Dynamic Voronoi Diagram for Moving Disks

Chanyoung Song, Jehyun Cha, Mokwon Lee, and Deok-Soo Kim

Abstract—Voronoi diagrams are powerful for understanding spatial properties. However, few reports have been made for moving generators despite their important applications. We present a topology-oriented event-increment (TOI-E) algorithm for constructing a Voronoi diagram of moving circular disks in the plane over the time horizon $[0, t^\infty)$. The proposed TOI-E algorithm computes the event history of the Voronoi diagram over the entire time horizon in $O(k_F \log n + k_C n \log n)$ time with $O(n \log n)$ preprocessing time and $O(n + k_F + k_C)$ memory for n disk generators, k_F edge flips, and k_C disk collisions during the time horizon. Given an event history, the Voronoi diagram of an arbitrary moment $t^* < t^\infty$ can be constructed in $O(k^* + n)$ time where k^* represents the number of events in $[0, t^*)$. An example of the collision avoidance problem among moving disks is given by predicting future conjunctions among the disks using the proposed algorithm. Dynamic Voronoi diagrams will be very useful as a platform for the planning and management of the traffics of unmanned vehicles such as cars on street, vessels on surface, drones and airplanes in air, and satellites in geospace.

Index Terms—Unmanned vehicles, moving vehicles, path planning, collision avoidance, topology event, weighted Voronoi diagram

1 Introduction

The opening ceremony of the 2018 Winter Olympic at Pyeongchang, Korea, witnessed a record-breaking light show of 1,218 Shooting Star drones by Intel Co. (Fig. 1a). Starting with 50 drones (2012; a joint effort of Austrian Ars Electronics Futurelab and German Ascending Technologies Co.; a football field in Krailling, Germany), drone swarms have evolved through a series of record breaking events: 100 (November, 2015; Flugplatz Ahrenlohe in Tornesch, Germany; Intel Co.), 500 (November 4, 2016; Krailling, Germany; Intel Co.), and 1,180 (December, 2017; Guangzhou, China; eHang Co.). The eHang Co. showed 1,374 drones on May 1, 2018 at Xian, China (Fig. 1b) and Intel with 2,018 drones on July 15, 2018 at Folsom, California (Fig. 1c). One of the key challenges of choreographing such a big swarm of drones is to plan collision-free flight paths between any pair of flying drones where location uncertainty exists for each drone.

Most motion scenes in drone swarms were not very dynamic but static. This might be because the computational theory for predicting and avoiding collisions among moving objects has not been sufficiently developed. Such a theory is becoming increasingly important not only for drone events but also in applications such as satellites in geospace [1], [2], [3], [4], airplanes in midair [5], [6], [7], surface vessels, and self-driving cars where collision detection and resolution is critical. We have recently choreographed a swarm of 58 drones (equipped with low-cost/low-accuracy GPS-sensors) where the collision-free dynamic paths were generated by the algorithm of the dynamic Voronoi diagram presented herein. Fig. 1d shows the snapshots of a small swarm of 19 drones which displays two initials "H" and "Y" of HanYang University: The collision-free paths between the two initials were generated by the proposed algorithm.

Voronoi diagrams are well-known for their powerful properties for efficiently answering to spatial queries among generators and have been extensively studied for various generator types such as ordinary points, circular disks in the plane, and spherical balls in the three-dimensional space. However, most studies were conducted for static generators in that their sizes and positions were fixed. Relatively few studies were reported on the construction of Voronoi diagrams of moving generators despite their potential in a wide array of potential applications. Dynamic Voronoi diagrams have applications among moving vehicles at high speeds as they can be used to efficiently solve collision avoidance problems among moving objects by predicting the future conjunctions among the particles.

This paper presents the data structure and algorithm for constructing a Voronoi diagram of moving circular disks in the plane. We call it the *dynamic Voronoi diagram of (moving) disks*. The principle idea of the proposed scheme is topology-oriented and event-increment (thus abbreviated as the TOI-E algorithm) and both solution accuracy and computational efficiency are guaranteed.

Let $D = \{d_1, d_2, \ldots, d_n\}$ be the set of mutually disjoint circular disks $d_i = (c_i, r_i)$ in the plane where c_i and r_i are its center and radius, respectively. Let $\mathscr{VD}^S(D)$ be the Voronoi diagram of the static disks in D. $\mathscr{VD}^S(D)$ is defined as the set of Voronoi cells where the Voronoi cell of d_i is defined as $\mathscr{VC}(d_i) = \{x \in \mathbb{R}^2 \mid dist(x, c_i) - r_i \leq dist(x, c_j) - r_j,\ i \neq j\}$ where $dist(x, y)$ is the Euclidean distance between x and y. Then, $\mathscr{VD}^S(D) = \{\mathscr{VC}(d_1), \mathscr{VC}(d_2), \ldots, \mathscr{VC}(d_n)\}$ [8]. $\mathscr{VD}^S(D)$

- C. Song, J. Cha, and M. Lee are with the School of Mechanical Engineering, Hanyang Univeristy, Seoul 04763, South Korea.
E-mail: {cysong.vdrc, jhcha.vdrc, mwlee.vdrc}@gmail.com.
- D.-S. Kim is with the Voronoi Diagram Research Center and HYU-HPSTAR-CIS Global High Pressure Research Center, and School of Mechanical Engineering, Hanyang Univeristy, Seoul 04763, South Korea.
E-mail: dskim@hanyang.ac.kr.

*Manuscript received 8 Dec. 2018; revised 4 Dec. 2019; accepted 9 Dec. 2019. Date of publication 16 Dec. 2019; date of current version 29 Apr. 2021.
(Corresponding author: Deok-Soo Kim.)
Recommended for acceptance by R. Zhang.
Digital Object Identifier no. 10.1109/TVCG.2019.2959321*

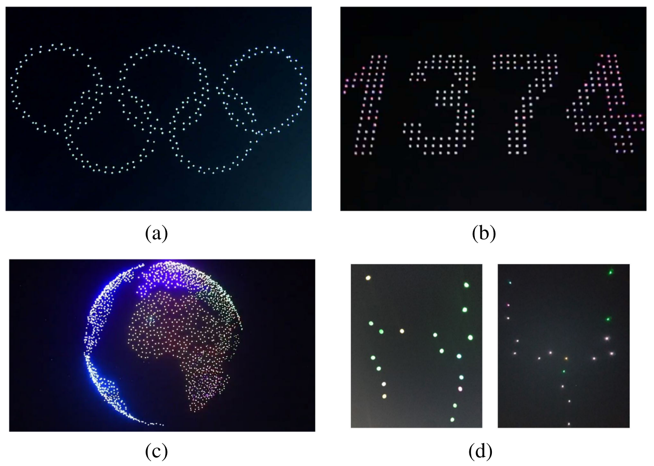


Fig. 1. Drone swarms. (a) 1,218 drones at Pyeongchang Winter Olympics, Korea (February 7, 2018; Intel Co.). (b) 1,374 drones at Xian, China (May 1, 2018; China; eHANG Co.). (c) 2,018 drones at Folsom, California, USA (July 15, 2018; Intel Co.). (d) 19 drones at Hanyang University, Seoul, Korea (July 18, 2018; Voronoi Diagram Research Center; Low-cost/low-precision GPS sensors were used.).

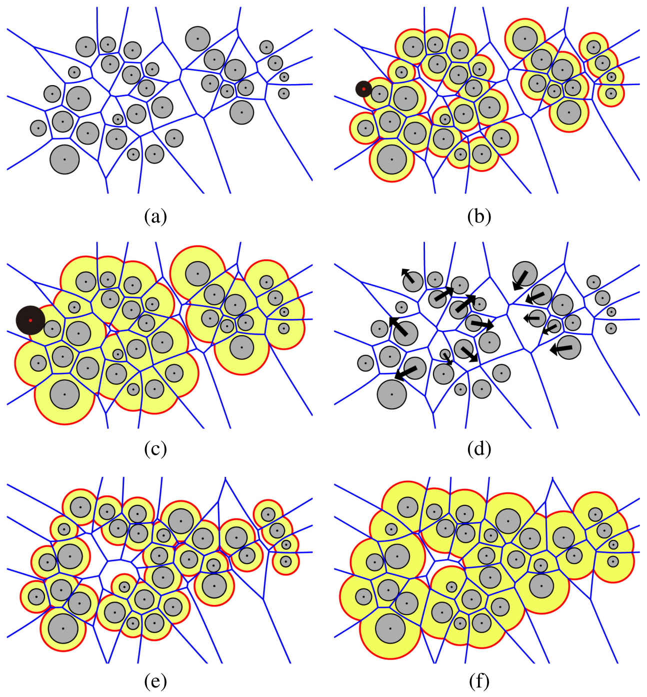


Fig. 2. Efficient recognition of the offsets, voids, and tunnels. (a) Disks and the Voronoi diagram at time $t_0 = 0$. (b) The offset corresponding to the probe π_1 with the radius r_1 at t_0 (Two voids and two tunnels exist). (c) The offset for π_2 with the radius $r_2 > r_1$ at time t_0 (The smaller void and narrower tunnel disappeared). (d) The disks with the velocities represented by the arrows at time t_0. (e) The offset corresponding to r_1 at time $t_1 = t_0 + \Delta t$ after the disks move (Two voids and one tunnel). (f) The offset corresponding to $r_2 > r_1$ at time t_1 (One void and the tunnel disappeared).

consists of $V^V = \{v_1^V, v_2^V, \ldots\}$, $E^V = \{e_1^V, e_2^V, \ldots\}$, and $C^V = \{c_1^V, c_2^V, \ldots c_n^V\}$, which denotes the sets of vertices, edges, and cells in the Voronoi diagram, respectively. $\mathscr{VD}^S(D)$ is usually stored in a winged-edge data structure or a variation, requiring $O(n)$ memory [9]. $\mathscr{VD}^S(D)$ can be constructed in $O(n\log n)$ time in the worst case but $O(n)$ on average. The distance equality between two disks defines an edge, which is a hyperbolic arc and that among three disks defines a vertex. If all the disks are of an equal size, $\mathscr{VD}^S(D)$ is identical to the ordinary Voronoi diagram of disk centers. Note that the Voronoi diagram of disks and the ordinary Voronoi diagram of points possess common properties but may also have significantly different properties. The Voronoi diagram of static disks will be termed as "Voronoi diagram" and several algorithms are known [10], [11], [12], [13].

Suppose that $d_i \in D$ is associated with a linear velocity $v_i = (v_i^x, v_i^y)$ with a constant speed. Hence, the location of d_i at $t \geq 0$ is given by $c_i(t) = c_i + v_i t,\ i = 1, 2, \ldots, n$. When $\|c_i - c_j\| = r_i + r_j$, d_i and d_j collide and bounce with modified velocities according to physical properties such as the coefficient of restitution or elasticity. See Appendix 1, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2019.2959321, for the details of collisions: i) The prediction of collision moment, and ii) the velocity vector changes after a collision (i.e., the changes in both directions and speeds of two disks). Our goal is to answer spatial queries among disks at an arbitrary moment t in the time horizon $[0, t^\infty)$ as efficiently and robustly as possible. If we can construct a static Voronoi diagram of disks at t, spatial queries can be efficiently answered. This requires the prediction of disk locations at t as well. Note that a deadlock situation does not occur among disks with unit coefficients of restitution because they are mutually exclusive and are associated with velocity vectors.

This is well-explained in Fig. 2. Fig. 2a shows a disk set D and its Voronoi diagram $\mathscr{VD}^S(D, t_0 = 0)$. Suppose that the Voronoi diagram has $m = O(n)$ V-edges. The red curve in Fig. 2b is the offset of the disks at t_0 by the radius r_1 of the black circular probe π_1, i.e., the locus of the center of π_1 touching at least one disk in D. Note that the intersection between two circular arcs of the offset curve occurs precisely on the Voronoi edges regardless of the radius of the probe. Hence, the offset can be correctly computed in $O(m)$ time by scanning the Voronoi edges. Note that it can be done more efficiently in $O(\log m + k)$ time where k denotes the number of intersections between offset arcs. This implies that the two voids in the biggest disk cluster and the two tunnels between the disk clusters can also be simultaneously computed at this time. The offset for the bigger probe π_2 with a radius $r_2 > r_1$ in Fig. 2c can also be computed using the same Voronoi diagram. The difference between Fig. 2b and 2c can be easily detected by the "offset-invariance property" of the Voronoi diagram and many related questions can be easily answered. For example, which void and tunnel have disappeared and what is the minimum probe radius that will block the existing tunnel.

The arrows in Fig. 2d denote the velocities of the disks with the speed represented by the arrow size. Fig. 2e and 2f show the moved disks together with their Voronoi diagrams at $t_1 = t_0 + \Delta t$ and their offsets for π_1 and π_2, respectively. It is clear that the offset of the moved disks can be efficiently computed once the Voronoi diagram corresponding to the moved disks is constructed.

In this paper, we discuss the construction of the dynamic Voronoi diagram $\mathscr{VD}(D)$ of moving disks in D over a time horizon that facilitates efficient instantiation of the static Voronoi diagram $\mathscr{VD}^S(D, t)$ of the moved disks at t. A naive approach might be to construct the entire static Voronoi diagram after updating disk locations at t. We avoid this *time-increment approach* for two reasons. First, a blind update of the disk locations after a constant time increment Δt might miss important events such as a collision between two disks because colliding moments tend to be irregularly spaced over time. An effective time increment to capture the moments of all the collisions over the entire time horizon would be computationally expensive and hard to choose. Second, it is not easy to relate the equivalent Voronoi entities in the Voronoi diagrams of two consecutive moments if they are independently constructed: It is necessary to perform a file inversion to answer the question "Which V-edge of the Voronoi diagram at t_i corresponds to which V-edge of the Voronoi diagram at t_{i-1}?". Relating Voronoi entities is critical for tracking geometric features of interest that are evolving over time.

An *event-based approach* progresses the time horizon by incrementing the time corresponding to the moments of significant events for the maintenance of a Voronoi diagram. The events include edge-flips and disk-collisions. Hence, each time increment is of a variable length. This is because we want to have the Voronoi entities in the initial Voronoi diagram at $t = 0$ survive as much as possible in the Voronoi diagram at t^∞ by minimizing the creation and/or deletion of Voronoi entities.

Suppose that the dynamic Voronoi diagram $\mathscr{VD}(D, t_i)$ at t_i is available and assume that the event of the most immediate future is known to occur at $t_{i+1} = t_i + \Delta t_i$. There might be two methods to construct $\mathscr{VD}(D, t_{i+1})$ in the event-based approach. The first is the *exhaustive delete-and-insert method* as follows. Choose a disk and delete it from $\mathscr{VD}(D, t_i)$ and insert it to the predicted location at t_{i+1} in $\mathscr{VD}(D, t_i)$. Deletion from and insertion to a Voronoi diagram implies that the topology structure is correctly maintained without and with the disk in the Voronoi diagram, respectively. Then, repeating the delete-and-insert operation for all disks in D transforms $\mathscr{VD}(D, t_i)$ to $\mathscr{VD}(D, t_{i+1})$. We avoid this method for two reasons. First, it is computationally expensive in that, for each time increment, the update takes $O(n^2)$ time because each delete-and-insert operation may take $O(n)$ time. Second, this approach dynamically deletes and creates Voronoi entities during program execution and thus it is not convenient to relate the Voronoi entities of two consecutive Voronoi diagrams in the time horizon.

We present a more efficient, robust, and useful method, called the *topology-oriented event-increment (TOI-E) method* which requires $O(\log n)$ time edge-flip operation for each time increment. The method provides a dynamic Voronoi diagram $\mathscr{VD}(D)$ as a data structure together with an accompanying algorithm. In due course, the algorithm also detects the moments of disk collision which can be used to predict disk locations at an arbitrary moment.

The dynamic Voronoi diagram $\mathscr{VD}(D)$ is represented by four items: i) All edge-flipping events, ii) all disk-colliding events, iii) the moments of all events, and iv) the initial static Voronoi diagram $\mathscr{VD}^S(D, t = 0)$ at $t = 0$. The chronological sequence of the events with their time stamps is called the *event history*. Assuming the initial static Voronoi diagram (taking $O(n\log n)$ time in the worst case), the proposed TOI-E method computes the event history over the entire time horizon in $O(k_F \log n + k_C n \log n)$ time and $O(n + k_F + k_C)$ memory, both in the worst case, for the n disk generators, k_F edge flips, and k_C disk collisions. Given an event history, the Voronoi diagram $\mathscr{VD}^S(D, t^*)$ of an arbitrary moment $t^* \in [0, t^\infty)$ can be constructed in $O(k^* + n)$ time where k^* represents the number of events in $[0, t^*)$. This is because processing each of k^* events takes $O(1)$ time and the geometry of $\mathscr{VD}^S(D, t^*)$ can be evaluated in $O(n)$ time using its topology information at t^*. Once a static Voronoi diagram at an arbitrary moment t is obtained, reasoning the spatial properties among the disks at that moment can be done efficiently.

The contribution of this paper is as follows.

- The first report of the data structure and algorithm of the dynamic Voronoi diagram of moving circular disks.
- The first implementation of the dynamic Voronoi diagram of moving disks. The library and accompanying utility programs with documents are available at Voronoi Diagram Research Center, Hanyang University (http://voronoi.hanyang.ac.kr).

In this paper, the notion of complexity of both time and memory is in the worst case sense unless otherwise stated. We assume disks are mutually disjoint except at the moment of contact. We assume that two events do not occur at the same time: Two Voronoi edges flip at different moments; Two pairs of disks collide at different moments; An edge-flip and a disk collision do not occur at the same time. We also assume a good polynomial solver. "V-" denotes "Voronoi": E.g., V-vertex means Voronoi vertex.

This paper is organized as follows. Section 2 presents related prior studies. Section 3 presents the overview of the dynamic Voronoi diagram. Section 4 presents the transitions between state changes of the Voronoi diagram when disks move and is the core of this paper. Section 5 presents the most intriguing case of the dynamic Voronoi diagram of disks: The shadow operation does not exist in other types of Voronoi diagrams such as the ordinary Voronoi diagram of points or power diagrams. Section 6 presents the representation of event history of dynamic Voronoi diagram over time horizon. Section 7 presents experimental results with discussions. Section 8 presents an important and emerging application of dynamic Voronoi diagram of moving disks and balls. Then, the paper concludes. There are three appendices, available in the online supplemental material.

2 Related Works

It is called "kinetic" in the literature when the initial generators remain throughout the entire life of a system in that no generator is added to or removed from the system. No increase or decrease of a disk radius is allowed either. Otherwise, it is called "dynamic." We note that the insertion of a new disk to an existing structure is a building block of the topology-oriented incremental algorithm for constructing a static Voronoi diagram of disks [13] and deletion can be similarly handled. Hence, the difference between kinetic and dynamic algorithms is marginal, and we prefer to use the term "dynamic" to include both.

The dynamic Voronoi diagram was first studied for the ordinary Voronoi diagram of points. Gowda *et al.* (1983) first

TABLE 1
The Degree of the Polynomial for the Dynamic Voronoi Diagram ($f(t)$ in Eq. (1))

	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

Generators move linearly at constant speeds.

discussed the dynamic Voronoi diagram of points as a repeated deletion-and-insertion of point generators [14]. Ever since, many studies have followed including Devillers *et al.* (1993) [15], [16], Roos [17], Lauritsen *et al.* (1994) [18], Fabritiis and Coveney (2003) [19], and Schaller *et al.* [20]. For moving points, Schaller and Meyer-Hermann reported that the event-based method was twenty times faster than the recomputation method [20], and Lauritsen *et al.* [18] and Fabritiis and Coveney [19] reported that locally updating a triangulation was faster than repeatedly using delete and insert operations. Studies on the dynamic power diagram (or equivalently regular triangulation) by Gavrilova and coworkers (1996) [21], [22], [23] and by Güibas *et al.* [24] and others [25], [26] followed to handle proximity problems among disks.

It is important to note the difference between the Voronoi diagram and power diagram corresponding to a set of circular disks. The Voronoi diagram provides correct Euclidean distance information regardless two disks intersect or not. In the case of power diagram, however, correct Euclidean distance information between non-intersecting disks is not directly available while the information about intersecting disks is correct. This property has an important consequence in the spatial reasoning for solving applications. Suppose we want to compute the offset of a set of arbitrary disks (at fixed locations) with an offset amount δ. Given the Voronoi diagram, we can compute each of two offsets corresponding to $\delta = \delta_0$ and δ_1, $\delta_0 \neq \delta_1$, in the linear time of the number of disks. See Fig. 2b and 2c. However, if we want to compute the same offsets using power diagram, we need to construct two different power diagrams. First, it is necessary to construct the power diagram of the disks enlarged by δ_0 and use the topology of the power diagram to find the intersecting (enlarged) disks in the linear time. Second, the same procedure needs to be repeated once more with the disks enlarged by δ_1. See [27] for details.

We are aware of only two previous studies on the dynamic Voronoi diagram of moving disks [22], [28], [29], [30]. The study by Gavrilava and Rokne (1999) is of particular importance as their formula for the flip time of the V-edge when disks linearly move with constant speeds is fundamental [22], [31]. They observed that the flip time of a V-edge can be computed by finding the condition for a set of four disks to be cotangent to a common circumcircle, say ξ. The problem was transformed to a problem where a set of three disks cotangent to a common line via the Möbius transformation, also called the linear fractional transformation [32]. Let $D = \{d_1(c_1, r_1), d_2(c_2, r_2), d_3(c_3, r_3), d_4(c_4, r_4)\}$ be a set of four disks where a disk d_i has the center $c_i = (x_i, y_i)$ and radius $r_i \geq 0$, $i = 1, 2, 3$, and 4. Suppose that d_4 is the smallest.

Lemma 1. *(From [22] with a correction) The time of the edge-flip is the minimum positive real root of the polynomial*

$$f(t) = A^2 + B^2 - C^2 = 0, \tag{1}$$

satisfying

$$g_i(t) = \frac{B(x_i - x_4) - A(y_i - y_4)}{C} < p_i, \tag{2}$$

where

$$A = \begin{vmatrix} x_1 - x_4 & r_1 - r_4 & p_1 \\ x_2 - x_4 & r_2 - r_4 & p_2 \\ x_3 - x_4 & r_3 - r_4 & p_3 \end{vmatrix}, \quad B = \begin{vmatrix} y_1 - y_4 & r_1 - r_4 & p_1 \\ y_2 - y_4 & r_2 - r_4 & p_2 \\ y_3 - y_4 & r_3 - r_4 & p_3 \end{vmatrix}, \quad C = \begin{vmatrix} x_1 - x_4 & y_1 - y_4 & p_1 \\ x_2 - x_4 & y_2 - y_4 & p_2 \\ x_3 - x_4 & y_3 - y_4 & p_3 \end{vmatrix}. \tag{3}$$

where $p_i = (x_i - x_4)^2 + (y_i - y_4)^2 - (r_i - r_4)^2, \quad i = 1, 2, 3$ *and the disk motions are analytic functions of time* t.

Eq. (1) tests if ξ is cotangent to four disk generators where all four are simultaneously either inside or outside of ξ. Eq. (2) tests if ξ is empty, i.e., all four disks are placed outside of ξ. Provided that a reliable polynomial root finding library is available, the roots of Eq. (1) satisfying Eq. (2) can be found in $O(1)$ time.

If generators move linearly at constant speeds, $f(t)$ for d-dimensional spherical balls is a polynomial of degree $2(d + 2)$ [31]. Hence, $f(t)$ is a polynomial of degree 8 for two-dimensional disks and of degree 10 for three-dimensional spherical balls. If generators are points in a plane, $f(t)$ is of degree 4. If generators are disks and the power distance is used, $f(t)$ is also of degree 4. Table 1 summarizes this observation.

The second study was Karavelas' Ph.D. thesis (2001) [28], [29] where the Voronoi diagram for moving disks was discussed in terms of the dual structure (whose theory was completed by the introduction of the quasi-triangulation [33], [34], [35]). Quoting Gavrilova's 1999 work in [22] for the event time prediction in the diagram, the thesis described topology operations but not at the level sufficient enough to judge the correctness of the algorithm.

In Voronoi diagrams, the representation of infinity is essential. A popular and straightforward representation is to place some phantom generators sufficiently far away from the input generators [9], [13]. Fig. 3a shows the Voronoi diagram of ten generator disks (clustered in the center) plus three phantom disks located sufficiently far away from the generator disks so that the V-cells of the generator disks are bounded. A more convenient representation is a circular container with a sufficiently large radius to contain all the input generators as Fig. 3b and 3c [36], [37]. Fig. 3b shows the Voronoi diagram of the same generator disks in a circular container. The geometry of the V-edges is quadratic: Elliptic between a generator disk and the container and hyperbolic between two generator disks. Fig. 3c shows a

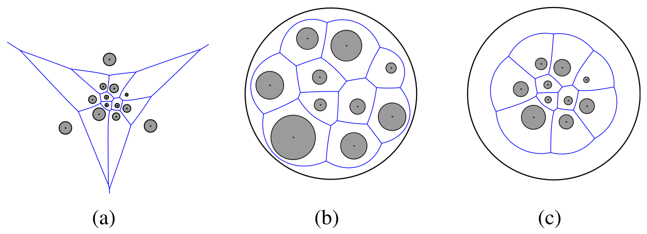


Fig. 3. Representation of infinity in the Voronoi diagram. (a) The conventional approach using phantom disks. (b) Infinity represented by a circular container. (c) An enlarged container to accommodate generator motions.

larger container to accommodate generator motions for a time interval. In this study, we use the approach in [36].

Scientific simulations frequently use particles to analyze critical measures such as contacts, short-, mid-, and long-range interactions among particles because the distribution functions of physical properties are useful for understanding nature [38]. Such interactions usually require a clever but complicated representation of the neighborhood relationships among particles to avoid combinatorial explosion. A well-known example is the simulation of a continuous-time dynamic billiard-like system consisting of many circular disks or spherical balls by Lubachevsky (1991) [39], [40], [41]. Other examples where particle interaction is critical include collision-avoiding path planning among moving vehicles [42], [43], [44], [45], [46], dancing and crowd simulation [47], [48], [49], [50], [51], collision-avoiding pedestrian path planning [52], [53], [54], [55], [56], [57], [58], evacuation process simulation [59], [60], etc. All such applications can take advantage of the information available in the Voronoi diagram.

Another application is packing circular disks and spherical balls in containers. Packing problem is an old and NP-hard problem. Since the first introduction in 1944 [61], the disk packing problem has been frequently formulated as nonlinear optimization problems [62], [63], [64], [65] and various heuristic methods were reported [66], [67], [68], [69], [70], [71]. Introducing the Shrink-and-Shake algorithm, Sugihara and Kim, together with coworkers (2004), showed that the geometric properties of disk arrangement could be a powerful tool for a good packing algorithm [36], [72], particularly using the Voronoi diagram of disks in a circular container [36]. Specht explicitly used the geometry information of the interstitial region in disk arrangement [73]. There are many emerging applications of the dynamic Voronoi diagram of moving disks such as coverage problems in mobile sensor networks [74], routing problems for multi-hop mobile ad-hoc networks [75], [76], target localization and tracking [77], and event-driven particle dynamics [78], [79]. Drone cinematography might be an emerging application if its three-dimensional counterpart is available [80], [81], [82].

3 Overview of the Dynamic Voronoi Diagram Algorithm

There are three types of discrete events in the proposed algorithm: Topology changes in the Voronoi diagram, collisions between disks, and velocity changes (due to collisions or other constraints). We assume linearity of disk motions and perfect elasticity; We ignore other physical properties such as friction with the plane. The skeleton of our algorithm given below follows Roos' work (1993) on the event-driven simulation of the dynamic Voronoi diagram of points [17], which was based on the framework of traditional discrete event simulation [83], [84]. Similar approaches have also been used by others [22], [23].

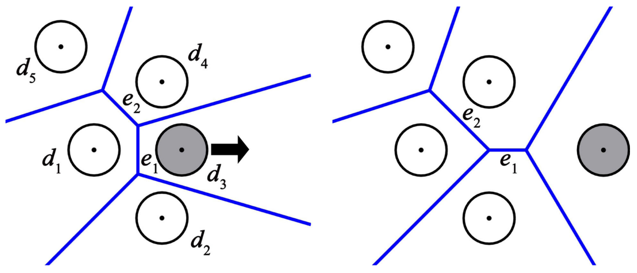


Fig. 4. (Ordinary Flip) Edge-flip of e_1 in a Voronoi diagram with ordinary V-cells. (a) Before the flip of e_1 and (b) after the flip.

Algorithm 1. DynamicVoronoiDiagram-Skeleton

```
1 (Step1. Preprocessing)
2   (a) Construct the initial static Voronoi diagram $\mathcal{VD}$ of a disk set $D$ at $t = 0$.
3   (b) For each V-edge, compute its potential flip time and collision time for generating disks.
4   (c) Initialize an event queue $Q$.
5 (Step2. Iteration) Iterate the following while time horizon is not over (or until there is no event in $Q$)
6   (a) Remove the nearest future event $E$ from $Q$ ($E$ is either edge flip or disk collision event).
7   (b) Process $E$ and accordingly update $\mathcal{VD}$, $D$, and $Q$.
```

Beginning with an initial static Voronoi diagram $\mathcal{VD}(D)$ at $t = 0$, we want to find all topology changes in the Voronoi diagram as disk generators move and the moments that disk pairs collide over the time horizon $[0, \infty)$. In practice, we are usually interested in a finite length of time $[0, t^\infty)$ where $t^\infty << \infty$. We keep the predicted events during $[0, t^\infty)$ in an ordered list $EventHistory$ so that we can quickly find the correct location of all the disks at an arbitrary moment $t \in [0, t^\infty)$, and the corresponding Voronoi diagram structure can be efficiently constructed.

Fig. 4a and 4b show Voronoi diagrams before and after the flip of the V-edge e_1 due to the motion of d_3 to the arrow direction, respectively. Before the flip, the V-edges e_1 and e_2 are defined by the disk sets $\{d_1, d_2, d_3, d_4\}$ and $\{d_1, d_3, d_4, d_5\}$, respectively. After the flip, e_1 is still defined by $\{d_1, d_2, d_3, d_4\}$ but e_2 is defined by $\{d_1, d_2, d_4, d_5\}$. The disk set for e_1 is unchanged but that for e_2 is changed. We call the set of the four disks an *edge generating disk set*, or *generating (disk) set* for a V-edge. A close observation shows the following. In Fig. 4a, the bisector for the V-edge e_1 is defined by the pair d_1 and d_3. The two V-vertices of e_1 are defined additionally by the two disks d_2 and d_4. We call the pair d_1 and d_3 the *crossing disk pair* (CDP) and the pair d_2 and d_4 the *traversing disk pair* (TDP) of e_1 and denote as $\mathrm{QUAD}(e_1) = \{\mathrm{CDP}(d_1, d_3), \mathrm{TDP}(d_2, d_4)\}$ where QUAD represents an *edge generating disk quadruplet* or *generating quadruplet* of e_1. Hence, the generating quadruplet of e_2 before the flip in Fig. 4a is $\mathrm{QUAD}(e_2) = \{\mathrm{CDP}(d_1, d_4), \mathrm{TDP}(d_3, d_5)\}$. After the edge flip in Fig. 4b, the generating disk set of e_1 remains unchanged but the generating quadruplet is changed by switching the roles, e.g., from $\mathrm{QUAD}(e_1)_{before} = \{\mathrm{CDP}(d_1, d_3), \mathrm{TDP}(d_2, d_4)\}$ to $\mathrm{QUAD}(e_1)_{after} = \{\mathrm{CDP}(d_2, d_4), \mathrm{TDP}(d_1, d_3)\}$. For e_2, the generating disk set is changed from $\{d_1, d_3, d_4, d_5\}$ to $\{d_1, d_2, d_4, d_5\}$. When a V-edge flips, there are changes in the number of V-edges bounding four V-cells. We call this type of flip an "Ordinary Flip" as it is applied to ordinary V-cells.

Definition 1. *An* ordinary V-cell *is bounded by more than three V-edges and when it is bounded by four V-edges, none of its adjacent V-cells are bounded by only two V-edges.*

For example, in Fig. 4a, after the flip, $\mathrm{VC}(d_1)$ and $\mathrm{VC}(d_3)$ have one V-edge less than they had before the flip and $\mathrm{VC}(d_2)$ and $\mathrm{VC}(d_4)$ one more than they had before. Note that there is no change in the total number of V-vertices, V-edges, or V-cells. In CDP and TDP, the order of the two disks is immaterial, and we keep the disks in the ascending order of indices.

Lemma 2. *Each V-edge of an ordinary V-cell has a unique flip time, and the corresponding topology change is well-defined.*

To construct $EventHistory$, we need to check if a V-edge flips as disks move and need to predict the moment t_{Flip} of an edge-flip event E_{Flip} if it does. Therefore, we scan all the edges in the initial Voronoi diagram at $t = 0$ to compute the flip time of each V-edge caused by all disk motions. We store the edge-flip event E_{Flip} in a priority queue Q_{Flip}, implemented as a heap, using the flip time t_{Flip} as the key. The root node of Q_{Flip} has the nearest future flip event. Then, we pop the root node, which has a flip event for an edge e, flip e, take care of related topology bookkeeping, and advance the time on the horizon. The bookkeeping takes care of the modification of the local topology configuration among the V-vertices, V-edges, and V-cells in the Voronoi diagram due to the flip, the update of the new flip time of the edges with a modified topology, and the relocation of the updated V-edges in Q_{Flip} according to the new key values. The flipped edge e should also be pushed into Q_{Flip} with a new flip time.

On the other hand, the collision time can be predicted with $\mathscr{VD}(D)$. If two disks collide in a sufficiently near future, they should define a V-edge in $\mathscr{VD}(D)$. Hence, for each V-edge in $\mathscr{VD}(D)$, we locate its two generating disks and check if they will collide or not. If they shall collide, we compute the collision time $t_{Collide}$ and store the collision event in another priority queue $Q_{Collide}$ using $t_{Collide}$ as the key. Given $d_1(c_1, r_1)$ and $d_2(c_2, r_2)$, a colliding moment is determined by $\|c_1 - c_2\| = r_1 + r_2$.

Lemma 3. *If disks d_i and d_j collide at t, there is a V-edge e of $\mathscr{VD}(D, t)$ where d_i and d_j are a crossing disk pair of e.*

We maintain two priority queues Q_{Flip} and $Q_{Collide}$ separately because edge-flip changes the topology of a Voronoi diagram whereas disk collision changes the states of the disks in D with modified velocities. There might be other types of events depending on applications. Q_{Flip} and $Q_{Collide}$ can also be merged into one queue if necessary.

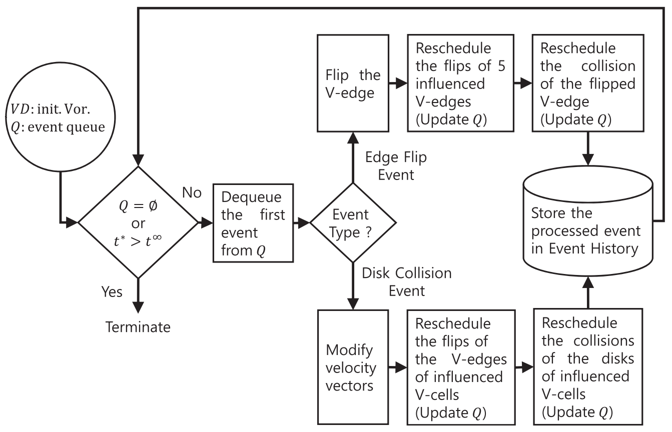
Given the initial arrangement of disks and its Voronoi diagram at $t = 0$, we now proceed at time $t \in [0, t^{\infty})$. We remove the root node of either Q_{Flip} or $Q_{Collide}$ whichever is the nearest future event. Let E be such an event and $t_E > 0$ be its event time. Depending on the type of E, the situation can be handled appropriately. If E is from Q_{Flip}, i)flip the corresponding V-edge, ii) re-compute the new edge-flip time for the five V-edges (i.e., the flipped V-edge e itself and those of four V-edges incident to e), iii) relocate each edge-flip event in an appropriate location in Q_{Flip}, iv) find the disks associated with e, both before and after the flip to get four disks in total, and v) update the disk collision times for the cases involving these four disks in $Q_{Collide}$. Note that the flipped V-edge among the five V-edges in step (ii) above always contains the current time t_E as one of the polynomial roots.

If E is from $Q_{Collide}$, the velocities after the collision can be calculated by i) modifying the velocities, ii) updating the new collision time in the neighborhood according to the new velocities and updating $Q_{Collide}$, and iii) updating the new flip time of the neighbor edges in Q_{Flip}. The algorithm is summarized in `Algorithm DynamicVoronoiDiagram`.

Algorithm 2. DynamicVoronoiDiagram

```
1  (Step1. Preprocessing)
2    (a) Given a set $D$ of initial disks, construct the static Voronoi diagram $\mathscr{VD}$ of $D$ at $t = 0$.
3    (b) For each V-edge, compute its potential flip time. Initialize a priority queue $Q_{Flip}$ by inserting all V-edges using the flip time as the key.
4    (c) For the disk pair defining each V-edge, compute its potential collision time. Initialize a priority queue $Q_{Collide}$ by inserting all such disk pairs using the collision time as the key.
5  (Step 2. Iteration) Iterate the following while the prediction time horizon is not over (or until there is no event in both $Q_{Flip}$ and $Q_{Collide}$).
6    (a) Remove the root node event $E$ of either $Q_{Flip}$ or $Q_{Collide}$ whichever is the nearest future event.
7    (b) If $E$ corresponds to an edge-flip event for an edge $e$, do the following.
8        (i) Perform the edge-flip by appropriately modifying the winged-edge data structure of $\mathscr{VD}$.
9        (ii) For the edge $e$, re-compute the new flip time and push into $Q_{Flip}$ if there exists a real solution $t$ greater than the current time.
10       (iii) For each of the four V-edges incident to $e$, re-compute the new flip time with a generator quadruplet with some new disks (as the local connectivity is modified) so that the new flip time becomes the new key value of the V-edge. Then, appropriately bubble up or down this edge in $Q_{Flip}$.
11       (iv) If $e$ flips, re-compute the new collision time for the modified disk pair of $e$ and bubble up or down the collision event in $Q_{Collide}$.
12   (c) If $E$ corresponds to a collision event between two disks, do the following.
13       (i) Modify the velocities of collided disks.
14       (ii) Update the collision time in the neighborhood reflecting the velocities and bubble up or down the collision events in $Q_{Collide}$.
15       (iii) Update the flip time of the neighbor edges and bubble up or down the flip events in $Q_{Flip}$.
```

Fig. 5 summarizes the flowchart of important steps in `Algorithm DynamicVoronoiDiagram`. The two event



VD: init. Vor.
Q: event queue
$Q = \emptyset$ or $t^* > t^\infty$
No
Yes
Terminate
Dequeue the first event from Q
Event Type ?
Edge Flip Event
Disk Collision Event
Flip the V-edge
Reschedule the flips of 5 influenced V-edges (Update Q)
Reschedule the collision of the flipped V-edge (Update Q)
Store the processed event in Event History
Modify velocity vectors
Reschedule the flips of the V-edges of influenced V-cells (Update Q)
Reschedule the collisions of the disks of influenced V-cells (Update Q)

Fig. 5. Flowchart of the proposed dynamic Voronoi diagram algorithm. (t^∞: Upper bound of the prediction time, t^*: time of the next event).

queues are merged into a single queue Q and the coefficient of restitution is ignored in the flowchart.

Step 1(a) takes $O(n\log n)$ time for n disks from a theoretical point of view (in our current implementation, we use the topology-oriented incremental algorithm, which takes $O(n^2)$ time but guarantees algorithmic robustness). Step 1 (b) takes $O(n\log n)$ time if the priority queue is implemented with a heap data structure. Flip time computation takes $O(n)$ time because the flip time of a V-edge can be computed in $O(1)$ time by Eqs. (3) and (2) and there are $O(n)$ V-edges in the Voronoi diagram of n disks in the plane. Step 1(c) takes $O(n\log n)$ time. Therefore, step 1 takes $O(n\log n)$ time.

Step 2(a) takes $O(\log n)$ time. Step 2(b)(i), (ii), (iii), and (iv) respectively take $O(1)$, $O(\log n)$, $O(\log n)$, and $O(\log n)$ time if the priority queue is implemented as a heap structure. A bubble up or down of a node with a modified key value to find its proper location in a priority queue takes $O(\log n)$ time. For the $O(\log n)$ time of Step 2(b)(iii), we have a direct pointer from each V-edge in the Voronoi diagram to its corresponding node in the priority queue. The time complexity of Step 2(c) is a little bit different in that $O(n)$ V-edges can be influenced by a single event of disk collision as shown in Fig. 6. Therefore, Step 2(c) takes $O(n\log n)$ time for each collision because we need to update $O(n)$ new event times on a priority queue based on heap data structure: We anticipate that it might take $O(\log n)$ on average in most cases.

The computational requirement of the entire Step 2 is proportional to the number of iterations, which is linear to the length of the time horizon $[0, t^\infty)$ and is given as follows.

Theorem 4. *`DynamicVoronoiDiagram` computes the event history of the time horizon in $O(k_F\log n + k_C n\log n)$ time with $O(n\log n)$ preprocessing time for n disk generators where k_F and k_C represent the numbers of edge-flip and disk-collision events over the time horizon, respectively.*

Let $k = k_F + k_C$ and ρ be the density of disks which is defined as the area of the union of the disks to the area of a minimum circle enclosing the generator disks and ν the average of the velocities assigned to the disks.

Lemma 5. *$k \propto t^\infty$, $k \propto \rho$, and $k \propto \nu$.*

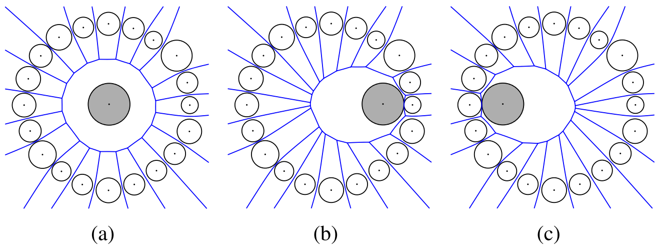


Fig. 6. The worst case of handling a collision event: In this case, a fast-moving disk d collides one of many, say $n-1$, disks where each share a V-edge with d. (a) The shaded disk in the middle of the disk arrangement moves fast. (b) and (c) The shaded disk moves from the initial placement in (a) where it collides with a disk in right or left part of the arrangement. At any moment, the shaded disk shares $n-1$ V-edges with the $n-1$ surrounding disks.

Lemma 6. *The number of V-edges and the number of V-vertices remain constant in the Voronoi diagram constructed by `DynamicVoronoiDiagram` at any moment $t \in [0, \infty)$.*

Proof. Let $\mathrm{QUAD}(e) = \{\mathrm{CDP}(d_{left}, d_{right}), \mathrm{TDP}(d_{start}, d_{end})\}$ be the quadruplet of a V-edge e. If e flips, one V-edge and one V-vertex are decremented from the V-cells of both d_{left} and d_{right}. On the other hand, one V-edge and one V-vertex are incremented in the V-cells of both d_{start} and d_{end}. Therefore, the number of Voronoi entities remains unchanged. □

Lemma 6 helps to improve the performance of the `DynamicVoronoiDiagram`. In its current form, in Step 2(a) and 2 (b), we explicitly remove and insert a node(s) from and to Q_{Flip}, respectively. We can instead bubble-up or down a V-edge(s) in Q_{Flip} with a new key value. The same idea of bubbling is also applied to Step 2(c) for disk collisions.

Lemma 7. *`DynamicVoronoiDiagram` takes $O(n+k)$ memory for n disks and k events.*

Proof. The representation of the topology structure takes $O(n)$ memory in the winged-edge data structure [33], [85], [86], [87]. The priority queue of edge-flip events has each of all the V-edges once having $O(n)$ elements, and the heap data structure for storing the priority queue takes $O(n)$ memory. Each of the edge-flip or collision events can be obviously stored in $O(1)$ memory, as will be presented in Section 6, and thus k events can be stored in $O(k)$ memory. Therefore, $O(n+k)$ memory is sufficient. □

Although it is output sensitive, the proposed algorithm is *compact* as it requires only $O(n+k)$ memory, is *responsive* as each flip and collision event is handled in $O(\log n)$ and $O(n\log n)$ time, respectively, and is *local* as it requires local changes in the topology of a Voronoi diagram.

4 State Transitions in Dynamic Voronoi Diagrams

The Voronoi diagram of disks is substantially different from the ordinary Voronoi diagram of points and the power diagram, mainly due to anomalies. Therefore, the algorithms to properly maintain their dynamic structures are also substantially different. The most critical problem resides in Step 2(a), which chooses the nearest future event in the time horizon. This is because in the Voronoi diagram of disks,

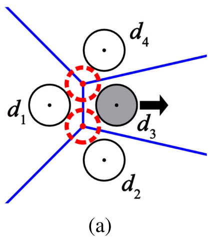


Fig. 7. (Ordinary Flip) Transition between ordinary V-cells. Equal-sized disks. (a) Before an edge-flip. (b) The edge contracts to a point. (c) After the edge-flip.

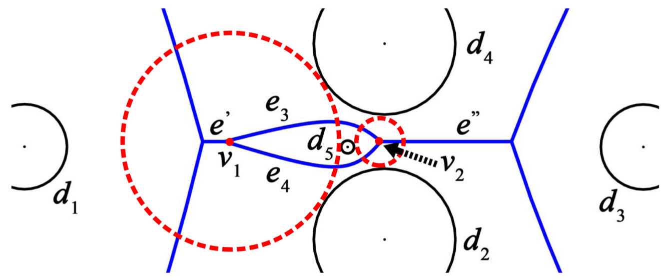


Fig. 8. A Voronoi diagram with an anomaly V-cell of d_5 trapped by two big disks d_2 and d_4.

there can be more than one V-edge with an identical event time. This does not happen in the other types of Voronoi diagrams. This section describes when such situations are encountered and how to handle them.

4.1 Transition in an Ordinary Case

When disks move, the Voronoi diagram undergoes a series of state transitions. Fig. 7 shows the transition of the Voronoi diagram of four identically sized disks where the shaded one is moving in the direction of the arrow. The red dotted circles in Fig. 7a are the maximum empty tangent circles corresponding to the V-vertices. Fig. 7b shows the moment, say t^*, when the four edge generating disks define a common empty circumcircle (computed by Eqs. (3) and (2)). The V-edge contracts to a point at t^* and immediately after this moment, the V-edge flips by an Ordinary Flip to result in Fig. 7c. If a same process begins with Fig. 7c and the shaded disk moves in the opposite direction of the arrow, Fig. 7a will result, also by an Ordinary Flip. An Ordinary Flip is reversible.

When the disks are of the same size, the Voronoi diagram is identical to an ordinary Voronoi diagram of disk centers. In the ordinary Voronoi diagram of points (and also in the power diagram), a cell is bounded by three or more V-edges, and no pathological situation occurs in the topology of a dynamic Voronoi diagram. The algorithm for the dynamic Voronoi diagram of points is rather simple.

Lemma 8. *(Necessary condition) Let* $\mathrm{QUAD}(e) = \{\mathrm{CDP}(d_{left}, d_{right}), \mathrm{TDP}(d_{start}, d_{end})\}$ *be the generating quadruplet of a V-edge e. If* d_{start} *and* d_{end} *are distinct, the V-edge e may flip in a foreseeable time under disk motions.*

Note that the two disks in CDP are always distinct and define a hyperbolic bisector. Lemma 8 implies that when the two disks in TDP are also distinct, the corresponding V-edge may flip in the near future. The actual flip time is the moment when an empty circumcircle to the four generating disks is defined by Eqs. (1) and (2). This implies the following observation.

Lemma 9. *V-edges with an identical generating disk set have an identical flip time.*

Recall that we assume no two events occur at the same time. Hence, two V-edges with different generating sets do not have an identical flip time. This assumption must be checked whenever two or more V-edges with an identical flip time occur.

4.2 Transition to a 2-Edge V-Cell: A Pathological Case

Consider five disks, d_1, d_2, d_3, d_4, and a tiny disk d_5 and the Voronoi diagram as shown in Fig. 8. Note the $d_5 \subset CH(d_2, d_4)$ where $CH(a, b)$ denotes the convex hull of a and b. In this case, there are two empty circumcircles among d_2, d_4, and d_5. The V-cell for d_5 is bounded by only two V-edges e_3 and e_4. The distinct V-vertices v_1 and v_2 are defined by an identical triplet of disks d_2, d_4, and d_5 where each V-vertex corresponds to one of the red-dotted empty circumcircles. This type of pathological case, called an anomaly, is unique in the Voronoi diagram of disks and does not happen in the ordinary Voronoi diagram of points or the power diagram. The situation related with an anomaly causes complications in the maintenance of the dynamic Voronoi diagram of disks.

In the Voronoi diagram of disks, unlike the ordinary Voronoi diagram of points, an edge generating disk set may not be unique for each V-edge. For example, in Fig. 8, both e_3 and e_4 are generated from an identical disk set $\{d_2, d_4, d_5\}$ of three disks, not four. As there are no four distinct disks to be cotangent to a common circumcircle, the flip condition in Lemma 8 cannot be satisfied, and thus the lemma cannot be applied. The generating quadruplet of e_3 is $QUAD(e_3) = \{\mathrm{CDP}(d_4, d_5), \mathrm{TDP}(d_2, d_2)\}$ and that of e_4 is $QUAD(e_4) = \{\mathrm{CDP}(d_2, d_5), \mathrm{TDP}(d_4, d_4)\}$. Note that the TDPs of both of the generating quadruplets degenerate. This condition violates the flip condition for e_3 and e_4 and thus these V-edges do not flip in the foreseeable future. The necessary condition for such a case is the $d_5 \subset CH(d_2, d_4)$ and we call d_5 to be *trapped* by d_2 and d_4. We call the V-cell of d_5 a *2-edge V-cell* implying that it is bounded by e_3 and e_4. Because the two bounding V-edges (in Fig. 8, e_3 and e_4) do not flip in the foreseeable future, we set their flip time as infinity and store it in the event queue Q_{Flip}. This proves the following lemma.

Lemma 10. *The two V-edges of (an isolated) 2-edge V-cell do not flip.*

A 2-edge V-cell is connected to the rest of the V-edge graph via two V-edges (which are called the "connecting V-edges"). Each of the two connecting V-edges has a unique generating disk quadruplet and thus has a unique flip time. In Fig. 8, for example, $e' = \{\mathrm{CDP}(d_2, d_4), \mathrm{TDP}(d_1, d_5)\}$ and $e'' = \{\mathrm{CDP}(d_2, d_4), \mathrm{TDP}(d_3, d_5)\}$. Note that the connecting V-edges have identical CDPs: d_2 and d_4 in this example.

The complications related with the 2-edge V-cell can be classified into two cases: i) Transitions between a triangular V-cell and an isolated 2-edge V-cell and ii) transitions between two trapped 2-edge V-cells.

4.3 Transition Between a Triangular V-Cell and a Trapped Isolated 2-Edge V-Cell

Consider, in Fig. 9a, a triangular V-cell of d_5 that has three V-edges e_5, e_6, and e_7. In this case, e_5, e_6, and e_7 are all

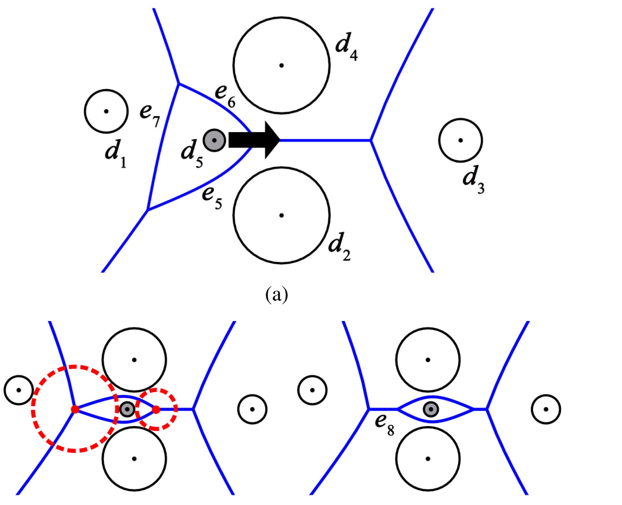


Fig. 9. (3-to-2 Flip) Transition between a triangular V-cell and an isolated 2-edge V-cell. (a) Before the edge-flip, (b) the V-edge e_7 is shrunken to a point, and (c) after the edge-flip.

defined by an identical set of four disks $\{d_1, d_2, d_4, d_5\}$ but with different generating quadruplets: e_5: $\{\mathrm{CDP}(d_2, d_5), \mathrm{TDP}(d_1, d_4)\}$; e_6: $\{\mathrm{CDP}(d_4, d_5), \mathrm{TDP}(d_1, d_2)\}$; e_7: $\{\mathrm{CDP}(d_1, d_5), \mathrm{TDP}(d_2, d_4)\}$. Suppose that d_5 moves to the right so that the $d_5 \subset CH(d_2, d_4)$ as shown in Fig. 9b. Then, d_5 defines a 2-edge V-cell and becomes trapped as shown in Fig. 9c.

In Fig. 9a, all three V-edges e_5, e_6, and e_7 around d_5 have an identical flip time because they all have an identical generating disk set $\{d_1, d_2, d_4, d_5\}$ and the three V-edges are placed in the priority queue Q_{Flip} with an identical key value. Therefore, when it is necessary to choose a V-edge to flip, it is critical to choose the right one.

In Fig. 9a, e_7 is the one to flip to e_8 in Fig. 9c. This case occurs only in a triangular V-cell because a 2-edge V-cell can be created only when a V-edge of the triangular V-cell flips. This case does not occur if a V-cell has four or more V-edges. The following lemma is proven.

Lemma 11. *All three V-edges of a triangular V-cell have an identical flip time.*

This lemma indeed implies that there can be at most three edges in a V-face with identical flip times. We call the V-cell of d_5 in Fig. 9a a *3-edge V-cell* and call the topology operation to transit from Fig. 9a, 9b, and 9c a 3-to-2 Flip.

Suppose that the disks are given as Fig. 9c and d_5 moves in the opposite direction (i.e., to the left). Then, the 2-edge V-cell in Fig. 9c transforms to the 3-edge V-cell in Fig. 9a via an Ordinary Flip of e_8. In this sense, a 3-to-2 Flip is irreversible.

How can the right V-edge among the three candidate V-edges be chosen? For example, in Fig. 9a, how to choose e_7 to flip from e_5, e_6, and e_7? Suppose that t^* is the edge-flip time. In the case of multiple V-edges with an identical edge-flip time, we temporarily move the generating disks to the location corresponding to t^* and check which V-edge actually shrinks to a point. Hence, we actually move at most four disks and compute the circumcircle only once, thus taking $O(1)$ time. We call this a `Move-&-Check` *operation*. In Fig. 9a, $\{d_1, d_2, d_4, d_5\}$ generates e_5, e_6, and e_7. This proves the following lemma.

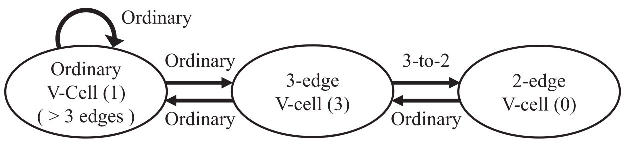

$$\begin{tikzcd}
\text{Ordinary V-Cell (1) } (>3 \text{ edges}) \arrow[loop above, "\text{Ordinary}"] \arrow[r, shift left, "\text{Ordinary}"] & \text{3-edge V-cell (3)} \arrow[l, shift left, "\text{Ordinary}"] \arrow[r, shift left, "\text{3-to-2}"] & \text{2-edge V-cell (0)} \arrow[l, shift left, "\text{Ordinary}"]
\end{tikzcd}$$

Fig. 10. State transition diagram in the big world of the dynamic Voronoi diagram of disks due to an edge-flip. The label on each arrow denotes the corresponding flip operation.

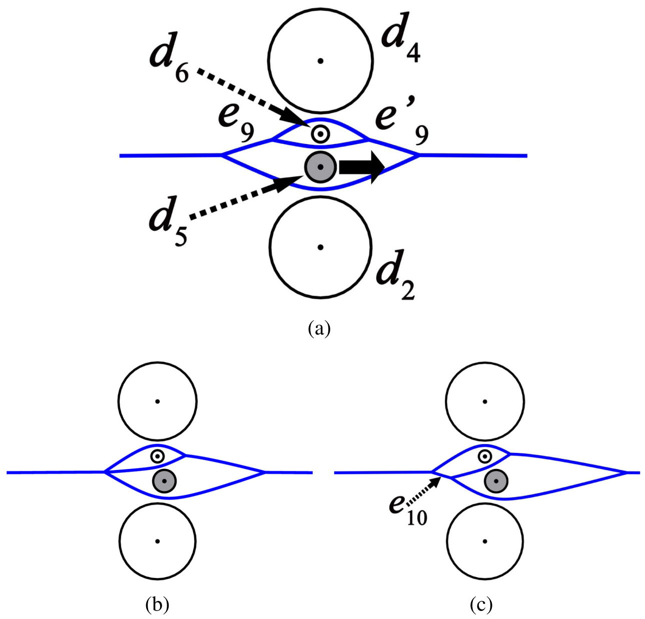


Fig. 11. (24-to-33 Flip) Two V-edges e_9 and e'_9 have an identical flip time. (a) Before the edge-flip (Configuration I), (b) a shrunken edge, and (c) after the edge-flip (Configuration X).

Lemma 12. *The transformation from a 3-edge V-cell to a 2-edge V-cell can be done by a 3-to-2 Flip operation in $O(1)$ time.*

Fig. 10 summarizes the discussion up to this point: The state transition of a single V-cell from an ordinary V-cell to a 3-edge V-cell to a 2-edge V-cell or vice versa.

4.4 Transition Between a Pair of Trapped V-Cells

This section describes the state transition of two trapped V-cells.

Consider a case where two small disks are trapped by two big disks. For example, see Fig. 11a where $d_5 \cup d_6 \subset CH(d_2, d_4)$. There are six different ways to define a bisector between two disks from a set of four disks, i.e., $C(4, 2) = 6$. As the bisector between the two big disks d_2 and d_4 is split into two connecting V-edges, there can be altogether at most seven V-edges. Recall that each of the two connecting V-edges has a unique flip time. The five bisectors contributing to the V-edges in the trapped region require careful investigation. Note that the V-edges form a planar graph. Consider the following three cases.

- Configuration I (Fig. 11a) One small disk is exposed to only one big disk, the other small disk is exposed to both big disks, and the small disks are adjacent to each other in the Voronoi diagram (i.e., they interact with each other). The two disks are linearly aligned and the disk configuration looks like the letter I.

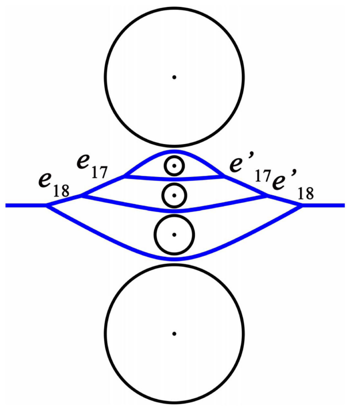


Fig. 12. The V-edges e_{17} and e'_{17} have an identical flip time, say t_1. Similarly, e_{18} and e'_{18} also have an identical flip time $t_2 \neq t_1$.

- Configuration X (Fig. 13a) Both of the small disks are exposed to big disks, and two small disks are adjacent to each other. The disk configuration looks like the letter X.
- Configuration W (Fig. 14a) Both of the small disks are exposed to both big disks, and the two small disks are not adjacent to each other (i.e., they do not interact with each other). The disk configuration looks like the letter W.

Hence, the following observation holds.

Lemma 13. *There are three, and only three, different configurations for each pair of trapped disks: Configurations I, X, and W.*

Proof. The V-cells of two trapped disks may be either adjacent or non-adjacent to each other. If they are adjacent to each other, there can only be two cases as follows: i) One is a two-edge V-cell, and the other has four V-edges (Fig. 11a) and ii) both have three V-edges (Fig. 13a). If they are not adjacent to each other, the only possible case is Fig. 14a. □

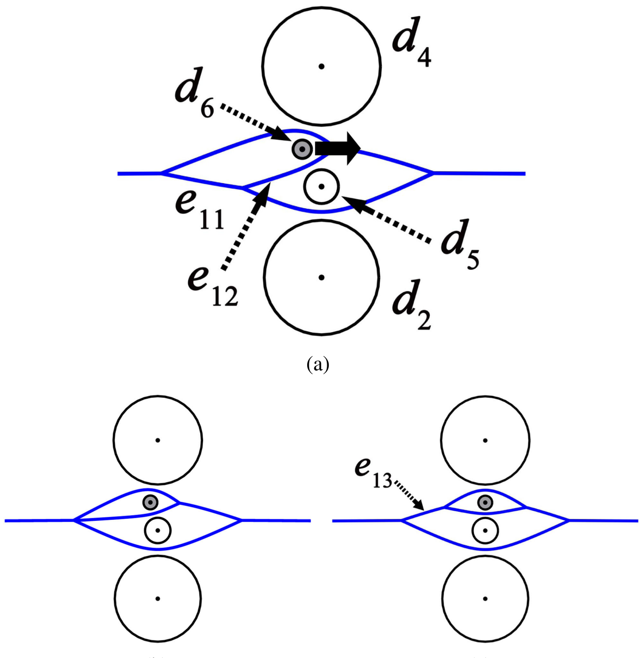


Fig. 13. (33-to-24 Flip) All five V-edges of the two 3-edge V-cells have an identical flip time (except for the connecting V-edges). (a) Before the edge-flip (Configuration X), (b) a shrunken edge, and (c) after the edge-flip (Configuration I).

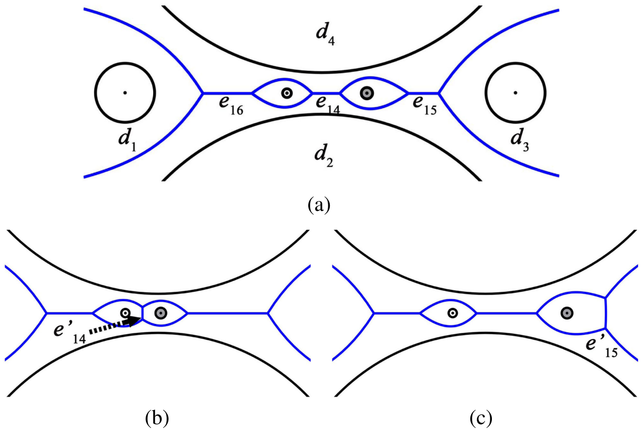


Fig. 14. (Ordinary Flip and 33-22 Flip) The three connecting V-edges e_{14}, e_{15} and e_{16} have a unique flip time. (a) Before the edge-flip (Configuration W), (b) an edge-flip of e_{14} to e'_{14}, (c) an edge-flip of e_{15} to e'_{15} (Configuration X).

4.4.1 Configuration I

In Configuration I in Fig. 11a, there is a 2-edge V-cell $VC(d_6)$. In this case, the bisector between d_4 and d_5 defines two distinct V-edges e_9 and e'_9 among the five V-edges in total. In this case, both e_9 and e'_9 have an identical generator quadruplet $\{\mathrm{CDP}(d_4, d_5), \mathrm{TDP}(d_2, d_6)\}$ and thus have an identical flip time by Lemmas 8 and 9. The other three V-edges are defined by only three distinct disks $\{d_4, d_5, d_6\}$ and $\{d_2, d_4, d_5\}$ and thus have degenerate generator quadruplets $\{\mathrm{CDP}(d_4, d_6),$ $\mathrm{TDP}(d_5, d_5)\}$, $\{\mathrm{CDP}(d_5, d_6), \mathrm{TDP}(d_4, d_4)\}$, and $\{\mathrm{CDP}(d_2, d_5),$ $\mathrm{TDP}(d_4, d_4)\}$ so that their flip times are not defined.

Lemma 14. *Configuration I has two V-edges with an identical flip time.*

When d_5 moves to the arrow direction, we need to choose the correct V-edge between e_9 and e'_9 to flip. In this example, e_9 flips. To make this choice correct, the Move-&-Check operation is used, i.e., we temporarily move the related four disks and measure the lengths of the V-edges. We call the topology operation from Fig. 11a, 11b, and 11c a *24-to-33 Flip* because the two V-faces with 2 and 4 V-edges are transformed to V-faces, both with 3 V-edges).

Lemma 15. *The 24-to-33 Flip can be done in $O(1)$ time.*

If the disks are arranged as in Fig. 12, e_{17} and e'_{17} have an identical flip time, say t_1; e_{18} and e'_{18} have another identical flip time $t_2 \neq t_1$. This case may also be repeated an arbitrary number of times and can be resolved by applying the Move-&-Check operation for each pair of such V-edges.

4.4.2 Configuration X

Fig. 13a shows a disk configuration that is identical to Fig. 11c. We call this disk configuration X in that the disk arrangement looks like the letter X. In this case, all of the five V-edges have an identical generating disk set $\{d_2, d_4, d_5, d_6\}$ and therefore an identical flip time. (Another possible explanation is as follows. Each of the two

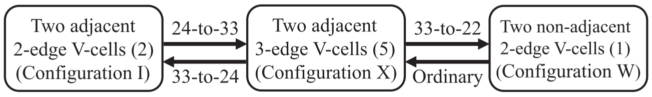

Two adjacent 2-edge V-cells (2) (Configuration I)
24-to-33
33-to-24
Two adjacent 3-edge V-cells (5) (Configuration X)
33-to-22
Ordinary
Two non-adjacent 2-edge V-cells (1) (Configuration W)

Fig. 15. State transition diagram in the small world of the dynamic Voronoi diagram of disks due to the edge-flip. The label on each arrow denotes the corresponding flip operation.

triangular V-faces have all three V-edges with an identical flip time. Because the two V-faces have a common V-edge, the five V-faces have an identical flip time.) All five of the V-edges have different generating quadruplets without any degeneracy. To determine the right V-edge to contract, we again use the $O(1)$ time `Move-&-Check` operation. Note that the disk configurations of Fig. 13a and 13c are identical to those of Fig. 11c and 11a, respectively. The topology operation from Fig. 13a, 13b, and 13c is called the *33-to-24 Flip* taking $O(1)$ time. Therefore, the following observation holds.

Lemma 16. *Configuration X has five V-edges with an identical flip time.*

Lemma 17. *The 33-to-24 Flip can be done in $O(1)$ time.*

4.4.3 Configuration W

In Fig. 14, each small disk defines an isolated 2-edge V-cell, and they are not adjacent to each other in the Voronoi diagram. Recall that the V-edges bounding the 2-edge V-cells do not flip by Lemma 10. On the other hand, each of the three connecting V-edges e_{14}, e_{15}, and e_{16} has a unique flip time because it has a unique generating disk set. Flip e_{14} (or e_{15}) by an Ordinary Flip operation results in Fig. 14b (or (c)) from (a). There can be an arbitrary number of tiny disks arranged in this fashion to form a W Configuration.

The topology operation from Fig. 14b to 14a is called a 33-to-22 Flip taking $O(1)$ time. The following observation holds.

Lemma 18. *Configuration W has three V-edges with distinct flip times.*

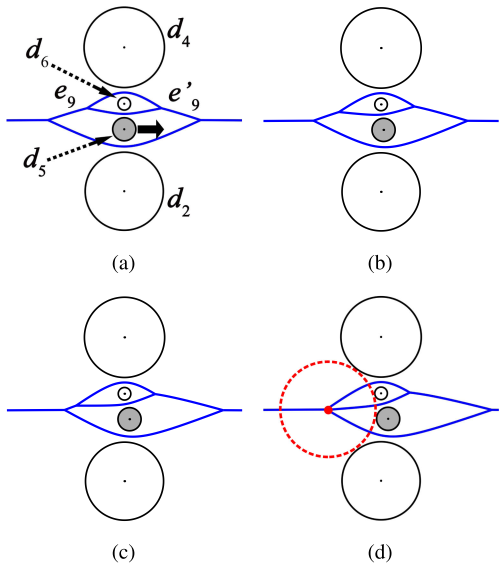


Fig. 16. Duplicates of Fig. 11. (a) Before the edge-flip ($\mathscr{VD}_a$), (b) a shrunken edge ($\mathscr{VD}_b$), (c) before the edge-flip ($\mathscr{VD}_c$), and (d) a shrunken edge ($\mathscr{VD}_d$).

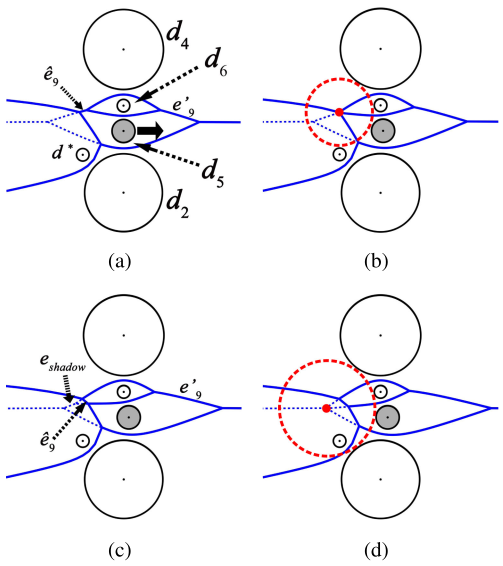


Fig. 17. Shadow flip of a V-edge. (a) Before the edge-flip ($\mathscr{VD}^*_a$), (b) a shrunken edge ($\mathscr{VD}^*_b$), (c) before the edge-flip ($\mathscr{VD}^*_c$), and (d) a shrunken edge ($\mathscr{VD}^*_d$).

Lemma 19. *A 33-to-22 Flip can be done in $O(1)$ time.*

Fig. 15 summarizes the state transitions in small worlds due to the flip of the V-edges during maintenance of the dynamic Voronoi diagram. Note that four V-edges can have an identical flip-time: See Appendix 2, available in the online supplemental material.

5 Shadow Operations : Shadow V-vertex and Shadow Flip

Fig. 16a and 16d are duplicates of Fig. 11a and 11b, respectively. Recall that only d_5 moves toward the arrow direction. Given the Voronoi diagram, say $\mathscr{VD}_a$, in Fig. 16a, the next edge-flip event occurs at the moment of $\mathscr{VD}_d$ in Fig. 16d. Fig. 16b and 16c show intermediate Voronoi diagrams.

Fig. 17 shows the Voronoi diagram, say $\mathscr{VD}^*$, with an additional disk d^*. Note that the V-cell of d^* trims off some V-cells of the Voronoi diagram $\mathscr{VD}$ in Fig. 16. The dotted curve segments in Fig. 17 are the trimmed portion of the V-edges in the corresponding $\mathscr{VD}$ and do not exist in $\mathscr{VD}^*$. For example, $\hat{e}_9$ in Fig. 17a is the trimmed edge of e_9 of $\mathscr{VD}_a$ in Fig. 16a.

Given the Voronoi diagram $\mathscr{VD}^*_a$ in Fig. 17a, the next edge-flip occurs when the V-edge $\hat{e}_9$ contracts to a point as shown in $\mathscr{VD}^*_b$ in Fig. 17b. Fig. 17c shows the Voronoi diagram $\mathscr{VD}^*_c$ after a tiny time increment. Be aware that there is a *shadow V-edge* e_{shadow} which is contained in the V-cell of the disk d^*. The shadow V-edge does not exist in the data structure of a real Voronoi diagram $\mathscr{VD}^*_c$.

Then, an important situation is encountered. The next edge-flip event is computed as $\mathscr{VD}^*_d$ as shown in Fig. 17d when the e_{shadow} contracts to a point. As explained previously in Section 4.4.1, the flip time of e'_9 is identical to that of e_{shadow} because both e'_9 and e_{shadow} are defined by the same set of four generating disks $\tilde{D} = \{d_2, d_4, d_5, d_6\}$. The algorithm recognizes if e_{shadow}, not e'_9, flips at the moment of Fig. 17d. Hence, in this case, we do not perform any actions but proceed to the next event in the time horizon. In this sense, we call it the *shadow flip* of the V-edge and the V-vertex at the center of the red circumcircle a *shadow Voronoi vertex*. A shadow flip is unique in the Voronoi diagram of disks and does not exist in an ordinary Voronoi diagram of points or a power diagram.

It is necessary to distinguish a shadow flip from a real flip. Suppose that we apply the $O(1)$ time `Move-&-Check` operation with the generating disk set $\tilde{D}$ to the edge-flip time so that we get the location of the shadow Voronoi vertex, say v. Then, we look into the root node r of the priority queue Q_{Flip} to check if the location of v is identical to either one of the V-vertices of r. If the location of v is not identical to either one of the V-vertices, this edge-flip is a shadow.

Lemma 20. *A shadow flip can be done in $O(1)$ time in the worst case.*

Theorem 21. *Given an event history, the dynamic Voronoi diagram of n disks over the time horizon $[0, t^{\infty})$ can be constructed in $O(k+n)$ time with $O(nlogn)$ time for preprocessing to construct the initial Voronoi diagram at $t=0$ where k denotes the number of events in the time horizon.*

6 Representation of the Event History

We store all the events of the dynamic Voronoi diagram in the time horizon in an *event history* in chronological order because any spatial queries on the generator disks at an arbitrary moment can be efficiently answered from the event history and the initial state of a system. A disk-collision event has three data items $(t_{Collide}, \uparrow d_1, \uparrow d_2)$: $t_{Collide}$ is the disk collision time, $\uparrow d_1$ and $\uparrow d_2$ are the pointers or indices to the colliding disks. The velocity vectors of the disks after a collision can be calculated. An edge-flip event stores two data items $(t_{Flip}, \uparrow e)$: t_{Flip} is the edge-flip time and $\uparrow e$ is the pointer to the V-edge to be flipped. We may represent $\uparrow e$ with a disk quadruplet which defines e to store event history information in a file to use in applications. Note that the computational requirement of an event history is heavy.

The event history is used to efficiently analyze the spatial properties of moving disks at $t^* \in [0, t^{\infty})$. Let t_k be the time of an kth event E_k in the event history. First, we scan the event history to locate E_{k-1} and E_k where $t_{k-1} \leq t^* < t_k$. Then, the topological structure of the Voronoi diagram at t^* can be constructed in $O(k)$ time by performing k edge-flips and/or disk-collisions. We want to construct a static Voronoi diagram $\mathscr{VD}^S(D, t^*)$ as quickly and robustly as possible.

There are two operators for obtaining the static Voronoi diagram $\mathscr{VD}^S(D, t^*)$ at $t^* \in [t_i, t_{i+1})$ from a history file and the initial set of disk generators.

- `Move disks` places the disk generators at the correct locations at t^*. There are three alternatives. For each event up to $t_i < t^*$: `Move-1` moves all disks at each event; `Move-2` moves all disks at each collision event; or `Move-3` moves only two colliding disks at each collision event.

  Then, all disks are moved to the final locations at t^*. `Move-3` shows the best performance in terms of both computational efficiency and numerical robustness because the number of arithmetic operations needed to obtain the disk locations at the last moment is minimized.
- `Construct` $\mathscr{VD}$ realizes the construction of the Voronoi diagram. There are two alternatives. `Construct-1` constructs the static Voronoi diagram of the disk generators located at t^*; `Construct-2` updates the topology of the Voronoi diagram by doing the edge-flip operations for all edge-flip events up to $t_i < t^*$ and evaluates the geometry of the static Voronoi diagram using the disks located at t^* with the current topology of the static Voronoi diagram.

Hence, there can be six different strategies defined from the combinations of the two operators. An appropriate approach can be chosen depending on parameters such as the densities, velocities, and radii of disks.

7 Experiments and Discussions

We have implemented the proposed algorithm and tested thoroughly for performance evaluation using the following computational environment: Intel Core i7-7700 3.60 GHz; 16 GB RAM; Windows 10 Professional (64 bit); Visual C++ on Microsoft Visual Studio Community 2017. We emphasize that we used only one core in this experiment.

7.1 Test Data

We have generated five types of disk sets. The first one is $\mathrm{REFERENCE} = \{D_1, D_2, \ldots, D_{10}\}$ where D_i is a model file containing $1000 * i$ mutually disjointed random disks. The disks in D_i are placed in a sufficiently large circular container $\mathscr{C}_i = (O, R_i)$ centered at the origin O with the radius R_i. Each disk $d_{i,j} = (c_{i,j}, r_{i,j}) \in D_i$ has a random radius $r_{i,j} \in [1.0, 10.0]$ and is centered at a random location $c_{i,j}$ in $\mathscr{C}_i$. Each disk is associated with the velocity vector of a random direction but of a unit speed 1.0. Let $\sigma \in [0.0, 1.0]$ be the coefficient of restitution between two disks which is defined as the ratio of the relative velocity after a collision to that before the collision. We assume that the coefficient of restitution of any disk pairs in REFERENCE are identically 1.0, i.e., are perfectly elastic. Let ρ be the density of D_i, which is defined as the ratio of the area of the union of the disks in D_i to the area of $\mathscr{C}_i$. It is known that $\rho \leq 0.9069$ (The equality holds for mono-sized disks). For all model files in the experiments in this paper, $\sigma = 1.0$ implying that the disks are perfectly elastic and $\rho \approx 0.05$ to investigate the long-range behavior of the particle systems, unless otherwise stated.

We created four additional types of disk sets to investigate the algorithm properties related to the number of events in the dynamic Voronoi diagram that is dependent on the initial speed of the disks, the size of the disks, the packing density of the disks, and the coefficient of the restitutions. Suppose that the D^{base} is a model file containing a set of 1,000 mutually disjoint disks that are randomly created in a circular container $\mathscr{C}$, in a way similar to REFERENCE. In other words, $d_i = (c_i, r_i) \in D^{base}$ has a random radius $r_i \in [1.0, 10.0]$ at a random location c_i in $\mathscr{C}$ with a random velocity vector with a unit speed, $\rho \approx 0.05$, and $\sigma = 1.0$ for D^{base}.

The four types of disk sets are as follows. SPEED-*VARIED* $= \{D_1^{speed}, D_2^{speed}, \ldots, D_{10}^{speed}\}$. The disks in D_k^{speed} are generated with a rule identical to that used for the D^{base} except that the initial speed of all the disks in the D_k^{speed} is

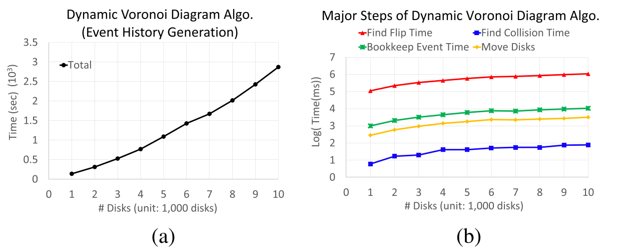


Fig. 18. Computation time (Data set: $\mathrm{REFERENCE}$). Ten sets of disks. #disks: 1,000, 2,000, . . ., 10,000. Random radii in $[1.0, 10.0]$ with random velocities but with a unit speed. $t^{\infty} = 1,000$. $\rho \approx 0.05$. $\sigma = 1.0$. (a) Total computation time. (b) Computation time for the four major steps. Be aware of the logarithm scale. (Red: Step 1; Green: Step 3; Yellow: Step 4; Blue: Step 2.).

set to a constant k. $\mathrm{SIZE\text{-}VARIED} = \{D_1^{size}, D_2^{size}, \ldots, D_{10}^{size}\}$ where the disks in D_k^{size} are similarly generated except that the radii r_k of the disks in D_k^{size} is randomly chosen so that $r_k \in [1.0, k]$. $\mathrm{DENSITY\text{-}VARIED} = \{D_1^{density}, D_2^{density}, \ldots, D_{10}^{density}\}$ where the disks in the $D_k^{density}$ are similarly generated except that the density ρ of the $D_k^{density}$ is $0.05 * k$. $\mathrm{RESTIT\text{-}VARIED} = \{D_1^{restit}, D_2^{restit}, \ldots, D_5^{restit}\}$ where the disks in D_k^{restit} are similarly generated, except that the coefficient of the restitution σ of D_k^{restit} is $0.5 + k/10.0$. For example, σ of $D_1^{restit} = 0.6$.

7.2 Correctness of the Constructed Dynamic Voronoi Diagrams

We thoroughly checked and found the correctness of the constructed Voronoi diagrams of all of the ten model files in the $\mathrm{REFERENCE}$ data set. All of the constructed Voronoi diagrams passed the tests performed as follows.

We first constructed the dynamic Voronoi diagram over the time horizon $[0, 1, 000]$ and stored the history file. Then, we uploaded the history file to perform the following test: We advanced the time horizon to t^{curr} by scanning the history file to get the topology and geometry of the Voronoi diagram at t^{curr}. Then, we computed the maximum empty circumcircle ξ at the moment for the three disk generators associated with each V-vertex and checked if ξ intersects any other disks. We used 10^{-6} as the tolerance to check the intersections.

We used two different strategies to advance the time horizon. The first is to increment the constant time interval Δt, i.e., $t = t + \Delta t$ where $\Delta t = 0.1$. The other is to increment the time to the moment of the next event and check the Voronoi diagram in the middle of the time interval between two consecutive events.

7.3 Computation Time Profile

We analyzed the computational efficiency properties of the proposed dynamic Voronoi diagram algorithm, which consists of four major tasks: Step 1) Find the possible flip time of the V-edges; Step 2) Find the possible collision time between the disks; Step 3) Do the bookkeeping for the flip or collision time in the priority queue; Step 4) Move the disks corresponding to a related event time.

Fig. 18a shows the total computation time of the proposed algorithm to the size of the model files of $\mathrm{REFERENCE}$. Recall that the time horizon is 1,000 units, $\sigma = 1.0$, $\rho \approx 0.05$. We observed a weak super-linearity,

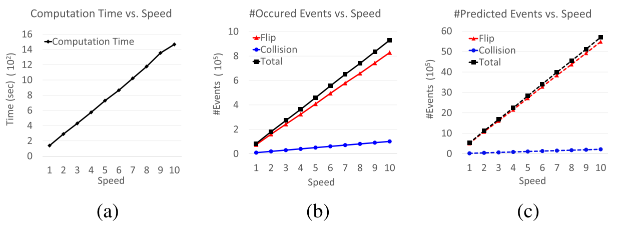


Fig. 19. Influence of the initial speeds of the disks (Data set: $\mathrm{SPEED\text{-}VARIED}$). Ten sets of 1,000 disks with an initial speed of 1.0, 2.0, . . ., and 10.0. Random radii in $[1.0, 10.0]$. Random moving directions for the disks. $t^{\infty} = 1,000$. $\sigma = 1.0$. (a) The computation time, (b) the number of events occurring, (c) the number of predicted events.

which verifies the Theorem 4. Fig. 18b shows four curves that correspond to the four major tasks above: Red: Step 1; Green: Step 3; Yellow: Step 4; Blue: Step 2. Be aware that the vertical axis of Fig. 18b is logarithm-scaled.

7.4 Influence of the Initial Speeds of the Disks

We studied the algorithm properties under the speed varying conditions. Fig. 19 shows the analysis using the $\mathrm{SPEED\text{-}VARIED}$ data set. Be aware that the time horizon is up to 1,000 units, $\sigma = 1.0$, $\rho \approx 0.05$. Fig. 19a shows the total computation time, which is strongly linear.

To verify the linearity, we further studied the number of events, which took the majority of the computation using this data set. Fig. 19b and 19c shows the numbers of events occurring and the predicted ones over the time horizon, respectively: Red: Flip events; Blue: Collision events; Black: Total events. Note that the horizontal axis denotes the initial speeds of the disks. Based on this analysis, we make the following observations:

- All curves are strongly linear.
- The gap between the red and blue curves is significant yet consistent.
- The scale difference between the predicted and the occurring events is an order of magnitude. Specifically, only 15 percent of the predicted flip events occurred and 42 percent for the collision events.

7.5 Influence of the Disk Sizes

We also studied the algorithm properties under size varying conditions using the $\mathrm{SIZE\text{-}VARIED}$ data set. Fig. 20a shows the events occurring with respect to the maximum disk radius (i.e., the upper bound of the range of the disk radius from which each disk radius is randomly determined): Red:

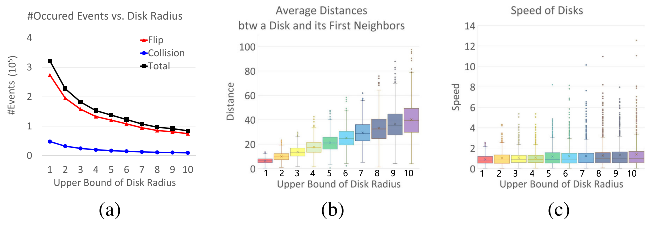


Fig. 20. Influence of disk sizes (Data set: $\mathrm{SIZE\text{-}VARIED}$). Ten sets of 1,000 disks. Upper-bound from which the disk radii were sampled in each set: 1.0, 2.0, . . ., and 10.0. Initial speed of the disks: 1.0. Random moving directions for the disks. $t^{\infty} = 1,000$. $\rho \approx 0.05$. $\sigma = 1.0$. (a) The number of events occurring. (b) The average distance between a disk and its first neighbors. (c) The speed of the disks.

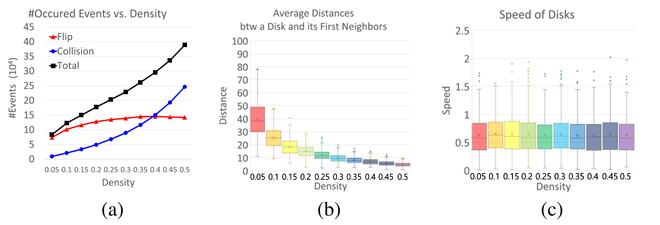


Fig. 21. Influence of densities (Data set: DENSITY-VARIED). Ten sets of 1,000 disks whose density ρ: $0.05, 0.10, \ldots, 0.50$. The disk radius is randomly set in $[1.0, 10.0]$. The initial speed of the disks: 1.0. Random moving directions for the disks. $t^{\infty} = 1,000$. $\sigma = 1.0$. (a) The number of events occurring. (b) The average distances between a disk and its first neighbors. (c) The speed of the disks.

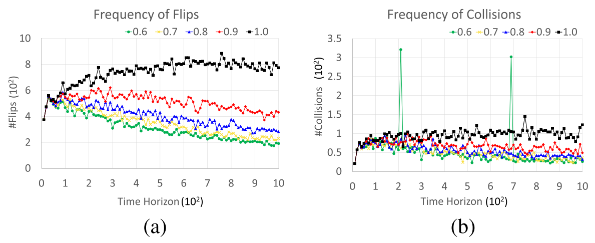


Fig. 22. Frequency of the dynamic Voronoi diagram events (Data set: RESTITE-VARIED). 5 sets of 300 disks whose coefficients of restitution are 0.6, 0.7, ..., and 1.0. The initial speed of the disks: 1.0. Random moving directions for the disks. $t^{\infty} = 1,000$. $\rho \approx 0.05$. (a) frequency of the flip events, and (b) frequency of the collision events.

Flip events; Blue: Collision events; Black: Total events (i.e., Red + Blue). Note that all curves are monotonic, decreasing in a nonlinear fashion which can be explained by Fig. 20b and 20c. Fig. 20b shows the average distances between each disk and its first Voronoi neighbors at the last moment in the time horizon, i.e., 1,000. Two disks d_i and d_j are Voronoi neighbors if and only if the V-cells of the two disks share a V-edge. We measure the average distance (between the boundaries of two disks) at the last moment in the belief that the system has reached a steady state. Fig. 20c shows the distribution of the disk speeds at the last moment of the time horizon. We made the following observations that explain the monotonic decrease shown in Fig. 20a: i) Both the distance and its variance increase as the maximum disk radius increases; ii) The average inter-disk distance is a strong linear increase; iii) The average speeds are very close to the initial condition of the unit speed 1.0 but with an overall increasing variance.

7.6 Influence of the Disk Densities

We studied the density varying conditions using the DENSITY-VARIED data set. Fig. 21a shows the events that occurred with respect to the density. The three curves correspond to the event types: Red: Flip events; Blue: Collision events; Black: Total events. The collision curve is super-linear, increasing while the flip curve shows a sub-linear increase pattern up to $\rho = 0.4$ when the curve seems to reach a steady state. The black curve shows a roughly linear increase. This observation can be nicely explained by Fig. 21b and 21c, which shows the average inter-disk distances between each disk and its first Voronoi neighbors and the disk speeds at the last moment of the time horizon, respectively. We make the following observations: i) Average distance is inversely proportional to density: This means that a higher density leaves a smaller allowable space for disk moves; ii) Disk speeds are roughly constant;

7.7 Influence of the Coefficient of the Restitutions

We studied the frequency of the events occurring over the time horizon using the RESTITE-VARIED data set for the variable coefficient of the restitutions. Fig. 22a and 22b shows the frequency of a flip and that of the collision events. In this experiment, we used a time horizon up to 1,000 to obtain a steady state. There are five curves where each corresponds to a coefficient of restitution σ: Green: 0.6; Yellow: 0.7; Blue: 0.8; Red: 0.9; Black: 1.0. Each point in the curves represents the number of events in the most recent time interval of 10 units. In other words, the point on the black curve of Fig. 22a at 500 on the time horizon denotes the number of flip events that have occurred during the time units $[490, 500)$. We make the following observations: i) If $\sigma = 1.0$, the system quickly reaches a steady state; ii) If $\sigma < 1.0$, the system goes through a transient state before reaching a steady state; iii) When $\sigma < 1.0$, the higher σ is, the longer the transient state is. The rapid increase during the transient state of the black curve is due to the large speed gain of the small to tiny disks, which would have caused the increases of both the flip and collision events. The two sharp peaks in Fig. 22b correspond to the situation where a tiny disk is captured and bounces back and forth several times between the container boundary and a very large disk.

8 Application: Collision-Free Path Planning for a Swarm of Moving Vehicles

The dynamic Voronoi diagram can be a vital tool for collision-free path planning of moving objects in space. In addition to well-known crowd and dancing simulation, there are examples such as conjunction prediction for space situational awareness in geo-space, midair airplane collision prediction, prediction of collisions between flying drones, and collisions between underwater objects.

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. 23 shows collision-free path planning among five moving disks in the plane. See the

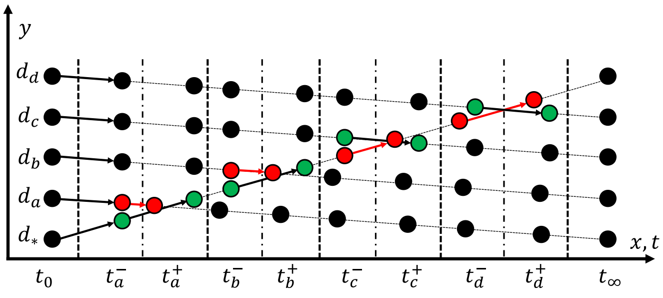


Fig. 23. Collision avoiding path planning for a swarm of drones (See the accompanying video). Five drones are moving linearly through the dotted line segments during the period from t_0 to t_∞. The predicted collision between d_* and d_a at t_a is avoided by slowing down the speed of d_a (red) while keeping the speed for d_* (green) the same. The other predicted collisions at t_b, t_c, and t_d can be similarly avoided.

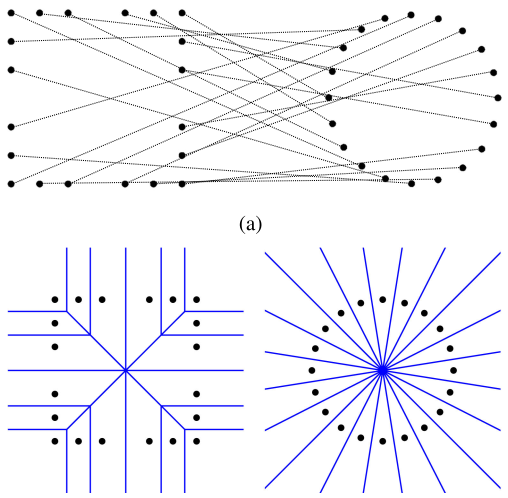


Fig. 24. (a) The tiny disks initially positioned on a square moves linearly through the black dotted line segments to arrive at the circle. (b) and (c) The Voronoi diagrams.

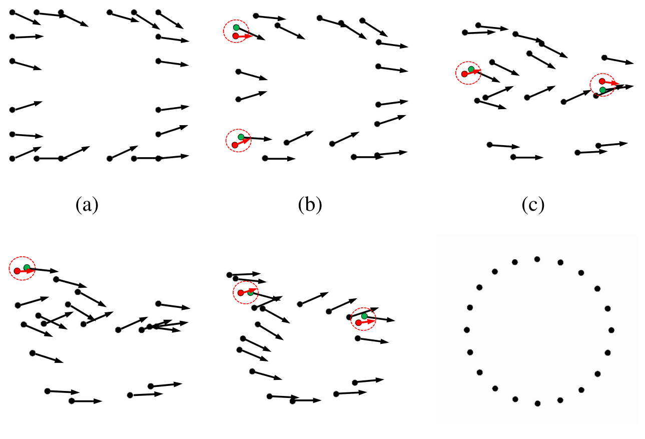


Fig. 26. Collision-free vehicle motion plan produced using a dynamic Voronoi diagram. The pair of red and green dots in each red ellipse is the vehicles that need to adjust speeds to avoid a predicted collision. (a) Initial scene ($t_0 = 0$), (b), (c), (d), (e) intermediate scenes (t_a, t_b, t_c, and t_c, resp.), (f) final scene (t^∞).

accompanying video. 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_∞ 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_∞ 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 $\mathscr{VD}(t_0)$ 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 $\mathscr{VD}(t_0)$. 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 $\mathscr{VD}(t_a^+)$. 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.

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, which we will leave for a future paper to report.

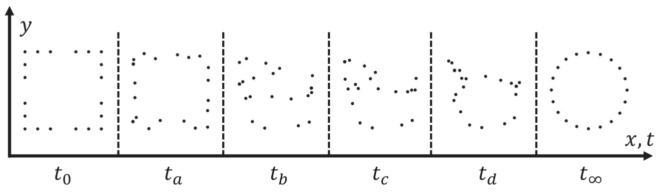


Fig. 25. Motion planning of 20 vehicles through collision-free paths. Each dot represents a linearly moving vehicle. The scene transforms from the initial scene (at t_0) to the final scene (at t_∞) after going through intermediate scenes (t_a, t_b, t_c, and t_c). Each region bounded by a dotted vertical separator corresponds to the space where the vehicles are located at the corresponding moment. Hence, the horizontal axis corresponds to both t and the x-coordinate.

See Fig. 24a. We want to transform twenty objects from the configuration of a rectangle at t_0 to that of a circle at t_∞. 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. 24b and 24c show the Voronoi diagrams of the objects in the two configurations. Fig. 25 shows some intermediate states of the transformation at $t_0 < t_a$, t_b, t_c, and $t_d < t_\infty$. Fig. 26 shows the step-by-step of Fig. 25. Note that the red and green objects in each red ellipse denote that their speeds are modified to avoid anticipated collisions.

9 Conclusion

Dynamic Voronoi diagram is useful for understanding the spatial properties of moving objects. Here, we propose a topology-oriented event-increment algorithm and its data structure for robust and efficient construction of a Voronoi diagram of moving disks. The main idea is to find all moments of the Voronoi diagram's topology change over time horizon and store the moments with related data in an event history. By scanning the event history, a Voronoi diagram at a specific moment in time horizon can be quickly constructed. The proposed algorithm was implemented and tested. The dynamic Voronoi diagram will be a useful platform for unmanned vehicle traffic management.

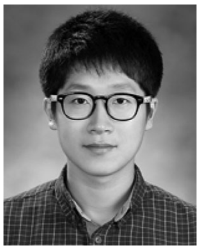
Acknowledgments

This work was supported by the National Research Foundation of Korea (MSIT) [Nos. 2017R1A3B1023591 and 2016 K1A4A3914691]. C. Song and J. Cha contributed equally to this work.

References

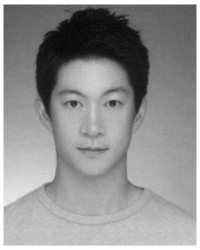
[1] F. R. Hoots, L. L. Crawford, and R. L. Roehrich, "An analytic method to determine future close approaches between satellites," *Celestial Mech.*, vol. 33, no. 2, pp. 143–158, 1984.
[2] J. R. Alarcón-Rodríguez, F. M. Martínez-Fadrique, H. Klinkrad, A. H. Rudolf, and F. B. de Frescheville, "Conjunction event predictions for operational ESA satellites," in *Proc. Space-Ops Conf.*, 2004, pp. 1–11.
[3] D. Casanova, C. Tardioli, and A. Lemaître, "Space debris collision avoidance using a three-filter sequence," *Monthly Notices*, vol. 442, no. 4, pp. 3235–3242, 2014.
[4] J. Cha *et al.*, "DVD-COOP: Innovative conjunction prediction using Voronoi-filter based on the dynamic Voronoi diagram of 3D spheres," in *Proc. Adv. Maui Opt. Space Surveillance Technol. Conf.*, 2017, pp. 1–15.
[5] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 179–189, Dec. 2000.
[6] M. Prandini, J. Hu, J. Lygeros, and S. Sastry, "A probabilistic approach to aircraft conflict detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 4, pp. 199–220, Dec. 2000.
[7] A. Alonso-Ayuso, L. F. Escudero, and F. J. Martn-Campo, "A VNS metaheuristic for solving the aircraft conflict detection and resolution problem by performing turn changes," *J. Global Optim.*, vol. 63, no. 3, pp. 583–596, 2015.
[8] D. Kim, Y. Cho, and D.-S. Kim, "Region expansion by flipping edges for Euclidean Voronoi diagrams of 3D spheres based on a radial data structure," in *Proc. Int. Conf. Comput. Sci. Appl.*, 2005, pp. 716–725.
[9] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, 2nd ed. Chichester, U.K.: Wiley, 1999.
[10] M. Sharir, "Intersection and closest-pair problems for a set of planar discs," *SIAM J. Comput.*, vol. 14, no. 2, pp. 448–468, 1985.
[11] D.-S. Kim, D. Kim, and K. Sugihara, "Voronoi diagram of a circle set from Voronoi diagram of a point set: I. Topology," *Comput. Aided Geometric Design*, vol. 18, pp. 541–562, 2001.
[12] D.-S. Kim, D. Kim, and K. Sugihara, "Voronoi diagram of a circle set from Voronoi diagram of a point set: II. Geometry," *Comput. Aided Geometric Design*, vol. 18, pp. 563–585, 2001.
[13] M. Lee, K. Sugihara, and D.-S. Kim, "Topology-oriented incremental algorithm for the robust construction of the Voronoi diagrams of disks," *ACM Trans. Math. Softw.*, vol. 43, no. 2, pp. 14:1–14:23, 2016.
[14] I. G. Gowda, D. G. Kirkpatrick, D. T. Lee, and A. Naamad, "Dynamic Voronoi diagrams," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 5, pp. 724–731, Sep. 1983.
[15] O. Devillers, M. Golin, K. Kedem, and S. Schirra, "Revenge of the dog: Queries on Voronoi diagrams of moving points," in *Proc. 6th Can. Conf. Comput. Geom.*, 1994, pp. 1–14.
[16] O. Devillers, M. Golin, K. Kedem, and S. Schirra, "Queries on Voronoi diagrams of moving points," *Comput. Geom. - Theory Appl.*, vol. 6, no. 5, pp. 315–327, 1996.
[17] T. Roos, "Voronoi diagrams over dynamic scenes," *Discr. Appl. Math.*, vol. 43, no. 3, pp. 243–259, 1993.
[18] K. B. Lauritsen, H. Puhl, and H.-J. Tillemans, "Performance of random lattice algorithms," *Int. J. Modern Phys. C*, vol. 5, no. 6, pp. 909–922, 1994.
[19] G. D. Fabritiis and P. V. Coveney, "Dynamical geometry for multiscale dissipative particle dynamics," *Comput. Phys. Commun.*, vol. 153, no. 2, pp. 209–226, 2003.
[20] G. Schaller and M. Meyer-Hermann, "Kinetic and dynamic delaunay tetrahedralizations in three dimensions," *Comput. Phys. Commun.*, vol. 162, no. 1, pp. 9–23, 2004.
[21] M. Gavrilova, J. Rokne, and D. Gavrilov, "Dynamic collision detection algorithms in computational geometry," in *Proc. 12th Eur. Workshop Comput. Geom.*, 1996, pp. 103–106.
[22] M. Gavrilova and J. Rokne, "Swap conditions for dynamic Voronoi diagrams for circles and line segments," *Comput. Aided Geometric Design*, vol. 16, no. 2, pp. 89–106, 1999.
[23] M. L. Gavrilova and J. Rokne, *Collision Detection Optimization in a Multi-particle System*. Berlin, Germany: Springer, 2002.
[24] L. J. Guibas, F. Xie, and L. Zhang, "Kinetic collision detection: Algorithms and experiments," in *Proc. Int. Conf. Robot. Autom.*, 2001, pp. 2903–2910.
[25] M. Vigo, N. Pla, and J. Cotrina, "Regular triangulations of dynamic sets of points," *Comput. Aided Geometric Design*, vol. 19, no. 2, pp. 127–149, 2002.
[26] T. Beyer, G. Schaller, A. Deutsch, and M. Meyer-Hermann, "Parallel dynamic and kinetic regular triangulation in three dimensions," *Comput. Phys. Commun.*, vol. 172, no. 2, pp. 86–108, 2005.
[27] D. Kim, M. Lee, Y. Cho, and D.-S. Kim, "Beta-complex versus alpha-complex: Similarities and dissimilarities," *IEEE Trans. Vis. Comput. Graphics*, to be published, doi: 10.1109/TVCG.2018.2873633.
[28] M. I. Karavelas, "Proximity structures for moving objects in constrained and unconstrained environments," PhD dissertation, Stanford University, CA, USA, Aug. 2001.
[29] M. I. Karavelas, "Voronoi diagrams for moving disks and applications," in *Proc. 7th Workshop Algorithms Data Struct.*, 2001, pp. 62–74.
[30] M. Karavelas and M. Yvinec, "Dynamic additively weighted Voronoi diagrams in 2D," in *Proc. Eur. Symp. Algorithms*, 2002, pp. 586–598.
[31] M. Gavrilova and J. Rokne, "Updating the topology of the dynamic Voronoi diagram for spheres in Euclidean d-dimensional space," *Comput. Aided Geometric Design*, vol. 20, no. 4, pp. 231–242, 2003.
[32] E. Kreyszig, *Advanced Engineering Mathematics*. Hoboken, NJ, USA: Wiley, 2010.
[33] D.-S. Kim, D. Kim, Y. Cho, and K. Sugihara, "Quasi-triangulation and interworld data structure in three dimensions," *Comput.-Aided Design*, vol. 38, no. 7, pp. 808–819, 2006.
[34] D.-S. Kim, Y. Cho, and K. Sugihara, "Quasi-worlds and quasi-operators on quasi-triangulations," *Comput.-Aided Design*, vol. 42, no. 10, pp. 874–888, 2010.
[35] D.-S. Kim, Y. Cho, K. Sugihara, J. Ryu, and D. Kim, "Three-dimensional beta-shapes and beta-complexes via quasi-triangulation," *Comput.-Aided Design*, vol. 42, no. 10, pp. 911–929, 2010.
[36] D. Kim, D.-S. Kim, and K. Sugihara, "Euclidean Voronoi diagram for circles in a circle," *Int. J. Comput. Geom. Appl.*, vol. 15, no. 2, pp. 209–228, 2005.
[37] F. Aurenhammer, R. Klein, and D.-T. Lee, *Voronoi Diagrams and Delaunay Triangulations*. Singapore: World Scientific, 2013.
[38] L. F. Henderson, "The statistics of crowd fluids," *Nature*, vol. 229, pp. 381–383, 1971.
[39] B. D. Lubachevsky and F. H. Stillinger, "Geometric properties of random disk packing," *J. Statist. Phys.*, vol. 60, no. 5, pp. 561–583, 1990.
[40] B. D. Lubachevsky, "How to simulate billiards and similar systems," *J. Comput. Phys.*, vol. 94, no. 2, pp. 255–283, 1991.
[41] B. D. Lubachevsky, F. H. Stillinger, and E. N. Pinson, "Disks versus spheres: Contrastion properties of random packings," *J. Statist. Phys.*, vol. 64, no. 3, pp. 501–524, 1991.
[42] J. Reif and M. Sharir, "Motion planning in the presence of moving obstacles," *J. ACM*, vol. 41, no. 4, pp. 764–790, 1994.
[43] A. Yang, W. Naeem, M. Fei, and X. Tu, "A cooperative formation based collision avoidance approach for a group of autonomous vehicles," *Int. J. Adaptive Control Signal Process.*, vol. 31, pp. 489–506, 2017.
[44] A. Sud, E. Andersen, S. Curtis, M. C. Lin, and D. Manocha, "Real-time path planning in dynamic virtual environments using multiagent navigation graphs," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 3, pp. 526–538, Jun. 2008.
[45] R. Geraerts and M. H. Overmars, "Creating high-quality roadmaps for motion planning in virtual environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 4355–4361.
[46] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1999, pp. 1024–1031.

[47] R. Narain, A. Golas, S. Curtis, and M. C. Lin, "Aggregate dynamics for dense crowd simulation," *ACM Trans. Graphics*, vol. 28, no. 5, 2009, Art. no. 122.
[48] D. Wolinski, M. C. Lin, and F. Pettré, "WarpDriver: Context-aware probabilistic motion prediction for crowd simulation," *ACM Trans. Graphics*, vol. 35, no. 6, 2016, Art. no. 164.
[49] O. D. Gyves, L. Toledo, and I. Rudomín, "Proximity queries for crowd simulation using truncated Voronoi diagrams," in *Proc. Motion Games*, 2013, pp. 87–92.
[50] B. C. Ricks and P. K. Egbert, "A whole surface approach to crowd simulation on arbitrary topologies," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 2, pp. 159–171, Feb. 2014.
[51] A. Golas, R. Narain, S. Curtis, and M. C. Lin, "Hybrid long-range collision avoidance for crowd simulation," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 7, pp. 1022–1034, Jul. 2014.
[52] Y. Suma, D. Yanagisawa, and K. Nishinari, "Anticipation effect in pedestrian dynamics: Modeling and experiments," *Physica A*, vol. 391, pp. 248–263, 2012.
[53] I. Karamouzas, N. Sohre, R. Hu, and S. J. Guy, "Crowd space: A predictive crowd analysis technique," *ACM Trans. Graphics*, vol. 37, no. 6, 2018, Art. no. 186.
[54] D. Wolinski, S. J. Guy, A.-H. Olivier, M. Lin, D. Manocha, and J. Pettré, "Parameter estimation and comparative evaluation of crowd simulations," *Comput. Graphics Forum*, vol. 33, no. 2, pp. 303–312, 2014.
[55] M. Kapadia, A. Beacco, F. Garcia, V. Reddy, N. Pelechano, and N. I. Badler, "Multi-domain real-time planning in dyanmic environments," in *Proc. 12th ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2013, pp. 115–124.
[56] G. Berseth, M. Kapadia, B. Haworth, and P. Faloutsos, "SteerFit: Automated parameter fitting for steering algorithms," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*, 2014, pp. 113–122.
[57] I. Karamouzas, P. Heil, P. van Beek, and M. H. Overmars, "A predictive collision avoidance model for pedestrian simulation," in *Proc. Int. Workshop Motion Games*, 2009, pp. 41–52.
[58] A. Sud, E. Andersen, S. Curtis, and M. C. Lin, "Real-time path planning in dynamic virtual environments using multiagent navigation graphs," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 3, pp. 526–538, May/Jun. 2008.
[59] R. Alizadeh, "A dynamic cellular automation model for evacuation process with obstacles," *Safety Sci.*, vol. 49, no. 2, pp. 315–323, 2011.
[60] P. A. Thompson and E. W. Marchant, "A computer model for the evacuation of large building populations," *Fire Safety J.*, vol. 24, no. 2, pp. 131–148, 1995.
[61] B. Segre and K. Mahler, "On the densest packing of circles," *Amer. Math. Monthly*, vol. 51, no. 5, pp. 261–270, 1944.
[62] Z. Drezner and E. Erkut, "Solving the continuous p-dispersion problem using non-linear programming," *J. Oper. Res. Soc.*, vol. 46, no. 4, pp. 516–520, 1995.
[63] C. D. Maranas, C. A. Floudas, and P. M. Pardalos, "New results in the packing of equal circles in a square," *Discr. Math.*, vol. 142, no. 1–3, pp. 287–293, 1995.
[64] J. A. George, J. M. George, and B. W. Lamar, "Packing different-sized circles into a rectangular container," *Eur. J. Oper. Res.*, vol. 84, no. 3, pp. 693–712, 1995.
[65] W. Huang and R. Xu, "Two personification strategies for solving circles packing problem," *Sci. China (Ser. E)*, vol. 42, no. 6, pp. 595–602, 1999.
[66] Z. Zeng, X. Yu, K. He, W. Huang, and Z. Fu, "Iterated tabu search and variable neighborhood descent for packing unequal circles into a circular container," *Eur. J. Oper. Res.*, vol. 250, no. 2, pp. 615–627, 2016.
[67] C. López and J. Beasley, "Packing unequal circles using formulation space search," *Comput. Oper. Res.*, vol. 40, no. 5, pp. 1276–1288, 2013.
[68] H. Akeb, M. Hifi, and R. M'Hallah, "A beam search algorithm for the circular packing problem," *Comput. Oper. Res.*, vol. 36, no. 5, pp. 1513–1528, 2009.
[69] Z. Lu and W. Huang, "Perm for solving circle packing problem," *Comput. Oper. Res.*, vol. 35, no. 5, pp. 1742–1755, 2008.
[70] M. Hifi and R. M'Hallah, "A dynamic adaptive local search algorithm for the circular packing problem," *Eur. J. Oper. Res.*, vol. 183, no. 3, pp. 1280–1294, 2007.
[71] E. Birgin and F. Sobral, "Minimizing the object dimensions in circle and sphere packing problems," *Comput. Oper. Res.*, vol. 35, no. 7, pp. 2357–2375, 2008.
[72] K. Sugihara, M. Sawai, H. Sano, D.-S. Kim, and D. Kim, "Disk packing for the estimation of the size of a wire bundle," *Japan J. Ind. Appl. Math.*, vol. 21, no. 3, pp. 259–278, 2004.
[73] E. Specht, "A precise algorithm to detect voids in polydisperse circle packings," *Proc. Roy. Soc.*, vol. 471, no. 2182, pp. 1–19, 2016.
[74] H. Mahboubi and A. G. Aghdam, "An energy-efficient strategy to improve coverage in a network of wireless mobile sensors with nonidentical sensing ranges," in *Proc. Veh. Technol. Conf.*, Jun. 2013, pp. 1–5.
[75] X.-Y. Li, P.-J. Wan, and O. Frieder, "Coverage in wireless ad hoc sensor networks," *IEEE Trans. Comput.*, vol. 52, no. 6, pp. 753–763, Jun. 2003.
[76] G. Gupta, R. K. Ghosh, and S. V. Rao, "A routing algorithm for multi-hop mobile ad-hoc networks using weighted delaunay triangulation," in *Proc. Int. Conf. Inf. Technol.*, 2003, pp. 1–6.
[77] K. Vu and R. Zheng, "Geometric algorithms for target localization and tracking under location uncertainties in wireless sensor networks," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 1835–1843.
[78] M. Barnett-Jones, P. A. Dickinson, M. J. Godfrey, T. Grundy, and M. A. Moore, "Transition state theory and the dynamics of hard disks," *Phys. Rev. E*, vol. 88, no. 5, pp. 1–5, 2013.
[79] M. N. Bannerman, S. Strobl, A. Formella, and T. Póschel, "Stable algorithm for event detection in event-driven particle dynamics," *Comput. Particle Mech.*, vol. 1, no. 2, pp. 191–198, 2014.
[80] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Trans. Graphics*, vol. 36, no. 4, 2017, Art. no. 132.
[81] Q. Galvane, "Directing cinematographic drones," *ACM Trans. Graphics*, vol. 37, no. 3, 2018, Art. no. 34.
[82] K. Xie *et al.*, "Creating and chaining camera moves for quadrotor videography," *ACM Trans. Graphics*, vol. 37, no. 4, 2018, Art. no. 88.
[83] G. Gordon, *System Simulation*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice Hall, 1978.
[84] B. K. Choi and D. Kang, *Modeling and Simulation of Discrete Event Systems*, 1st ed. Hoboken, NJ, USA: Wiley, 2013.
[85] M. Mäntylä, *An Introduction to Solid Modeling*. New York, NY, USA: Freeman, 1988.
[86] K. Lee, *Principles of CAD/CAM/CAE Systems*. Boston, MA, USA: Addison-Wesley, 1999.
[87] D.-S. Kim, Y. Cho, and D. Kim, "Euclidean Voronoi diagram of 3D balls and its computation via tracing edges," *Comput.-Aided Design*, vol. 37, no. 13, pp. 1412–1424, 2005.



Chanyoung Song received the BS degree from the Department of Industrial Engineering, in 2014, and is currently working toward the PhD degree in the School of Mechanical Engineering, Hanyang University. His research interest include the theory and application of computational geometry, especially the Voronoi diagram of disks and spheres.

Jehyun Cha received the BS degree from the Department of Industrial Engineering, Hanyang University, Seoul, South Korea, in 2013. He is currently working toward the PhD degree in the School of Mechanical Engineering, Hanyang University, Seoul, South Korea. His research interests include developing and applying the theory and software library for geometry of moving objects in 3D using the dynamic Voronoi diagram.



Mokwon Lee received the BS and PhD degrees from Hanyang University, in 2012 and 2019, respectively. He is a postdoctoral researcher with Voronoi Diagram Research Center, Hanyang University, Seoul, South Korea. His research interest include computational geometry.

Deok-Soo Kim received the BS degree from Hanyang University, South Korea, in 1982, the MS degree from the New Jersey Institute of Technology, in 1985, and the PhD degree from the University of Michigan, in 1990. He is currently a professor with the School of Mechanical Engineering, Hanyang University, South Korea. Before, he joined the University in 1995, he worked with Applicon, and Samsung Advanced Institute of Technology, Korea. His current research interests include the theory and applications of Voronoi diagrams.