Monte Carlo Method for Collision Probability Calculations using 3D Satellite Models

Willem H. de Vries

and

Donald W. Phillion

Lawrence Livermore National Laboratory

ABSTRACT

We developed an efficient method for calculating the collision probability using a Monte Carlo approach. The method requires knowledge of the full 6x6 covariance matrix information for each of the objects under consideration, and is capable of incorporating not just the positional uncertainty information, but also the velocity component of the uncertainties in its calculation. This ensures a higher level of accuracy and robustness over strictly analytic methods, albeit at the cost of a larger computational load. It is part of the Testbed Environment for Space Situational Awareness (TESSA) development effort at Lawrence Livermore National Laboratory (LLNL).

This paper will describe our implementation as well as an overview of the capabilities and expectations for the cases where orbital refinement has reduced the size of the uncertainty ellipsoids to much less than a kilometer. Under this regime a spherical approximation of the collision cross-section is not utilizing the full potential of the available information. The combination of direct or indirect attitude information of the satellites, their detailed 3D mesh models, and the relatively accurate information on the size, shape, and separations of the uncertainty ellipsoids can be used to not only refine the collision calculation, but also allow for detailed assessment of the relative likelihood of various impact scenarios. The distribution of Monte Carlo trajectories that form the collection of collision cases is, provided the uncertainties are small enough, distinctly non uniform across the combined satellite cross-section shape. This can significantly modify the relative collision rates based on surface area alone (for instance, the collision geometry and relative positions of the uncertainties may make a hit on the main body more likely than an impact on the solar panels, even though the latter are larger). In cases where a satellite might survive a collision (e.g., a small piece of debris puncturing a solar panel), we can now augment the probability of collision with the odds of survival given a collision. Furthermore, this information allows us to constrain the possible impact scenarios a-posteriori, reducing the number of computationally costly hydro-code simulations we have to run for our detailed debris modeling capabilities (cf. K. Springer et al. in these proceedings for a report on our Cosmos - Iridium analysis).

1. INTRODUCTION

The Lawrence Livermore National Laboratory (LLNL) has been developing a comprehensive simulation environment for Space Situational Awareness (SSA) for over two years now [1]. Development was well underway when the two satellites Iridium 33 and Cosmos 2251 collided over Siberia on Feb 10, 2009, and so we were able to rapidly analyze the debris aspects of the collision [2]. We investigated a range of collision geometry scenarios and checked the predicted debris distribution and their dispersion over time with actual published orbital elements. Regardless of this particular test case, where we used the system to effectively conduct a post-mortem of the collision, we always envisioned this system to handle future, and potential events. As such, it needed two additional components: a module that can predict close approaches between two objects, and a component that can calculate the collision probability. Even a single what-if scenario can end up involving large numbers of actual orbiting objects and their potential secondary debris under collision. Furthermore, an actual, large population of debris objects with sizes between 1 and 10 cm has been inferred [3,4], even though they are currently too small to be tracked by the space network. Any scenario involving this hazardous small-debris population will have to handle

even larger numbers of orbiting objects. The new modules, therefore, are designed to be efficient, scalable and easily parallelizable in order to handle this in a timely fashion.

This paper describes the two modules in some detail; with the main emphasis on the collision probability calculation and its applications in cases where the size of the uncertainty ellipsoids have been reduced to sub-100 meter levels. This can be achieved by obtaining targeted observations of both objects involved with the conjunction [5], either using ground or space-based platforms.

2. CONJUNCTION ANALYSIS

Our conjunction analysis (CA) module is primarily used to screen the large number of close approaches between two objects in the catalog for ones that we are interested in. Given the large number of tracked objects in the current NORAD catalog (about 15000 at the moment), it should not come as a surprise that a significant fraction of these objects come close to each other on a regular basis. If we use 10 km as a maximum separation threshold, the number of close conjunctions is about 30000 per day at the moment. Given the ever-increasing satellite population, and an anticipated future increase in the size of the tracked catalog (by at least an order of magnitude), it becomes clear that one needs an efficient way of dealing with this. Every time one is interested in close approaches, either in the context of potential collision threats with debris, or, for instance, knowing when a certain asset can observe another, the initial step is running the CA module. This section describes our implementation.

First, we assume that we do not have an externally available ephemeris catalog for all the tracked objects over the period we are interested in, so we need to generate this before we can start the CA. Ideally, one would like to run the full force model implementation to calculate the position and time pairs for all objects. This would be especially useful in regimes where, for instance, drag or radiation pressure can influence the orbit of the object. The major downside is the huge computational burden making this a very time consuming proposition. However, we do not require extreme accuracy for this particular task since we are not using this module to figure our whether two objects actually are going to collide. Furthermore, we are using the results either in a statistical sense, i.e., determining the rate of close approaches against a constellation of satellites, or we are using the module to select a subset of objects that have close approaches. Provided we are using a large enough threshold of at least a few times the inferred spatial uncertainty of the catalog (for instance, use a 10 km threshold for the ~1km accuracy NORAD catalog), we can assume that *all* of the much closer approaches are included and that some incompleteness sets in near the threshold. For our applications this is acceptable. If one is concerned about reaching completeness at 10 km thresholds, one simply runs the CA with a sufficiently larger one, and then down-selects the results. Given these relaxed requirements, we generate our ephemeris data based solely on the Two-Line Element (TLE) catalog using the Standard General Perturbation (SGP4) propagator.

We evaluate the positions for all objects concerned on a fixed time grid. The number of evaluations per day can be varied (see below), but we typically step it at 90-second intervals. Once all the positions have been calculated and stored in memory, the actual process of running the conjunctions begins. At this point we like to point out two obvious modes of parallelism: first, among objects (A against B, A against C etc), and second among time-slices (A against B over t_0 to t_1, A against B over t_1 to t_2 etc). A nice advantage of the SPG4 propagator (over the force model propagator) is that it can be evaluated at any given time without knowledge of previous evaluations. As such many parallel CA request can be spread out over time. The actual conjunction kernel is kept as simple as possible, allowing for easy porting to for instance GPU cores. First, the code evaluates the separation between two objects using the position of each. Since we are evaluating these on a time-grid, all position arrays are time synced. If the function $S^2(t)$ denotes the square of the spatial separation between the two objects as a function of time, the code determines its minima over the period of interest by stepping through time. At each minimum, it uses the $S^2(t)$ evaluations just before and after to calculate the parabola that passes through these 3 points. Then determining the time t_{min} at $-b/2a$ is trivial, and a final pair of SGP4 position evaluations at t_{min} gives the actual minimum separation $S_{min}=S^2(t_{min})$. Note that we use the parabola to determine the t_{min} and not the S_{min}. The function $S^2(t)$ is clearly not a parabola near t_{min}, resulting in bad fits. However, the function is symmetric in almost all cases (unless there is significant acceleration during the encounter), so the minimum the parabola and the $S^2(t)$ function agree in *time*.

This approach is different than for instance, the Alfano/Negron Close Approach Software (ANCAS) [6], which tries to describe the object separation by the use of higher order polynomials. While this does allow for large time-steps,

it comes at a rather large computational overhead. In Table 1, we compare their test case of an encounter between Endeavour and a Cosmos booster (see the tables 2a and 2b in [6]). As can be seen, ANCAS does yield reasonably accurate results for 600-second time steps, but our approach yields significantly better results at smaller step sizes. We match ANCAS at 200-second step sizes. Finer time-grids result in better accuracy at the cost of longer execution size and larger memory overheads.

Table 1. Comparison of separation determinations

Method	Step Size [s]	Time error [s]	Separation error [m]
ANCAS	600	0.1	227.7
S^2 parabola	600	1.762	11468.3
	300	-0.396	1170.1
	200	0.124	129.5
	150	-0.082	57.2
	90	-0.011	0.96
	40	<1ms	0.003
	20	<1ms	<1mm
Actual (TLE based)	t_{min}=2449162.022318 (1993/6/23 12:32:08.921) S_{min} = 4451.907		

Not all pairs of objects are handled in this way, as we have certain filters in place. Objects that have orbital regimes that do not overlap to within a few times the threshold distance are skipped immediately. The parabola determination step is skipped if the grid minimum itself is much larger than the threshold distance (this distance depends on the magnitude of the relative velocity between the objects). Both these filters have been tested against runs without filters and have been detuned enough to ensure they are not rejecting valid cases. The overall execution times, including the generation of all the ephemerides of all objects, are listed in Table 2 for a representative set of target machines. The runs times are for a full NORAD catalog (15APR09) cross-correlated with itself for a 24-hour time-span, resulting in 28654 conjunctions within 10 km. This particular catalog contained 12875 objects.

Table 2. Conjunction analysis execution times for a full-on-full catalog

Architecture	Logical Cores	Execution time [s]
Dual Intel Xeon 2.26 Ghz Quad Core desktop	1	1405.72
(each processor contains 4 cores capable of supporting	2	708.35
two concurrent threads each, for a total of 16)	4	395.34
	8	237.60
	16	146.28
Nvidia Tesla C1060 GPU	240	170.43
Nvidia Tesla S1070 GPU	960	44.06
Nvidia GTX 285 GPU – consumer grade graphics card	240	145.58

As can be seen in Table 2, it is not hard to estimate the execution time given a catalog size and a period over which we want to find conjunctions. The number of permutations increases quadratically with catalog size, and linearly with the period length. If one wanted to finish a full-on-full conjunction request over a single day using a catalog containing 1 million objects within an hour of runtime, one would have to allocate on the order of 2500 cores to the process (everything else being equal). Since this amount of cores is well within the capabilities of the current generation of supercomputers, we feel confident that the performance of our CA module meets future requirements.

3. COLLISION PROBABILITY METHODOLOGY

As mentioned before, we are not using the CA module to assess the likelihood of collision. It does provide some handle on collision probability though, but only in a statistical sense. One can estimate the collision half-life time for a particular satellite by using the ratio of the satellite cross-section to the conjunction threshold distance in combination with the number of close approaches by other objects over a long period of time. What we are really after, however, are the odds of a collision for a *particular* close encounter. For this, we need to have information beyond the TLE orbital elements: for each object we need to know its covariance matrix and the corresponding state vector. Typically these are obtained as by-product of an orbit determination or refinement (based on observations), or, in the absence of one's own observations, in the form of a catalog. In our case, we calculated force-model orbits to TLE generated "observations" to arrive at the covariance matrix and state vector information. Regardless of the source of these, the method described below applies to both.

Before we proceed further, we need to point out the following assumptions and specific requirements. First and foremost, we will assume that the covariance matrix is constant over the period of the close conjunction. This assumption only breaks down in cases where the approach velocity is so low that the conjunction itself takes a significant amount of time and hence occurs over a non-trivial portion of the orbit. Scenarios like these mainly play out in the GEO belt where almost all objects orbit in tandem, resulting in very low approach speeds. Since we are mainly concerned with the LEO and MEO regimes, with their very diverse orbits and inherently larger orbital speeds, low velocity approaches (on the order of meters/s) are exceedingly rare. The second, and more implicit assumption is that the covariance matrix accurately describes the uncertainties associated with the position and velocity components of the object (see [7] for a discussion). If it does not represent the state of knowledge on the object, then any derived collision probability is useless. We are furthermore assuming that the state vector (i.e., the position and velocity components of the objects) represents an unbiased estimate of both quantities.

The collision probability (CP) module does the following. It first makes sure that both covariance matrices and state-vectors have been correctly propagated to the time-stamp of closest approach (as given by the CA module). This propagation is typically done in the equinoctial coordinate system using numerical partials in order to minimize the introduction of artificial noise. It then derives the eigenvalues and eigenvectors of each covariance matrix using standard numerical methods. Based on the two state-vectors, it determines the relative approach vector (the difference between the two velocity components of the state-vectors), and sets up a new coordinate system in which both objects approach each other along the z-axis, with the x and y axis being the basis vectors of the collision plane. The 3D Gaussian uncertainty distribution functions for each object (based on the magnitude of the eigenvectors, and centered on each state-vector) are transformed into this new frame of reference. Off-axis values in the covariance matrix represent the cross-correlation terms present in these distributions. For our analytical probability estimates, both these distributions are first projected onto the collision plane (i.e., collapsing along the z-axis) and combined into a single probability function. The collision probability is then based on the 2D integral evaluation of this PDF. We have implemented a strict numerical approach (based on [8]), as well as the method by Chan [9], which uses a transformation into a 1D integral using Rician functions. Both these methods are implemented for comparison purposes to our Monte Carlo approach, and are almost always in good agreement (see [10]). They also provide an estimate in cases where the very low probability affects the discrete MC approach in terms of sparsity of rays. This typically happens far out in the Gaussian wings, with probabilities below 10^{-7} or so.

Table 3. Combinatory methods for determining closest approaches of MC instances

MC sub method	Description
1. straight projection onto collision plane	Varies positions according to Cov. Matrix. Ignores velocity variations, assumes all conjunctions happen at same t_{min}
2. linear	Includes velocity variations, uses linear tracks to determine closest approach, allows t_{min} to vary (i.e., closest approaches can happen above and below the plane)
3. curved	Includes velocity variations, uses relevant section of (curved) orbit to determine closest approach, allows t_{min} to vary

In our MC approach, we do not combine the two distribution functions, but generate a large number of samples for each. This, combined with our ability to handle complex cross-sections provides a distinct advantage over the analytic approach, albeit at a larger computational cost. Once we have generated the random samples for each object (varying both the position and velocity components based on the eigenvalues of the covariance matrix), we then cross-correlate the two distributions and determine through a number of distinct ways, listed in Table 3, the distance of closest approach for a particular pair of MC state-vectors.

Finally, we determine whether these closest approaches constitute collisions or not. In the simple case of a spherical cross-section, we just need to test whether the closest separation is smaller than the cross-section radius, and for the more realistic satellite models we need to include the relative attitudes and account for the approach geometry. We will return to this in more detail in Section 5.

4. STATISTICAL ROBUSTNESS

For each object, we generate a large, random population of state-vectors based on the information contained in the covariance matrix. The questions now are, how many random instances are needed, and what level of confidence can one associate with the result? To answer this, we are first going to look at permutations.

Assume we generated N state-vectors for each object (for a total of 2N). Combining these two populations, which effectively is what one does calculating spatial separations, potentially allows for NxN distinct permutations. For large N, this could potentially yield a huge savings over having to generate N^2 (computationally costly, truly independent) random state-vectors while only 2N would suffice (see, for instance, [11]). Unfortunately, as it turns out, the quality of randomness using the NxN permutations is not as good as N^2 truly independent pairs, and this is basically due to the repetitive re-use of points. If one has N independent pairs of separations, then the fraction *f* denotes the ratio of separations determined to be a hit over the total number of pairs (i.e., *f* will approximate the collision probability). The uncertainty in the value *f* is given by the binomial distribution:

$$\sigma_{\mathrm{bin}}(f) = \sqrt{\frac{f(1-f)}{N}} \qquad (1)$$

In Figure 1, left panel, we have plotted the behavior of the σ as based on 132 independent runs using log(N)=3,4,5,6 sample sizes (solid squares) versus the expected binomial behavior (dotted line). As can be seen for these 2 particular cases, the agreement is very good. Therefore, given a resulting probability *f* and a known sample size N, one can easily estimate the 1σ uncertainty in the value of *f* by using Eqn. 1. The sole caveats are the assumption that *f* represents the true collision probability, and that one has indeed N independent separation samples. Trying to generate large numbers of samples through permutations does not guarantee this however. In Figure 1, right panel, we plot two different permutation methods, one using *all* NxN permutations, and one using Nx√N in an attempt to not "overuse" the samples. The plot shows the quality of randomness based on the measured σ in the P_{coll} distribution versus the binomial σ, but does so in units of $N_{effective}$, i.e., it calculates the equivalent N if the measured σ were binomial using $N_{binomial} = f(1-f)/\sigma^2$. If the randomness quality of the permutation sample is less than the expected binomial value, then the N_{eff} will be less than the NxN claimed, illustrating the problem with generating combinatory samples. The binomial case (in red) shows, per definition, that $N_{claimed} = N_{effective}$. The small variations are due to the limited size of the sample (132 runs). As can be seen for the permutation cases, neither of them matches the binomial numbers. In particular the NxN permutation case underperforms by quite a bit. This means that one tends to severely overestimate the accuracy of the collision probability calculation based on an NxN permutation sample. The situation improves a little when one considers cases where either the nominal miss distance is larger than a few times the size of the uncertainty ellipsoid (see case 3 in the right panel).

In our CP module, we use the Nx√N permutation technique in order to avoid having to run N^2 samples. Our value of N is typically around 500000, for an Nx√N total of 3.5×10^8. The inferred accuracy of the collision probability determination is determined by calculating the binomial σ using only 10% of the total number of permutations. This one order of magnitude reduction in accuracy matches the behavior in Fig 1, right panel. In cases where one really needs an accurate assessment of the collision probability, there are two ways of handling this. First, one can run many independent CP determinations, and establish the rms variations of the outcomes, or secondly, one can run a

single case with large N, forego on permutations, and use the binomial σ. Neither of these is very efficient though, which is the reason why we chose to use the Nx√N permutation case as a compromise.

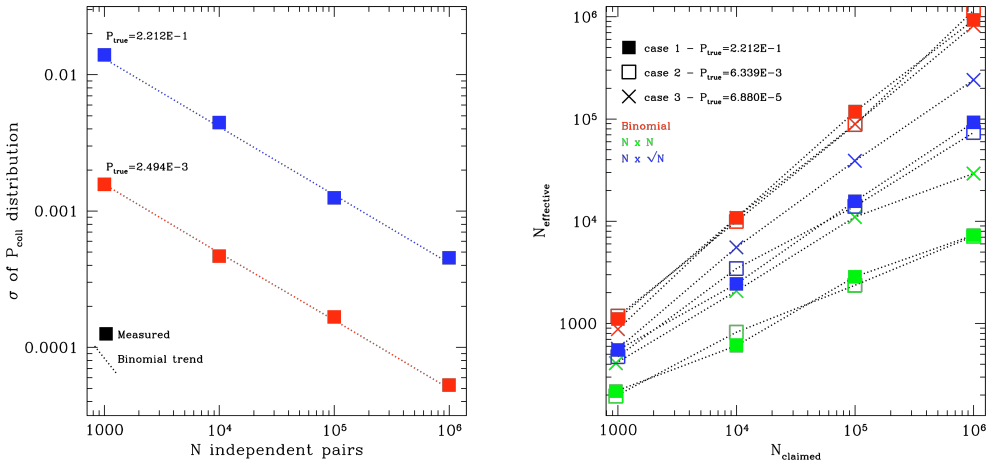

Figure 1. [Left Panel] Comparison between σ behaviors of varying sample sizes N (in solid squares), versus the predicted binomial behavior (dotted line). The two test cases are: $\sigma_{x1,2}=\sigma_{y1,2}=1.0$, separation < 1.0 in blue, and $\sigma_{x1,2}=\sigma_{y1,2}=1.0$, separation < 0.1 in red, and each of which has been run 132 times to build up sample statistics. Both distributions are centered on the origin. Note the very good agreement between the measured σ and the binomial σ [Right Panel] Comparison between inferred and actual σ of the collision probability determination, expressed in terms of equivalent binomial N. Per definition, for the binomial case in red $N_{claimed} = N_{effective}$. This is not the case for either the NxN or the Nx√N permutation cases, though the latter comes close for small P_{true} values.

5. EXAMPLE CASE WITH 3D MODELS

In a lot of cases, debris especially, we do not know the actual spatial orientation of the object, nor its behavior over time. Debris pieces and defunct satellites tend to rotate and/or tumble, potentially causing large variations in collisional cross-sections. For these cases we wrap our ignorance in a single, spherical "effective" cross-section. On the other hand, if we do have accurate attitude information, we should not ignore it.

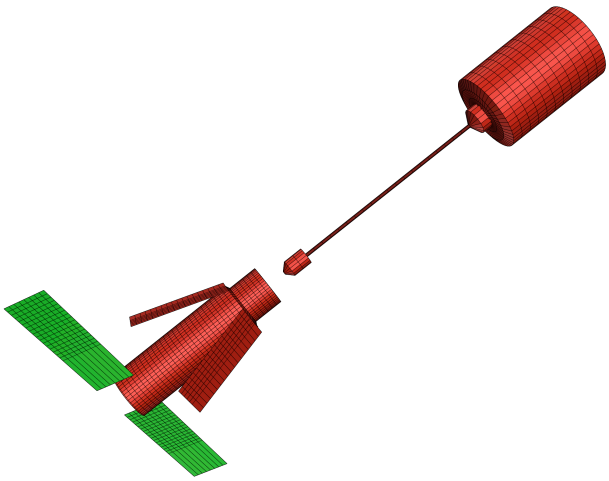

Figure 2. Sample of meshed satellites. On the left is an Iridium-like satellite, and on the right is a cosmos-like model. In this frame, the center of the Earth is to the top right.

This Section describes our implementation and its use in cases where the size of the uncertainty ellipsoids has become sufficiently small (less than 100 m). As far as the attitude information goes, we either have the actual information, or we can make an educated guess. If both objects involve active (or passive) attitude control measures, then a fair assumption can be made. In our example in this Section, we have an Iridium-like satellite and a cosmos-like satellite. The former is active, which implies its solar panels are pointing at the Sun and the communications package is pointed towards the Earth. This provides a good handle on its attitude. The cosmos has a passive gravity boom pointing away from Earth's center, and since it is more or less rotationally symmetric, we have a good handle on the attitude as well. In a particular scenario (not the actual Cosmos – Iridium collision on February 12, 2009), we arrive at the relative geometry shown in Fig. 2. Each satellite has been meshed up and oriented according to our assumed attitudes. Figure 2 shows their relative positioning along the velocity difference vector (i.e., both move perpendicular to the plane of the paper, one in and one out). Our code then downgrades the (potentially rather high fidelity) mesh resolution to larger "pixels" in the collision plane. We typically use either 10 or 30cm square pixels. State-vectors are generated for each object as before, but now the code has to determine whether a particular pair represents a collision by testing for overlap between the two pixel maps. This can be done efficiently given the limited resolution and typically small extent of the pixel maps.

Everything is analogous to the method described in Section 3, but in addition to the collision probability we now also have meaningful, spatially resolved statistics on the collision geometry. In Figure 3, we show the distribution of the state-vector pairs that resulted in a collision. The relative orientation is the same as in Fig. 2. We can identify the shape as the combination of the outlines of the two satellites. As such, it is straightforward to determine the impact scenario based on the shape and offsets from the origin. Area 1 has the boom of the Cosmos satellite hitting the main body of the iridium satellite. As one translates along the diagonal towards the lower left, this boom starts hitting the solar panels in area 2. Area 3 is the main Cosmos body hitting the solar panels, and finally, area 4 is Cosmos body on Iridium body.

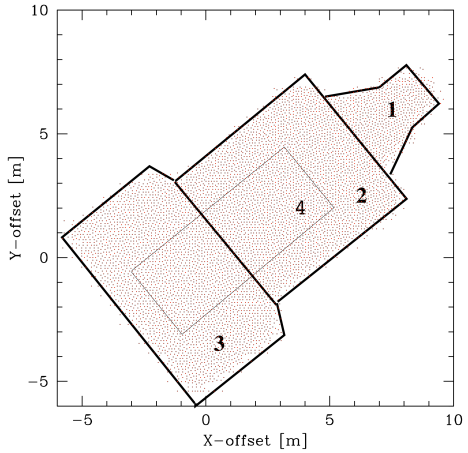

Figure 3. Spatial distribution of MC rays in the collision plane that resulted in contact between the two satellites. The relative orientation is similar to Fig. 2. Numbered areas are described in the text.

Maps like this can be used to assess the relative impact scenarios, and possibly, if one of the objects is a small debris piece, relative survivability (a puncture of a solar panel might not render the satellite inoperable). If, as in Fig. 3, the distribution of these MC rays is rather uniform, this becomes a matter of relative surface area (as projected onto the collision plane). However, in cases where either the size of the uncertainty ellipsoid is small enough, or the object large enough (the International Space Station for instance), one will see a density gradient across the overlap surface. This density gradient can substantially modify the relative collision probabilities. To illustrate this effect, we show in Fig. 4 four cases of varying orbital accuracy. The combined size (of the three principal axes added in quadrature) of the uncertainty ellipsoids are, from left to right, red 6.3m green 8.7m, red 12.5m green 17.3m, red 25.0m green 34.6, red 50.0m green 69.2m. The top set of panels show the actual spatial distribution of the state-vectors in the collision plane. The solid squares denote the centroids of the distributions (the nominal miss distance is 25m). As can be seen towards the panels on the right, most of the gradient disappears if the distributions become

large relative to the miss distance and the overall extent of the cross-section. Uncertainties on the order of 10m or so can be achieved through targeted observations over a reasonable period (24 hours or so, see [5]).

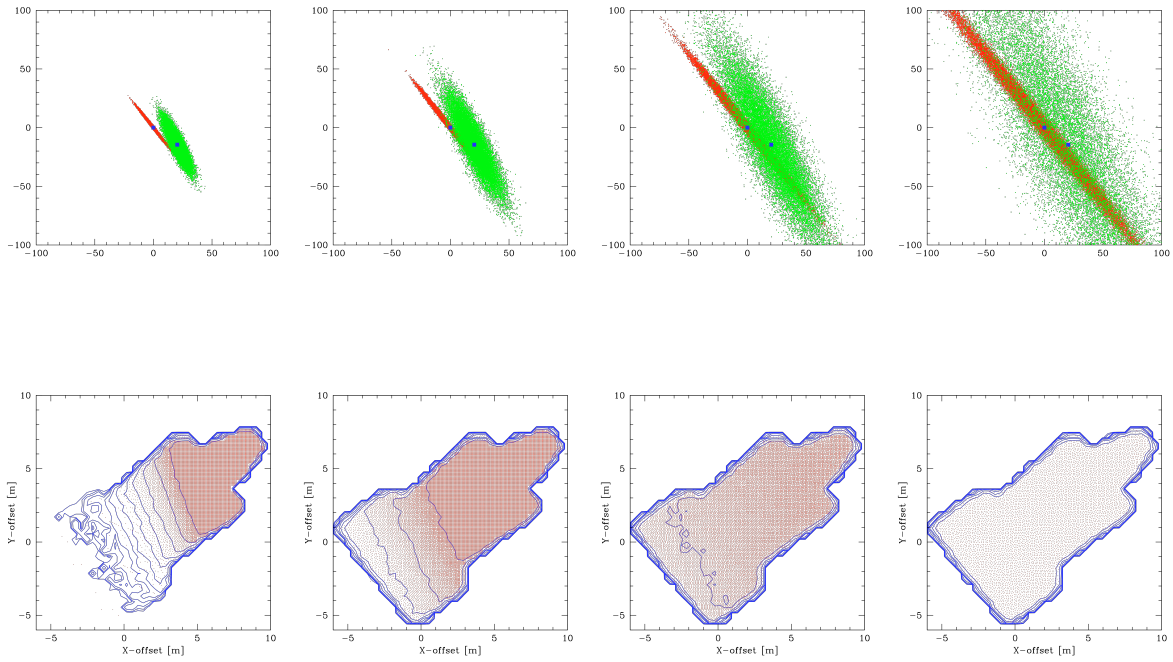

Figure 4. [Top set of 4 panels] Scatter plots of the two state-vector distributions in the collision plane for each satellite (Iridium in red, Cosmos in green), as function of size (σ's double between each panel, starting with 6.25m and 8.65m, respectively). [Bottom set of 4 panels] Corresponding hit distribution maps (see Fig. 3 for a description). The contour lines indicate a doubling of the local density.

Table 4 lists how the relative impact scenarios are modified from the uniform distribution case. These likelihoods are significantly modified from the numbers based just on exposed surface area, especially at the extremely high precision levels of the leftmost panel. Even the more realistic 2nd panel allows us to constrain the very costly high-fidelity hydro-code collision runs by weighting them by their relative likelihood (see [12] in these proceedings for a description of our collision modeling).

Table 4. Relative impact likelihoods

Area (see Figs. 3 and 4)	Uniform	2nd panel on left (Fig. 4)	Leftmost panel (Fig. 4)
1. Boom on body	11.57%	22.62%	45.07%
2. Boom on panels	33.36%	47.78%	49.90%
3. Body on panels	28.60%	9.55%	0.20%
4. Body on body	26.48%	20.06%	4.84%

6. SUMMARY

We developed both the conjunction analysis and collision probability modules for our TESSA environment with an eye on future requirements. The NASA Orbital Debris Program Office currently estimates that the number of debris particles between 1 and 10 cm (still large enough to severely damage a satellite) is on the order of 500 000 [13], quite a bit larger than the currently tracked population of 10 cm and larger. As we have described earlier, our conjunction analysis module is capable of handling sample sizes of these magnitudes. With planned SSN upgrades, both in sensitivity and new capabilities (e.g., the Space Based Space Surveillance system), handling sample sizes this large will have to become routine.

Our collision probability module is capable of handling a large range of different engagement scenarios. It will calculate 5 different assessments of the collision probability (two analytical approaches, and 3 Monte Carlo based ones), all of which agree very well under standard, short-term encounter, cases. While it can handle deviations from non-linear motions by the objects, in its current form it cannot be applied to cases where encounters take a long time and / or have covariance matrices that are not constant (e.g., particular GEO scenarios with very low Δv, or LEO cases with high drag). We are working on implementing these capabilities under the MC framework by basically force model propagating each individual ray.

The other focus of our collision probability discussion is on the regime where one has exquisite orbital fidelity (uncertainties less than 100 m). While currently only attainable in cases where both objects carry GPS receivers, it is well within range of current technology. Either a ground-based or space-based network of modest sensors is capable to refine known orbits to better than 100 m by taking multiple observations of an object over a period of up to a few days [5]. Note that this only possible for objects with known orbits, since the assets with their narrow fields of view will have to know where to point (by a sensor tasking algorithm). Networks like this are not useful for a survey mode that can be used to detect new debris. Once in this sub-100 m regime, we show that it is useful to have detailed models of the satellites (and debris if possible) since a simple spherical cross-section becomes rather inaccurate. For instance, long gravity booms or large solar panels significantly affect the collision probability. Furthermore, since almost none of the collisions are head-on, the size, shape, and separation of the two uncertainty ellipsoids affect the likelihood distribution across the target. We intend to use this information to constrain the number and relative geometry of our high fidelity hydro-code collision simulations [12], but it can equally well be applied to assess survivability estimates in cases of collisions between a satellite and a small piece of debris.

7. ACKNOWLEDGEMENTS

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

8. REFERENCES

1. Olivier, S. S., A Simulation and Modeling Framework for Space Situational Awareness, Proc. Adv. Maui Optical and Space Surveillance Technol. Conf., Wailea, Maui HI, 2008
2. Olivier, S. S., Cook, K., Fasenfest, B., et al., High Performance Computer Modeling of the Cosmos-Iridium Collision, Proc. Adv. Maui Optical and Space Surveillance Technol. Conf., Wailea, Maui HI, 2009
3. Liou, J-C., Matney, M. J., Anz-Meador, P. D., Kessler, D., Jansen, M., and Theall, J. R., The New NASA Orbital Debris Engineering Model ORDEM2000, NASA/TP-2002-210780, 2002
4. Liou, J-C., Collision Activities in the Future Orbital Debris Environment, *Advances in Space Research*, Vol. 38, 2102-2106, 2006
5. Henderson, J. R., Nikolaev, S., Phillion, D. W., de Vries, W. H., Pertica, A. J., and Olivier, S. S., Intelligent Sensor Tasking for Space Collision Mitigation, proceedings of the SPIE, Vol. 7691, 2010

6. Alfano, S., Determining Satellite Close Approaches, Part II., *Journal of Astronautical Sciences*, Vol. 42, 143-152, 1994
7. Vallado, D. A., and Seago, J. H., Covariance Realism, AAS/AIAA Paper 09-304, Astrodynamics Specialist Conf., Pittsburgh, PA, 2009
8. Foster, J. L., and Estes, H. S., A Parametric Analysis of Orbital Debris Collision Probability and Maneuver Rate for Space Vehicles, NASA/JSC-25898, 1992
9. Chan, F. K., *Spacecraft Collision Probability*, Aerospace Press, 2008
10. Alfano, S., Review of Conjunction Probability Methods for Short-term Encounters, AAS/AIAA Paper 07-148, Space Flight Mechanics meeting, Sedona, AZ, 2007
11. Alfano, S., Satellite Conjunction Monte Carlo Analysis, AAS/AIAA Paper 09-233, Space Flight Mechanics meeting, Savannah, GA, 2009
12. Springer, H. K. et al., Satellite Collision Modeling with Physics-based Hydrocodes: Debris Generation Predictions of the Cosmos-Iridium Collision Event and Other Impact Events. These proceedings, 2010
13. NASA Orbital Debris Program, http://orbitaldebris.jsc.nasa.gov/index.html