Space Debris Collision Detection using Reachability (Benchmark Proposal)

Kerianne Hobbs[1,2], Peter Heidlauf[1], Alexander Collins[1], and Stanley Bak[3]

[1] Air Force Research Laboratory, Aerospace Systems Directorate
{kerianne.hobbs, peter.heidlauf.1, alexander.collins.3}@us.af.mil

[2] Georgia Institute of Technology
kerianne@gatech.edu

[3] Safe Sky Analytics
stanleybak@gmail.com

Abstract

Space debris tracking and collision prediction is a growing world-wide problem as more and more objects are placed into orbit. While traditional methods simulate particles with Gaussian uncertainty to make collision predictions, we instead analyze the problem from a reachability perspective. The problem appears to require methods capable of quickly analyzing high-dimensional nonlinear systems, but we take advantage multiple kinds of problem structure to show that reachability analysis may be viable for this problem. In particular we present an initial analysis approach that uses numerical simulation for reachability analysis, and interval arithmetic with AABB trees for fast collision detection. The analysis uses a variable size time step with a counter-example guided abstraction refinement (CEGAR) method to increase analysis speed without sacrificing accuracy. Our approach can analyze upwards of thousands of orbiting objects faster than real-time, where each object is subject to some initial state uncertainty.

1 Introduction

Historically, space situational awareness has been a combined effort by NASA, the Department of Defense (DoD) and the intelligence community. While the techniques used to predict satellite and debris conjunctions have been effective, they feature a very high false alarm rate and are overwhelmed for the large number of objects, more than 100 objects that are regularly changing orbits, and high uncertainty in the locations of resident space objects. While 23,000 objects larger than 10 cm in diameter are regularly tracked by the U.S. government, 500,000 objects between 1 and 10 cm as well as more than 100 million objects smaller than 1 cm are predicted to exist [13]. The sheer number of objects presents many challenges in the n-to-n prediction of a conjunction between any two objects. In addition to the growing number of

DISTRIBUTION A: Approved for public release; distribution unlimited. Case Number: 88ABW-2018-2108, completed on 23 Apr 2018.

objects, the conjunction prediction problem is further complicated by slow ascent or descent satellites and commercial satellite maintenance or inspection services that regularly change orbits. Furthermore, space surveillance can provide an accurate update on the positions and velocities of resident space objects once every few days, with a significant amount of uncertainty between observations. A variety of academic models have been used to predict spacecraft position over time, typically using simulation-based analysis [10, 1, 18, 2, 5].

Several orbit and rendezvous benchmarks have recently been proposed for linearized dynamics based on the Clohessy-Wiltshire Hill (CWH) equations [14, 16, 6]. These dynamics are applicable in a small section of the orbit where linearization is reasonable. Nonlinear orbital dynamics has also been considered in the case of coaxial and coplanar orbits [17]. This work used hybridization [3, 12] to analyze the nonlinear dynamics and prove collision avoidance for the two-satellite case. Here, in contrast, we consider the more general orbit case, with different axes and different planes of orbit. Further, we aim to analyze thousands of orbiting objects concurrently.

What enables analysis for such high-dimensional systems is that the dynamics of orbiting objects are disjoint from one another, and so rather than analyzing a single high-dimensional system, we analyze thousands of low-dimensional systems and combine the results together. In the simplest case, each orbiting object can be modeled as a one-dimensional dynamical system, where the variable is the angle within the orbit, the true anomaly ν. Verification is nonetheless challenging as the safety property requires a nonlinear mapping between each object's ν and a 3d position in space, and then an n-to-n collision check to be performed at every point in time.

We first present a nonlinear orbital dynamics model for the general case of noncoaxial noncoplanar orbits in Section 2. Next, Section 3 describes the verification problem and proposes a verification approach based on simulation-based reachability analysis, fast collision-detection data structures, and CEGAR-based variable step size.

2 Orbital Dynamics Modeling

This section describes the basic motion of an object in orbit about Earth, at increasing levels of complexity. A summary of the variables to be used in this section is presented in Table 1.

Table 1: Summary of Orbital Elements and Parameters

Symbol	Units	Range or Calculation	Name
ν	rad	$[0, 2\pi]$	true anomaly
e	-	$[0, 1]$	eccentricity
h	km	$[6371, 1.5 \times 10^6]$	altitude (at apogee)
a	km	$[6371, 1.5 \times 10^6]$	semi-major axis
i	rad	$[0, \pi]$	inclination
Ω	rad	$[0, 2\pi]$	right ascension of the ascending node
ω	rad	$[0, 2\pi]$	argument of perigee
p	km	$p = a(1 - e^2)$	semi-latus rectum
μ	$\mathrm{km}^3/\mathrm{s}^2$	3.986012×10^5	geocentric gravitational parameter for Earth
$\mathcal{F}_{\mathtt{ijk}}$	-	-	inertial reference frame
$\mathcal{F}_{\mathtt{pqw}}$	-	-	orbital reference frame
☊	-	-	ascending node
♈	-	-	vernal equinox

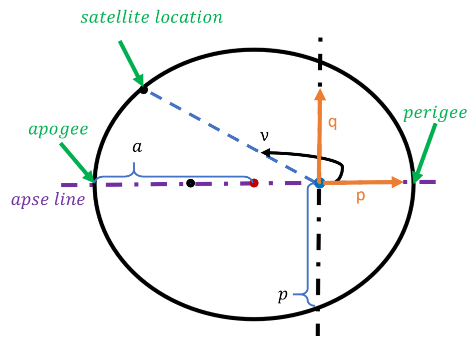

Figure 1: Orbital Plane

2.1 Orbits on a Plane

Orbits are modeled as ellipses in a plane with the Earth at one focus, as shown in Figure 1. The ellipse is defined by the semi-major axis a, which is the furthest distance from the center of the ellipse to the edge, and eccentricity e, which is a measure of how much the orbit deviates from circular. The location of the satellite within the orbit is described by the true anomaly ν, which is the angle from the unit vector pointing to perigee and the position vector pointing from the center of the Earth to the satellite. The cartesian coordinates are defined in the orbital reference frame $\mathcal{F}_{\mathtt{pqw}}$ centered on the center of the Earth, where $\vec{\mathtt{p}}$ is a unit vector pointing from the center of the Earth to perigee, $\vec{\mathtt{q}}$ is orthogonal within the plane and aligned with the semilatus rectum, and $\vec{\mathtt{w}}$ completes the orthogonal set.

Also specified in the orbital frame is the semi-latus rectum, as described by

$$p = a(1 - e^2). \tag{1}$$

The distance from the center of the Earth to the position of the satellite is calculated using

$$r = \frac{p}{(1 + e\cos(\nu))}. \tag{2}$$

Within the orbital plane, the evolution of the spacecraft is modeled using Kepler's law of equal areas

$$\dot{\nu} = \sqrt{\frac{\mu}{p^3}}(1 + e\cos\nu)^2. \tag{3}$$

The position $\vec{r}_{\mathtt{pqw}}$ of the spacecraft in the plane is given by equation

$$\vec{r}_{\mathtt{pqw}} = \begin{bmatrix} r\cos\nu, & r\sin\nu, & 0 \end{bmatrix}^T \tag{4}$$

2.2 Orbits in an Inertial Frame

The semi-major axis a, eccentricity e, and true anomaly ν are the first three of six orbital elements. The remaining three can be visualized using Figure 2. The argument of perigee ω is the angle within the plane that is measured from the ascending node Ω (the vector pointing from the center of the Earth to the point in the three-dimensional orbit that crosses the Earth's

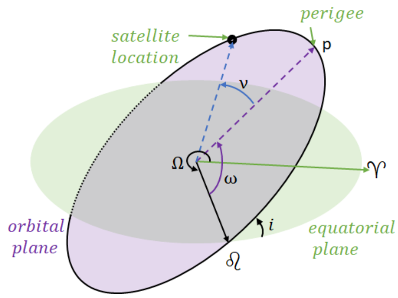

Figure 2: Orbits in the Inertial Frame

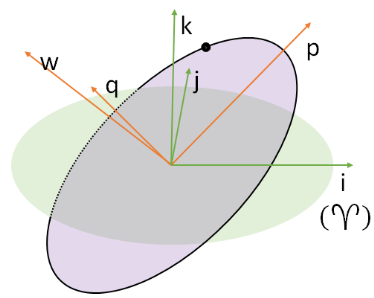

Figure 3: Orbital and Inertial Reference Frames

equatorial plan from the southern to the northern hemisphere) to the vector pointing from the center of the earth to perigee $\vec{\mathtt{p}}$. The inclination i is the angle from the equatorial plane to the orbital plane at the ascending node ☊. The right ascension of the ascending node Ω is the angle from the vernal equinox ♈ (vector from Earth's center to the Sun when aligned with the equatorial plane on the vernal equinox in the Northern Hemisphere) counter clockwise to the ascending node ☊.

Translating the motion from the inertial frame $\mathcal{F}_{\mathtt{ijk}}$ to the orbital frame $\mathcal{F}_{\mathtt{pqw}}$ is accomplished with a rotation matrix $R_{\mathtt{pi}}$. The typical spacecraft convention is classic 3-1-3 Euler angle rotation. The first rotation is by the argument of perigee ω about the z-axis, followed by a rotation about the intermediate x-axis by the orbital inclination i, and finally a rotation about the second intermediate z-axis by the right ascension of the ascending node Ω. There is a singularity when the inclination i is 0 because the first and third rotations are about the same axis. Going from the orbital frame to the inertial frame is the transpose of the inertial to orbital frame rotation ($R_{\mathtt{ip}} = R_{\mathtt{pi}}^T$). The calculation is done with

$$\vec{r}_{\mathtt{ijk}} = R_{\mathtt{ip}}\vec{r}_{\mathtt{pqw}} = [R_3(-\Omega)R_1(-i)R_3(-\omega)]^T\vec{r}_{\mathtt{pqw}} = R_3^T(-\omega)R_1^T(-i)R_3^T(-\Omega)\vec{r}_{\mathtt{pqw}}, \tag{5}$$

where

$$R_{\mathtt{ip}} = \begin{bmatrix} c\Omega c\omega - s\Omega c i s\omega & s\Omega c\omega + c\Omega c i s\omega & s i s\omega \\ -c\Omega s\omega - s\Omega c i c\omega & -s\Omega s\omega + c\Omega c i c\omega & s i c\omega \\ s i s\Omega & -c\Omega s i & c i \end{bmatrix}, \tag{6}$$

and c is an abbreviation for cosine and s is an abbreviation for sine. In inertial frame $\mathcal{F}_{\mathtt{ijk}}$, the x, y, and z components are

$$\begin{aligned} x_{\mathtt{i}} &= [c\Omega c\omega - s\Omega c i s\omega]x_{\mathtt{p}} + [s\Omega c\omega + c\Omega c i s\omega]y_{\mathtt{p}} + [s i s\omega]z_{\mathtt{p}}, \\ y_{\mathtt{i}} &= [-c\Omega s\omega - s\Omega c i c\omega]x_{\mathtt{p}} + [-s\Omega s\omega + c\Omega c i c\omega]y_{\mathtt{p}} + [s i c\omega]z_{\mathtt{p}}, \text{ and} \\ z_{\mathtt{i}} &= [s i s\Omega]x_{\mathtt{p}} + [-c\Omega s i]y_{\mathtt{p}} + [c i]z_{\mathtt{p}}. \end{aligned} \tag{7}$$

2.3 General Orbits

Rather than starting with the full three dimensional case including all six orbital elements, conjunction prediction algorithms can be evaluated with increasingly more complex combinations of orbits. The case of coaxial and coplanar orbits is the simplest model of two spacecraft orbits and may be defined by only three orbital elements: the semimajor axis a, the eccentricity e, and the true anomaly ν. This case is shown in Figure 4. The apse line of each orbit is aligned and ν_1, ν_2 are measured from the same line.

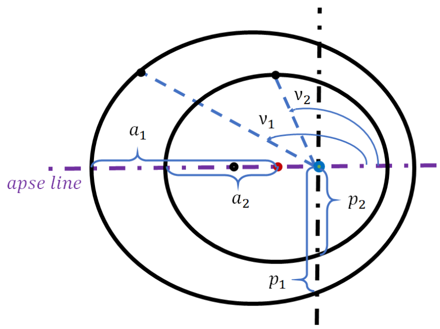

Figure 4: Coaxial Coplanar Orbits

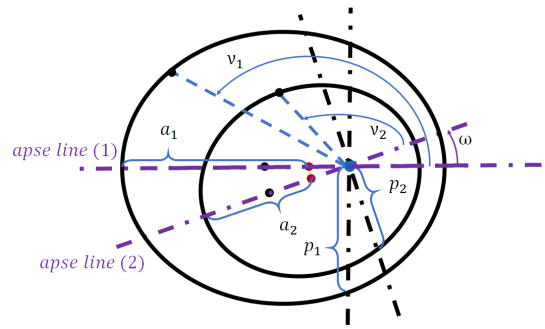

Figure 5: Noncoaxial Coplanar Orbits

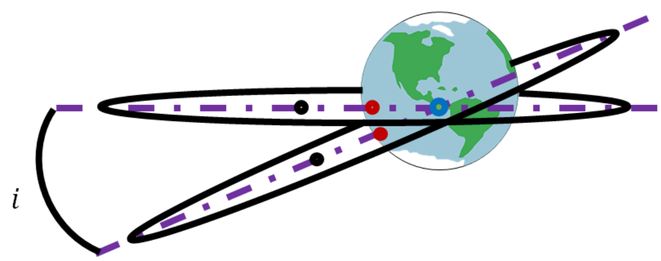

Figure 6: Noncoaxial Noncoplanar Orbits

The case of noncoaxial but coplanar orbits adds one level of complexity with the addition of the argument of perigee ω, to the vector pointing from the center of the earth to perigee $\vec{\mathsf{p}}$. The simplest model of two noncoaxial coplanar spacecraft orbits assumes that Ω is the same for both orbits, and only angle between perigee vectors differs ($\vec{\mathsf{p}}_1 \neq \vec{\mathsf{p}}_2$). Then ω may be used as the angle between the two perigee vectors. This is shown in Figure 5.

Moving to noncoplanar orbits adds the complexity of a third dimension in Cartesian space. The simplest way to model this is to assume that one orbit is on the reference plane (such as the equatorial plane) and the other orbital plane differs by its inclination i. Advancing from this case results in the use of the full six orbital elements. This case is shown in Figure 6.

3 Satellite Conjunction Verification

In this section, we first state the satellite conjunction verification problem. Then, we describe our solution approach which takes advantage of several types of problem structure. Finally, we evaluate the scalability of the proposed method.

3.1 Problem Description

The space debris collision detection problem is, given a list of initial parameters for n satellites and a time bound, prove that no two satellites collide (get within Δ distance) at any point in time. We consider the simplest case where satellite parameters are known constants, and the only uncertainty arises from the satellite angle on its orbit, the true anomaly ν, which has dynamics defined by Equation 3. We want to analyze n-to-n collision avoidance on thousands satellites, and the proof of safety should be done in dense time.

3.2 Solution Approach

Initially, this problem appears to be thousand-dimensional nonlinear system, which is unlikely to be solved by off-the-shelf reachability methods that can only scale to a dozen or so state variables [8]. In order to analyze this system, we must leverage problem structure.

The first type of structure we use that the dynamics is disjoint. Satellites move independently of each other when they orbit, and so the state variables do not interact. The derivative of each satellite's ν variable only depends on itself, and no other state variable. This allows us to convert the n-dimensional reachability problem into n one-dimensional reachability problems, and then recombine the reachable sets together after computation based on common notions of time [11].

The second type of structure used is that the decomposed problems are all one-dimensional. Due to the continuity of trajectories of the system, if no states from the boundaries of the initial set reach an unsafe state, then no states from the whole initial set (including the interior) can reach an unsafe state [9]. To do this, a trajectory would have to pass through a state that was reached by a boundary point, because trajectories must be continuous. For a high-dimensional system, this does not help too much since the boundary of an n-dimensional set will be an $n-1$ dimensional set, that still contains an infinite number of points. In the case of a 1-d system, however, the boundary of an initial set of states, an interval, is just the lower and upper bounds of the interval. This means that we only need to analyze trajectories from the lower and upper bounds and assure they are safe.

Since we only analyze specific trajectories of a system, we take a numerical verification approach and assume that numerical simulations can produce accurate system behaviors. Using an off-the-shelf Runge-Kutta numerical simulator, we will simulate the lower and upper bounds of ν to predict the time-evolved system states. This also can be used for analyzing intervals of time. Since orbits do not change direction, we can get the range of reachable ν values in an interval of time by sampling the endpoints of the range at the start of the time interval and at the end of the time interval, and taking the minimum and maximum over all four values.

A further challenge with this model is that the safety property is not expressed in terms of ν, but requires converting ν to Cartesian x, y and z coordinates before doing a distance check. This is done according to Equation 7, which involves computing sine and cosine of ν as well as some of satellite orbit parameters. To soundly convert from ranges of ν values to ranges of x, y and z coordinates, we use interval analysis methods [15]. This means the reachable set we use to check for collisions will always be a box, with the property that as the ν uncertainty tends towards zero, the volume of the output box will also tend towards zero.

The collision check needs to be done in an n-to-n fashion. We consider three ways to implement this check: exhaustive search, k-d trees, and AABB trees.

The exhaustive search method of conjunction prediction evaluates the distance between every pair of objects. If using an off-the-shelf reachability tool, this is likely the method that would be used. We can take advantage of symmetry in the distance check in order to reduce the number of required checks, but we still expect this method to scale quadratically. The number of required checks with this approach is $\frac{n(n+1)}{2}$.

However, since we know our problem deals with 3-d spatial coordinates, we can leverage better data structures to reduce the number of checks in the safety condition. The first idea was to use k-d trees. A k-d tree is a binary structure in which the median distance of all points along one axis is determined, grouping the points into those on either side of the median. This process is repeated for each dimension at each level of the tree before being recursively applied, until each level of the tree contains only two points. This enables the tree to be searched relatively quickly for nearest neighbors.

Although k-d trees are efficient for nearest-neighbor point searches, after computing reachability we will have sets of states. It is unclear how to extend k-d trees to do collision checking between sets that may have different sizes.

The last idea we considered was using axis-aligned bounding-box (AABB) trees [4]. AABB trees provide a bounding-volume at each inner-node in the tree which that contains the volume of children of that node. They are often used for real-time n-to-n collision detection in video games due to their flexibility and high-performance. Since they do support encoding sets of states, we can insert the reachable regions for each satellite into an AABB tree, and then efficiently check for collisions between sets.

The last optimization we do deals with the time-stepping strategy. Since we want to do analysis with a time bound on the order of days, ideally we would consider large time steps (intervals of time to be analyzed). However, large time steps will lead to large ranges of ν reachable for each satellite, which would leave to large boxes in Cartesian coordinates. This could cause false positives when checking for collisions. Using small time steps, however, would lead to a large number of steps that needs to be analyzed. For this reason, we propose to use a CEGAR-based variable time stepping strategy. We start with a minimum time step such as 0.1 second, and do a step of that size. If the step succeeds (no collisions are detected), we advance time and double the step size. If collisions are detected, we halve the step size and try again without advancing time. If the step size is ever the minimum (0.1 seconds) and a collision is detected, we give up and output that a collision is possible.

3.3 Evaluation

We evaluate the proposed approach on two aspects. First, we measure the runtime of various aspects of the algorithm, comparing a brute force collision check versus the AABB tree method. Next, we analyze the effectiveness of the dynamic time-stepping strategy.

We analyze the system by generating uniformly random satellite parameters. We are concerned with satellites in low-earth orbit, so the satellite height in km, h, is chosen from $[6951, 7221]$. The satellite orbits we are interested are generally round so we use an eccentricity e in the range $[0, 0.004]$. The angles ν, ω, and Ω are all taken from the range $[0, 2\pi]$ and i is taken from the range$[0, \pi]$.

In our Python-based implementation, we use `scipy`'s `integrate.RK45` method for satellite simulation, and the `pyibex` library to perform interval arithmentic. The measurements are performed on an Intel i5-5300U CPU running at 2.30GHz with 16GB main memory.

First we compare the brute force search method versus the AABB tree approach. We using a small collision distance of 100 meters (in the infinity norm), with small initial ν uncertainty of 0.0001. With these parameters, no collisions should exist in the time frames analyzed.

In the 100 satellite case, with a reach time bound of one hour, the brute force method took 6.9 seconds, whereas the AABB tree approach used 4.8 seconds. Increasing the number of satellites to 1000, and decreasing the reach time bound to one minute, the brute force method needed 28.3 seconds (about half of real-time), whereas the AABB approach completed in 5.4 seconds. Finally, with a reach time bound of 10 seconds and 5000 satellites, the brute force method required 260 seconds, whereas the AABB tree approach needed 12.5 seconds. As expected, the AABB method was significantly faster, but still it was still slower than real-time for the $n = 5000$ case.

Table 2: Computation Time Comparison

Number of Satellites	Time Bound	Brute Force	AABB (dynamic)	AABB (static)
100	1 hr	6.9 s	4.8 s	
1000	1 min	28.3 s	5.4 s	80 s
5000	10 s	260 s	12.5 s	

To analyze the effect of the dynamic time-stepping strategy, we used the AABB method with 1000 satellites for one minute of reach time. A static stepping strategy which always used the minimum step of 0.1 seconds required 80 seconds to complete, compared with 5.4 seconds with the dynamic time stepping approach. In this case, without dynamic times steps, the method runs much slower than real-time. Further, in the dynamic case, the majority of the runtime, 80%, is spent in the interval arithmetic library constructing the boxes of reachable Cartesian coordinates. This means that AABB trees are sufficiently fast so the collision checking is not the bottleneck. The results are summarized in Table 2.

Overall, our evaluation shows that reachability analysis, with the enhancements described, can be viable for the space debris collision avoidance problem with a few thousand objects. If we want to scale to tens of thousands or hundreds of thousands of objects, however, further speedups are needed. In the 5000 satellite case, all our current methods were slower than real-time, which is unusable in practice. One idea for speedup would be to consider different time-steps for different objects. Currently, if there is any collision, all objects get rolled back and a smaller time-step is attempted. As we add more objects, the potential for collisions will increase, and so time-step rollbacks will become more common. We will investigate per-object time-steps in future work.

4 Conclusion

In this paper, we presented a space debris collision detection problem and initial analysis approach. In the current setup, we considered orbital dynamics with a single variable, the angular position ν. An immediate extension, then, would be to consider ranges for the other orbital parameters, corresponding to errors in sensing space objects. For fixed parameter ranges, we might directly be able to use the existing interval arithmetic approach without modification, simply using the interval ranges when computing the Cartesian coordinates. A more realistic setup, however, would give these parameters their own dynamics that spread out over time. In particular, we could consider the semi-major axis and eccentricity to be variables with a small initial uncertainty that can slowly diverge with a known minimum and maximum rate. This would mean we no longer have 1-d systems where we can use simulations for reachability, but instead we would need to use more sophisticated nonlinear reachability methods [7].

Although we focused on orbital dynamics, many of the problems encountered are more general. For example, swarm robotics can also require verifying n-to-n collision avoidance, and would benefit from an AABB structure to check the safety property. Further, although these system do not have 1-d dynamics, they do consist of isolated systems that generally fly independently, and so decomposition may still be possible.

References

[1] S. Alfano. Determining satellite close approaches, part 2. *Journal of the Astronautical Sciences*, 42:143–152, 1994.

[2] K. T. Alfriend, M. R. Akella, J. Frisbee, J. L. Foster, D.-J. Lee, and M. Wilkins. Probability of collision error analysis. *Space Debris*, 1(1):21–35, 1999.

[3] S. Bak, S. Bogomolov, T. A. Henzinger, T. T. Johnson, and P. Prakash. Scalable static hybridization methods for analysis of nonlinear systems. In *International Conference on Hybrid Systems: Computation and Control*, 2016.

[4] G. v. d. Bergen. Efficient collision detection of complex deformable models using aabb trees. *Journal of Graphics Tools*, 2(4):1–13, 1997.

[5] K. Chan. Spacecraft collision probability for long-term encounters. *Advances in the Astronautical Sciences*, 116(1):767–784, 2003.

[6] N. Chan and S. Mitra. Verifying safety of an autonomous spacecraft rendezvous mission. In G. Frehse and M. Althoff, editors, *ARCH17. 4th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 48 of *EPiC Series in Computing*, pages 20–32. EasyChair, 2017.

[7] X. Chen, E. Abraham, and S. Sankaranarayanan. Taylor model flowpipe construction for nonlinear hybrid systems. *Real-Time Systems Symposium*, 2012.

[8] X. Chen, M. Althoff, and F. Immler. Arch-comp17 category report: Continuous systems with nonlinear dynamics. In G. Frehse and M. Althoff, editors, *ARCH17. 4th International Workshop on Applied Verification of Continuous and Hybrid Systems*, volume 48 of *EPiC Series in Computing*, pages 160–169. EasyChair, 2017.

[9] T. Dang. *Verification et synthese des systemes hybrides.* PhD thesis, INPG, oct 2000.

[10] K. J. DeMars, Y. Cheng, and M. K. Jah. Collision probability with gaussian mixture orbit uncertainty. *Journal of Guidance, Control, and Dynamics*, 37(3):979–985, 2014.

[11] J. Green. Compositional bounded reachability using time partitioning and abstraction, 2012.

[12] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Algorithmic analysis of nonlinear hybrid systems. *IEEE transactions on automatic control*, 43(4):540–554, 1998.

[13] S. A. Hildreth and A. Arnold. Threats to u.s. national security interests in space: Orbital debris mitigation and removal. Technical report, Congressional Research Service, 2014.

[14] B. HomChaudhuri, M. Oishi, M. Shubert, M. Baldwin, and R. S. Erwin. Computing reach-avoid sets for space vehicle docking under continuous thrust. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 3312–3318. IEEE, 2016.

[15] L. Jaulin. *Applied interval analysis: with examples in parameter and state estimation, robust control and robotics*, volume 1. Springer Science & Business Media, 2001.

[16] C. Jewison and R. S. Erwin. A spacecraft benchmark problem for hybrid control and estimation. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 3300–3305. IEEE, 2016.

[17] T. T. Johnson, J. Green, S. Mitra, R. Dudley, and R. S. Erwin. Satellite rendezvous and conjunction avoidance: Case studies in verification of nonlinear hybrid systems. In *International Symposium on Formal Methods*, pages 252–266. Springer, 2012.

[18] R. P. Patera. Space vehicle conflict-avoidance analysis. *Journal of guidance, control, and dynamics*, 30(2):492–498, 2007.