ESA'S COLLISION RISK ASSESSMENT AND AVOIDANCE MANOEUVRES TOOL (CORAM)

Juan Antonio Pulido[(1)], **Noelia Sánchez**[(2)], **Ignacio Grande**[(3)] **and Klaus Merz**[(4)]
[(1)(2)(3)]*Elecnor Deimos Space, Ronda de Poniente, 19, 28760 Tres Cantos, Spain*
[(4)]*ESA/ESOC Space Debris Office, Robert Bosch Str. 5, 64293 Darmstadt, Germany*

Abstract: *The protection of active satellites from impacts by space debris objects is a two-stage process which requires, as first step, the identification and assessment of conjunction events and, as second step, the design and execution of avoidance manoeuvres. In order to assist ESA's Space Debris Office (SDO) with these tasks, Elecnor Deimos has developed CORAM, a custom-made Collision Risk Assessment and Avoidance Manoeuvre computation tool fully integrated within the SDO's operational environment. CORAM implements the latest developments in the algorithms for computation of collision risk as well as an avoidance manoeuvre algorithm for the design of the best suited avoidance strategy. This paper describes CORAM's architecture, algorithms and capabilities that will help ESA's space debris analysts in the collision avoidance problem. It presents also some examples of collision avoidance scenarios.*

Keywords: *Collision Risk Computation, Encounter Mitigation Manoeuvre*

1. Introduction

The CORAM SW package has been developed to support satellite operators in the evaluation of conjunction events regarding the computation of collision risk and the analysis of suitable avoidance manoeuvres. CORAM is integrated in ESA's operational environment, and has been designed and developed to be complementary to the operational Collision Risk ASSessment tool, CRASS [1], which is the first stage in the conjunction analysis at ESA's SDO. CRASS screens the Two-Line Elements (TLE) received from the USSTRATCOM, propagating the objects' orbits along the analysis time interval and checking if close encounters occur with the operational satellite under study. In addition to this general screening, SDO receives specific conjunction summary messages (CSM) from the JSpOC for the operated spacecraft (obviously in this case the screening process is not needed). CRASS implements the risk algorithm developed by Alfried and Akella [2].

CORAM complements CRASS' functionalities, since on one hand, it implements additional risk computation algorithms for a refined assessment of the collision risk. On the other hand, CORAM can analyse different manoeuvre avoidance strategies in order to help in the selection of the most suitable strategy. These two main functionalities are implemented in two different SW modules:

- CORCOS (COllision Risk COmputation Software) is devoted to the computation of collision risk between two objects: a piece of debris with an operative satellite, an operative satellite with another satellite, or even between two pieces of debris (this last case will not allow any avoidance strategy to be undertaken). CORCOS can analyse orbits with high relative velocity (where some assumptions in the risk function allow fast computation algorithms), but also for low relative velocities (which require more complex algorithms). Depending on the input settings, CORCOS can analyse an identified conjunction event, or can be configured to search for all close encounters of a

target-chaser pair in a given time interval and analyse the collision probability of each encounter afterwards.

- CAMOS (Collision Avoidance Manoeuvre Optimisation Software) is devoted to the evaluation of different mitigation strategies through the optimisation of avoidance manoeuvre parameters. The optimisation problem can be defined in a very flexible way, e.g., the risk function can be minimized for a given (fix) manoeuvre size, or the minimum delta-V can be computed such that a defined risk level is achieved.

In the following sections both tools will be described in terms of architecture, algorithms, and capabilities. After that, some interesting examples will be presented.

2. Software Architecture

The CORAM SW suite is made of two main modules, CORCOS and CAMOS. Both tools are expected to be used in the risk mitigation of an encounter event. A typical operational sequence is (see Fig. 1):

- CRASS detects an encounter between the operated satellite and a piece of debris, or a conjunction summary message (CSM) is received from the JSpOC.
- CORCOS is used to refine the risk value computed by CRASS.
- If the collision probability is too high, CAMOS is executed to analyse different avoidance strategies.
- The mission analyst selects one case in the analysed strategies to be further studied for implementation. CAMOS is run for a second time, now for this selected case only, producing manoeuvre information used by CORCOS.
- The new conditions of the conjunction event/s, considering the designed manoeuvre/s are analysed with CORCOS.

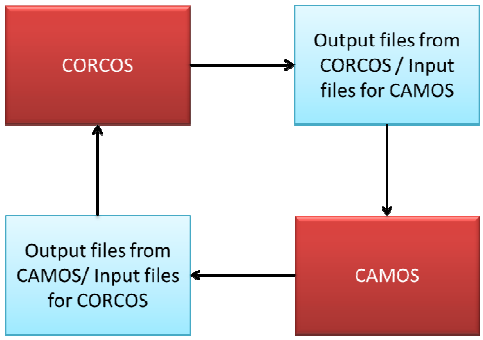

Figure 1. Operational relationship between CORCOS and CAMOS

The second CORCOS execution is required not only because CORCOS provides more information of the encounter/s, but also because some risk computation algorithms not available in CAMOS (e.g., Monte Carlo) can be used to assess the effect of the manoeuvre/s.

In addition to the operational relationship, there is an important dependence between CORCOS and CAMOS, in the sense that CAMOS re-uses several functionalities from CORCOS (orbit

initialisation and propagation, probability computation function, etc), as will be explained hereafter.

2.1. CORCOS Architecture

The CORCOS execution flow is shown in Fig. 2, which shows the main functions, called in a sequential way to perform the risk computation, with the following steps:

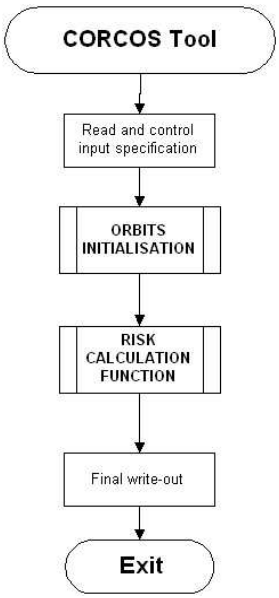

Figure 2. CORCOS general structure chart

The CORCOS execution flow is shown in Fig. 2, which shows the main functions, called in a sequential way to perform the risk computation, with the following steps:

- At process start the input files will be read and their data checked for consistency to enable a successful software execution.
- Next, the Orbits Initialisation Function will be invoked, for preparation of the input data in a consistent form for ulterior utilization within CORCOS.
- Once the state vectors and covariances are prepared and locally computed ephemeredes are calculated, the Risk Calculation Function (RCF) is called. This is the main functionality of CORCOS that may also be invoked by CAMOS. Provided with the set of input parameters previously described, RCF computes the set of closest approaches, which is the set of times of closest approaches (TCA), distances at TCA, and the corresponding collision probabilities. Different collision probability algorithms, as well as a Monte Carlo approach are available.
- Before ending the process, the outputs are delivered in appropriate formats, and plotting scripts are generated for a further visualization of the results, allowing the user an easy understanding of the outcome.

2.2. CAMOS Architecture

The CAMOS general structure chart can be seen in Fig. 3. It shows the main SW modules in hierarchical order from left to right. These are:

- Main program function. Performs process initialisation and termination activities, as well as high-level process management.
- Strategy analysis module. Manages the strategies: number of defined strategies, configuration of each one: number of manoeuvres, definition of each one, treatment of parameters (strategy or optimisation parameters).
- Optimisation module. Gradient projection optimisation algorithm.
- Optimisation parameter and function module. It is the link between the optimisation algorithm and the physical world, i.e., the trajectory and probability functions.
- Trajectory module. Manages the trajectory: initial state and covariance, arc propagation and storage.
- Probability computation function. Implements the different algorithms for computation of collision risk.
- Propagation function. Orbit propagation, arc by arc.

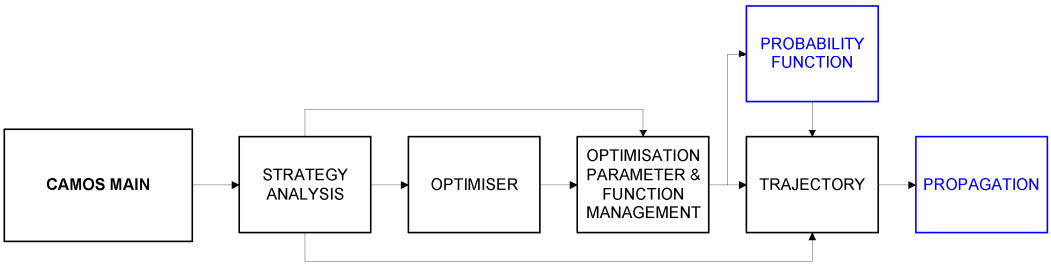

Figure 3. CAMOS general structure chart

Probability computation and propagation functions (shown as blue rectangles in Fig. 3) are re-used from CORCOS. The Trajectory module uses also other functions from CORCOS (orbits initialisation and encounter time search and refinement).

3. CORCOS Module Description

CORCOS module is the tool responsible for input/output of scenario files, propagation, close encounters searching and collision risk assessment. It also contains all the supporting libraries for reference frame transformation and mathematical routines.

CORCOS has been designed as a versatile tool, supporting different types of inputs to configure the scenario to be analysed using several collision risk algorithms, with support for high-speed and low-speed encounters involving spherical or complex objects:

- Orbit information for each object can be given as a state vector in one of the several reference frames supported, ephemeris file to be interpolated, a TLE file or using a CSM

as input. Each object can use a different input that is automatically managed in the software.

- Covariance information, used to compute the collision risk, can be given as a covariance matrix, a look-up table for TLE objects or read from a CSM file. Covariance is propagated together with the state vector for any input type to the time of closest approach, and its epoch may differ from the orbit epoch.
- Regarding the propagation capabilities, CORCOS uses a force-model propagator with all the perturbations needed to provide an accurate propagation, configurable as input. It also supports instantaneous and low-thrust manoeuvres, with modelled execution errors. Both Runge-Kutta 7(8) and Adams-Bashforth integrators are available, and the propagator is used also for the covariance. In addition, SPG4 is used to propagate TLEs, switching to the force-model propagator if a manoeuvre is configured.

Once all conjunctions inside the configured time span have been identified/detected, CORCOS will evaluate the conjunction geometry and the collision risk associated. Several algorithms are available, depending on the collision speed and the geometry of the objects.

3.1. Spherical Geometry

If both objects are spherical, there are several well-known algorithms that can be used:

- Alfriend & Akella (see [3]) a well-known method to compute collision risk that performs the two-dimensional integration of the hard body projection in the encounter plane.
- Patera's method [4] performs the contour integration of the projection, computing the same result as the Alfriend & Akella method in a faster way.
- Maximum Probability, assuming spherical covariance, using the maximum likelihood approach [3]. Figure 4 shows the existence of such a maximum for every encounter distance.
- Maximum probability according to Klinkrad's algorithm [9].
- Covariance scaling, where the covariance is scaled for both objects in a given interval and for every scale factor, the covariance is evaluated using the method in [4]. This method preserves the shape and orientation of the covariance matrix of each object and it is useful when the covariance is not well-known.

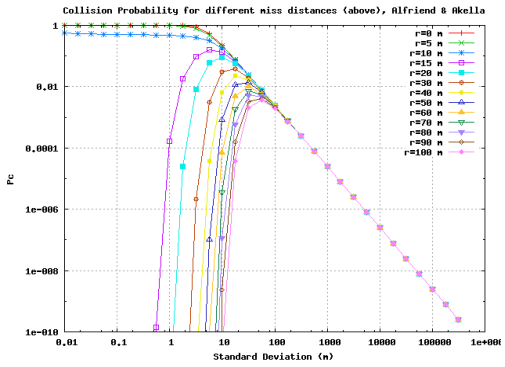

Figure 4. Evaluation of performance for different encounter geometries and orbital accuracy

During CORAM development, these algorithms, and some other finally discarded (Chan, Alfano and Foster), have been tested to check the performance, both in terms of run-time and accuracy (see Fig. 5). Additionally, extensive analysis of performance under different conditions of geometry and covariance values was executed (see Fig. 4).

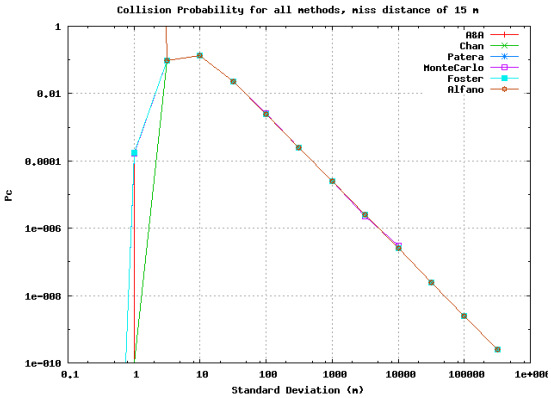

Figure 5. Comparative results of some collision risk algorithms for spherical case

3.2. Complex Geometry

If one of the objects, or both, are complex (composed of oriented boxes), a new method to calculate the collision risk has been devised.

While in the spherical case the hard-body object (collision volume) can be computed as another sphere whose radius is the sum of the radii of the two original spheres, in the complex case this hard body computation is more complicated. It is accomplished by assuming constant attitude and calculating the Minkowski sum [7] of the two objects, and then projecting it onto the encounter plane. Additionally, the collision volume shall be translated to the B-plane, by means of the projection of the vertices of such volume, as shown in Fig. 6.

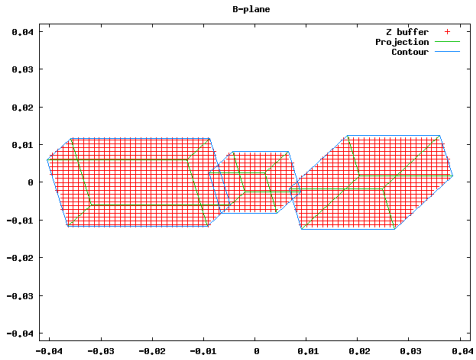

Figure 6. Representation of encounter plane with the projection of the boxes forming the collision volume (one satellite built by three boxes, and the other based on a unique box), and the Z-buffer evaluation

The encounter plane is then discretized and sampled. A z-buffer grid [8] is constructed where every cell of the grid is a true/false indicator of the "shadow" of the hard body onto the encounter plane. Every grid contains a small amount of contribution of the collision risk and the last step is to compute the risk associated to every shadowed grid and sum them up.

The need of considering the actual objects geometry instead of assuming spherical case is very much dependent on the miss-distance and the values of the covariances of the orbital data. The following examples provide some graphical representation of a head-on encounter of two satellites (one satellite built by three boxes, and the other based on a unique box). Geometry of the encounter is shown in Fig. 7. Dashed line indicates the equivalent cross-section area assuming spherical geometry whereas the green circle is the nominal encounter point.

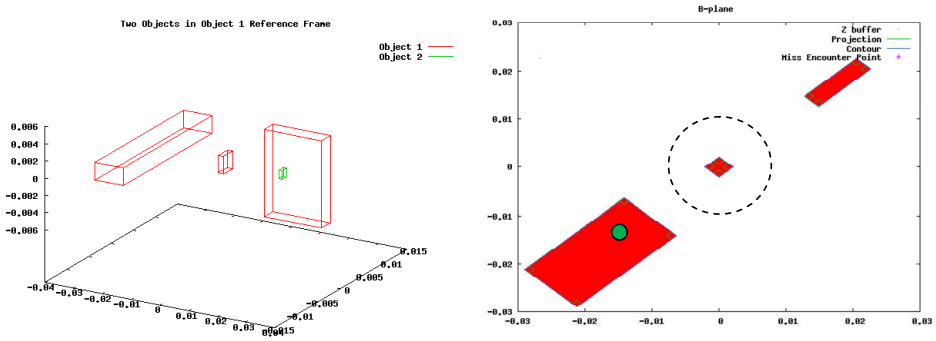

Figure 7. Example encounter geometry (left) and B-plane representation (right)

In the case of accurate orbital data (small covariance values, about 1 m), the miss-encounter would be perfectly estimated with a high accuracy, only considering the actual geometries of the objects. Otherwise, the integration of the risk along the spherical projection would provide a very low collision risk. This case is represented in Fig. 8, left plot. The computed collision probability with the complex-geometry algorithm here described is 0.9959, whereas the collision risk computed by algorithms based on spherical assumptions is $1.49 \cdot 10^{-14}$.

In the case of larger uncertainties in the orbital position of the two objects (about 100 m), the probability density function is spread across larger areas of the B-plane (Fig. 8, right plot), providing very similar results when integrating the risk along the actual object geometries than integrating the risk along the equivalent circle. The computed risk is $2.14 \cdot 10^{-3}$ for the two cases.

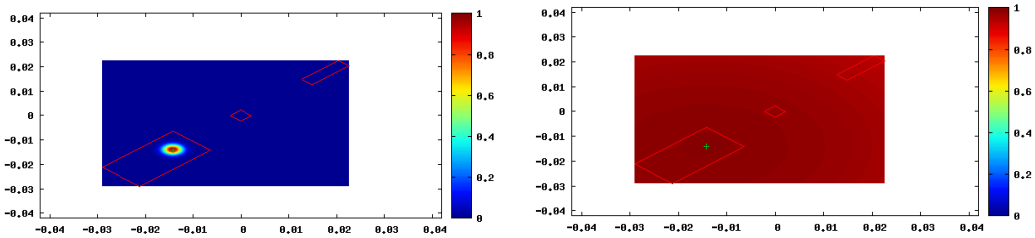

Figure 8. Probability density function for the case of very good orbit accuracy (~1m, left) and low orbit accuracy (~100m, right) along the B-plane

3.3. Minkowski Sum

To easily compute the Minkowski sum of two complex objects, it is better to divide the objects in convex shapes and compute the sum by pairs, for all combinations and then reconstruct the final object. However, the actual 3D object calculation is not required, only its projection onto the encounter plane. It is possible to skip the 3D reconstruction of the Minkowski sum and calculate the projection directly.

For that, the Minkowski sum is computed for every two boxes (or box-sphere) of the objects but only for the vertex points, without reconstructing any information about the faces. The resulting sum will be also convex.

Those points are then projected onto the encounter plane, and the convex hull that the points form is calculated. This convex hull is the contour of the projected Minkowski sum, represented by convex closed irregular polygon.

The entire z-buffer is checked to evaluate what cells of the grid are inside the polygon. Only cells not previously shadowed by other polygon are checked by means of a fast point-in-polygon algorithm.

These steps are repeated for every box-box pair of the complex objects, and the resulting z-buffer grid is evaluated to calculate the collision risk.

In order to do that, it is possible to use Alfriend & Akella or Patera methods on each cell. It can be easily done by replacing every cell by an equivalent circle in the encounter plane and applying a collision risk method to them. The final sum provides the total collision risk.

The z-buffer offers several advantages:

- It is relatively fast.
- It solves the problem of self-shadowing, where different parts of the objects can be accounted several times in the computation of the collision risk. The z-buffer cells have only two states (in shadow / not in shadow) it is not possible to have overlapped sections counting twice.
- It can be easily extended to include other basic shapes, as long as they are convex or could be divided in convex shapes.
- Allows calculating the cross-section of a complex body from a certain point of view, which can be used to estimate the area exposed to atmospheric drag or solar radiation pressure.

3.4. Low Speed Encounters

The previously commented methods are in principle applicable only to high-speed encounters, where a linear relative motion and constant orientation, cross section and position uncertainties during the encounter can be assumed.

In a low-speed encounter, however, the conjunction parameters may change in time and it is not possible to evaluate the risk just at the time of closest approach, it is necessary to take into account the whole encounter interval.

An interval-slicing method based on Patera's work [5] has been employed. The method divides the collision interval in slices, and the collision risk is evaluated for every slice. For each slice, the same assumptions as in the high-speed encounter are valid (constant covariance and orientation, linear motion) and the slices can be made as small as necessary for these assumptions to be correct.

To calculate the collision risk of each slice, any other high-speed collision risk algorithm can be used, with a scaling factor to take into account only the contribution of the slice, and not the whole encounter. This means that this method can be used with spherical objects and also for complex geometry objects.

The instantaneous risk (red curve in Fig. 9) computed at each slice may be larger than the final computed risk along the interval, since it accounts at every slice as if the out-of B-plane component of the miss-distance is null. Once this fact is properly accounted to evaluate the instantaneous Pc rate (green curve), the cumulated risk can be derived.

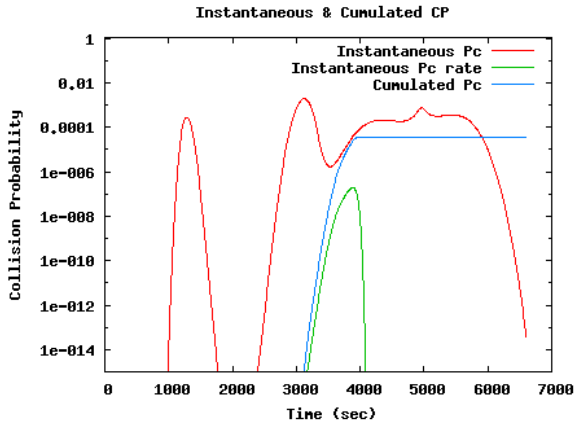

Figure 9. Example of accumulated collision probability along an encounter interval in the case of low speed

3.5. Monte Carlo

In addition to the analytical or semi-analytical methods previously described, CORCOS can also simulate the encounter using a Monte Carlo approach, valid for low-speed and high-speed encounters and with any geometry combination. This simulation, however, is much slower than other methods and the main use is to check the results of other methods or to avoid the propagation of the covariance matrix. An example of Monte Carlo use in included in Fig. 5 with the rest of algorithms.

The collision detection problem involving complex geometries has been solved using the separating axis test, [6], a very fast test valid for arbitrarily oriented boxes.

The user may select the number of steps for the Monte Carlo simulation, or alternatively, the user may configure the accuracy and confidence value to estimate the number of runs automatically. A Wilson score test is used to determine the confidence interval of the result.

4. CAMOS Module Description

As already mentioned, the computation of the optimal avoidance manoeuvre is performed by the CAMOS module. CAMOS uses most of the functionalities developed for CORCOS:

- Trajectory initialisation (state vector and covariance).
- Orbit acceleration modelling and propagation.
- Object properties initialisation.
- Encounter time search and refinement.
- Collision risk computation, both for low and high speed encounters. Only the analytical methods are used, due to the requirements of the optimisation algorithm described in the following paragraphs. Monte Carlo cannot be used by CAMOS, while complex geometries can be used only if the probability function is evaluated just as output (not as cost function or constraint).

Operationally, CAMOS is usually run once a close encounter between two objects has been analysed by CORCOS, and the obtained collision risk is high enough to deserve the study of an avoidance strategy.

CAMOS can be run in two modes:

- **Parametric analysis mode**. This mode can assess one or several strategy analyses, where *strategy analysis* should be understood as a one-dimensional or two dimensional parametric execution of a manoeuvre optimisation problem. This mode allows the user to evaluate, e.g., the effect of the manoeuvre execution time on the collision risk, with optimised manoeuvre direction for each selected value of the manoeuvre execution time in the grid. As example, Fig. 10 shows the effect of a 1-cm/s manoeuvre on the distance of closest approach (DCA) as function of the execution time, and Fig. 11 shows the effect of a manoeuvre on the distance of closest approach (DCA) as function of manoeuvre size and of the execution time.
- **Evaluation mode**. It runs just one case within one strategy, and produces specific output files to allow CORCOS to evaluate the selected case (with the newly designed manoeuvres) with risk methods not available to CAMOS. The user will usually run CAMOS in evaluation mode for the most interesting cases found by running CAMOS previously in parametric mode. Only one case can be evaluated at a time. In addition, this mode can produce optional information on the evolution vs. time of certain trajectory functions, like longitude, latitude, eclipse or location over the South Atlantic region. As example, Fig. 12 shows the evolution of the longitude of a GEO satellite while Fig. 13 shows the location over the South Atlantic Anomaly for a LEO satellite.

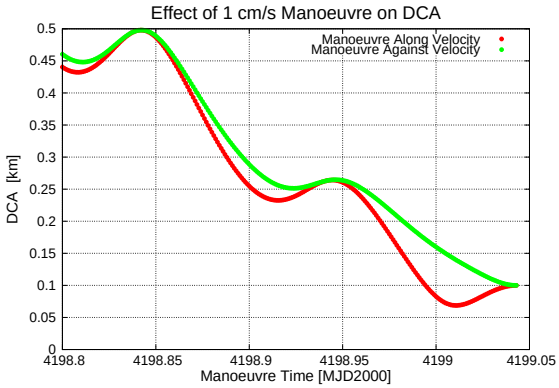

Figure 10. Example of one-dimensional parametric analysis results

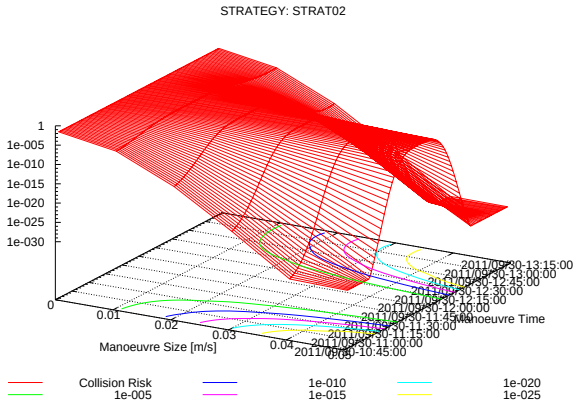

Figure 11. Example of two-dimensional parametric analysis results

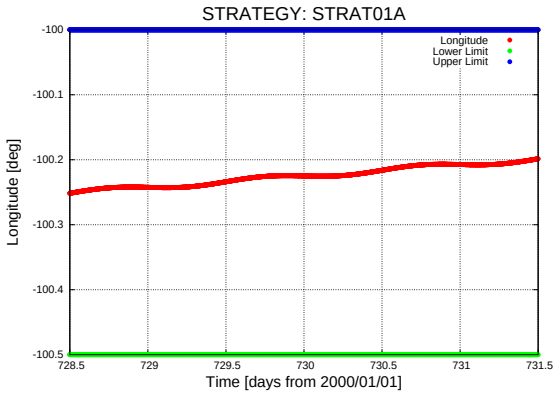

Figure 12. Longitude of a GEO satellite vs. time

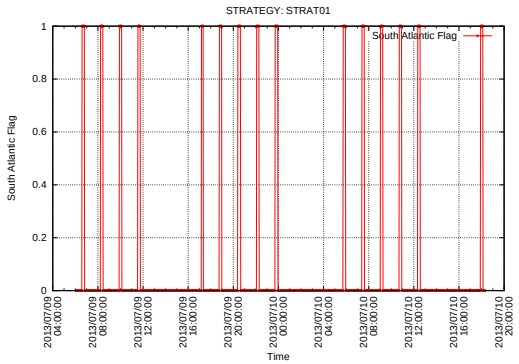

Figure 13. Pass over the South Atlantic region for a LEO satellite

The configuration of each strategy analysis is very flexible:

- Manoeuvres can be low-thrust or impulsive
- Manoeuvre directions can be provided in different reference systems:
  - Mean Earth equator of epoch J2000.0
  - True equator and equinox of date
  - Mean equator and equinox of date
  - Local orbital (radial, in-track, cross-track)
  - Local intrinsic (along-velocity, momentum, binormal)
- Each manoeuvre parameter (manoeuvres central time, size, azimuth and elevation) can be defined as fixed, a parameter of the strategy analysis, or an optimisation parameter
- Bounds can be set on manoeuvre parameters, and specific direction constraints can be configured
- Within the optimisation, the cost function can be selected as the collision risk, the total delta-V or the distance of closest approach (separation vector modulus, or its projection in along-track, cross-track or radial direction)
- Constraints can be set-up in the resulting trajectory: longitude and latitude for GEO satellites, and orbital period and ground track drift for LEO satellites.

CAMOS uses a gradient optimisation package called OPTGRA [10], developed by ESA/ESOC/Flight Dynamics, to find the optimum manoeuvre parameters in each configured problem. The algorithm can deal with equality and inequality constraints. It looks for the optimum solution by moving the initial optimisation parameters tangential to the constraints, and in the direction of steepest descent of the cost function.

Since gradient methods are local optimisation techniques, the solutions found by the algorithm must be understood as local optima and, therefore, must be analysed critically by the analyst in search of the global optimum. For example, manoeuvre execution times have an effect on collision risk that can have a certain sinusoidal component (with its period equal to the orbital period). In that case, the gradient optimisation algorithm would select the local optimum closest to the initial manoeuvre time. In any case, since the tool allows analysing several strategies in one run, each with different selection of strategy or optimisation parameters, the presence of such local optima can be investigated by selecting the manoeuvre time as a strategy parameter instead of an optimisation parameter.

5. Example of Avoidance Manoeuvre Design with CAMOS

In this section several examples of manoeuvres strategies analysed with CAMOS will be presented, starting with simple cases and increasing the difficulty to demonstrate CAMOS capabilities.

5.1. One Encounter, one Manoeuvre Case

The first example is a simulated encounter between two objects, both on MEO orbit, co-planar, with same semi-major axis and eccentricity, and in contra-rotating orbits, resulting in repetitive frontal approaches. Both trajectories are contained in the Y=0 plane. The orbit size is 8000x10400km. The encounters occur at perigee and at apogee. A 2D and 3D view of the trajectories can be seen in Fig. 14.

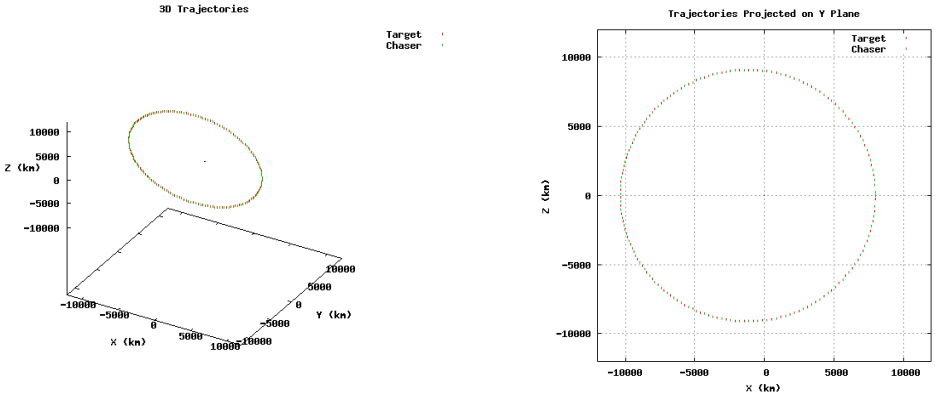

Figure 14. Synthetic case target and chaser trajectories

To start with a simple case, let us focus on one encounter, corresponding to the ascending node of the target at X=8000 km, corresponding to the perigee of both orbits. We will define a strategy with a 1-cm/s manoeuvre along the velocity vector (e.g., representing an operational case in which only tangential manoeuvres are allowed, with a minimum size of 1 cm/s). The manoeuvre execution time is selected as parameter, with the analysis interval spanning from one day before TCA to just a few seconds before TCA. Fig. 15 show the collision risk (computed with Patera's algorithm) and distance at closest approach, respectively. This simple, one-dimensional case is computed relatively quickly by CAMOS, and helps the mission analyst in evaluating possible manoeuvre locations for highest effectiveness.

The behaviour shown in Fig. 15 is easy to explain: when the manoeuvre is located around apogee, it is most effective in changing the perigee radius, and the DCA is highest. When the manoeuvre is located at perigee, it has no effect in changing the own perigee radius, and the DCA is lowest and collision risk highest. The effect repeats with the orbital period (around 146 minutes).

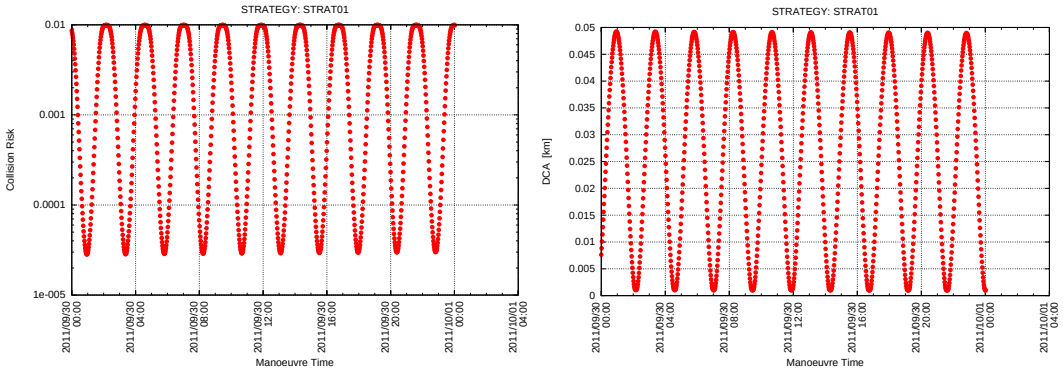

Figure 15. Collision risk (left) and distance at closest approach (right)

CAMOS shows also other results of the parametric analysis, like the change in the TCA and the separation vector at TCA projected in the along-track, cross-track, and radial directions. Fig. 16 left shows radial component of the separation vector at TCA. It can be seen that distance at closest approach corresponds, almost entirely, to the radial component, with the negative sign indicating that the chaser is bellow the target at TCA. On the other hand, since the target's orbital period is increased, the encounter is delayed in time, as can be seen in Fig. 16 right, where a positive value signifies a delay in the TCA. A linear trend with a super-imposed periodic component can be seen in this last figure. The linear trend is obviously due to the fact that, the earlier the manoeuvre is executed, the larger is the accumulated delay up to TCA for a given change in orbital period. The periodic effect is due to the fact that, for a fixed delta-V, the highest orbital period change is obtained when the manoeuvre is executed at perigee.

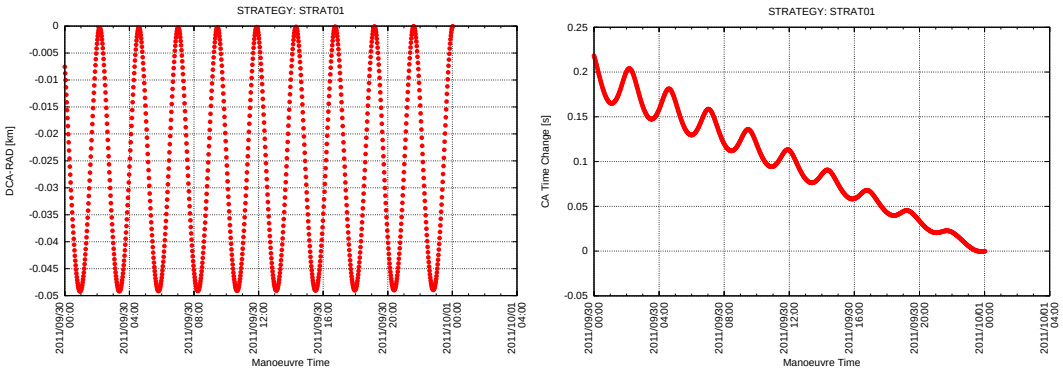

Figure 16. Radial component of separation vector (left) and change in TCA (right)

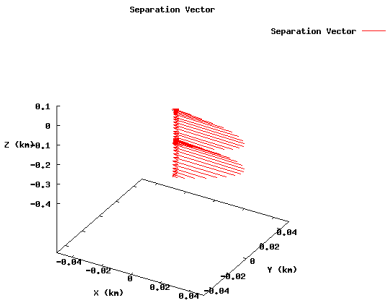

Figure 17. Separation vector in inertial coordinates

A 3D view of the separation vector evolution is shown in Fig. 17 in inertial frame, for the manoeuvre location up to two revolutions prior to encounter. The separation vector is contained in the Y plane, and is parallel to the X axis. The early manoeuvres produce a delay in TCA, that is, the location of the closest approach moves towards negative values in Z direction.

The second and third avoidance strategies analysed in this example consider manoeuvres normal to the velocity. In one case, the manoeuvre is in the out-of-plane direction, while in the other case the manoeuvre is in the in-plane direction. The results can be seen in Fig. 18, where the out-of-plane manoeuvre effect is shown in red. This manoeuvre produces a rotation of the orbital plane around the line joining the orbital centre with the manoeuvre execution point. The result is a cross-track effect which is minimum (in absolute terms) when the manoeuvre is executed around the encounter location and half a revolution earlier, and maximum a quarter and three quarter of a revolution earlier. The manoeuvre has no effect in radial direction.
On the other hand, the in-plane manoeuvre produces a change in eccentricity and argument of perigee, which translates into a radial effect at TCA (see Fig. 18 right, in green), and no effect in the cross-track component (left plot).

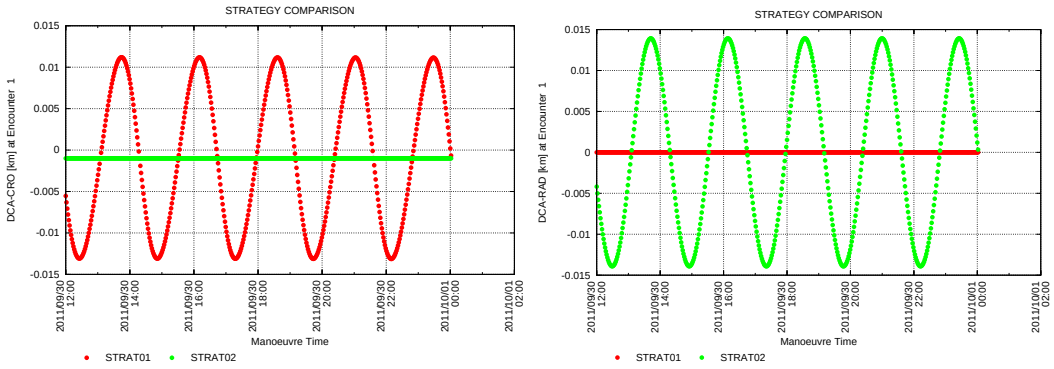

Figure 18. Effect of manoeuvres normal to velocity

We can define a 2-dimensional strategy analysis by considering the manoeuvre size as second analysis parameter. We can see the effect on the collision risk in Fig. 11, where the selected manoeuvre execution time interval (parameter 1 in the analysis) corresponds to a complete orbit revolution, and the delta-v interval from 0 to 5 cm/s in increments of 1 cm/s. For the most favourable manoeuvre location, and for the highest analysed delta-v, the risk is well below 10^{-30}, but the program considers it negligible and returns that value as lowest limit.

5.2. Multi-Encounter Case

Both CORCOS and CAMOS can analyse several close encounters between a pair of chaser and target orbits. In the presented case, we know that there is another close approach half revolution after the first analysed one. This can be configured very easily, simply extending the period for initial encounter search. Figure 19 is the equivalent to Fig. 15, but now considering the second encounter. It can be seen that, where the 1 cm/s along-velocity manoeuvre is best in avoiding one of the encounters, it is worst for the other, as could be expected.

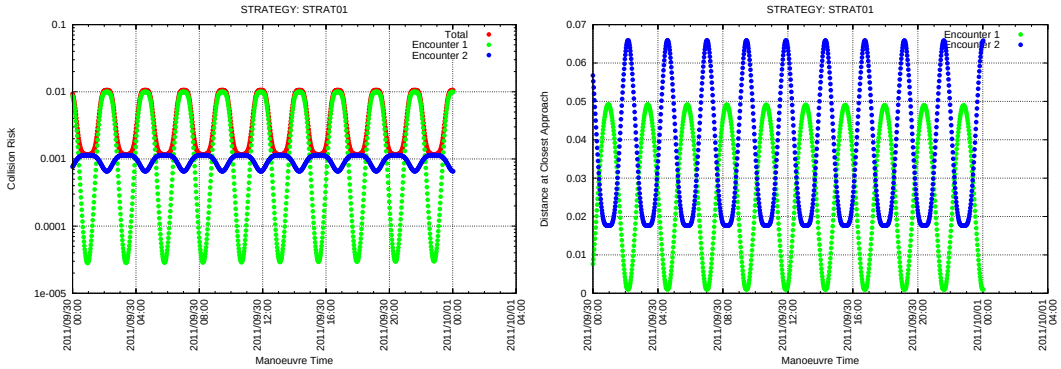

Figure 19. Collision risk (left) and distance at closest approach (right) for first strategy

A similar conclusion can be reached from the 2-D analysis with time and delta-v as strategy parameters. Figure 20 is now the equivalent to Fig. 11, showing that the areas with lowest collision risk in the one-encounter case correspond now to the least favourable areas for the second encounter and vice-versa.

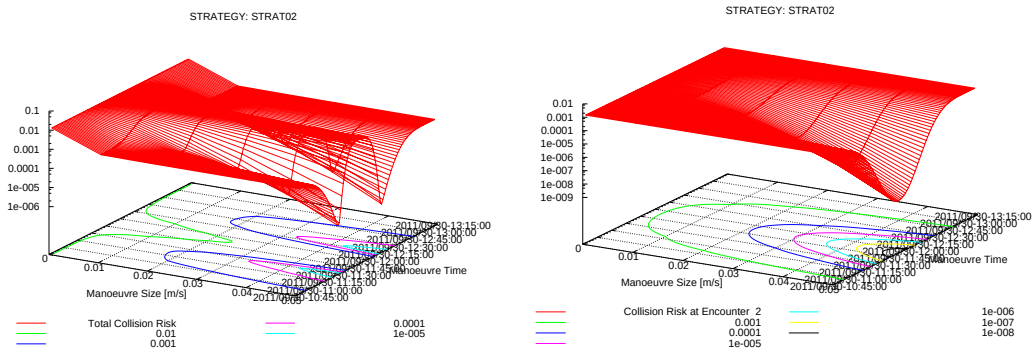

Figure 20. Two-dimensional parametric analysis results for the two encounter case: total collision risk (left) and collision risk for the second encounter (right)

Repeating the analysis with the out-of-plane manoeuvre, we can see in Fig. 21:

- The 1-cm/s manoeuvre produces a maximum change of about 12 m in the cross-track direction of the separation vector when the manoeuvre is executed a quarter of a revolution prior to the encounter, considering that that component of the separation vector is close to 0 without manoeuvres. The maximum absolute cross-track separation, but with negative sign, occurs when the manoeuvre is executed three quarters of a revolution before the encounter. In total, this produces a periodic behaviour of the collision risk, with a period twice the orbital period.
- In the case of the second encounter, the situation is different: without manoeuvres, the cross-track separation is -18 m. That increases to -34 m executing the manoeuvre a quarter of a revolution before, but reduces to -2 m three quarters before. As result, the effect on the collision risk of that encounter is also periodic, but now with same period as the orbit.
- The 1-cm/s manoeuvre is much less effective for the first encounter than the tangential manoeuvre (see Fig. 19).

- For the second encounter, the situation is the opposite: it is more effective the out-of-plane manoeuvre.

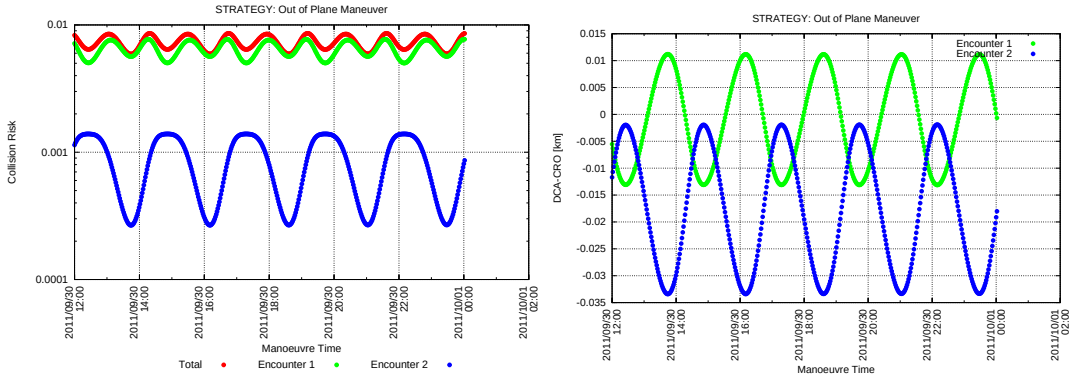

Figure 21. Collision risk (left) and distance at closest approach (right) for out-of-phase strategy

5.3. Multi-Encounter, Multi-Manoeuvre Case

In our example case, the presence of two encounters in different locations makes the mitigation with one manoeuvre a sub-optimal solution from the point of view of the fuel consumption. In this respect, it makes very good sense to study the possibility of executing two manoeuvres, each one targeting one of the encounters. The plots shown in the previous sections give us a very good indication of where to place each manoeuvre, which helps us in setting-up CAMOS for quick results. In order to have a comparative analysis, we present here three different options:

- Two manoeuvres, both along-velocity (tangential case), each one targeting one of the encounters
- A tangential manoeuvre to mitigate the first encounter and an out-of-plane manoeuvre for the second. A priori, this solution makes sense because, for the same manoeuvre, Fig. 21 (out-of-plane case) shows a higher reduction of the collision risk of the second encounter than Fig. 19 (tangential manoeuvre)
- One manoeuvre case (just to assess if the two-manoeuvre solutions are worth the effort)

In the three cases, a one-dimensional parametric analysis is configured, with the cumulative collision probability defined as analysis parameter. The cost function is the total delta-v, and the manoeuvre locations are optimisation parameters. In this respect, this is the first real optimisation analysis (i.e., there are degrees of freedom in the optimisation) shown in this paper, since in previous examples there were no parameters to optimise and no optimisation was needed.

The results can be seen in Fig. 22, where the plot shows the total delta-v of each strategy. The strategies with two manoeuvres require almost the same delta-v (in the plot, the tangential-tangential case “TANG-TANG” overwrites the tangential-out of plane case “TANG-OUT”), while the one manoeuvre strategy “ONE-TANG” requires 0.5-0.6 cm/s more. In this hypothetical case, it would be up to the operational team to decide if the gain is worth the additional burden of the second manoeuvre.

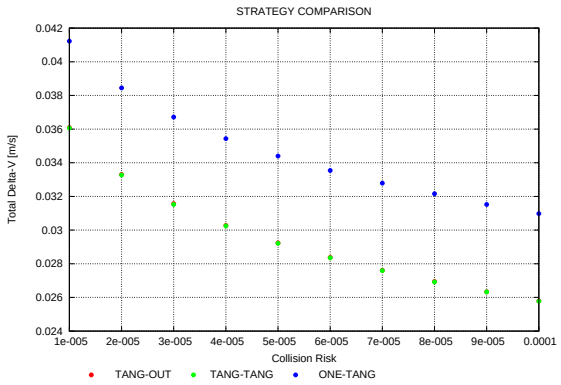

Figure 22. Total delta-V

6. Conclusion

The capabilities of CORAM have been presented in this paper, both in terms of encounter analysis (performed with CORCOS) to compute the collision risk, and in terms of avoidance manoeuvre analysis (performed with CAMOS).

CORCOS is a powerful collision risk analysis tool, which complements CRASS in several ways:

- It implements several collision risk algorithms for spherical geometry:
  - Alfriend & Akella
  - Patera
  - Maximum Probability, assuming spherical covariance, using the maximum likelihood approach
  - Maximum probability according to Klinkrad's algorithm
  - Covariance scaling, where the covariance is scaled for both objects in a given interval
- It can calculate the probability of collision for complex geometry objects (composed of boxes)
- It can analyse encounter with low velocity, where the hypothesis considered for high speed algorithms are no longer valid.
- Monte Carlo can be used in any of the scenarios considered
- It is very flexible in terms of configuration and input options
- Not only collision probability is calculated for each encounter, but also a great amount of additional information (orbit geometry, encounter geometry, etc) is computed for a complete assessment of the approach event.

On the other hand, CAMOS helps the space debris analyst in the selection of the most adequate collision avoidance manoeuvres:

- It performs fast one-dimensional and two-dimensional analysis of manoeuvre avoidance strategies

- The treatment of the manoeuvre design parameters in the analysis is very flexible: each parameter can be treated as a parameter of the strategy analysis, an optimisation parameter, or fixed.
- Constraints can be configured very easily: collision probability, manoeuvre parameter bounds, encounter geometry parameters (e.g., minimum distance)
- The cost function can be selected with a high degree of flexibility: the collision probability (to be minimized), total delta-v (to be minimized as well), separation vector (modulus or its components, to be maximized).

In summary, CORAM is a powerful suite supporting the space debris analyst in the mitigation of collision events between operational spacecraft and orbital debris.

7. References

[1] J. R. Alarcón-Rodríguez, F. M. Martínez-Fadrique, H. Klinkrad, "Development of a Collision Risk Assessment Tool" Advances in Space Research. Vol 34, 2004, pp. 1120–1124.

[2] M.R. Akella, K.T. Alfriend. "Probability of Collision Between Space Objects", Journal of Guidance, Control and Dynamics, Vol.23, No.5, 2000, pp.769-772.

[3] K. Chan, Spacecraft Collision Probability. Aerospace Press, 2008.

[4] Patera, R. (2001), "General Method for Calculating Satellite Collision Probability". Journal of Guidance, Control & Dynamics, vol.24:716-722.5

[5] Patera, R. (2003). "Satellite Collision Probability for Nonlinear Relative Motion". Journal for Guidance, Control & Dynamics, vol. 26, No. 5, 728-733.

[6] Ericson, Christer (2006). "Real-time Collision Detection". Elsevier.

[7] http://en.wikipedia.org/wiki/Minkowski_addition. Last accessed: 21-4-2013.

[8] http://en.wikipedia.org/wiki/Z-buffering.. Last accessed: 21-4-2013.

[9] Klinkrad, H. (2006). "Space Debris: Models and Risk Analysis". Springer Praxis Books / Astronautical Engineering

[10] Pulido-Cobo, J.A., Schoenmaekers , J. (2000). "The Gradient Method Adapted for SMART-1 Trajectory Optimisation". ESOC/Flight Dynamics Technical Note.

[11] Stoll E., K. Merz, H. Krag, B. D'Souza, B. Bastida Virgili. "Probability Assessment for the Rapideye Satellite Constellation". Proceeding of the Sixth European Conference on Space Debris, Darmstadt, 22nd-25th April 2013.Journal for Guidance, Control & Dynamics, vol. 26, No. 5, 728-733.