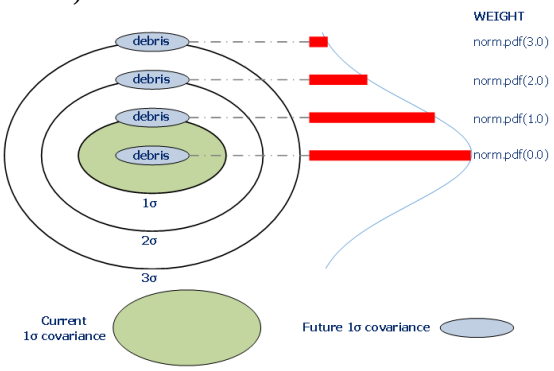


Figure 5. Scattering process performed in the B-plane in order to generate a set of feasible miss vectors in the future

Once a future combined position covariance matrix has been predicted, the approach proposed by EUMETSAT consists in performing a pseudo Monte Carlo analysis based on the data of the latest available update of the event, that is, performing a scattering process where a large number of feasible future miss vectors are obtained using the latest combined position covariance matrix in the B-plane, as qualitatively illustrated in Figure 5. Since position errors are expected to follow a zero-mean Gaussian distribution, this set of feasible miss vectors can be obtained from a 2D Gaussian distribution centred on the latest available miss vector. The shape of this Gaussian distribution is given by the latest available combined position covariance matrix in the B-plane.

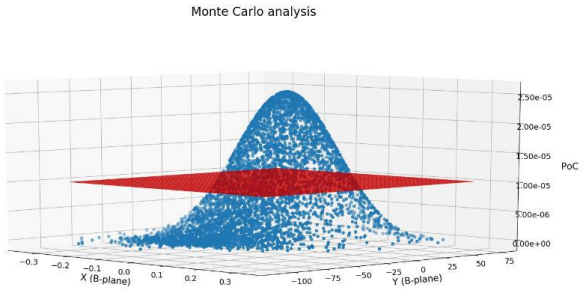


Figure 6. PoC population predicted by the Monte Carlo approach

Once the set of miss vectors has been generated, their associated PoC values can be computed considering the future position covariance matrices predicted previously. In this way, a population of the most likely future PoC values is obtained and the evolution of the risk can be characterized in terms of probability (probability of getting a future PoC value above a defined threshold during the analysis). This probability is defined as the share of points of the computed PoC population in the Figure 6 above the previous threshold.

An alternative approach, following GMV's interpretation of this method as a conditional probability method, is presented next. The set of future miss vectors can also be generated by meshing a region of the B-plane that contains the 3σ error ellipse defined by the latest combined position covariance matrix (see Figure 7). In that way, if the latest combined covariance matrix is realistic enough, it is ensured that most of the feasible future miss vectors are considered during the prediction. Just like for the Monte Carlo approach, a population of future PoC values can be obtained from the points of the mesh although, since these points are not equally likely, each member of the computed PoC population must be weighted. The weighted factors required can be obtained from the 2D Gaussian probability density function (PDF) associated with the latest combined position covariance.

Once the values of the PoC population have been properly weighted, the probability of getting a future PoC value above a defined threshold can be estimated as the weighted share of points above this threshold. This method based on creating a mesh of points is chosen to be implemented in ***autoca***. Both methods yield to the same results, with the advantage of the second being less demanding in terms of computation resources.

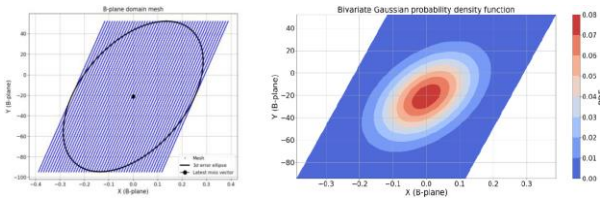


Figure 7. Uniform B-plane meshing and its associated of weighting factors for the weighted scattering approach.

Once the values of the PoC population have been properly weighted, the probability of getting a future PoC value above a defined threshold can be estimated as the weighted sum of points above this threshold. The computed PoC population can be characterized by its mean Probability of Collision (i.e., expected value), which is given by the Eq. (5):

$$\mu = E[P_c(X,Y)] = \iint P_c(x,y) f(x,y) dxdy \quad (5)$$

where X and Y are random variables associated to each component of the miss vector in the B-plane. This expected value represents the integral over the considered domain of feasible future miss vectors of the weighted probability of collision conditioned to a given future miss vector in the B-plane (integrand of the previous expression). Besides, since the probability of collision is a univariate random variable, the dispersion of its distribution can be measured computing the Eq. (6):

$$\sigma = \sqrt{var(P_c)} = \sqrt{E[P_c^2] - \mu} \quad (6)$$

The previous mean value and its dispersion can be used to provide a quantitative estimate of the expected value of the PoC in the future. In this way, this methodology is not only able to provide a prediction of the evolution trends of the PoC, but also to quantitatively forecast an estimated value of the risk in the future.

4.4. Operational implementation of collision avoidance manoeuvre

The strategy is to use AI/ML to estimate and predict criticality. Then, operationally, these results will be compared with accepted criticality thresholds, and combined with a systematic consideration of the relevant mission constraints (defined by the O/O) for the Collision Avoidance Manoeuvre recommendation (CAM). Appropriately using all this info allows to define a unified workflow for a software system that ultimately recommends the CAM decision to the operator (who has the final say), hence a software that reflects existing operational processes, but with increased level of automation and functionality. For this purpose, it is important to consider the following key aspects:

- Satellite operators only consider executing CAMs when the conjunction exceeds thresholds for specific criticality metrics (features) such as those previously described: probability of collision in different forms like standard PoC or Scaled PoC, miss distance and radial miss distance;
- Satellite operators try to wait as much as possible to perform CAM in order to make use of the information closest to TCA to take the right decision, as well as to minimize the time the satellite is taken out of its nominal operational orbit producing useful data for the mission.
- Most of the operational and platform constraints can be translated into two main types of timing constraints: they either impact when (and how) the CAM can be performed, or they define the latest time when the decision can be taken, to still be able to implement the CAM.

As such, it seems reasonable to evaluate how each constraint impacts the time when the manoeuvre can be performed or when the CAM decision can be taken, relative to the time when the close approach is estimated to take place. This analysis is done in a systematic way, analysing each constraint either individually or in combination for constraints that depend on each other.

Once all constraints have been evaluated in this manner, a timeline relative to TCA can be defined, to help establish the latest time of CAM decision (i.e. point of no return), based on which the software workflow can then be defined as well. The main operational and platform constraints which affect the time budgets relative to TCA are: CDM provider response time, satellite operator operational overheads, conjunction geometry, already planned manoeuvres, ground station contacts, forbidden locations or interval where the satellite should not go/cannot go, eclipsing periods, South Atlantic Anomaly, the type of secondary object (active and maneuverable satellite or not), the propulsion capabilities, AOCS constraints, and alternative communication paths to increase the opportunities to uplink the manoeuvre commands.

To use these constraints in the implementation of an automated collision avoidance software, the strategy is a systematic approach to first determine the optimum CAM for a given event and, based on the burn time of this optimum CAM, determine the latest CAM decision time. This is based on the fact that all the available information, in the form of received CDMs as well as

AI/ML criticality estimates, can be combined with those constraints that determine when and how CAM can be performed to obtain the desired reduction in criticality. Then, with a list of viable CAM options, the optimal one can be selected as that with burn time closest to TCA, as this would result in the latest decision time, allowing operators to wait as much as possible. With the burn time of the optimum CAM, a timeline can be defined backwards (from TCA towards present) as presented in Figure 8 using the constraints that impact the time when the final decision must be made, to compute this point of no return in the decision process.

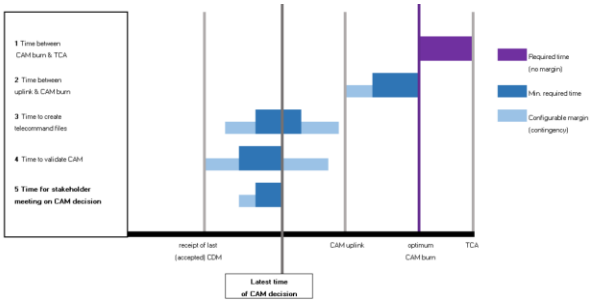


Figure 8. Simple representation of constraints-based time budgets in CAM timeline

The focus of the timeline is to show the constraints identified previously as critical to appropriately determining the latest time of CAM decision for a given event. Each of those processes are described next, going backwards in time:

Time between CAM burn & TCA. Essentially, this is the result of the selection of the optimum CAM through the filtering of the parametric analysis (previous step). The time budget to execute this burn is essentially given by the burn time of the optimum CAM; this gathers the effects of all the constraints identified previously that impact when and how the CAM can be executed. It is shown with no margins on the timeline because the reduction in criticality is estimated based on assuming this burn takes place exactly at that time, so this must be ensured operationally as accurately as possible.

Time between uplink & CAM burn. Different operators have different requirements for how much ahead of the actual burn time the command must be received on-board. This time is represented by the dark blue part in the plot, and the light blue part on its left indicates the potential increase if the best command opportunity is available only some quantifiable time before the start of this time interval, due to constraints imposed by available ground stations or alternative communications paths (if available).

Time to create tele-command files. This is evaluated several times, in the context of uplink constraints, AOCS constraints and pre-loaded CAM; the latter are disregarded here, as justified before. Creating these tele-command files involves rather standard processes, unless specific cases require a more tailored solution which could greatly increase the time. In any case, a time estimate for this would be provided by the operator as a configurable parameter; it is not the responsibility of the automated collision avoidance software to create the tele-command files. Once these are available, the uplink can be done in a very short time, hence not requiring an additional time budget. Furthermore, although these files are currently created after making the final CAM decision, but using the automated collision avoidance software it may be suggested to start this activity slightly before. This is because AI/ML processes for criticality estimation will lead to more reliable info than what is currently used operationally. More confidence in these means that if the software recommends to start creating the files, they will most probably be used in the end, so starting this would not be a waste of resources; in fact, it could allow delaying the moment when the final decision must be taken.

Time to validate CAM. This is one of the most important processes that typically requires collaboration with the original CDM provider to confirm that executing the optimum CAM would not create further high risk conjunctions; this check is currently the last input in the final CAM decision processes. One main hypothesis of the proposed implementation is that the screening volume initially requested by the operator to the CDM provider is large enough that the orbit obtained by applying the maximum allowable ΔV would still be within that initial screening volume if propagated for the screening period (e.g. 7 days). This would allow the operator to perform the validation independently from the original CDM provider; this could greatly reduce the time budget for CAM validation, which is also to be provided by the O/O as a configurable parameter.

Time for stakeholder meeting on CAM decision. This time is needed so that the different experts that are responsible for the satellite can meet to assess the recommendations made by the automated collision avoidance software and take the final decision.

5. ARTIFICIAL INTELLIGENCE: PROBLEM FORMULATION

Having a reliable prediction of the criticality of an event allows the human operator to make decisions in a more informed manner and can help with navigating operational constraints and lead to less costly manoeuvres. There are multiple approaches that can be employed in order to predict the criticality. An important aspect related to the available data is that conjunction events are described by updates (in the form of CDMs) in a sequential manner. In such a setting, models specifically tailored for sequential data can be employed.

5.1. Long Short Term Memory

Recurrent Neural Networks such as LSTM (Long Short Term Memory) allow both training and making inferences using sequences of arbitrary length. In the case of conjunction events, the input can be represented by the sequence of available CDMs. The basic unit of training is represented by an event (a sequence of data from the corresponding CDMs). Therefore, even though the events are analysed individually, the model learns to generalise, as it is exposed to a training set containing multiple events.

5.2. Covariance prediction

Prediction of the covariance elements of the secondary satellite by means of machine learning techniques represents one of the main goals of the application. These elements are further used to predict the PoC, based on which the CAM decisions are based.

The covariance prediction as part of the CA analysis is a key feature that allows satellites' operators to anticipate the expected collision risk, avoiding the planning of CAM strategies that are not really needed or with a lower cost in terms of delta V when they are really needed.

The three main diagonal positional elements of the covariance matrix in RTN reference frame are selected for the prediction. The model has been trained for four different prediction time-intervals of the covariance elements:

- 24 hours after the last available update.
- 48 hours after the last available update.
- 72 hours to the time of close approach.
- At the time of closest approach.

5.3. Attention Mechanism

In order to increase the performance given by the LSTM model, an attention mechanism was proposed. The core idea behind the attention mechanism is that it allows the LSTM model to selectively focus on valuable parts of the evolution of an event and therefore, learn the association between them.

Given the implied task of forecasting values based on several timeframes, an attention mechanism was proposed for testing to assess its performance impact on the several types of predictions. The mechanism has been proposed due its ability to solve the biggest problem in seq2seq tasks, which consists on decoding a variable length sequence based on a single context vector. It enables the utilisation of all hidden time steps of the input sequence during the decoding process.

There are two types of attention mechanisms proposed to be tested in the current section. The first one is proposed by *Bahdanau et all* [5] which is formally called additive attention, the second being an attention mechanism proposed by *Luong et all* [6]. The main difference between them is how they score similarities between the current decoder input and encoder outputs.

In order to minimize the effect of hyper-parameters on the performance, the models were trained using the same hyper parameter space. For each draw of random values within the hyper-parameter space, all three models were trained (the Luong attention, Badhanau and a plain LSTM model)

While experimenting, there were prediction types where the Luong attention was not able to provide any usable result, obtaining a Figure of Merit (metric explained in the next section) score below zero. Because of this the models with Luong attention from further data analysis were discarded for this application.

6. PRELIMINARY ANALYSIS RESULTS

6.1. Dataset filtering

The pre-processed dataset is composed of a collection of 151106 events, consisting of a total of 2095280 CDMs that are originated by the 18th SPCS for LEO satellites. A distribution of the number of CDMs/event of the pre-processed dataset can be seen in the Figure 9.

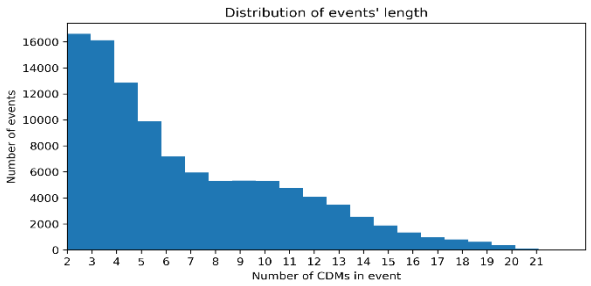


Figure 9. Distribution of number of updates per event

After filtering the dataset, it was observed that 20% of events only had one update, around 40% of the updates are meaningful for the secondary object and the time between meaningful updates is of 15 hours in median.

6.2. Dataset Scaling

The figures presented in this section illustrate the initial dataset and the result of applying Quantile Transformer. Figure 10 and Figure 11 illustrate the unscaled distribution of the data

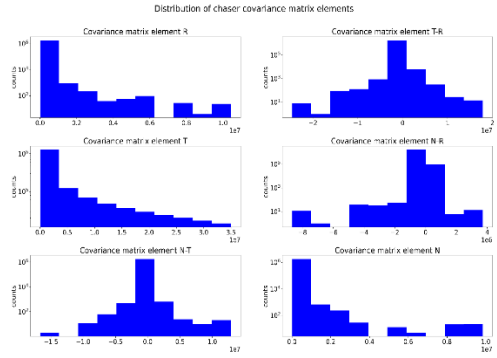


Figure 10. Features used as input for ML: unscaled covariance matrix elements

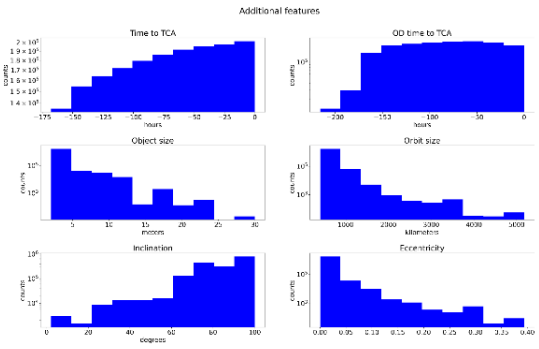


Figure 11. Distribution of CDM parameters.

Figure 12 and Figure 13 illustrate the Quantile Transformer scaled distribution of the data

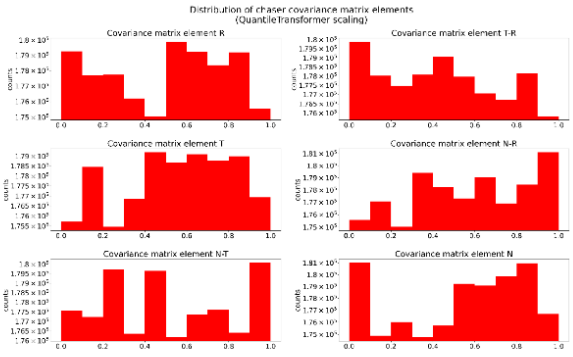


Figure 12. Covariance matrix elements features scaled with Quantile Transformer

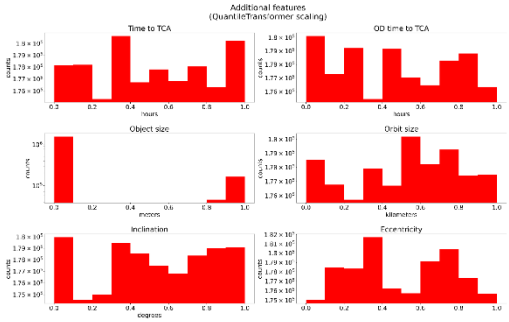


Figure 13. Additional features scaled with Quantile Transformer

6.3. Metric used: FoM

For evaluating model results, a metric called Figure of Merit was defined in Eq.(7) , which is designed to represent objectively the model performance on a given validation dataset.

The range of some values even after filtering outliers is large. The logic behind this metric is that a robust metric that would provide representative information with respect to the model performance is needed, while weighting the outliers less.

$$FoM = 1 - avg\begin{pmatrix} med(rel_{err_{t1}}), med(rel_{err_{t2}}), \\ \dots, med(rel\_err_{tn}) \end{pmatrix} \qquad (7)$$

For each target label that the model is trained to predict, the relative error of the predicted value with respect to the target value is computed. The median is selected from the distribution formed by the relative errors for a specific target throughout all the events. The median was chosen instead of the average because the median represents the middle score for a set of data arranged in order of magnitude, thus being less affected by outliers or skewed data. The medians are averaged, which gives us an average “median” relative error.

Finally, the result value is subtracted from 1 in order to normalize the results. This step ensures a value that is human readable. The upper bound value is 1, which indicates no error within the model prediction.

6.4. Covariance Prediction Results

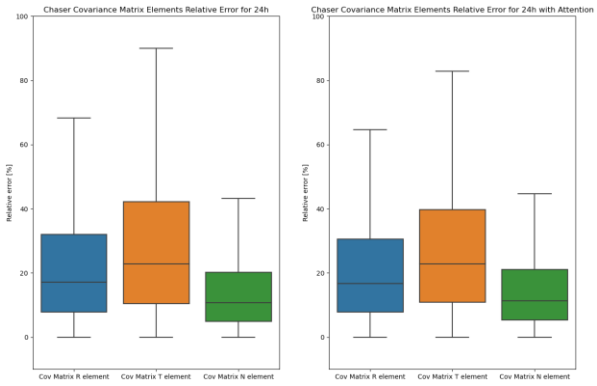
In order to find the Machine Learning models with the highest performance, large empirical tests were performed on the LSTM and the LSTM with attention mechanism models.

The tables and figures presented in this section illustrate the results in terms of FoM for all covariance prediction cases:

- 24 hours after the last available update
- 48 hours after the last available update
- 72 hours to the time of closest approach
- Time of closest approach.

6.4.1. 24 hours after the last available update.

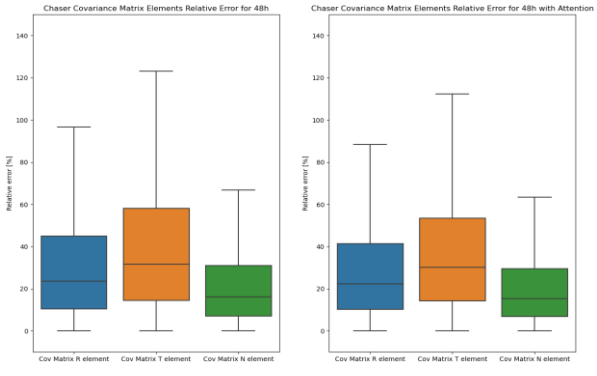
24h covariance evolution model	LSTM	LSTM with Attention
FoM score	0.826	**0.828**



From the error spread figure, a mild improvement can be seen in the spread of the relative error for the T element, which represents the hardest element to predict, as well as mild improvements in the R and N elements.

6.4.2. 48 hours after the last available update.

48h covariance evolution model	LSTM	LSTM with Attention
FoM score	0.805	0.806

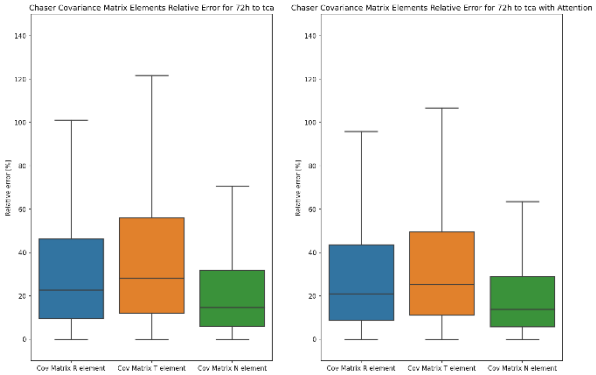


In this prediction case, the spread of the T element relative error in the Bahdanau attention is higher but the median is lower. The same pattern can be observed for the R and N element's relative error.

This is visible in the table of results, where the Badhanau attention obtained a slightly higher FoM.

6.4.3. 72 hours to the time of closest approach.

72h before TCA covariance model	LSTM	LSTM with Attention
FoM score	0.852	**0.861**

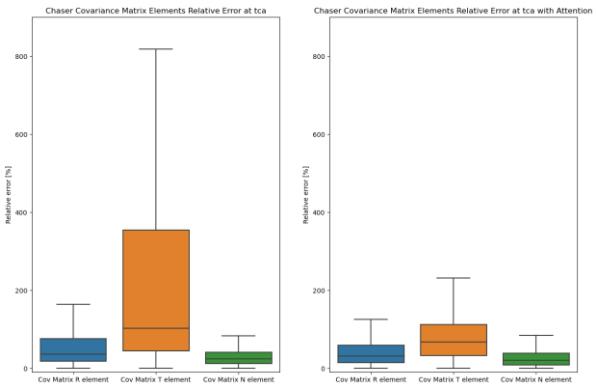


In the 72 hours to TCA case, both the spread and median of the relative error for all elements offer an improvement.

This is visible in the table of results, where the Badhanau attention obtained a slightly higher FoM.

6.4.4. Time of closest approach

Covariance at TCA model	LSTM	LSTM with Attention
FoM score	0.513	**0.702**



It is expected that the attention mechanism to achieve the highest performance increase when predicting at a

far point in time such as predicting the covariance elements at the time of the closest approach.

From both the figure and the results table, the attention model outperformed the simple LSTM model by a considerable margin.

From the error spread figure, a significant difference in the spread of the relative error can be seen. Furthermore, one can notice a substantial difference in the median of the relative errors, especially in the hardest element to predict, the T element.

7. CONCLUSIONS

The number of high-risk conjunction events will scale progressively and thus the collision avoidance manoeuvre automation is required.

The probability of collision and predicting its evolution over time have a major contribution to the collision risk assessment.

The operational constraints strongly condition the collision avoidance maneouvre (CAMs) design process and the decision to perform a manoeuve is taken depending on the criticality when the latest time of CAM decision is reached.

As an outcome of the large scale empirical tests performed, one can observe that the use of the attention mechanism, proposed for time series tasks, has achieved a higher performance compared to the fine-tuned classic LSTM models.

The Machine Learning preliminary results serve as a starting point within our mission to achieve an autonomous CAM decision software, having a peak of 0.861 in terms of Figure of Merit.

The LSTM model enhanced with the attention mechanism has improved the highest accuracy for the four covariance prediction cases.

8. REFERENCES

[1] 18th, Space Control Squadron Joint Force Space Component Command. *Spaceflight safety handbook for satellite operators.* 2019.

[2] D., Escobar Antón, Pérez Cambriles A., Martinez Fadrique F.M., and Águeda Maté A. “closeap: GMV’S solution for collision risk assessment.” *ESA, 8th European Conference on Space Debris.* 2009

[3] Laporte, F. “Scaled PoC: Definition of a Formal Process, part 1. .” *CNES Presentation on Scaled PoC.* n.d.

[4] D. Aguilar, P.L. Righetti, D. Lázaro , "Next-day conjunction risk prediction at EUMETSAT"

[5] Bahdanau, Dzmitry & Cho, Kyunghyun & Bengio, Y.. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. ArXiv. 1409.

[6] Luong, Minh-Thang, Pham, Hieu and Manning, Christopher D. "Effective approaches to attention-based neural machine translation." *ArXiv preprint arXiv:1508.04025*

[7] *Akella, M. R. and Alfriend, K. T., 2000. The Probability of Collision Between Space Objects. In: Journal of Guidance, Control and Dynamics, 23(5). DOI:* 10.2514/2.4611

[8] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. 2011. “Scikit-learn: Machine Learning in Python”. JMLR 12, pp. 2825-2830

[9] Quantile transformer. *Scikit-learn Documentation.* [online] Available at: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html

[10] Non-linear transformation. *Scikit-learn Documentation.* [online] Available at: https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-transformer