AstroLibrary: A Library for Real-time Conjunction Assessment and Optimal Collision Avoidance

Shawn SH Choi[*] [(1)(2)], Peter JH Ryu[*] [(1)(2)], Minwoo Ji[(2)], John Kim[(1)(2)], Lowell Kim[(1)(2)], Kyuil Sim [(1)(3)], Jaedong Sung[(4)] , Jae Wook Song[(3)], Misoon Mah[(5)], and Douglas DS Kim[(1)(2)§]

[(1)] SpaceMap Inc, 1103-1 IT/BT bldg., 222, Wangsimni-ro, Seongdong-gu, Seoul, Korea
[(2)] School of Mechanical Eng./Voronoi Diagram Research Center, Hanyang University, Seoul, Korea
[(3)] Department of Industrial Engineering, Hanyang University, Seoul, Korea
[(4)] SSA Research Office, Korea Aerospace Research Institute (KARI), Daejeon, Korea
[(5)] M&K Research and Development Inc., VA 20155, USA
** Equally contributed. § Corresponding author: douglas.kim@spacemap42.com*

ABSTRACT

Geospace is crowded and will be more crowded. This trend increases the probability of collisions between space objects rather rapidly. As objects fly at very high speed, collision consequence is catastrophic. However, accurate and efficient conjunction assessment (CA) and collision avoidance (COLA) have long been challenges even with current space catalogues of O(10^4) size. As space catalogue size will surge rapidly because of increased number of new satellites, improved sensor capabilities, and the Kessler syndrome, it will be worse unless a paradigm-transforming computational method is devised. Here we present the SpaceMap method that can perform real-time CA and near real-time COLA for O(10^6) or more objects, provided that the spatiotemporal proximity among satellites is represented in a concise data structure through a preprocessing. The theoretical and computational basis is the Voronoi diagram which has been known as the most concise and efficient data structure for spatiotemporal reasoning among many objects in 2D and 3D spaces. The algorithms are implemented in C++ and is available as AstroLibrary which has RESTful APIs and Python package that can be called from application programs. With the library, efficient application programs can be easily developed for challenging spatiotemporal problems by anyone with elementary programming skill. Experimental results are also presented.

1 Introduction

There are many space objects. There will be many more because of new satellites and improved sensor capabilities. Starlink and Guowang constellations of O(10^4) satellites and OneWeb and Kuiper constellations of O(10^3) satellites are well-known. According to a recent study of radio spectrum filings of International Telecommunication Union [1], there might be more than one million new satellites. Space debris is another story of space objects. According to ESA [2], there are O(10^4) space objects of larger than 10 cm, O(10^6) objects of larger than 1 cm, and O(10^8) objects of larger than 1 mm. NASA reported similarly [3]. The Kessler syndrome is another key consideration. On the other hand, only a small subset of the space objects is observed and catalogued. E.g., Space-Track has 44,700 objects as of Oct 2023 [4]. As space objects fly at an extremely high speed, collision impact can be catastrophic even with a tiny debris.

Hence, the prediction and avoidance of potential collisions of space assets have been important research subjects. However, the current computing technology struggles in processing the current catalogue of O(10^4) size for CA/ COLA. We view as the main hurdle is in software rather than hardware. The principal conjunction algorithm is, namely, the **1984 three-filter algorithm**, which chops the timeline into a set of

mutually exclusive time segments and checks the conjunction between every object pair in each time segment. While this seminal algorithm has profoundly contributed to the field, it has a few fundamental drawbacks for the future of space industries. It is designed to solve the spatial problems between two objects at a time, not for many objects. It is coordinate dependent. The memory requirement is critically dependent on the length of time steps is critical: Memory requirement is inverse proportional to the length of time step. E.g., according to our experiment, using only about 21,000 objects from Space-Track, more than 700GB memory is easily consumed for a 24-hour window with 1-sec time step. There are many prior studies about accelerating the three-filter algorithm using hashing mechanisms such as 3D buckets, kd-trees, or sieves [5-7]. Another important perspective: The computed results were not used for any other purpose at all except CA/COLA - they were abandoned.

Space environment is rapidly changing from three seemingly colliding perspectives: (i) the increases in the size of the space catalogue, (ii) the needs for real-time CA/COLA, and (iii) the rise of new applications with more challenging computational requirement, such as optimization problems.

The space catalogue size is expected to surge, perhaps faster than an exponential growth. The Intelligence Advanced Research Activity (IARPA) announced in July 2023 the Space Debris Identification and Tracking (SINTRA) program [8], which aims to detect, track, and characterize space debris as small as 1 mm. NASA issued a challenge, due in November 2023, of detecting/characterizing, tracking, and remediating debris of 1 mm ~ 10 cm size [9]. There are parallel commercial space-based SSA efforts for detecting mm- to cm-sized objects, e.g., ARCA Dynamics, Digantara [10], etc. We anticipate these, and expected similar efforts to follow, will result in humongous catalogues which will make existing algorithms hard to produce CA/COLA solutions at reasonable speed.

Despite the expected huge catalogue, critical use-cases will add another dimension to the challenge. Human-on-board is the most critical use-case because astronauts' safety must be guaranteed, e.g., orbital tours. If the current transportation cost of O($1,000)/kg to LEO reaches at O($100)/kg, e.g., via Starship, both passenger and freight transportations between cities might be realized to supplement air transportation, if not replace it. For these use-cases, "fast computation" is not sufficient: The "**real-time CA/COLA**" will be one of the determinants for the success of this transportation type after updating catalogue itself in real-time with real-time obtained new state data.

We want to solve the CA/COLA problems in the following setting where $\otimes$ represents a Cartesian product:

$$\text{Huge catalogue} \otimes \text{Real-time CA/COLA} \otimes \text{Real-time data update} \otimes \text{Optimization problems}. \quad (1)$$

Here we present a unified computational framework that performs real-time CA/COLA for the catalogues of O(10^6) ~ O(10^8) size, provided with a moderate amount of computational resources. The theoretical basis of the technology is the Voronoi diagram which has been known as the most concise and efficient data structure for spatiotemporal reasoning among many objects in 2D and 3D spaces. We first preprocess space objects to produce important spatiotemporal events in timeline. Then, CA can be answered in real-time by simply scanning the events. COLA can also be solved very quickly by performing the CA for multiple maneuver alternatives. Note that each alternative may cause secondary and tertiary conjunctions which are known computationally demanding to resolve [11]. The CA of a trajectory through M waypoints is processed in the O(M) time given the preprocessing. Beware that it is independent of catalogue size! If N cores are used for computation, it can be answered in the O(M/N) time. We emphasize that our algorithm is strong-scalable. A variation of the CA algorithm can be later used to marginally update the catalogue using the SSA data collected in real-time during flights. Diverse trajectory types are homogeneously handled in our framework: stationary orbits, launch trajectories, orbit transfer trajectories, etc. The SpaceMap method also efficiently solves more challenging spatiotemporal optimization, intelligence

problems with unparalleled efficiency. We are well-aware of the NASA's concern about the heavy computation requirement of CA/COLA algorithms [11].

The proposed method can be verified at the SpaceMap on AWS: **spacemap42.com**. The algorithms are implemented in C++ and are available as **AstroLibrary** which provides RESTful APIs and a Python package that can be embedded in application programs. The C++ version will also be available. We developed it for all stakeholders in space to have easy access to our algorithms so that application programs for challenging spatiotemporal problems can be easily and conveniently developed by anyone with elementary programming skill. We believe that the current humongous space investments will be sustainable only if they make profits and profits can be maximized with good software for both safety and optimization of space assets. We show experimental results of some functions related to CA/COLA using the LEO TLE data downloaded from Space-track. Unless otherwise stated, computational environment is as follows - OS: Ubuntu 20.04; CPU: 64core / 128thread (2.7GHz); Memory: 512GB.

2 Related prior studies

Collision has always been an issue. The intuitive 1984 three-filter algorithm was seminal for the space community [12]. The first perigee-apogee filter quickly reduces combinatorial search space by comparing perigees and apogees of two objects. The second path filter screens out an object pair if the minimum distance between their orbital geometries exceeds threshold. The third time filter chops timeline into a set of short time intervals and solves the conjunction problem of each object pair (i.e., the distance problem between two objects at the same moment). This algorithm has significantly contributed to the space community but has fundamental limitations for the changing space environment in New Space Age.

Since 1984, there were abundant studies regarding CA. They were mostly about the acceleration of the three-filter algorithm using geometric hashing with 3D buckets or kd-trees. In 2002, the smart sieve method was developed as an improvement that considers the escape velocity in filtering distance and has been the best-available algorithm for CA [5]. The smart sieve method inherits the limitations of the three-filter algorithm. Most commercial systems implement the sieves. Observe that all acceleration schemes in fact attempted to reduce the search space of pairwise conjunction computation.

We have introduced the Voronoi diagram approach to CA/COLA in AMOS2017 [13] and have improved the algorithm since then [13-15]. Voronoi diagrams are the most concise data structure for representing spatial relationship among particles, called generators, in space. It is the tessellation of the space where each tile, called Voronoi cell, is the set of locations which are closer to the corresponding generator than any others. It is particularly well-suited for efficiently solving spatial reasoning problems among generators in 2D and 3D spaces. Suppose that generators move, and we want to solve spatiotemporal, i.e., space-time, reasoning problems among the moving generators. We solve this problem by first constructing the dynamic Voronoi diagram DVD of the moving generators and take advantage of the spatial reasoning power of Voronoi diagrams at distinct moments. For details, refer to [16].

3 Catalogue conjunctions

Catalogue conjunction is the conjunction between catalogue objects. Astro-1 can answer the CA of a particular catalogue object or of all catalogued objects in a few milli-seconds using the preprocessing DB. We benchmarked the quality of conjunction solutions against Celestrak [17]. Astro-1 synchronizes the TLE download time with Celestrak: Three times every day at 0800, 1600, and 2400 UTC.

. 1 shows the correlation of the CA solutions produced by Astro-1 and Celestrak for the Top-10, Top-100, Top-1,000 and Top-10,000 conjunctions. **Error! Reference source not found.**(a - d) are DCAs. Referring to

the red diagonal, Astro-1 tends to be slightly more conservative than Celestrak while their correlation is high. We confirmed that both programs produced identical conjunction set of catalogue object pairs. Repeated experiments reproduced identical results. We also compared the Astro-1 solutions with those produced by STK to reach the same conclusion. **Error! Reference source not found.**(e - h) are difference of TCAs between Astro-1 and Celestrak: The average is very close to zero with tiny deviations. **Error! Reference source not found.**(i - l) are the probability of collisions in the log-scale with base 10. Astro-1 uses the maximum collision probability formula proposed by Alfano [18, 19], as Celestrak does, given as

$$P_{c\,max}(\sigma_x^2) = \exp\left(-\frac{x_e^2}{2\sigma_x^2}\right)\left[1 - \exp\left(-\frac{\alpha r_A^2}{2\sigma_x^2}\right)\right] = f(r_A, x_e) \quad (2)$$

where α is aspect ratio AR given as AR = σ_x/σ_z = 3, r_A is the hard-body radius HBR, x_e is DCA, σ_x^2 is the error variance that is given as $\sigma_x = x_e/\sqrt{2}$ = DCA/$\sqrt{2}$. Hence, Pc is a function of HBR and DCA. As Astro-1 and Celestrak have a good agreement in their DCAs, the difference of the probability profile lies in the different methods two programs estimate HBR. We retrieve the geometric attributes of both primary and secondary objects from the ESA database and calculate the minimal sphere enclosing the objects. Then, we define HBR by adding the radii of the two enclosing spheres. If geometric attributes are not available from the ESA database, we assign a default radius of 1 m now. The access to the equivalent NASA database is desperately needed. We expect the probabilities of Astro-1 and Celestrak will be more consistent when we estimate HBR better.

4 Phantom conjunctions

Phantom is a spatiotemporal object not listed the catalogue, and **phantom conjunction** is the conjunction between a phantom and an object in the space catalogue. A phantom may be a new satellite that one plans to insert to orbit, a spaceplane that flies inter-orbital space, a hypothetical object. A phantom propagates together with catalogue objects and is associated with its trajectory in the form of TLE, ephemeris, etc. It turns out that the “phantom” concept is useful for a variety of applications.

The efficient assessment of phantom conjunctions has many immediate, critical applications. Examples include the optimal design of constellations for carrying our missions safely and efficiently, constellation evaluation for insurance policy decision, etc.

Suppose that we want to efficiently perform the CA of a phantom with the preprocessing DB of the catalogue. The basic idea of the phantom CA algorithm is to do the spatiotemporal neighborhood search in the Voronoi diagram and to perform the CA. The key determinant is how to reduce the solution space as much as possible while no solution is missing.

Fig. 2(a) shows the computation time for phantom conjunctions by Astro-1 using 100 phantoms for 24-hour window. Old TLE data of 100 Starlink satellites played the role of phantoms. The top black lines show the computation time of our current code: The solid and dotted lines correspond to the conjunctions computed within the thresholds of 100 km and 10 km, respectively. 100 km threshold means that the conjunctions within 100 km distance are found. The difference is negligible. Note that about half of the time is taken for uploading the preprocessing DB to memory. The blue lines correspond to the scenario that the preprocessing DB is resident on memory. We expect to achieve the bottom, red line after the current re-engineering of the code. We want to emphasize that doubling computing power halves computation time: Our algorithm is strong-scalable. Fig. 1(b) shows the number of conjunctions which are defined by a phantom w.r.t. threshold distance over a 24-hour window. We estimate that about 10% more catalogued objects are visited during CA. The computation cost of finding the spatiotemporal neighborhoods is tiny as shown in Fig. 2(a).

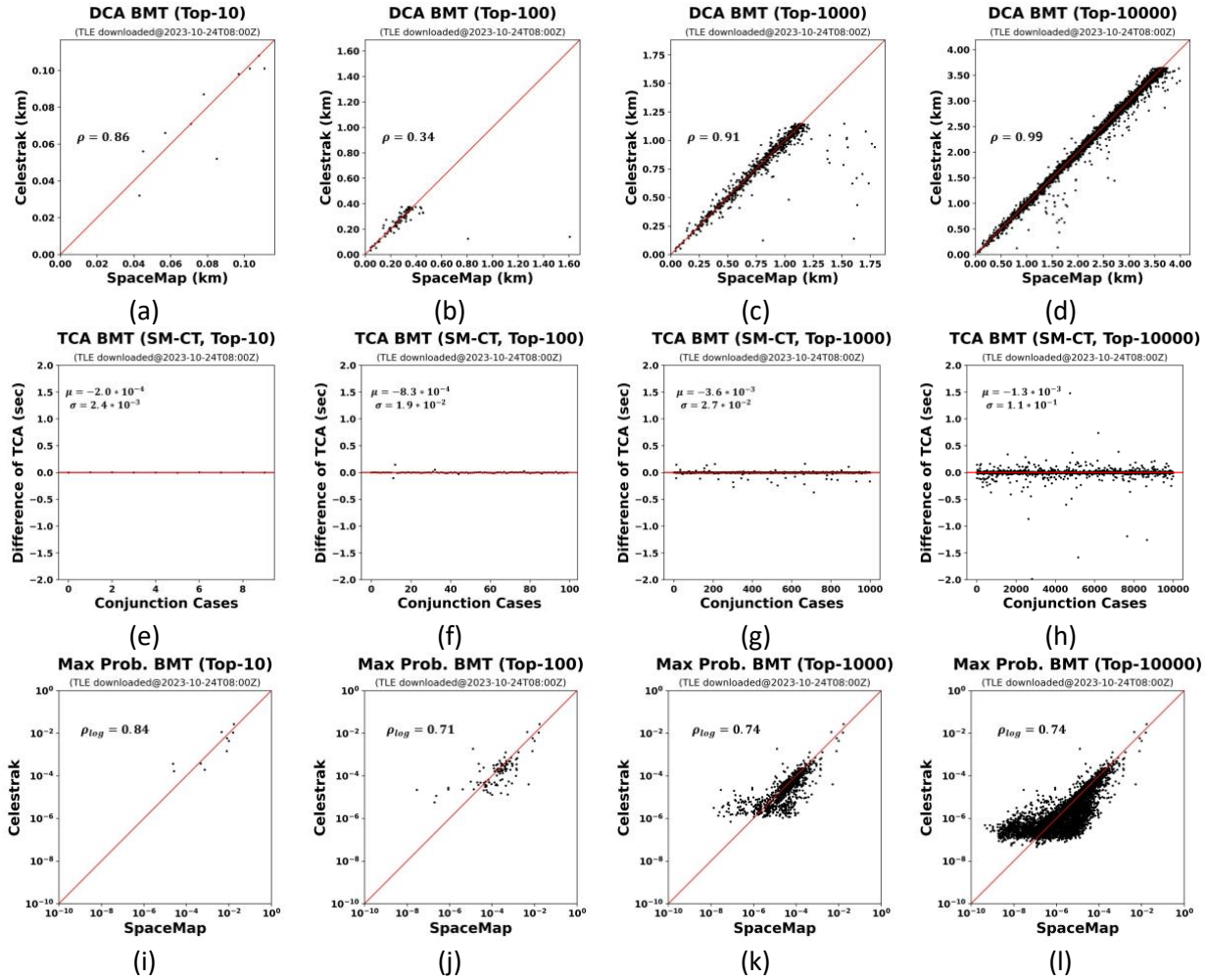
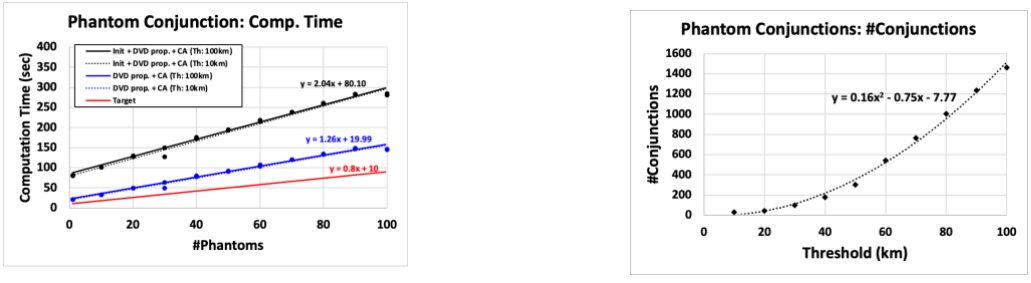


Fig. 1. Comparison of the CA solutions between SpaceMap and Celestrak's Socrates. TLE downloaded at 08:00, OCT 24, 2023 (UTC). Celestrak conjunction data downloaded at 14:00, OCT 24, 2023 (UTC). (a - d) Distances of closest approach (DCA). (e - h) Differences of the time of closest approach (TCA). (i - l) Probability of collision. (a, e, i) Top-10 conjunctions. (b, f, j) Top-100. (c, g, k) Top-1,000. (d, h, l) Top-10,000.



(a) (b)

Fig. 2. Phantom conjunction. 24-hour. TLE data (21,000 objects). Altitude mask 500-600km used. Old TLE data of 100 Starlink satellites played the role of phantoms. Th: Threshold. Each phantom is associated with waypoints of 1Hz sampling rate, i.e. 86,400 (= 3,600*24) points. (a) Computation time. Threshold difference has tiny effect. Top black: Current code. Middle blue: Preprocessing DB on memory. Red: Expected after re-engineering. (b) The number of conjunctions of a phantom w.r.t. threshold distances.

5 Optimal collision avoidance

Suppose that we want to perform a COLA maneuver of a spacecraft which can be a satellite, spaceplane, etc. We claim that Astro-1 can produce the maneuver plan better and faster than anyone else can produce with a same catalogue. The idea of the algorithm is simple: Generate-&-Test. We first generate sufficiently many candidate trajectories and evaluate them quickly by taking advantage of the fast phantom CA algorithm. In other words, treat each candidate trajectory as a phantom. Then, we choose the best one.

Let c(p, s, dca, tca) = conjunction(primary, secondary, DCA, TCA) be the conjunction between p and s with dca and tca that we are trying to avoid. Let $W[t_s, t_e)$ be the time window that we want to evaluate the quality of a maneuver trajectory. Note that tca ∈ W. Let TRJ = $\{trj_1, trj_2, .., trj_n\}$ be the set of *n* trajectories where each is associated with a spatiotemporal chain, i.e., TLE, ephemeris, etc. For convenience of discussion, suppose that each trajectory is associated with a set of m time-stamped waypoints, e.g., $trj_i = \{(x_{i1}, t_{i1}, \psi_{i1}), (x_{i2}, t_{i2}, \psi_{i2}), .., (x_{im}, t_{im}, \psi_{im})\}$ where x_{ij} is the spatial coordinate of the waypoint that the spacecraft passes through at the time t_{ij}. A waypoint is associated with a non-negative propellant consumption ψ_{ij}.

We want to evaluate the quality of all elements in the TRJ set. Suppose that W is 24-hour long and we want to start the maneuver in the middle of W. In other words, we want the maneuver to begin at tca-12h and evaluate the quality of the maneuver until tca+12h. Hence, $W[t_s, t_e) = W$[tca-12h, tca+12h). Note that some waypoints in the front of each trajectory is associated with positive propellant usages.

How can we define the quality of a trajectory? There may be abundant definitions.

Formulation 1. Maximum miss distance with propellant consumption constraint: Constrained integer linear programming problem

A simple yet effective measure of trajectory quality is the miss distance. We want to maximize the miss distance of trajectory and formulate a MaxMin problem. Suppose that each trajectory corresponds to a phantom that follows the trajectory, and we want to find its miss distance against the entire catalogue objects. Let δ_{ijk} be the distance between x_{ij}, i.e., the j-th waypoint of trj_i and the k-th object of the catalogue at the time t_j. Let y_i be a binary decision variable such that $y_i = 1$ if trj_i is chosen; Otherwise, $y_i = 0$. In addition, we add a constraint of maximum propellant consumption, say Ψ. Then, this formulation can be given as a constrained linear integer programming problem as follows.

$$\underset{i \in TRJ}{Max} \sum_{j \in trj_i} \underset{k \in CAT}{Min} \{\delta_{ijk} \cdot y_i\} \tag{3-1}$$

$$y_i \cdot \sum_{j \in trj_i} \{\psi_{ij}\} \leq \Psi \text{ for i =1,2, ··, n.} \tag{3-2}$$

where y_i is a binary decision variable for i=1, 2, ⋯, n and CAT is the catalogue set. The subscript and set symbol are intentionally abused for notational simplicity. Consider a trajectory trj ∈ TRJ. To find the miss distance of trj, we need to calculate the DCA of each of the m waypoints of trj. The calculation of the DCA of a waypoint wp requires to do the CA of wp against all catalogue objects at the moment that wp is defined. Hence, for 1Hz sampling rate over 24h with 21,000 catalogue objects, each trajectory requires more than 1.8 billion (specifically 1,814,400,000 = 86,400 * 21,000) distance calculations between two points unless a good method helps. This cost is linear to catalogue size.

Voronoi diagram, however, helps reduce this computation significantly. For a phantom following trj, defines the miss distance, i.e., DCA, with one of the catalogue objects in the Voronoi neighborhood in timeline. Given a 3D Voronoi diagram, the average number of neighbor objects sharing the Voronoi face is usually 15 ~ 16 for random object distribution. Note that the 2D and 3D kissing numbers are 6 and 12, respectively. Suppose that 20 be a relaxed upper bound. Then, less than 1.8 million (specifically 1,728,000

= 86,400 * 20) distance calculations are required for each trajectory. Voronoi diagram yields more than 1,000 times computation reduction for each trajectory. It is important to note that this cost of the Voronoi-based approach is independent of catalogue size.

Formulation 2. Optimization of the weighted sum of miss distance and propellant consumption: Unconstrained integer linear programming problem

Suppose we move the propellant constraint into the objective function. Then we have a multi-objective optimization problem that aims to maximize the minimum distance and minimize propellant usage. In general, a multi-objective optimization problem results in a set of points in the objective function space, referred to as the Pareto optimal solutions that lie on the Pareto front (the set of all non-dominated solutions). In this case, the speed of calculation for d_{ijk} would facilitate the solution process.

$$\underset{i \in TRJ}{Max} \left\{ \sum_{j \in trj_i} w_1 \cdot \underset{k \in CAT}{Min}\{\delta_{ijk} \cdot y_i\} + w_2 \cdot \{\sum_{j \in trj_i} \psi_{ij}\} \cdot y_i \right\} \quad (4)$$

6 K-nearest neighbors in timeline

Consider a phantom following through a trajectory trj. We want to find the catalogue object that approaches closest to the phantom during the window. The object, say o_{1t}, can be found by checking the distance between its waypoint and the locations of the catalogue objects, both at t in the window. Let us denote o_{1t} as 1-nn denoting the closest one. The red curve in **Error! Reference source not found.**(a) shows the profile of distances between o_{1t} and its nearest neighbor 1-nn at every second. The red star denotes the minima and defines the miss distance of the phantom. **Error! Reference source not found.**(b) and (c) show the distance profile of the phantoms where the miss distances are middle and largest among the miss distances defined by all phantoms. The stars denote the minima of the curves. The distance profiles are all by-produced from the CA algorithm.

Given a phantom, suppose now that we want to find the object o_k which defines the k-th smallest distance to the phantom over the window, thus called k-NN. Recall that the solution process of the phantom conjunction problem by-produces k-NN. **Error! Reference source not found.** (d) shows the closest approach distance (CAD-1) graph of the 100 phantoms. The horizontal axis denotes different phantoms. The vertical axis denotes the miss distance of each phantom. **Error! Reference source not found.**(e) shows the CAD-k graph, k = 3: The blue and green curves correspond to the second and third closest approaching objects to each phantom, respectively. **Error! Reference source not found.**(f) shows the weighted sum of the curves. CAD-w3 corresponds to the uniformly weighted sum of the three curves of CAD-3. CAD-w5 similarly corresponds to CAD-5. All these curves can be produced with negligible amounts of additional computation.

The optimal COLA formulations above are deterministic. Note that an optimization model may be defined to incorporate uncertainties using the nearest neighbor information available in our k-NN output.

7 Incremental update of preprocessing database

The SpaceMap technology is based on the preprocessing database DB. Hence, it is important to construct DB efficiently, concisely, and robustly. This goal applies to both offline and online constructions. “Offline” means the batch construction of DB of catalogue on regular basis, e.g., three times a day with the entire catalogue downloaded from the SSA databases, e.g., Space-Track TLE DB. “Online” means the incremental update of DB by reflecting marginal change of the catalogue state, where the change implies (A) that of states of some catalogue objects, (B) the deletion of some catalogue objects, and (C) the insertion of new

objects to catalogue. There are many use-cases for real-time, on-line database update. A variation of the phantom conjunction algorithm can be used for this purpose.

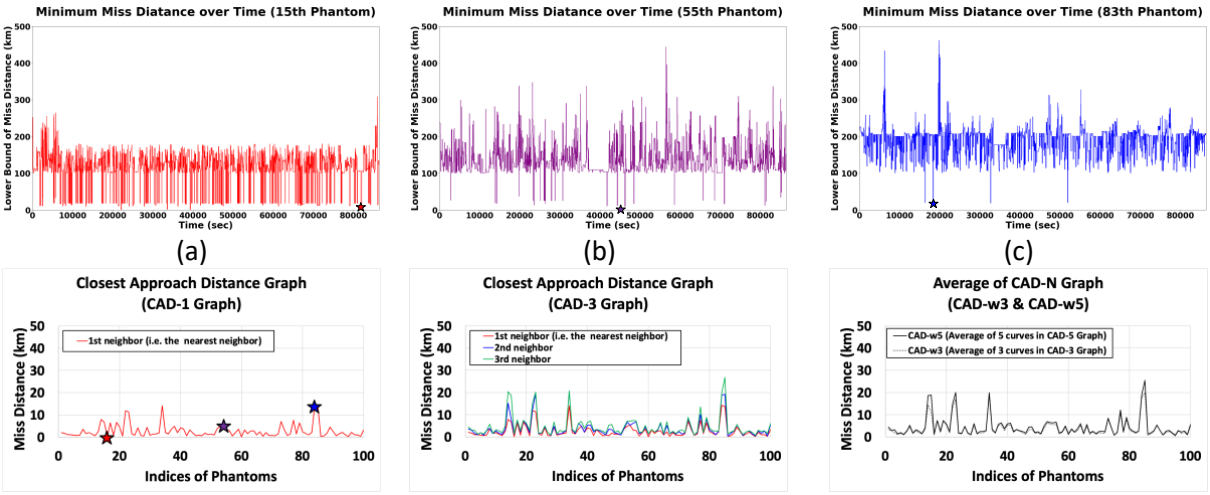


Fig. 3. Nearest neighbours and their applications. (a,b,c) Nearest neighbour profiles of three phantoms over the window. Horizontal axis: Timeline (24h). Vertical axis: Lower Bound of miss distance to the nearest neighbour. Different colours correspond to the different phantoms. Red: The phantom which has the smallest miss distance of all phantoms over the time. Purple: The phantom which has the 70th smallest miss distance of all phantoms over the time. Blue: The phantom which has the largest miss distance of all phantoms over the time. (d,e,f) Closest Approach Distance (CAD) graphs of the 100 phantoms. Horizonal axis: different phantoms. Vertical axis: Miss distance of each phantom against the entire catalogue objects during the window. (d) The closest approach distance (CAD) graph. (e) The CAD-3 graph, i.e., the the distances to the third closest approaching objects. (f) The weighted sum of the k curves in CAD-k.

8 SpaceMap's 42 services for space industries

SpaceMap's vision is to contribute to humanity with safer, more efficient and sustainable space. **Our mission** is to materialize the vision by developing and promoting real-time algorithms for many, if not all, critical decision-making problems for space assets using moderate computational resources. SpaceMap provides **three access-points** for users. AstroLibrary is for application program developers. Astro-1 and AstroOrca are the other two for owners/operators on the platform. Astro-1 is the real-time CA/COLA in a unified manner and AstroOrca is for hard space optimization problems and intelligence problems.

SpaceMap has **two orthogonal opinions to space sustainability**: physical and financial. For space to be sustainable and valuable natural resources for humanity, collisions in space should be predicted and avoided as much as possible. We answer it with Astro-1. For space economy to be sustainable, investments should be sustainable. For investment to be sustainable, prior investments should be profitable. For an investment to be profitable, the utilization of the invested resources must be optimized. Due to the extreme conditions of space industry, i.e., extreme speeds for both object flights and catalogue size increases, optimization problems are hard to solve. On the other hand, there are and will be many space optimization problems. The solution methods to these problems will require to be developed quickly and easily. A good library is necessary for sustainable space investment. AstroLibrary is our answer to this issue. This is actually what had happened with the math library, e.g. math.h, in computer programming in 1980s. No more reinvention of the wheels in space!

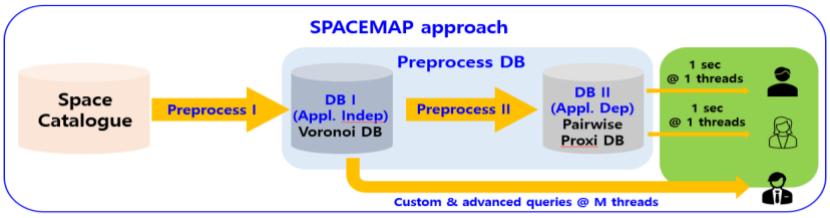


Fig. 4. SpaceMap solution to the following fundamental question: Where should we allocate more computational resources and where should we achieve real-time responses?

8.1 How to use AstroLibrary?

The SpaceMap functions are available as APIs of AstroLibrary. Anyone with moderate programming skills can develop application programs using APIs. The spatiotemporal Voronoi engine is written in C++. There are three types of AstroLibrary (Table. 1). The Python and C++ versions can be installed in application programmers' computing resources, securely disconnected from internet if necessary. On the other hand, web API, also known as RESTful API, works as followings. When the API embedded in user's application program is executed, the computing server of SpaceMap runs to produce solutions. Table. 1 summarizes the three types of AstroLibrary and compares their features. AstroLibrary can be downloaded from GitHub (https://github.com/SPACEMAP42/AstroLibrary). The python package is set to be released on PIP (Package Installer for Python).

Table. 1. Three service types of AstroLibrary and their features.

	Library Type (A)	Programming Skill Needed (B)	Computing Platform(C)	Data Security (D)	Price (E)	Maintenance Easiness (F)
1	RESTful API	Low	SpaceMap	W/ Network	Low	High
2	Python Library	Low	Customer	W/O Network	High	Medium
3	C++ Library	High	Customer	W/O Network	High	Medium

9 Conclusions

In this paper, we presented the real-time CA/COLA features of SpaceMap which is designed for O(10^6) or more objects. The theoretical and computational basis is the Voronoi diagram which has been known as the most concise and efficient data structure for spatiotemporal reasoning among many objects in 2D and 3D spaces. The algorithms, implemented in C++, is available as AstroLibrary which has RESTful APIs and a Python package. With this library, efficient application programs can be easily and conveniently developed for challenging spatiotemporal problems by anyone with elementary programming skill. We expect that our algorithm will replace the seminal three-filter algorithm in future.

The three-filter algorithm is intuitive and easy to implement. A graduate student with moderate programming skill can implement it within a few months, if not weeks, to produce reasonably good conjunction reports. Due to this characteristic, companies tend to independently implement the three-filter algorithm with slight differences in acceleration methods, programming language environment, etc. We have confirmed this phenomenon with many companies including a recent report [20]. However, it easily consumes significant resources to make the implementation efficient and robust. Regarding CA/COLA, reinventions of the wheels seem common. Its consequence is noteworthy: It is costly, it takes long, but solution quality might be questionable.

10 Acknowledgements

This research was supported by the National Research Foundation of Korea (2017R1A3B1023591) and ITRC (Information Technology Research Center) support program (RS-2023-00259061) by IITP. This research was also partially supported by AOARD funding No. FA2386-22-1-4067.

11 References

1. Falle, A.*, et al.* One million (paper) satellites, *Science*, vol. 382, pp, 150-152, 2023.
2. ESA. "Space Environment Statistics," https://www.esa.int/, accessed 29 Oct 2023.
3. "Space Situational Awareness (SSA) and Space Traffic Management (STM) Workshop," https://www.youtube.com/, accessed 31 Oct 2023.
4. Space-Track. "Space-Track," https://www.space-track.org/, accessed 14 Jun 2023.
5. Alarcón Rodríguez, J.*, et al.*, Collision Risk Assessment with a Smart Sieve'Method, Joint ESA-NASA Space-Flight Safety Conference, 2002.
6. Budianto-Ho, I.*, et al.*, Scalable Conjunction Processing using Spatiotemporally Indexed Ephemeris Data, Advanced Maui Optical and Space Surveillance Technologies Conference, 2014.
7. Abernathy, B.*, et al.*, The CAOS-D architecture for conjunction analysis, Infotech@ Aerospace 2011, 2011.
8. "SPACE DEBRIS IDENTIFICATION AND TRACKING(SINTRA)," https://www.iarpa.gov/research-programs/sintra, accessed 29 Oct 2023.
9. "NASA Seeks Solutions to Detect, Track, Clean Up Small Space Debris," https://www.nasa.gov/coeci/, accessed Oct 29 2023.
10. Forbes India. "Digantara: Eye in the sky," https://www.forbesindia.com/, accessed 1 Nov 2023.
11. Nag, S.*, et al.* Prototyping operational autonomy for space traffic management, *Acta Astronautica*, vol. 180, pp, 489-506, 2021.
12. Hoots, F.R.*, et al.* An analytic method to determine future close approaches between satellites, *Celestial mechanics*, vol. 33, pp, 143-158, 1984.
13. Cha, J.*, et al.*, DVD-COOP: Innovative conjunction prediction using Voronoi-filter based on the dynamic Voronoi diagram of 3D spheres, Advanced Maui Optical and Space Surveillance Technologies Conference, 2017.
14. Choi, S.S.*, et al.*, SPACEMAP: Real-time Web Server for Safer, more Sustainable and Efficient Space, Advanced Maui Optical and Space Surveillance Technologies Conference, 2022.
15. Choi, S.S.*, et al.* Conjunction Assessments of the Satellites Transported by KSLV-II and Preparation of the Countermeasure for Possible Events in Timeline, *Journal of Space Technology and Applications*, vol. 3, pp, 118-143, 2023.
16. Song, C.*, et al.* Dynamic Voronoi Diagram for Moving Disks, *IEEE Trans Vis Comput Graph*, vol. 27, pp, 2923-2940, 2021.
17. "SOCRATES," https://celestrak.org/SOCRATES/, accessed 30 Oct 2023.
18. Alfano, S. Relating position uncertainty to maximum conjunction probability, *The Journal of the Astronautical Sciences*, vol. 53, pp, 193-205, 2005.
19. Chen, L.*, et al. Orbital Data Applications for Space Objects*: Springer, 2017.
20. Singh, S. A new tool for conjunction analysis of ISRO's operational satellites, Close Approach Prediction Software: CLAPS, *Journal of Space Safety Engineering*, vol. 7, pp, 290-294, 2020.