
**AFRL-AFOSR-JP-TR-2020-0006**

# Innovative Conjunction Prediction in Space Situational Awareness (SSA) using Voronoi Diagram with 3D Spheres

**Deok-Soo Kim**
**HANYANG INDUSTRY-UNIVERSITY COOPERATION**
**HAENGDANG 1DONG, SEONGDONG-GU, SEOULRM110. HIT, HANYANG UNIV. 17**
**SEOUL, 133-791**
**KR**

**07/11/2020**
**Final Report**



Air Force Research Laboratory
Air Force Office of Scientific Research
Asian Office of Aerospace Research and Development
Unit 45002, APO AP 96338-5002

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From - To)* |
|---|---|---|
| 11-07-2020 | Final | 30 Sep 2017 to 29 Sep 2019 |

**4. TITLE AND SUBTITLE**
Innovative Conjunction Prediction in Space Situational Awareness (SSA) using Voronoi Diagram with 3D Spheres

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**
FA2386-17-1-4050

**5c. PROGRAM ELEMENT NUMBER**
61102F

**6. AUTHOR(S)**
Deok-Soo Kim

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
HANYANG INDUSTRY-UNIVERSITY COOPERATION
HAENGDANG 1DONG, SEONGDONG-GU, SEOULRM110. HIT, HANYANG UNIV. 17
SEOUL, 133-791 KR

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
AOARD
UNIT 45002
APO AP 96338-5002

**10. SPONSOR/MONITOR'S ACRONYM(S)**
AFRL/AFOSR IOA

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**
AFRL-AFOSR-JP-TR-2020-0006

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
A DISTRIBUTION UNLIMITED: PB Public Release

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**
Collision avoidance is one of the critical tasks to keep geospace safe and efficient. This research studied the detection and resolution of conjunction prediction for collision avoidance to develop and implement an algorithm to based on the dynamic Voronoi diagram (DVD) of 3D spherical balls. A DVD algorithm for spherical balls was successfully developed, implemented, and tested. Then the team developed and implemented Conjunctive Orbital Objects Predictor and Planner (COOP2) algorithm/program based on the DVD algorithm. This demonstrated a proven mathematical capability to detect all conjunctions without any missing cases. The COOP2 was tested against TLE data obtained from Korea Aerospace Research Institute (KARI), which includes the orbital movements of the following five Korean satellites: KOMPSAT-2, KOMPSAT-3, KOMPSAT-3A, and KOMPSAT-5. Given a detected conjunction, COOP2 can produce the optimal maneuver plan by quickly evaluating alternatives using the event history stored in the COOP-HSTRY file. Another demonstration of the correctness and performance of COOP2 was conducted using drone swarming. COOP2 generated the collision-free flight path of each drone in order that the performance and correctness of the algorithm/program could be validated and verified. A demonstration of the drone swarming control can be seen at https://www.youtube.com/watch?v=lkqo3AXguKw&t=3s.

**15. SUBJECT TERMS**
SSA, Conjunction, Prediction, Voronoi

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | CHEN, JERMONT |
| Unclassified | Unclassified | Unclassified | SAR | | 19b. TELEPHONE NUMBER *(Include area code)* 315-227-7007 |

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

**Final Report on AOARD (Award No: FA2386-17-1-4050)**

# Innovative Conjunction Prediction in Space Situational Awareness (SSA) using Voronoi Diagram with 3D Spheres

February 22, 2020

# Contents

## 1. Project Overview

### 1) Title: Innovative Conjunction Prediction in Space Situational Awareness using the Voronoi Diagram of 3D Spheres

**2) Name of Principal Investigators:** Deok-Soo Kim

- e-mail address: dskim@hanyang.ac.kr
- Institution: Hanyang University, Seoul, Republic of Korea
- Mailing Address: School of Mechanical Engineering 1102, IT/BT bldg., Hanyang University 222 Wangsimni-ro, Seoul
- Phone: +82-2-2220-0472
- Fax: N/A

**3) Period of Performance**: September/30/2017 – September /29/2019

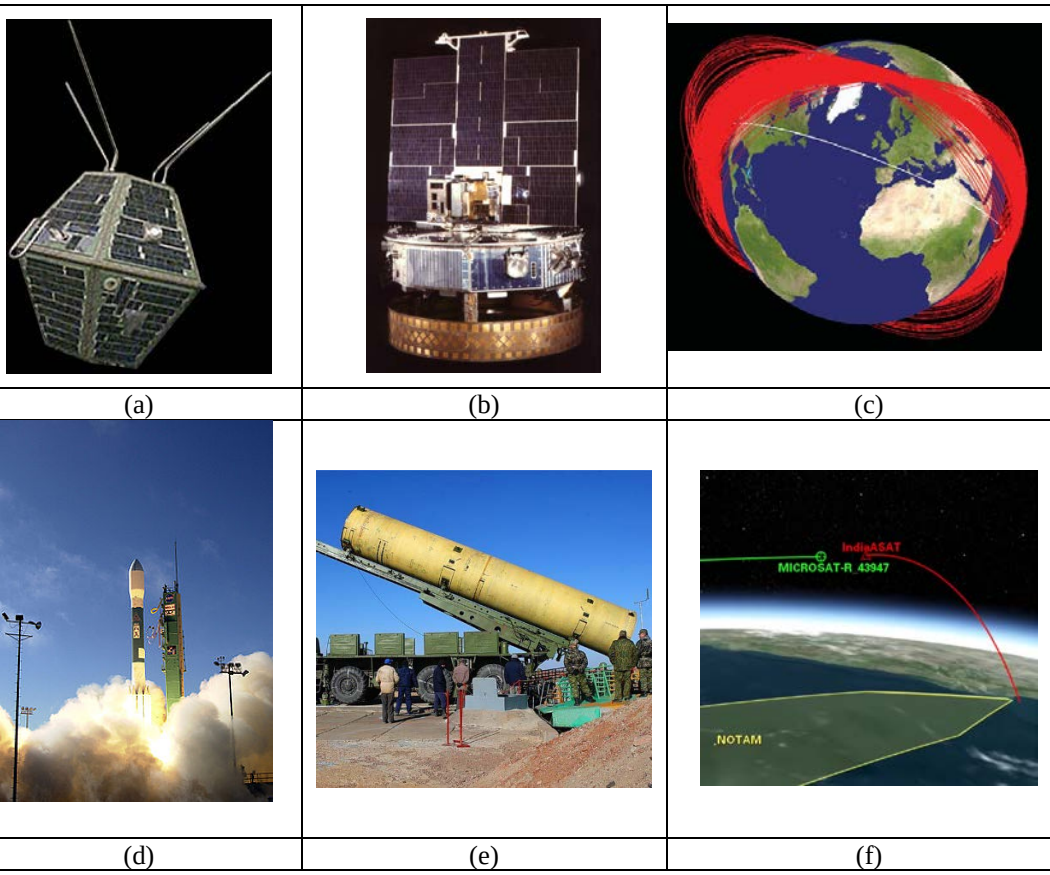
### 4) Abstract

Collision avoidance is one of the critical tasks to keep geospace safe and efficient. This research studies the detection and resolution of conjunction for collision avoidance and develops an algorithm to predict the conjunction based on the dynamic Voronoi diagram (DVD) of 3D spherical balls. We have successfully developed, implemented, and tested the DVD algorithm for 3D spheres. Then we have developed and implemented the **COOP2 (Conjunctive Orbital Objects Predictor and Planner)** algorithm/program, based on the DVD algorithm. COOP2 shows a proven mathematical capability to detect all conjunctions without any missing case. We tested COOP2 using the TLE data obtained from Korea Aerospace Research Institute (KARI) which includes the orbital motions of the following five Korean satellites: KOMPSAT-2, KOMPSAT-3, KOMPSAT-3A, and KOMPSAT-5. We developed an advanced capability of COOP2 which, given a detected conjunction, produces the optimal maneuver plan by quickly evaluating alternatives using the event history stored in the COOP-HSTRY file. We have validated the correctness and performance of COOP2 algorithm and program using **drone swarming**. Suppose that we fly a swarm of drones where each drone follows its own path in the 3D space. In this scenario, just like Resident Space Objects (RSO), we were able to use the COOP2 program to generate the **collision-free flight path of all drones** so that the performance and correctness of the algorithm/program could be validated and verified. (For the demonstration of drone swarming, refer to Fig. 37 and **YouTube link: https://www.youtube.com/watch?v=Ikqo3AXguKw&t=3s**.)

## 2. Introduction

There are many man-made Resident Space Objects (RSOs) that orbits Earth. As of January 2019, more than 5,400 rocket launches (excluding failures) since 1957 which placed about 8,950 satellites into Earth orbit: about 5,000 satellites remained in orbit including about 1,950 operational ones. European Space Agency (ESA) estimated approximately 34,000 debris of size bigger than 10 cm, about 900,000 debris of size 1~10 cm, 128 million ones of size 1mm ~ 1 cm [1]. This number will increase rapidly due to accidental **satellite collisions** (e.g. 2009 collision between Iridium 33 and Kosmos 2251; Produced >2,000 catalogued debris), planned **anti-satellite missile test**, **new satellite launches**, etc.

Among these, the anti-satellite (ASAT) missile test would be a main cause of creating debris and we compile some historical events related to ASAT in Fig. 1(a-f). The world's first anti-satellite missile test was carried out against a special target spacecraft, the DS-P1-M, in February 1970 by Soviet Union. The US' first successful anti-satellite missile test against a gamma ray spectroscopy satellite Solwind P78-1 was completed on 13 September 1985. The test resulted in 285 cataloged pieces of orbital debris. On 11 January 2007, the China destroyed a defunct Chinese weather satellite, Fungyun (FY-1C) [7]. This event was the **largest recorded creation of space debris in history with more than 2,000 pieces of trackable size** (golf ball size or larger) officially catalogued in the immediate aftermath, and an estimated 150,000 debris particles [8][9]. Ever since several anti-satellite tests were performed: e.g. on 21 February 2008, the U.S. Navy's destruction of the malfunctioning U.S. spy satellite USA-193 [10], on November 18 2015, the flight test of Russia's direct ascent anti-satellite missile [11], and on 27 March 2019, India's test.




(a) (b) (c)

(d) (e) (f)

Fig. 1. The examples for anti-satellite tests:

(a) The target satellite (DS-P1-M) of Soviet Union for the world's first successful intercept of satellite completed in February 1970 by Soviet Union [2]: Many tests carried out against a special target spacecraft, DS-P1-M, after the first and second test on 27 October 1967 and 28 April 1968, respectively. The first successful test (the second overall) achieved 32 hits (each could penetrate 100 mm of armour) [3].

(b) The target satellite (Solwind P78-1) of US for the U.S.'s first successful anti-satellite missile test on 13 September 1985: Solwind P78-1 was a gamma ray spectroscopy satellite launched in 1979 and orbiting at 525 km [4]. Its main purpose had been the study of the solar wind, among other things. An aircraft F-15 that carried a missile took off from Edwards Air Force Base, climbed to 11613 m (38100 ft) and vertically launched the missile at the Solwind P78-1. The test resulted in 285 cataloged pieces of orbital debris. 1 piece of debris remained in orbit to at least May 2004 [5] but had deorbited by 2008 [6].

(c) The known orbit planes of the debris that produced by the destruction of a defunct Chinese weather satellite, FY-1C on 11 January 2007 [7] (orbits exaggerated for visibility, source: https://en.wikipedia.org/wiki/2007_Chinese_anti-satellite_missile_test): The destruction was reportedly carried out by an SC-19 ASAT(Anti-satellite weapons) missile with a kinetic kill warhead similar in concept to the American Exoatmospheric Kill Vehicle. FY-1C was a weather satellite orbiting Earth in polar orbit at an altitude of about 865 km (537 mi). This event was the largest recorded creation of space debris in history with more than 2,000 pieces of trackable size (golf ball size and larger) officially catalogued in the immediate aftermath, and an estimated 150,000 debris particles [8][9].

(d) The U.S. spy satellite (USA-193) launched on 14 December 2006 and destroyed on 21 February 2008 by the U.S. Navy using a ship-fired RIM-161 Standard Missile 3. due to its malfunctioning (source: https://en.wikipedia.org/wiki/USA-193): The destruction of USA-193 created 174 pieces of orbital debris that were cataloged by the U.S. military [10]. While most of this debris re-entered the Earth's atmosphere within a few months, a few pieces lasted slightly longer due to the fact that they were thrown into higher orbits. The final piece of USA-193 debris did not re-enter until 28 October 2009 [10].

(e) Russia's anti-satellite missile (PL-19 Nudol) (source: https://twitter.com/mt_strat/status/811697894877175808?lang=ca): The successful flight test of Russia's direct ascent anti-satellite missile, known as PL-19 Nudol, took place on 18 November 2015, according to defence officials familiar with reports of the test [11].

(f) Analysis and debris simulation of India's direct ascent anti-satellite weapons test conducted on 27 March 2019 called Mission Shakti [12] (source: https://www.space.com/india-anti-satellite-missile-test-agi-simulations.html): The interceptor was able to strike a test satellite at a 300-kilometre (186 mi) altitude in low earth orbit (LEO), thus successfully testing its ASAT missile. The interceptor was launched at around 05:40 UTC at the Integrated Test Range (ITR) in Chandipur, Odisha and hit its target Microsat-R after 168 seconds. The impact generated more than 400 pieces of orbital debris with 24 having apogee higher than ISS orbit [13][14].

Attention should also be provided for the increased number of deployed small satellites in geospace. In recent years, the **trend of deployment of spacecrafts** is more, **smaller, and lower cost civilian ones** compared to the few, large, and expensive government ones [15]. For example, SpaceX has permission to launch 12,000 satellites to low Earth orbit for its Starlink project which is a satellite constellation consisting of thousands of **massively produced small satellites** (Fig. 2).

These increasing space objects altogether, particularly in Low Earth Orbit (LEO), significantly accelerate collision risks among space objects [16] as also implied by the Kessler syndrome (also called collisional cascading) [17]. We have very recently observed a big event that could lead to a catastrophe in LEO: a near-miss of the two defunct satellites on 29 January 2020, 23:39:35 UTC, IRAS (Infrared Astronomical Satellite), launched in 1983, and GGSE-4 (Gravity Gradient Stabilization Experiment 4), launched in 1967 (Fig. 3) [18].

As RSOs move at high speed of up to 16 km/s if head-on, the impact of collision between space objects can be catastrophic. To better prevent unexpected conjunctions and collisions among objects and to preserve the geospace, particularly LEO, for future, it is necessary to have a method to predict and prevent collisions and eventually develop **space traffic management (STM) system**. STM will be increasingly important as the geospace is more commercialized such as the expected popularity of suborbital space tourism and/or commercial personal spaceflight [19, 20]. Since **the prediction and prevention of collision/conjunction among RSOs is one of the most critical issues in STM** [21, 22], the detection, tracking, identification, cataloging, etc. of all observable RSOs in the orbit, altogether referred to as space situational awareness (SSA), is necessary. Securing a perfect SSA is both costly and complicated. Joint Space Operations Center (JSpOC, https://www.space-track.org/) maintained by the United States Strategic Command's (USSTRATCOM) is a good resource [23].

The value of conjunction prediction is obvious in that, **if a conjunction is correctly predicted, a collision-avoiding evasive maneuver of an RSO can be planned and executed**. When it is possible, an optimal maneuver pathway might need to be determined or designed by possibly evaluating the effect of each hypothesized maneuver to future conjunctions. The formulation of this optimization problem involves parameters that can be obtained from multiple executions of conjunction prediction, each time with modified ephemeris. This implies a very high frequency of executing conjunction prediction than it is done today. E.g. Collision Risk Assessment tool (CRASS), developed by GMV/ESA, forecasts conjunctions on daily basis with a prediction time window of one week, a policy made in the consideration of a trade-off between orbit prediction accuracy and reaction time to a predicted conjunction [24, 25].

Here we report the development and implementation of an innovative conjunction prediction and maneuver planning algorithms using the (dynamic) Voronoi diagram of 3D spherical balls. The developed **COOP2**[1] **(Conjunctive Orbital Objects Predictor and Planner) algorithm/program** can **predict conjunctions** and **find the optimal maneuver pathway to avoid a predicted conjunction situation** among the RSOs in JSpOC Space Catalogue. The COOP2 algorithm/program is event-based, general purpose (beyond pairwise conjunction prediction), efficient, accurate, and independent of the coordinate system. The algorithm is based on the dynamic Voronoi diagram of moving three-dimensional spheres. Its computation result is efficiently re-playable for diverse analyses on the fly.

[1] We called COOP2 as DVD-COOP (Dynamic Voronoi Diagram-based Conjunctive Orbital Objects Predictor) in our preliminary studies.

Fig. 2. Starlink project (source: https://en.wikipedia.org/wiki/SpaceX_Starlink): Starlink is a satellite constellation being constructed by SpaceX. The constellation will consist of thousands of massively produced small satellites.

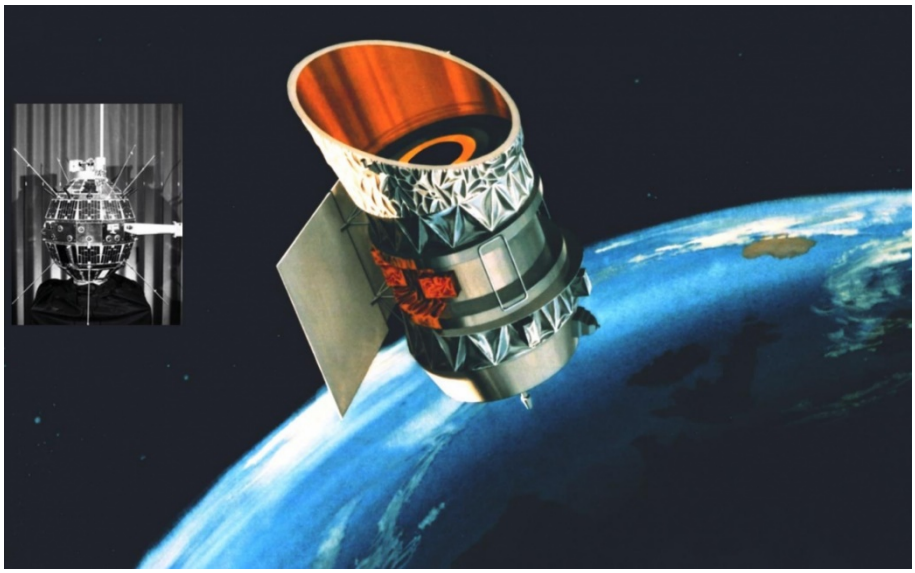


Fig. 3. The near-miss of the defucnt GGSE-4 and IRSA satellites: (source: https://www.nasaspaceflight.com/2020/01/close-call-two-satellites-avoid-collision/). On 29 January 2020, 23:39:35 UTC, two satellites, IRAS (Infrared Astronomical Satellite), launched in 1983, and GGSE-4 (Gravity Gradient Stabilization Experiment 4), launched in 1967, was expected to pass as closely as 12 meters with an estimated risk of collision of 5%. Fortunately, it turns out that there were no new tracked debris following the incident.

## 3. Project Objectives

### 3.1. Literature Review

A naïve approach to conjunction prediction might be as follows. Suppose that Space Catalogue has $N$ objects (or equivalently $N$ orbits) and thus $O(N^2)$ object pairs. It is easy to make linear approximations of two orbits corresponding to some discrete moments in time. Consider a line segment $L$ with an approximation error $\varepsilon$. Let $\hat{O}$ be the replica of an object $O$, for notational convenience, with a null size: $\hat{O}$ moves through $L$ while $O$ moves through its elliptic orbit. Let $S$ be a sphere, called a **replica sphere**, with the radius $\varepsilon$. It can be easily shown that, at an arbitrary moment of time, if two objects are conjunctive, the replica spheres corresponding to the two objects are also conjunctive. Hence, a real conjunction can be detected by checking the intersection between two replica spheres enlarged to appropriate sizes according to the linear approximation errors of paths. Without loss of generality, it can be assumed that each full elliptic orbit is approximated by $M_L$ line segments. This leads to $O(M_L N^2)$ pairs of line segments to be tested for the interference between two enlarged moving replica spheres.

Suppose that some point locations are sampled from the two line segments, say $L_1$ and $L_2$, corresponding to replica spheres $S_1$ and $S_2$, respectively, and the replica spheres are placed at these sampled locations to check if they intersect each other or not. Hence, it takes $O(M_R)$ intersection tests for a sphere pair for $M_R$ sampling points on each line segment (assuming that moving speed is properly considered). Hence, it takes **$O(M_L M_R N^2)$ time for a single complete elliptic orbit for each of all the space objects**. Considering the size of satellite compared to the lengths of $L_1$ and $L_2$ (i.e. the size vs. traveling distance of satellites), $M_R$ should be very big. The main idea of this naïve method can be said to **enumerate the space as effectively as possible with sampled points in the absence of efficiency consideration**.

The naïve approach is to check all pairwise objects for all possible moments – This is infeasible due to computational requirement. A trade-off exists between sampling frequency and computation cost: A high computation cost is required in order to reduce the chance of missing conjunction (as shown in Fig. 4(b)). Be aware that this approach cannot eliminate the possibility of missing intersection regardless how high a sampling frequency is.

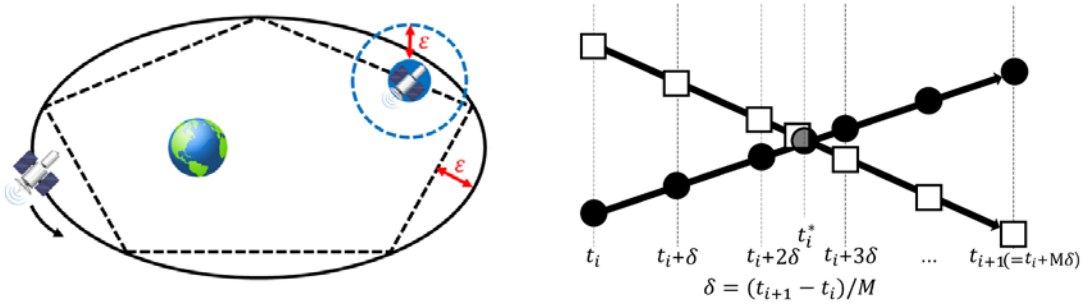



Fig. 4. Naïve approach for conjunction prediction. (a) Piecewise linear approximation of an orbit with five segments. (b) Enumerating locations on the linear approximation with a finite points seperated by $\delta$ in time. The collision at $t_i^*$ is not detected by the suggested sampling.

To reduce the computational requirement of the naïve approach, many studies were conducted. Most existing studies were about the **reduction of search space by filtering out orbit combinations that were guaranteed to be free of any conjunction** [26-35] which were the improvements of the initial idea of the three-filter approach proposed by Hoots et al. (1984) [36]: Apogee/perigee filter (Filter I), orbit path filter (Filter II), and time filter (Filter III). Note that the first two filters (I and II) are time-independent and geometric. Casanova et al. (2014) reported the following [23]: Given 372,816 (=864*863/2) object pairs from 864 objects, Filter I reduces the number of pairs down to 218,459 and Filter II further reduces to 49,739 pairs, resulting into a total of 86.7% reduction of object pairs in the explicit intersection test. Filter III needs to be applied to the rest, 13.3%.

A natural effort to speed-up the computation efficiency would be to use **a spatial hash** which was devised for a more direct access to objects thus hopefully reducing the search space of the naïve approach. The kd-trees were introduced for conjunction prediction. Let $\delta$ be the time step of each prediction time slice. Then the idea of using a **kd-tree** in conjunction prediction is **to extend that of spatial hash** in that the trajectory of each orbit during a time step $\delta$ in the prediction time window is a basic unit of information. For each of *X*-, *Y*-, and *Z*-axis motions, both the minimum and maximum of travel interval is stored in a six-dimensional kd-tree. **Two objects whose trajectory during $\delta$ do not overlap is safely filtered out from further study of conjunction**. Fig. 5 shows the example of checking whether the trajectories of two objects overlap. The red, blue, and green curves represent the trajectory of three objects, $x_A$, $x_B$, and $x_C$, respectively. Note that the x-axis of the graph is temporal axis while y-axis is spatial axis. Therefore, the graphs given in Fig. 5 represents the one-dimensional movements by time. There is overlap between coordinate range of $x_A$ and $x_B$. Therefore, object pair $\{x_A, x_B\}$ is considered as possible conjunction pair and passed for further investigation. Meanwhile, object pair $\{x_A, x_C\}$ is considered as conjunction free pair because there is no overlap.

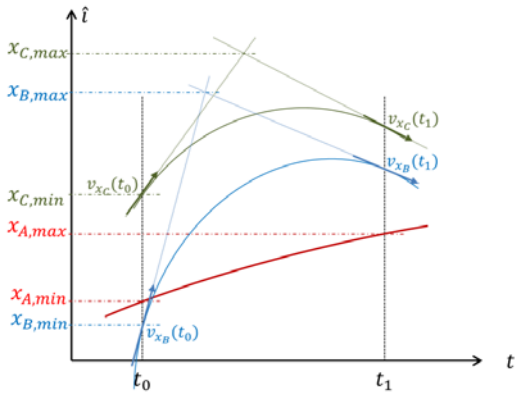



Fig. 5. The example of checking whether the trajectories of two objects overlap. The trajectories of $x_A$, $x_B$, and $x_C$ are represented as red, blue, and green lines [23].

As far as we are aware of, the initial use of a kd-tree in conjunction prediction was the Moving Object Processing System (MOPS) system of University of Hawaii's Pan-STARRS began in 2002 [37-40]. Kd-trees were used to efficiently answer a variety of spatial queries in conjunction prediction [41]. CAOS-D system also employed the kd-tree for conjunction prediction [23]. They suggested to define a three-dimensional grid system, say G, which should include all orbits and contain a sufficient number of buckets which were represented as a three-dimensional array. During a time step $\delta$ of each prediction time slice, each object orbits through an elliptic arc, say $\xi$, whose positions and tangent vectors corresponding to the beginning and ending of $\delta$ provide a planar triangle T on the plane of travel in the three-dimensional space. We observe that this triangle is identical to the control polygon of the rational quadratic Bezier curve of the elliptic arc. Hence, **for each orbit arc (and thus each object), the elements of the grid system G which intersect the triangle T can be marked**. Hence, the grid element marked by both objects only needs to be further studied for the possible conjunction. [23] reported an experimental result: An object in Low Earth Orbit (LEO) with a 60 s time step required a bounding box with dimensions of over 400 km whereas a grid cell dimension was 50 km by 100 km. Spatial hash requires to fine tune several parameters.

Kd-trees recursively subdivide in principle both the set of data points and the corresponding Euclidean space into progressively finer subsets and sub-regions. A node of the tree corresponds to both a subset of the space and a subset of the input data points. Given **a height-balanced kd-tree constructed in *O*(*N* log *N*) time and stored in *O*(*N*) memory**, an insertion or a nearest neighbor search could be done in *O*(log *N*) time on average. Many other queries can also be done very efficiently.

Summary: The previous approaches have the following properties:

- Tailored for **pairwise conjunctions**
- Coordinate system dependency
- **Difficulty to replay**
- **Time increment-based**

The previous approaches seem **not easy to replay computation results in order to answer other spatial queries** than that was done during the computation. One of the reasons is each kd-tree (or spatial hash) corresponding to each time step $\delta$ requires O($N$) memory. It seems that the kd-tree and spatial hash methods can find all conjunctions if correctly implemented while the naïve method may miss a conjunction. We emphasize that **there must be other types of queries, beyond pairwise conjunction, in diverse applications** which are both theoretically more involved and computationally more demanding: E.g. **triplet-wise and quadruplet-wise conjunctions**. Another useful capability might be a quick and efficient replay of a prediction result with different analyses. To deal the conjunction between object pairs, the kd-tree approach is fast enough. However, this approach **could not be applied to the conjunction between object triplets and quadruplet case**. This is the reason why we propose the Voronoi diagram in this project.

## 3.2. Research Goals and Summary on Performance

**(Research Goals)** This research studies the detection and resolution of conjunction for collision avoidance and develops an algorithm to predict the conjunction based on the dynamic Voronoi diagram of 3D spherical balls. The research goals of this project are as follows:

- Development of the dynamic Voronoi diagram (DVD) algorithm for 3D spherical balls,
- Implementation and test of the DVD algorithm,
- Development of the conjunction prediction algorithm,
- Implementation of the conjunction prediction algorithm, and
- Validation of the conjunction prediction algorithm using drones.

**(Summary on Performances)** We have successfully developed, implemented, and tested the DVD algorithm for 3D spherical balls. Then we developed and implemented the **COOP2 (Conjunctive Orbital Objects Predictor and Planner) algorithm/program**, based on the DVD algorithm for 3D spherical balls, which showed a proven mathematical capability to **detect all conjunctions without any missing case**. We reported the prior results in the AMOS2017 conference. We tested the COOP2 using small TLE data obtained from Korea Aerospace Research Institute (KARI) which includes the orbital movements of the following five Korean satellites: KOMPSAT-2, KOMPSAT-3, KOMPSAT-3A, and KOMPSAT-5. We developed a new capability of COOP2 which, **given a detected conjunction, produces the optimal maneuver plan by quickly evaluating alternatives using the event history stored in the COOP-HSTRY file**. We reported this result in the AMOS2018 conference and filed patent applications with the ideas in COOP2. We implemented and tested the developed COOP2 algorithm as a web server (http://voronoi.hanyang.ac.kr/COOP2)

We have validated the correctness and performance of COOP2 algorithm and program using **drone swarming**. Suppose that we fly a swarm of drones where each drone follows its own path in the 3d space. In this scenario, just like Resident Space Objects (RSOs), we used the COOP2 program to generate the **collision-free flight path of each drone** so that the performance and correctness of the algorithm/program could be validated and verified. As our access to the motion change of RSOs is almost impossible, this validation using drones was necessary (For drone swarming demonstration, **Refer to Fig. 37 and YouTube link: https://www.youtube.com/watch?v=Ikqo3AXguKw&t=3s**). We published the ideas in COOP2 using the DVD algorithm and its application to drone swarming in a top-notch journal, ***IEEE Transactions on Visualization and Computer Graphics*** (Refer to the attachment "A-1.PDF").

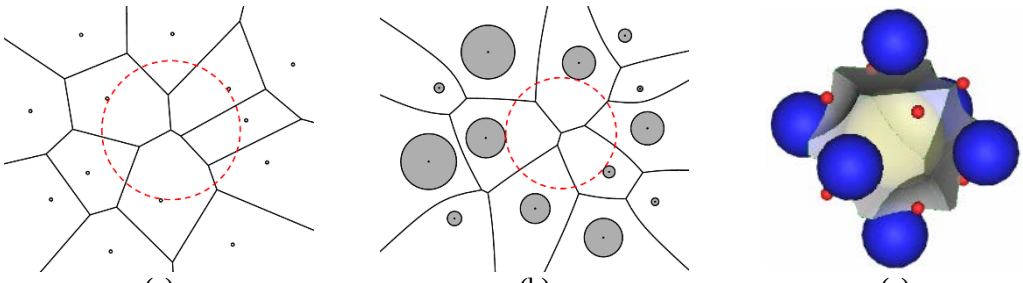
## 4. Brief Review on Voronoi Diagrams

### 4.1. Concept of Voronoi diagrams

Given a set $P = \{p_1, p_2, ..., p_N\}$ where $p_i$ is a point generator in a $d$-dimensional space, the **ordinary Voronoi diagram** $Vor(P)$ of $P$ is **a tessellation of the space where each cell of the tessellation is the set of locations where each location is closer to the generator of the cell than to the other generators** [42, 43]. Usually the Euclidean distance is employed. Fig. 6(a) shows an example of such an ordinary Voronoi diagram in the plane. The boundary between two adjacent point generators is the perpendicular bisector of the two generators. Given a Voronoi diagram, many spatial queries can be efficiently and correctly (or at least most accurately) answered. The red dotted circle is the biggest empty circle that can be placed in the middle of the point generators where the empty circle does not contain any input generator. This query can be correctly answered in $O(N)$ time if $Vor(P)$ is available. The ordinary Voronoi diagram has long been used for efficiently solving spatial reasoning problems in a variety of disciplines. $Vor(P)$ can be constructed in $O(N^2 \log N)$ time for $d = 3$ ($O(N \log N)$ for $d = 2$) in the worst case. Note that the worst case combinatorial complexity of $Vor(P)$ is $O(N^2)$. The average time complexity of $Vor(P)$ is $O(N)$ for $d$ both 2 and 3. However, its robust construction is in practice of critical concern. $Vor(P)$ can be robustly constructed by the topology-oriented incremental (TOI) algorithm in $O(N^3)$ time in the worst case but $O(N)$ time on average.

Let $B=\{b_1, b_2, \ldots, b_N\}$ where $b_i$ is a spherical ball in the $d$-dimensional space where $b_i \cap b_j = \phi$, $i \neq j$. The **Voronoi diagram** $VD(B)$ of $B$ can be similarly defined as **a tessellation of the space where each cell of the tessellation is the set of the locations where each location is closer to the boundary of the corresponding ball than to the other balls**. In VD where $d = 3$, we refer the set of locations equidistant to a pair of balls as a Voronoi face, abbreviated as a V-face, that to three balls a V-edge, and that to four balls a V-vertex. It is relatively recent that important properties of this type of Voronoi diagram for $d = 2$ and 3 were discovered and the construction algorithms were devised [44-54]. Fig. 6(b) shows an example in the plane with the biggest empty circle found in O($N$) time. There are many important applications that can be efficiently solved using this type of Voronoi diagram. Fig. 6(c) shows the Voronoi diagram of spheres where $d = 3$.

Numerical robustness is one of the most fundamental challenge of the Voronoi diagram of circles and spheres yet to be studied. No robust Voronoi diagram construction algorithm has been known for 3-dimensional spheres while its counterpart in the plane was recently reported with an implementation [54]. We have recently succeeded to extend our TOI-algorithm of the plane to the three-dimensional space.



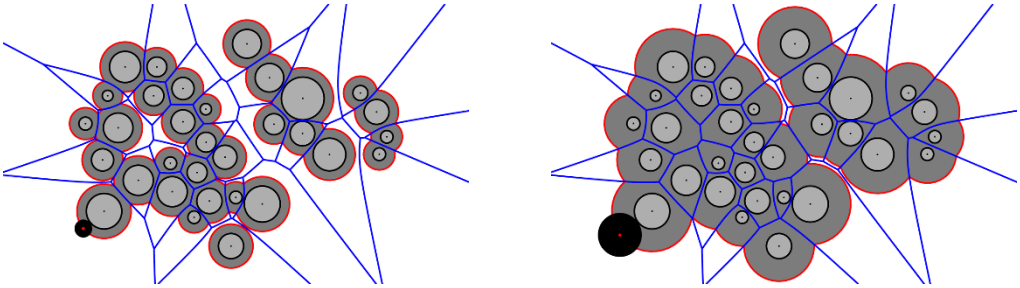
(a) (b) (c)

Fig. 6. Voronoi diagrams (a) Voronoi diagram of points (2D). (b) Voronoi diagram of circular disks (2D). (c) Voronoi diagram of spherical balls (3D)

### 4.2. Useful Properties of Voronoi diagrams

Fig. 7 shows an application of offset of the Voronoi diagram of disks in the plane to demonstrate its capability: It shows how the Voronoi diagram can be used for solving difficult application problems correctly, efficiently, and conveniently. Suppose that a set of $N$ input disks is given (Fig. 7, the light grey circles). The locus of the center of a circular black probe touching the input disks defines the offset of constant distance (the red piecewise circular arc curve in Fig. 7(a)). Its computation takes $O(N^2)$ time if we do a naïve check between the offset circles of each input disk pair and the algorithm is complicated to get correct solution. However, **the intersection between two offset arcs occurs on the Voronoi edge of the Voronoi diagram of the input disks**. Hence, if the Voronoi diagram is available, **the correct offset can be computed in $O(N)$ time simply by checking each V-edge in the Voronoi diagram**. The algorithm is simple and fast. In addition, the existence of void structure within the left disk cluster and the tunnel (channel) structure between two clusters can also be easily recognized.

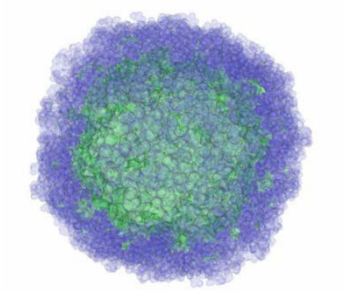
- Voronoi diagrams are useful for accurately, efficiently, and conveniently solving spatial reasoning problems among particles.



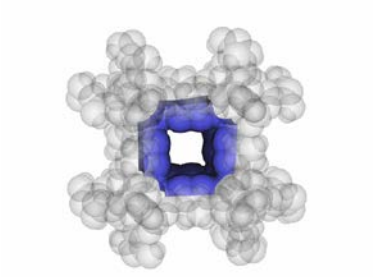
(a) (b)

Fig. 7. $O(N)$ time computation of offset curves with the Voronoi diagram. An offset curve corresponds to the center of circular probe (black). (a) Small probe. A void and two tunnels are recognized. (b) Bigger probe. Both void and tunnels disappeared.

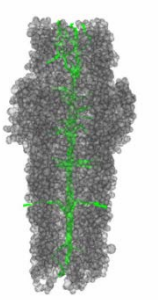
Fig. 7(b) shows that the same Voronoi diagram can be used for a bigger probe: The fact that neither void nor tunnel exists can be also recognized. Similar observation holds for its three-dimensional counterpart for the arrangement of spherical atoms and various application problems were solved, particularly for molecular problems [55-62] as shown in Fig. 8 (a-d). The following summary is well-known.



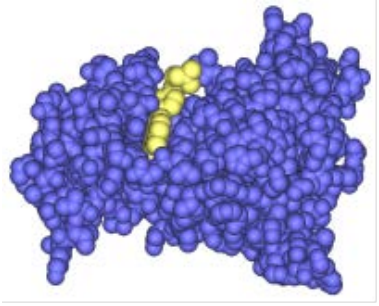
(a) Void structures of ferritin (32,952 atoms) (1MFR, warehouse for Fe atoms)



(b) Pore structure (tunnel) in metal organic framework



(c) Potassium channel (9,915 atoms) (2VDD, K+ ion passage)



(d) Protein-ligand docking: Tyrosine Kinase vs. Gleevec

Fig. 8. Applications for Voronoi diagram of 3d spherical balls

### 4.3. Dynamic Voronoi diagram (DVD)

The useful properties of Voronoi diagrams can be extended to the case where some particles move. In this section, we discuss how we can obtain the dynamic Voronoi diagram of moving particles. In summary, the useful properties extend to the dynamic case if the **dynamic Voronoi diagram** can be efficiently constructed.

**(An example: 2D)** Fig. 9 shows the Voronoi diagrams of disk generators where one disk moves to the right according to its velocity vector indicated by the black arrow. Each of the eight Voronoi diagram instances is associated with its time stamp $t_0$, $t_1$,…, $t_7$. The algorithm of the dynamic Voronoi diagram (DVD) algorithm detects the moments where critical events occur to maintain a correct Voronoi diagram. The DVD algorithm begins with the initial Voronoi diagram $VD_0$ at $t_0$. The topology of $VD_1$ at $t_1$ is identical to that of $VD_0$ whereas the topology of $VD_3$ is different from that of $VD_0$. So, we compute the moment $t_2$ when the topology changes. Note that this moment corresponds to the moment that a V-edge of $VD_0$ shrinks to a point and at the same time the shrunk point begins to grow to another V-edge at $t_3$. Hence, $t_2$ is the moment of edge-flipping in that a V-edge of $VD_0$ flips in the topology structure. In the example, there are two more edge-flipping moments as indicated by the big blue ellipses. It is known that the edge-flipping moment can be computed by solving an 8-th degree polynomial for four disks of arbitrary sizes which linearly move with arbitrary speed in the plane. In three-dimensional space, the flipping moment from a V-edge to a V-face or vice versa can be computed by solving a 10-th degree polynomial [63, 64]. Correctly solving the root-find problem of these polynomials is not trivial at all.

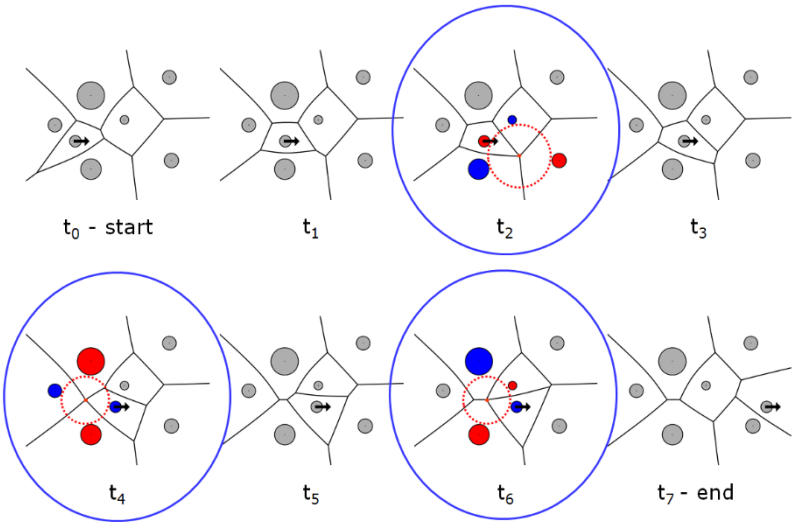
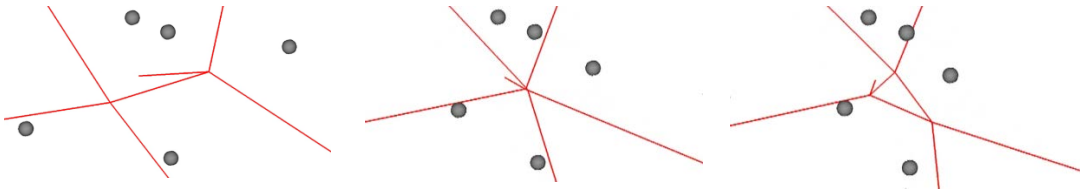



Fig. 9. Dynamic Voronoi diagram where one disk linearly moves. ($t_0$) Initial state. ($t_1$) before the first V-edge flips, ($t_2$) the moment when a V-edge flips (i.e. a V-edge shrinks to a point). ($t_3$) after the V-edge flips. ($t_4$) another V-edge flips. Similar observations follow in ($t_5$) ($t_6$) and ($t_7$).

**(Extension to 3D)** Different from 2D examples in Fig. 9, in 3-dimensional space, there are two types of flip event: edge-flip (from a V-edge to a V-face) and face-flip (vice versa) as shown in Fig 10. Note that the five spheres are involved in flip event of 3-dimensional space while four disks are concerned in 2-diensional space. This makes the flipping moment from a V-edge to a V-face or vice versa be computed by solving a 10-th degree polynomial [65].



(a) (b) (c)

Fig 10. Edge-flip/face-flip in a Voronoi diagram of spheres: (a)-(c)/(c)-(a), respectively. (a) Before an edge-flip. (b) The edge contracts to a point. (c) After an edge-flip

**(Flipping time in 3D)** Let $S = \{s_1(c_1, r_1), s_2(c_2, r_2), s_3(c_3, r_3), s_4(c_4, r_4), s_5(c_5, r_5)\}$ be a set of five spheres where a sphere $s_i$ has the center $c_i(x_i, y_i, z_i)$ and radius $r_i \geq 0$, $i = 1, 2, 3, 4$, and 5. Suppose that $s_5$ is the smallest. The flip time of $S$ is computed by solving the following polynomial (1) which are about the condition for a set of five spheres to be cotangent to a common circumsphere.

$$A_1^2 + A_2^2 + A_3^2 - A_4^2 = 0 \qquad (1)$$

$$\text{where } A = \begin{vmatrix} x_1 - x_5 & y_1 - y_5 & z_1 - z_5 & r_1 - r_5 \\ x_2 - x_5 & y_2 - y_5 & z_2 - z_5 & r_2 - r_5 \\ x_3 - x_5 & y_3 - y_5 & z_3 - z_5 & r_3 - r_5 \\ x_4 - x_5 & y_4 - y_5 & z_4 - z_5 & r_4 - r_5 \end{vmatrix}$$

and $A_i$ is obtained by replacing the i-th column of $A$ by the column $[p_j]_{j=1}^4$,

where $p_j = (x_j - x_5)^2 + (y_j - y_5)^2 + (z_j - z_5)^2 - (r_j - r_5)^2, j = 1, 2, 3, 4$ and the sphere motions are analytic functions of time t. (See details in [65].)

If generators (points/disks/spherical balls) move linearly at constant speeds, the polynomial of Eq. (1) for d-dimensional spherical balls is of degree 2(d + 2). Hence, the degree of polynomial is 8 for two-dimensional disks and 10 for three-dimensional spherical balls. If generators are points in a plane, the degree is 4. If generators are disks and the power distance is used, the degree is also 4. Table 1 summarizes this observation.

Table 1. The degree of the polynomial in Eq. (1) for the dynamic Voronoi diagram [65] (Generators move linearly at constant speeds).

| | 2-dimension | 3-dimension | d-dimension |
|---|---|---|---|
| Ordinary VD of points | 4 | 5 | d+2 |
| VD of disks (spheres) | 8 | 10 | 2(d+2) |
| Power diagram | 4 | 5 | d+2 |

**(DVD-algorithm)** Let $N$ be the number of spheres in the three-dimensional space. The DVD algorithm of spheres consists of three major steps:

i) Construct the initial Voronoi diagram.
ii) Compute the initial events of the Voronoi diagram.
iii) Process the most immediate event in future, propagate its influence in the neighbor, and update the event set.

The algorithm repeats Step iii) for a sufficient number of times. Step i) takes $O(N^3)$ time in the worst case but $O(N)$ on average, particularly for the conjunction prediction problem. Be aware that the combinatorial complexity of the Voronoi diagram in the three-dimensional space is already quadratic: I.e. $M = O(N^2)$ where $M$ represents the number of V-vertices, V-edges, and/or V-faces. However, $M = O(N)$ on average. It is safe to consider the average case in conjunction prediction. Step ii) takes $O(M \log M)$ time if the events are stored in a priority queue implemented by heap data structure where each of its nodes corresponds to a V-edge using the event time as a key. Step iii) takes $O(K \log M)$ time for processing $K$ events during the prediction time window. This is because the processing of each event from the queue takes $O(\log M)$ time if the heap is correctly maintained (i.e. height-balanced). Note that $K \propto N$, $K \propto V$, and $K \propto T$ ($V$: the velocity of objects; $T$: the length of the prediction time). Both the Voronoi diagram and priority queue require $O(M)$ memory. The events occurred over the prediction time horizon is stored in an event history queue which takes $O(K)$ memory. We store the predicted history of consecutive flipping events in the **COOP-HSTRY file** taking $O(K)$ memory.

Given the COOP-HSTRY file, the topology structure of the Voronoi diagram of any arbitrary moment $t^*$ in the prediction window can be efficiently produced (i.e. $O(k)$ time for $k$ flipping events before $t^*$) by scanning the appropriate portion of the flipping history. Then, the construction of the Voronoi diagram at $t^*$ requires only $O(N)$ time with the valid topology structure information.

- Given the flipping history in the COOP-HSTRY file, it takes $O(k + N)$ time for constructing the Voronoi diagram at an arbitrary moment $t^*$ where there are $k$ flipping events before $t^*$.
- This proves that the proposed algorithm is event-based, efficiently replay-able. Voronoi diagrams are general-purpose and independent of coordinate system.

**(Root-finding of polynomials)** One of the most critical and time-consuming step for DVD algorithm is to find the roots of a polynomial in Eq. (1). If the root could not be computed correctly or accurately, the topology of the Voronoi diagram would be also incorrect. Fig. 11 shows the catastrophic consequence in dynamic Voronoi diagram of disks by using a poor root-finding library. Fig. 11 (a) shows mutually disjoint disks and the corresponding Voronoi diagram at $t_0 = 0$ and Fig. 11 (b) a zoom-up of the northeast corner. Consider a V-edge *e* in Fig. 11 (b) which is defined by two red disks. Because the root of the polynomial for the event time of *e* is not correctly located, *e* does not flip. This false decision results in incorrect Voronoi diagram. Fig. 11 (c) shows the disk set and its incorrect Voronoi diagram where some Voronoi edges crosses each other at $t_1 > t^*$ where $t^*$ is the flip time at which *e* should have flipped. Continuing the algorithm propagates the incorrect topology of the Voronoi diagram as shown in Fig. 11 (d), (e), and (f). This erroneous situation shows the importance of the accuracy of root-finding process for DVD algorithm.

Refer to the following article for the details on the content of this section.

- Chanyoung Song. Jehyun Cha, Mokwon Lee, Deok-Soo Kim, Dynamic Voronoi Diagram for Moving Disks, *IEEE Transactions on Visualization and Computer Graphics*, DOI: 10.1109/TVCG.2019.2959321, 2019.

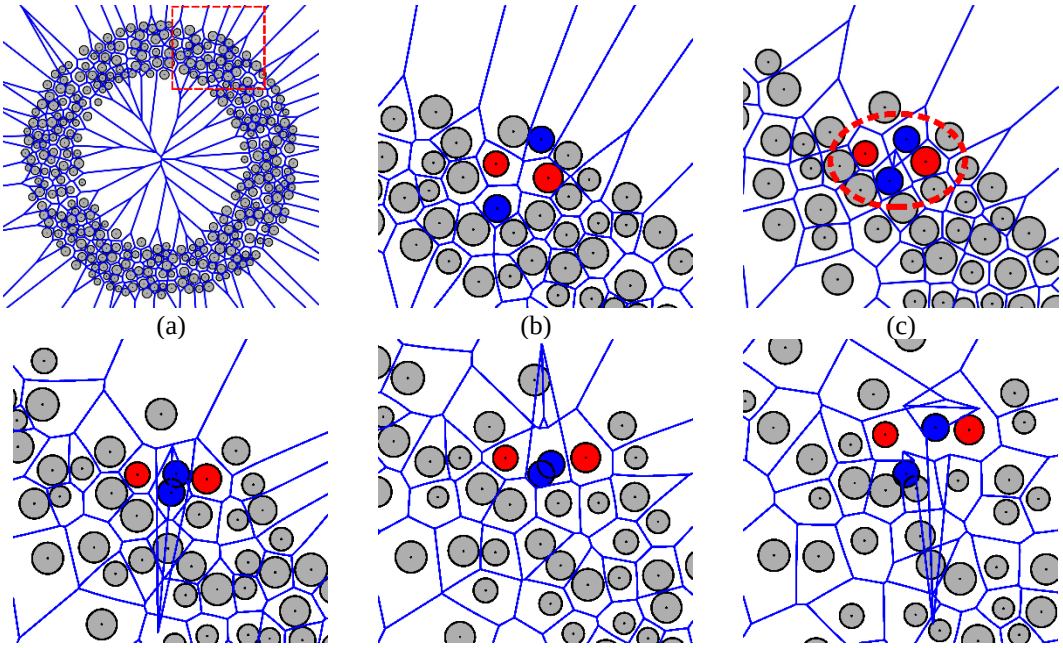



Fig 11. Missing an edge-flip event due to incorrect polynomial root finding in 2-diminsion. (a) A set of 324 static disks and the Voronoi diagram at $t_0$, (b) a zoom-up of the northeast corner of (a), (c) a missed edge-flip at $t_1$, (d), (e), and (f) the undetected disk collisions due to the missing edge-flip ($t_2$, $t_3$, and $t_4$) ($t_0 < t_1 < \ldots < t_4$).

## 5. COOP2 (Conjunctive Orbital Object Predictor and Planner)

Suppose that we model Resident Space Objects (RSOs) catalogued in JSpOC using spherical balls. Then we can define and compute the dynamic Voronoi diagram (DVD) of moving spherical balls. Given the DVD, we develop the COOP2 (Conjunctive Orbital Object Predictor and Planner) algorithm/program as shown in Fig. 12 using the powerful capabilities of DVD in spatial reasoning accurately, efficiently, and conveniently. COOP2 predicts the conjunctions among moving RSOs and generates COOP-HSTRY files. Once COOP2 predicted conjunction(s) and generated COOP-HSTRY, COOP2 can find the optimal maneuver pathway using the algorithm explained in Section 6. The developed DVD and COOP2 algorithms are based on the Voronoi Diagram (V) library which is developed by Voronoi Diagram Research Center (http://voronoi.hanyang.ac.kr).

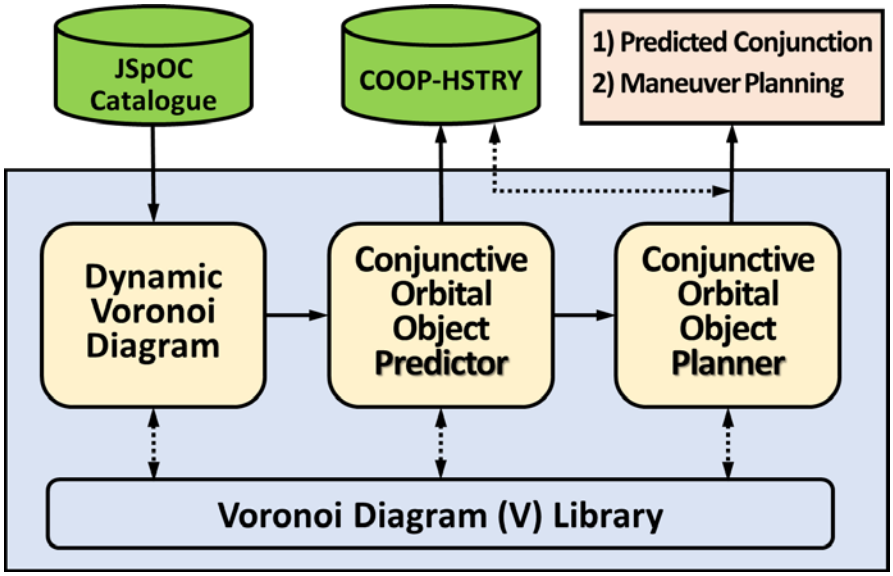


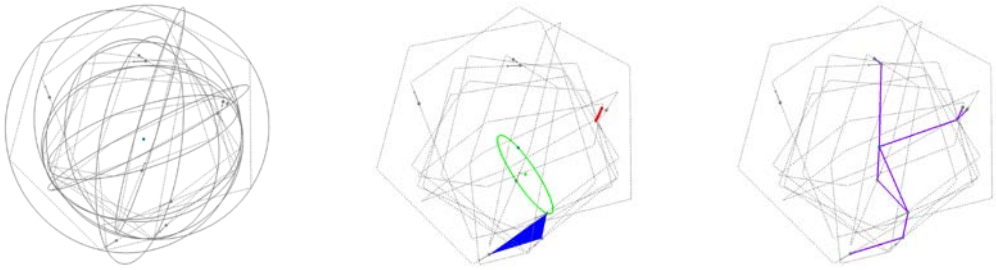

Fig. 12. Computational flow of COOP2 (Conjunctive Orbital Objects Predictor and Planner):

The DVD assumes that **the generators (points, disks, spherical balls) move linearly at constant speeds** as discussed in Section 4.3. In this section, we discuss the proximity analysis to define a model which is used for the DVD and COOP2. Then we develop the COOP2 algorithm which can predict all of the conjunctions of the space objects using spherical ball model that move linearly at constant speeds.

### 5.1. Analysis of Proximity among Orbital Objects in COOP2

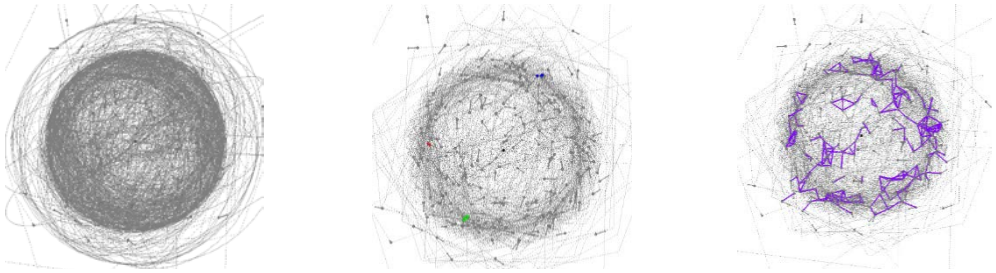
Fig. 13(a) shows the replicas of 10 geospace objects in Space Catalogue at an arbitrary moment, their orbits, and the **piecewise linear approximations** with five line segments. Recall that in this report an object follows an elliptic orbit whereas an object replica moves through a linear approximated path. Fig. 13(b) shows the **closest replica pair** (the red line segment) which defines the shortest distance between two replicas, the **closest triplet** which defines the minimum radius circle (the green circle), and the minimal triplet which defines the minimum area triangle (the blue triangle). Fig. 13(c) shows the **proximal pairs whose distances are within a threshold**, in this case 6,000 km. Given the Voronoi diagram of the moment of interest in the prediction time window, all answers of the queries in Fig. 13(b) and (c) can be produced in **O(*M*)** time in the worst case where *M* is the number of entities (i.e. V-vertices, V-edges, and V-faces) of the Voronoi diagram. For the space objects, ***M* = *O*(*N*) on average where *N* represents the number of replicas**.

Fig. 14 shows the same analyses as Fig. 13 but with 250 replicas: Fig. 14(c) shows the proximal pairs within 1,500 km. Fig 6 and 7 correspond to Fig 13 and 14 with the orbit resolution level of 20 line segments, respectively. We observe that the outputs are different depending on the resolution level. Be aware that the results in Fig. 13 through 16 are from object replicas, not from objects.



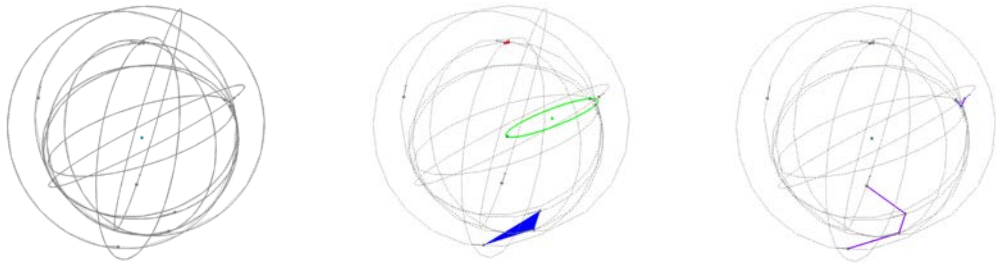
(a) (b) (c)

Fig. 13. An example of proximity analysis for COOP2. Ten object replicas. Orbit approximation with five line segments. Object replicas follow approximated piecewise linear paths. (a) Orbits and their linear approximations. (b) Object replicas: The closest pair (the red line segment), the closest triplet (which defines the minimum radius circle; the green circle), and the minimal triplet (which defines the minimum area triangle; the blue triangle). (c) Object replicas: The proximal pairs whose distances are within 6,000 km.



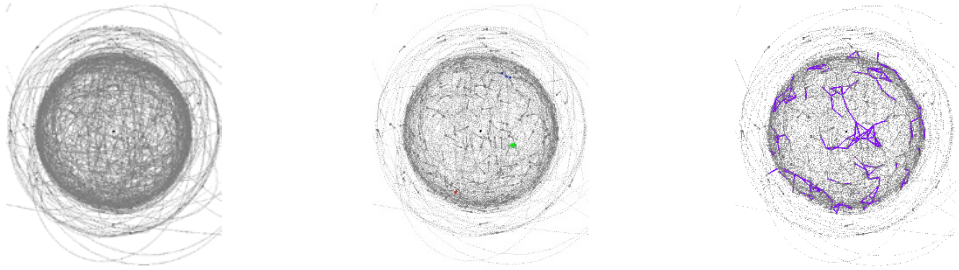
(a) (b) (c)

Fig. 14. Same as Fig. 13 with 250 object replicas. (a) Object replicas and the approximations of the orbits. (b) Proximity analysis as Fig. 13. (c) The pairs within 1,500 km.



(a) (b) (c)

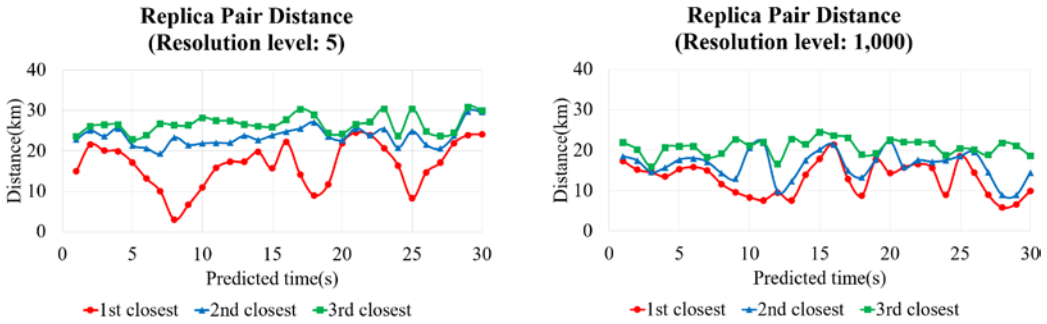
Fig. 15. Repeat of Fig. 13 with the orbit approximation with 20 line segments. The output is different from Fig. 13.



(a) (b) (c)

Fig. 16. Repeat of Fig. 13 with the orbit approximation with 20 line segments. The output is different from Fig. 13.

Fig. 17 shows the distances of 10,000 replica pairs with short distances over prediction time window of 30 seconds. The red curve corresponds to the distance between two closest replicas; The blue one for the replica pair with the second shortest distance; The green one for the third shortest distance. Fig. 17(a) and (b) correspond to the resolution levels of five and one thousand line segments, respectively. **The three curve values at an arbitrary moment can be all computed in *O*(*M*) time for *M* Voronoi entities if the Voronoi diagram is available.** Similar analyses can also be done easily for **the closest triplet, the minimal triplet, etc., all in *O*(*M*) time**. Recall that $M = O(N)$ on average for $N$ replicas.




(a) (b)

Fig. 17. Distances between replicas in the neighbor. 10,000 replicas of objects from Space Catalogue. 30 seconds prediction time window. (Red: The closest pair; Blue: The second closest pair; Green: The third closest pair). (a) Approximation with 5 line segments. (b) Approximation with 1,000 line segments.

## 5.2. Quality and Efficiency of COOP2 Solutions

The COOP2 (Conjunctive Orbital Object Predictor and Planner) algorithm guarantees to include the globally shortest distance pair of orbit objects in the reduced replica pair set. Therefore, if such an object pair defines a conjunction, the algorithm guarantees to find it with a small size of search space. In addition, it can be easily and effectively parallelized.

### 5.2.1. Guaranteed conjunction solution

Let $O_A$ and $O_B$ be two objects on the respective orbits and consider two consecutive moments $t_i$ and $t_{i+1}$ in the prediction time window (Fig. 18). Let $L$ be the line segment connecting the two points $O_i$ and $O_{i+1}$ corresponding to the two moments on the orbit of $O_A$ (i.e. $O_i = O_{A,i} = O_A(t_i)$,). Suppose that a replica of $O_A$, say $\hat{O}_A$, is moving through $L$ (= $L_A$) during the time interval $\Delta t = t_{i+1} - t_i$ with a constant speed.

Let $\varepsilon_A$ be the positional error of the replica $\hat{O}_A$ from the orbit object $O_A$. Suppose that $\Delta t$ is small. We model the elliptic orbit corresponding to $\Delta t$ with a circular arc: $r(t_i) = r(t_{i+1})$ where $r(t_i)$ represents the distance of $O_A$ from the earth at $t_i$. We further model as $O_A$ moves at a constant angular velocity: $\theta(t) - \theta(t_i) = 2\omega(t - t_i)$, $0 < \omega \in R$, where $\theta(t)$ is the true anomaly at $t$ (i.e. the angle between the orbit object and the direction of periapsis as seen from the earth). $\hat{O}_A$ moves linearly at a constant speed.

For notational convenience, let $O = O(t)$, $\hat{O} = \hat{O}(t)$, and $r = r(t_i) = r(t_{i+1})$. Let $T = t - t_i$ be the elapsed time from $t_i$ and $\Theta = \theta(t) - \theta(t_i)$ be the angular displacement of $O$ from $t_i$. Let $\Theta/2 = \omega T$ for some constant $\omega$ as $\Theta$ is linear to $T$ (explained above). Then, the positional error $\varepsilon = \varepsilon_A$ is given by the law of cosine as

$$\varepsilon^2 = l^2 + q^2 - 2lq\cos\beta \qquad (1)$$

from the triangle $\triangle O_i O\hat{O}$. Let $l$ be the displacement of $\hat{O}$ from $O_i$, $q$ the distance between $O$ and $O_i$, and $\beta = \angle O O_i \hat{O}$.

Then,

$$l = |L|T/\Delta t \tag{2}$$

where $|L|$ is the length of $L$. Note that $\hat{O}$ moves along $L$ with a constant speed. From the isosceles $\triangle O_i O F$, the following equations are obtained.

$$q = 2r \sin\frac{\theta}{2} = 2r \sin \omega T \tag{3}$$

$$\beta = \frac{\pi-\theta}{2} - \alpha = \frac{\pi}{2} - (\omega T + \alpha) \tag{4}$$

where $\alpha = \angle \hat{O} O_i F$ and do not change between $t_i$ and $t_{i+1}$. Substituting Eq.s (2), (3), and (4) into Eq. (1) produces

$$\varepsilon^2 = l^2 + q^2 - 2lq\cos\beta = l^2 + q^2 - 2lq\sin(\omega T + \alpha) = \frac{|L|^2 T^2}{\Delta t^2} + 4r^2 \sin^2 \omega T - \frac{4|L|Tr \sin \omega T \sin(\omega T+\alpha)}{\Delta t}. \tag{5}$$

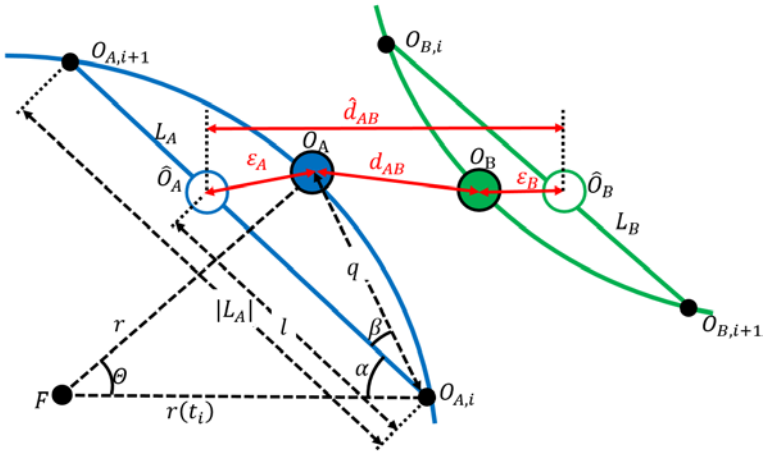



Fig. 18. Positional error between an orbit object and its replica at an arbitrary moment $t_i \leq \hat{t} \leq t_{i+1}$. The elliptic orbit is approximated by a circular arc with a constant angular velocity for a small time interval.

We verified Eq. (5) by calculating the true positional errors of a linearly moving replicas from the corresponding objects on their elliptic orbit in Space Catalogue. Orbits are approximated by ten line segments in the way that traveling time between consecutive vertices are identical. Fig. 19(a) shows the experimental result of the object INS-1A (Eccentricity: 0.001; Semi-major axis: 6,878 km). The red circles are the calculated errors *CalcError* by Eq. (5) which are almost identical to the real errors *RealError* (thus hidden by the red circles). The green curve corresponds to *CalcError* – *RealError*. For each line segment approximation, we evaluated 100 sample points with a constant time increment. As $r(t_i) \neq r(t_{i+1})$ on the elliptic orbit, we used $r = (r(t_i) + r(t_{i+1}))/2$ for the circle approximation. Note that the calculated, predicted positional error by Eq. (5) in this case is very close to the real one only with ten line segment approximation.

Fig. 19(b) and (c) show the same experiments with the objects BREEZE-M R/B (Eccentricity: 0.05; Semi-major axis: 45,461 km) and DELTA 1 R/B(1) (Eccentricity: 0.2; Semi-major axis: 8,642 km), respectively. The blue circles denote the real errors. We observe that **the deviation of predicted positional error increases as orbit eccentricity increases**. In Fig. 19(c), the largest deviation is even about 100 km: Recall that we approximated the orbits with just ten line segments!

Note that the green curve behaves in a bit weird fashion at the common locations between elliptic orbit and its piecewise linear approximation. This is because we assumed that $r = r(t_i) = r(t_{i+1})$ in the error formula in Eq. (5). We also observe that the positional error decreases as the true anomaly approaches 180 degree. Fig. 19(d) shows the profile of orbit eccentricity in the entire Space Catalogue.

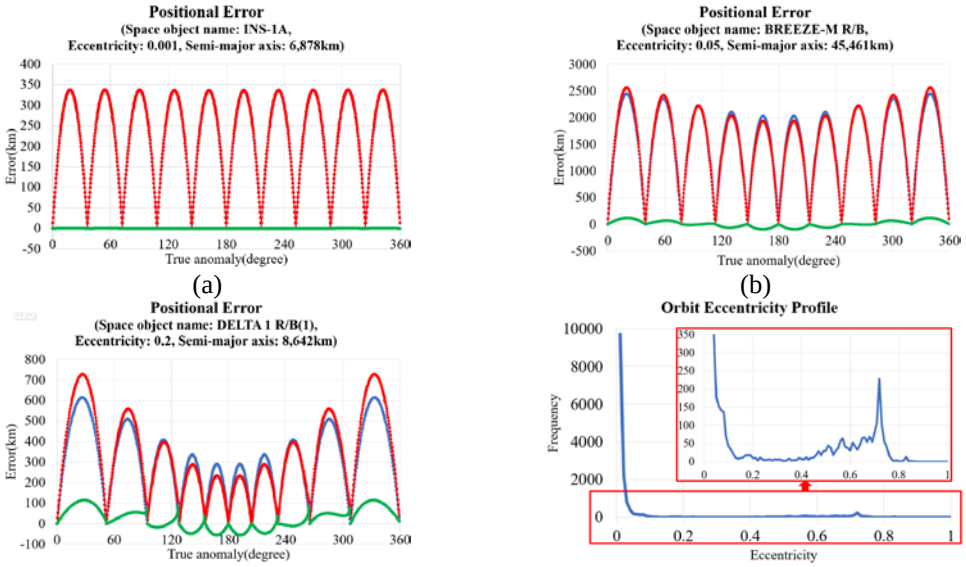



Fig. 19. The real and calculated positional errors. Red: calculated error; Blue: real error; Green: Red – Blue. (a) Space object INS-1A (Eccentricity: 0.001; Semi-major axis: 6,878 km), (b) BREEZE-M R/B (Eccentricity: 0.05; Semi-major axis: 45,461 km), (c) DELTA 1 R/B(1) (Eccentricity: 0.2; Semi-major axis: 8,642 km), (d) The profile of the eccentricity of the orbits in Space Catalogue.

In a similar fashion, $L_B$ and $\hat{O}_B$ of the orbital object $O_B$ can be defined together with the positional error $\varepsilon_B$. Recall that the replicas move through piecewise linear paths with constant speeds while objects move through their respective elliptic orbits under Keplerian motion.

Suppose that the true shortest distance $d^*_{AB}$ between $O_A$ and $O_B$ occurs at $t_i \le t^* \le t_{i+1}$ at the locations, say $O^*_A$ and $O^*_B$, on their respective orbits. Suppose that $\hat{O}_A$ and $\hat{O}_B$ are identified, from the **Voronoi diagram**, as the **closest pair among all possible replica pairs** at $t_i \le \hat{t} \le t_{i+1}$ (Fig. 19). Note that this closest pair between replicas can be found from the Voronoi diagram by checking the V-faces, thus taking O($M$) time for $M$ Voronoi faces where $M$ = O($N$) on average for our problem for $N$ orbital objects. It is important to note that $t^*$ is not necessarily identical to $\hat{t}$.

Let $\hat{d}_{AB}$ be the distance between the centers of the replicas $\hat{O}_A$ and $\hat{O}_B$ at $\hat{t}$. Then, it is obvious that the shortest distance $d^*_{AB}$ between two elliptic orbits satisfies

$$\hat{d}_{AB} - (\varepsilon_A + \varepsilon_B) \le d^*_{AB} \le \hat{d}_{AB} + (\varepsilon_A + \varepsilon_B). \quad (6)$$

Let $\varepsilon = \max(\varepsilon_A, \varepsilon_B)$. Then,

$$\hat{d}_{AB} - 2\varepsilon \le d^*_{AB} \le \hat{d}_{AB} + 2\varepsilon. \quad (7)$$

Eq. (6) and (7) imply that **the replica pair $\hat{O}_A$ and $\hat{O}_B$ with the shortest distance $\hat{d}_{AB}$ on the linear approximations may not necessarily correspond to the object pairs with the shortest distance $d^*_{AB}$ among all objects on orbits**.

The same equation, however, also implies that we can exclude to filter out any replica pair $\widehat{O_i}$ and $\widehat{O_j}$ which satisfies the condition

$$|\hat{d}_{IJ} - \hat{d}_{AB}| \geq 2\varepsilon \tag{8}$$

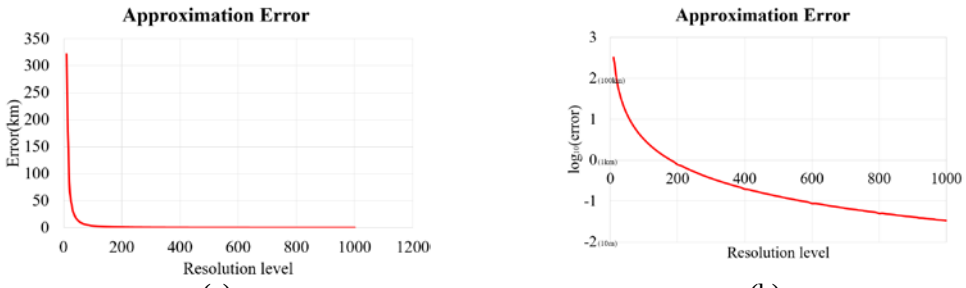
from the consideration. **The globally shortest distance pair of objects on the orbits can be correctly found from the set of replica pairs after the filter in Eq. (8) is applied**. Let *X* be the set of replica pairs, called the reduced replica pair set, after the filtering process.
It can be shown that $t^*$, $O_A^*$, $O_B^*$, and $d_{AB}^*$ can be found by applying numerical procedure to each reduced replica pair ($\hat{O}_A$, $\hat{O}_B$) using $\hat{t}$ as an initial condition. We make the following observations:

- **(Obs 1. Existence of Solution)** The shortest distance object pair exists in the reduced replica pair set *X*.
- **(Obs 2. Size of Search Space)** The value of $\varepsilon$ and the size of *X* have a strong positive correlation.

### 5.2.2. Size of search space

The history of the events (i.e. V-edge flip and V-face flip events) during the entire prediction time window is stored in the COOP-HSTRY file. Given a COOP-HSTRY file, the topology of the Voronoi diagram of any arbitrary moment, say *t*, in the conjunction prediction time window can be quickly computed in the linear time of the number *k* of events up to *t* and the geometry of the Voronoi diagram *VD*(*t*) at *t* can be computed in the linear time of the number *N* of object replicas. The COOP2 program can be easily modified to efficiently solve many other useful spatial queries. Two examples:

- Suppose that a user wants to identify the neighbor replicas of a particular replica, say $\hat{O}$, within a threshold $\tau$ at *t*. Then, locating the replica of the object *O* in *VD*(*t*) and checking the V-faces of the V-cell of $\hat{O}$ can solve the problem. Given *VD*(*t*), this query can be done in O(N) time.
- Suppose that a user wants to see the distance of the nearest neighbor of a particular replica $\hat{O}$ over the entire prediction time window. This can also be done quickly by scanning the entire COOP-HSTRY file by performing the flipping events and checking the V-faces of the V-cell corresponding to $\hat{O}$. This query can be done in *O*(*k N*) time.



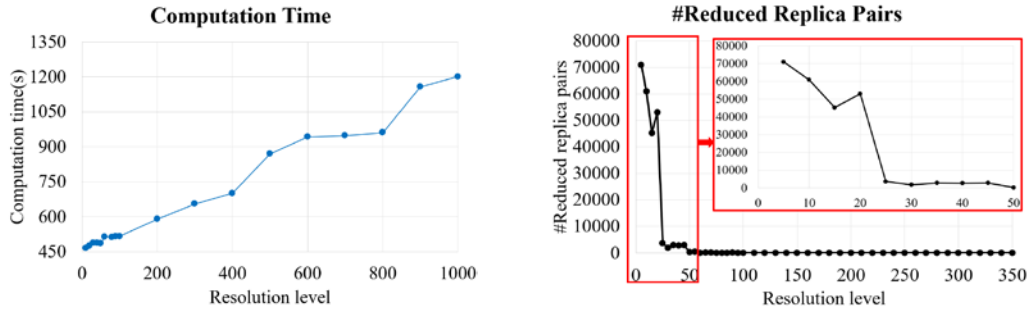

(a) (b)

Fig. 20. Approximation error with respect to the number of approximating line segments of an orbit. (a) Absolute error. (b) Log-scale error.

Approximation error is obviously a function of resolution level. See Fig. 20. The resolution level eventually transforms to computational requirement, firstly, to find conjunctions among replicas and secondly, to find the conjunctions among real orbital objects. Fig. 21(a) shows the computation time to find the conjunctions among replicas, not among orbital objects, including the computation of the Voronoi diagram of both the initial and the dynamic over the entire prediction time window. Note that the curve in Fig. 21(a) increases in a strongly linear fashion. This is because **there are more velocity vector changes as there are more approximating line segments for orbits during the prediction time**. When a velocity vector is modified, to maintain a correct Voronoi diagram, it is necessary to update the V-flipping time of the V-edges and V-faces in the neighborhood of the velocity-changed replica. **The number of velocity changes is obviously linear to the number of the line segments**.

The search space size observation (Obs 2) above says that if we approximate two orbits with more line segments, we need to check a smaller set of reduced replicas which guarantees no missing conjunctions among orbital objects. Fig. 21(b) shows the size of the reduced replica pair set which contains the shortest distance (i.e. the real conjunction) among orbital objects as the consequence of Obs 2 above. **As the resolution level gets higher (i.e. the approximation error becomes smaller), there are fewer replica pairs from which the real conjunction should be tested**.
However, Fig. 21(a) says that the computation to get the reduced replica set takes more time for resolution level increase. Fig. 21(a) and (b) together yield a trade-off which implies the existence of an optimal condition of resolution level which can be determined by a computational experiment conjunction problem. The optimality may depend on platform. Thus, we make the following observation.

- **(Obs 3. Optimality of Efficiency)** There exists an optimal resolution level for the minimum computation time in conjunction prediction.




(a) (b)

Fig. 21. The influence of approximation quality by resolution level (10,000 objects from Space Catalogue, 30 seconds prediction time window, Intel Core i7-6700 3.4Ghz, 16Gb RAM, Win10 64bit, one core used). An orbit approximation resolution level is the number of line segments approximating an orbit. (a) Computation time of both the initial Voronoi diagram and the dynamic Voronoi diagram (i.e. the V-flip history) over the entire conjunction prediction time window. (b) The number of reduced replica pairs.

## 6. Conjunction Prediction and Maneuver Planning with COOP2

The COOP2 algorithm/program can perform spatial reasoning among RSOs accurately, efficiently, and conveniently. When it runs to predict future events, the COOP2 program produces the COOP-HSTRY data which stores the information about topology change in the Voronoi diagram of *n* RSOs during the prediction time. With the COOP-HSTRY data and the initial static Voronoi diagram at $t = 0$, the static **Voronoi diagram for arbitrary time *t* can be computed in $\mathbf{O}(|\boldsymbol{H}| + \boldsymbol{n})$ time in the worst case where $|\boldsymbol{H}|$ represents the size of COOP-HSTRY data before *t***. Hence, any spatial reasoning at *t* can be done by the efficiently computed static Voronoi diagram at that moment.

Among diverse applications of the COOP-HSTRY data, we are interested in the following two problems: Conjunction interval search problem (CIS-problem) and maneuver path optimization problem (MPO-problem). The CIS-problem is to find the conjunction interval between all possible pairs of RSOs. MPO-problem is to find the best conjunction-avoiding maneuver path by generating-and-testing through quick-and-accurate evaluation of multiple hypothesized paths.

Suppose that Space Catalogue has *n* RSOs (or equivalently *n* orbits) and thus $O(n^2)$ RSO pairs. We make a linear approximation of each orbit with $M_L$ line segments corresponding to some discrete moments in time. Consider a line segment *L* approximating an elliptic orbit. Let $\hat{O}$ be the replicas of an RSOs *O*: $\hat{O}$ move through *L* with a constant speed whereas *O* move through its elliptic orbit. Let $\varepsilon$ be the positional error between *O* and $\hat{O}$. We take advantage of the fact that reasoning the spatial proximity among the linearly moving replicas can be easily done via the Voronoi diagram of the replicas.

### 6.1. Conjunction Interval Search (CIS)

A conjunction interval is an interval within a prediction time window where the mean distance between any two RSOs is less than a predefined threshold $\theta$. A conjunction interval contains one or more Time of Closest Approach (TCA) which gives a local minimum of the distance between any two RSOs. A CIS-problem is to **find all RSO pairs and time interval where each has an inter-object distance less than $\boldsymbol{\theta}$**. In a CIS-problem, being a building block of SSA/STM, we are interested in an efficient, convenient, and robust algorithm that finds all conjunction intervals within a prediction time window using the COOP-HSTRY data. We emphasize that the **CIS-problem is one of the most fundamental geometric problems in SSA/STM**.

Suppose that Space Catalogue is a set of $n$ RSOs where each RSO follows an elliptic orbit, i.e. $O(t) = \{o_1(t), o_2(t), \ldots, o_n(t)\}$ where $O(t)$ is the Catalogue. Consider a line segment $L_i$ that approximates the elliptic orbit of $o_i$. Let $\hat{O}(t) = \{\hat{o}_1(t), \hat{o}_2(t), \ldots, \hat{o}_n(t)\}$ denotes the replicas for the objects of $O(t)$. The object $o_i$ moves through its elliptic orbit and $\hat{o}_i$ moves through $L_i$ and has a positional error $\varepsilon_i$ from $o_i$.

Mathematically, the CIS-problem is to find all RSO pairs $o_i$ and $o_j$where each pair is associated with a time interval $t_{start}$ and $t_{end}$ where

$$d_{ij}(\tau) \leq \theta \tag{9}$$

for the inter-object distance $d_{ij}(\tau)$ and $\tau \in [t_{start}, t_{end}]$. In other words, a CIS-problem is to find all $\{o_i, o_j, t_{start}, t_{end}\}$, called the **CI-quadruplet**, satisfying Eq. (9).

Let $\hat{d}_{ij}$ denotes the distance between two replicas $\hat{o}_i$ and $\hat{o}_j$. We showed in Eq. (6) that

$$\hat{d}_{ij} - (\varepsilon_i + \varepsilon_j) \leq d_{ij} \leq \hat{d}_{ij} + (\varepsilon_i + \varepsilon_j) \tag{10}$$

Eq. (10) implies that the true distance between RSOs can be bound by the distance between replicas if their positional errors are correctly estimated. From Eq. (9) and (10), the equation below is satisfied:

$$\hat{d}_{ij} - (\varepsilon_i + \varepsilon_j) \leq d_{ij} \leq \theta \tag{11}$$

Eq. (11) can be easily transformed to

$$\hat{d}_{ij} \leq \theta \ + (\varepsilon_i + \varepsilon_j). \tag{12}$$

Eq. (12) leads to the following Lemma.

**Lemma 1.** If $\hat{d}_{ij} \leq \theta \ + (\varepsilon_i + \varepsilon_j)$, then $d_{ij} \leq \theta$.

Lemma 1 implies that **the proper distance threshold for replicas can be bounded with the original threshold and positional errors**. Let $[\hat{t}_{start}, \hat{t}_{end}]$ be the conjunction interval for a replica pair satisfying Eq. (12). In other words, $\hat{d}_{ij}(\tau) \leq \theta \ + (\varepsilon_i + \varepsilon_j), \tau \in [\hat{t}_{start}, \hat{t}_{end}]$. From Lemma 1, it can be also shown that

$$[t_{start}, t_{end}] \subset [\hat{t}_{\text{start}}, \hat{t}_{end}]. \tag{13}$$

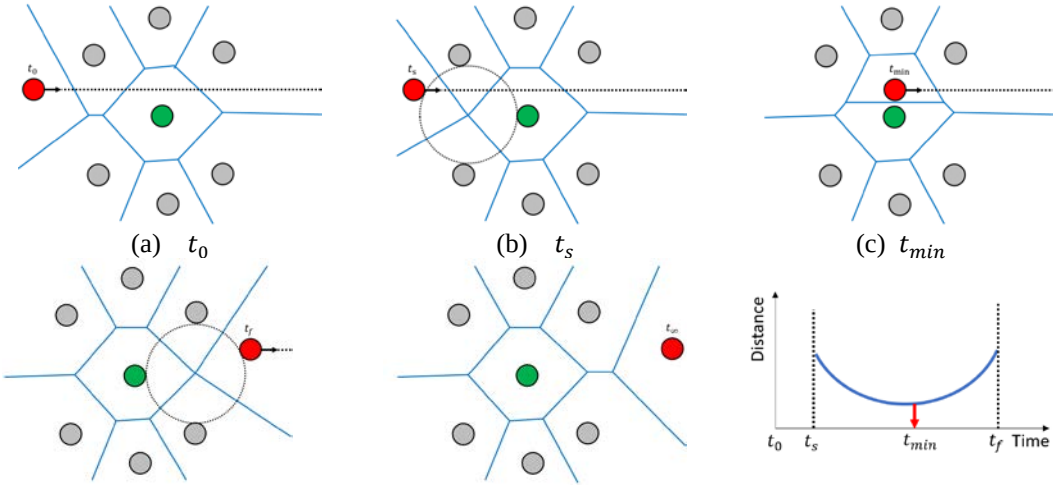
Eq. (13) implies that we are guaranteed not to miss any conjunction interval if we find the replica pairs satisfying Eq. (12) and closely investigate their corresponding orbital objects. Hence, the following theorem holds.

**Theorem 2. The conjunction interval for replica always contains the conjunction interval for RSOs**.

**(Example: 2D)** Fig. 22 shows an example of the process to track the inter-replica distance using the COOP-HSTRY data in the plane. Note that the figure illustrates a relative motion analysis. While we give the explanation in the plane, the observation easily applies to the three-dimensional space. All particles in the figure are fixed except the horizontally moving red one and we want to analyze the distance between the red and green. Suppose Fig. 22(a) is an initial state at time $t_0$: The red and green are not V-neighbors to each other which implies that they are not the first order neighbors in the Voronoi diagram (We call two objects V-neighbors if they share a Voronoi edge). In this case, we consider they do not define a distance.

Suppose the red moves linearly with a constant speed to the direction given by the arrow and also suppose that there exists $t_s > t_0$ such that four objects, including both the green and the red, define a common empty tangent circle as shown in Fig. 22(b). This implies that, at $t_s$, the red and green begin to be V-neighbors and thus the inter-object distance is defined. From the algorithm point of view, the Voronoi edge which shrinks to the point (corresponding to the center of the tangent circle) flips. The distance is well-defined until $t_f > t_s$ where another edge-flip occurs at $t_f$ as shown in Fig. 22(d). Note that the local closest approach occurs at $t_{min}$ as shown in Fig. 22(c). The red and green are not V-neighbors any more after $t_f$ and thus no distance is defined between the red and green at $t_\infty > t_f$ in Fig. 22(e). Fig. 22(f) shows the distance function between the red and green objects. For the relative position $p = \{p_1, p_2\}$ and velocity $v = \{v_1, v_2\}$, $p, v \in \mathbb{R}^2$, the distance $d$ between linearly moving replicas with constant speeds is given as a square-rooted quadratic function

$$d^2 = \sum v_i^2 t^2 + 2 \sum v_i p_i t + \sum p_i^2. \tag{14}$$




(a) $t_0$ (b) $t_s$ (c) $t_{min}$

(d) $t_f$ (e) $t_\infty$ (f)

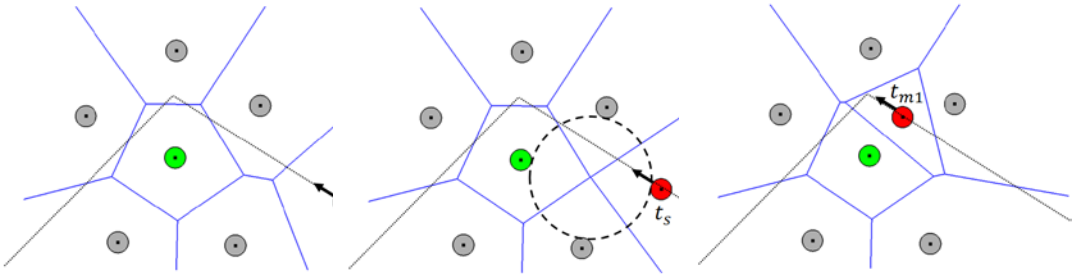
Fig. 22. An example of distance curve generation using the COOP-HSTRY data in the plane without velocity change. Suppose that we want to analyze the distance between the red(moving) and green(fixed). (a) Initial state at time $t_0$ (b) Edge-flip event time. (c) Moment of local closest approach. (d) Another edge-flip event time. (e) Arrival time. (f) Arbitrary distance curve between the red and green.

Therefore, the following lemma holds.

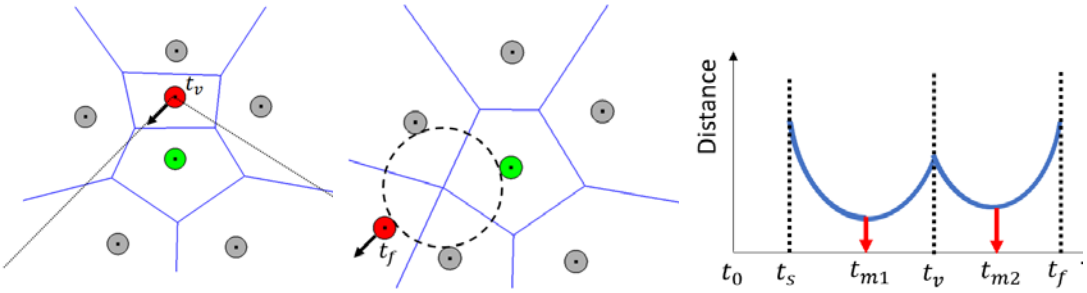
**Lemma 3.** Suppose that **there is no velocity change for two linearly moving objects**. Then, the **distance between the two objects consists of one, and only one, curve segment** of the function in Eq. (14).

The non-zero segment of the distance function is bounded by the two times, say $t_s$ and $t_f$, given by the two edge-flip events available from the COOP-HSTRY data.

Fig. 23 shows a more general example with a velocity change while two particles maintain their V-neighborhood. In this example, all conditions are identical with those of the example of Fig. 22 except the red changes its velocity vector. When the velocity change event occurs at $t_v$, as shown in Fig. 23(c), the distance function changes as well. Therefore, the whole non-zero distance function of this example consists of two curve segments where each is defined by Eq. (14) and has its own local minima.




(a) (b) (c)




(d) (e) (f)

Fig. 23. An example of distance curve generation using the COOP-HSTRY data in the plane with velocity change. Suppose that we want to analyze the distance between the red(moving) and green(fixed). (a) Initial state at time $t_0$ (b) Edge-flip event time. (c) Moment of one local closest approach. (d) velocity change event time. (e) Another edge-flip event time. (f) Arbitrary distance curve between the red and green.

Suppose that there are $h_v$ velocity changes in COOP-HSTRY data between $t_s$ and $t_f$. Then, the distance function consists of $h_v + 1$ non-zero curve segments where the junctions between two consecutive segments are given by the velocity-change events in the COOP-HSTRY data. In 2D, the distance between two replicas begins to be defined when the V-edge between them is created via an edge-flip operation and vanishes when the V-edge vanishes via another edge-flip operation. In 3D, the distance between two replicas begins to be defined when the V-face between them is created via the edge-flip operation and vanishes when the V-face between them vanishes via a face-flip. Note that, in 3D, a V-face flips to a V-edge and a V-edge flips to a V-face.

Let $|F_{initVD}|$ be the **number of the V-faces in the initial Voronoi diagram** and $|H_{eFlip}|$ the **number of the edge-flip events in the COOP-HSTRY file**. Then, in 3D, the number of total distance functions defined within the entire prediction time window is $|F_{initVD}| + |H_{eFlip}|$, which is the number of V-faces. For each distance function, a velocity change creates a new segment of distance function. Thus, the total number of distance function segments within the entire prediction time window is $O(|F_{initVD}| + |H|)$, where the $|H|$ is the size of COOP-HSTRY during the entire prediction time window. As each distance curve can be computed in a constant time, the next theorem holds.

**Theorem 5.** Time and space complexities of the CIS-problem are $O(|F_{initVD}| + |H|)$ in the worst case.

We want to emphasize, in 2D, that the distance function between two RSOs is defined if, and only if, the two RSOs share a Voronoi edge. Hence, there are O($n$) distance functions in the system at an arbitrary moment for $n$ RSOs as there are O($n$) Voronoi edges. In 3D, there can be O($n^2$) distance functions for $n$ RSOs because two RSOs define a distance function if, and only if, the two RSOs share a Voronoi face.

## 6.2. Maneuver Path Optimization (MPO)

If two RSOs are predicted to be sufficiently close to require a collision mitigating maneuver, we need to maneuver one RSO so that the collision risk can be resolved. Given a configuration of two RSOs, there are usually many alternatives of maneuvers and one would be better than another in terms of a given measure. Hence, these alternatives need to be evaluated by the measure. To the best of our knowledge, subjective judgements by human experts are inputs to the decision-making process of these maneuvers.

In this report, we propose an efficient, convenient, and robust method using the COOP2 algorithm and COOP-HSTRY data as follows. Consider a CI-quadruplet $\{o_i, o_j, t_{start}, t_{end}\}$ with $d_{ij} \leq \theta$ for the threshold $\theta$. We want to maneuver $o_i$, called an **intruding RSO** or **intruder**, while $o_j$ stays on its original orbit. Suppose that the COOP-HSTRY data and a CIS-problem solver are available. Then, we take the "**generate-and-test approach"** in that we first quickly generate multiple **hypothetical maneuver paths** for $o_i$ and test each hypothesis by solving the CIS-problem quickly. We call the **imaginary RSO** orbiting through the hypothesized path an **image** of the intruding RSO. We simply collect the hypothesized paths which are guaranteed to be safe and choose the best one in the given measure as the optimal maneuver path. For simplicity in this report, **we generate hypothesized paths by modifying the orbiting plane**. However, **in practice**, a maneuver hypothesis has either **a higher or a lower altitude than its original orbit**.

In this section, we explain **an algorithm to evaluate a hypothesized path using the COOP-HISTRY data**. Note that the hypothesized path, also its image on the hypothesized path, does not exist in the COOP-HSTRY data. To solve the CIS-problem for each hypothesized path, it is necessary to find the neighbor of the object image on the hypothesized path to define the inter-object distance. Thus, the **MPO-problem** solving process consists of two steps for each hypothesized path: i) **Search the neighbors** and ii) **solve the CIS-problem**.

### 6.2.1. Neighbor Search for each Hypothesized Maneuver Path

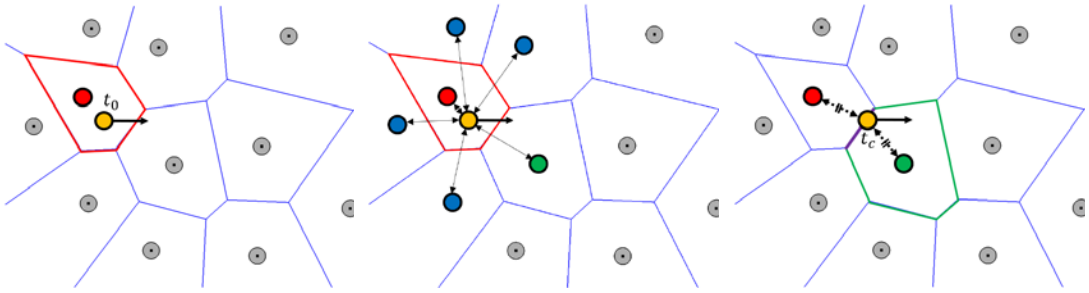
We want to **solve the CIS-problem of the imaginary object *I* on the hypothesized path $\pi$ against all RSOs**, but using the replicas of the RSOs. In order to do it, we also need to create the replica of the image *I* following through a piecewise linear path. Let $\hat{I}$ be the replica of *I* and $\hat{o}_{nearest}(t)$ be the nearest neighbor of $\hat{I}$. This means that the nearest neighbor of $\hat{I}$ is the owner of the Voronoi cell that contains $\hat{I}$. The set of neighbors of $\hat{I}$ is its nearest neighbor $\hat{o}_{nearest}(t)$ plus the neighbors of $\hat{o}_{nearest}(t)$ in a similar concept introduced earlier. Note that $\hat{o}_{nearest}(t)$ is indeed the replica of an orbital object. Let $d_i^I$ be the Euclidean distance between *I* of an intruder on a hypothesized path $\pi$ and a RSO $o_i$ on an arbitrary orbit. Let $\hat{d}_i^I$ be the counterpart of $d_i^I$ for the replicas.

For a given moment *t*, we can find $\hat{o}_{nearest}$ simply by finding the owner of the Voronoi cell containing $\hat{I}$. To find $\hat{o}_{nearest}(t)$ for an arbitrary moment *t*, we trace the COOP-HSTRY data until *t* is reached while we detect the moment, called the **transition moment**, that the nearest neighbor is changed by the motion of $\hat{I}$. We call this process the **Nearest Neighbor Traversal** (NNT). Given an initial nearest neighbor $\hat{o}_{nearest}(t)$ of $\hat{I}$, the moment that the nearest neighbor changes is one that the distance from $\hat{I}$ to $\hat{o}_{nearest}(t)$ is equal to the distance from $\hat{I}$ to one of the neighbors of $\hat{o}_{nearest}(t)$. Therefore, the following lemma is proved.

**Lemma 6.** Let $\hat{o}_\sigma$ be one of the Voronoi neighbors of $\hat{o}_{nearest}(t)$. Then, a nearest neighbor transition occurs when $\hat{d}^I_{nearest} = \hat{d}^I_\sigma$.

If all transition moments are computed for a given initial nearest neighbor, $\hat{o}_{nearest}(t)$ for arbitrary time $t$ can be found.

**(Example: 2D)** Fig. 24 shows an example of the NNT process. Orange ball in Fig. 24(a) denotes $\hat{I}$ and red ball denotes the nearest neighbor $\hat{o}_{nearest}(t_0)$, respectively. Fig. 24(b) shows the Voronoi neighbors of $\hat{o}_{nearest}(t_0)$ (i.e. the blue and green balls). Suppose that the green ball has the transition moment $t_c$. Then, at $t_c$ in Fig. 24(c), the orange ball crosses the Voronoi edge between the Voronoi cells of the red and green balls and the nearest neighbor of the orange changes to the green ball at $t_c$.




(a) (b) (c)

Fig. 24. An example of the NNT process. (a) The initial state at $t_0$ with an image (i.e. the orange ball) and its nearest neighbour (i.e. the red ball). (b) Transition moment search with the neighbours of the current nearest neighbour (i.e. the green and blue balls). (c) Transition moment $t_c$ when the orange ball crosses the Voronoi edge between the red and green balls.

The **transition moment needs to be updated** when the following occurs:

1. The **velocity** of any of the associated balls (i.e. the red, blue, and green balls) changes.
2. One of the **V-neighbors** are changed through the topology update.

The computation of the transition depends on the size of COOP-HSTRY and the following lemma can be proved.

**Lemma 7.** For a given imaginary RSO *I*, the nearest neighbor for the entire prediction time window can be found in $O(|H|)$ in the worst case.

### 6.2.2. CIS-problem for an Imaginary RSO

The CIS-problem for an imaginary RSO *I* can be solved as before except that the neighbor changes not only when topology update occurs but also when a nearest neighbor transition occurs.

**(Example: 2D)** Fig. 25 shows an example of the process to track the inter-object distance using the COOP-HSTRY data in the plane. In Fig. 25, suppose that we want to analyze the distance between the replica of the imaginary RSO $\hat{I}$ (i.e. the orange) and the fixed object (i.e. the green) where the orange moves horizontally. The nearest neighbor and its V-cell are shown as the red ball and polygon, respectively. Fig. 25(a) is the initial state at $t_0$: The distance between the orange and green balls is not defined because the green is not a neighbor of the red. At $t_1$ , the first transition in Fig. 25(b), the nearest neighbor is changed so that the green becomes a neighbor of the orange. Thus, the distance begins to be defined from $t_1$. In Fig. 25(c) and (d), new transitions occur at $t_2$ and $t_3$, respectively, and the neighbourhood condition between the orange and green is maintained until $t_3$. After $t_3$, as shown in Fig. 25(e), they are not neighbor anymore and thus no distance function can be defined. The distance function from $t_1$ to $t_3$ consists of one, and only one, curve segment as there is no velocity change at all.

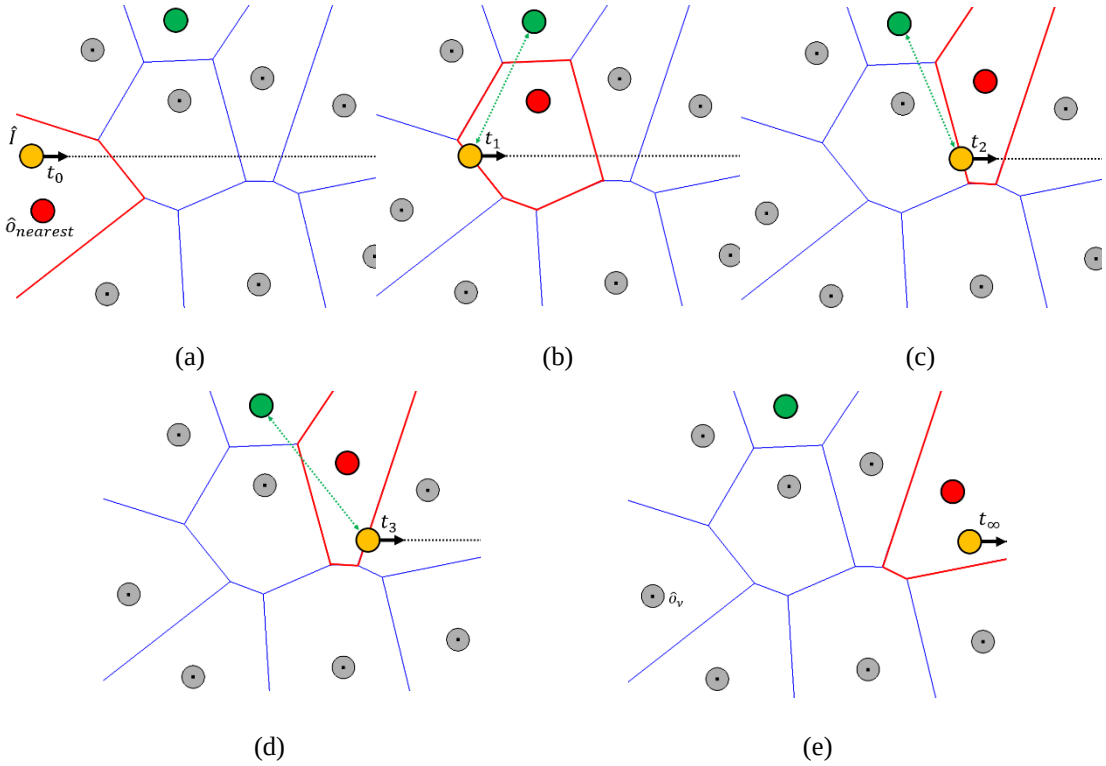



Fig. 25. An example of the distance function generation for the replica of an imaginary RSO using the COOP-HSTRY data with a velocity change in the plane. We want to study the distance between the orange and the green. (a) Initial state at $t_0$ (b) First transition at $t_1$: The green becomes a neighbour of the yellow. (c) Next transition. (d) The last transition at $t_3$: The orange and green are no more neighbours. (e) No distance defined after $t_3$.

In summary, there are three events in processing the COOP-HSTRY data as follows:

- **Topology update event**: If a topology update changes the neighbor of the candidate, a distance function is either created or disappears.
- **Velocity change event**: If a neighbor replica changes its velocity, a new segment of distance function is created.
- **Nearest neighbor transition event**: Given a transition, a distance function is created or disappears for the new neighbor or the old neighbor, respectively. The distance functions for all the common neighbors for the new nearest and old nearest neighbors remain identical.

The number of distance function segments for an imaginary RSO depends on the size of the COOP-HSTRY data. Thus, the worst-case time complexity of CIS for the hypothesis is $O(|H|)$.

**Lemma 8.** There are $O(|H|)$ segments of distance functions for an imaginary RSO over the entire prediction time window in the worst case.

**Corollary 9.** The CIS-problem of an imaginary RSO can be solved in $O(|H|)$ time in the worst case.

Lemma 7 and Corollary 9 lead to the following theorem.

**Theorem 10.** The optimal maneuver plan among *m* hypothesis can be found in $O(m|H|)$ time in the worst case.

### 6.3. Scalability of Conjunction Prediction and Planning

The conjunction-prediction using the COOP2 algorithm is scalable in many aspects.

- **Prediction-window scalability (PW-scalability)**: PW-scalability is based on the observation that the **prediction time window can split into a set of mutually exclusive time slices** and the conjunction prediction of each time slice can be done independent of each other. Thus, the predictions of all time slices can be done in parallel, if an actual collision does not occur between two objects in a preceding time slice. Note that the beginning configuration of each time slice can be easily found. This type of scalability has been commonly used by other existing algorithms/programs.

- **Flip-influence scalability (FI-scalability)**: FI-scalability is unique in the COOP2 algorithm. When a **V-edge (or a V-face) flips** in the topology structure of the Voronoi diagram, there are **a number of V-edges and V-faces whose flipping event time in future need to be re-calculated and these re-calculations are independent of each other**. Hence, these re-calculations can be done in parallel. According to our current experiment, the number of re-calculations per flip is 46 on average and 64 in the worst case. These numbers refer to the possible scale of the speed-up due to FI-scalability. As the largest portion of the computation time of DVD-COOP is this re-calculation of event time, the FI-scalability will easily achieve 46 times acceleration if it is appropriately reflected in algorithm design.

- **Neighbor-pair scalability (NP-scalability)**: NP-scalability is based on the observation that the **conjunction interval search for each distance function can be done independently**. Suppose that we have found the distance functions between the replica pairs. The conjunction interval for replicas can be found instantly from the distance function. To find the true conjunction interval for the orbiting RSOs and the TCA, the solution space search in the conjunction interval for replicas is needed. The solution process of the CIS-problem (distance function → CI for replicas → CI for RSOs → TCA) is independent of each distance function and thus can be solved independently.

- **Hypothesis-evaluation scalability (HE-scalability)**: HE-scalability is based on the observation that the **evaluation of each hypothesized maneuver path can be processed independently**. Suppose that we have generated several hypotheses that follow different orbits. Each hypothesis needs to be evaluated in terms of its CI and TCA and the evaluation for each hypothesis can be processed independently. Recall that each hypothesis evaluation can also be parallelized by the NP-scalability.

Fig. 26 summarizes the structure of these scalabilities. The current version of COOP2 runs on a single core: The computation speed can be significantly improved if these scalabilities are properly incorporated. We plan to implement the scalable version of the proposed algorithm soon.

**Scalabilities in DVD-COOP**

| | COOP-HSTRY Generation | CIS-problem | MPO-problem |
|---|---|---|---|
| Scalability | **Prediction-window scalability(PW)** : Depends on #Cores. | | |
| | **Flip-influence scalability(FI)** : Depends on adjacent Voronoi entities( ≈ 10 ) | **Neighbor-pair scalability(NP)** : #Neighbor pairs (≈ #V-faces/#Cores) | **Hypothesis-evaluation scalability(HE)** : #Hypothesis |

Fig. 26. The structure of scalability properties in the COOP2 algorithm.

## 7. Experiments and Discussion

We have implemented the developed maneuver path optimization algorithm and tested for performance evaluation. Computation environment: i7-6700 3.4Ghz CPU, 16Gb RAM, Microsoft Visual Studio C++ 2017. Note that we used only one core in this experiment. All data used in the experiment is TLE data downloaded from JSpOC.

### 7.1. Experiment on Computational Efficiency

Fig. 27(a) depicts the computation time for generating the COOP-HSTRY data vs. the number of RSOs. The curve shows a super linear pattern. Fig. 27(b) depicts the search time for conjunction interval (CI-search time) vs. the number of objects: It shows a super linear pattern. Fig. 27(c) depicts the computation time for generating the COOP-HSTRY data vs. the number of line segments for approximating each orbit: It shows a linear pattern. Fig. 27(d) depicts the CI-search time vs. the number of line segments for approximating each orbit: It shows a linear pattern.

Summary: Both the time for COOP-HSTRY generation and CI-search show super linear trends to the number of RSOs whereas both show linear trends with respect to the number of line segments per orbit. Because the time for COOP-HSTRY generation is in proportion to the size of the COOP-HSTRY data, the time for CI-search time is positively correlated with the size of the COOP-HSTRY data: This observation supports Theorem 2 and 10.

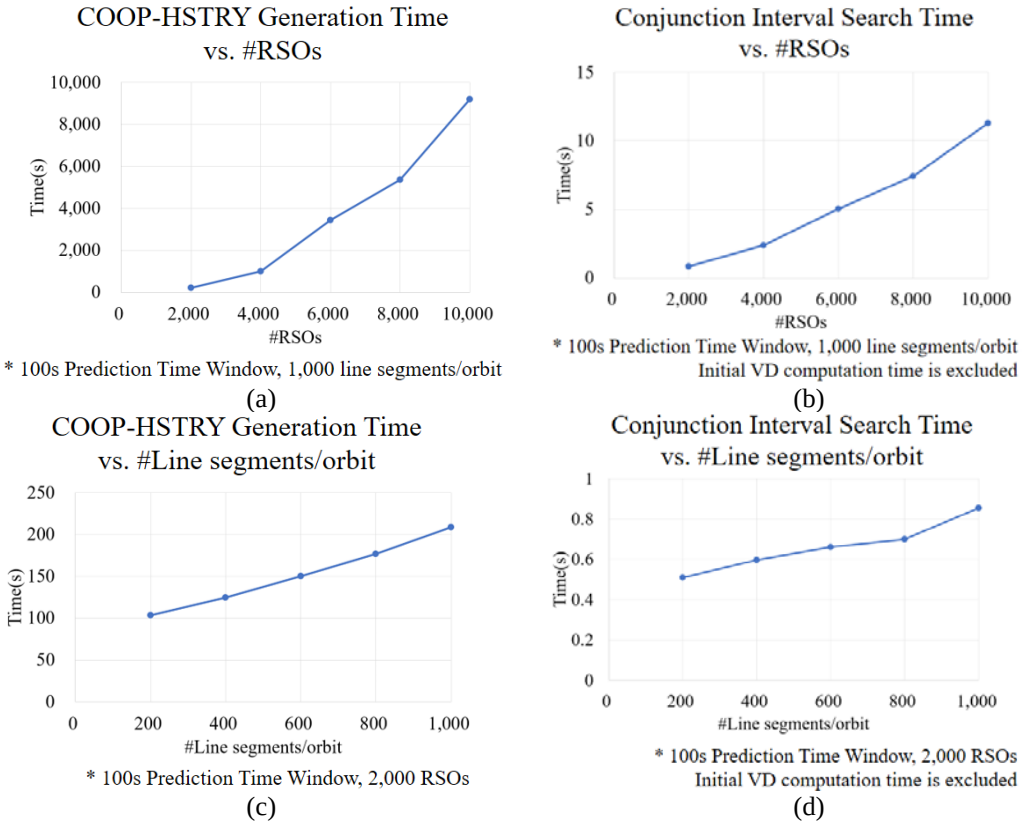



Fig. 27. Computation time graphs. (a) COOP-HSTRY generation time vs. #RSOs. (b) Conjunction Interval Search Time vs. # RSOs (c) COOP-HSTRY generation time vs. #Line segments/orbit. (d) Conjunction Interval Search Time vs. #Line segments/orbit

## 7.2. Verification and Validation: Maneuver Planning for Satellites

We have tested the maneuver path optimization for the Korean KOMPSAT satellite series against 1,000 RSOs downloaded on July 10, 2018 from the low earth orbit (LEO) NORAD TLE data. We predicted 10,000 seconds (about 3h) from 0:00 AM of July 12, 2018. In this experiment, we changed the angle of the plane containing the orbit of each satellite to generate hypothesized maneuver path: In practice, a maneuver hypothesis has **either a higher or a lower altitude than its original orbit**. This hypothesis generation might be rather simple to be realistic but is **used in this study to demonstrate the capability of the COOP2 for SSA and STM**.

The results are shown in Fig. 28. See Fig. 28(c): The green curve represents the orbit of KOMPSAT-4 (NORAD ID 40536) RSO shown as the green ball. Note that the ball radius is 100km. COOP2 reported the KOMPSAT-4 was swing by 38km away from the RSO 15505 after 4,742 seconds. See the zoom-up view shown as the box in the right: The green and red balls are very close. The blue curve and ball similarly represent the hypothesized path and the imaginary RSO. In the examples in Fig. 28, except Fig. 28(d) for 10 degrees, we rotate the planes containing the orbits 20 degrees about the line passing through the RSO position at the initial location at $t = 0$ and the center of the earth. Running the MPO-problem with one hypothesized maneuver path of the given rotation angle, we find the TCA situation predicted by the CIS-problem is resolved. The blue balls represent the imaginary RSOs at the identical time with the green RSOs which were at TCA situations. Recall that we currently solve this problem using replicas: This is the reason why the blue ball in Fig. 28(b) is off the blue curve in the zoom-up. This observation does not preclude the existence of a TCA of the imaginary RSO with another RSO. In this example of Fig. 28(c), the solving the CIS-problem of the hypothesized path against all RSOs yields a new TCA with the distance 76km with RSO 19256 in a certain time within the prediction time window. Hence, we can conclude from this computational experiment that the imaginary RSO following the hypothesized path is safer than the original. The other figures show other KOMPSAT series satellites: KOMPSAT-2 (Fig. 28(a)); KOMPSAT-3 (Fig. 28(b)), and KOPSAT-5 (Fig. 28(d)).

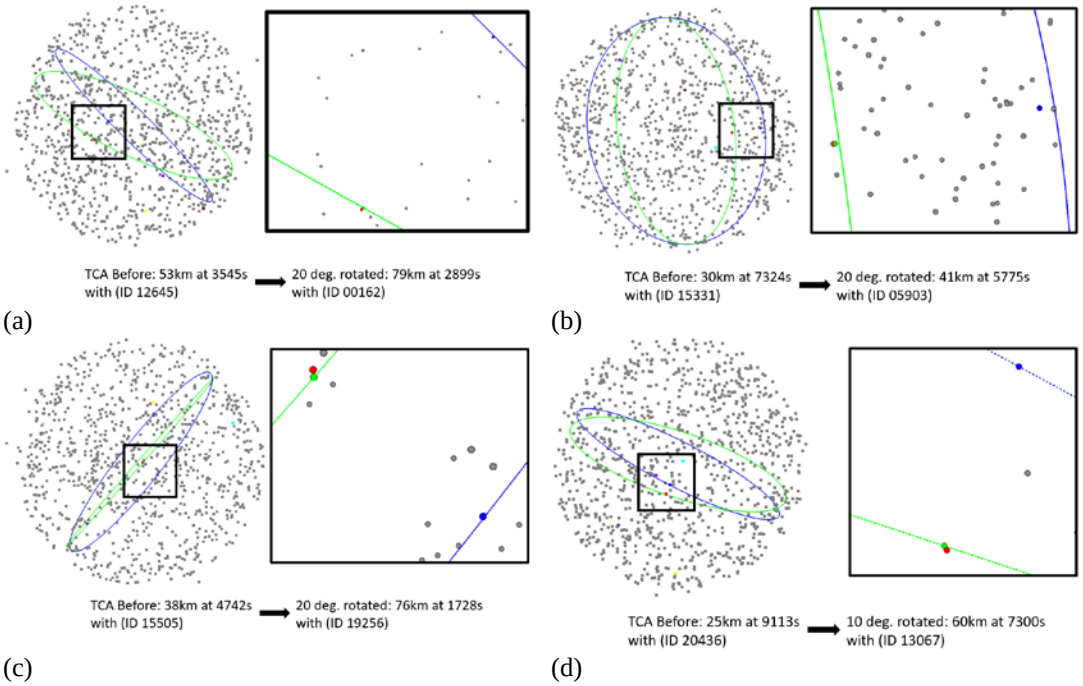



Fig. 28. MPO-problems for four satellites in the KOMPSAT series. The green curve is the orbit of the satellite shown as the green ball. The red ball represents the secondary RSO at the TCA. The blue curve represents a hypothesized path of an imaginary RSO shown as the blue ball. (a) KOMPSAT-2 (NORAD ID 29268). (b) KOMPSAT-3 (NORAD ID 38338). (c) KOMPSAT-4 (NORAD ID 40536). (d) KOMPSAT-5 (NORAD ID 39227).

### 7.3. Report of the Developed Web Server COOP2

We have implemented COOP2 algorithm as a web server (address: http://voronoi.hanyang.ac.kr/coop2) so that anyone connects to COOP2 server and tests the functionalities of COOP2. Fig. 29 shows the overview of COOP2 web server. COOP2 downloads TLE data from JSpOC catalogue and updates COOP-HSTRY data for the prediction time window.

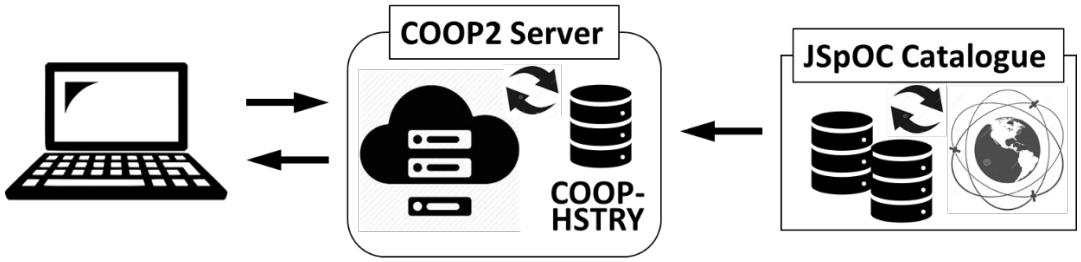



Fig. 29. Overview of COOP2 Web Server

Currently, COOP2 web server provides the answers for the following three queries in conjunction assessment.

- Query 1. Find all proximal events in time horizon.
- Query 2. Find proximal events for an object-of-interest.
- Query 3. Find optimal maneuver plan for an object-of-interest.

where

1) Object-of-interest (OOI): The object that we are investigating, e.g. a particular satellite,
2) Proximal event: An event that two objects (including OOI) are closer than a predefined threshold distance, and
3) Optimal maneuver plan: The best maneuver path among possible variations of the current path of OOI. In current implementation, we modify angle of the plane containing the orbit. Here the angle is defined by the plane which passes through the center of Earth. In this experimental program, we enumerate the angle by +- 1 degree from the current plane.

Running environment of COOP2 is as follows.

1) Hardware: Intel Xeon 6132 Gold CPU(14 Cores), 196G RAM, Nvidia Tesla V100 GPU (5,120 cuda cores, 16G VRAM)
2) Software: NVIDIA® NSight™ Eclipse Edition on Ubuntu 18.04 LTS

Fig.30 shows the user interface of COOP2 server. Fig.31. and Fig.32. show the examples of the summary web pages for computed results. Fig.31 shows the result of Query 1 which reports all proximal events. Finding all proximal event within 24h time window for 300 objects takes 39 seconds. Query 2 takes the object id and reports the proximal events for a satellite that we are interested in. Fig.32 shows the result for Query 2. Finding all proximal events that object with ID 00446 involved with the same condition takes about 7.5 seconds. Query 3 takes both the object id and the moment-of-maneuver. Then COOP2 finds optimal maneuver path by evaluating 20 alternatives. The alternatives are generated within a prediction time window (1 h). Note: Maneuver path modification occurs instantaneously (without a smooth transition).

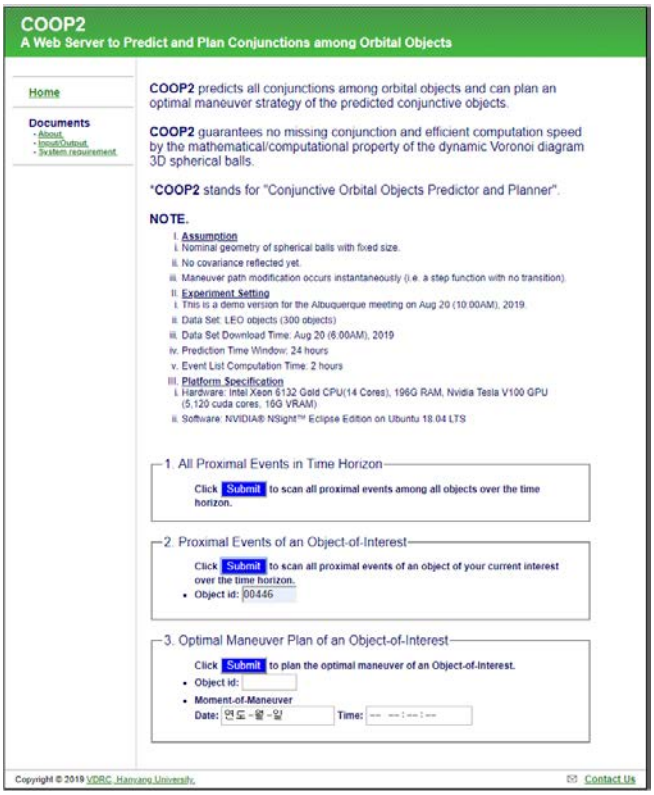


Fig.30. The main interface of COOP2 webserver.

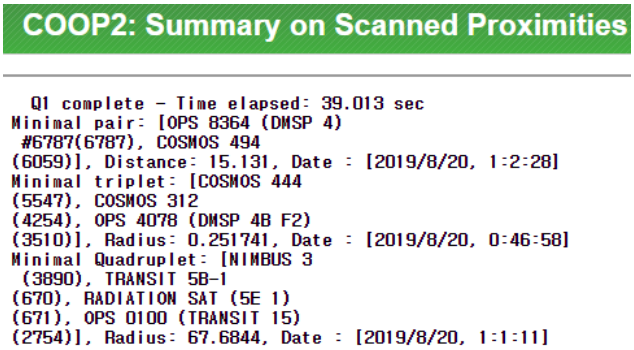


Fig.31. The summary page for computed result of COOP2 web server for Query 1.

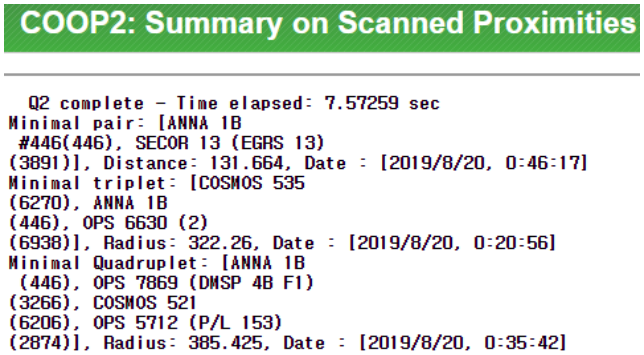


Fig.32. The summary page for computed result of COOP2 web server for Query 2.

## 7.4. Idea Validation using Drones

COOP2 (Conjunctive Orbital Object Predictor and Planner) algorithm can be **a vital tool for collision-free path planning of moving objects in space**. One of the key technical challenges of **choreographing a big swarm of drones**, such as the opening ceremony of the 2018 Winter Olympic in Pyeongchang, Korea, is to avoid a collision between drones. Fig. 33 shows collision-free path planning among five moving disks in the plane. For presentation convenience, the disks are of the same size, assuming the same level of location uncertainty, and the path of each disk is assumed to be linear. The five disks are vertically located at $t_0$ on the left, and each disk moves to the location corresponding to $t_\infty$ on the right. Each disk moves linearly with a constant initial speed at $t_0$ given by the line segment between the two points at $t_0$ and $t_\infty$ and the time difference $t_\infty - t_0$. Suppose that the disks are $d_*$, $d_a$, $d_b$, $d_c$, and $d_d$ from bottom to top at $t_0$ and their speeds are $v_*$, $v_a$, $v_b$, $v_c$, and $v_d$, respectively. Let **$VD(t_0)$ be the Voronoi diagram of the five disks at $t_0$**. The disks are configured so that **$d_*$ contacts $d_a$ at $t_a > t_0$**, which **can be predicted from $VD(t_0)$**. Hence, we **change the speeds of both $d_*$ and $d_a$ at $t_a^- < t_a$ respectively to $v_*$' and $v_a$' so that the predicted collision can be avoided**. The red and green colors of $d_*$ and $d_a$ denote speed changes. Then, after the collision is avoided at $t_a^+ > t_a$, we change the speeds of the two disks once more, $v_*$'' and $v_a$'', so that the original goal of simultaneous arrival is met. Note that there are different strategies to decide $t_a - t_a^-$, $t_a^+ - t_a$, $v_*$', $v_a$', $v_*$'', and $v_a$'' where all can be easily calculated. In the figure, the red and green colors correspond to the changed and unchanged speeds, respectively, while the path geometries remain linear. In fact, the collisions are avoided by slowing down the speeds of the red disks.

After the first collision is avoided, a second collision is predicted between $d_*$ and $d_b$ at $t_b$ from $VD(t_a^+)$. We change the speeds of both $d_*$ and $d_b$ at $t_b^- < t_b$ respectively to $v_*$''' and $v_b$' so that the second collision can be avoided and, at $t_b^+ > t_b$, we change the speeds of the two disks once more, say $v_*$''' and $v_b$'', to simultaneously arrive at the targets. Similar processes can avoid a third predicted collision between $d_*$ and $d_c$ and a fourth one between $d_*$ and $d_d$.

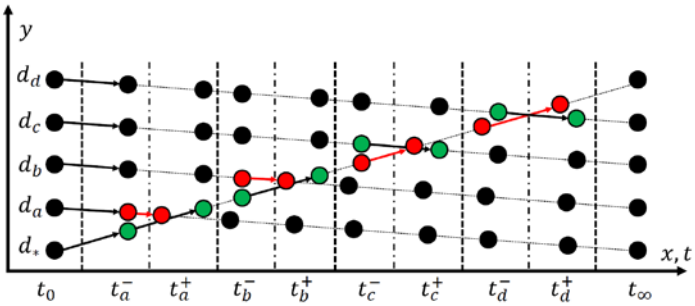



Fig.33. Collision avoiding path planning for a swarm of drones. Five drones are moving linearly through the dotted line segments during the period from $t_0$ to $t_\infty$. The predicted collision between $d_*$ and $d_a$ at $t_a$ is avoided by slowing down the speed of $d_a$ (red) while keeping the speed for $d_*$ (green) the same. The other predicted collisions at $t_b$, $t_c$, and $t_d$ can be similarly avoided.

**There could be a number of ways to avoid a predicted collision between two disks. The approach used above is to change the speeds of the disks while the geometry of their respective linear paths is preserved** and can be applied to copters with four, six, eight, or more wings. On the other hand, it is also possible to provide curved path(s) for one of either disk or both by taking advantage of the information available in the Voronoi diagram. This applies to those drones with fixed wings where a changed speed cannot be realized due to the aerodynamic constraints. We note that collision-avoidance may require a series of more complicated operations.

See Fig. 34(a). We want to **transform** twenty objects **from the configuration of a rectangle at $t_0$ to that of a circle at $t_\infty$**. Consider that the dotted line segments between the objects of the two configuration denote a 1-to-1 correspondence. We want the objects move linearly through the 1-to-1 mapping. Fig. 34(b) and (c) show the Voronoi diagrams of the objects in the two configurations. Fig. 35 shows some intermediate states of the transformation at $t_0 < t_a$, $t_b$, $t_c$, and $t_d < t_\infty$. Fig. 36 shows the step-by-step of Fig. 35. Note that **the red and green objects in each red ellipse denote that their speeds are modified to avoid anticipated collisions**.

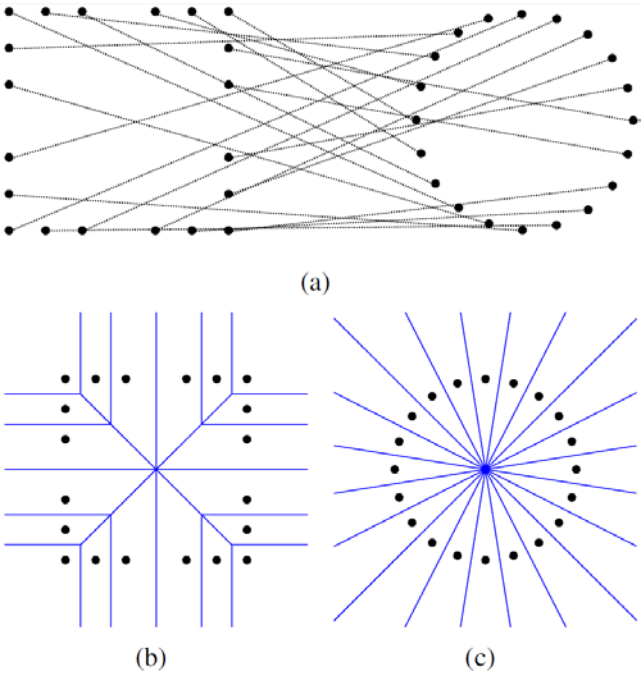



Fig. 34. (a) The tiny disks initially positioned on a square moves linearly through the black dotted line segments to arrive at the circle. (b) and (c) The Voronoi diagrams

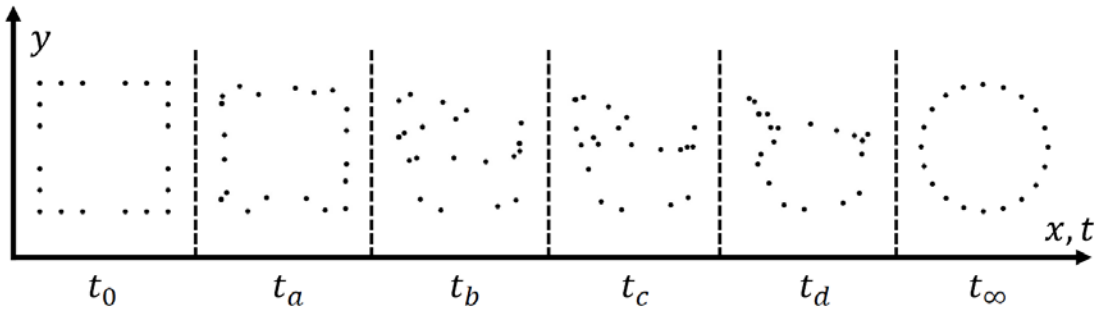



Fig. 35. Motion planning of 20 vehicles through collision-free paths. Each dot represents a linearly moving vehicle. The scene transforms from the initial scene (at $t_0$) to the final scene (at $t_\infty$) after going through intermediate scenes ($t_a$, $t_b$, $t_c$, and $t_d$). Each region bounded by a dotted vertical separator corresponds to the space where the vehicles are located at the corresponding moment. Hence, the horizontal axis corresponds to both time t and the x-coordinate.

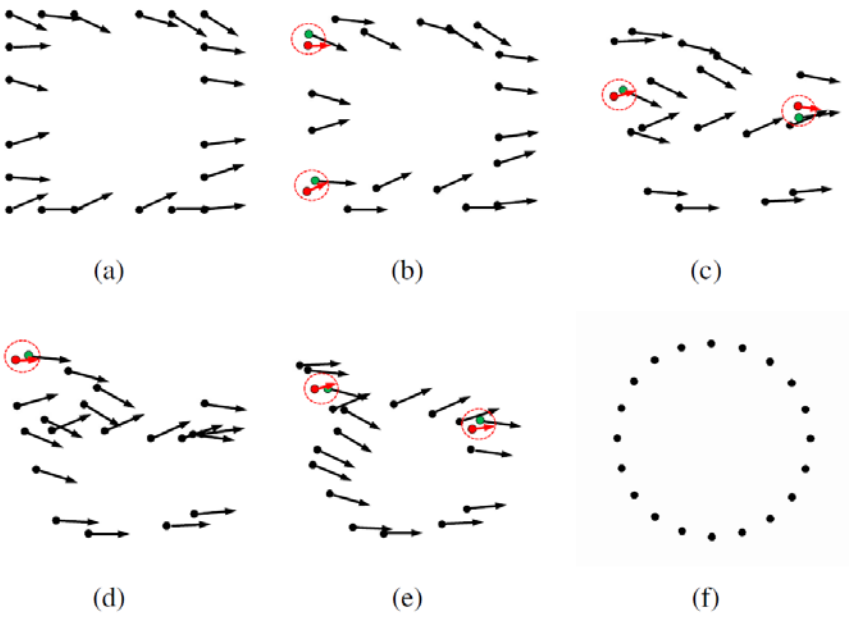



Fig. 36. Collision-free vehicle motion plan produced using a dynamic Voronoi diagram. The pair of red and green dots in each red ellipse is the vehicles that need to adjust speeds to avoid a predicted collision. (a) Initial scene ($t_0 = 0$), (b), (c), (d), (e) intermediate scenes ($t_a$, $t_b$, $t_c$, and $t_d$, resp.), (f) Final scene ($t_\infty$).

**(Drone Swarming)** For the validation of COOP2 algorithm, we demonstrated drone swarming using 58 drones. To this end, we computed the collision-free flight path of each drone. Fig.37 (a) and (b) shows the snapshots of two scenes from the demonstration: an airplane and taegeuk mark.


Fig. 37. Still images of drone swarming (58 drones, November 05 2018). (a) airplane. (b) taegeuk mark. (**YouTube link: https://www.youtube.com/watch?v=Ikqo3AXguKw&t=3s**)

**(Drone specification)** We used quadcopter drones for choreographing swarms of drone as shown in Fig. 38. Note that the GPS of a drone is low-cost/low-accuracy equipment: precision < 2.5 m, which makes it difficult to choreograph the drones delicately.

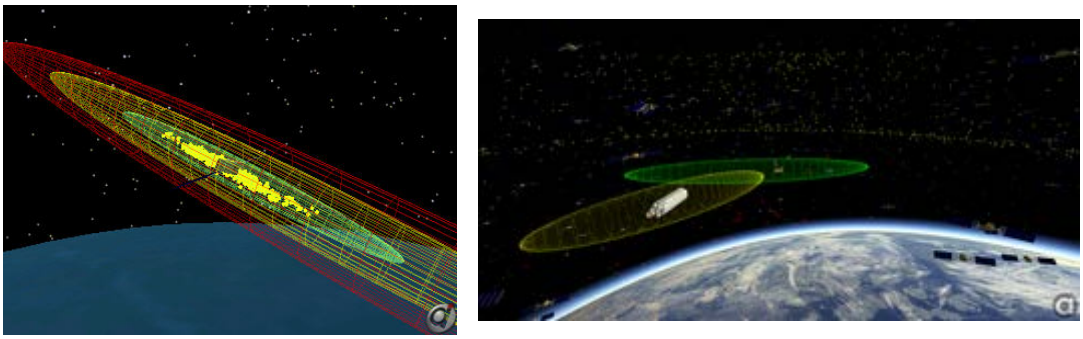
(a) (b) (c)

Fig 38. Quadcopter for drone swarming. (Size: 25cm X 25cm X 24cm; Total weight: 600g; Flight time: 11-12 min. (with hovering); Flight firmware: PX4 v1.8.2; Frame/PDB(power distribution boards): custom-made); GPS: Mini uBLOX NEO-M8N, precision: < 2.5m; Electric Speed Control (ESC): Littlebee Pro, current: 20A; Motor: EMAX MT2206 II, 1900KV; Propeller: Gemfan Prop 5040 Tri Blade; Battery: LiPo 3S, 2200mAh; Flight controller: Pixracer, CPU: 180MHz ARM Cortex, SRAM: 256KB; Wifi module: ESP8266, 2.4GHz; LED: Adafruit WS2812S) (a) Top view. (b) Side view. (c) Bottom view.

1. Basic information
   a. Size: 25cm X 25cm X 24cm
   b. Total weight: 600g
   c. Flight time: 11-12 min. (with hovering)
   d. Flight firmware: PX4 v1.8.2
   e. Frame/PDB(power distribution boards): custom-made
2. GPS
   a. Model: Mini uBLOX NEO-M8N
   b. Precision: < 2.5m
3. Electric Speed Control (ESC)
   a. Littlebee Pro
   b. Current: 20A
4. Motor
   a. EMAX MT2206 II, 1900KV
5. Propeller
   a. Gemfan Prop 5040 Tri Blade
6. Battery
   a. LiPo 3S
   b. 2200mAh
7. Flight controller
   a. Model: Pixracer
   b. CPU: 180MHz ARM Cortex
   c. SRAM: 256KB
8. Wifi module
   a. Model: ESP8266
   b. Frequency: 2.4GHz
9. LED
   a. Model: Adafruit WS2812S

## 8. Future Studies

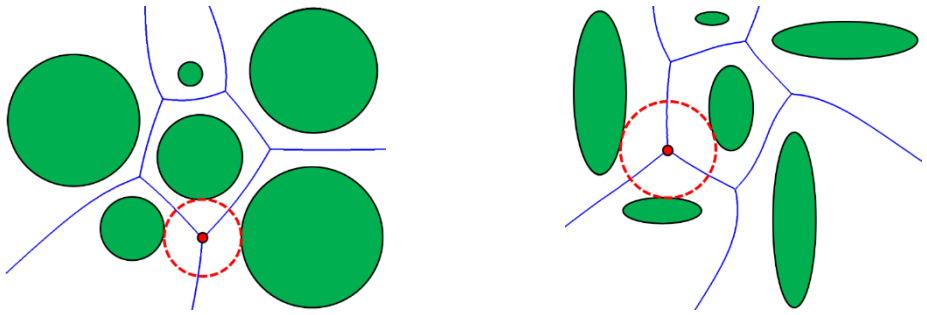
This study assumes that Resident Space Objects (RSOs) in orbital environment do not have any uncertainty in their locations (Note: The proposed COOP2 algorithm/program currently represents the locational uncertainty of an orbital object as a fixed-radius sphere.). Hence, the COOP2 algorithm should be enhanced in order to include realistic physics dominating the orbital motion of RSOs around Earth. Ellipsoidal covariance contours are used for realistic modeling of RSO's orbital movement where the ellipsoid changes its shape over time [66] (See Fig. 39).



(a) (b)

Fig. 39. location uncertainty as ellipsoidal covariance contours: (a) the ellipsoidal covariance contours for different probability (source: https://ai-solutions.com/_help_Files/orbitdetermination_smp.htm). (b) conjunction assessment using the covariance contours (source: https://en.wikipedia.org/wiki/FreeFlyer).

To properly model the ellipsoidal time-varying covariance and develop (accurate and efficient) algorithms to predict and best avoid conjunctions among orbital objects, a computational geometry theory for the dynamic Voronoi diagram of shape-changing 3D ellipsoidal objects should be developed and implemented. Fig. 40 shows an example of the Voronoi diagrams for circular disks and ellipses in 2D whose topologies are determined at the vertices. It is known that the Voronoi vertex (shown as the center of the red dotted circle) of (non-moving) 2D ellipses is a root of polynomial of degree higher than 150. PI and its research group of this project have endeavored to solve this problem and they successively completed a research project on 3D printing which uses the Voronoi diagram of ellipses and ellipsoids.



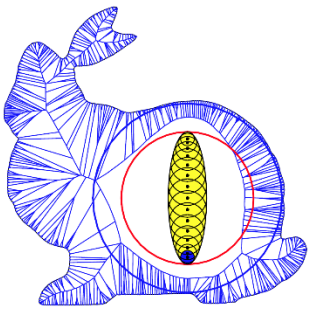
(a) (b)

Fig. 40. Voronoi diagrams in 2D space. The center of red circle corresponds to a Voronoi vertex and determines the Voronoi diagram topology. (a) Voronoi diagram of circular disks. (b) Voronoi diagram of ellipses.

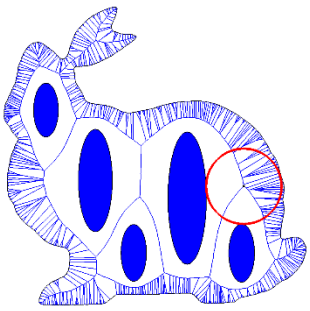
Suppose that we print a product using a 3D mesh model in Fig. 41(a). We could have a polygonal shape (a cross-sectional polygon) by intersecting the mesh model with a plane. Fig. 41(c) shows that ellipses are packed inside the polygonal shape using the Voronoi diagram of ellipses in a polygon. The idea was to exploit the Voronoi diagram of circular disks in a polygon where a set of disks approximates each ellipse (Fig. 41(b)). We project the packed ellipses onto other parallel cross sections and then create elliptical cylinders by extruding the ellipses (Fig. 41(d)). We applied this technique to eliminate the materials inside a 3D-printing product [67] (See Fig. 41(f)). This research output could result in the reduction of the printing time, product weight, energy consumption, and eventually cost.

Very recently, we developed a novel approach to hollowing with ellipsoidal voids (Fig. 41(e)) which uses the Voronoi diagram of ellipsoids [68]. The method for correct and efficient maintenance of the Voronoi diagram of moving and shape-changing 3D ellipsoids is much harder and is a challenge. Once we have the Voronoi diagram of moving and shape-changing 3D ellipsoids, the covariance propagation for uncertainty prediction will be included in COOP2 to improve the accuracy of the prediction. We believe that this future research work will substantially and significantly contribute to the SSA community.

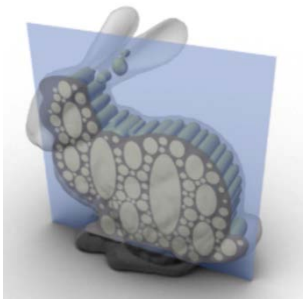
(a) intersection between 3D mesh model and plane



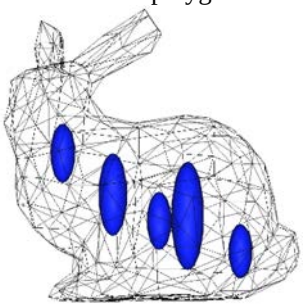
(b) Voronoi diagram an ellipse inside a polygon



(c) ellipse packing in a polygon



(d) creation of elliptical cylinders inside (a) using (c)



(e) creation of ellipsoidal voids using Voronoi diagram

(f) printed product

Fig. 41. Hollowing with elliptical voids for 3D printing

## 9. Conclusion

Collision avoidance is one of the critical tasks to keep geospace safe and efficient. This research studies the detection and resolution of conjunction for collision avoidance and develops and implements an algorithm to predict the conjunction based on the dynamic Voronoi diagram (DVD) of 3D spherical balls. We have successfully developed, implemented, and tested the DVD algorithm for spherical balls. Then we developed and implemented COOP2 (Conjunctive Orbital Objects Predictor and Planner) algorithm/program, based on the DVD algorithm, which showed a proven mathematical capability to detect all conjunctions without any missing case. We tested the COOP2 using TLE data obtained from Korea Aerospace Research Institute (KARI) which includes the orbital movements of the following five Korean satellites: KOMPSAT-2, KOMPSAT-3, KOMPSAT-3A, and KOMPSAT-5. Given a detected conjunction, The COOP2 algorithm/program can produce the optimal maneuver plan by quickly evaluating alternatives using the event history stored in the COOP-HSTRY file. We have validated the correctness and performance of COOP2 using **drone swarming**. We could use the COOP2 to generate the **collision-free flight path of each drone** so that the performance and correctness of the algorithm/program could be validated and verified. (For the demonstration of drone swarming, refer to Fig. 37 and **YouTube link: https://www.youtube.com/watch?v=Ikqo3AXguKw&t=3s**) We believe that the COOP2 algorithm/program will play a key role for Space Situational Awareness and Space Traffic Management.

## 10. References


[1] Space debris by the numbers, European Space Agency, January, 2019. http://www.esa.int/Safety_Security/Space_Debris/Space_debris_by_the_numbers

[2] "Исторические сведения "Истребитель спутников" – программа". Army.lv.

[3] "Сезон космической охоты (крылатые ракеты, противоспуниковая система ИС). Смотреть онлайн. История России" [Season of space hunting (cruise missiles, anti-satellite IP system). Watch online. Russian history]. Statehistory.ru.

[4] Tatman, Logan, et al., Orbital Debris Propagation in Solwind Anti-Satellite Event, AIAA Scitech 2019 Forum. 2019.

[5] History of On-Orbit Satellite Fragmentations (13th ed.), NASA Orbital Debris Program Office, May 2004. pp. 12, 136.

[6] History of On-Orbit Satellite Fragmentations (14th ed.), NASA Orbital Debris Program Office, May 2008. pp. 15, 144.

[7] Kelso, T. S, Analysis of the 2007 Chinese ASAT Test and the Impact of its Debris on the Space Environment, 8th Advanced Maui Optical and Space Surveillance Technologies Conference, Maui, HI, Vol. 7. 2007.

[8] "Chinese Anti-satellite Test Creates Most Severe Orbital Debris Cloud in History", *The Orbital Debris Quarterly News*, Vol. 11 (2), NASA JSC, April 2007.

[9] "ISS crew take to escape capsules in space junk alert", BBC, March 24, 2012. (https://www.bbc.com/news/science-environment-17497766)

[10] Data retrieved from the U.S. military's public satellite catalog maintained at "Space Track", (www.space-track.org)

[11] "Russia Flight Tests Anti-Satellite Missile", The Washington Free Beacon, December 2, 2015. (https://freebeacon.com/national-security/russia-conducts-successful-flight-test-of-anti-satellite-missile/)

[12] Chaudhury, Dipanjan Roy, "Explained: What's Mission Shakti and how was it executed?", The Economic Times, March 28, 2019 (https://economictimes.indiatimes.com/news/politics-and-nation/explained-whats-mission-shakti-and-how-was-it-executed/articleshow/68607473.cms?from=mdr)

[13] Chappell, Bill, "NASA: Debris From India's Anti-Satellite Test Raised Threat To Space Station", NPR, April 2, 2019 (https://www.npr.org/2019/04/02/709032198/nasa-debris-from-indias-anti-satellite-test-raised-threat-to-space-station)

[14] "Two Breakup Events Reported", The Orbital Debris Quarterly News, NASA JSC, August 2019 (https://orbitaldebris.jsc.nasa.gov/quarterly-news/pdfs/odqnv23i3.pdf).

[15] Radtke, J. et al. The impact of the increase in small satellite launch traffic on the long-term evolution of the space debris environment, In Proceedings of the 7th European Conference on Space Debris, 2017.

[16] Liou, J.C. and Johnson, N.L., Risks in Space from Orbiting Debris, *Science*, Vol. 311, 340-341, 2006.

[17] DRMOLA, Jakub; HUBIK, Tomas. Kessler syndrome: System dynamics model. *Space Policy*, Vol. 44, 29-39, 2018.

[18] https://medium.com/@leolabs_space/the-iras-ggse-4-close-approach-a99de19c1ed9.

[19] Peeters, W. From suborbital space tourism to commercial personal spaceflight. *Acta Astronautica*, 66(11-12), 1625-1632, 2010.

[20] Chang, Y. W. The first decade of commercial space tourism. *Acta Astronautica*, 108, 79-91, 2015.

[21] Johnson, N. L. Space traffic management concepts and practices. *Acta Astronautica*, 55(3-9), 803-809, 2004.

[22] Cukurtepe, H. and Akgun, I. Towards space traffic management system. *Acta Astronautica*, 65(5-6), 870-878, 2009

[23] Budianto-Ho, I.A. et al, Scalable Conjunction Processing using Spatiotemporally Indexed Ephemeris Data, In Advanced Maui Optical and Space Surveillance Technologies Conference, 2014.

[24] Alarcón-Rodríguez, J.R. et al, Conjunction event predictions for operational ESA satellites, In Proceedings of the Space-Ops 2004 Conference, 2004.

[25] Alarcón-Rodrıguez, J.R. et al, Development of a collision risk assessment tool, *Advances in Space Research*, Vol. 34, 1120-1124, 2004.
[26] Casanova, D. et al, Space Debris Collision Avoidance Using a Three-filter Sequence, *Monthly Notices of the Royal Astronomical Society*, Vol. 442, 3265-3242, 2014.
[27] Wisniowski, T. and Rickman, H., Fast Geometric Method for Calculating Accurate Minimum Orbit Intersection Distances, *Acta Astronomica*, Vol. 63, 293-307, 2013.
[28] Chen, H. et al, Conformal Prediction for Anomaly Detection and Collision Alert in Space Surveillance, In Proceedings of SPIE 8739, 2013.
[29] Alfano, S., Toroidal Path Filter for Orbital Conjunction Screening, *Celestial Mechanics and Dynamical Astronomy*, Vol. 113, 321-334, 2012.
[30] Crassidis, J. et al, Space Collision Avoidance, In Proceedings of National Symposium on Sensor and Data Fusion, 2011.
[31] Armellin, R. et al, Computing the Critical Points of the Distance Function between Two Keplerian Orbits via Rigorous Global Optimization, *Celestial Mechanics and Dynamical Astronomy* Vol. 107, 377-395, 2010.
[32] Woodburn, J. et al, A Description of Filters for Minimizing the Time Required for Orbital Conjunction Computations, *Advances in the Astronautical Sciences*, Vol. 135, 1157-1173, 2009.
[33] Healy, L.M., Close Conjunction Detection on Parallel Computer, *Journal of Guidance, Control, and Dynamics*, Vol. 18, 824-829, 1995.
[34] Coppola, V. and Woodburn, J., Determination of Close Approaches Based on Ellipsoidal Threat Volumes, *Advances in the Astronomical Sciences: Spaceflight Mechanics*, Vol. 102 1013-1024, 1999.
[35] Carlson, R. and Lee, J., Detecting Near Collisions for Satellites, *IEEE transactions on aerospace and electronic systems*, Vol. 33, 921-929, 1997.
[36] Hoots, F.R. et al, An Analytic Method to Determine Future Close Approaches between Satellites, *Celestial mechanics*, Vol. 33, 143-158, 1984.
[37] Kubica, J. et al, Efficient Intra-and Inter-night Linking of Asteroid Detections Using Kd-trees, *Icarus*, Vol. 189, 151-168, 2007.
[38] Kouprianov, V., ISON Data Acquisition and Analysis Software. In 6th European Conference on Space Debris, 2013.
[39] Denneau, L. et al, The Pan-STARRS Moving Object Processing System. *Publications of the Astronomical Society of the Pacific*, Vol. 125, 357, 2013
[40] Mercurio, M. et al, A Hierarchical Tree Code Based Approach For Efficient Conjunction Analysis, In AIAA/AAS Astrodynamics Specialist Conference, 2012.
[41] Bentley, J.L., Multidimensional Binary Search Trees Used for Associative Searching., *Communications of the ACM*, Vol. 18, 509-517, 1975.
[42] Alarcón-Rodriguez, J.R. et al, Collision Risk Assessment With a "Smart Sieve" Method, In *Joint ESA-NASA Space-Flight Safety Conference*, 2002.
[43] Okabe, A. et al, *Spatial Tessellations - Concepts and Applications of Voronoi Diagram*, John Wiley & Sons, 1992.
[44] Kim, D.-S. et al, Voronoi Diagram of a Circle Set from Voronoi Diagram of a Point Set: I. Topology, *Computer Aided Geometric Design*, Vol. 18, 541-562, 2001.
[45] Kim, D.-S. et al, Voronoi Diagram of a Circle Set from Voronoi Diagram of a Point Set: II. Geometry, *Computer Aided Geometric Design*, Vol. 18, 563-585, 2001.
[46] Kim, D.-S. et al, Euclidean Voronoi diagram of 3D Balls and Its Computation Via Tracing Edges, *Computer-Aided Design*, Vol. 37, 1412-1424, 2005.
[47] Kim, D. and Kim, D.-S., Region-Expansion for the Voronoi Diagram of 3D Spheres, *Computer-Aided Design*, Vol. 38, 417-430, 2006.
[48] Kim, D.-S. et al, Quasi-Triangulation and Interworld Data Structure in Three Dimensions, *Computer-Aided Design*, Vol. 38, 808-819, 2006.
[49] Kim, D.-S. et al, Three-Dimensional Beta Shapes, *Computer-Aided Design*, Vol. 38, 1179-1191, 2006.
[50] Kim, D.-S. et al, Quasi-worlds and Quasi-operators on Quasi-triangulations, *Computer-Aided Design*, Vol. 42, 874-888, 2010.
[51] Kim, D.-S. et al, Three-dimensional Beta-shapes and Beta-complexes via Quasi-triangulation, *Computer-Aided Design*, Vol. 42, 911-929, 2010.

[52] Kim, D.-S. et al, Querying Simplexes in Quasi-triangulation, *Computer-Aided Design*, Vol. 44, 85-98, 2012.
[53] Kim, D.-S. et al, QTF: Quasi-triangulation File Format, *Computer-Aided Design*, Vol. 44, 835-845, 2012.
[54] Lee, M. et al, Topology-oriented Incremental Algorithm for the Robust Construction of the Voronoi Diagrams of Disks, *ACM Transactions on Mathematical Software*, Vol. 43, 14, 2016.
[55] Kim, D.-S. et al, Euclidean Voronoi Diagrams of 3D Spheres: Their Construction and Related Problems from Biochemistry, *Lecture Notes in Computer Science*, Vol. 3604, 255-271, 2005.
[56] Kim, D.-S. et al, Crystal Structure Extraction in Materials using Euclidean Voronoi Diagram and Angular Distributions among Atoms, *Journal of Ceramic Processing Research*, Vol. 6, 63-67, 2005.
[57] Ryu, J. et al, Molecular Surfaces on Proteins via Beta Shapes, *Computer-Aided Design*, Vol. 39, 1042-1057, 2007.
[58] Kim, D.-S. et al, BetaDock: Shape-priority Docking Method Based on Beta-complex, *Journal of Biomolecular Structure & Dynamics*, Vol. 29, 219-242, 2011.
[59] Kim, D.-S. et al, Beta-decomposition for the Volume and Area of the Union of Three-dimensional Balls and Their Offsets, *Journal of Computational Chemistry*, Vol. 33, 1252-1273, 2012.
[60] Shin, W.-H. et al, GalaxyDock2: Protein-Ligand Docking using Beta-Complex and Global Optimization, *Journal of Computational Chemistry*, Vol. 34, 2647-2656, 2013.
[61] Kim, J.-K. et al, BetaCavityWeb: a Webserver for Molecular Voids and Channels, *Nucleic Acids Research*, Vol. 32, W413-W418, 2015.
[62] Park, C.H. et al, Nanocrack-Regulated Self-Humidifying Membranes, *Nature*, Vol. 532, 480-496, 2016
[63] Gavrilova, M. and Rokne, J., Swap Conditions for Dynamic Voronoi Diagrams for Circles and Line Segments, *Computer Aided Geometric Design*, Vol. 16, 89-106, 1999.
[64] Chanyoung Song. Jehyun Cha, Mokwon Lee, Deok-Soo Kim, Dynamic Voronoi Diagram for Moving Disks, *IEEE Transactions on Visualization and Computer Graphics*, DOI: 10.1109/TVCG.2019.2959321, 2019.
[65] Gavrilova, M. and Rokne, J., Updating the Topology of the Dynamic Voronoi Diagram for Spheres in Euclidean D-dimensional Space, *Computer Aided Geometric Design*, Vol. 20, 231-242, 2003.
[66] STK: defining threat volume. See http://help.agi.com/stk/index.htm#cat/Cat03-01.htm.
[67] Mokwon Lee, Qing Fang, Youngsong Cho, Joonghyun Ryu, Ligang Liu, and Deok-Soo Kim, Support-free hollowing for 3D printing via Voronoi diagram of ellipses, *Computer-Aided Design*, 101, pp.23-36, 2018.
[68] Seunghwan Choi, Mokwon Lee, Hyunwoo Kim, Jehyun Cha, Chanyoung Song, Joonghyun Ryu, and Deok-Soo Kim: Support-Free Hollowing with Spheroids and Efficient 3D-Printingwith Circular Printing Motions using Voronoi Diagrams, *Additive Manufacturing*, 2020 (under revision).

## 11. Appendix

**[A: Paper in peer-reviewed journal]**

A-1) Chanyoung Song. Jehyun Cha, Mokwon Lee, Deok-Soo Kim, Dynamic Voronoi Diagram for Moving Disks, *IEEE Transactions on Visualization and Computer Graphics*, DOI: 10.1109/TVCG.2019.2959321, 2019.

**[B: Papers in peer-reviewed conference proceedings]**

B-1) Jehyun Cha, Joonghyun Ryu, Mokwon Lee, Chanyoung Song, Youngsong Cho, Paul Schumacher, Misoon Mah, Deok-Soo Kim, DVD-COOP for Optimal Design of Maneuver Path for Conjunctive Objects, *Advanced Maui Optical and Space Surveillance Technologies Conference(AMOS)*, September 12-14, Wailea Marriot, Maui, Hawaii, USA, 2018.

B-2) Jehyun Cha, Drew McNeely, Joonghyun Ryu, Misoon Mah, Moriba Jah, Deok-Soo Kim, 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(AMOS)*, September 19-22, Wailea Marriot, Maui, Hawaii, USA, 2017.

**[C: Patents] (Refer to separate documents F-2 and F-3 below)**

C-1) Deok-Soo Kim, Jehyun Cha, Joonghyun Ryu, Mokwon Lee, Chanyoung Song, Youngsong Cho, Method for predicting collision and avoiding conflict between multiple moving bodies, USA (PCT application number: PCT/KR2018/006052, application number: 16618271)
C-2) Deok-Soo Kim, Jehyun Cha, Joonghyun Ryu, Mokwon Lee, Chanyoung Song, Youngsong Cho, Method for predicting collision and avoiding conflict between multiple moving bodies, Europe (PCT application number: PCT/KR2018/006052, application number: 18808957.7)
C-3) Deok-Soo Kim, Jehyun Cha, Joonghyun Ryu, Mokwon Lee, Chanyoung Song, Youngsong Cho, Method for predicting collision among moving objects, Republic of Korea (patent number: 10-1928190)
C-4) Deok-Soo Kim, Joonghyun Ryu, Jehyun Cha, Mokwon Lee, Chanyoung Song, Youngsong Cho, Method for predicting and avoiding collision and conflict/conjunction among moving objects, Republic of Korea (patent number: 10-2067363)

**[D: Interactions with industry or with Air Force Research Laboratory scientists or significant collaborations that resulted from this work]**

D-1) A whitepaper submitted to AFOSR/AFRL (Innovative (Pseudo) Realtime Conjunction Prediction and Planning using Voronoi Diagram of Covariances on VerSSA; Jointly proposed with Prof. Roberto Furfaro (robertof@email.arizona.edu) at University of Arizona, US; Submitted to Dr. Fariba Fahroo)
D-2) An invited presentation at AFOSR Computational Math Review, Arlington, VA, Aug 12-16, 2019 (Moving Particles Geometry Problem via Voronoi Diagrams, Invited by Dr. Fariba Fahroo)
D-3) An invited presentation at Kirtland AFB, NM, Aug 20, 2019 (Voronoi Diagram: Space Object Conjunction Assessment and Other Applications, Invited by Dr. Tom Cooley)

**[E: Attachments – PDF files for A, B, C, and D]**

E-1) A-1, B-1, B-2, C-1, C-2, C-3, C-4, D-1

**[F: Etc. – PDF files as separate documents]**

F-1) Fedral_Financial_Report_sf-425_17IOA050.PDF
F-2) Report_of_Inventions_and_Subcontracts_DD0882-1.PDF (for C-1 and C-2)
F-3) Report_of_Inventions_and_Subcontracts_DD0882-2.PDF (for C-3 and C-4)