Problems on Planar Voronoi Diagrams

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Ioannis Mantas

under the supervision of
Prof. Evanthia Papadopoulou

January 2022

Dissertation Committee

Prof. Piotr Didyk	Università della Svizzera italiana, Lugano, Switzerland
Prof. Ioannis Emiris	National and Kapodistrian University of Athens, and "Athena" Research Center, Athens, Greece
Prof. Luca Gambardella	Universita della Svizzera italiana, Lugano, Switzerland
Prof. Rodrigo Silveira	Universitat Politècnica de Catalunya, Barcelona, Spain

Dissertation accepted on January 2022

Research Advisor	PhD Program Director
Prof. Evanthia Papadopoulou	**The PhD program Director *pro tempore***

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Ioannis Mantas
Lugano, January 2022

Abstract

Computational Geometry is the field of Computer Science that studies algorithmic problems which can be expressed in terms of Geometry. A geometric structure that has attracted the interest of researchers over the last centuries is the *Voronoi diagram*. It is a powerful geometric object which has diverse applications on problems where proximity information is required. Given a set of sites, their Voronoi diagram is the subdivision of the space into regions, such that all points in a region have the same nearest site. Many generalizations of this simple concept have been considered, including generalized sites, spaces and distance functions. In this dissertation we study three problems related to generalized planar Voronoi diagrams.

The first topic is related to *color Voronoi diagrams*, where each site is a *cluster* of points and the distance between a point and a cluster is the minimum Euclidean distance. Color Voronoi diagrams are motivated by facility location problems and sampling based approximation schemes for Voronoi diagrams. We focus on the *farthest color Voronoi diagram*, which is a *min-max* type of Voronoi diagram. We present combinatorial properties, conditions for the diagram to have linear complexity, and efficient construction algorithms.

Secondly, we study the *rotating rays Voronoi diagram*, a Voronoi structure where the input sites are rays and the distance between a point and a ray is an oriented angular distance. We demonstrate how such a diagram finds applications in various floodlight illumination problems; coverage problems where a domain has to be covered by floodlights-wedges. Motivated by these illumination problems, we study the rotating rays Voronoi diagram in various domains of interest, as for example the plane, polygons, and curves. For all the domains considered we present combinatorial and algorithmic results.

Finally, we consider a well-known deterministic linear-time algorithmic scheme for Voronoi diagrams. We generalize a combinatorial result on selecting leaves in embedded binary trees that the scheme requires, aiming to make this algorithmic technique applicable to larger classes of planar Voronoi diagrams.

Acknowledgements

With the completion of this dissertation, a beautiful long journey has come to an end. A journey that has been "full of adventure, full of discovery". A journey filled with positive moments, joy and excitement, but also many times with frustration and disappointment. Living in Lugano, surrounded by such a charming landscape, in a nice country like Switzerland, surely played its role in the positive outcome of this journey. But still, if it had not been for the nice people that I had the chance to meet and interact with, this journey would not have been the same. So, taking this chance, I list some people that I would like to thank.

First, and foremost, I want to thank my advisor Evanthia Papadopoulou. I feel lucky that Evanthia selected me to join her research group in such a stimulating environment at Università della Svizzera italiana. Evanthia did her best in appropriately guiding me throughout my dissertation, and it was with her guidance that this dissertation has been successfully completed, and that I have become a better researcher, but also a better person. I am also particularly thankful to Evanthia because she gave me the freedom to collaborate with other people, to choose some of the problems I worked on, and also to travel around, in order to enrich my knowledge and to conduct research.

I also feel very grateful to all four members of my dissertation committee for the time they devoted to my dissertation. First, I want to thank Piotr Didyk and Luca Gambardella, the internal committee members, for the interest they showed and their feedback, although they have not been working on similar topics. Additionally, I want to thank a lot Ioannis Emiris who definitely influenced me in starting this dissertation. As an undergraduate student, I took three of his courses (including a Computational Geometry one), all three with some research twist, so his positive influence cannot be questioned. My sincere thanks also go to Rodrigo Silveira. Apart from his (direct) positive influence by co-authoring a paper, he indirectly had an even bigger positive influence by organizing a very inspiring research event in Barcelona, where I was hosted for 5 weeks. In this event, I had the opportunity to meet and collaborate with many interesting people, something that shaped greatly the course of my dissertation.

All these years in the office would not have been the same without my two colleagues, and friends, Kolja Junginger and Martin Suderland. I am really happy for spending my everyday life with them, by having lunch, drinking beers, playing board games, going for hikes and even going for holidays together. Our mutual support and understanding was key in the successful outcome of our dissertations, and very importantly, we also had an excellent collaboration which resulted in several interesting publications.

I feel privileged to have had the chance to meet, interact and collaborate with so many people while working on this dissertation. I want to thank them all individually. Chee Yap for his stay in Lugano which was full of new research insights and interesting discussions about life; it is always a pleasure to discuss with Chee. Vera Sacristán, who together with Rodrigo, organized the aforementioned inspiring event and welcomed me in Barcelona. Carlos Seara for his humour and creativity in posing new problems (which we later tackled), and Carlos Alegría for (accidentally) initiating a very satisfying collaboration, by asking me (without knowing me) a question about Voronoi diagrams. Marko Savić for welcoming me in his hometown, Novi Sad, and finally, Hendrik Schrezenmaier with whom the three of us (together with Marko) had very nice long working sessions.

Apart from being a period full of research it has also been a period full of good times. And during these good times, I had the chance to share beautiful moments with numerous people. I want to thank everyone for the walks by the lake, the hikes, the nights out, the discussions, the arguments, the drinks, and everything else. I should particularly mention Esteban, Vasilis, Panos and Giorgos with whom I shared such good times for a particularly long period. Also, many thanks to (the last generation of) the Openspace people as a whole.

Many thanks also go to Dimitris for his advice and support especially during the difficult times. A very special person, that I am very content for having by my side is Chiara. I am grateful to her for all the beautiful moments, the affection and the support; Chiara is a wonderful person, and my life in Lugano would not have been the same without her. Finally, I need to thank my big beautiful family, my parents and all my siblings. My family environment has always been a safe haven, and it is this family environment that shaped me to become who I am, and this is something to embrace and cherish.

Contents

Figures

Chapter 1

Introduction

Our daily lives are heavily influenced by computers: we carry with us computers, we work with computers, and we entertain ourselves using computers. These computers continuously perform computations to serve our needs, and we require them not only to work *correctly*, but also to work *fast*. Underlying, behind all these computations are *algorithms*; well-defined procedures that given a certain input, they will produce a certain output. Algorithms are the roadmaps which computers use to accomplish any given task. In the world of *Computer Science and Engineering*, the quest for *correct and efficient algorithms* is neverending.

Many tasks, or problems, involve discrete geometric objects. Algorithms dealing with such problems are called *geometric algorithms*, and the corresponding field of Computer Science is termed *Computational Geometry*. One of the most famous objects in Computational Geometry is the *Voronoi diagram*. The Voronoi Diagram is a versatile geometric structure encoding proximity information among a given set of objects called *sites*. It subdivides the underlying space into maximal regions with respect to the neighboring sites, according to a given distance function.

In its simplest form, the Voronoi diagram is defined on a set of points in the Euclidean plane; it is the partition of the plane into regions such that any two points in a region share the same nearest point-site. See an example of a classic Voronoi diagram of 10 points in Fig. 1.1.

This simple concept can be generalized in numerous ways. For instance, the sites instead of points, may be circles, polygons, or point-clusters. The underlying space need not be $\mathbb{R}^2$; they are often studied in higher dimensions, but also in other non Euclidean spaces. The distance function between a point in space and a site can also vary; for example any L_p distance can be considered, or if the sites

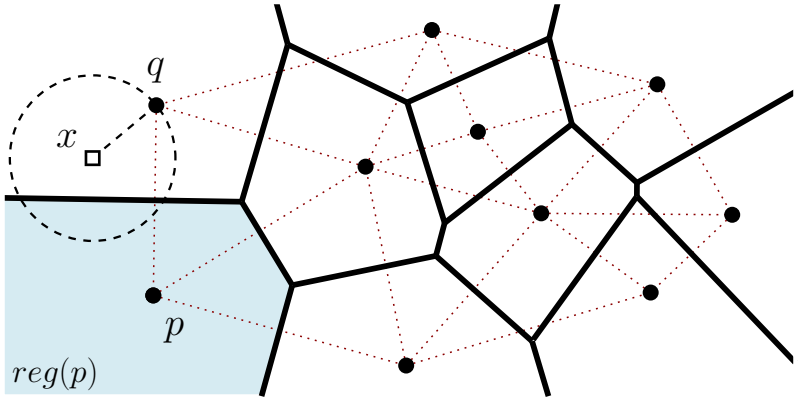


Figure 1.1. The nearest Voronoi diagram of a set of 10 points in $\mathbb{R}^2$. The corresponding graph structure is shown with thick segments. The nearest Voronoi region of a point-site p ($reg(p)$) is highlighted. The Euclidean distance between a point x to its nearest site (point q) is shown dashed. The thin dotted segments illustrate the Delaunay triangulation of the same set of points.

are *more complex*, the distance can have various interpretations, as for example the minimum or the maximum distance.

Voronoi diagrams and their generalizations find applications in different areas of Computer Science and Engineering, as for instance in surface reconstruction, operations research, and geographic information systems. They also find applications in diverse sciences, as for example in Geology, Biology, and Chemistry. A comprehensive reference with many applications of Voronoi diagrams is the book of Okabe, Boots, Sugihara, and Chiu [2009]. Another valuable reference is the book of Aurenhammer, Klein, and Lee [2013], which offers an extensive list of combinatorial and algorithmic results for various Voronoi diagrams.

Voronoi diagrams have a long history. They owe its name to Georgy Voronoy (1868-1908) who formalized these structures in [1908a; 1908b], although earlier appearances date back to René Descartes (1596 - 1650) in [1644], and to Lejeune Dirichlet (1805 - 1859) in [1850]. Another object with a long history, closely related to the Voronoi diagram is the *Delaunay Triangulation*. It was first introduced and studied by Boris Delaunay (1890 – 1980) in [1934] and it is the *geometric dual* structure of the Voronoi diagram. An example of a Delaunay triangulation is shown in Fig. 1.1. A nice historical overview of Voronoi diagrams is given in the book of Okabe et al. [2009], and a discussion on the popularization of the term "Voronoi diagram" can be found in the memoir of Shamos [1999].

1.1 Basic concepts of Voronoi diagrams

In this section, we review some basic concepts and results related to Voronoi diagrams, before proceeding to the specific Voronoi diagrams which we consider. A standard nearest Voronoi diagram can be formally defined as follows.

Definition 1.1. The *nearest Voronoi diagram* of a set of sites, in a given space, is the subdivision of this space into maximal regions such that all points within one region have the same nearest site. These regions are called *Voronoi regions*.

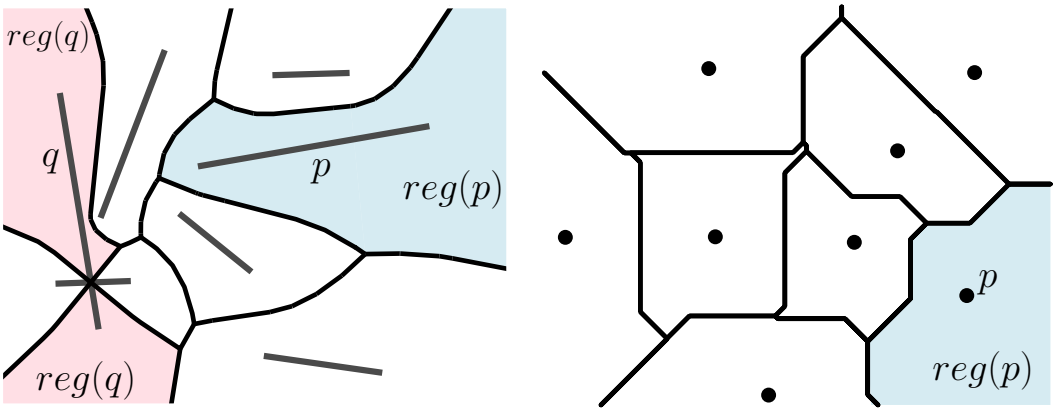
In the simplest form, the input sites are points in the plane, and the distance considered is the *Euclidean distance*[1]. This diagram is well-studied, its properties are known and there exist construction algorithms with optimal time complexity. More specifically, given a set of n points in $\mathbb{R}^2$, the nearest point Voronoi diagram has $O(n)$ combinatorial complexity. Each Voronoi region is a non-empty convex polygonal region containing containing the corresponding point-site. An edge on the boundary of two adjacent Voronoi regions (called a *Voronoi edge*) is a line-segment, or a ray, that is part of the bisector of the two corresponding point-sites. Further, only point-sites on the *convex hull*[2] have unbounded Voronoi regions. The above can be observed in the diagram of Fig. 1.1.

There are many construction algorithms which employ different paradigms, as for example *divide & conquer*, *plane sweep*, *lift-up to 3-space*, and *(randomized) incremental construction*. For the nearest Voronoi diagram of points, many of these algorithms take $O(n \log n)$ time, which is optimal, as there exists an $\Omega(n \log n)$ lower bound in the time required to construct the diagram. We review some of these algorithmic techniques in more detail in Section 2.1.

As already mentioned the Voronoi diagram is the geometric dual of the, equally famous, *Delaunay triangulation*. Given the Voronoi diagram, simply consider a line segment between any two point-sites that have adjacent Voronoi regions; this forms the Delaunay triangulation of the set of points. This duality can be observed in Fig. 1.1. The Delaunay triangulation has the property that the smallest angle among all triangles is maximized, avoiding essentially the formation of *thin* triangles, which are often not desirable.

[1]The *Euclidean distance* between two points in the plane $p = \{p_x, p_y\}$ and $q = \{q_x, q_y\}$ is $d(p,q) := \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$

[2]The *convex hull* of a set of points is the smallest convex set containing this set of points. An example of a convex hull is illustrated in Fig. 1.4a



(a) A set of 7 line segments. (b) A set of 8 points with the L_∞ distance.

Figure 1.2. Two examples of generalized nearest site Voronoi diagrams.

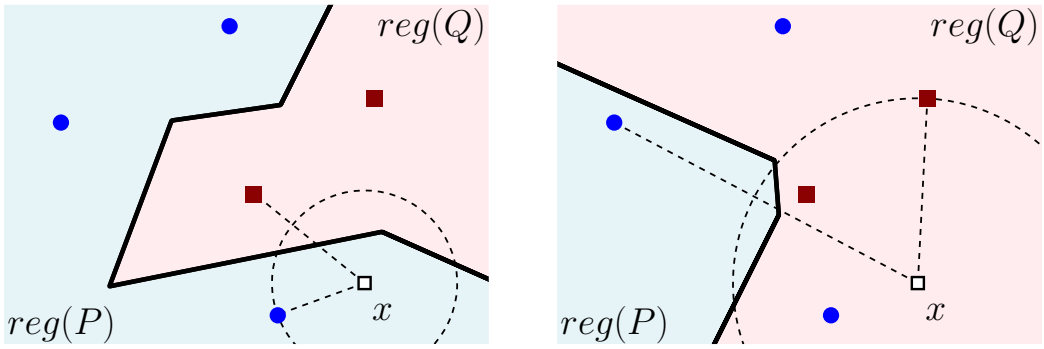
Generalized Voronoi diagrams

Voronoi diagrams can be generalized in many different ways, and by examining Definition 1.1, we can observe that some terms, as "*nearest*", "*sites*" and "*space*", can have multiple interpretations. Each such interpretation results in a different generalized Voronoi diagram. We now look into some common generalizations, and give some indicative results and references from the literature.

Input sites. So far we considered points as sites, which are quite simple. The input sites may consist of any geometric object, as for instance, line segments, circles, or clusters (sets) of points. More complex input sites often lead to more challenging structures with higher complexity, as more complex are bisectors involved, which may consist of curves that are non-linear, disconnected, or closed. Recall that Voronoi edges are induced by bisectors of input sites.

As an example observe a Voronoi diagram of line segments in Fig. 1.2a; a Voronoi region can have many connected components (when line segments intersect), and the boundary of two Voronoi regions contains also parabolic arcs. The diagram of n line segments has $O(n+I)$ complexity, where is I the number of intersections among the segments. It can be constructed: in $O(n \log n)$ time if the segments are pairwise non-intersecting, or in $O(n\alpha(n)\log n + I)$ time if the segments intersect; see, Yap [1987] and Bae [2016].

Indicatively, other examples of input sites include *curved objects*, as e.g., in Alt et al. [2005] and Emiris et al. [2006], or *clusters of objects* as, e.g., in Papadopoulou [2004] and Bae [2014].



(a) Using the minimum cluster distance. (b) Using the maximum cluster distance.

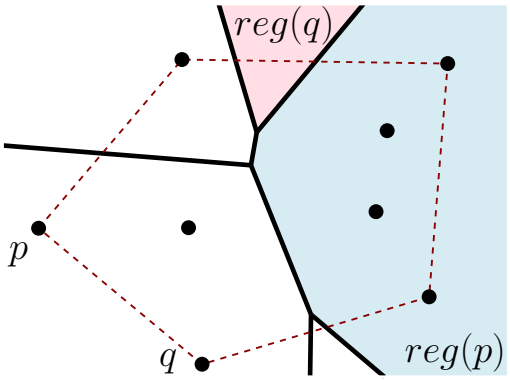
Figure 1.3. The nearest Voronoi diagrams of 2 clusters of points (cluster P(●) with 3 points and cluster Q(■) with 2 points) under different distance functions. The distance of a point x to its nearest site is shown dashed.

Distance function. Different Voronoi diagrams occur when different distance measures are considered. Most commonly the Euclidean distance L_2 is used. Other common measures include the L_1 (*Manhattan*) or L_∞ distances, or in general any L_p distance function. Actually, Voronoi diagrams using the L_1 and L_∞ distances, are related under a $\pi/2$-rotation. In Fig. 1.2b we illustrate a Voronoi diagram of points using the L_∞ distance. See Lee and Wong [1980] or Papadopoulou and Lee [2001] for two instances of diagrams using the L_∞ distance.

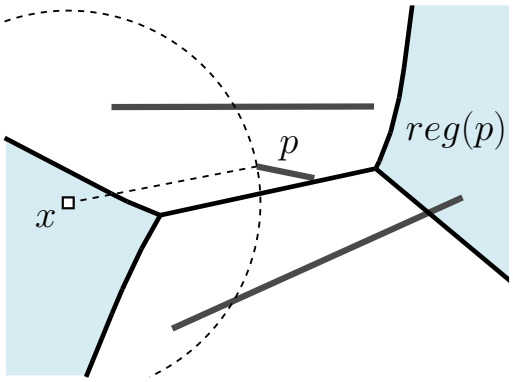
Additionally, other common distance functions, include *polyhedral distance functions*, as in Boissonnat et al. [1998] and Aurenhammer et al. [2021], and distance functions which associate weights (multiplicatively or additive) to each site; see, e.g., Aurenhammer and Edelsbrunner [1984] or Aurenhammer [1987].

When certain generalized input sites are considered the notion of the distance between a point and an input site can be interpreted in different ways. For example, given a cluster of points, the distance considered might be the *minimum (Euclidean) distance* or the *maximum (Euclidean) distance*. In Fig. 1.3 the nearest Voronoi diagrams of 2 point-clusters using the minimum and the maximum distance are illustrated. Most commonly the minimum distance is used but there are examples of Voronoi diagrams of generalized input sites that use the maximum distance; see e.g., Papadopoulou and Lee [2004] for point-clusters as sites, or Setter et al. [2010] for circles as sites.

Underlying space. Apart from the Euclidean space, Voronoi diagrams are often considered in $\mathbb{R}^3$, or in higher, d-dimensional, spaces. The complexity of the diagrams grow as the number of dimensions grow. For example, the complexity



(a) The diagram of 8 points. The dashed polygon bounds the convex hull of the set of points.



(b) The diagram of 3 line segments. The distance of a point x to its farthest site (segment p) is shown dashed.

Figure 1.4. Two farthest site Voronoi diagrams.

of the nearest point Voronoi diagram is exponential in the number of dimensions, more precisely, the diagram in $\mathbb{R}^d$ has $\Theta(n^{\lceil d/2 \rceil})$ complexity in the worst case. Further, it can be constructed in time $O(n^{\lceil d/2 \rceil} + n \log n)$, using the convex hull of the set of points in $d+1$ dimensions; see Klee [1980] and Chazelle [1993].

Voronoi structures have also been studied when the underlying space is some non-Euclidean space, or it is restricted to some geometric object. An example is the *spherical Voronoi diagram*, where the space is a a sphere in $\mathbb{R}^3$ under the *geodesic distance*; see, e.g., Na et al. [2002]. Another diagram using the geodesic distance is the *geodesic Voronoi diagram*, where the underlying domain is a simple polygon in $\mathbb{R}^2$; see, e.g., Papadopoulou and Lee [1998] and Oh [2019].

In this dissertation, we deal with planar Voronoi diagrams where the input site are objects in $\mathbb{R}^2$ and the underlying space is the Euclidean plane.

Higher order Voronoi diagrams

Another generalization of Voronoi diagrams concerns, the neighbors with respect to which the subdivision of the underlying space is done. For example in the nearest Voronoi diagram, the subdivision is done with respect to the nearest neighbor. When the subdivision is done with respect to the farthest neighbor, we speak of the *farthest (site) Voronoi diagram*. Such a diagram is defined as follows.

Definition 1.2. The *farthest Voronoi diagram* of a set of sites in a given space, is the subdivision of this space into maximal regions, such that all points within one region have the same farthest site.

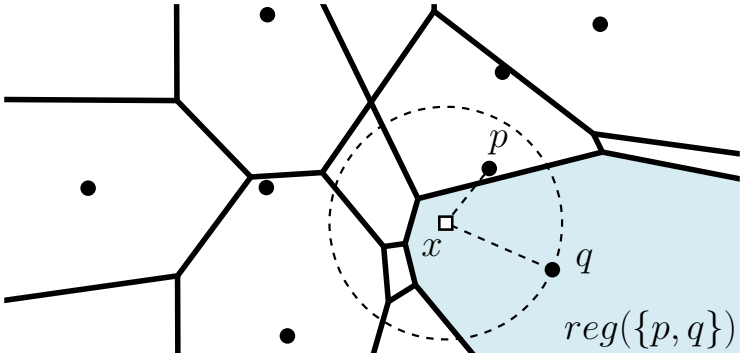


Figure 1.5. The order-2 Voronoi diagram of a set of 8 points in $\mathbb{R}^2$. The regions of a pairs of points (p,q) is shown shaded. The distance of a point x to its two nearest sites (points p and q) is shown dashed.

A farthest Voronoi diagram of a set of points is illustrated in Fig. 1.4a. The properties of the farthest point Voronoi diagram are well known; see Shamos and Hoey [1975]. It has $O(n)$ complexity, it can be computed in $O(n \log n)$ time, each Voronoi region is connected, and only points on the boundary of the convex hull have a non-empty region. It is worth observing in Fig. 1.4a, that the diagram has a tree structure, and that the circular ordering of the points along the boundary of the convex hull, is the reverse of the circular ordering of the unbounded faces.

Farthest Voronoi diagrams often have a simpler structure than their nearest site counterparts. For instance, the farthest point Voronoi diagram in the L_1 metric has $O(1)$ complexity. The farthest line segment Voronoi diagram, studied by Aurenhammer et al. [2006] and Papadopoulou and Dey [2013], has a tree structure of $O(n)$ complexity. A farthest Voronoi diagram of line segments is illustrated in Fig. 1.4b.

A more general concept is that of *higher-order*, or *order-k Voronoi diagrams*. In the order-k Voronoi diagram the space is subdivided into regions with respect to the k nearest neighbors. It can be defined as follows.

Definition 1.3. The *order-k Voronoi diagram* of a set of sites in a given space, is the subdivision of this space into maximal regions, such that all points within one region have the same k nearest sites.

An instance of an order-2 Voronoi diagram of 8 points is illustrated in Fig. 1.5. The order-k Voronoi diagram for $k = 1$ coincides with the nearest Voronoi diagram and for $k = n - 1$ with the farthest Voronoi diagram. Note that the parameter k is meaningful for $1 \leqslant k \leqslant n - 1$.

Higher order diagrams for certain values of k, have more complicated structure. Lee [1982] showed that the order-k Voronoi diagram of a set of n points has $O(k(n-k))$ complexity, and it can be constructed in $O(k^2 n \log n)$ time. For the order-k diagram of line segments Papadopoulou and Zavershynskyi [2016] showed that it has complexity $O(k(n-k))$, if $k \geqslant n/2$, and $O(k(n-k)+I)$ complexity, if $k < n/2$ (where I is the number of pairwise segment intersections).

Unifying frameworks for Voronoi diagrams

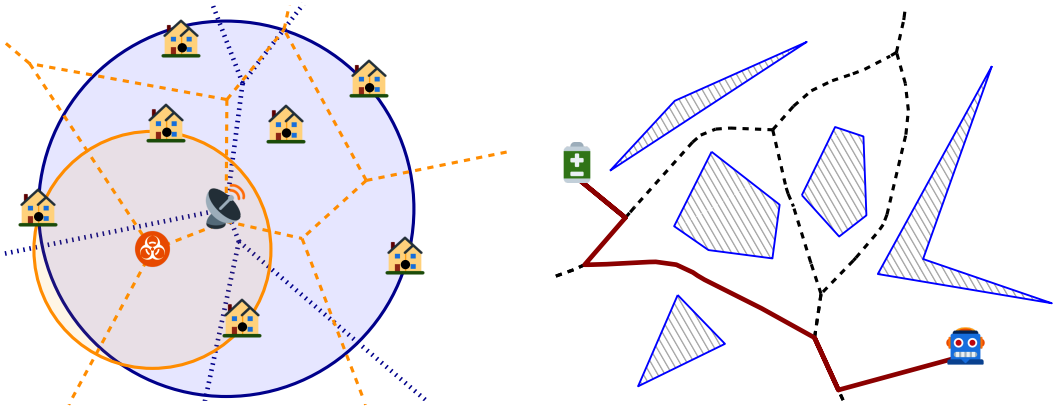
We already saw that Voronoi diagrams can be generalized in many ways. Since early times, there has been an interest in defining a unifying framework to cover various cases of these generalized diagrams. Two notable approaches are that of Edelsbrunner and Seidel [1986], for Voronoi diagrams in arbitrary dimension, and of Klein [1989], for planar Voronoi diagrams.

Edelsbrunner and Seidel [1986] described a generic scheme where a Voronoi diagram in $\mathbb{R}^d$ can be seen as the *lower envelope* of an *arrangement of hypersurfaces* in $\mathbb{R}^{d+1}$. More specifically each input site induces a distance function in one dimension higher, that is a hypersurface in $\mathbb{R}^{d+1}$. This yields an arrangement of hypersurfaces in $\mathbb{R}^{d+1}$, and the projection of the lower envelope of this arrangement down to $\mathbb{R}^d$ corresponds to the Voronoi diagram. Further, this approach yields not only the nearest site, but all higher order Voronoi diagrams, which are encoded in different *levels* of the arrangement. We describe in more detail this approach and look at some examples in Section 2.1.1.

Klein [1989] introduced the concept of *abstract Voronoi diagrams*. Instead of defining a Voronoi diagram via input sites and distance measures, a diagram is now defined via the underlying system of bisecting curves. There is one bisecting curve for each pair of input sites, and the only requirement to fall under the umbrella of abstract Voronoi diagrams, is that the bisecting curves satisfy a set of simple combinatorial properties, called *axioms*. This framework covers many of the generalized Voronoi diagrams described earlier. Abstract Voronoi diagrams have $O(n)$ complexity and can be constructed in $O(n \log n)$ time. We review the axioms of abstract Voronoi together with related results in Section 2.1.2.

Applications of Voronoi diagrams

Voronoi diagrams efficiently encode proximity information among different objects, and thus, they can be useful in many different application domains. Following, we indicatively describe a few applications. We refer to the book of Okabe et al. [2009] for a thorough list of applications.



(a) A set of 7 points. The smallest enclosing disk is centered on the (dotted) farthest Voronoi diagram. The largest empty disk is centered on the (dashed) nearest Voronoi diagram.

(b) The Voronoi diagram of a set of obstacles (polygons) shown dashed. The robot can safely reach its target by following the path along the Voronoi diagram, shown with thick red segments.

Figure 1.6. Two examples of applications of Voronoi diagrams.

Several applications are related to answering *nearest neighbor queries*. Suppose that while being located somewhere we need to send a mail, and we want to find out of n post-offices, the nearest one. The trivial approach (examining all the distances and keeping the minimum) would take $O(n)$ time. Instead, given then Voronoi diagram of the post-offices, we could simply locate the Voronoi region of the post-office in which we would lie, and this can be done in $O(\log n)$ time. Such type of queries show up in spatial databases and in clustering and classifications problems. For example, both the standard *k-means* clustering method, and the basic *k-nearest neighbor (k-nn)* classification method perform a large number of such queries. Note that Voronoi diagrams may not be favorable in higher dimensions, as their complexity has an exponential dependency in the dimension.

Other applications are related to *facility location*. Consider the following examples, and refer also to Fig. 1.6a. Suppose that given a set of houses we need to install a telecommunications antenna, in a way that it covers all the houses but also its range is minimized. The solution is given by the *smallest enclosing disk* which is centered on the farthest Voronoi diagram. Alternatively suppose we wanted to install a waste dump, in a way that it is as far as possible from the houses but also not *arbitrarily far*, so that it can still serve the houses. The solution is given by the *largest empty disk* which is realized on the nearest Voronoi diagram. Many variants of these objects, as *minimum-width annuli*, *color spanning objects*, or objects allowing *outliers* are realized using different Voronoi diagrams.

Some applications of Voronoi diagrams are related to spatial interpolation, and modeling spatial data or processes. Actually, many early studies were motivated by such applications. In 1644, Dirichlet used Voronoi diagrams to model the distribution of matter in the solar system, and Thiessen, in 1911, used Voronoi diagrams for polygonal spatial interpolation related to meteorology. Another standard example, is the modeling of spatial processes, using the *Voronoi growth model*. Conceptually, one can think of circles growing from the input point-sites, until these circles *conflict* with each other. Such patterns appear in natural sciences, like crystallography or ecology involving animals or plants.

Another famous application is related to *surface reconstruction*; see Amenta et al. [1998b]. Given a sample point set from a surface, the Voronoi diagram can be used in conjunction with its dual structure, the Delaunay triangulation, to reconstruct the surface. In Robotics, in *path planning*, the Voronoi diagram can be used to model the navigation of a robot in an environment with *obstacles*. Given the Voronoi diagram of the obstacles, the robot can safely navigate along the edges of the Voronoi diagram following a path of large distance from the obstacles. Refer to the illustration of Fig. 1.6b for an example.

Summing up, Voronoi diagrams are powerful tools that find applications in problems that distance information is important. Still, even the notion of distance can be interpreted broadly; we will see later how we can model, a seemingly unrelated *art-gallery problem* by defining an appropriate distance measure (and Voronoi diagram) tailored to the problem.

1.2 Voronoi diagrams in this dissertation

In this dissertation, we consider Voronoi diagrams where the underlying space is $\mathbb{R}^2$, and look into generalizations involving different input sites and distance functions. We focus on three different topics. In the next sections we define each of the three topics and our related research goals. In Section 1.2.1 we consider *color Voronoi diagrams*, in Section 1.2.1 the *rotational Voronoi diagram*, and in Section 1.2.1 a linear-time algorithmic scheme for Voronoi diagrams.

1.2.1 Color Voronoi diagrams

The first topic we consider is related to *color Voronoi diagrams*, where each site is a set of points, which we refer to as *cluster*. Conceptually each cluster is identified by a distinct color; hence the name. The distance between a point and a cluster P is realized by the nearest point in P. This is formally defined as follows.

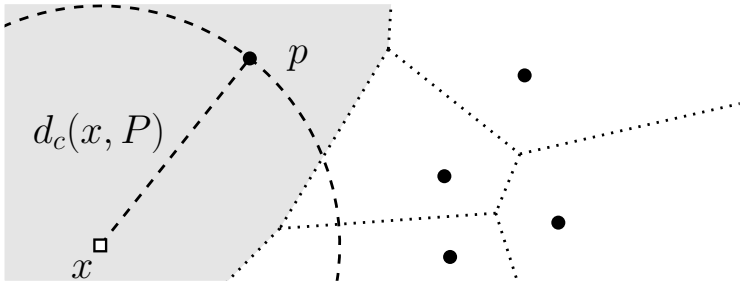


Figure 1.7. A cluster P (•) consisting of 5 points. The distance of a point x to P (realized by point p) is shown dashed. Dotted is the nearest point Voronoi diagram of P and shaded is the Voronoi region of p.

Definition 1.4. Given a cluster P and a point $x \in \mathbb{R}^2$ the *(minimum) distance* from x to P is

$$d_c(x,P) := \min_{p \in P} d(x,p).$$

Refer to Fig. 1.7 for an illustration of the distance function. Observe that when the distance $d_c(x,P)$ is realized by a point p, then x belongs into the Voronoi region of p in the nearest Voronoi diagram of P.

Given a set $\mathcal{P}$ of m clusters of points, with n total points, we define the nearest site Voronoi diagram using the minimum distance as follows.

Definition 1.5. The *nearest color Voronoi diagram* of a set of clusters $\mathcal{P}$, is the subdivision of $\mathbb{R}^2$ into *nearest color (Voronoi) regions*.
The *nearest color region of a cluster* $P_i \in \mathcal{P}$ is

$$n_c reg(P_i, \mathcal{P}) := \{x \in \mathbb{R}^2 \mid d_c(x,P_i) < d_c(x,P_j) \ \forall P_j \in \mathcal{P} \setminus \{P_i\}\}.$$

The *nearest color region of a point* $p \in P_i$ is

$$n_c reg(p, \mathcal{P}) := \{x \in n_c reg(P_i, \mathcal{P}) \mid d(x,p) < d(x,q) \ \forall p \in P_i \setminus \{q\}\}.$$

The nearest color Voronoi diagram is a simple "*min-min*" diagram, which can be easily derived from the nearest Voronoi diagram of the points of all clusters. Thus, the results of the nearest point Voronoi diagram directly apply, i.e., the diagram has $O(n)$ combinatorial complexity and $O(n \log n)$-time construction algorithms. An example of such a Voronoi diagram, is shown in Fig. 1.3a.

In this dissertation we are particularly interested in the farthest counterpart, the farthest color Voronoi diagram which we formally define as follows.

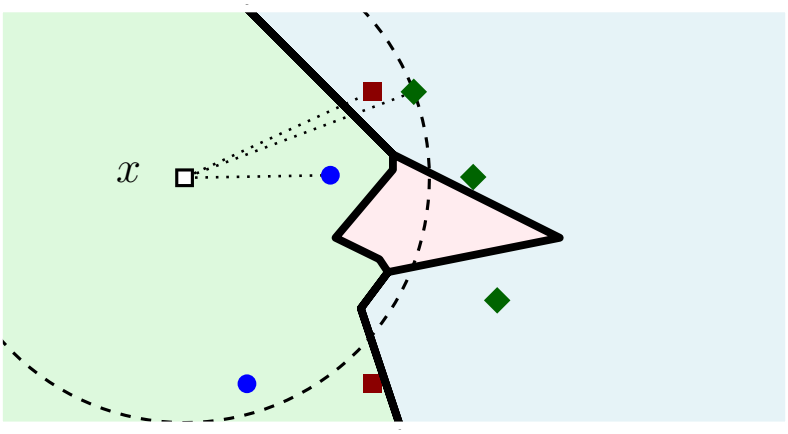


Figure 1.8. The farthest color Voronoi diagram of 3 clusters of points (■, ◆, ●). Points in a region are farther away from the cluster of the respective color. The distance of a point x to each cluster is shown dotted, and the farthest distance (◆) is represented by the dashed circle.

Definition 1.6. The *farthest color Voronoi diagram* of a set of clusters $\mathcal{P}$, is the subdivision of $\mathbb{R}^2$ into *farthest color (Voronoi) regions*.
The *farthest color region of a cluster* $P_i \in \mathcal{P}$ is

$$f_c reg(P_i, \mathcal{P}) := \{x \in \mathbb{R}^2 \,|\, d_c(x, P_i) > d_c(x, P_j) \;\forall\, P_j \in \mathcal{P} \setminus \{P_i\}\}.$$

The *farthest color region of a point* $p \in P_i$ is

$$f_c reg(p, \mathcal{P}) := \{x \in f_c reg(P_i, \mathcal{P}) \,|\, d(x, p) < d(x, q) \;\forall\, p \in P_i \setminus \{q\}\}.$$

The farthest color Voronoi diagram, is a "*max-min*" diagram; it can be thought as generalizing both the nearest and farthest Voronoi diagrams of points. Refer to Fig. 1.8 for an illustration of a farthest color Voronoi of 3 clusters. Observe point x: the distance to each of the 3 clusters is shown dotted. Point x belongs to the farthest color region of cluster (◆) which has the greatest minimum distance.

Regarding the complexity of the diagram, Huttenlocher et al. [1993] showed an $\Omega(mn)$ lower bound in the worst-case complexity, and a matching $O(mn)$ upper bound was given by Abellanas et al. [2001b]. The current best algorithms are an $O(mn \log n)$-time algorithm by Huttenlocher et al. [1993] and an $O(n^2)$-time algorithm using the approach of Edelsbrunner et al. [1989]. A detailed review of the combinatorial and algorithmic results is given in Section 2.2.1.

In this dissertation we further examine the farthest color Voronoi diagram. Our motivation comes from the diverse applications it finds. It is useful in facility location problems, as it can yield *minimum color spanning disks*. Related to *shape*

matching, it can measure the minimum Hausdorff distance, under translation, between two shapes (represented by clusters). It can also be used to construct approximate farthest Voronoi diagrams of arbitrary input sites. A detailed review of the applications of color Voronoi diagrams is given Section 2.2.3.

Research goals related to color Voronoi diagrams

As already mentioned, given a set of m clusters, of n total points, the worst-case complexity is settled to be $\Theta(mn)$. In practice, it seems that instances realizing a worst-case complexity do not often come up, and it is more likely that the diagram has $O(n)$ complexity. Our goal is to delve further into the structural properties of the diagram, and understand what are the conditions that contribute to an increased complexity of the diagram. Using such properties, we are interested in identifying necessary and sufficient conditions for the diagram to have $O(n)$ complexity. In this scope, we also examine the connection of color Voronoi diagrams with the framework of abstract Voronoi diagrams.

A particular class of input clusters are the ones which are pairwise *linearly separable*, i.e., that have pairwise disjoint convex hulls. This is perhaps the most natural class of input clusters to study (apart from arbitrary input clusters), as linearly separable input sites often come up. Our goal is to study the diagram of such clusters and to understand if linear separability is a condition for the diagram to have a nice structure, e.g., having $O(n)$ complexity.

On the algorithmic side, we already mentioned two schemes: An $O(mn \log n)$-time that is optimal if $m = \Theta(1)$. An $O(n^2)$-time that is optimal if $m = \Theta(n)$ and the diagram achieves the worst-case complexity, i.e., it has $\Theta(mn) = \Theta(n^2)$ complexity. Still, these algorithms do not have a satisfactory performance in cases when the diagram has $O(n)$ complexity. In conjunction with our study for conditions for $O(n)$ complexity, we want to design algorithms whose time complexity will depend on the specific properties of the input clusters, and which can outperform the existing algorithms (assuming an $O(n)$ combinatorial complexity).

1.2.2 Rotating rays Voronoi diagram

A second Voronoi diagram that we consider is the *rotating rays Voronoi diagram*. It is a nearest site Voronoi diagram, where the input is a set of rays, and the distance between a point x to a ray r is the *oriented angular distance*.

The oriented angular distance is given by the minimum angle α such that, after counterclockwise rotating r around its apex by α, ray r *sees* (or *touches*) x.

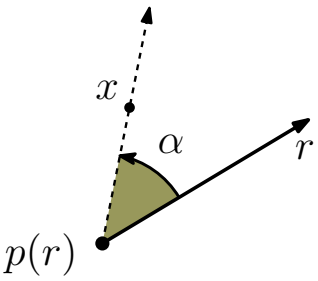


Figure 1.9. The angular distance (angle α) from a ray r to a point x.

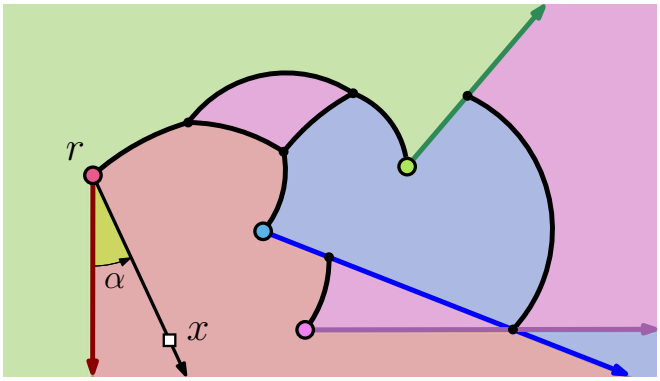


Figure 1.10. The rotating rays Voronoi diagram of 4 rays (↓, →, ↘, ↗). The distance (angle α) of a point $x \in \mathbb{R}^2$ to its nearest site (ray r) is shown.

Refer to Fig. 1.9 for an illustration of the distance definition. The distance can be formally defined as follows.

Definition 1.7. Given a ray r and a point $x \in \mathbb{R}^2$, the *oriented angular distance* from x to r is the minimum counterclockwise angle from r to a ray with apex $p(r)$ passing through x. We denote this distance by $d_\angle(x,r)$, and set $d_\angle(p(r),r) = 0$.

Given set $\mathcal{R}$ of n input rays, we can now define the nearest site Voronoi diagram of rays with respect to the oriented angular distance.

Definition 1.8. The *rotating rays Voronoi diagram* of a set of rays $\mathcal{R}$, is the subdivision of $\mathbb{R}^2$ into *nearest ray (Voronoi) regions*.
The *nearest ray region* of a ray $r \in \mathcal{R}$ is

$$r_\angle reg(r) := \{\, x \in \mathbb{R}^2 \mid \forall s \in \mathcal{R}\backslash\{r\} :\ d_\angle(x,r) < d_\angle(x,s) \,\}.$$

An example of a rotating rays Voronoi diagram of 4 rays is illustrated in Fig. 1.10. The Voronoi diagram draws its name by the following intuitive description: consider having a set of rays which start rotating counterclockwise with the same speed, and each point in $\mathbb{R}^2$ is assigned to the Voronoi region of the ray which sees the point first. This is analogous to the circle growing perspective for the nearest point Voronoi diagram, which we described earlier.

To the best of our knowledge, the rotating rays Voronoi diagram has not been considered before, and research on Voronoi diagrams with similar input sites or distance measure are very limited. This Voronoi diagram is of interest as it can be used to solve classes of *floodlight illumination problems*.

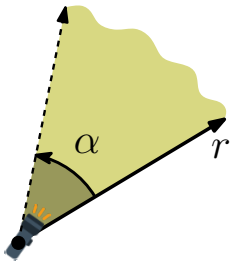


Figure 1.11. A floodlight (🔦) of aperture α aligned with a ray r.

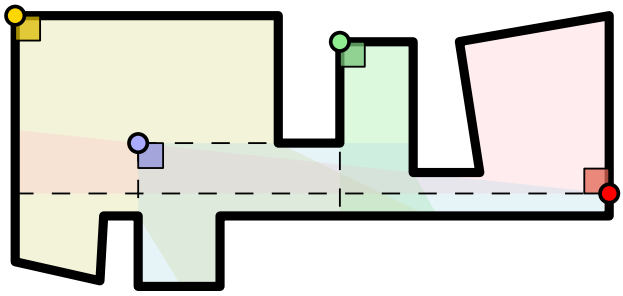


Figure 1.12. Illuminating an art gallery with 4 floodlights of aperture 90°. The area illuminated by each floodlight is shaded with the respective color.

Floodlight illumination problems are *art-gallery* type of problems where a given domain has to be guarded (covered) by a set of wedges, or using the literature terminology: a given domain has to be *illuminated* by *floodlights*. A floodlight of aperture α is called an *α-floodlight*, and given a ray r, an α-floodlight is said to be *aligned with the ray r*, if the right side of the wedge coincides with r. Refer to Fig. 1.11 for an illustration of an α-floodlight aligned with a ray r. Several variants of floodlight illumination problems require the floodlights to be of uniform angle; refer to Fig. 1.12 for an example of a polygonal domain illuminated by 4 floodlights of aperture $\pi/2$. We review some variants and results on floodlight illumination problems in Section 2.3.1.

A class of floodlight illumination problems is the *Brocard illumination problem* which owes its name to Henri Brocard (1845-1922) and an old geometric problem. The *Brocard illumination problem* can be defined as follows.

Definition 1.9. Given is a domain $\mathcal{D}$, and a set $\mathcal{R}$ of n rays with an α-floodlight aligned with each ray. What is the minimum angle α^* needed to illuminate $\mathcal{D}$ with the set of α^*-floodlights? This is called the *Brocard illumination problem* and α^* is called the *Brocard angle*.

Brocard illumination problems are defined on different domains such as the plane, polygons, and curves. Results for polygonal domains have been recently presented by Alegría-Galicia et al. [2017]. We review results on the original problem by Brocard and the Brocard illumination problem in Section 2.3.2.

Our motivation for studying the rotating rays Voronoi diagram, comes form its application to Brocard illuminations problems. Interestingly, we can reduce the Brocard illumination problem to the construction of a rotating rays Voronoi diagram, as the Brocard angle is realized on the graph structure of the diagram. We describe this application in detail in Section 2.3.3.

Research goals related to the rotating rays Voronoi diagram

Our first goal related to the rotating rays Voronoi diagram, is to get a solid understanding of the angular distance function and the resulting bisectors. This will then help us understand the structure of the diagram, as edges of the diagram are parts of bisectors. First, we want to consider $\mathbb{R}^2$ as the underlying space and to obtain bounds in the combinatorial complexity of the diagram together with construction algorithms. After having a construction algorithm for the diagram in $\mathbb{R}^2$, we can also solve the Brocard illumination problem within the same time, as the Brocard angle is realized at a point on the graph structure of the diagram.

Apart from the plane, we want to study the rotating rays Voronoi diagram in polygons and curves. Such domains are common in the literature of floodlight illumination problems, and we want to obtain combinatorial and algorithmic results in order to solve the respective Brocard illumination problems. For the Brocard illumination problem of polygonal domains, as already mentioned, there is an $O(n \log n)$-time algorithm for convex polygons and an $O(n^3 \log^2 n)$ for arbitrary simple polygons. As a starting point, for polygonal domains we aim to study the class of convex polygons and to design a faster, $o(n \log n)$-time, algorithm.

1.2.3 Linear-time algorithms for planar Voronoi diagrams

Another topic we consider is related to construction algorithms for planar Voronoi diagrams. As already mentioned, there is an $\Omega(n \log n)$ lower bound for the time needed by any algorithm to construct the nearest point Voronoi diagram.

However, for certain settings when additional information is known (as an *ordering* of the regions), some diagrams can be constructed faster. In 1989, Aggarwal, Guibas, Saxe, and Shor [1989] introduced a deterministic linear-time algorithm to solve the following problem:

(1) compute the nearest Voronoi diagram of points in convex position, given the ordering of the points along the convex hull.

The graph of such a Voronoi diagram has a tree structure and its regions are connected. The algorithm can be easily adapted to also construct other Voronoi diagrams of point-sites with a tree structure, and connected regions, such as:

(2) update a nearest Voronoi diagram after deletion of a point;
(3) compute a farthest Voronoi diagram, given the ordering of the points along the convex hull;
(4) compute an order-k Voronoi diagram, given its order-$(k-1)$ counterpart.

Although the algorithm of Aggarwal et al. [1989] is asymptotically optimal with respect to the time complexity, it is fairly complicated. Around the same time, Chew [1990] presented a very simple algorithm using a randomized incremental approach, which has expected $O(n)$ time complexity.

Since then, the framework of Aggarwal et al. [1989] has been used, and extended, in various ways to tackle various $O(n)$-time Voronoi constructions, including the *medial axis* of a simple polygon by Chin et al. [1995], the *Hamiltonian* abstract Voronoi diagram by Klein and Lingas [1994], and some *forest-like* abstract Voronoi diagrams by Bohler et al. [2014].

For generalized sites, other than points in the plane, or for abstract Voronoi diagrams, deterministic linear-time algorithms for the counterparts of problems (1)-(4) have not been known so far. This includes the diagrams of very simple geometric sites such as line segments and circles in the Euclidean plane. A major complication over points is that the underlying diagrams have disconnected Voronoi regions. Recently, Junginger and Papadopoulou [2018] presented a randomized $O(n)$-time technique for abstract Voronoi diagrams, using a relaxed Voronoi structure, called a *Voronoi-like diagram*. A preliminary algorithm for the farthest line segment Voronoi diagram was presented by Khramtcova and Papadopoulou [2017].

Research goals related to linear-time construction algorithms

With the recent advance of Junginger and Papadopoulou [2018] to generalize the randomized scheme of Chew [1990] to abstract Voronoi diagrams, it remains an open problem whether this can also be done for the deterministic scheme of Aggarwal et al. [1989]. It is of particular interest to examine if the relaxed Voronoi-like structures introduced to generalize the randomized scheme, can also be used to generalize the deterministic scheme.

The above does not seem to be a simple task, but we want to make a first step in this direction. The scheme of Aggarwal et al. [1989] is based on a combinatorial result on embedded binary trees, and this is necessary to get any $O(n)$-time algorithm based on their approach. By generalizing this combinatorial result to trees inspired by the Voronoi-like structures, we hope to make the scheme applicable to a larger class of planar Voronoi diagrams.

In a nutshell, the result on binary trees shows the existence of a set of leaves with some *desired property* and a $O(n)$-time algorithm to select them. Intuitively, the binary tree represents the graph structure of a Voronoi diagram, leaves represent input sites, and the desired property is pairwise disjoint Voronoi regions.

1.3 Dissertation contributions

We now summarize our contribution for the three problems on planar Voronoi diagram that we considered in this dissertation.

Farthest color Voronoi diagram

- We study the structural properties and the combinatorial complexity of the diagram for different configurations of input sites. More specifically:
  - We show an one to one correspondence between the unbounded faces of the diagram and the unbounded faces of the Hausdorff Voronoi diagram; this implies that the diagram has only $O(n)$ unbounded faces.
  - We identify the *straddle*[3], as a necessary condition for the diagram to have a *large* number of bounded faces. More specifically, we show that the diagram has $O(n+s(\mathcal{P}))$ bounded faces, where $s(\mathcal{P}) = O(mn)$ is the *straddling number*[3] of the set of clusters $\mathcal{P}$.
  - Using the above results, we refine the existing $O(mn)$ tight upper bound on the combinatorial complexity to $O(n+s(\mathcal{P}))$.
  - We study the connection of the diagram to abstract Voronoi diagrams, and give a necessary and sufficient for the input clusters to fall under this framework. We also show that *disk-separable clusters*[4] fall under the abstract Voronoi diagram framework.
  - We study the complexity of pairwise linearly separable clusters and give a $\Omega(n+m^2)$ lower bound in the worst-case complexity of the diagram. This is quite surprising, as it implies that linearly separable clusters can realize the $\Theta(n^2)$ worst-case complexity, if $m = \Theta(n)$.
- We look into construction algorithms, mainly by adapting existing algorithmic techniques to the particular setting of the farthest color Voronoi diagram. We obtain the following results.

[3]This can be very intuitively seen as a relaxation of a crossing: a pair of points (q_1, q_2) *straddles* a pair of points (p_1, p_2) if the line through (p_1, p_2) intersects ("*straddles*") the line segment $\overline{q_1 q_2}$. The *straddling number* of a set of clusters is simply the sum of all straddles of all pairs of points. Refer to Section 3.2.2 for the exact definition.

[4]A set of clusters is *disk-separable*, if for each cluster there exists a disk containing that cluster and no point from any other cluster.

  - An $O((n + s(\mathcal{P}))\log^3 n)$-time algorithm for arbitrary input clusters. Our algorithm is efficient and requires more time only at the presence of straddles, which can increase the complexity of the diagram. For *realistic* input clusters, the parameter $s(\mathcal{P})$ is expected to be small, i.e., $s(\mathcal{P}) = O(n)$, and our algorithm outperforms the existing ones.
  - An optimal $O(n \log n)$-time algorithm for clusters that satisfy the conditions of abstract Voronoi diagrams (such as disk-separable clusters).

Rotating rays Voronoi diagram

- We define and study the structural properties of the rotating rays Voronoi diagram, a Voronoi structure which had not been considered before.
- Given a set of rays we reduce the Brocard illumination problem to the construction of a rotating rays Voronoi diagram of the same set of rays, and show that the Brocard angle is realized at a vertex of the diagram. This paves the way for further research in both Brocard illumination problems and the rotating rays Voronoi diagram.
- We consider the diagram in the plane and obtain the following results.
  - We give an $\Omega(n^2)$ lower bound for the worst-case complexity of the diagram. Further, such a bound can be realized even if the rays are pairwise non-intersecting. We complement this with a $\Theta(n^2)$ worst-case lower bound for the complexity of a single Voronoi region.
  - Regarding an upper bound on the complexity of the diagram, we show that it is $O(n^{2+\epsilon})$, where $\epsilon > 0$ is an arbitrarily small positive constant. Using the same results we get, as a by-product, an algorithm with the same time complexity, i.e., $O(n^{2+\epsilon})$ time.
  - We solve the Brocard illumination in $\mathbb{R}^2$ in $O(n^{2+\epsilon})$ time, and we also show that the Brocard angle takes values between $2\pi/n$ and 2π.
- Motivated by the Brocard illumination of convex polygons, we consider the domain to be a convex polygon bounded by the set of input rays. We show the following results.
  - We study the diagram restricted to the interior of the polygon and show that it has $\Theta(n)$ complexity and a tree structure.
  - We give an algorithm to construct the diagram in optimal deterministic $\Theta(n)$ time. To do this we employ diverse techniques, including an

$\Theta(n)$-time construction algorithms for abstract Voronoi diagrams, for smaller parts of the problem.

- Using the above algorithm, we solve the Brocard illumination problem of a convex polygon in optimal $\Theta(n)$ time. We also show that the Brocard angle takes values between 0 and $\pi/2 - \pi/n$.
- We also present a significantly simpler $O(n \log n)$-time algorithm, which employs a "*collapsing*" technique.

- We restrict the target domain to be one or more curves. We show how the Voronoi diagram can be constructed with a generic approach of envelopes in 2-space, and discuss how this applies to different curved domains.

Towards linear-time construction algorithms

- We generalize the combinatorial result of Aggarwal et al. [1989] to *marked trees*, i.e., embedded binary trees whose leaves are partitioned into two sets, *marked* and *unmarked*. As already described, the combinatorial result shows the existence of a set of leaves with some *desired property* (pairwise disjoint Voronoi regions) and an $O(n)$-time algorithm to select them. Intuitively in marked trees, only the marked leaves are *important* for the selection and the unmarked leaves can be considered as *clutter*. Our generalization is composed of two parts.
  - We show that there exists a constant fraction $1/10$ of the marked leaves that have the desired property.
  - We give an algorithm to select in time $O(\frac{1}{1-p}n)$, a constant fraction p of the marked leaves with the desired property. This is done by introducing a trade-off parameter $p \in (0,1)$, to account for the unknown ratio and distribution among the marked and the unmarked leaves.

1.4 Dissertation outline and publications

The remainder of this dissertation is organized as follows:

- Chapter 2 gives the necessary background for better comprehending this dissertation and gives a review of the literature. We describe useful concepts and algorithmic techniques related to the Voronoi diagrams which we consider in the dissertation, and we also review the existing results, including applications.

- Chapter 3 deals with the farthest color Voronoi diagram. We present structural properties, refined combinatorial bounds, conditions for Voronoi diagrams of linear complexity, and algorithmic results.
- Chapter 4 is concerned with the rotating rays Voronoi diagram. We present combinatorial and algorithmic results regarding different domains of interest, and we apply them to floodlight illumination problems.
- Chapter 5 focuses on linear-time construction algorithms for classes of planar Voronoi diagrams. We generalize part of the deterministic linear-time framework of Aggarwal et al. [1989].
- Chapter 6 concludes the dissertation and discusses future directions related to the topics that this dissertation deals with.

Publications

The following is a list of the publications co-authored by me during my time as PhD candidate. In each publication the authors are ordered alphabetically.

- Chapter 3 is based on the following publication:
  Farthest Color Voronoi Diagrams: Complexity & Algorithms
  Mantas, Papadopoulou, Sacristán, and Silveira [2021a]
  → *Journal version* (to be submitted)
  → *Conference version*
  14th Latin American Theoretical Informatics Symposium (LATIN 2020)
  → *Preliminary short version*
  35th European Workshop on Computational Geometry (EuroCG 2019)
- Chapter 4 is based on the following publication:
  The Voronoi Diagram of Rotating Rays with Applications to Floodlight Illumination
  Alegría, Mantas, Papadopoulou, Savić, Schrezenmaier, Seara, and Suderland [2021]
  → *Journal version* (to be submitted)
  → *Conference version*
  29th Annual European Symposium on Algorithms (ESA 2021)
  → *Preliminary short versions*
  37th European Workshop on Computational Geometry (EuroCG 2021)
  XIX Spanish Meeting on Computational Geometry (EGC 2021)

- Chapter 5 is based on the following publication:
  On Selecting Leaves with Disjoint Neighborhoods in Embedded Trees
  Junginger, Mantas, and Papadopoulou [2021a]

  → *Journal version*
  Discrete Applied Mathematics (DAM), Elsevier, 2021
  (Special issue of CALDAM 2019 - Invited paper)
  → *Conference version*
  5th Annual International Conference on Algorithms and Discrete Applied Mathematics (CALDAM 2019)

The following publications have been co-authored by me, while I was a PhD candidate but are not part of this dissertation.

- **Certified Approximation Algorithms for the Fermat Point and n-Ellipses**
  Junginger, Mantas, Papadopoulou, Suderland, and Yap [2021b]

  → *Conference version*
  29th Annual European Symposium on Algorithms (ESA 2021)
  → *Preliminary short version*
  36th European Workshop on Computational Geometry (EuroCG 2020)

- **New Variants of Perfect Non-crossing Matchings**
  Mantas, Savić, and Schrezenmaier [2021b]

  → *Journal version* (to appear)
  Discrete Applied Mathematics (DAM), Elsevier
  (Special issue of CALDAM 2021 - Invited paper)
  Preprint: arXiv:2001.03252
  → *Conference version*
  7th Annual International Conference on Algorithms and Discrete Applied Mathematics (CALDAM 2021)
  → *Preliminary short versions*
  XIX Spanish Meeting on Computational Geometry (EGC 2021)
  36th European Workshop on Computational Geometry (EuroCG 2020)

Chapter 2

Background and literature review

In this chapter we give the necessary background to better comprehend this dissertation, and we also review the literature of the topics we deal with, together with applications. In Section 2.1, we describe some useful basic algorithmic paradigms and notions related to Voronoi diagrams. In Section 2.2, we review the existing results on Voronoi diagrams which have clusters as input sites. In Section 2.3, we discuss results on floodlight illumination problems and Voronoi diagrams of rays.

2.1 Basic techniques on planar Voronoi diagrams

In Section 2.1.1, we describe basic algorithmic paradigms for the construction of Voronoi diagrams, and in Section 2.1.2, we review the notion of abstract Voronoi diagrams. We start by giving some useful notation.

Given a set of sites $\mathcal{S}$, we denote the nearest Voronoi region of a site $s \in \mathcal{S}$ by

$$vreg(s,\mathcal{S}) := \{s \in \mathbb{R}^2 \,|\, d(x,s) < d(x,r)\ \forall s \in \mathcal{S} \setminus \{r\}\}.$$

The Voronoi diagram of a set of sites $\mathcal{S}$ is the union of all the Voronoi regions of the sites of $\mathcal{S}$. Each Voronoi diagram induces a planar subdivision, hence there exists a corresponding graph structure. We denote the graph structure of a Voronoi diagram by

$$\mathrm{VD}(\mathcal{S}) := \mathbb{R}^2 \setminus \bigcup_{s\in\mathcal{S}} vreg(s,\mathcal{S}).$$

To describe the features of such a plane graph we will be speaking of *Voronoi edges* and *Voronoi vertices*. If a Voronoi region has more than one connected

components, we will refer to each connected component as a *face*. For convenience, when it is clear from the context we may interchangeably refer to both the planar subdivision and its graph structure as simply the Voronoi diagram.

2.1.1 Algorithmic techniques

Following, we briefly describe some common algorithmic techniques for the construction of Voronoi diagrams. We describe the algorithms in terms of the nearest point Voronoi diagram. The schemes which we describe is *divide & conquer*, *lifting to 3 dimensions*, and *(randomized) incremental construction*.

Before proceeding, note that there is an $\Omega(n \log n)$ time lower bound on the time complexity that any algorithm requires, and this holds true even if the points are already sorted according to their x-coordinate; see Shamos [1978] and Djidjev and Lingas [1991].

Divide & conquer. The algorithm starts by recursively splitting a set of sites $\mathcal{S}$ into two subsets $\mathcal{S}_A$ and $\mathcal{S}_B$, until each set has a constant number of sites. Then, the two subsets are constructed recursively to obtain $\mathrm{VD}(\mathcal{S}_A)$ and $\mathrm{VD}(\mathcal{S}_B)$. In a third phase, the two diagrams are *merged* in order to obtain $\mathrm{VD}(\mathcal{S}_A \cup \mathcal{S}_B) = \mathrm{VD}(\mathcal{S})$.

To merge the two diagrams it is necessary to construct the *merge curve*, that is, the set of Voronoi edges in $\mathrm{VD}(\mathcal{S}_A \cup \mathcal{S}_B)$ which are equidistant to sites $p \in \mathcal{S}_A$ and $q \in \mathcal{S}_B$. Intuitively, if we think of the merging phase, as "*gluing together*" the Voronoi diagrams, the merge curve is the part where this "*gluing*" happens. To construct the merge curve a *starting point/edge* has to be identified from which we can start *tracing* the merge curve, i.e., constructing it edge by edge until it is completely constructed. Refer to Fig. 2.1 for an illustration of a merging phase of two nearest point Voronoi diagrams.

When the sites are points, the splitting phase can be done according to the x-coordinate, that is, $\mathcal{S}_A$ contains the leftmost half points and $\mathcal{S}_B$ the rightmost half points. This split guarantees a *nice* merge curve consisting of a single unbounded chain; see the curve in Fig. 2.1. As a result a starting point can be easily found and the overall merging phase takes $O(n)$ time, yielding an $O(n \log n)$ overall time complexity. This algorithm was first described by Shamos and Hoey [1975].

Unfortunately, such a nice merge curve is not always possible to obtain. In several diagrams, the merge curve consists of many components which can also be bounded. In these cases, identifying starting points is not easy, as it may require performing multiple *point location* queries. Using standard techniques, a single point location query in a Voronoi diagram can be performed in $O(\log n)$ time; see Kirkpatrick [1983]. Hence, merging efficiently two Voronoi diagrams

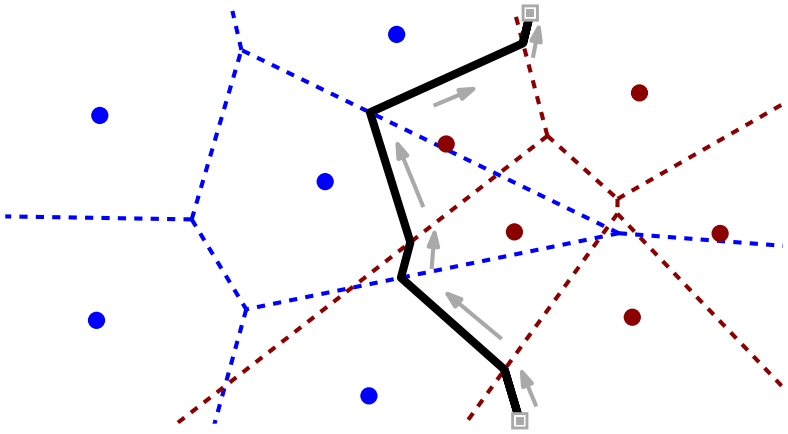


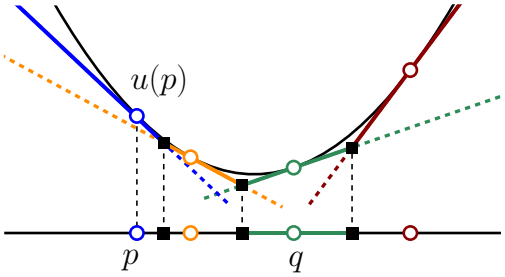
Figure 2.1. A merging phase of a nearest point Voronoi diagram. The two diagrams VD($\mathcal{S}_A$) and VD($\mathcal{S}_B$) are shown dashed. The black thick curve is the merge curve of VD($\mathcal{S}_A \cup \mathcal{S}_B$) . The arrows (⟶) schematize tracing, where (■) indicates the starting and ending points.

is a non-trivial task for any divide & conquer algorithm. We will see an example of a divide and conquer algorithm dealing with such cases in Section 3.5.

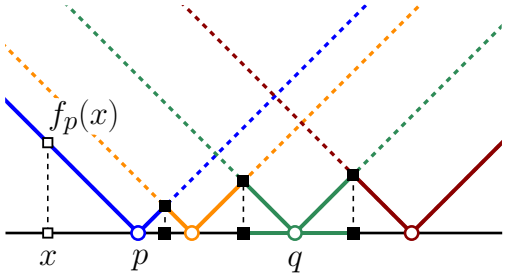
Incremental construction. Let $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ be the set of input point-sites, and let $\mathcal{S}_i := \{s_1, s_2, \ldots, s_i\}$. The algorithm starts by constructing the Voronoi diagram of a constant number of points, and then it incrementally *inserts* the points of $\mathcal{S}$, one by one, until the final diagram VD($\mathcal{S}$) is obtained. At step i of the algorithm, point s_i is inserted into the existing diagram VD($\mathcal{S}_{i-1}$), to obtain VD($\mathcal{S}_i$). This method was first described by Green and Sibson [1978].

The insertion operation of a point s_i can be seen a special case of a merging operation described earlier, where the merge curve is the boundary of the region $vreg(s_i, \mathcal{S}_i)$. To construct $vreg(s_i, \mathcal{S}_i)$, first the point s_i has to be located in the diagram VD($\mathcal{S}_{i-1}$), as this will reveal a starting point to start tracing the boundary of the region. Then, tracing can be done in time linear in the number of the neighbors of s_i in VD($\mathcal{S}_i$). It is not hard to see that s_i might be incident to $O(i)$ sites, hence the insertion of site s_i can take time $O(i)$, and this can lead to a total $\sum_{i=1\ldots n} O(i) = O(n^2)$ time complexity.

The above worst-case scenario, although possible, is not expected to happen often. A common approach is to resort to *randomization*, by taking a random permutation of the input points in the beginning. Refer to Clarkson and Shor [1989] and Guibas et al. [1992], for randomized schemes yielding algorithms with expected $O(n \log n)$-time complexity.



(a) A site p is lifted to the parabola (point $u(p)$), and induces the hyperplane tangent to the parabola at $u(p)$ (line, shown dashed). The upper envelope projected to $\mathbb{R}^1$ yields the Voronoi diagram



(b) A site p induces a distance function f_p, where a point $x \in \mathbb{R}^1$ is lifted to the point $f_p(x) = (x, d(x,p))$ (wedge, shown dashed). The lower envelope projected to $\mathbb{R}^1$ yields the Voronoi diagram

Figure 2.2. A nearest Voronoi diagram of a set of 4 points (○, ○, ○, ○) in $\mathbb{R}^1$ (horizontal line) constructed as the envelope of functions in $\mathbb{R}^2$, based on two different approaches. The envelopes are shown with solid colored segments, and the Voronoi vertices with (■). The Voronoi region of a site q (○) is highlighted.

Lift-up to 3 dimensions. Planar Voronoi diagram are closely related to arrangements of surfaces in the 3-dimensional space. More generally, a Voronoi diagram in $\mathbb{R}^d$ is related to an arrangement of hypersurfaces in $\mathbb{R}^{d+1}$ dimensions. We describe two different approaches, both of which are based on the framework of Edelsbrunner and Seidel [1986]. The description is given for the diagram in $\mathbb{R}^2$, and an accompanying illustration is given in Fig. 2.2 for the diagram in $\mathbb{R}^1$.

The first approach is to *lift up* each point-site $p \in \mathcal{S}$ to the unit paraboloid in $\mathbb{R}^3$, i.e., $p = (p_1, p_2) \mapsto u(p) = (p_1, p_2, p_1^2 + p_2^2)$. Then for each lifted point $u(p)$ take the plane that is tangent to the paraboloid at point $u(p)$. As a result, each site p yields a surface (plane) F_p. The collection $\mathcal{F}$ of these n surfaces is an *arrangement of surfaces* in $\mathbb{R}^3$. Now if we take the *upper envelope* (pointwise maximum) of $\mathcal{F}$ and project it down to $\mathbb{R}^2$, this yields the nearest Voronoi diagram of $\mathcal{S}$. Refer to Fig. 2.2a for the illustration of an 1-dimensional instance.

The second approach is to define for each point-site $p \in \mathcal{S}$ a 3-dimensional distance function f_p which takes each point $x \in \mathbb{R}^2$ and *lifts it up* to a point $f_i(x)$ with height equal to the distance of x to p, i.e., $x \mapsto f_p(x) = (x, d(x,p))$ As a result, each site p yields a surface (cone) F_p, and the collection $\mathcal{F}$ of these n surfaces induces an arrangement of surfaces in $\mathbb{R}^3$. The *lower envelope* (pointwise minimum) of $\mathcal{F}$ projected down to $\mathbb{R}^2$, yields the nearest Voronoi diagram of $\mathcal{S}$. Refer to Fig. 2.2b for the illustration of an 1-dimensional instance.

The aforementioned method is a very general and powerful approach. Apart from being applicable to higher dimensional Voronoi diagrams, it can also be used to derive all higher order Voronoi diagrams. Each of the $n-1$ *levels* of the arrangement of surfaces corresponds to a Voronoi diagram of different order.

Overall, to construct any planar Voronoi diagram it suffices to obtain the lower envelope of an arrangement of appropriately defined 3-dimensional surfaces. Sharir [1994] showed that the lower envelope of surfaces which are *well-behaved* has $O(n^{2+\epsilon})$ combinatorial complexity and it can also be constructed in $O(n^{2+\epsilon})$ time. We will see the requirements for surfaces to be well-behaved together with some examples of Voronoi diagrams modeled via this approach in Sections 4.2 and 4.4.

2.1.2 Abstract Voronoi diagrams

We now review the framework of *abstract Voronoi diagrams*, introduced by Klein [1989], which unifies various concrete instances of planar Voronoi diagrams.

In this framework, instead of sites and distances, Voronoi diagrams are defined on the underlying system of bisectors. Given two sites s and r, their bisector $b(s,r)$ divides the plane into two regions: *the dominance region of s over r*, denoted $\mathrm{dr}(s,r)$, and $\mathrm{dr}(r,s)$ defined analogously. The nearest Voronoi region of a site s, can be simply defined as the intersection of all related dominance regions:

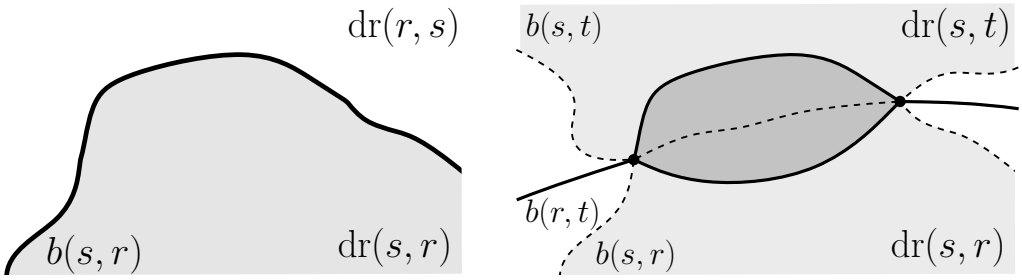
$$vreg(s,\mathcal{S}) := \bigcap_{r\in\mathcal{S}\setminus\{s\}} \mathrm{dr}(s,r).$$

Refer to Fig. 2.3 for an illustration of the above notions. To fall under the abstract Voronoi diagram framework, the system of bisectors has to satisfy a set of simple combinatorial properties, called *axioms*. The axioms require that for each subset of sites $\mathcal{S}' \subseteq \mathcal{S}$, the following hold.

(A1) For any two sites $r,s \in \mathcal{S}'$, their bisector $b(s,r)$ is an unbounded Jordan curve.
(A2) For any site $s \in \mathcal{S}'$, its Voronoi region $vreg(s,\mathcal{S}')$ is non-empty and connected.
(A3) The closure of the union of all the Voronoi regions in $\mathrm{VD}(\mathcal{S}')$ covers R^2.

We collectively refer to the three previous conditions as *AVD axioms*. Refer to the diagram of Fig. 2.3b and observe how the AVD axioms are satisfied. Note that in the original definition of Klein [1989] there was also a fourth axiom, but was later proved by Klein et al. [2009] that it can be subsumed without complications.

Voronoi diagrams satisfying the AVD axioms, have $O(n)$ complexity, and they can be constructed in deterministic $O(n\log n)$ time with a divide & conquer algorithm; see Klein [1989]. Alternatively, Klein et al. [1993] gave a randomized incremental construction with expected $O(n\log n)$ time complexity.



(a) A bisecting curve $b(s,r)$ and the resulting dominance regions $\mathrm{dr}(s,r)$ (shown shaded) and $\mathrm{dr}(r,s)$.

(b) The set of bisectors (dashed), the Voronoi diagram $\mathrm{VD}(\{s,r,t\})$ (solid), and the region $\mathrm{vreg}(s,\{s,r,t\})$ (shaded).

Figure 2.3. A set of 3 sites $\{s,r,t\}$ and their Voronoi diagram from the perspective of Abstract Voronoi diagrams.

Higher order abstract Voronoi diagram have also been considered in the literature. Mehlhorn et al. [2001] studied the farthest abstract Voronoi diagram. They proved that although a single region can have $\Theta(n)$ faces, the diagram has $O(n)$ overall complexity, a tree structure, and that it can be computed in expected $O(n \log n)$ time. Recently, Bohler, Cheilaris, Klein, Liu, Papadopoulou, and Zavershynskyi [2015] studied the order-k abstract Voronoi diagram showing that it has complexity $O(k(n-k))$. Algorithms to construct order-k diagrams have been given by Bohler et al. [2016, 2019].

Classes of abstract Voronoi diagrams have also been studied in conjunction with the aforementioned linear-time construction schemes. The scheme of Aggarwal et al. [1989] was applied by Klein and Lingas [1994] to *Hamiltonian abstract Voronoi diagrams*[1], and it was applied by Bohler et al. [2014] to *forest-like abstract Voronoi diagrams*[2]. Junginger and Papadopoulou [2018] applied the scheme of Chew [1990], to the fundamental problem of updating an abstract Voronoi diagram after deletion of a site (without additional assumptions).

Other recent extensions of the framework include bisecting curves which are closed (relaxed axiom A1) and sites having disconnected Voronoi regions (relaxed axiom A2); see Bohler et al. [2014; 2017].

[1] In *Hamiltonian abstract Voronoi diagram* there exists a *Hamiltonian curve* which visits each Voronoi region exactly once, in every subset of sites. See Section 4.3.3 for more details.

[2] *Forest-like abstract Voronoi* diagrams extend over the Hamiltonian ones, and allow the Hamiltonian curve to visit each Voronoi region more than once (but still remain connected).

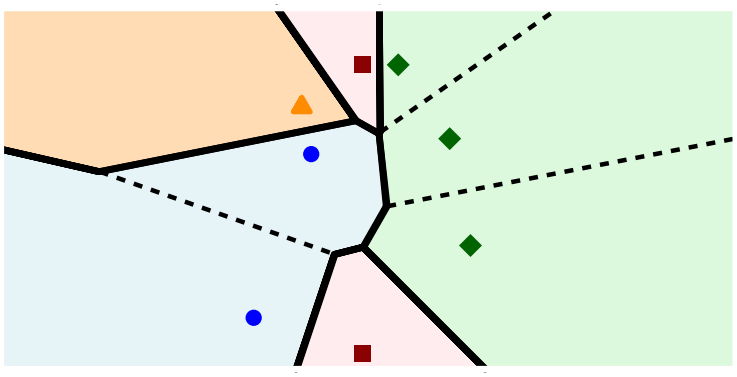


Figure 2.4. The nearest color Voronoi diagram of a set $\mathcal{P}$ of $m = 4$ clusters (■, ◆, ●, ▲) of $n = 8$ total points. The dashed edges indicate the finer subdivision of each Voronoi region.

2.2 Cluster Voronoi diagrams

In this section, we review results on Voronoi diagrams where the input sites are clusters of objects. In Section 2.2.1, we consider color Voronoi diagrams of points clusters, which is a topic this dissertation deals with, and review related algorithmic and combinatorial results. In Section 2.2.2, we review other cluster Voronoi diagrams which have been considered in the literature. In Section 2.2.3, we discuss different applications of color Voronoi diagrams.

2.2.1 Color Voronoi diagrams

Let the input be a set $\mathcal{P} = \{P_1, ..., P_m\}$ of m clusters of points, where no two clusters share a common point and $m > 1$. Let the set of all points be $\mathcal{P}^* := \bigcup_{P_i \in \mathcal{P}} P_i$, with $|\mathcal{P}^*| = n$. We consider the minimum distance as described in Definition 1.4.

Nearest Color Voronoi Diagram. The nearest color Voronoi diagram, as described in Definition 1.5, is the nearest site Voronoi diagram of point clusters, under the minimum distance. An instance of a nearest color Voronoi diagram of 4 clusters is shown in Fig. 2.4. Note that each nearest color region of a cluster of P can be *augmented* by an *internal subdivision* coming from the Voronoi diagram $\mathrm{VD}(P)$; see the dashed edges in Fig. 2.4.

The nearest color Voronoi diagram can be directly derived from the nearest Voronoi diagram of the points of all clusters, as the graph structure of the augmented nearest color Voronoi diagram coincides with $\mathrm{VD}(\mathcal{P}^*)$, the Voronoi

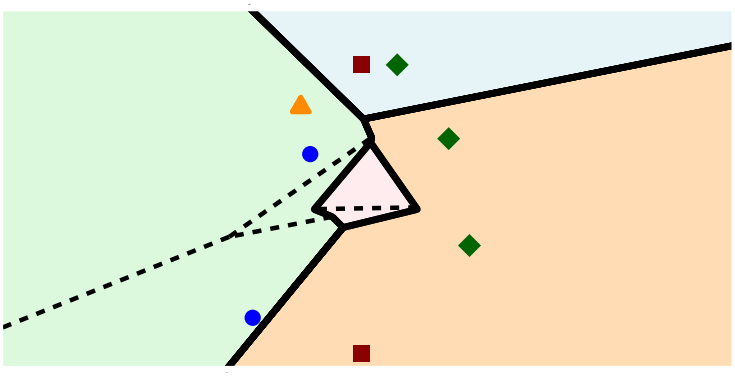


Figure 2.5. The farthest color Voronoi diagram of a set of 4 clusters of 8 total points. (It is the same set of clusters as the one shown in Fig. 2.4).

diagram of all points. As a consequence, the diagram in $\mathbb{R}^2$ has $O(n)$ complexity and it can be constructed in $O(n \log n)$ time. The diagram in $\mathbb{R}^d$, $d \geqslant 3$, has $\Theta(n^{\lceil d/2 \rceil})$ worst-case complexity and it can be constructed in $O(n^{\lceil d/2 \rceil})$ time.

An *output-sensitive*[3] algorithm to compute the nearest color Voronoi diagram in $\mathbb{R}^2$ was given by Bremner et al. [2005], with time $O(n \log k)$, where $k \leqslant n$ is the number of points that contribute to the complexity of the diagram. For $\mathbb{R}^d$, $d \geqslant 3$, an output-sensitive algorithm can be derived from the results of Eppstein [2022]. Note that both of these algorithms do not explicitly study color Voronoi diagrams, but *classification problems*; we give more details regarding such problems in Section 2.2.3.

Farthest Color Voronoi Diagram. The farthest color Voronoi diagram, as described in Definition 1.6, is the farthest site Voronoi diagram of point clusters, under the minimum distance. An example of a farthest color Voronoi diagram of 4 clusters is shown in Fig. 2.5. Similar to its nearest counterpart, each farthest color region of a cluster P can be augmented by the diagram $\mathrm{VD}(P)$; see the dashed edges in Fig. 2.5.

The farthest color Voronoi diagram was first studied by Huttenlocher et al. [1993]. They showed that the combinatorial complexity of the diagram is upper bounded by $O(mn\alpha(mn))$, and they gave an $\Omega(mn)$ lower bound in the worst case complexity. The worst case complexity was later settled to $\Theta(mn)$ by Abellanas et al. [2001b] who gave an $O(mn)$ upper bound.

[3] An algorithm is called *output-sensitive*, if its time complexity, depends on the size of the output, instead of, or in addition to, the size of the input.

Algorithms for the farthest color Voronoi diagram. We briefly describe the current two best algorithms, of Huttenlocher et al. [1993] and Edelsbrunner et al. [1989], which have $O(mn \log n)$ and $O(n^2)$ time respectively. Both are based on a lifting to 3-space approach, as described in Section 2.1.1. We only describe the lifting transformations and not the details regarding the envelopes computation.

The algorithm of Huttenlocher et al. [1993] can be described as follows. For each cluster $P_i \in \mathcal{P}$, we take every point $x \in \mathbb{R}^2$ and lift it to 3-space, with height equal to the nearest point in P_i, i.e., $x \mapsto x_i = (x, d_c(x, P_i))$. This yields a surface, called the *Voronoi surface* of cluster P_i. Then, we take the upper envelope of all m Voronoi surfaces, and project it down to $\mathbb{R}^2$. This induces the farthest color Voronoi diagram of $\mathcal{P}$.

The algorithm of Edelsbrunner et al. [1989] can be described as follows. For each cluster $P_i \in \mathcal{P}$, we take every point $p \in P_i$ and lift it in 3-space to the paraboloid, at point p_{par}. Then, we take the unique plane that touches paraboloid at a point p_{par}. Following, we consider the upper envelope of all $|P_i|$ planes, which yields a surface corresponding to cluster P_i. If we consider the lower envelope of all m surfaces and project it down to 2-space, we obtain the farthest color Voronoi diagram of $\mathcal{P}$.

Note that the algorithm of Edelsbrunner et al. [1989] was originally designed for the Hausdorff Voronoi diagram, a "*minmax*" diagram, which we define shortly after. Its was observed that it can be adapted for the farthest color Voronoi diagram by simply *reversing the procedure* (compute the lower envelope of all upper envelopes, instead of the upper envelope of all lower envelopes) by Claverol et al. [2018], who studied point clusters of cardinality 2.

Other results on the farthest color Voronoi diagram of points. Recently some special configurations of input clusters were examined that admit a diagram of $O(n)$ combinatorial complexity. Bae [2012] considered "*well-clustered sites*", essentially clusters satisfying some nice properties and yielding a $O(n)$-size diagram. Iacono et al. [2017] considered clusters being the vertices of axis-aligned rectangles (clusters of cardinality 4), and gave an $O(n \log^2 n)$-time algorithm. Claverol et al. [2018] studied clusters being the endpoints of segments (clusters of cardinality 2), and gave an $O(n \log n)$-time algorithm for parallel segments.

Some other settings were also considered by Huttenlocher et al. [1993] in their paper. They studied the diagram in $\mathbb{R}^3$ and gave an $O(mn^2\alpha(mn))$ complexity bound and an $O(n^2 m^{1+\epsilon})$-time algorithm, for any $\epsilon > 0$. Further, they considered the diagram under the L_∞ metric in $\mathbb{R}^2$, and they gave an $O(mn)$ upper bound on the complexity and a matching $\Omega(mn)$ worst case lower bound.

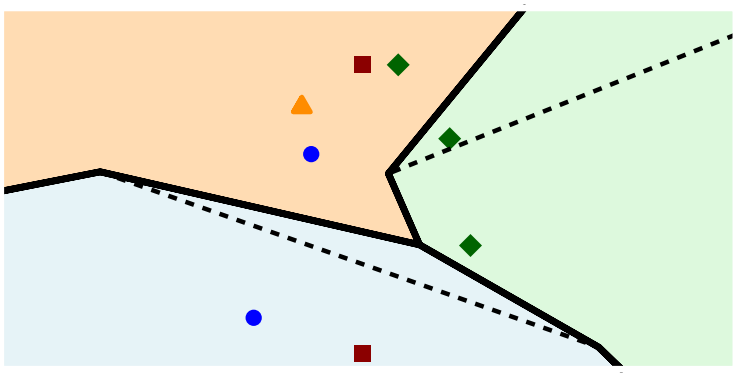


Figure 2.6. The Hausdorff Voronoi diagram of a set of 4 clusters of 8 total points. (It is the same set of clusters as the one shown in Figs. 2.4 and 2.5).

2.2.2 Other cluster Voronoi diagrams

We now review some other Voronoi diagrams where the input sites are clusters (sets) of objects and have been considered in the literature.

Hausdorff Voronoi diagram. A cluster Voronoi diagram that has been widely studied is the *Hausdorff Voronoi diagram*. It is a "*max-min*" type of diagram, and essentialy the "*dual*" of the farthest color Voronoi diagram (which is a "min-max" diagram). Given a set of of m clusters of n total points, the Hausdorff Voronoi diagram is a nearest site Voronoi diagram, where the distance between a point $x \in \mathbb{R}^2$ and a cluster P is the *maximum distance*, i.e.,

$$d_f(x,P) := \max_{p \in P} d(x,p).$$

An example of a Hausdorff Voronoi diagram is illustrated in Fig. 2.6.

The Hausdorff Voronoi diagram was first considered by Edelsbrunner et al. [1989], who gave an $O(n^2\alpha(n))$ upper bound, and an algorithm with the same time complexity. Papadopoulou and Lee [2004] improved the upper bound to $O(n^2)$, reducing also the time complexity of the aforementioned algorithm to $O(n^2)$. The diagram has been extensively studied since then. It has $O(n + \mathrm{cr}(\mathcal{P}))$ combinatorial complexity where $\mathrm{cr}(\mathcal{P})$ is the number of *crossings*[4] between clusters in $\mathcal{P}$. Further, if the clusters are pairwise non-crossing it falls under the abstract Voronoi diagram framework; see Papadopoulou and Lee [2004]

[4]The number of *crossings* between two clusters P and Q, is the number of pairs of segments on the boundary of the convex hull of $P \cup Q$, which have one endpoint in P and one in Q.

Many different algorithmic paradigms have been considered for its construction. The current fastest algorithms are a randomized incremental construction by Arseneva and Papadopoulou [2019] with expected time complexity $O((\mathrm{cr}(\mathcal{P})+n\log m)\log n)$, a divide and conquer algorithm by Iacono et al. [2017] with time $O((n+\mathrm{cr}(\mathcal{P}))\log^3 n)$, and the lifting to 3-space algorithm of Edelsbrunner et al. [1989] with time $O(n^2)$. Additional algorithms include a plane-sweep by Papadopoulou [2004] and a parallel divide and conquer by Dehne et al. [2006].

Farthest color Voronoi diagram of line segments. Another related diagram is the farthest color Voronoi diagram of line segments. Each input now is a cluster (set) of line segments, and the distance between a point to a cluster, is realized by the nearest segment belonging to the cluster.

The diagram was first studied in the paper of Huttenlocher et al. [1993]. They showed that the diagram has complexity $O(n^2 2^{\alpha(n)})$ and it can be computed in $O(n^2\alpha(n)\log n)$ time. When the L_∞ metric is considered it has complexity $O(n^2\alpha(n))$ and $O(n^2\log n)$ construction time. Later, Bae [2014] gave a tight worst case complexity bound of $\Theta(kn+h)$, where h is the number of crossings between the line segments (for any L_p metric). Using these bounds, Bae [2014] also improved the time complexity of the algorithm of Huttenlocher et al. [1993].

A special case occurs, when the segments of each input cluster form a simple polygon; this is known as the *farthest polygon Voronoi diagram*. The case of pairwise disjoint polygons was studied by Cheong et al. [2011], who showed that each Voronoi region is connected, an $O(n)$ complexity upper bound, and an $O(n\log^3 n)$-time construction algorithm. The farthest polygon Voronoi diagram was also studied by Zhu and Xu [2013].

Other Voronoi diagrams of clusters. For completeness, we mention some other cluster Voronoi diagrams which have been considered in the literature.

In the *2-site Voronoi diagram*, given are pairs of points, i.e., clusters of size 2, and various distance measures are considered between a point in the plane and a site, as for example, the sum, product or the difference of the two distances. Refer to Barequet et al. [2002, 2013] for some results related to such diagrams.

Another work is by Huang et al. [2021] regarding *influence-based Voronoi diagrams of clusters*. Here, the input is a set of (possibly overlapping) clusters of points and the distance function is a collective distance measure defined by all points in a cluster. These diagrams are studied in an approximate sense, and they build over an earlier work by Chen et al. [2017]

2.2.3 Applications of color Voronoi diagrams

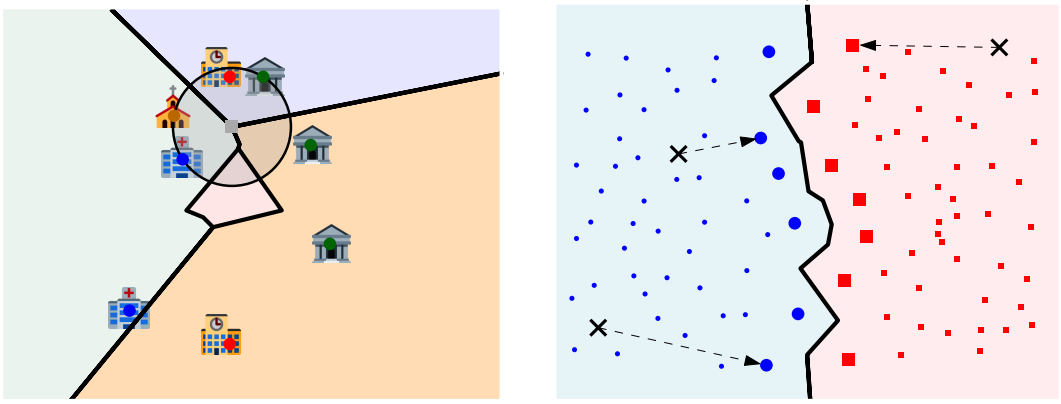
In this section, we discuss some applications of color Voronoi diagrams that have been considered in the literature.

Facility Location problems. Farthest-site related problems involving clusters appear in several different situations. Clusters may represent locations of facilities of the same type that can be accessed interchangeably, while the farthest distance allows to give worst-case scenario bounds on the distance to reach an object of each type. For instance, consider the following typical *facility location problem*: given locations of multiple types of facilities (e.g., hospitals, schools, etc.), each type represented by a cluster, we want to find a location such that the distance to all services is minimized. This can be found using the *minimum color spanning disk*, a disk containing a point of each color, which can be extracted efficiently from the farthest color Voronoi diagram, as it is realized at a vertex or an edge of the diagram; see Abellanas et al. [2001b]. Refer to Fig. 2.7a for an illustration. Such problems can also arise in spatial databases, see Guo et al. [2015] or Chen et al. [2020] for examples.

Many other *color spanning objects* have been considered in the literature, which can also be obtained using analogous Voronoi diagrams, like the *minimum color spanning annulus*, or the *minimum color spanning square*. For related results on *color spanning objects* see Abellanas et al. [2001a], Das et al. [2009], Fleischer and Xu [2010], Khanteimouri et al. [2013], or Acharyya et al. [2018].

Minimum Hausdorff distance. The farthest color diagram finds applications in *shape matching problems*, where the *similarity* between two shapes, represented by finite point sets, has to be measured. One such similarity measure is the *Hausdorff distance*[5]. Huttenlocher et al. [1993] showed how to find the translation that minimizes the Hausdorff distance between two sets A and B, by using the upper envelope of $|A| + |B|$ *Voronoi surfaces* (as discussed in Section 2.2.1), defined by the sets $A_i = \{a_i - b_j \,|\, b_j \in B\}$ and $B_j = \{a_i - b_j \,|\, a_i \in A\}$. The minimum Hausdorff distance under translation between A and B is realized at the global minimum of the upper envelope of these Voronoi surfaces. A similar technique in a dynamic setting, was used by Huttenlocher et al. [1992] to solve the same problem allowing additionally rotation. Refer to Veltkamp and Hagedoorn [2001] for more information on shape matching.

[5]The *Hausdorff distance* between two sets A and B is $H(A,B) = \max(h(A,B), h(B,A))$, where $h(A,B) = \max_{a_i \in A} \min_{b_j \in B} d(a_i, b_j)$ is the *directed Hausdorff distance* between A and B.



(a) The farthest color Voronoi diagram, of 4 clusters (🏥, ⛪, 🏛, 🏢). The minimum color spanning disk is centered (■) on the diagram.

(b) The nearest color Voronoi diagram of 2 clusters (●, ■). The relevant points are highlighted. 3 query points (×) are classified using the nearest neighbor rule.

Figure 2.7. Two examples of applications of color Voronoi diagrams.

Classification problems. In *classification problems* given is a set of point clusters (*training set*), and new (query) points need to be assigned (*classified*) to one of the existing clusters. A standard classification method is the nearest neighbor rule, where the query point is assigned to the cluster of its nearest neighbor; see Cover and Hart [1967]. It is not hard to see that such a classification rule induces a nearest point Voronoi diagram. The diagram can be constructed in $O(n \log n)$ time and queries can be answered in $O(\log n)$ time using standard methods. Often large datasets are involved, so it is natural to look for ways to reduce the size of the clusters without altering the quality of the nearest neighbor rule.

From the viewpoint of color Voronoi diagrams, it is not hard to see that the nearest color Voronoi diagram (without internal subdivision) is useful, as it provides a way to correctly reduce the size of the dataset: points which do not induce Voronoi edges are not *relevant* and can be safely removed without affecting the correctness of the nearest cluster query. Refer to the example of Fig. 2.7b; observe that out of the 100 points, only 13 are relevant, and suffice to correctly classify any given query. This is interesting as, if we are given a diagram, with k relevant points, we can answer queries in $O(\log k)$ time, instead of $O(\log n)$.

The above brings the question of efficient output-sensitive algorithms for the the nearest color Voronoi diagram. Bremner et al. [2005] constructs the diagram in the plane in $O(n \log k)$ time, calling it the *decision boundary*. Recently, Eppstein [2022] gave in higher dimensions output-sensitive algorithms for finding the relevant points, from which the diagram can be obtained as a by-product.

Other applications. Other farthest-site problems involving point-clusters appear when considering imprecision in geometric data, as points clusters are a natural way to represent the possible locations of an object, whose exact location is unknown; see e.g., Jørgensen et al. [2011]. In this setting, the farthest color Voronoi diagram encodes proximity information, allowing to efficiently solve problems involving pairs of points or larger clusters; see e.g., Ding and Xu [2011] or Arkin et al. [2015].

The farthest color Voronoi diagram has also been used by Bae et al. [2010] to solve variants of the *Steiner tree problem* and *sensor deployment problems* in wireless sensor networks by Lee et al. [2013]. Finally, another recent application was given by Claverol et al. [2018], who combined the diagram in a novel way with the "dual" Hausdorff Voronoi diagram, in order to find *stabbing circles* for line segments.

2.3 Voronoi diagrams of rays and illumination problems

In this section, we review the literature related to the Voronoi diagrams of rays and the respective illuminations problems. In Section 2.3.1, we review general floodlight illumination problems. In Section 2.3.2 we discuss results on Brocard illumination. In Section 2.3.3 we describe how the rotating rays Voronoi diagram can solve Brocard illumination problems and review related Voronoi structures.

2.3.1 Floodlight illumination

In *art gallery problems* given is a domain, which is often polygonal, and the goal is to appropriately place *guards* in order to guard (or cover) the domain. Most commonly, the guards have an unrestricted field of view, and can guard any point that is *visible*[6] by them. Most questions related to art gallery problems revolve around minimizing the number of guards used, and there are many interesting variants. The guards may be restricted to lie on the boundary of the domain (edges or vertices) or may have to be assigned to some predefined candidate positions. Refer to O'Rourke [1987, 2017] for a list of results.

Modeling the guards as having an unrestricted field of view is not always realistic. There are many applications where the "guards" have a limited field of view, as for instance, (surveillance) cameras, directional antennae, or even humans. To correctly model these problems, the field of view should be represented by a

[6]Given a guard p in a domain $\mathcal{D}$, a point $x \in \mathcal{D}$ is *visible* by p, if the line segment $\overline{px}$ does not intersect the boundary of $\mathcal{D}$.

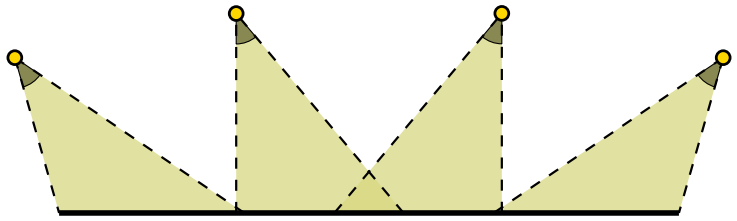


Figure 2.8. Stage (line segment) illumination by 4 floodlights of aperture 40°.

wedge, with aperture depending on the intended application. Such art gallery problems involving wedges as guards, are called *floodlight illumination problems*, and a guard with field of view of angle α is called an *α-floodlight*. Following we briefly review the literature of floodlight illumination problems.

Floodlight illumination results. Consider the following problem. Given is a set of n angles, representing the aperture of n floodlights, and a set of n points, representing the location of the apices of the floodlights. The goal is to assign the floodlights to the points and then orient them appropriately, such that a target domain is illuminated. Different domains have been examined in the literature.

When the target domain is $\mathbb{R}^2$, Bose et al. [1997] showed that if each floodlight has an angle of at most π, then a solution exists only if the sum of all angles exceeds 2π, and in such cases a solution can be found in $O(n \log n)$ time. They also considered the target domain to be a wedge. Hardness results for illuminating wedges were given by Steiger and Streinu [1998] and Cary et al. [2010].

Another variant is the *stage illumination problem*, where the domain is a line segment, called *stage*; see an example in Fig. 2.8. Ito et al. [1998] showed that the general problem is NP-complete. A significantly easier problem, is when the location of the apices is fixed, but they are free to rotate around, and the goal is to minimize the sum of all angles. $O(n \log n)$-time algorithms for this problem were given by Contreras et al. [1998b] and Dietel et al. [2008]. An *energy-aware* variant of stage illumination was studied by Eisenbrand et al. [2008].

Polygonal domains are also a common domain of interest for floodlight illumination problems. Given a convex polygon of n vertices, O'Rourke [1987] asked whether we can illuminate the polygon using a set of $k \leqslant n$ vertex floodlights whose apertures sums up to π. The authors gave a negative answer to that. Urrutia [2000] showed that any convex polygon can be illuminated by 3 vertex floodlights whose apertures sums up to π, and Ismailescu [2008] showed that the same can be achieved using 4 floodlights of aperture $\pi/2$. Another related question is whether a convex polygon can be illuminated using n vertex

π/n-floodlights. O'Rourke [1987] gave a negative answer (counterexample) for large n, and positive answers are currently known only for $n = 3, 4$.

For arbitrary simple polygons Estivill-Castro et al. [1995] and Spillner and Hecker [2002] study the problem of finding the minimum angle α such that a set of n (uniform angle) α-floodlights illuminates the input polygon. Tóth [2000, 2002, 2003] asks for the minimum number of (uniform) α-floodlights necessary to illuminate the polygon, and gives bounds for different values of α. Some other related results are given by Bagga et al. [1996], Abello et al. [1998], and Abdelkader et al. [2015].

2.3.2 Brocard illumination

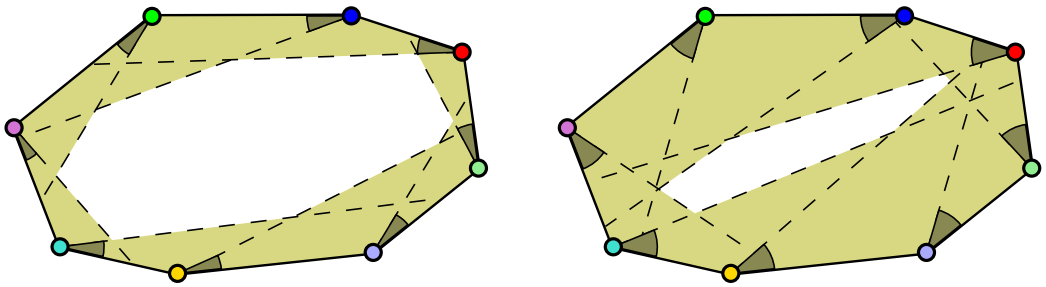
A particular type of floodlight illumination is the *Brocard illumination*, which we described in Definition 1.9. The Brocard illumination problem is a generalization of on an old geometric problem, called the *Brocard problem*. We describe this problem using the following extract from Guggenbuhl [1953], which nicely defines the original problem and highlights its importance.

> "*During the last half of the nineteenth century there was a great revival of interest in the field of geometry. A large part of this interest had its origin in a simple problem submitted to a contemporary mathematical periodical by a French army officer. The problem was to find a point O within a triangle ABC such that the angles OAB, OBC and OCA would be equal. The name of the army captain who submitted the problem was Brocard; Pierre René Jean-Baptiste Henri Brocard.*"

The aforementioned point O is called the *Brocard point*, and the angle which realizes it is called the *Brocard angle*. The problem of finding the Brocard point, after being solved for triangles, was later generalized to convex polygons with more vertices. For $n > 3$, a Brocard point does not always exist; a polygon which admits such a point is called a *Brocard polygon*. Despite the long history, except triangles, only *harmonic polygons*[7] are known to be Brocard; see Casey [1888]. An overview of the early history of the problem is given by Bernhart [1959].

Brocard illumination problems. An alternative way of viewing the Brocard problem is in terms of floodlight illumination. Suppose an α-floodlight is aligned with each edge of the polygon, and assume that α is initially set to 0, and it starts growing in the counterclockwise direction (facing the interior), until the entire

[7] *Harmonic polygons* are either regular polygons or polygons obtainable from regular polygons after cyclic inversion.



(a) The floodlight angle ($\measuredangle$) is $\alpha = 20°$. (b) The floodlight angle ($\measuredangle$) is $\alpha = 35°$.

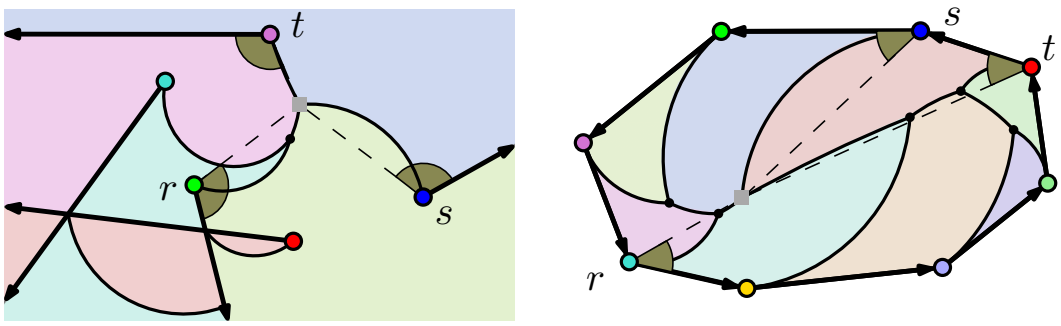
Figure 2.9. Brocard illumination of a convex polygon with 8 vertices.

interior of the polygon is illuminated; see an illustration of the setting in Fig. 2.9. If the last point of the polygon to be illuminated, is illuminated simultaneously by all the floodlights, then this point is the *Brocard point*, and the angle realizing it is the *Brocard angle*.

As already mentioned though, very few classes of polygons admit a Brocard point. Hence, motivated by the floodlight illumination interpretation, it is natural to generalize the problem and ask for the last point to be illuminated (not necessarily by all rays), and the (minimum) angle realizing this, which is the Brocard angle. Further, it is reasonable to generalize the problem of finding the Brocard angle to domains other than polygons, such as the entire plane and curves. A formal definition of the Brocard illumination problem was given in Definition 1.9.

Known results for the Brocard illumination problem. Given a convex polygon with n vertices, we can trivially detect if it is a Brocard polygon in $O(n)$ time (by checking the angles of three pairs of consecutive vertices), and then compute the Brocard angle in such cases in $O(1)$ time. An algorithm for the computation of the Brocard angle of arbitrary simple polygons was recently presented by Alegría-Galicia et al. [2017], who first studied this problem. The authors gave an $O(n^3 \log^2 n)$ time algorithm, and complemented this result with an $O(n \log n)$-time algorithm for convex polygons[8]. There are no results for the Brocard illumination problem for domains other than polygonal ones.

[8]The $O(n)$-time complexity for convex polygons claimed in Alegría-Galicia et al. [2017] is not correct. The correct time complexity of the algorithm is $O(n \log n)$.



(a) The domain is $\mathbb{R}^2$.

(b) The domain is a convex polygon.

Figure 2.10. Two examples of the application of the rotating rays Voronoi diagram to Brocard illumination problems. The Brocard angle, shown highlighted (◿), is realized at a Voronoi vertex (■) by the 3 rays r,s and t.

2.3.3 Voronoi diagrams of rays and their applications

We now describe how the rotating rays Voronoi diagram can be useful for the Brocard illumination problem. More specifically, given a set of rays $\mathcal{R}$ each aligned with a floodlight and a target domain $\mathcal{D}$ to be illuminated, the Brocard angle is realized on the graph structure of the rotating rays Voronoi diagram of $\mathcal{R}$ restricted to $\mathcal{D}$.

To see that, consider the rotating rays perspective described earlier. Assume that the floodlights are initially aligned with the rays, and then they start rotating counterclockwise, i.e., they start with $\alpha = 0$, and α keeps growing. As they rotate, points in the domain $\mathcal{D}$ become illuminated, and this rotation is being done, i.e., α keeps growing, until the entire $\mathcal{D}$ is illuminated. The last point in $\mathcal{D}$ to be illuminated, will be the point that has maximum angular distance to its nearest ray, and so the Brocard angle will be

$$\alpha^* = \max_{x \in \mathcal{D}} \min_{r \in \mathcal{R}} d_{\angle}(x, r).$$

By the properties of the angular distance function, the last illuminated point will be on a vertex or an edge of the diagram; more details are found in Section 4.1.

For a better comprehension, refer to the two examples of Fig. 2.10. In Fig. 2.10a given is a set of 5 rays and the target domain is $\mathbb{R}^2$. In Fig. 2.10b given is a set of 8 rays bounding a convex polygon, and the target domain is the interior of the polygon. In both examples, the Voronoi vertex realizing the Brocard angle is highlighted together with the 3 rays that induce it.

Since the Brocard angle is realized at a vertex or an edge of the diagram, this means that once we are given the diagram we can traverse its graph structure to

find the vertex or edge of maximum distance. Traversing a planar graph can be done using standard methods in linear time in its size. Hence, there is an interest in studying the rotating rays Voronoi diagram in diverse settings. We will see in Chapter 4 some properties of the points realizing the Brocard angle.

As a final note, we remark, that apart from the application of the rotating rays Voronoi diagram to the Brocard illumination problem, there are many real-world problems where devices have a limited field of view, and can be modeled as floodlight illumination problems with floodlights of uniform aperture. For instance, these can be (surveillance) cameras, or directional antennae; see Berman et al. [2007], Kranakis et al. [2011], Neishaboori et al. [2014], and Czyzowicz et al. [2015] for some examples. It would be interesting to examine how the applications of the rotating rays Voronoi could be broadened by considering such problems.

Other related Voronoi diagrams. As already mentioned, the rotating rays Voronoi diagram has not been considered before. Following, we describe two diagrams that have been studied in the literature and demonstrate some similarities.

In their work, de Berg et al. [2017] defined a Voronoi diagram with *rotational distance costs*. The sites are rays and the angular distance they consider is *unoriented* (bidirectional) in contrast to the one we consider which is oriented (unidirectional). In the simple *angle-only* case they consider, the two diagrams present similarities. In their more general case they consider, they incorporate in the distance measure some sense of movement; this is because they were motivated by the study of *dominance regions* in the analysis of soccer matches, as proposed by Taki et al. [1996].

Another diagram, which is in some sense related, is the *angular Voronoi diagram* studied by Asano et al. [2006] for the purpose of *mesh improvement*. There the input sites are line segments and the distance measure is the *visual angle*.

Chapter 3

Farthest color Voronoi diagram

This chapter presents our results on the farthest color Voronoi diagram. It is based on the following publication:

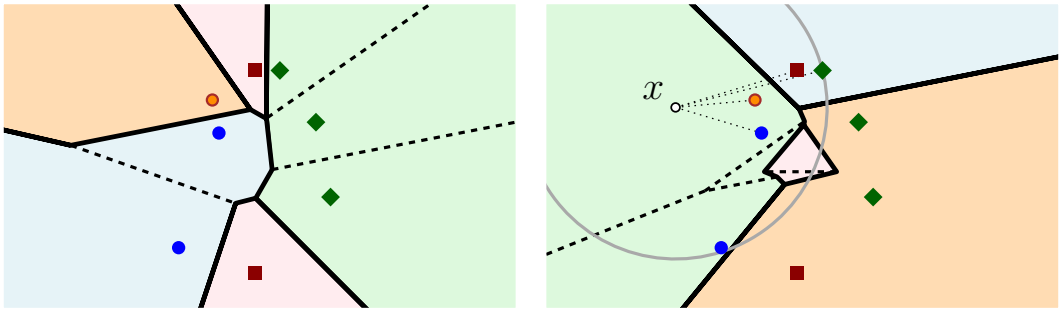
> *I. Mantas, E. Papadopoulou, V. Sacristán, and R. Silveira.* Farthest Color Voronoi Diagrams: Complexity and Algorithms. In *Proceedings of the 14th Latin American Symposium on Theoretical Informatis (LATIN 2020)*, pages 151-164. Springer, 2021.

In Section 3.1 we give some notation, and basic notions related to the diagram. In Section 3.2 we look into the structural properties of the diagram and refine its combinatorial complexity bound. In Section 3.3 we study necessary and sufficient conditions under which the diagram has linear combinatorial complexity. In Section 3.4 we consider classes of linearly separable input clusters, and give a lower bound construction that achieves a worst-case quadratic complexity. Section 3.5 is concerned with construction algorithms, and Section 3.6 concludes the chapter.

3.1 Preliminaries

Let the input be $\mathcal{P} := \{P_1, ..., P_m\}$, a set of m clusters of points in $\mathbb{R}^2$, $m > 1$, where no two clusters share a common point. Let the set of all points be $\mathcal{P}^* = \bigcup_{P_i \in \mathcal{P}} P_i$, with $|\mathcal{P}^*| = n$. For simplicity, we assume that $\mathcal{P}^*$ is in *general position*, i.e., no three points are collinear and no four points are cocircular.

Notation and definitions. Throughout this chapter we use the following notation. Given two points p and q, we denote by $d(p,q)$ the Euclidean distance



(a) Diagram NCVD($\mathcal{P}$) (augmented).

(b) Diagram FCVD($\mathcal{P}$) (augmented). The circle shows the farthest color disk of x.

Figure 3.1. The color Voronoi diagrams of a set $\mathcal{P}$, where $m = 4$ and $n = 8$.

between p and q, by $L(p,q)$ the line through p and q, by $\overline{pq}$ the line segment with endpoints p and q, and by $b(p,q)$ the Euclidean bisector of p and q. Given three points p,q and r, we denote by $C(p,q,r)$ the circle through p,q and r and by $D(p,q,r)$ the corresponding disk.

The convex hull of a set of points P is denoted by $CH(P)$. Given a planar region $f \subset \mathbb{R}^2$, we denote by ∂f its boundary, and by $\overline{f}$ its closure. We call each connected component of a planar region f, a face of f.

Two clusters P and Q are called *linearly separable* if their convex hulls are disjoint, i.e., $CH(P) \cap CH(Q) = \varnothing$. A set of clusters is called linearly separable if all clusters are pairwise linearly separable.

Recall, that the *ordinary Voronoi diagram* of a point set P is the subdivision of $\mathbb{R}^2$ into *ordinary Voronoi regions*, where the region of a point $p \in P$ is $vreg(p,P) = \{x \in \mathbb{R}^2 \,|\, d(x,p) < d(x,q)\ \forall q \in P \backslash \{p\}\}$. Further, the graph structure of the diagram, sometimes called the *Voronoi skeleton*, of P is $\mathrm{VD}(P) = \mathbb{R}^2 \backslash \bigcup_{p \in P} vreg(p,P)$.

In color Voronoi diagrams, we consider the *minimum distance* (see Definition 1.4) between a point $x \in \mathbb{R}^2$ and a cluster P, i.e.,

$$d_c(x,P) = min_{p \in P} d(x,p).$$

Recall that the *nearest color Voronoi diagram* of $\mathcal{P}$ is a subdivision of $\mathbb{R}^2$ into *nearest color (Voronoi) regions* (see Definition 1.5), where the nearest color region of a cluster $P_i \in \mathcal{P}$ is

$$n_c reg(P_i, \mathcal{P}) = \{x \in \mathbb{R}^2 \,|\, d_c(x,P_i) < d_c(x,P_j)\ \forall P_j \in \mathcal{P} \backslash \{P_i\}\}.$$

The nearest color Voronoi diagram of $\mathcal{P}$ can be directly derived from the ordinary Voronoi diagram of $\mathcal{P}^*$, as the nearest color Voronoi region of a cluster P_i

corresponds to the union of the ordinary Voronoi regions of its points in $\mathcal{P}^*$, in particular, $\overline{n_c reg(P_i, \mathcal{P})} = \bigcup_{p \in P_i} \overline{vreg(p, \mathcal{P}^*)}$. We denote the graph structure of the nearest color Voronoi diagram by

$$\text{NCVD}(\mathcal{P}) := \mathbb{R}^2 \setminus \bigcup_{P_i \in \mathcal{P}} n_c reg(P_i, \mathcal{P}).$$

An example of a nearest color Voronoi diagram is illustrated in Fig. 3.1a.

We defined the *farthest color Voronoi diagram* of $\mathcal{P}$, as the subdivision of $\mathbb{R}^2$ into farther color (Voronoi) regions (see Definition 1.6), where the *farthest color region of a cluster* $P_i \in \mathcal{P}$ is

$$f_c reg(P_i, \mathcal{P}) = \{x \in \mathbb{R}^2 \mid d_c(x, P_i) > d_c(x, P_j) \ \forall P_j \in \mathcal{P} \setminus \{P_i\}\},$$

and the *farthest color region of a point* $p \in P_i$ is

$$f_c reg(p, \mathcal{P}) = \{x \in f_c reg(P_i, \mathcal{P}) \mid d(x, p) < d(x, q) \ \forall q \in P_i \setminus \{p\}\}.$$

We denote the graph structure of the farthest color Voronoi diagram of $\mathcal{P}$ by

$$\text{FCVD}(\mathcal{P}) := \mathbb{R}^2 \setminus \bigcup_{P_i \in \mathcal{P}} f_c reg(P_i, \mathcal{P}).$$

As we have already discussed, a farthest color region $f_c reg(P_i, \mathcal{P})$ is subdivided into finer regions by the Voronoi skeleton $\text{VD}(P_i)$. We call the subgraph $\text{VD}(P_i) \cap f_c reg(P_i, \mathcal{P})$ the *internal skeleton* of $f_c reg(P_i, \mathcal{P})$, We denote the graph structure of the farthest color Voronoi diagram, *augmented* by the internal skeletons, by

$$\text{FCVD}_a(\mathcal{P}) := \mathbb{R}^2 \setminus \bigcup_{p \in \mathcal{P}^*} f_c reg(p, \mathcal{P}).$$

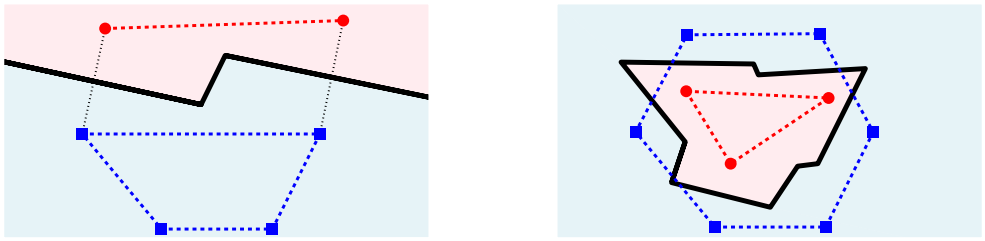
An example of a farthest color Voronoi diagram is illustrated in Fig. 3.1b, where internal skeletons are shown with dashed edges.

Color bisectors. We now define the bisector between two clusters of points under the minimum distance.

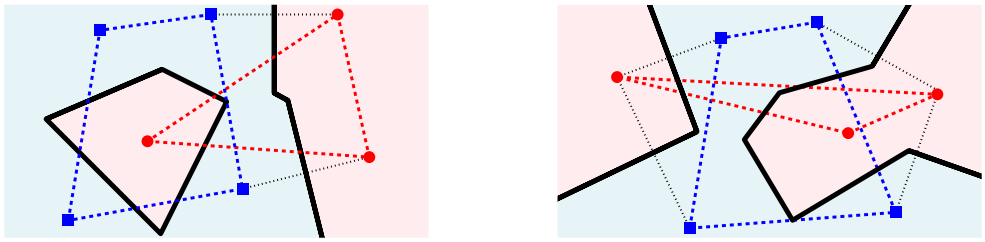
Definition 3.1. Given two clusters P and Q, their *color bisector* is

$$b_c(P, Q) := \{x \in \mathbb{R}^2 \mid d_c(x, P) = d_c(x, Q)\}.$$

Refer to Fig. 3.2 for an illustration of various cases of color bisectors. Color bisectors have the following properties.



(a) $\mathrm{CH}(P) \cap \mathrm{CH}(Q) = \varnothing$ (b) $\mathrm{CH}(P) \subset \mathrm{CH}(Q)$



(c) $\mathrm{CH}(P) \cap \mathrm{CH}(Q) \neq \varnothing$ (d) $\mathrm{CH}(P) \cap \mathrm{CH}(Q) \neq \varnothing$

Figure 3.2. Different color bisectors of two clusters P (●) and Q (■). The dashed edges are $\partial\mathrm{CH}(P)$ and $\partial\mathrm{CH}(Q)$. The thin dotted edges are the segments on $\partial\mathrm{CH}(P \cup Q)$ with one endpoint in P and one in Q.

Lemma 3.1. *The bisector $b_c(P,Q)$ is a subgraph of $VD(P \cup Q)$ consisting of disjoint unbounded chains and cycles. Each unbounded chain of $b_c(P,Q)$ corresponds to a distinct pair of edges on $\partial CH(P \cup Q)$ with one endpoint in P and one in Q.*

Proof. The bisector $b_c(P,Q)$ is a subgraph of $\mathrm{VD}(P \cup Q)$, by its definition. The corresponding Voronoi diagram can be seen as a 2-colorable map, where $vreg(p, P \cup Q)$ is colored blue and $vreg(q, P \cup Q)$ is colored red, for any $p \in P$ and $q \in Q$. The boundary of this map is exactly the color bisector $b_c(P,Q)$, thus, it may only consist of disjoint unbounded chains and cycles; refer, e.g., to Preparata and Shamos [2012]. The remaining properties of $b_c(P,Q)$ directly derive from the well-known relation between unbounded Voronoi edges in $\mathrm{VD}(P \cup Q)$ and edges on the convex hull $CH(\mathcal{P} \cup Q)$. □

As a corollary, a color bisector has complexity $O(|P| + |Q|)$. From the correspondence between segments on the boundary of $CH(P \cup Q)$ and the unbounded chains of $b_c(P,Q)$, it follows that if P and Q are linearly separable, then $b_c(P,Q)$ is a single unbounded chain; see Fig. 3.2a. Further, $b_c(P,Q)$ consists only of cycles if and only if $P \subset CH(Q)$ or $Q \subset CH(P)$; see Fig. 3.2b.

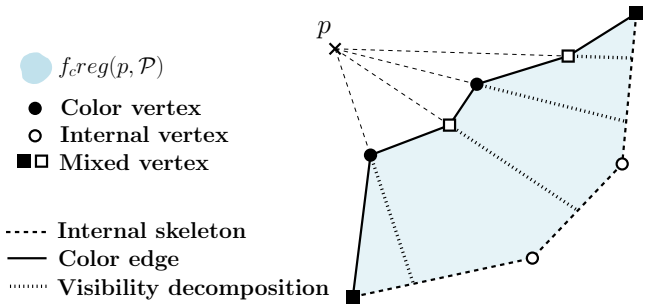


Figure 3.3. The features of the augmented farthest color Voronoi diagram illustrated on a bounded face of a point p.

Features of the farthest color Voronoi diagram. The augmented farthest color Voronoi diagram contains different types of edges and vertices; see Fig. 3.3 for an illustration of these features. These are the following.

- A *color edge* is a subset of a color bisector, so any point on a color edge is equidistant to the two clusters which induce it.
- An *internal edge* is an edge of the internal skeleton, so it is an edge of the ordinary Voronoi diagram of a cluster.
- A *color vertex* is incident to three color edges, so it is equidistant to the three clusters that induce the three color edges.
- A *mixed vertex* is incident to two color edges and one internal edge, so it is equidistant to the two clusters inducing the color edges.
- An *internal vertex* is a vertex of the internal skeleton, so it is a vertex of the ordinary Voronoi diagram of a cluster.

The combinatorial complexity of the farthest color Voronoi diagram is given by the sum of the number of edges, vertices and faces of $\mathrm{FCVD}_a(\mathcal{P})$. By Euler's formula it suffices to bound the number of any of these elements and the complexity of the diagram follows.

The augmented $\mathrm{FCVD}_a(\mathcal{P})$ defines a *farthest color disk*, for every point $x \in \mathbb{R}^2$. This is the disk centered at x of radius $d(x,p)$, where $x \in \overline{f_c reg(p,\mathcal{P})}$ and $p \in P$; see for example the circle centered at x in Fig. 3.1b. This disk contains no point of P in its interior, and it contains at least one point from every cluster in $\mathcal{P}$.

Cluster hull. We now review the definition of the *cluster hull* of $\mathcal{P}$ from Papadopoulou and Lee [2004]; refer to Fig. 3.4 for an illustration. It is a (non-simple) closed polygonal chain, which characterizes the unbounded faces of the Hausdorff Voronoi diagram. It is of interest, as we later show in Section 3.2.1 that it also characterizes the unbounded faces of the farthest color Voronoi diagram.

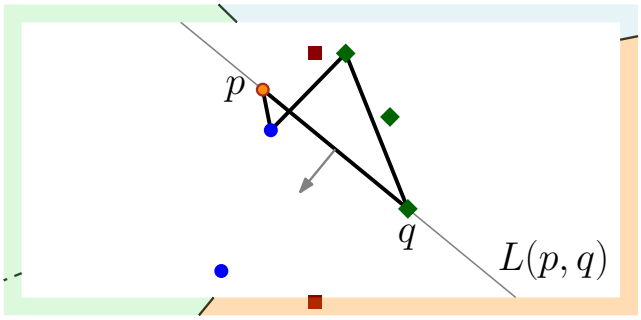


Figure 3.4. The polygonal chain is the cluster hull of the set of clusters $\mathcal{P}$ shown in Fig. 3.1. Points p and q are hull vertices and $\overline{pq}$ is a hull edge. The arrow depicts the unit vector normal to $\overline{pq}$. The surrounding frame illustrates the sequence of unbounded edges and faces of $\mathrm{FCVD}_a(\mathcal{P})$.

Definition 3.2. Given a set of clusters $\mathcal{P}$, a point $p \in P$ is a *hull vertex* if there exists a line ℓ through p such that P lies entirely in one halfplane defined by ℓ, while every cluster Q in $\mathcal{P}\backslash\{P\}$ intersects the other halfplane.

Given two hull vertices $p \in P$ and $q \in Q$ (with possibly $P = Q$), the segment $\overline{pq}$ is a *hull edge* if the line $L(p,q)$ leaves P and Q entirely on one side and every cluster in $\mathcal{P}\backslash\{P,Q\}$ intersects the other halfplane. The hull edge $\overline{pq}$ is associated with a unit vector normal to $\overline{pq}$, that points in the direction away from P and Q.

The hull edges sorted by the circular ordering of their normal vectors define a closed polygonal chain, called the *cluster hull* of $\mathcal{P}$ and denoted by $CLH(\mathcal{P})$.

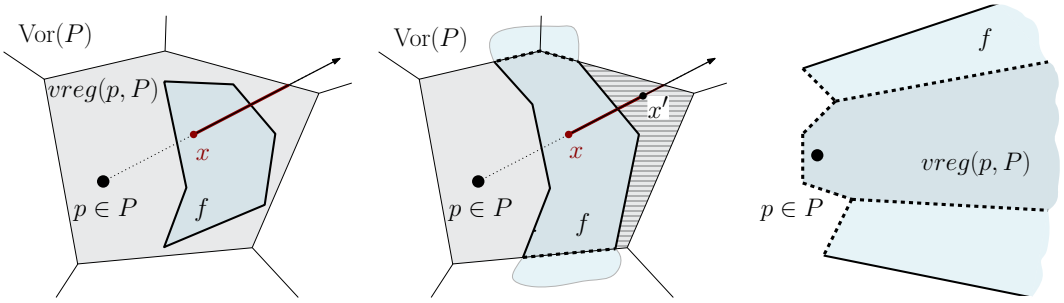
Note that given a cluster $P \in \mathcal{P}$, only points on $\partial CH(P)$ can be hull vertices of $CLH(\mathcal{P})$, as only points on $\partial CH(P)$ can have unbounded faces in $\mathrm{FCVD}(\mathcal{P})$. This is because given a point $p \in P$, $vreg(p,P)$ is unbounded if and only if p lies on $\partial CH(P)$, combined with the fact that $f_c reg(p,\mathcal{P}) \subset vreg(p,P)$.

3.2 Properties and combinatorial complexity

In this section we study the properties and the complexity of the farthest color Voronoi diagram. In Section 3.2.1, we look into the structure of farthest color regions and we bound the number of unbounded faces. In Section 3.2.2, we bound the number of bounded faces and refine the complexity bounds.

3.2.1 Structural properties and unbounded faces

Farthest color regions satisfy the following *visibility property*.



(a) A bounded face with empty T; point x violates Proposition 3.2.

(b) A bounded face with disconnected T; point x violates Proposition 3.2.

(c) An unbounded face with a tree of T having two unbounded edges; point p lies in f.

Figure 3.5. Illustrations for the proof of Proposition 3.3. The blue highlighted area is the face f. The dashed lines are the internal skeleton $T = f \cap \mathrm{VD}(P)$. The gray area is $\mathrm{vreg}(p, P)$.

Proposition 3.2. *For any point $x \in f_c reg(p, \mathcal{P})$, let r be the ray emanating from x, along $L(p, x)$, in the direction away from p. Then the region $f_c reg(p, \mathcal{P})$ contains the entire segment $r \cap vreg(p, P)$, where p is a point in P.*

This property is symmetric to a corresponding visibility property of the Hausdorff Voronoi diagram (see Papadopoulou and Lee [2004]) and has already been pointed out in Bae [2012]. Using this we can prove the following structural property of a farthest color region.

Proposition 3.3. *A face f of $f_c reg(P, \mathcal{P})$ satisfies the following:*

(1) If f is bounded, then its internal skeleton $f \cap VD(P)$ is a non-empty tree whose leaves are mixed vertices of ∂f.

(2) If f is unbounded, then its internal skeleton $f \cap VD(P)$ may be empty. If non-empty, it is a forest whose leaves are mixed vertices of ∂f and points at infinity; each leaf at infinity is the root of a distinct tree of this forest.

Proof. Let $T = f \cap \mathrm{VD}(P)$ be the internal skeleton of f. T may not contain a cycle, as no point $p \in P$ is contained in f (as $m \geqslant 2$), and so $vreg(p, P)$ cannot be entirely contained in f. Thus, T must be a tree or a forest. By definition, the leaves of T are mixed vertices of f, or points at infinity, if f is unbounded.

Suppose now that f is bounded; refer also to Fig. 3.5a. If $f \cap \mathrm{VD}(P) = \varnothing$, then $f \subset vreg(p, P)$, for some $p \in P$. But then the visibility property would not hold for any point x in f. Thus, $T \neq \varnothing$.

Suppose further that T is not connected; refer also to Fig. 3.5b. Then, f splits the region $vreg(p,P)$, for some $p \in P$, in two connected components. Point p lies in one of the two components, as $p \notin f$. For any point x' in the other component there exists a point $x \in f$ not satisfying the visibility property. Hence, T is connected.

Finally, suppose that f is unbounded. Then T may be empty, as for example the blue region in the diagram of Fig. 3.1b. If T is non-empty, each tree of T must have exactly one unbounded edge; refer to Fig. 3.5c. Suppose a tree has two unbounded edges, then there exists a point $p \in P$, having $vreg(p,P)$ bounded by these two edges, and since these two edges are connected, this implies that $vreg(p,P) \subseteq f$, so $p \in f$, a contradiction. □

The following lemma shows that unbounded faces of FCVD($\mathcal{P}$) are characterized by the cluster hull $CLH(\mathcal{P})$.

Lemma 3.4. *Given a set of clusters $\mathcal{P}$ the following two hold:*

(1) A region $f_c reg(p,\mathcal{P})$ is unbounded if and only if p is a vertex of $CLH(\mathcal{P})$.

(2) A counterclockwise traversal of $CLH(\mathcal{P})$ corresponds to the clockwise ordering of the unbounded edges in $FCVD_a(\mathcal{P})$.

Proof. We first show that there is a 1-to-1 correspondence between the unbounded edges of the farthest color Voronoi diagram and the Hausdorff Voronoi diagram. Refer also to Fig. 3.6. More specifically, let $e = \overline{pq}$ be an edge of $CLH(\mathcal{P})$, with $p \in P$, $q \in Q$ (it is possible that $P = Q$), and let H_R and H_L be the two halfplanes induced by the line $L(p,q)$. We show that $\text{FCVD}_a(\mathcal{P})$ has an unbounded edge $e_f \subset b(p,q)$, which is a ray pointing towards H_L, if and only if the HVD has an unbounded edge $e_h \subset b(p,q)$ pointing on the opposite direction of e_f, i.e., towards H_R.

Let e_h be an unbounded edge of the Hausdorff Voronoi diagram of $\mathcal{P}$ directed towards H_R. By Papadopoulou and Lee [2004], this is equivalent to: (a) $\overline{H_R}$ entirely contains clusters P and Q; and (b) H_R does not entirely contain any cluster $R \in \mathcal{P}\backslash\{P,Q\}$.

Let e_f be an unbounded edge of $\text{FCVD}_a(\mathcal{P})$ directed towards H_L. By the definition of a farthest color disk (degenerated to a halfplane) this is equivalent to: (a') H_L does not contain any point from clusters P and Q; and (b') $\overline{H_L}$ contains at least one point from every cluster $R \in \mathcal{P}\backslash\{P,Q\}$.

But both these statements are equivalent, i.e., $(a) \Leftrightarrow (a')$ and $(b) \Leftrightarrow (b')$, defining the one-to-one correspondence between the unbounded edges of the

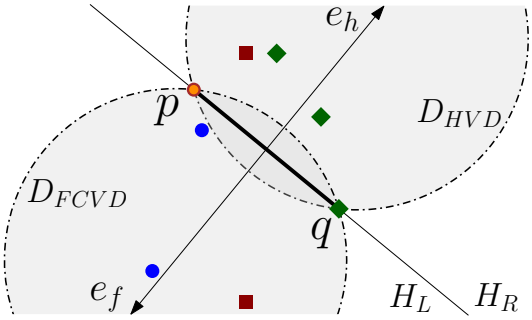


Figure 3.6. Illustration for the proof of Lemma 3.4. D_{FCVD} is a farthest color disk and D_{HVD} a Hausdorff disk[1]. D_{FCVD} with infinite radius degenerates to the halfplane H_L and D_{HVD} degenerates to the halfplane H_R.

two diagrams. Hence, statement (1) follows from an analogous statement regarding the Hausdorff Voronoi diagram; see Papadopoulou and Lee [2004].

In the farthest color Voronoi diagram, the cyclic orientation of the unbounded edges is reversed from the Hausdorff Voronoi diagram. This is because each normal vector associated to a hull edge which corresponds to an unbounded edge of $\text{FCVD}_a(\mathcal{P})$) points to the opposite direction as compared to the Hausdorff Voronoi diagram of $\mathcal{P}$. Hence, statement (2) also follows. □

The cluster hull of $\mathcal{P}$ has $O(n)$ complexity, and in combination with Lemma 3.4, this directly implies the following property.

Proposition 3.5. *$FCVD_a(\mathcal{P})$ has $O(n)$ unbounded faces.*

Proof. We argue that $CLH(\mathcal{P})$ has $O(n)$ complexity, as implied by Papadopoulou and Lee [2004]. Then, in combination with Lemma 3.4, the claim can be directly derived.

Papadopoulou and Lee [2004] proved that the Hausdorff Voronoi diagram of $\mathcal{P}$ has complexity $O(n+x)$, where x is the number of mixed vertices of the diagram (defined analogously to the farthest color Voronoi diagram). A mixed vertex is incident to an internal edge, and each bisector can have at most two occurrences as internal edge at infinity (two occurrences can appear only if the corresponding cluster is of size two). There are $O(n)$ overall internal edges, hence, there are $O(n)$ mixed vertices related to the unbounded edges of the Hausdorff Voronoi diagram. Thus, $CLH(\mathcal{P})$ has complexity $O(n)$, and because of Lemma 3.4, $\text{FCVD}_a(\mathcal{P})$ has $O(n)$ unbounded faces. □

[1] A *Hausdorff disk* is defined analogously to a farthest color disk: a Hausdorff disk defined by $p \in P$ and $q \in Q$, contains P and Q in its interior, but does not contain any other cluster $R \in \mathcal{P} \backslash \{P, Q\}$.

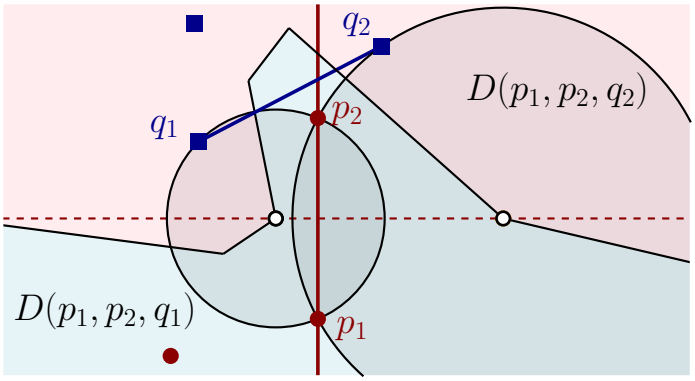


Figure 3.7. Illustration of a straddle (q_1, q_2 (■) straddles p_1, p_2 (●)) and the proof of Lemma 3.6.

3.2.2 The straddling number and bounded faces

It follows Proposition 3.5 that only bounded faces can contribute to the possibly non-linear combinatorial complexity of FCVD($\mathcal{P}$). Thus, we it suffices to count the mixed vertices incident to bounded faces. To this aim we define the notion of *straddles*.

Definition 3.3. A cluster Q, and in particular a pair of points $q_1, q_2 \in Q$, is said to *straddle* a pair of points $p_1, p_2 \in P$ if the disks $D(p_1, p_2, q_1)$ and $D(p_1, p_2, q_2)$ contain no points of P and Q in their interior.

Refer to Fig. 3.7 for an illustration of a straddle. The term *straddle* is motivated by the following necessary condition for a straddle to occur.

Lemma 3.6. *If $D(p_1, p_2, q_1)$ and $D(p_1, p_2, q_2)$ contain no points of P and Q in their interior, then the segment $\overline{q_1 q_2}$ intersects (straddles) the line $L(p_1, p_2)$.*

Proof. Refer to Fig. 3.7. Suppose that the disks $D(p_1, p_2, q_1)$ and $D(p_1, p_2, q_2)$ are empty. Then, point q_1 lies on $C(p_1, p_2, q_1) \backslash D(p_1, p_2, q_2)$ and point q_2 lies on $C(p_1, p_2, q_1) \backslash D(p_1, p_2, q_2)$. Hence, line $L(p_1, p_2)$ separates them and so $\overline{q_1 q_2} \cap L(p_1, p_2) \neq \varnothing$. □

Note that if points q_1, q_2 straddle points p_1, p_2, then the segments $\overline{q_1 q_2}$ and $\overline{p_1 p_2}$ may or may not intersect. We define the *straddling number* parameter as follows.

Definition 3.4. Given a set of clusters $\mathcal{P}$, let $s(p_i, p_j)$ denote the number of clusters in $\mathcal{P}$ that straddle p_i, p_j. The *straddling number of* $\mathcal{P}$ is

$$s(\mathcal{P}) := \sum_{P \in \mathcal{P}} \sum_{(p_i, p_j) \in P} s(p_i, p_j).$$

Lemma 3.7. *The straddling number $s(\mathcal{P})$ is $O(mn)$.*

Proof. A straddle induced to (p_i, p_j) by a pair (q_a, q_b) corresponds to a pair of vertices of the color bisector $b_c(P,Q)$, which are the centers of the disks $D(p_i, p_j, q_a)$ and $D(p_i, p_j, q_b)$. Such vertices are incident to a Voronoi edge of $\mathrm{VD}(P \cup Q)$, which is a portion of the bisector $b(p_i, p_j)$.

A bisector $b(p_i, p_j)$ can have only one occurrence as a Voronoi edge in the ordinary Voronoi diagram $\mathrm{VD}(P \cup Q)$, thus, Q may straddle (p_i, p_j) at most once; hence $s(p_i, p_j) \leqslant m - 1$. Further, only pairs (p_i, p_j) inducing Voronoi edges in $\mathrm{VD}(P)$ can be straddled by some other $Q \in \mathcal{P}$. Overall, there are $\sum_{P \in \mathcal{P}} O(|P|) = O(n)$ Voronoi edges in $\mathrm{VD}(P)$, so $O(n)$ pairs of points out of all m clusters may be straddled. Summing up the straddling number is $O(mn)$. □

In the following lemma, we show a property of consecutive mixed vertices along a bisector, which we then use to bound the total number of mixed vertices.

Lemma 3.8. *Let v_1, v_2 be two consecutive mixed vertices on bisector $b(p_1, p_2)$, where $p_1, p_2 \in P$, such that the segment $\overline{v_1 v_2}$ is outside of $f_c reg(P, \mathcal{P})$ (see Fig. 3.8). Then v_1 and v_2 are induced by two points q and r respectively, which belong to the same cluster $Q \in \mathcal{P}$.*

Proof. Refer to Fig. 3.8. Let q and r be the two points that together with p_1 and p_2 induce vertices v_1 and v_2 respectively. Then v_1 is the center of $D(p_1, p_2, q)$, and v_2 is the center of $D(p_1, p_2, r)$. We show that if $q \in Q$, then $r \in Q$ as well.

Without loss of generality let $b(p_1, p_2)$ be horizontal and v_1 be to the left of v_2. This implies that q is to the left of $L(p_1, p_2)$ and r to the right. Suppose for contradiction that $r \in R \neq Q$. Disk $D(p_1, p_2, r)$ is a farthest color disk. It contains at least one point from every cluster $\mathcal{P} \backslash \{P, R\}$, so it contains a point $q_x \in Q$. Disk $D(p_1, p_2, q)$ is also a farthest color disk and contains no point from clusters P and Q in its interior. So, q_x lies in $D(p_1, p_2, r) \backslash D(p_1, p_2, q)$. Point q_x also defines a farthest color disk $D(p_1, p_2, q_x)$; the center v_x of $D(p_1, p_2, q_x)$ is a mixed vertex along $b(p_1, p_2)$, and since $q_x \in D(p_1, p_2, r) \backslash D(p_1, p_2, q)$, the vertex v_x lies between v_1 and v_2. This is a contradiction as $\overline{v_1 v_2}$ lies outside $f_c reg(P_i, \mathcal{P})$ and no other vertex v_x can be on $b(p_1, p_2)$ between v_1 and v_2. □

Proposition 3.9. *FCVD$(\mathcal{P})$ has $O(n + s(\mathcal{P}))$ bounded faces.*

Proof. For any cluster $P \in \mathcal{P}$ and for any pair (p_i, p_j) inducing an edge e in $\mathrm{VD}(P)$, we count the number of mixed vertices appearing along e. By Lemma 3.8, only the two outermost mixed vertices along e may not be the result of a straddle, as any other consecutive such pair of vertices must be. Thus, there at most

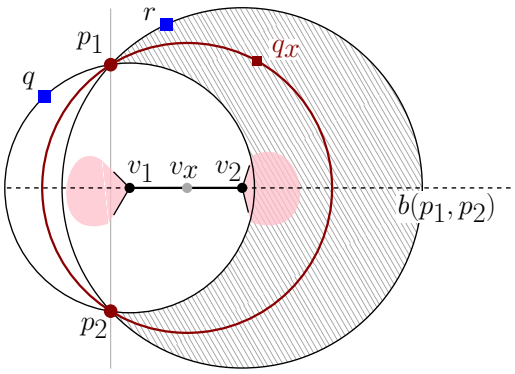


Figure 3.8. Illustration for the proof of Lemma 3.8. Point $q_x \in D(p_1,p_2,r)\backslash D(p_1,p_2,q)$ (shown shaded), so $u_x \in \overline{u_1u_2}$.

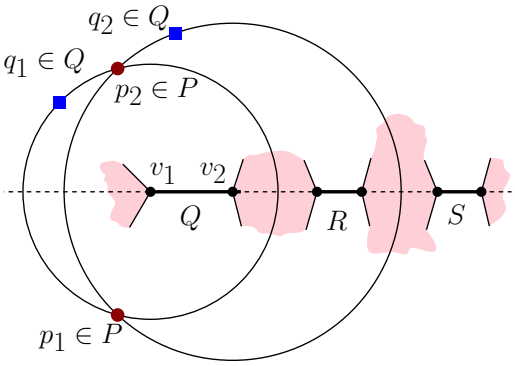


Figure 3.9. A sequence of consecutive pairs of mixed vertices appearing along bisector $b(p_1,p_2)$. The red shaded regions indicate $\mathrm{f_c reg}(P,\mathcal{P})$.

$2\cdot s(p_i,p_j)+2$ mixed vertices incident to e. These vertices, if present, appear consecutively along e; see Fig. 3.9. Each pair of consecutive mixed vertices may create one bounded face incident to e, so, in total, are at most $s(p_i,p_j)+1$ bounded faces incident to e; see the red regions in Fig. 3.9.

The ordinary Voronoi diagrams of the clusters in $\mathcal{P}$ have $\sum_{P\in\mathcal{P}} O(|P|) = O(n)$ Voronoi edges, hence, $\sum_{P\in\mathcal{P}}\sum_{(p_i,p_j)\in P}(1+s(p_i,p_j)) = n+s(\mathcal{P})$, concluding the proof. □

Combining Proposition 3.5 and Proposition 3.9, we can obtain the following.

Theorem 3.1. *FCVD*$(\mathcal{P})$ *has* $O(n+s(\mathcal{P}))$ *combinatorial complexity.*

The above theorem refines the $O(mn)$ upper bound of Abellanas et al. [2001b], and also gives a first sufficient condition for FCVD$(\mathcal{P})$ to have $O(n)$ complexity, that is, if $s(\mathcal{P}) = O(n)$.

3.3 Conditions for linear combinatorial complexity

In this section we delve further into the structural properties of the farthest color Voronoi diagram and look for conditions under which the diagram has $O(n)$ complexity.

Admissible clusters. We first consider its relation with abstract Voronoi diagrams. The AVD axioms, described in Section 2.1.2, in the context of color Voronoi diagrams can be described as follows. For every subset $\mathcal{P}' \subseteq \mathcal{P}$:

(A1) The color bisector $b_c(P,Q)$, for any two clusters $P,Q \in \mathcal{P}'$, is an unbounded Jordan curve.
(A2) The nearest color region $n_c reg(P,\mathcal{P}')$, for any cluster $P \in \mathcal{P}'$, is non-empty and connected.
(A3) The closure of the union of all nearest color regions in NCVD$(\mathcal{P}')$ covers $\mathbb{R}^2$.

Recall that we call admissible a set of input sites that satisfies the AVD axioms. For an admissible set of clusters, the properties of the farthest abstract Voronoi diagrams (see Mehlhorn et al. [2001]) directly apply to FCVD$(\mathcal{P})$, and we can derive the following.

Proposition 3.10. *If $\mathcal{P}$ is admissible, then FCVD$(\mathcal{P})$ is a tree of complexity $O(n)$.*

The tree structure of FCVD$(\mathcal{P})$ can be observed in the instance of Fig. 3.11b. Note that the augmented FCVD$_a(\mathcal{P})$ might have some faces bounded by internal edges. Following we give a necessary and sufficient condition for a set of clusters to be admissible.

Proposition 3.11. *A set of clusters $\mathcal{P}$ is admissible if and only if the following two conditions hold for every $P \in \mathcal{P}$:*
(1) P is not contained within the convex hull of any other cluster in $\mathcal{P}$.
(2) The nearest color region $n_c reg(P,\mathcal{P})$ is connected.

Proof. Suppose $\mathcal{P}$ is admissible. Then by (A2), condition (2) trivially holds. By (A1), any bisector is unbounded, thus, by Lemma 1, condition (1) holds as well.

Suppose $\mathcal{P}$ satisfies conditions (1) and (2), and let $\mathcal{P}'$ be a subset of $\mathcal{P}$. The ordinary Voronoi diagram of $\mathcal{P}'^*$ covers $\mathbb{R}^2$, and for any $P \in \mathcal{P}'$, the closure of $n_c reg(P,\mathcal{P}')$ equals $\bigcup_{p\in P} \overline{vreg(p,\mathcal{P}'^*)}$. So, it follows that each region $n_c reg(P,\mathcal{P}')$ is non-empty, and that axiom (A3) holds for FCVD$(\mathcal{P}')$.

In ordinary Voronoi diagrams, given a point set P and a subset $P' \subseteq P$, it is known that $vreg(p,P) \subseteq vreg(p,P')$, for any $p \in P' \subseteq P$. As a result, $n_c reg(P,\mathcal{P}) \subseteq n_c reg(P,\mathcal{P}')$ for any $\mathcal{P}' \subset \mathcal{P}$, and since $n_c reg(P,\mathcal{P})$ is connected, then also $n_c reg(P,\mathcal{P}')$ is connected. So, the connectivity requirement of axiom (A2) is satisfied.

It remains to show axiom (A1). Recall that the bisector $b_c(P,Q)$ is a subgraph of VD$(P \cup Q)$. We already proved that $n_c reg(P,\mathcal{P}')$ is connected for every $\mathcal{P}' \subseteq \mathcal{P}$, where $P \in \mathcal{P}'$. Thus, for any $P,Q \in \mathcal{P}$, both $n_c reg(P,P\cup Q)$ and $n_c reg(Q,P\cup Q)$ are connected, and by the properties of color bisectors, $b_c(P,Q)$ is a single connected component. Finally, by condition (1), no cluster is contained in the convex hull of another cluster, hence, due to Lemma 3.1, every color bisector is a single unbounded curve, satisfying (A1). □

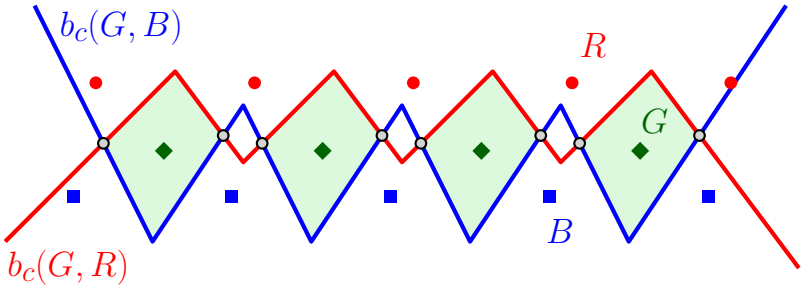


Figure 3.10. A set of 3 linearly separable clusters R(●), G (◆), and B(■), where two bisectors $b_c(G,B)$ and $b_c(G,R)$ intersect $\Theta(n)$ times. This causes the (shaded) nearest color region $\mathrm{n_c reg}(G,\{R,G,B\})$ to have $\Theta(n)$ faces.

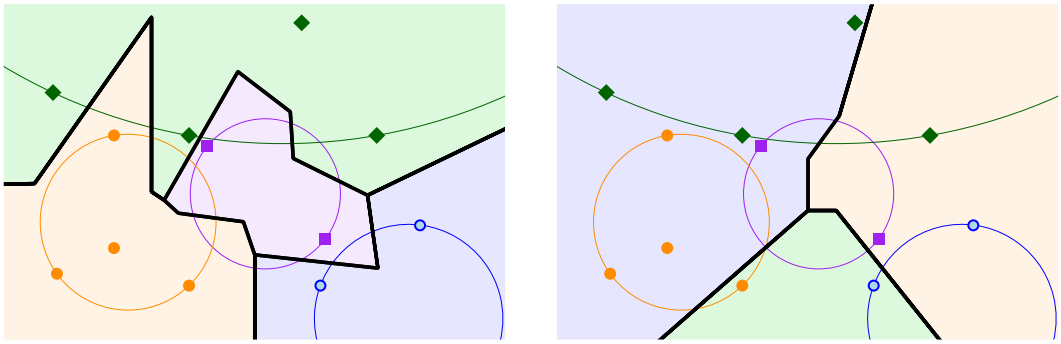
We remark that linear separability is not a condition under which the farthest color Voronoi diagram falls under the framework of abstract Voronoi diagrams. For example, the bisectors of three linearly separable clusters $b_c(P,Q)$ and $b_c(Q,R)$, may intersect $\Theta(|P|+|Q|+|R|)$ times, as shown in Fig. 3.10, which directly violates the region-connectivity requirement of axiom (A2). Thus, unlike the Hausdorff Voronoi diagram (see Papadopoulou and Lee [2004]), linear separability is not a condition that can guarantee an a farthest color Voronoi diagram of $O(n)$ complexity.

Deciding admissibility of clusters. After having a necessary and sufficient condition for clusters to be admissible, we want to examine how efficiently we can check this condition. Following, we show that we can check the admissibility of linearly separable clusters in $O(n\log n)$ time. This is of particular interest, as later in Section 3.5, we describe how to construct the diagram of such clusters in $O(n\log n)$ time as well.

Proposition 3.12. *Given a linearly separable set of clusters $\mathcal{P}$, we can decide if $\mathcal{P}$ is admissible in $O(n\log n)$ time.*

Proof. $\mathcal{P}$ is linearly separable, so by Lemma 3.1 color bisectors are unbounded curves, satisfying condition (1) of Proposition 3.11. Hence, it suffices to check condition (2). This can be done by constructing the diagram NCVD($\mathcal{P}$) using standard $O(n\log n)$-time algorithms and then traversing NCVD($\mathcal{P}$) in $O(n)$ time to check the connectivity of the nearest color regions. □

If the input clusters are not linearly separable then the question is how efficiently can we check condition (1), i.e., that no cluster is contained with the convex of another. For this question, there exists a quadratic lower bound. More



(a) Diagram NCVD($\mathcal{P}$). (b) Diagram FCVD($\mathcal{P}$) (tree structure).

Figure 3.11. A set $\mathcal{P}$ of 4 disk-separable clusters together with the corresponding empty disks, and the two color Voronoi diagrams.

precisely, it is not possible to find containing pairs among m k-sided polygons in time $O(m^{2-\epsilon})$, for any $k = O(m^{\delta})$ and any $\epsilon, \delta > 0$. This was pointed out to us by David Eppstein in a post on the *Stack Exchange platform*[2]; see the post for the exact statement with its proof.

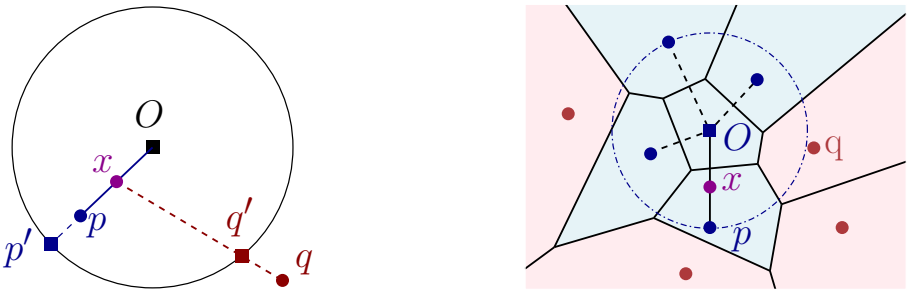
The above result is also interesting for the related farthest polygon Voronoi diagram studied by Cheong et al. [2011]. Bounded regions in this diagram occur when polygon containment relations are present, and dealing with them dominates the time complexity of the algorithms. A potential efficient identification of these relations could lead to improved algorithms.

Disk separable clusters. We now define *disk-separability* and show that it gives a sufficient condition for a set of clusters to be admissible. A set of clusters $\mathcal{P}$ is called *disk-separable* if, for every cluster $P \in \mathcal{P}$, there exists a disk $ED(P)$, which contains P and does not contain any point from any other cluster $Q \in \mathcal{P}$. Refer to Fig. 3.11 for an example of disk-separable clusters.

Theorem 3.2. *If $\mathcal{P}$ is disk-separable, then $\mathcal{P}$ is admissible.*

Proof. Suppose $\mathcal{P}$ is disk-separable. Then axiom (A3) and the non-emptiness of regions of (A2) trivially hold, similarly to the proof of Proposition 3.11. Further, axiom (A1) is satisfied as disk-separability implies linear separability. For any two P, Q we can find a separating line: if $ED(P) \cap ED(Q) \neq \emptyset$, consider the line through the intersection points of $ED(P) \cap ED(Q)$, else if $ED(P) \cap ED(Q) = \emptyset$, consider any line leaving $ED(P)$ and $ED(Q)$ in two different halfplanes. It remains to show that $n_c reg(P, \mathcal{P})$ is connected, for every cluster $P \in \mathcal{P}$.

[2]url: https://cstheory.stackexchange.com/questions/46154



(a) For any $x \in \overline{Op} : d(x,p) < d(x,q)$. (b) For any $p \in P : \overline{Op} \in \mathrm{n_c reg}(P,\mathcal{P})$.

Figure 3.12. Illustrations for the proof of Theorem 3.2.

First, observe that given a disk B centered at O, a point $p \in B$ and a point $q \notin B$, for any point $x \in \overline{Op}$: $d(x,p) < d(x,q)$ holds; see Fig. 3.12a. Indeed, the circle centered at x passing through p is fully contained in B, so the distance from x to any point outside B, such as q, is larger than the distance from x to p.

Let P be a cluster of $\mathcal{P}$, p be a point of P, and let $ED(P)$ denote a disk that contains P and no point from another cluster. Consider the line segment $\overline{Op}$, where O is the center of the disk $ED(P)$; refer also to Fig. 3.12b. For any point $x \in \overline{Op}$ the distance $d(x,p)$ is smaller than $d(x,q)$, for any $q \in \mathcal{P}^* \backslash \{P\}$. So, $x \in n_c reg(P,\mathcal{P})$ and segment $\overline{Op}$ lies entirely in $n_c reg(P,\mathcal{P})$. Therefore, $\forall x \in \overline{Op}$ and $\forall p \in P$, point $x \in n_c reg(P,\mathcal{P})$, and thus region $n_c reg(P,\mathcal{P})$ is connected. □

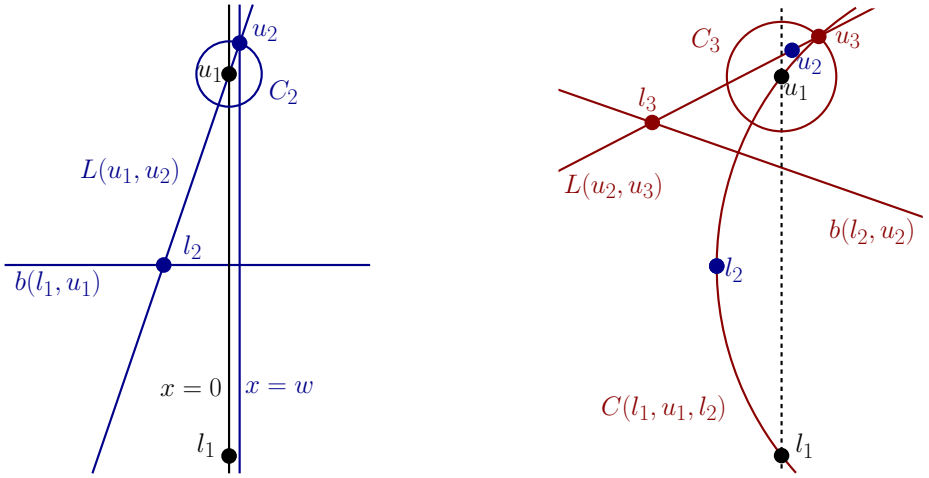
We already saw that linear separability does not guarantee the admissibility of a set of clusters. In the next section, we further investigate linear separability and show that it does not even guarantee a diagram of linear complexity.

3.4 A lower bound for linearly separable clusters

In this section we show that the farthest color Voronoi diagram may have quadratic complexity, even if the input set of clusters is linearly separable. To this aim, we define a set $\mathcal{P}$ of m linearly separable clusters of cardinality 2 such that the FCVD($\mathcal{P}$) contains $\Theta(m^2)$ mixed Voronoi vertices. For the rest of the section, $\mathcal{P}$ is a set of clusters defined as

$$\mathcal{P} := \{P_i := \{l_i, u_i\}, 1 \leqslant i \leqslant m\}.$$

We construct the set $\mathcal{P}$ as follows; refer also to Fig. 3.13. Let $l_1 = (0,0)$ and $u_1 = (0, 2^m)$. Let C_i, $2 \leqslant i \leqslant m$, be a set of concentric circles centered at u_1, each of radius 2^{-m+i-2}. Each *upper* point u_i is placed on circle C_i. Each *lower* point l_i



(a) Placement of P_1 (●) and P_2 (●). (b) Placement of P_3 (●).

Figure 3.13. Construction of the set $\mathcal{P}$. For P_i, with $i > 4$, the placement is analogous to P_3.

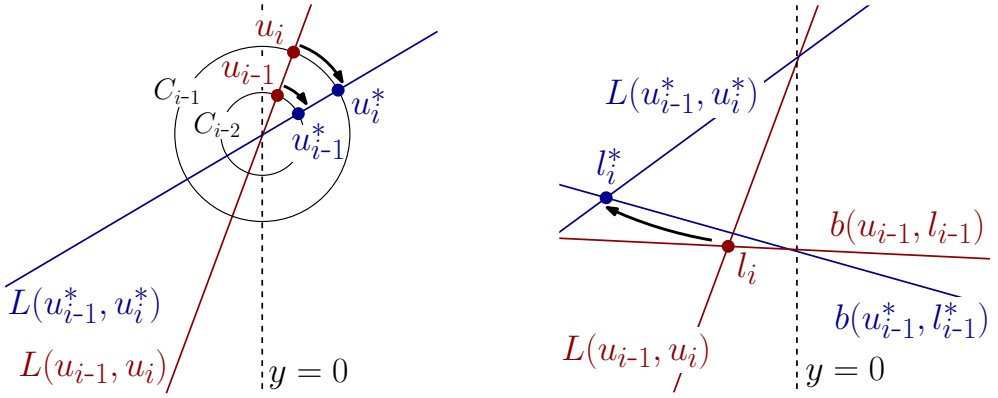
is placed on bisector $b(l_{i-1},u_{i-1})$. We control the placement of all points using a parameter w, with $0 \leqslant w \ll 2^{-m}$. In particular, point u_2 is placed at the upper intersection point of circle C_2 and the vertical line $x = w$. Each *lower* point l_i, for $i \geqslant 2$, is placed at the intersection point of line $L(u_{i-1},u_i)$ and bisector $b(l_{i-1},u_{i-1})$. Each *upper* point u_i, for $i \geqslant 3$, is placed on the upper intersection of circles C_i and $C(l_{i-2},u_{i-2},l_{i-1})$.

The construction of the set of clusters $\mathcal{P}$ is summarized as follows:

$$l_i = \begin{cases} (0,0), & \text{if } i = 1 \\ L(u_{i-1},u_i) \cap_{up} b(l_{i-1},u_{i-1}), & \text{if } i \geqslant 2 \end{cases} \qquad u_i = \begin{cases} (0,2^m), & \text{if } i = 1 \\ C_i \cap_{up} (x = w), & \text{if } i = 2 \\ C_i \cap_{up} C(l_{i-2},u_{i-2},l_{i-1}), & \text{if } i \geqslant 3 \end{cases}$$

Observe that quantity w controls the placement of all points except l_1, u_1. As we later show, for $w = 0$, all points lie on the y-axis. As w increases, lower points are translated up and left, while upper points are translated down and right. The effect of w on the points can be observed in Fig. 3.14. Quantity w needs to be sufficiently small, so that, for every $i < j$, point l_j lies within the disk DD_i whose diameter is defined by l_i, u_i; see the example in shown in Fig. 3.15. Refer to our Geogebra applet[3] for an interactive visualization of the set $\mathcal{P}$ and the effect of changing w.

[3]url: http://compgeom.inf.usi.ch/FCVD/lowerbound



(a) Shifting of upper points (b) Shifting of lower points

Figure 3.14. Illustration of how points are shifted for $w^* > w$.

We will prove the following regarding the complexity of the constructed FCVD$(\mathcal{P})$. The proof is given in Section 3.4.1.

Proposition 3.13. *The diagram of the constructed set FCVD$(\mathcal{P})$ has combinatorial complexity $\Theta(m^2) = \Theta(n^2)$.*

Combining the lower bound stemming from Proposition 3.13 with the trivial $\Omega(n)$ lower bound, we conclude the following, which is the main result of the section.

Theorem 3.3. *Given a linearly separable set of clusters $\mathcal{P}$, FCVD$(\mathcal{P})$ has $\Omega(n+m^2)$ combinatorial complexity in the worst case.*

3.4.1 Correctness of the lower bound construction

In this section we give all the statements needed to prove Proposition 3.13. Let $x(p)$, resp. $y(p)$, denote the x-coordinate, resp. y-coordinate, of a point p, and assume that a line $L(a,b)$ is oriented from a to b.

Lemma 3.14. *For $w = 0$, all points lie on the y-axis, with their y coordinates ordered as follows: $y(l_i) < y(l_j) < y(u_i) < y(u_j)$ for any $1 \leqslant i < j \leqslant m$.*

Proof. If $w = 0$, then $x(u_2) = 0$. Then, by simple induction, every line $L(u_{i-1}, u_i)$ and every circle $C(l_{i-2}, u_{i-2}, l_{i-1})$ coincides with the y-axis. Thus, all points lie on the y-axis. For any $i \geqslant 2$, the coordinates of the points are the following:

$$l_i = (0, 2^m - 2^{m-i+1} + 2^{i-2-m}/3 - 2^{2-i-m}/3) \tag{3.1}$$

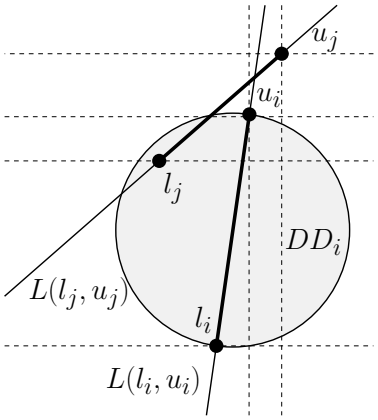


Figure 3.15. The relation between two clusters P_i and P_j, for $i > j$.

$$u_i = (0, 2^m + 2^{i-2-m}) \tag{3.2}$$

The claimed ordering of the y coordinates then immediately follows from Eqs. (3.1) and (3.2) concluding the proof. □

Using Eqs. (3.1) and (3.2) of Lemma 3.14 we can derive the following.

Corollary 3.15. *For $w = 0$, and for any $1 \leqslant i < j \leqslant m$, $d(l_i, l_j) = \Theta(2^{m-i+1})$ and $d(u_i, u_j) = \Theta(2^{j-m})$.*

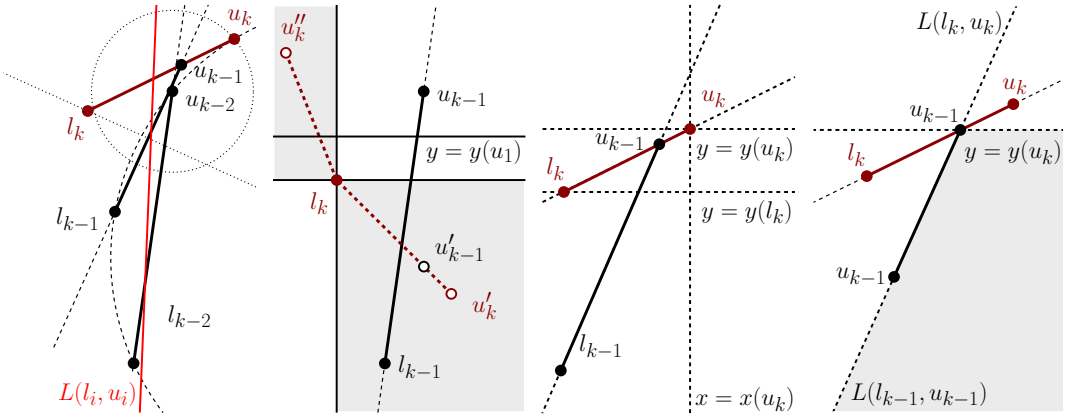
The above corollary implies that the distance between any two upper points is negligible compared to the distance between any two lower points. Further, this property is maintained for any valid quantity w, since the distance between a point and the respective point for $w = 0$, is larger by at most a constant factor.

The following lemma points out properties of the clusters in $\mathcal{P}$. Refer to Fig. 3.15 for an illustration of these properties.

Lemma 3.16. *Assuming that $l_j \in DD_i$, for any $i < j$, the following hold:*
(a) Point l_j is to the left of line $L(l_i, u_i)$ and point u_j is to its right.
(b) $0 < slope(L(l_j, u_j)) < slope(L(l_i, u_i))$.
(c) $y(l_i) < y(l_j) < y(u_i) < y(u_j)$ and $x(u_i) < x(u_j)$.
(d) Cluster P_i is to the right of line $L(l_j, u_j)$.

Proof. First note that $y(l_j) < y(u_1)$, i.e., all lower points lie below u_1. This is because $l_j \in \mathrm{DD}_1$ and the disk DD_1 lies below the horizontal line $y = x(u_1)$, since the segment $\overline{l_1 u_1}$ is vertical.

We prove the statements (a)-(d) by induction on j. Base case: for $j = 2$ the statements can be directly verified by the construction; see Fig. 3.13a. Inductive



(a) For property (a). (b) For property (b). (c) For property (c). (d) For property (d).

Figure 3.16. Illustrations for the proof of the properties of $\mathcal{P}$ of Lemma 3.16.

hypothesis: suppose that the statements hold for any $j < k$. We show that they also hold for $j = k$.

(a) Refer to Fig. 3.16a. By the inductive hypothesis l_{k-1} is to the left of $L(l_{k-2},u_{k-2})$ and $l_{k-1} \in \mathrm{DD}_{k-2}$, thus, $u_k = C_k \cap_{up} C(l_{k-2},u_{k-2},l_{k-1})$ lies to the right of $L(l_{k-1},u_{k-1})$. Point l_k is placed on $L(u_{k-1},u_k)$ below u_{k-1}, so l_k lies to the left of $L(l_{k-1},u_{k-1})$. By the inductive hypothesis, for any $i < k-1$, $slope(L(l_i,u_i)) > 0$ and $\overline{l_{k-1}u_{k-1}}$ intersects $L(l_i,u_i)$. Thus, $L(l_i,u_i)$ also intersects $\overline{l_ku_k}$, leaving l_k on the left side and u_k on the right side.

(b) From statement (a), it follows that $slope(L(l_k,u_k)) < slope(L(l_{k-1},u_{k-1}))$. Further, by the inductive hypothesis $slope(L(l_k,u_k)) < slope(L(l_i,u_i))\ \forall i < k-1$. It remains to show that $slope(L(l_k,u_k)) > 0$. We proved that l_k is to the left of $L(l_{k-1},u_{k-1})$ and that $y(l_k) < y(u_1)$. Suppose for contradiction, that $slope(L(l_k,u_k)) \leqslant 0$; refer also to Fig. 3.16b. Then u_k lies either in the *bottom right* quadrant defined by l_k (see u'_k in Fig. 3.16b), or in the *top left* quadrant (see u''_k in Fig. 3.16b) If it is in the bottom right quadrant, then by construction point u_{k-1} lies on $L(u_k,l_k)$, and so $y(u_{k-1}) < y(l_k) < y(u_1)$ violating statement (c). If it is in the top left quadrant, the u_k is to left of line $L(l_{k-1},u_{k-1})$ violating statement (a). In both cases we derive a contradiction.

(c) Refer to Fig. 3.16c. As discussed in the proof of (b), if $y(u_k) \leqslant y(u_{k-1})$, then this leads to a contradiction as it results in $slope(L(l_k,u_k)) \leqslant 0$. Thus, we have $y(u_k) > y(u_{k-1})$, and by the inductive hypothesis $y(u_k) > y(u_i)\ \forall i < k-1$. Since $slope(L(l_k,u_k)) < slope(L(l_{k-1},u_{k-1}))$ and u_k is to the right of $L(l_{k-1},u_{k-1})$, it follows that $x(u_k) > x(u_{k-1})$, and by the inductive hypothesis $x(u_k) > x(u_i)\ \forall i < k-1$. Point l_k is to the left of $L(l_{k-1},u_{k-1})$ and by the

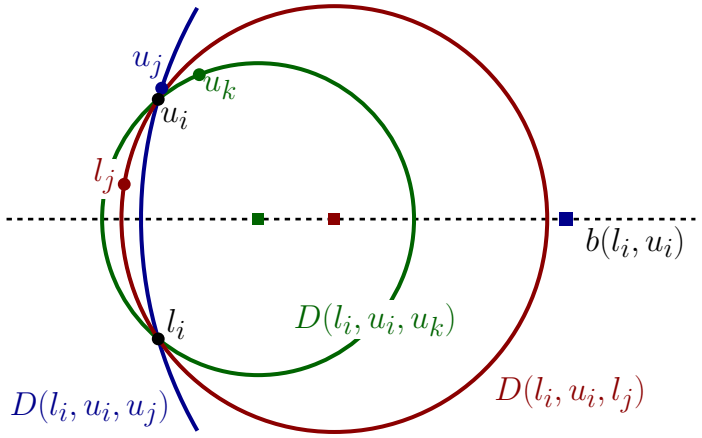


Figure 3.17. Cluster P_i straddled by clusters P_j, P_k, with $i < j < k$. The farthest color disks $D(l_i, u_i, u_j), D(l_i, u_i, l_j), D(l_i, u_i, u_k)$ are shown, and the corresponding centers (potential mixed vertices) along the bisector $b(l_i, u_i)$.

lemma assumption $l_k \in \mathrm{DD}_{k-1}$. Since $slope(L(l_{k-1}, u_{k-1})) > 0$, it follows that $y(l_k) > y(l_{k-1})$. By the inductive hypothesis we have $y(l_k) > y(l_i)\ \forall i < k-1$.

(d) By the placement of P_k it follows directly that P_{k-1} lies to the right of $L(l_k, u_k)$. By the inductive hypothesis, for any $i < k-1$, cluster P_i is to the right of $L(l_{k-1}, u_{k-1})$, and since $y(u_{k-1}) > y(u_i)$, it follows that P_i is also to the right of $L(l_k, u_k)$; see the shaded wedge in Fig. 3.16d. □

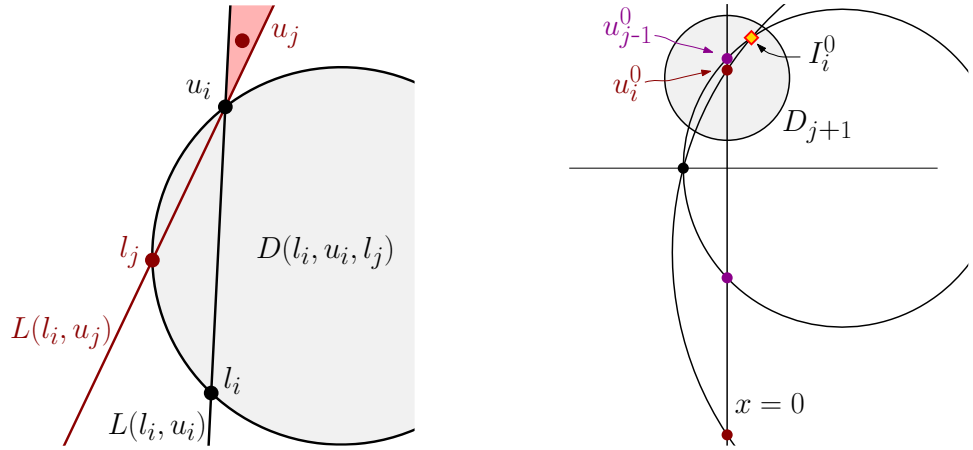
By Lemma 3.16(d), it follows that $\mathcal{P}$ is linearly separable, and by Lemma 3.16(a) it follows that cluster P_j straddles cluster P_i, for any $i < j$, since both clusters have only 2 points. So we can derive the following.

Corollary 3.17. *The constructed set $\mathcal{P}$ is linearly separable with $s(\mathcal{P}) = \Theta(m^2)$.*

Following, we show that for any m there exists a valid w to construct $\mathcal{P}$. Recall that quantity w is valid if point l_j lies within the disk DD_i, for any $i < j$.

Lemma 3.18. *For any m, there exists $w > 0$ such that $l_j \in DD_i$, $\forall i < j$.*

Proof. For $w = 0$, the lower point l_j lies in the interior of DD_i, for all $1 \leqslant i < j \leqslant m$, as it can be easily verified by the coordinates of the points given in Lemma 3.14. Suppose we slightly increase w to be infinitesimally positive. Then by the construction, the slope of $L(u_1, u_2)$ decreases infinitesimally from the y-axis, displacing point l_2 infinitesimally to its left. In turn, the entire construction rotates infinitesimally clockwise, with $slope(L(u_i, u_{i+1})) < slope(L(u_{i-1}, u_i))$ decreasing infinitesimally over the y-axis. Since the displacement of all points remains infinitesimally close to the y-axis, while $d(l_i, u_i)$ remains almost intact and



(a) Point u_j lies in the highlighted wedge, so $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, l_j))$.

(b) Simplified setting showing that the intersection I^0_i (highlighted) lies in D_{j+1}.

Figure 3.18. Illustrations for the proof of Lemma 3.19.

being much larger than the maximum point displacement, l_j must continue to lie in the interior of DD_i for all $i < j$, by continuity. Thus, for small enough w, point $l_j \in DD_i$, for all $i < j$. □

From now on we assume that w is sufficiently small, so that $l_j \in \mathrm{DD}_i \ \forall i < j$. In the sequel we focus on a bisector $b(l_i, u_i)$ of a cluster P_i and point out the possible ordering of mixed vertices along $b(l_i, u_i)$; see an example in Fig. 3.17. Each mixed vertex corresponds to the center of a disk through l_i, u_i, and the disks are ordered according to their radii as follows.

Lemma 3.19. *For any $i < j < k$, $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, l_j)) \geqslant \mathrm{r}(D(l_i, u_i, u_k))$.*

Proof. First we show that $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, l_j))$, refer to Fig. 3.18a. Consider the disk $D(l_i, u_i, l_j)$. By Lemma 3.16(a), point u_j lies to the right of $L(l_i, u_i)$. By Lemma 3.16(d), segments $\overline{l_i u_i}$ and $\overline{l_j u_j}$ do not intersect, thus, u_j lies to the left of $L(l_j, u_i)$. So, u_j lies in the upper wedge defined by $L(l_j, u_i)$ and $L(l_i, u_i)$; see the shaded wedge in Fig. 3.18a. This wedge is outside of $D(l_i, u_i, l_j)$, therefore, $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, l_j))$.

To show the second inequality, we first show that $\mathrm{r}(D(l_i, u_i, l_j)) \geqslant \mathrm{r}(D(l_i, u_i, u_{j+1}))$ (the equality trivially holds only for $j = i + 1$). By the configuration of the clusters, as proved in Lemma 3.16, $\mathrm{r}(D(l_i, u_i, l_j)) > \mathrm{r}(D(l_i, u_i, u_{j+1}))$ is equivalent to $u_{j+1} \in D(l_i, u_i, l_j)$. To prove that $u_{j+1} \in D(l_i, u_i, l_j)$, let $I_i := C(l_i, u_i, l_j) \cap_{up} C(l_{j-1}, u_{j-1}, l_j)$ be the upper intersection point of the two circles. Recall that

$u_{j+1} = C_{j+1} \cap_{up} C(l_{j-1}, u_{j-1}, l_j)$, hence, it suffices to show that $I_i \in D_{j+1}$, where D_{j+1} is the disk of circle C_{j+1}.

Refer to Fig. 3.18b. Let us first prove a simplified version of this statement using points u_i^0, l_i^0, for $w = 0$. Let $I_i^0 := C(l_i^0, u_i^0, l_j) \cap_{up} C(l_{j-1}^0, u_{j-1}^0, l_j)$. We consider all points l_i, u_i, with $i < j$, as being collinear, while l_j is not. We allow l_j to be translated on $b(l_{j-1}^0, u_{j-1}^0)$ to the left by a small amount t as long as l_j remains in DD_i. We can prove that $I_i^0 \in D_{j+1}$ by solving the respective system of equations using the *Wolfram Mathematica* software[4].

In particular, we show that if $u_1 = (0, 2^m)$, $u_i^0 = (0, 2^m + 2^{i-2-m})$, $u_{j-1}^0 = (0, 2^m + 2^{j-3-m})$, $l_i^0 = (0, 2^m - 2^{m-i+1} + 2^{i-2-m}/3 - 2^{2-i-m}/3)$, $l_{j-1}^0 = (0, 2^m - 2^{m-j+2} + 2^{j-3-m}/3 - 2^{3-j-m}/3)$ and $l_j = (-t, 2^m - 2^{m-j+1} + 2^{j-2-m}/3 - 2^{2-j-m}/3)$, for $0 < t < d(l_{j-1}^0, u_{j-1}^0)/2$. Then, $I_i^0 \in D_{j+1}$, for any $3 \leqslant i \leqslant j-2 \leqslant m-3$.

The intuition for studying I_i^0, comes from the fact that as w goes to 0, all points continuously move towards the y-axis, until when $w = 0$, and all points are on the y-axis. So, we investigate a construction of this form (having sufficiently small w), and by treating points accordingly. On the contrary, since we need to compare circles, we do not want them to degenerate to lines, so we allow l_j to be translated on $b(l_{j-1}, u_{j-1})$ to the left of the y-axis.

We now go back from the simplified statement, i.e., $I_i^0 \in D_{j+1}$, to the one we want to prove, i.e., $I_i \in D_{j+1}$. Consider the points l_i, u_i and I_i for some $w > 0$ and let l_i^*, u_i^*, I_i^* denote the respective points for some quantity $w^* > w$. We will show that $d(u_1, I_i^*) < d(u_1, I_i)$. As described earlier, by increasing w, the set of clusters $\mathcal{P}$ rotates clockwise, and $slope(L(l_i^*, u_i^*)) < slope(L(l_i, u_i))$ (see also Fig. 3.14). Consider now the distance $d(l_i, u_i)$; we argue that the minimum of $d(l_i, u_i)$, over all valid values of w, is realized when $w = 0$, i.e., when $l_i = l_i^0, u_i = u_i^0$ and both l_i, u_i lie on the y-axis. Then by the continuity of the construction $d(l_i^*, u_i^*) > d(l_i, u_i)$ holds. To see that $\min_w d(l_i, u_i) = d(l_i^0, u_i^0)$, consider the distance between points discussed in Corollary 3.15; as w increases the displacement of the points from the y-axis increases, and hence $d(l_i, u_i) > d(l_i^0, u_i^0)$. Using the same arguments we can show that $d(l_j^*, L(l_i^*, u_i^*)) > d(l_j, L(l_i, u_i))$ holds.

Refer to Fig. 3.19. Since $d(l_j^*, L(l_i^*, u_i^*)) > d(l_j, L(l_i, u_i))$, the two disks defining point I_i^* have smaller radii as opposed to the disks defining I_i. Combining this property with the aforementioned (clockwise) rotation of the complete set of clusters, it follows that the intersection point l_i^* is rotated around u_1 with $d(u_1, I_i^*) < d(u_1, I_i)$ (observe the trajectories of the points I_i in Fig. 3.19).

Since $d(u_1, I_i^*) < d(u_1, I_i)$, for $w^* > w$, then the upper bound of $d(u_1, I_i)$, over all valid w, is realized when $w \to 0$. This coincides with the value of $d(u_1, I_i^0)$

[4]url: *https://www.wolfram.com/mathematica*

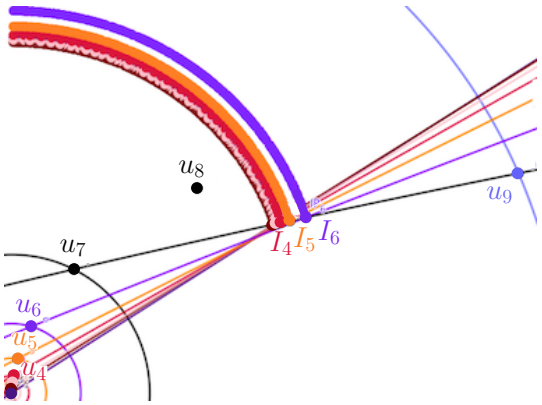


Figure 3.19. An instance drawn with our Geogebra applet, with $m = 9, j = 8$. The black circle is $C(l_{j-1}, u_{j-1}, l_j)$, i.e., $C(l_7, u_7, l_8)$, and the thin colored circles are $C(l_i, u_i, l_{j-1})$, for $i = 1 \ldots j-2$, i.e., $C(l_i, u_i, l_7)$, for $i = 1 \ldots 6$. The thick curves show the trajectory of points I_i, as w increases (starting at $w = 0$). For an increased $w^* > w$, point I_i is rotated clockwise, moving slightly closer to u_1.

when the translation of l_j along $b(l^0_{j-1}, u^0_{j-1})$ is infinitesimally small ($t \to 0$). Thus, $d(u_1, I_i) < d(u_1, I^0_i)$ for a small enough t, and since we already proved that $I^0_i \in D_{j+1}$, it follows that $I_i \in D_{j+1}$, and so $u_i \in D_{j+1}$.

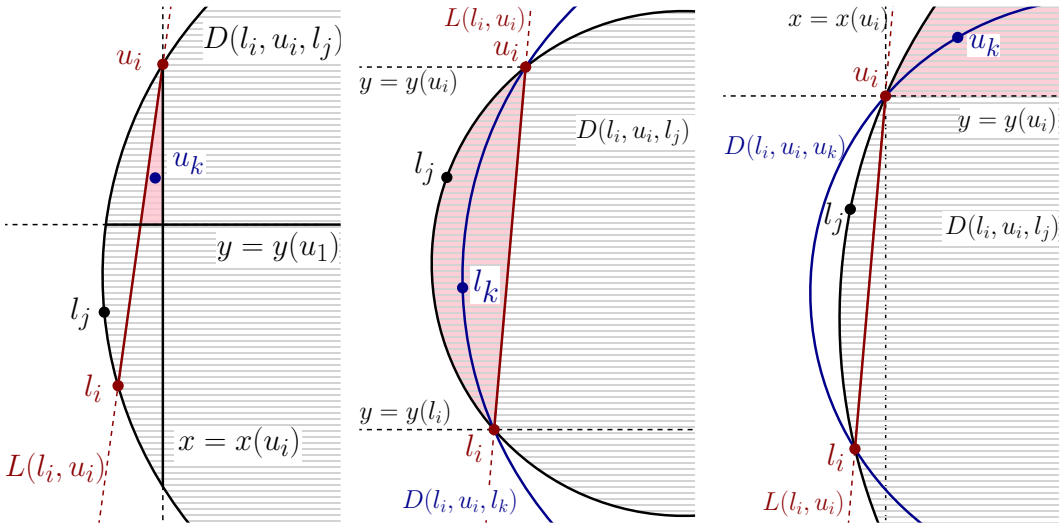
Since $\mathrm{r}(D(l_i, u_i, l_j)) > \mathrm{r}(D(l_i, u_i, u_{j+1}))$ and $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, l_j))$, it follows that $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, u_{j+1}))$, and so thus, $\mathrm{r}(D(l_i, u_i, u_j)) > \mathrm{r}(D(l_i, u_i, u_k))$, for any $k > j$. Combining this with $\mathrm{r}(D(l_i, u_i, l_j)) > \mathrm{r}(D(l_i, u_i, u_{j+1}))$, we get $\mathrm{r}(D(l_i, u_i, l_j)) > \mathrm{r}(D(l_i, u_i, u_k))$, concluding the second part of the proof. □

Following, we show that all the disks induced by triplets of points, as described before, are farthest color disks and induce vertices in FCVD($\mathcal{P}$).

Lemma 3.20. *Disks $D(l_i, u_i, l_j)$ and $D(l_i, u_i, u_j)$ are farthest color disks, $\forall i < j$, i.e., they contain one point of every cluster P_k, $k \neq i, j$. In particular:*

(a) if $k < i < j$ then $u_k \in D(l_i, u_i, l_j)$ and $u_k \in D(l_i, u_i, u_j)$ (see Fig. 3.20a);
(b) if $i < k < j$ then $l_k \in D(l_i, u_i, l_j)$ and $l_k \in D(l_i, u_i, u_j)$ (see Fig. 3.20b);
(c) if $i < j < k$ then $u_k \in D(l_i, u_i, l_j)$ and $u_k \in D(l_i, u_i, u_j)$ (see Fig. 3.20c);

Proof. (a) Suppose $k < i < j$; refer to Fig. 5.8a. Since $i > k$, by Lemma 3.16(c), $x(u_i) > x(u_k)$, thus, u_k is to the left of the vertical line $x = x(u_i)$. Also, u_k is to the right of line $L(l_i, u_i)$, due to Lemma 3.16(a). So, u_k lies in the lower wedge defined by $L(l_i, u_i)$ and $x = x(u_i)$. This wedge intersects $D(l_i, u_i, l_j)$, because both $L(l_i, u_i)$ and $x = x(u_i)$ define chords on the disk and the apex of the wedge



(a) Case (a): $k < i < j$. (b) Case (b): $i < k < j$. (c) Case (c): $i < j < k$.

Figure 3.20. Illustrations for the proof of Lemma 3.20.

u_i lies on $C(l_i, u_i, l_j)$. Moreover, $y(u_k) > y(u_1)$, by Lemma 3.16(c), so u_k lies above the horizontal line $y = y(u_1)$, which also defines a chord on $D(l_i, u_i, l_j)$. So the triangle defined by $L(l_i, u_i), y = y(u_1), x = x(u_i)$ lies entirely in $D(l_i, u_i, l_j)$ (see the red shaded region in Fig. 5.8a). Point u_k lies in this triangle, thus, $u_k \in D(l_i, u_i, l_j)$. Using the same arguments, we can show that this triangle is a also a subset of $D(l_i, u_i, u_j)$, thus $u_k \in D(l_i, u_i, u_j)$.

(b) Suppose $i < k < j$; refer to Fig. 5.8b. Since $i < k < j$, we know by Lemma 3.16(c) that $y(l_i) < y(l_k) < y(l_j) < y(u_i)$, and thus, both points l_k, l_j lie in the horizontal strip defined by $y = y(l_i)$ and $y = y(u_i)$. Moreover, both l_j, l_k are to the left of line $L(l_i, u_i)$, by Lemma 3.16(a). Since $k < j$, by Lemma 3.19, the disks have radii: $\mathrm{r}(D(l_i, u_i, l_k)) > \mathrm{r}(D(l_i, u_i, l_j))$. So, $D(l_i, u_i, l_k) \subset D(l_i, u_i, l_j)$ in the halfplane to the left of the oriented line $L(l_i, u_i)$ (see the red shaded region in Fig. 5.8b). Therefore, the part of $D(l_i, u_i, l_k)$ to the left of $L(l_i, u_i)$ is contained in $D(l_i, u_i, l_j)$. Further, since that part of $D(l_i, u_i, l_k)$ is contained in the horizontal strip defined by $y = y(l_i)$ and $y = y(u_i)$, it also contains l_k. Thus, $l_k \in D(l_i, u_i, l_j)$. From Lemma 3.19, the disks have radii: $\mathrm{r}(D(l_i, u_i, l_k)) > \mathrm{r}(D(l_i, u_i, u_j))$, so with the same arguments we can show that $l_k \in D(l_i, u_i, u_j)$.

(c) Suppose $i < j < k$; refer to Fig. 5.8c. Since $k > i$, then $y(u_k) > y(u_i)$, due to Lemma 3.16(c), and thus u_k is above the horizontal line $y = y(u_i)$. Moreover, u_k is to the right side of the line $L(l_i, u_i)$, by Lemma 3.16(a). Thus, u_k lies in the upper wedge defined by $L(l_i, u_i)$ and $y = y(u_i)$. Also $k > j$, so, due to

Lemma 3.19, the disks have radii: $\mathrm{r}(D(l_i,u_i,l_j)) > \mathrm{r}(D(l_i,u_i,u_k))$. So, the part of $D(l_i,u_i,u_k)$ to the right of $L(l_i,u_i)$ is contained in $D(l_i,u_i,l_j)$. Therefore, this also holds for the part of $D(l_i,u_i,u_k)$ in the upper right wedge defined by $L(l_i,u_i)$ and $y = y(u_i)$, where u_k lies (see the red shaded region in Fig. 5.8c). Thus, $u_k \in D(l_i,u_i,l_j)$. Due to Lemma 3.19, $\mathrm{r}(D(l_i,u_i,u_j)) > \mathrm{r}(D(l_i,u_i,u_k))$ holds. Similarly, we can infer that $u_k \in D(l_i,u_i,u_j)$. □

Summarizing, regarding the constructed set of clusters $\mathcal{P}$, we can prove (and re-state) the following regarding the complexity of FCVD($\mathcal{P}$).

Proposition 3.13. *The diagram of the constructed set FCVD($\mathcal{P}$) has combinatorial complexity $\Theta(m^2) = \Theta(n^2)$*

Proof. By Lemma 3.20, $D(l_i,u_i,l_j)$ and $D(l_i,u_i,u_j)$ are farthest color disks for any $i < j$, each inducing a mixed vertex in FCVD($\mathcal{P}$). More specifically, the bisector of cluster P_i is incident to $2(m-i)$ mixed vertices. These vertices appear in consecutive pairs (as in Fig. 3.9) and each pair of them delimits a bounded face as proved in Proposition 3.9. Overall, there are $\Theta(m^2)$ mixed vertices, as well as bounded faces, and the claim follows. □

As we mentioned earlier, the above result, combined with the $\Omega(n)$ lower bound, yields the $\Omega(n+m^2)$ lower bound in the worst-case combinatorial complexity of the diagram of linearly separable families (stated in Theorem 3.3).

As a final note, observe that as defined, the constructed set of clusters $\mathcal{P}$ does not satisfy the general position assumption, as every four points $(l_{i-2},u_{i-2},l_{i-1},u_i)$ are cocircular and every three points (l_i,u_i,u_{i-1}) are collinear. However, if desired, general position can be easily enforced by infinitesimally translating the points during the construction as follows. Point u_i, for $i \geqslant 3$, can be translated on circle C_i towards the interior of $C(l_{i-2},u_{i-2},l_{i-1})$, and point l_i, for $i \geqslant 2$, can be moved along bisector $b(l_{i-1},u_{i-1})$ towards the y-axis.

3.5 Construction algorithms

We design an algorithm using the standard divide & conquer paradigm. First we split $\mathcal{P}$ into two sets $\mathcal{P}_A$ and $\mathcal{P}_B$ of roughly equal size as follows: if there exists P, with $|P| \geqslant n/2$, then $\mathcal{P}_A = \{P\}$ and $\mathcal{P}_B = \mathcal{P}\backslash\{P\}$; else, we set $\mathcal{P}_A = \{P_1,\ldots,P_k\}$ and $\mathcal{P}_B = \{P_{k+1},\ldots,P_m\}$ such that $|\mathcal{P}_A^*| = \Theta(|\mathcal{P}_B^*|) = \Theta(n)$. Then, we recursively compute FCVD($\mathcal{P}_A$) and FCVD($\mathcal{P}_B$), and finally, we merge them to obtain FCVD($\mathcal{P}$).

To merge FCVD($\mathcal{P}_A$) and FCVD($\mathcal{P}_B$) we need to construct the *merge curve*, which is the set of color edges in FCVD($\mathcal{P}_A \cup \mathcal{P}_B$) belonging to color bisectors $b_c(P,Q)$ with $P \in \mathcal{P}_A$ and $Q \in \mathcal{P}_B$. The merge curve may consist of linearly many connected components, either unbounded and bounded. To construct it, a *starting point* is first identified on each component, and then, the component gets *traced* through FCVD$_a(\mathcal{P}_A)$ and FCVD$_a(\mathcal{P}_B)$.

We first discuss how each step is performed by our algorithm and then we describe the algorithmic results which we obtain.

A. Tracing a component of the merge curve given a starting point. To trace a component efficiently, in time linear in its size, we can adapt standard techniques exploiting the so-called *visibility decomposition*, which has been defined for the Hausdorff Voronoi diagram by Papadopoulou and Lee [2004].

The *visibility decomposition* further refines the FCVD$_a(\mathcal{P})$. For each region $f_c reg(p,\mathcal{P})$, and for each color or mixed vertex u on $\partial f_c reg(p,\mathcal{P})$, we partition the region by $L(p,u) \cap f_c reg(p,\mathcal{P})$ (see Fig. 3.3). The visibility property of Proposition 3.2, guarantees that the intersection $L(p,u) \cap f_c reg(p,\mathcal{P})$ is connected.

Given a ray (corresponding to an edge of the merge curve) with which we are tracing, we compare its intersection points with the two diagrams FCVD($\mathcal{P}_A$) and FCVD($\mathcal{P}_B$), and keep the one which is closer and discard the other one. When looking for these intersection points though, it is important to move simultaneously on the faces of the visibility decomposition of both diagrams, so that we can stop when the first intersection point is reached. In this way, we avoid spending $O(n)$ time to find a single intersection point that may be discarded (recall that a single face of FCVD$_a(\mathcal{P})$ can have $\Theta(n)$ complexity). Secondly, using the visibility property it follows that (analogously to Papadopoulou and Lee [2004]), the merge curve does not intersect an edge of the visibility decomposition more than once. Hence, at each step, no face of the visibility decomposition is visited more than twice and so, tracing takes linear time in the size of the merge curve.

B1. Finding a starting point on an unbounded component of the merge curve. To identify such starting points we use the cluster hull, similarly to how it is used in the Hausdorrff Voronoi diagram by Papadopoulou and Lee [2004]. This is possible due to the one-to-one correspondence between the hull edges and the unbounded edges of FCVD$_a(\mathcal{P})$, proved in Lemma 3.4. More precisely, during the algorithm, together with the diagrams FCVD($\mathcal{P}_A$) and FCVD($\mathcal{P}_B$), we keep their hulls, *CLH*($\mathcal{P}_A$) and *CLH*($\mathcal{P}_B$), and prior to merging the diagrams we merge the hulls. Merging *CLH*($\mathcal{P}_A$) and *CLH*($\mathcal{P}_B$) yields the lines which correspond to

the unbounded edges of the diagram FCVD($\mathcal{P}_A \cap \mathcal{P}_B$). This can be done in time $O(|CLH(\mathcal{P}_A)| + |CLH(\mathcal{P}_B|)$, as shown by Papadopoulou and Lee [2004].

B2. Finding a starting point on an bounded component of the merge curve. Each bounded component of the merge curve encloses a portion of an internal skeleton, as shown in Proposition 3.3. Hence, to identify starting points, we can search for vertices and edges of the internal skeletons that may be enclosed in such components.

For each internal vertex u of FCVD$_a(\mathcal{P}_A)$, with $u \in f_c reg(P, \mathcal{P}_A)$, we point-locate u in FCVD$_a(\mathcal{P}_B)$ to find Q, for which $u \in f_c reg(Q, \mathcal{P}_B)$. Then, we compare the distances $d_c(u, P)$ and $d_c(u, Q)$. If $d_c(u, P) \leqslant d_c(u, Q)$, we start tracing the component from u, else if $d_c(u, P) > d_c(u, Q)$, we discard u. This is repeated for the internal vertices of FCVD$_a(\mathcal{P}_B)$. There are $O(n)$ internal vertices in total, and point location of a single vertex can be done in $O(\log n)$ time using standard methods, see e.g., Kirkpatrick [1983], resulting in $O(n \log n)$ overall time at each step.

However, not every bounded component of the merge curve needs to contain an internal vertex, so we also need to search for internal edges that can have portion(s) enclosed in some component. Since by Proposition 3.3, the internal skeleton of a bounded face is a tree, it follows that if no internal vertex appears, then the skeleton is a single edge. Our approach is to search each internal edge in order to find the portion(s) which appear in the merged diagram. We use the data structure of Iacono et al. [2017], which allows for efficient searches of intersections between two plane graphs, in an analogous way as it was used for the Hausdorff Voronoi diagram.

In more detail, for each internal edge e of FCVD$_a(\mathcal{P}_A)$, with $e \in f_c reg(P, \mathcal{P}_A)$, we construct a binary search tree which implicitly stores the intersections of e with FCVD$_a(\mathcal{P}_B)$. Every node x of the tree corresponds to an intersection point of an edge $uv \subseteq e$ (portion of e) with FCVD$_a(\mathcal{P}_B)$. The left child of x contains the portion ux, and the right child contains the portion xv. To use the data structure of Iacono et al. [2017], it is necessary to define a rule of how to navigate the tree, meaning to which child we should move when we are at a node x. We use the following rule.

Suppose that x is an intersection point of $uv \subseteq e$ with an internal edge of FCVD$_a(\mathcal{P}_B)$ and suppose $x \in f_c reg(Q, \mathcal{P}_B)$. We compare the distances $d_c(x, Q)$ and $d_c(x, P)$. (i) If $d_c(x, Q) > d_c(x, P)$, we consider the two portions ux and xv separately. For ux (and analogously for xv) we compare the distances at the endpoint u, i.e., $d_c(u, P)$ and $d_c(u, Q)$. If $d_c(u, P) \geqslant d_c(u, Q)$, we search the

subtree of ux as there exists some $y \in ux$ with $y \in f_c reg(P, \mathcal{P}_A \cup \mathcal{P}_B)$. Else if $d_c(u,Q) > d_c(u,P)$ we do not search further the subtree of ux, as there exists no such point y. (ii) If $d_c(x,P) \geqslant d_c(x,Q)$, we start tracing the bounded component from point x. Assume the $x_l x$ and $x x_r$ were the portions of edge uv contained in the bounded component that was traced. Then, we continue searching in both subtrees of ux_l and $x_r v$. Finally, suppose that x is an intersection point of $uv \subseteq e$ with a color edge of $\text{FCVD}_a(\mathcal{P}_B)$. The difference is that x is equidistant to two clusters $Q_1, Q_2 \in \mathcal{P}_B$, i.e., $d_c(x,Q_1) = d_c(x,Q_2)$, so the navigation rules remain the same, but at each comparison, the distance to both clusters Q_1, Q_2 has to be considered.

This is repeated for the internal edges of $\text{FCVD}_a(\mathcal{P}_B)$. Each point location query takes $O(\log n)$ time, and since the search trees are balanced, having a depth of $O(\log n)$, a single traversal from the root to a leaf takes $O(\log^2 n)$ time.

Algorithmic results. Putting all the above together, we can derive the following algorithmic result for arbitrary sets of input clusters.

Theorem 3.4. *FCVD$(\mathcal{P})$ can be constructed in $O((n + s(\mathcal{P})) \log^3 n)$ time.*

Proof. Consider the algorithm previously described. At each recursive step, starting points on unbounded components are found in $O(n)$ time and given a starting point a component is traced in linear time. It remains to identify how much time is overall needed to find starting points on bounded components.

For each internal edge e not entirely contained in the traced components, we build a search tree. The set of all search trees, for all $O(n)$ edges, can be constructed in $O(n \log n)$ time (see Iacono et al. [2017]). Identifying a portion of e requires a traversal of the search tree which takes $O(\log^2 n)$ time. When traversing the tree, at some nodes the search might continue to both children, as portions of e may appear in more than one components, but this has to be a result of a straddle, as follows from Lemma 3.8. Moreover, a single straddle occurs in at most one recursive step, since a pair of clusters may be considered at most once in two sets $\mathcal{P}_A, \mathcal{P}_B$. Hence, an edge e might be considered at most $s(e) = O(m)$ times, over all steps. This bounds the portions to be identified and the time all search trees are traversed, resulting in $O((n + s(\mathcal{P})) \log^3 n)$ total time. □

The time complexity of our algorithm depends on the parameter $s(\mathcal{P})$. Although $s(\mathcal{P})$ is in the worst case $\Theta(mn)$ for arbitrary clusters, and $\Omega(n + m^2)$ for linearly separable clusters, this number could be small in practice. In fact, we saw that linearly separable clusters, for which $s(\mathcal{P}) = \Theta(m^2)$ are quite degener-

ate. When the straddling number is small, i.e., $s(\mathcal{P}) = O(n)$, our algorithm has time complexity $O(n \log^3 n)$ outperforming the existing algorithms.

For admissible sets of clusters, we can derive an improved time complexity as follows.

Theorem 3.5. *Given an admissible set of clusters $\mathcal{P}$, FCVD$(\mathcal{P})$ can be constructed in $O(n \log n)$ time.*

Proof. Consider the described divide and conquer scheme. By Proposition 3.10, FCVD$(\mathcal{P})$ is a tree, hence, the merge curve has only unbounded components. Starting points on the unbounded components are identified in $O(n)$ time, and the remaining tracing of the merge curve takes also $O(n)$ time. So, each recursive step takes $O(n)$ time, and this results in an $O(n \log n)$-time algorithm overall. □

Note that for an admissible set of clusters, we can construct FCVD$(\mathcal{P})$ in randomized $O(bn \log n)$ time, where b is the time to obtain a color bisector, following the construction of the farthest abstract Voronoi diagram by Mehlhorn et al. [2001]. However, b can be $\Theta(n)$ for color bisectors, thus, a direct application of this algorithm would result in $O(n^2 \log n)$ time.

3.6 Conclusion

In this chapter we presented our work related to color Voronoi diagrams, and more specifically to the farthest color Voronoi diagram. Our goal was to get a better understanding of the structural properties of the diagram, and to see how we can use them to obtain better combinatorial and algorithmic results. The main results we obtained can be summarized as follows.

Regarding the combinatorial properties of the diagram, we studied the structure and number of the bounded and unbounded faces. We identified the straddling parameter $s(\mathcal{P})$, as the parameter which is responsible for the creation of bounded faces, which lead to increased complexity. Using this we refined the upper bound on the complexity of the diagram to $O(n+s(\mathcal{P}))$, where $s(\mathcal{P}) = O(mn)$.

Further, we looked for necessary or sufficient conditions for the diagram to have $O(n)$ complexity, focusing particularly on the connection to the abstract Voronoi diagrams framework. Looking for conditions for $O(n)$ complexity, we also studied linearly separable clusters and we proved that linear separability is not such a condition. In fact, we showed that linearly separable clusters can realize a diagram with quadratic $\Omega(n + m^2)$ worst case complexity.

Finally, we also considered construction algorithms. We designed a divide & conquer algorithm, which has time complexity $O(n+s(\mathcal{P}))$, for arbitrary clusters;

this is quite nice, as $s(\mathcal{P})$ is expected to be small $O(n)$ in practice. For special cases of input clusters, as for example admissible clusters, we showed that the algorithm has optimal $O(n \log n)$ time complexity.

While studying color Voronoi diagrams, many interesting questions have come up. We list some open questions and discuss future directions related to color Voronoi diagrams in Section 6.1.

As a final note, in this chapter, we did not discuss any implementation issues regarding the construction of the farthest color Voronoi diagram. A useful tool in the research conducted was the implementation of Panos Cheilaris and Elena Arseneva, which was also used to create some of the figures. This implementation uses the *CGAL* library[5], following the *exact computing paradigm*, and it is based on an algorithmic approach similar to Setter et al. [2010].

[5]url: *https://www.cgal.org/*

Chapter 4

Rotating rays Voronoi diagram

This chapter presents our results on the rotating rays Voronoi diagram and the Brocard illumination problem. It is based on the following publication:

> *C. Alegría, I. Mantas, E. Papadopoulou, M. Savić, H. Schrezenmaier, C. Seara, and M. Suderland.* The Voronoi Diagram of Rotating Rays With applications to Floodlight Illumination. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

In Section 4.1 we give the definitions and the basic concepts related to the diagram. In Section 4.2 we study the diagram in the entire plane. In Section 4.3 we consider the diagram where the underlying domain is a convex polygon bounded by the input rays, and in Section 4.4 we restrict the domain of interest to be some curve. In each of the three previous sections, we describe combinatorial and algorithmic results and we also give solutions to the Brocard illumination problem. Section 4.5 concludes the chapter.

4.1 Preliminaries

Let the input be a set $\mathcal{R}$ of n rays in the plane. Given a ray r, we denote its apex by $p(r)$, its supporting line by $l(r)$, and its direction in the unit circle S^1 by $\widehat{d}(r)$. For three points $A, B, C \in \mathbb{R}^2$, we denote by $\angle(A,B,C)$ the counterclockwise angle at point B between the rays BA and BC.

Recall that we defined the (oriented) angular distance (see Definition 1.7) from a point x to a ray r, as the minimum counterclockwise angle α from r to a ray with apex $p(r)$ passing through x. We denote the angular distance by $d_\angle(x,r)$, and moreover, we define $d_\angle(p(r),r) = 0$.

It is easy to see that the angular distance is not a metric. Further, observe that the range of $d_\angle(x,r)$ is $[0,2\pi)$ and that there is a discontinuity at 2π.

Angular bisectors. Before examining the properties of the rotating rays Voronoi diagram, we first need to examine the bisectors between two sites. Using the oriented angular distance, we define the bisector of two rays as follows.

Definition 4.1. Given two rays r and s, the *dominance region* of r over s, denoted by $\mathrm{dr}(r,s)$, is the locus of points with smaller angular distance to r than to s, i.e.,

$$\mathrm{dr}(r,s) := \{\, x \in \mathbb{R}^2 \mid d_\angle(x,r) < d_\angle(x,s) \,\}.$$

The *angular bisector* of r and s, denoted by $b_\angle(r,s)$, is the curve delimiting $\mathrm{dr}(r,s)$ and $\mathrm{dr}(s,r)$.

Different instances of angular bisectors are illustrated in Fig. 4.1. Given two rays r and s, let $I := l(r) \cap l(s)$. The angular bisector $b_\angle(r,s)$ is the union of the two rays r and s, and a circular arc a that connects $p(r)$ to $p(s)$. The arc a belongs to the *bisecting circle* $C_b(r,s)$, which we define as follows:

- If I, $p(r)$, and $p(s)$ are pairwise different, then $C_b(r,s)$ is the circle through I, $p(r)$, and $p(s)$. The arc a contains I if, and only if, I lies either on none or on both r and s; see Fig. 4.1a, Fig. 4.1b and Fig. 4.1c.

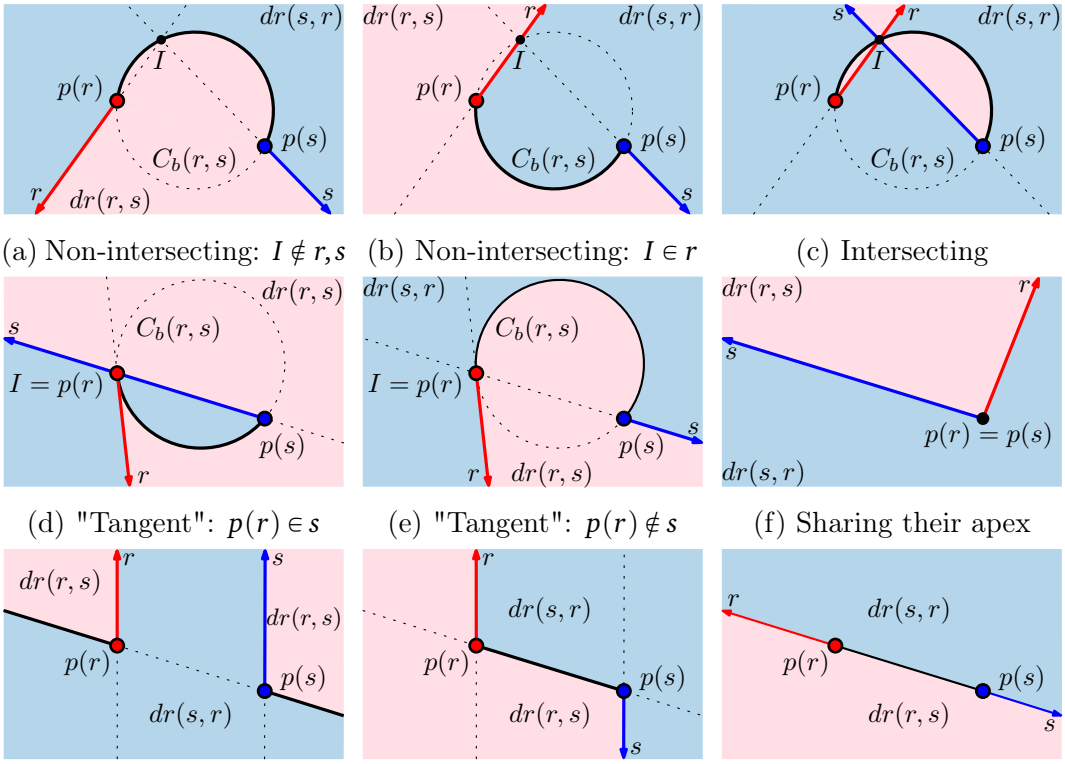
- If $I = p(r)$ and $I \neq p(s)$, then $C_b(r,s)$ is the circle tangent to $l(r)$ passing through $p(r)$ and $p(s)$. Both a and r lie on the same side of $l(s)$ if, and only if, $p(r)$ lies on s; see Fig. 4.1d and Fig. 4.1e. We analogously define $C_b(r,s)$ if $I = p(s)$ and $I \neq p(r)$.

- If $p(r) = p(s)$, then both $C_b(r,s)$ and a degenerate to a single point; see Fig. 4.1f.

- If $l(r)$ and $l(s)$ are parallel, then $C_b(r,s)$ degenerates to the line through $p(r)$ and $p(s)$. If $\widehat{d}(r) = \widehat{d}(s)$, then a consists of two halflines; see Fig. 4.1g. If instead $\widehat{d}(r) = -\widehat{d}(s)$, then a degenerates to a line segment; see Fig. 4.1h and Fig. 4.1i.

Note that our definition of a bisector is slightly different than the usual, which is the locus of points equidistant to two sites. This is due to the discontinuity of the distance function at 2π. In the following lemma we justify the above description of the bisectors.

Lemma 4.1. *Let r and s be two non-(anti)-parallel rays. The bisector $b_\angle(r,s)$ consists of the two rays and a subset of the bisecting circle $C_b(r,s)$, as described above.*



(a) Non-intersecting: $I \notin r,s$ (b) Non-intersecting: $I \in r$ (c) Intersecting

(d) "Tangent": $p(r) \in s$ (e) "Tangent": $p(r) \notin s$ (f) Sharing their apex

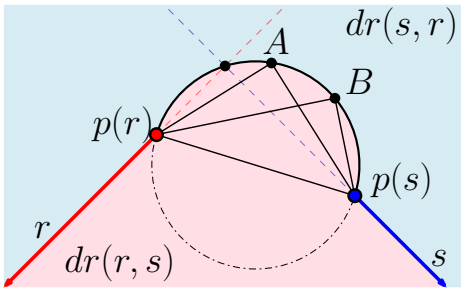
(g) Parallel (h) Anti-parallel: $l(r) \neq l(s)$ (i) Anti-parallel: $l(r) = l(s)$

Figure 4.1. The bisector of two rays r (⟶) and s (⟶) in different configurations. The bisector consists of r, s, and a subset of the circle $C_b(r,s)$ (black curve). The dominance regions are shaded with the respective color.

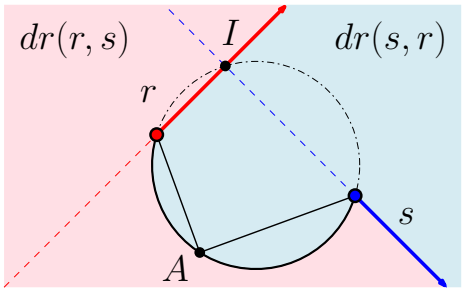
Proof. Any point slightly to the left of ray r has a distance close to 0 to ray r, whereas any point slightly to the right of ray r has a distance close to 2π. Hence the rays r and s are part of the bisector $b_\angle(r,s)$. In the next step we show that all points equidistant to both rays all lie on a common circle.

Let A and B be two points which are equidistant to both rays r and s; see Fig. 4.2a. This means that $\angle(B,p(r),A) = \angle(B,p(s),A)$. We now show that $\angle(p(r),A,p(s)) = \angle(p(r),B,p(s))$, which implies that $p(r),p(s),A,B$ all lie on a circular arc connecting $p(r)$ and $p(s)$ by the inscribed angle theorem:

$$\begin{aligned}\angle(p(r),A,p(s)) &= \pi - \angle(p(s),p(r),A) - \angle(A,p(s),p(r)) \\ &= \pi - (\angle(p(s),p(r),B) + \angle(B,p(r),A)) \\ &\quad - (\angle(B,p(s),p(r)) - \angle(B,p(s),A)) \\ &= \pi - \angle(p(s),p(r),B) - \angle(B,p(s),p(r))\end{aligned}$$



(a) Any two points equidistant to rays r and s lie on a common circle.



(b) Point I lies on the common circle with all the points equidistant to rays r and s.

Figure 4.2. Illustrations for the proof of Lemma 4.1.

$$= \angle(p(r), B, p(s)).$$

In the final step we show that $I = l(r) \cap l(s)$ lies on the common circle with all the equidistant points. If I lies on both (resp. none) of the rays r and s then $d_\angle(I,r) = d_\angle(I,s) = 0$ (resp. $d_\angle(I,r) = d_\angle(I,s) = \pi$). In this case I is equidistant to both rays and therefore clearly on the common circle.

Let us assume that I lies on exactly one of the rays r and s; see Fig. 4.2b. Let A be a point equidistant to both rays, i.e. $\angle(I, p(r), A) = \pi + \angle(I, p(s), A)$. Then,

$$\begin{aligned}\angle(p(s), A, p(r)) &= 2\pi - \angle(A, p(r), I) - \angle(p(r), I, p(s)) - \angle(I, p(s), A) \\ &= \pi - \angle(p(r), I, p(s)).\end{aligned}$$

Therefore by the inscribed angle theorem A, $p(r)$, I and $p(s)$ lie on opposite sides of a common circle. □

Angular difference. Following, we give some deeper insight regarding the properties of angular bisector, by considering the *angular difference* of the two rays.

Definition 4.2. The *angular difference* between r and s, denoted by $\mathrm{diff}_\angle(r,s)$, is the angle by which you have to rotate s counterclockwise around its apex, such that r and s become parallel.

The angular difference of two rays is illustrated in Fig. 4.3. Note that for any two non-parallel rays r and s we have that $\mathrm{diff}_\angle(r,s) + \mathrm{diff}_\angle(s,r) = 2\pi = 0$.

Remark 4.2. *Given a pair of rays r and s, the distance function is monotone along the circular arc of their bisector $b_\angle(r,s)$, and strictly monotone if the lines $l(r)$ and $l(s)$ are not parallel. If the lines $l(r)$ and $l(s)$ are parallel, then the distance is constant along the entire circular part of the bisector $b_\angle(r,s)$.*

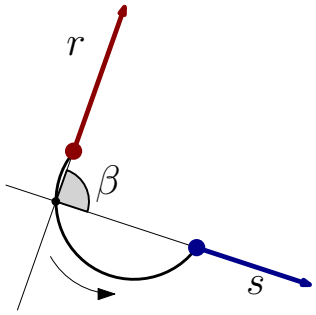


Figure 4.3. The angular difference $\mathrm{diff}_\angle(r,s) = \beta$. The angular distance is increasing from $p(r)$ to $p(s)$

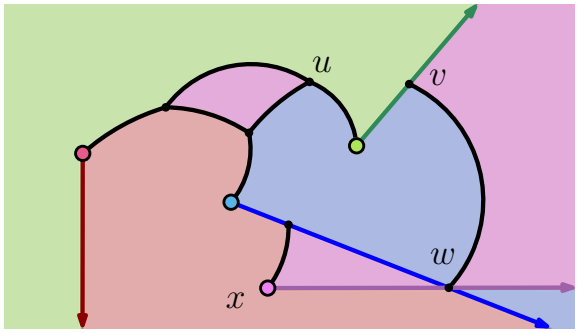


Figure 4.4. Illustration of the features of $\mathrm{RVD}(\mathcal{R})$: arc $\overline{vw}$ is a circular edge, segment $\overline{xw}$ is a ray edge, u is a proper vertex, v is a mixed vertex, w is an intersection vertex, and x is an apex vertex.

If instead $\mathrm{diff}_\angle(r,s) > \mathrm{diff}_\angle(s,r)$, or equivalently $\mathrm{diff}_\angle(s,r) < \pi$, then the distance function along the bisector $b_\angle(r,s)$ from $p(s)$ to $p(r)$ is monotone increasing. Moreover, walking along the boundary of $dreg(s,r)$ in counterclockwise order, the distance function on the circular part of bisector $b_\angle(r,s)$ is monotone increasing.

Rotating rays Voronoi diagram. Recall that we defined the rotating rays Voronoi diagram (see Definition 1.8), as the subdivision of $\mathbb{R}^2$ into *nearest ray (Voronoi) regions*, where the region of a ray $r \in \mathcal{R}$ is

$$r_\angle reg(r) = \{\, x \in \mathbb{R}^2 \mid \forall s \in \mathcal{R}\backslash\{r\} :\ d_\angle(x,r) < d_\angle(x,s) \,\}.$$

Observe that a Voronoi region $r_\angle reg(r)$ can be equivalently defined as the intersection of all the dominance regions of r, that is

$$r_\angle reg(r) = \bigcap_{s \in \mathcal{R}\backslash\{r\}} \mathrm{dr}(r,s).$$

We denote the graph structure of the rotating rays Voronoi diagram of $\mathcal{R}$ by

$$\mathrm{RVD}(\mathcal{R}) := \left(\mathbb{R}^2 \backslash \bigcup_{r \in \mathcal{R}} r_\angle reg(r) \right) \cup \mathcal{R}.$$

Note that it is necessary to include $\mathcal{R}$ in the above definition, as points along the rays are not equidistant to two sites and hence are not included in $\mathbb{R}^2 \backslash \bigcup_{r \in \mathcal{R}} r_\angle reg(r)$.

For simplicity, we can assume that unless otherwise stated no two supporting lines of rays are parallel and that no two rays share an apex. As a result we

have only bisectors of the forms illustrated in Figs. 4.1a to 4.1e. We distinguish the following two types of edges and four types of vertices of RVD($\mathcal{R}$). Refer to Fig. 4.4 for an illustration.

- A *circular edge* is a subset the circular part of a bisector, so any point on a circular edge is equidistant to the two rays which induce it.
- A *ray edge* is a subset of a ray, so any point on a ray edge has zero distance to the ray which induces it.
- A *proper vertex* is incident to three circular edges, so it is equidistant to the three rays that induce the three circular edges.
- A *mixed vertex* is incident to one circular edge and two ray edges that are subsets of the same ray. It is equidistant to the two sites inducing the circular edge and has zero distance to the site inducing the ray edges.
- An *intersection vertex* is incident to one circular edge and four ray edges, all induced by two sites. It is equidistant to the two sites, with distance 0.
- An *apex vertex* is incident to one circular edge and one ray edge, where the site inducing the ray edge is one of the two sites inducing the circular edge. It has distance 0 to the site inducing the ray edge, but is not equidistant to the other site.

RVD($\mathcal{R}$) is a planar graph with bounded maximum degree. Hence, when looking into upper bounds of the complexity of the diagram, it suffices to bound any of the number of vertices, edges or faces.

4.2 Diagram in the plane

In this section we study RVD($\mathcal{R}$) in the plane. We first look at some properties and combinatorial complexity bounds. Then we consider the problem of illuminating the plane with a set of floodlights aligned with $\mathcal{R}$.

4.2.1 Properties, complexity, and an algorithm

Assuming that no two rays of $\mathcal{R}$ are parallel to each other, we show that the following two simple structural properties hold.

Proposition 4.3. *RVD($\mathcal{R}$) has exactly n unbounded faces, one for each ray.*

Proof. Given a ray r, to examine the unbounded portion of $r_{\angle}reg(r)$, consider the intersection of RVD($\mathcal{R}$) with a disk D of a sufficiently large radius, so that D

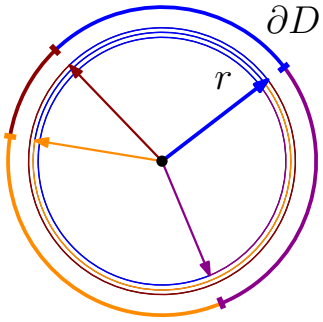


Figure 4.5. Intersection of a diagram of 4 rays with a large disk D. Dominance regions are circular arcs on ∂D.

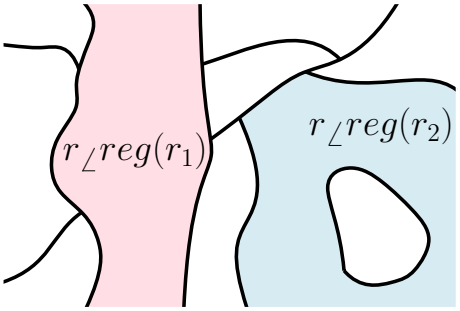


Figure 4.6. Two impossible cases for a diagram. A "corridor" ($r_\angle reg(r_1)$) and an "island" ($r_\angle reg(r_2)$).

contains all vertices of RVD($\mathcal{R}$), and the bisecting circles of all bisectors. Refer to Fig. 4.5 for an illustration.

Given a site $s \in \mathcal{R}\backslash\{r\}$, the intersection of the dominance region dr(r,s) with ∂D is a circular arc of ∂D going counterclockwise from $r \cap \partial D$ to $s \cap \partial D$. Region $r_\angle reg(r)$ is the intersection of all the dominance regions of r. Thus, $r_\angle reg(r) \cap \partial D$ is the intersection of $n-1$ circular arcs all starting from r. This coincides with the circular arc ending at the first ray on the counterclockwise ordering along ∂D from r. So, for any ray r, region $r_\angle reg(r)$ has exactly one unbounded face. □

Proposition 4.4. *RVD($\mathcal{R}$) is connected.*

Proof. Assume that RVD($\mathcal{R}$) is not connected and that it has two connected components. Then, some region $r_\angle reg(r)$ disconnects RVD($\mathcal{R}$) by either having an unbounded face with two occurrences at infinity, creating a "*corridor*", or by enclosing a component in it, creating an "*island*"; refer to the illustration of Fig. 4.6.

The existence of a corridor is excluded by the proof of Lemma 4.3. For the existence of an island, consider the disconnected component of RVD($\mathcal{R}$) *surrounded* by $r_\angle reg(r)$. This component consists of at least a face of a region $r_\angle reg(s)$ for some $s \in \mathcal{R}$. Then, also in RVD($\{r,s\}$), there is an island inside $r_\angle reg(r)$. Thus, $b_\angle(r,s)$ has a bounded connected component, in contradiction to the fact that each bisector is a single unbounded curve. □

We now study the combinatorial complexity of RVD($\mathcal{R}$). Regarding a lower bound, it is not hard see that if we have a set $\mathcal{R}$ of n pairwise intersecting rays, RVD($\mathcal{R}$) has $\binom{n}{2} = \Theta(n^2)$ vertices at the intersection of rays and thus the diagram has $\Omega(n^2)$ worst-case complexity. Following, we show that this bound also holds for pairwise non-intersecting rays.

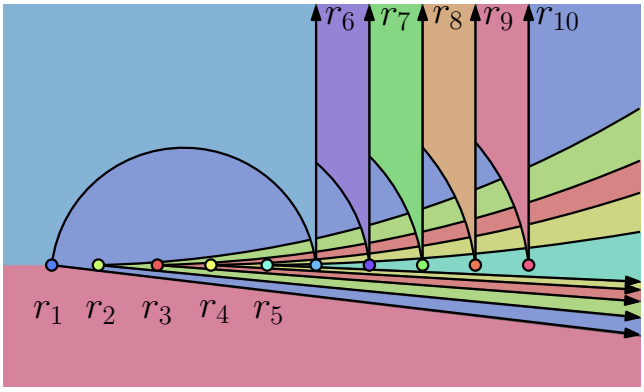


Figure 4.7. A set $\mathcal{R}$ of 10 pairwise non-intersecting rays with RVD($\mathcal{R}$) having $\Theta(n^2)$ complexity. The region $r_{\angle}reg(r_i)$, $i = 1,\ldots,4$, has $\Theta(n)$ bounded faces.

Theorem 4.1. *The worst-case combinatorial complexity of RVD($\mathcal{R}$) has an $\Omega(n^2)$ lower bound, even if the rays are pairwise non-intersecting.*

Proof. To realize the bound, we describe a construction of a set of rays $\mathcal{R}$, where $\Theta(n)$ sites have a Voronoi region with $\Theta(n)$ bounded faces each. An illustration of this construction for a set of 10 rays is illsutrated in Fig. 4.7. Observe that the Voronoi region $r_{\angle}reg(r_i)$, for $i = 1,\ldots,4$, has a bounded face incident to the ray r_j, for $j = 6,\ldots,10$. The construction can be described as follows.

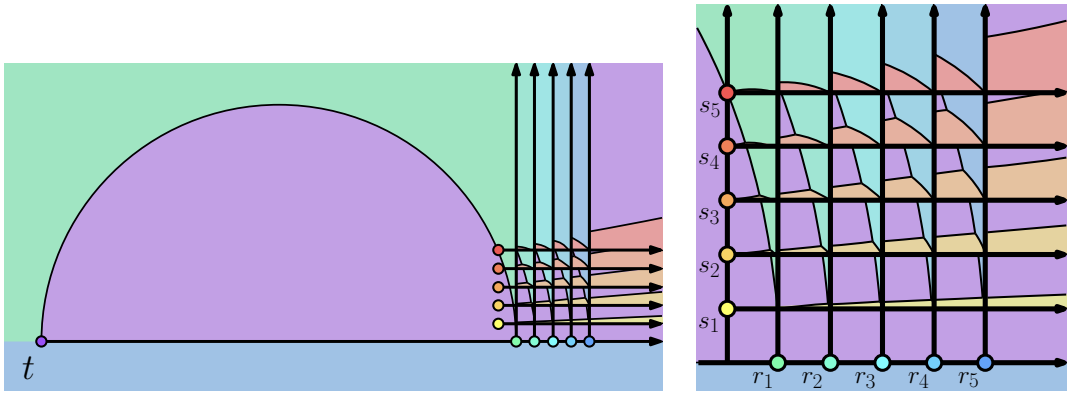
We set $n = 2m$ and let $p(r_i) = (i,0)$, $i = 1,\ldots,2m$, with rays $r_{m+1},\ldots,r_{2m}$ pointing vertically upwards. For $i = 1,\ldots,m$, let the direction of r_i be $\widehat{d}(r_i) = (\sin\alpha_i, \cos\alpha_i)$ with $\alpha_1 \in (3\pi/2, 2\pi)$ and $\alpha_i = \alpha_{i-1} + \epsilon_i$ where $\epsilon_i > 0$ for $i = 2,\ldots,m$. We choose ϵ_i one by one, in the increasing order of i, so that both r_i and r_{i+1} have a face between any two consecutive upward shooting rays. This is always possible to do since we can choose ϵ_i small enough so that, at any x-coordinate, with $x < 2m$, the circular part of $b_{\angle}(r_i, r_{i+1})$ is arbitrarily close to the x-axis and, thus, it is below the circular part of $b_{\angle}(r_{i-1}, r_i)$.

Hence, each region $r_{\angle}reg(r_i)$, with $i = 1,\ldots m$ has $\Theta(n)$ bounded faces and complexity. So, the constructed diagram has $\Theta(n^2)$ complexity and the lower bound follows, concluding the proof. □

It follows from the construction of Theorem 4.1 that a single Voronoi region can have $\Omega(n)$ combinatorial complexity. We extend this as follows.

Theorem 4.2. *A Voronoi region of RVD($\mathcal{R}$) has $\Theta(n^2)$ complexity in the worst case.*

Proof. We first argue that the complexity of a region is $O(n^2)$. A vertex v of RVD($\mathcal{R}$) can be defined by a triplet of rays $r,s,t \in \mathcal{R}$. The bisectors $b_{\angle}(r,s)$



(a) Construction zoomed out. (b) Construction zoomed in.

Figure 4.8. A set $\mathcal{R}$ of 11 rays with RVD($\mathcal{R}$). The region $r_{\angle}reg(t)$ has $\Theta(n^2)$ faces, one in each cell of the grid formed by $\{r_1,\ldots,r_5,s_1,\ldots,s_5\}$.

and $b_{\angle}(r,t)$ intersect $O(1)$ times, hence, RVD($\{r,s,t\}$) has $O(1)$ vertices. Now consider a ray r and its region $r_{\angle}reg(r)$. All but at most $O(n)$ vertices on the boundary of $r_{\angle}reg(r)$ are defined by r and a pair of sites. There are $\Theta(n^2)$ pairs, each inducing $O(1)$ vertices on $r_{\angle}reg(r)$, so $r_{\angle}reg(r)$ has $O(n^2)$ vertices.

We now give a construction of $n = 2m+1$ rays, where a single region has $\Theta(n^2)$ complexity; refer to the construction of Fig. 4.8. We first create a grid structure. For $i = 1,\ldots,m$, let r_i be a ray with $p(r_i) = (i,0)$ shooting vertically upward and let s_i be a ray with $p(s_i) = (0,i)$ shooting horizontally to the right; see Fig. 4.8b. For all $(i,j) \in \{1,\ldots,m-1\}^2$, let $R(i,j)$ be the square $[i,i+1)\times[j,j+1)$. Each square $R(i,j)$ is made up of two faces of RVD($\{r_1,\ldots,r_m,s_1,\ldots,s_m\}$), one belonging to $r_{\angle}reg(r_i)$ and one belonging to $r_{\angle}reg(s_j)$. Now let $\alpha(i,j) := \max\{\min\{d_{\angle}(x,r_i),d_{\angle}(x,s_j)\} \mid x \in R(i,j)\}$ and let $\alpha_{\min} := \min\{\alpha(i,j) \mid (i,j) \in \{1,\ldots,m-1\}^2\}$. It is easy to see that $\alpha_{\min} < \arctan 1/(m-1)$.

We now introduce another ray t, so that $\max\{d_{\angle}(x,t) \mid x \in [1,n-1]^2\} < \alpha_{\min}$. This can be achieved if $p(t) = (-n^2,0)$ and t is shooting horizontally to the right; see Fig. 4.8a. This means that in each $R(i,j)$, for $(i,j) \in \{1,\ldots,n-1\}^2$, t will visit some point before any of the rays r_i or s_j, implying that $r_{\angle}reg(t)$ has $\Theta(n^2)$ faces. □

The above directly implies an $O(n^3)$ upper bound on the complexity of RVD($\mathcal{R}$). A similar upper bound can also be obtained using the extended abstract Voronoi diagram framework of Bohler and Klein [2014]. Following, we show how the distance function can be adapted in order to apply the general $O(n^{2+\varepsilon})$ upper bound by Sharir [1994]. As a by-product, we also obtain a construction algorithm.

Theorem 4.3. *For any $\epsilon > 0$, RVD$(\mathcal{R})$ has $O(n^{2+\epsilon})$ combinatorial complexity. Further, RVD$(\mathcal{R})$ can be constructed in $O(n^{2+\epsilon})$ time.*

Proof. Each site r induces a function $d^r_\angle(x) = d_\angle(x, r)$ which maps a point $x = (x_1, x_2) \in \mathbb{R}^2$ to its angular distance from r. The RVD can be seen as the projection of the lower envelope of the graphs of these distance functions in 3-space to the plane. For algebraic distance functions, Sharir [1994] gives complexity bounds for this lower envelope accompanied with algorithmic results. The angular distance functions though are not algebraic. Therefore, our strategy is to find algebraic functions d^r_{alg} that are equivalent to the functions $d^r_\angle$ for the computation of the lower envelope, i.e. they fulfill the following property: $d^r_\angle(x) < d^s_\angle(x) \Leftrightarrow d^r_{\text{alg}}(x) < d^s_{\text{alg}}(x)$ for all $r, s \in \mathcal{R}$ and $x \in \mathbb{R}^2$.

Without loss of generality, assume that $p(r)$ lies on the origin and r is facing to the right in positive x_1-direction of the coordinate system. Let $x \in \mathbb{R}^2$ and $\alpha := d^r_\angle(x)$. Then we want to set $d^r_{\text{alg}}(x) := 1 - \cos(\alpha)$ if $0 \leqslant \alpha \leqslant \pi$, and $d^r_{\text{alg}}(x) := 3 + \cos(\alpha)$ if $\pi \leqslant \alpha < 2\pi$. The function $x \mapsto \cos(\alpha)$ is indeed algebraic since it is obtained by first scaling x to unit length and then mapping it to its first coordinate. Then we have

$$d^r_{\text{alg}}(x_1, x_2) = \begin{cases} 0 & \text{if } x_1 = x_2 = 0, \\ 1 - \frac{x_1}{\sqrt{x_1^2 + x_2^2}} & \text{if } x_1 \neq 0,\ x_2 \geqslant 0, \\ 3 + \frac{x_1}{\sqrt{x_1^2 + x_2^2}} & \text{otherwise.} \end{cases}$$

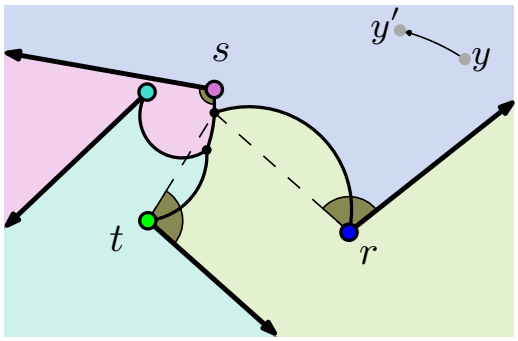
Since d^r_{alg} consists of three patches, which are all algebraic and have simple domain boundaries, applying the results of Sharir [1994] to these functions yields the claimed results. □

4.2.2 Brocard illumination of the plane

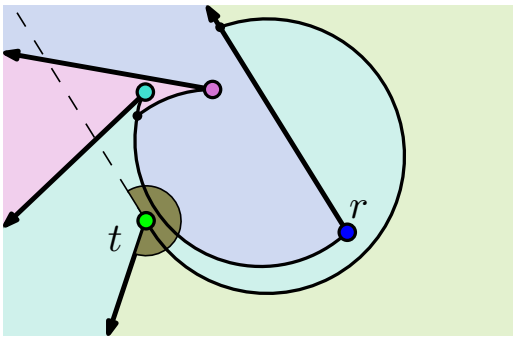
We now look into the Brocard illumination problem in $\mathbb{R}^2$. Recall that given a set of rays $\mathcal{R}$, and an α-floodlight aligned with each ray, the problem asks for the Brocard angle (see Definition 1.9), the minimum angle needed to illuminate a target domain, $\mathbb{R}^2$ in this case. The Brocard angle is

$$\alpha^* = \max_{x \in \mathbb{R}^2} \min_{r \in \mathcal{R}} d_\angle(x, r).$$

Let $x^* \in \mathbb{R}^2$ be a point that realizes α^*, or the last point to be illuminated considering the rotating rays perspective.



(a) α^* is realized on a vertex of RVD($\mathcal{R}$) by rays r, s, t. Point y' is further than y to its nearest ray.



(b) α^* is realized on a ray (r) at infinity, by ray t.

Figure 4.9. Two examples of the Brocard angle ($\measuredangle$) on a set $\mathcal{R}$ of 4 rays in $\mathbb{R}^2$.

Proposition 4.5. *The Brocard angle of a set $\mathcal{R}$ of rays is realized at a vertex of RVD($\mathcal{R}$), or at a point at infinity along a ray.*

Proof. Suppose x^* is not on RVD($\mathcal{R}$), but instead inside a Voronoi region of a ray r. Then, we can always find a point with larger angular distance by simply moving in counterclockwise direction on the circle with center $p(r)$ and radius $d(p(r), x^*)$; see for example the points y and y' in Fig. 4.9a. Hence x^* lies on RVD($\mathcal{R}$).

The distance along a circular edge is monotone. Therefore, the distance at one of its two endpoints is at least as big as the distance at any point in the interior of the edge. The same argument also holds true for the distances along ray edges.

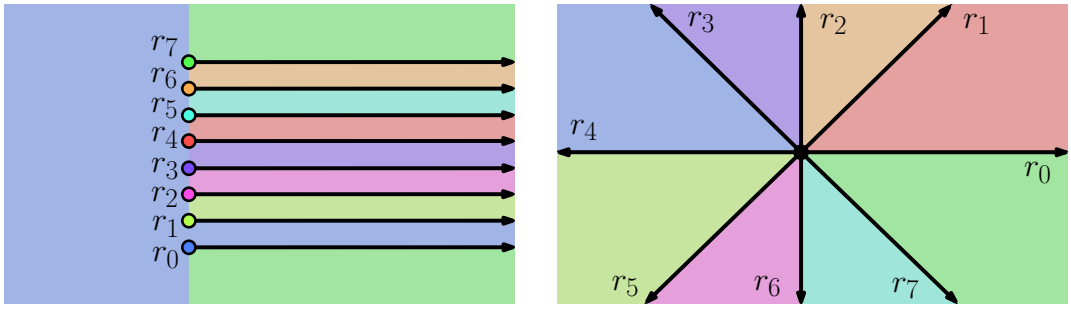
Hence, a point with maximum distance, i.e., a point realizing the Brocard angle, is either at a vertex of RVD($\mathcal{R}$) (see Fig. 4.9a), or a point at infinity on a ray of $\mathcal{R}$ (see Fig. 4.9b), concluding the proof. □

The above implies that we can find x^*, and hence α^*, by first constructing RVD($\mathcal{R}$) in $O(n^{2+\epsilon})$ time, as proved in Theorem 4.3, and then traversing the diagram. RVD($\mathcal{R}$) is a plane graph, so it can be traversed in linear time in its size using standard methods. This results in the following theorem.

Theorem 4.4. *The Brocard angle of a set $\mathcal{R}$ of rays can be found in $O(n^{2+\epsilon})$ time.*

We conclude this section by giving tight bounds on the value of the angle α^*.

Proposition 4.6. *The set of values that the Brocard angle of a set of rays in $\mathbb{R}^2$ can achieve is $[2\pi/n, 2\pi]$.*



(a) $\alpha^* = 2\pi$. (b) $\alpha^* = 2\pi/n$.

Figure 4.10. Sets of 8 rays realizing the bounds of the Brocard angle in $\mathbb{R}^2$.

Proof. For the upper bound consider a set $\mathcal{R}$ of n parallel rays: let r_i have $p(r_i) = (i, 0)$ and $\widehat{d}(r_i) = (1, 0)$ for $i \in \{0, \ldots, n-1\}$. An example of such a set of 8 rays is illustrated in Fig. 4.10a. Observe that the last point to be illuminated is the point on r_0 at infinity, i.e., point $(0, +\infty)$, which will be illuminated by r_{n-1} when α reaches 2π; hence the upper bound follows.

For the lower bound, consider that in order to illuminate the entire $\mathbb{R}^2$, all the points at *infinity* should also be illuminated. To illuminate these points, the sum of the angles of all rays, should be at least 2π. Hence, in the best case, a point at infinity is seen by exactly one ray, and a $2\pi/n$ lower bound follows.

A construction where the $2\pi/n$ lower bound is achieved the following. Let $\mathcal{R}$ be a set of n rays having apex at $(0, 0)$ and with the property that any two consecutive rays have an angular difference of $2\pi/n$. See an example of such a construction with 8 rays in Fig. 4.10b. The last points to be illuminated will be all the points on the right side of each ray r_i. These points are illuminated simultaneously by r_{i-1} when α reaches $2\pi/n$. Further, the above construction can be easily adapted to attain any value in $(2\pi/n, 2\pi)$, by expanding a wedge formed by two consecutive rays and shrinking all the other analogously. □

4.3 Diagram of a convex polygon

We now turn our attention to the Brocard illumination problem of a convex polygon. We are given a convex polygonal domain $\mathcal{P}$ with n vertices, and we want to find the Brocard angle α^* of $\mathcal{P}$. By computing a Voronoi diagram of rays restricted to $\mathcal{P}$, we show how to compute α^* in optimal $\Theta(n)$ time. For simplicity, when it is clear from the context we also refer to the boundary $\partial\mathcal{P}$ as simply $\mathcal{P}$.

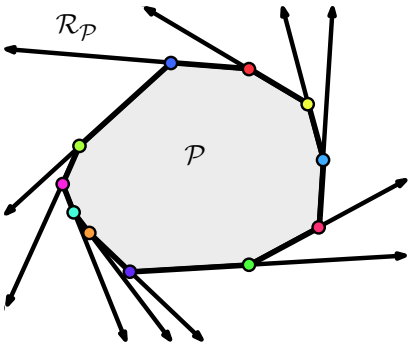


Figure 4.11. A convex polygon $\mathcal{P}$ and the corresponding set of rays $\mathcal{R}_\mathcal{P}$.

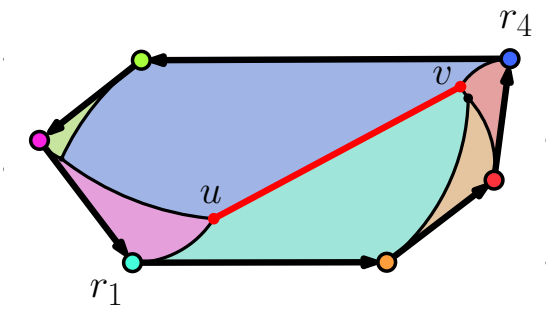


Figure 4.12. A polygon $\mathcal{P}$ with parallel edges (r_1, r_4), together with PRVD$(\mathcal{R}_\mathcal{P})$. The brocard angle is realized at every point on the edge uv.

Let $\mathcal{R}_\mathcal{P}$ be the set of n rays such that each ray has a vertex $v \in P$ as apex, and passes through the successor of v in the counterclockwise order of the vertices of $\mathcal{P}$. See an example of a polygon $\mathcal{P}$ and the set $\mathcal{R}_\mathcal{P}$ in Fig. 4.11. We define

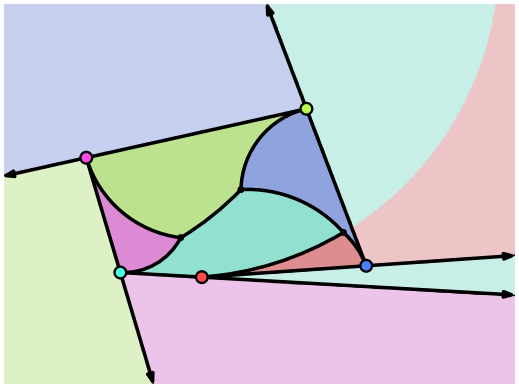
$$\text{PRVD}(\mathcal{R}_\mathcal{P}) := \text{RVD}(\mathcal{R}_\mathcal{P}) \cap \mathcal{P},$$

to be the Voronoi diagram of $\mathcal{R}_\mathcal{P}$ restricted inside the polygon $\mathcal{P}$; see, e.g., the diagrams in Fig. 4.12 and Fig. 4.13a. Note that we only consider PRVD$(\mathcal{R}_\mathcal{P})$, which we show to be of $\Theta(n)$ complexity, instead of the entire RVD$(\mathcal{R}_\mathcal{P})$, which may have $\Theta(n^2)$ complexity.

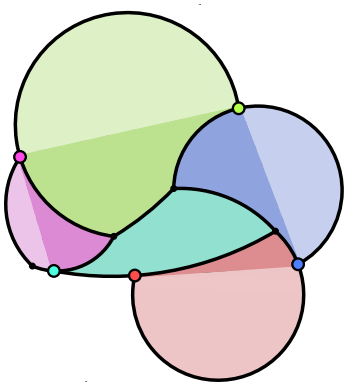
The rest of the section is organized as follows. In Section 4.3.1, we describe some useful properties of PRVD$(\mathcal{R}_\mathcal{P})$. In Section 4.3.2 we describe a simple $\Theta(n \log n)$ time algorithm to construct PRVD$(\mathcal{R}_\mathcal{P})$. In Section 4.3.3 we give an optimal $\Theta(n)$-time construction algorithm. In Sections 4.3.4 and 4.3.5 we give more details regarding parts of the $\Theta(n)$-time algorithm. Finally, in Section 4.3.6, we apply our results on PRVD$(\mathcal{R}_\mathcal{P})$ to find the Brocard angle α^* of $\mathcal{P}$.

4.3.1 Properties of the diagram

For the sake of simplicity we make the following assumptions. First, that no 3 vertices of $\mathcal{P}$ are collinear. Second, that no point of $\mathcal{P}$ is equidistant to 4 rays; this implies that every vertex of PRVD$(\mathcal{P})$ is incident to 3 edges. Finally, that there are no parallel edges in $\mathcal{P}$, i.e., that there are no anti-parallel rays in $\mathcal{R}_\mathcal{P}$; this guarantees that the Brocard angle is realized at a unique point, which is a Voronoi vertex of RVD$(\mathcal{R}_\mathcal{P})$. Alternatively, if there are anti-parallel rays in $\mathcal{R}_\mathcal{P}$, the



(a) The diagram PRVD($\mathcal{R}_\mathcal{P}$). RVD($\mathcal{R}_\mathcal{P}$) outside $\mathcal{P}$ is shown faded.



(b) The diagram DD($\mathcal{R}_\mathcal{P}$). DD($\mathcal{R}_\mathcal{P}$) outside $\mathcal{P}$ is shown faded.

Figure 4.13. A polygon $\mathcal{P}$ of 5 vertices together with PRVD($\mathcal{R}_\mathcal{P}$) and DD($\mathcal{R}_\mathcal{P}$).

Brocard angle may be realized on any point of a single Voronoi edge, which is part of the bisector of the 2 anti-parallel edges; see, e.g., the edge $\overline{uv}$ in Fig. 4.12.

Disk diagram. To help prove some properties of PRVD($\mathcal{R}_\mathcal{P}$), we define an auxiliary Voronoi diagram, the *disk (Voronoi) diagram (DD)*. The basic idea of the disk diagram is to use instead of angular bisectors only the bisecting circles (without the rays). This is because the structure of the bisector system, and consequently of the diagram, is simpler to study. Still, as we will show the disk diagram coincides with the rays Voronoi diagram, in the domain of interest, i.e., inside $\mathcal{P}$.

Formally, given two rays r and s, we define the *DD bisector* of r and s to be the entire bisecting circle $C_b(r,s)$. The *DD dominance region* of r over s, denoted by $\mathrm{dr}_D(r,s)$, is either the interior or the exterior of that circle, depending on the angular difference of the two rays. If $\mathrm{diff}_\angle(r,s) < \pi$, then $\mathrm{dr}_D(s,r)$ is the interior of $C_b(r,s)$ and $\mathrm{dr}_D(s,r)$ the exterior, and the other way round if $\mathrm{diff}_\angle(r,s) \geqslant \pi$.

The disk diagram of a set of rays $\mathcal{R}_\mathcal{P}$, is the union of all *DD regions*, where for a ray $r \in \mathcal{R}$, its DD region is

$$dreg(r) := \bigcap_{s \in \mathcal{R}_\mathcal{P} \setminus \{r\}} \mathrm{dr}_D(r,s).$$

See an example in Fig. 4.13b. We denote the graph structure of the diagram by DD($\mathcal{R}_\mathcal{P}$). Note that for $n \geqslant 3$ the diagram does not cover $\mathbb{R}^2$, as there are areas having a cyclic dominance relation among sites (the outer area in Fig. 4.13b).

An interesting observation is that each point in the neighborhood of the circular part of a bisector is associated to the same ray-site in both the rotating

rays Voronoi diagram and the disk diagram. Since in the interior of $\mathcal{P}$, RVD($\mathcal{R}_\mathcal{P}$) consists only of circular parts of bisectors, RVD($\mathcal{R}_\mathcal{P}$) and DD($\mathcal{R}_\mathcal{P}$) are exactly the same in $\mathcal{P}$. Observe this equivalence in the two illustration of Fig. 4.13.

Lemma 4.7. *Each region $dreg(r)$ of DD($\mathcal{R}_\mathcal{P}$) is connected and contains $p(r)$ on its boundary.*

Proof. By definition, the region $dreg(r)$ is formed by taking the intersection of $n-1$ disks and complements of disks. The boundary of each of these disks is $C_b(r,s)$ for some s, and since by definition the apex $p(r)$ lies on $C_b(r,s)$, it also lies on the boundary of the intersection of all these disks.

To see that the region $dreg(r)$ is connected, we perform an inversion of the plane using $p(r)$ as the inversion center, and a circle of arbitrary radius as the inversion circle. This inversion maps circles passing through the inversion center to lines passing through the inversion center, so each dominance region $\mathrm{dr}_D(r,s)$ maps to a halfplane. The intersection of halfplanes is connected, and since the inversion preserves connectivity, region $dreg(r)$ is also connected. □

The following statement follows immediately from Lemma 4.7 and the corresponding equivalence of PRVD($\mathcal{R}_\mathcal{P}$) and DD($\mathcal{R}_\mathcal{P}$) in $\mathcal{P}$.

Proposition 4.8. *PRVD($\mathcal{R}_\mathcal{P}$) is a tree structure of $\Theta(n)$ complexity.*

We now turn our attention back to PRVD($\mathcal{R}_\mathcal{P}$); we will use again the disk diagram in Section 4.3.5.

Lemma 4.9. *A region $r_\angle reg(r_i)$ of PRVD($\mathcal{R}_\mathcal{P}$) has the following form: it consists of a single face sharing the polygon edge (of the corresponding ray). The distance on the boundary $\partial r_\angle reg(r_i)$ has a global maximum r_i^* and the distance along $\partial r_\angle reg(r_i)$ is monotone increasing from r_i, and r_{i+1}, towards the point realizing the maximum.*

Proof. Refer also to Fig. 4.14 (where $r_i = r_1$). Consider the sequence of sites whose faces are adjacent to the face of r_i in counterclockwise order. We show (i) that this sequence is actually a sub-sequence of $(r_{i+1}, r_{i+2}, ..., r_n, r_1, ..., r_{i-1})$, and (ii) that the distance along the sequence is monotonically increasing.

(i) Let r_j be a ray such that $r_\angle reg(r_i)$ is adjacent to $r_\angle reg(r_j)$. By Lemma 4.7, each region is connected and incident to its corresponding ray, so the union of and $r_\angle reg(r_j)$ *splits* the polygon in two simply connected components; see the regions $r_\angle reg(r_1)$ and $r_\angle reg(r_5)$ in Fig. 4.14. Given a second ray r_k whose region is adjacent to the region $r_\angle reg(r_i)$, it follows that both r_k and the edge between $r_\angle reg(r_i)$ and $r_\angle reg(r_k)$ have to be in the same connected component, as also

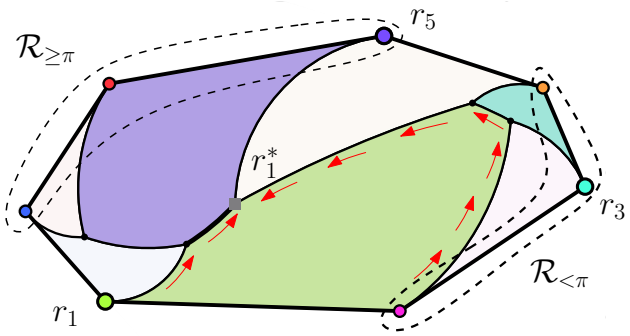


Figure 4.14. Illustration of the properties of a region $r_\angle reg(r_1)$. The distance along the boundary $\partial r_\angle reg(r_1)$ is increasing towards the maximum r_1^* (■). The sites in $\mathcal{R}_\mathcal{P}\backslash r_1$ are split in two sets $\mathcal{R}_{<\pi} = \{r_2, r_3, r_4\}$ and $\mathcal{R}_{\geqslant\pi} = \{r_5, r_6, r_7\}$. The (neighboring) regions of r_1 and r_5 split $\mathcal{P}$ in two connected components.

$r_\angle reg(r_k)$ is connected; see $r_\angle reg(r_3)$ in Fig. 4.14. Thus, the order of r_j and r_k along the boundary of the polygon and the face of r_i is the same.

(ii) We partition the set $\mathcal{R}_\mathcal{P}\backslash r_i$ in two sets depending on the angular difference with r_i. The set $\mathcal{R}_{\geqslant\pi}$, of rays with angular difference $\text{diff}_\angle(r_j, r_i) \geqslant \pi$, and the set $\mathcal{R}_{<\pi}$, of rays with angular difference $\text{diff}_\angle(r_j, r_i) < \pi$; see the dashed curves in Fig. 4.14. Along the chain of rays $\mathcal{R}_{<\pi}$ (resp. $\mathcal{R}_{\geqslant\pi}$) the distance is increasing from r_{i+1} (resp. r_i) towards the point realizing r_i^*.

We give an inductive argument for the monotonicity property along chain $\mathcal{R}_{<\pi}$, starting initially only with the Voronoi diagram of the two rays r_i and r_{i+1}, and then incrementally adding more rays of the set $\mathcal{R}_{<\pi}$ in counterclockwise order. The base case follows directly from the properties of the bisectors, see Remark 4.2. Suppose we are now adding site r_k to $\text{RVD}(\{r_i, \ldots, r_{j-1}\})$. Because of property (i), if there is an edge between r_i and r_j, then it is incident to $p(r_i)$, i.e., it is the last one along the chain of edges of face $r_\angle reg(r_i)$. Let v be the other endpoint of the edge between r_i and r_j. Since $r_j \in \mathcal{R}_{<\pi}$, the distance along the edge r_i and r_j is monotone increasing, from v to $p(r_i)$. Further, by the induction hypothesis, the distance along the chain of edges between r_i and all sites bounding $r_\angle reg(r_i)$ in $\text{RVD}(\{r_i, \ldots, r_{j-1}\})$ is monotonically increasing in counterclockwise order, from r_{i+1} to v.

The proof for $\mathcal{R}_{\geqslant\pi}$ is analogous, but instead, the distance increases in clockwise order. □

Corollary 4.10. *Given a vertex $u \in PRVD(\mathcal{R}_\mathcal{P})$ at least two incident edges have a distance increasing towards u.*

Proof. Assume for the sake of contradiction, that there is a vertex u with two

incident edges having distance decreasing towards u. These two edges are part of a chain of edges bounding a region $r_{\angle} reg(r_i)$; this contradicts Lemma 4.9. □

Lemma 4.11. *Given an angle c, the set of all points in the interior of $\mathcal{P}$, which have at least distance c to their nearest site, form a convex polygon B.*

Proof. Note the set of all points at distance c from a ray of $\mathcal{R}_{\mathcal{P}}$ is a half-line. Then B is a convex polygon, since it is the intersection of the halfplanes defined by all the rays in $\mathcal{R}_{\mathcal{P}}$, after being rotated by c. □

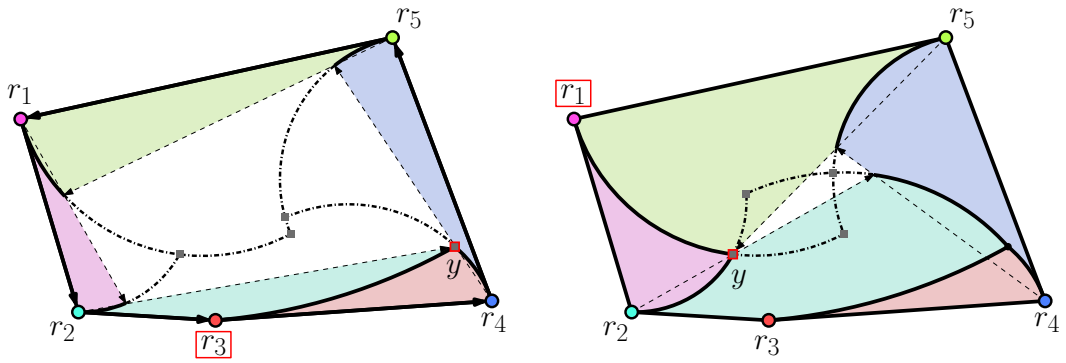
4.3.2 A simple $O(n \log n)$-time algorithm

Before describing the optimal $\Theta(n)$-time algorithm to construct PRVD$(\mathcal{R}_{\mathcal{P}})$, we first describe a simple $\Theta(n \log n)$-time algorithm. The algorithm employs a "*collapse*" strategy: starting at the boundary of the domain (where all the points have distance zero to its nearest site), the algorithm gradually constructs the diagram adding edges and vertices with greater distance until the vertex of maximum distance is reached.

Algorithms employing a similar strategy have been designed to construct the farthest point Voronoi diagram by Skyum [1991] and the farthest segment Voronoi diagram by Aurenhammer et al. [2006], gradually discovering the edges and vertices of the diagram in decreasing distance (from $+\infty$ until the vertex of minimum distance). The $O(n \log n)$-time algorithm of Alegría-Galicia et al. [2017] finds the Brocard angle in a similar manner, without constructing PRVD$(\mathcal{R}_{\mathcal{P}})$.

Algorithm outline. Refer also to Fig. 4.15. The algorithm starts at the vertices of $\mathcal{P}$, which are all starting points of edges of PRVD$(\mathcal{R}_{\mathcal{P}})$. For every pair of edges that are consecutive in circular order, their next intersection point is computed (if one exists). Out of these intersection points, the one with minimum distance to its nearest site is the next vertex of the diagram. This "collapse" event is processed by (i) constructing the vertex, (ii) constructing the edges leading to this vertex, (iii) removing the edges from further consideration, and (iv) starting a new edge. At the constructed vertex a face "*collapses*", since it is fully constructed and will not be considered again. The new edge is part of the bisector of the two faces neighboring the collapsed face. This procedure, of computing and processing new "collapse" events, is repeated until all the remaining edges intersect in a single point. This point is the vertex of PRVD$(\mathcal{R}_{\mathcal{P}})$ with maximum distance (which also realizes the Brocard angle).

A pseudocode description is given in Algorithm 1. Let $\mathcal{L}$ denote the circular list of rays, where $\mathcal{L}.\textsc{prev}(r)$ and $\mathcal{L}.\textsc{next}(r)$ return respectively the previous and



(a) 1st event: Vertex y is induced by (r_2, r_3, r_4). The region of r_3 "collapses".

(b) 2nd event: Vertex y is induced by (r_5, r_1, r_2). The region of r_1 "collapses".

Figure 4.15. Illustration of the $O(n \log n)$-time algorithm on a polygon of 5 vertices. The two first events are shown, together with all candidate vertices (■). In a 3rd event (not shown) there is one candidate vertex induced by (r_2, r_4, r_5).

the next ray in the list, and where $\mathcal{L}.\text{NEXT}(r_n) = r_1$, $\mathcal{L}.\text{PREV}(r_1) = r_n$. $\mathcal{Q}$ denotes a min priority queue (with standard $\mathcal{Q}$.PUSH and $\mathcal{Q}$.POP operations) which takes triplets of (distance, ray, vertex) and sorts them according to the distance.

Algorithm 1: $O(n \log n)$-time algorithm to construct PRVD$(\mathcal{R}_\mathcal{P})$.

Input : A convex polygon $\mathcal{P}$ with $n \geqslant 3$ vertices.
Output: The diagram PRVD$(\mathcal{R}_\mathcal{P})$.

$\mathcal{Q} \leftarrow \varnothing$; `// Min priority queue by angular distance`
$\mathcal{L} \leftarrow (r_1, r_2, ..., r_n)$; `// Circular list of rays`
for each ray $r \in \mathcal{L}$ **do**
  if *exists real vertex w in RVD*$(\{\mathcal{L}.\text{PREV}(r), r, \mathcal{L}.\text{NEXT}(r)\})$ **then**
    $\mathcal{Q}$.PUSH$((d_\angle(r, w), r, w))$;
while $|\mathcal{Q}| > 0$ **do**
  $(d, r, v) \leftarrow \mathcal{Q}$.POP$()$; `// Get the min distance candidate`
  Add v and the edges increasing towards v to PRVD$(\mathcal{R}_\mathcal{P})$;
  Remove ray r from $\mathcal{L}$;
  Remove from $\mathcal{Q}$ the events associated with ray r ;
  if *exists real vertex w in RVD*$(\{\mathcal{L}.\text{PREV}(r), r, \mathcal{L}.\text{NEXT}(r)\})$ **then**
    $\mathcal{Q}$.PUSH$((d_\angle(r, w), r, w))$;
return *PRVD*$(\mathcal{R}_\mathcal{P})$;

Proposition 4.12. *Algorithm 1 constructs PRVD*$(\mathcal{R}_\mathcal{P})$ *in* $\Theta(n \log n)$ *time.*

Proof. The time complexity of the algorithm is straightforward. The algorithm take $O(n \log n)$ time to sort the first n events, (lines 3-5). Then, there are $n-2$ events and each event takes $O(\log n)$ time (lines 6-12), so the algorithm takes $O(n \log n)$ time overall.

Following we prove the correctness of the algorithm. The main idea is to construct the features of the diagram (vertices and edges) in increasing distance to their nearest site. First note, that because of Lemma 4.11, the set of points with same distance to their closest site form a single cycle. So to keep track of the edges which are *"under construction"* it suffices to use a circular list.

Next we argue that the algorithm correctly finds the next vertex, assuming that all vertices with smaller distance have already been constructed. Corollary 4.10 implies that every vertex has at least 2 edges with increasing distance towards it, Thus, the next vertex is the intersection of a pair of edges, which are under construction. Further the 2 involved edges need to be consecutive, because the cyclic property of Lemma 4.11 would be violated otherwise. Since we are constructing the diagram in increasing distance, the algorithm picks the candidate vertex with smallest distance.

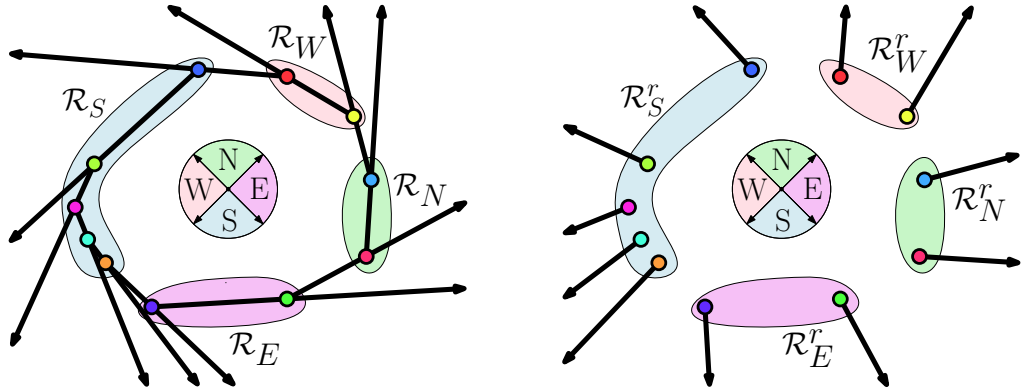
At each vertex event, the algorithm starts constructing a new edge. The algorithm does not miss any new edges because edges can only start at vertices. This is due to the distance function on an edge being monotone and not exhibiting local minima; see Remark 4.2. □

4.3.3 An optimal $O(n)$-time algorithm

We now show how PRVD($\mathcal{R}_\mathcal{P}$) can be constructed in optimal deterministic $\Theta(n)$-time. As we have already seen in Section 4.3.1, PRVD($\mathcal{R}_\mathcal{P}$) has a tree structure and each Voronoi region is connected. This suggest a possible connection to the abstract Voronoi diagram framework. As we have already discussed in Section 2.1.2, if a Voronoi diagram satisfies the AVD axioms, then this framework provides several results, both combinatorial and algorithmic.

Unfortunately, as we will see in the next section, this framework is not directly applicable to PRVD($\mathcal{R}_\mathcal{P}$). Still motivated by this, the key ingredient of our algorithm is to appropriately split the problem into sub-problems that admit Voronoi diagrams with a simpler structure, which can satisfy the AVD axioms.

Algorithm outline. Refer also to the accompanying pseudocode of Algorithm 2. In a first step, we partition $\mathcal{R}_\mathcal{P}$ into four sets $\mathcal{R}_N, \mathcal{R}_W, \mathcal{R}_S$ and $\mathcal{R}_E$ of consecutive rays, depending on whether a ray points to the (N)orth, (W)est, (S)outh or (E)ast respectively; see an example in Fig. 4.16a. In a second step we transform each set



(a) The sets of rays $\mathcal{R}_N, \mathcal{R}_W, \mathcal{R}_S, \mathcal{R}_E$. (b) The sets of rays $\mathcal{R}^r_N, \mathcal{R}^r_W, \mathcal{R}^r_S, \mathcal{R}^r_E$.

Figure 4.16. The partitioning of the set of rays in $\mathcal{R}_\mathcal{P}$ before and after rotation.

$\mathcal{R}_d$, $d \in \{\text{N,W,S,E}\}$ into a set $\mathcal{R}^r_d$ in which every ray of $\mathcal{R}_d$ is rotated clockwise by an angle of $\pi/2$ (or equivalently rotated by $-\pi/2$); see the example of Fig. 4.16b. Then, we construct each diagram $\text{RVD}(\mathcal{R}^r_d)$ independently; see Fig. 4.17 Finally, in the last phase we merge the four diagrams to obtain $\text{PRVD}(\mathcal{R}_\mathcal{P})$. This is done in two phases, we first merge $\text{RVD}(\mathcal{R}^r_W)$ with $\text{RVD}(\mathcal{R}^r_S)$ and $\text{RVD}(\mathcal{R}^r_N)$ with $\text{RVD}(\mathcal{R}^r_E)$; see Fig. 4.20. Then, we merge $\text{RVD}(\mathcal{R}^r_W \cup \mathcal{R}^r_S)$ with $\text{RVD}(\mathcal{R}^r_N \cup \mathcal{R}^r_E)$ restricted to $\mathcal{P}$; see Fig. 4.21.

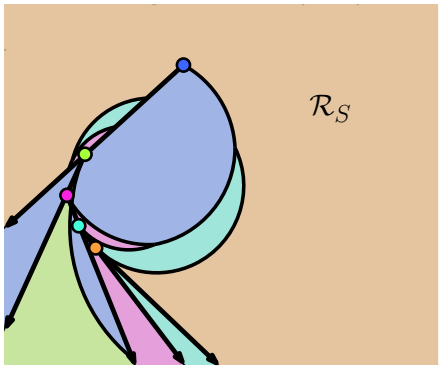
Algorithm 2: $O(n)$-time algorithm to construct $\text{PRVD}(\mathcal{R}_\mathcal{P})$.

Input : A convex polygon $\mathcal{P}$ with $n \geqslant 3$ vertices.
Output: The diagram $\text{PRVD}(\mathcal{R}_\mathcal{P})$.

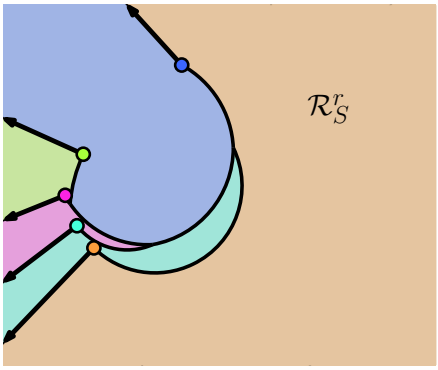
1 $\{\mathcal{R}_N, \mathcal{R}_W, \mathcal{R}_S, \mathcal{R}_E\} \leftarrow$ **Split** $\mathcal{R}_\mathcal{P}$;
2 **for** each $d \in \{N, W, S, E\}$ **do**
3 $\quad \mathcal{R}^r_d \leftarrow$ **Rotate** $\mathcal{R}_d$;
4 $\quad$ **Construct** $\text{RVD}(\mathcal{R}^r_d)$;
5 $\text{RVD}(\mathcal{R}^r_W \cup \mathcal{R}^r_S) \leftarrow$ **Merge** $\text{RVD}(\mathcal{R}^r_W)$ and $\text{RVD}(\mathcal{R}^r_S)$;
6 $\text{RVD}(\mathcal{R}^r_N \cup \mathcal{R}^r_E) \leftarrow$ **Merge** $\text{RVD}(\mathcal{R}^r_N)$ and $\text{RVD}(\mathcal{R}^r_E)$;
7 $\text{PRVD}(\mathcal{R}_\mathcal{P}) \leftarrow$ **Merge** $\text{RVD}(\mathcal{R}^r_W \cup \mathcal{R}^r_S)$ and $\text{RVD}(\mathcal{R}^r_N \cup \mathcal{R}^r_E)$;
8 **return** $PRVD(\mathcal{R}_\mathcal{P})$;

We describe in detail the construction of the 4 smaller diagrams in Section 4.3.4 and the merging phase in Section 4.3.5. Consequently we will show the following, which is the main results of this section.

Theorem 4.5. *Given a convex polygon $\mathcal{P}$, we can construct $PRVD(\mathcal{R}_\mathcal{P})$ in optimal deterministic $\Theta(n)$ time.*



(a) Diagram RVD($\mathcal{R}_S$).



(b) Diagram RVD($\mathcal{R}_S^r$).

Figure 4.17. Voronoi diagrams of a set $\mathcal{R}_S$ and the rotated set $\mathcal{R}_S^r$.

4.3.4 $O(n)$-time algorithm: Constructing the 4 partial diagrams

In this section, we describe how to construct for each of the 4 subsets of rays $\mathcal{R}_d^r$, the diagram RVD($\mathcal{R}_d^r$) in $\Theta(|\mathcal{R}_d^r|)$ time. To do so, we use the abstract Voronoi diagrams framework. We will show that the system of bisectors of $\mathcal{R}_d^r$ satisfies the AVD axioms. For a set $\mathcal{R}$ of rays, the AVD axioms can be stated as follows.

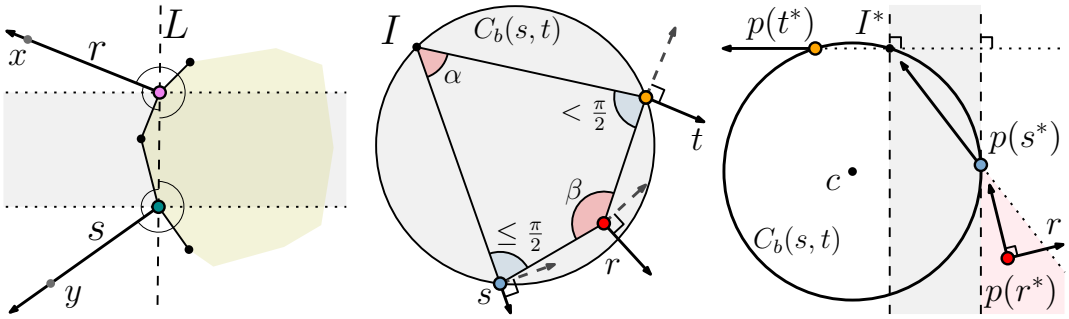
(A1) The bisector $b_\angle(r,s)$, $\forall r,s \in \mathcal{R}$, is an unbounded Jordan curve.
(A2) The region $r_\angle reg(r)$ in RVD($\mathcal{R}'$), $\forall \mathcal{R}' \subseteq \mathcal{R}$ and $\forall r \in \mathcal{R}'$, is connected.
(A3) The closure of the union of all regions in RVD($\mathcal{R}'$), $\forall \mathcal{R}' \subseteq \mathcal{R}$, covers $\mathbb{R}^2$.

Observe that, in the way we partitioned $\mathcal{R}_p$, any two rays r and s on the same subset have an angular difference of at most $\pi/2$, i.e., $\min\{\text{diff}_\angle(r,s), \text{diff}_\angle(s,r)\} \leqslant \pi/2$ holds. This is a key property in proving the following statement.

Lemma 4.13. *The system of bisectors of $\mathcal{R}_d^r$ satisfies the AVD axioms.*

Proof. We prove each of the three axioms separately.

Axiom (A1): Let r,s be a pair of rays in $\mathcal{R}_d^r$, let $x \in r\backslash\{p(r)\}$, resp. $y \in s\backslash\{p(s)\}$, denote a point lying on r, resp. s, and let L be the line passing through $p(r)$ and $p(s)$; see Fig. 4.18a. Due to the convexity of the polygon, it follows that $\angle(x,p(r),p(s)$ and $\angle(p(s),p(r),x)$ are greater than or equal to $\pi/2$; hence r lies in the closed halfplane orthogonal to L incident to $p(r)$ which does not contain $p(s)$. Analogously s, lies in the in the closed halfplane orthogonal to L incident to $p(s)$ that does not contain $p(r)$. Thus, the horizontal strip defined by the two halfplanes separates r and s, and they are non-intersecting. By Lemma 4.1, the bisector of two non-intersecting rays is an unbounded Jordan curve.



(a) (A1): The horizontal strip (shown shaded) separates r and s.

(b) (A2): $p(r)$ is between $p(s)$ and $p(t)$, so it lies in $C_b(s,t)$.

(c) (A2): The vertical strip (shown shaded) separates c and r.

Figure 4.18. Illustrations for the proof of Lemma 4.13.

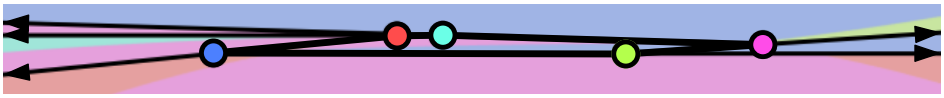
Axiom (A2): It suffices to prove the property for any subset of $\mathcal{R}_d^r$ of size three (see Klein et al. [2009]). By Lemma 4.3, each Voronoi region has exactly one unbounded face, so if a region is disconnected, then it has at least one bounded face. Observe that the diagram of 3 rays can have at most one proper vertex. Thus, a bounded face in the diagram can appear only incident to a ray, and it suffices to show that no ray intersects twice the bisecting circle of the other two rays.

Let $\{r,s,t\}$ be a subset of $\mathcal{R}_d^r$. We will show that an arbitrary ray of $\{r,s,t\}$, say r, does not intersect twice $C_b(s,t)$. Without loss of generality we assume that $\text{diff}_\angle(t,s) < \text{diff}_\angle(s,t)$.

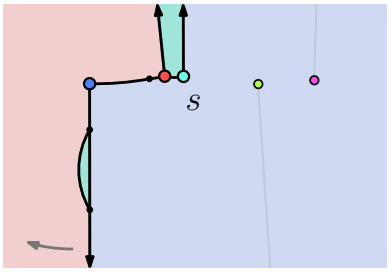
In the case that $p(r)$ lies in the interior of $C_b(s,t)$, then r intersects $C_b(s,t)$ exactly once and the claim follows. Assume that $p(r)$ appears between $p(s)$ and $p(t)$ along the polygon chain, and let I be the intersection of the supporting lines $l(s)$ and $l(t)$; see Fig. 4.18b. Because of the rotation of rays, we have $\angle(p(r),p(s),I) \leqslant \pi/2$ and $\angle(I,p(t),p(r)) < \pi/2$. Thus, by the properties of cyclic quadrilaterals, the $p(r)$ lies in the interior of $C_b(s,t)$, the circle passing through $p(s)$, $p(t)$ and I; see, e.g., that $\alpha+\beta > \pi$ in Fig. 4.18b.

It remains to examine the case that $p(r)$ lies outside $C_b(s,t)$, and $p(r)$ appears before or after both $p(s)$ and $p(t)$. We prove the case when $p(r)$ appears before $p(s)$ and $p(t)$. The other case is analogous.

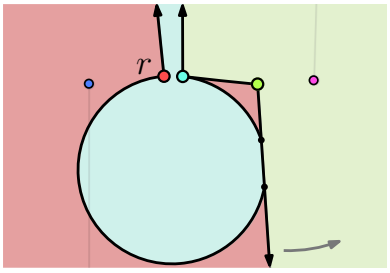
Let c denote the center of $C_b(s,t)$, let r^* (resp. s^*, t^*) denote the ray r (resp. s, t) rotated counterclockwise by $\pi/2$ around its apex, and let I^* denote the intersection point between the supporting lines $l(s^*)$ and $l(t^*)$. Without loss of generality, we assume that t^* is a horizontal ray pointing left. Refer also to Fig. 4.18c. First note that, if r intersects twice $C_b(s,t)$, then c lies to the right



(a) The Voronoi diagram of the complete set of rays without rotation.



(b) A subset of 3 rays rotated by $-\pi/2$. The region of s (●) has 2 faces. To be connected, more rotation is needed.



(c) A subset of 3 rays rotated by $-\pi/2$. The region of r (●) has 2 faces. To be connected, less rotation is needed.

Figure 4.19. An example of a (thin) polygon with 5 rays justifying Remark 4.15.

of the (directed) line $l(r^*)$, so it suffices to prove that c lies to the left of $l(r^*)$. To show this observe that $p(r^*)$ lies to the right of the vertical line through $p(s^*)$ since $\mathrm{diff}_{\angle}(t^*, r^*) \leqslant \pi/2$, and to the left of $l(s^*)$ since r^* and s^* are induced by a convex polygon; see the red region in Fig. 4.18c. On the other hand, note that the vertical line through I^* separates C from $p(r^*)$. Hence, c lies to the left of r^*, which proves the claim and concludes the proof.

Axiom (A3): The diagram $\mathrm{RVD}(\mathcal{R}_d^r)$ is defined by distance functions, one for each site in $\mathcal{R}_d^r$, whose domain is the entire plane. Hence, any point in the plane belongs to the closure of a region of $\mathrm{RVD}(\mathcal{R}_d^r)$. □

Since each Voronoi region is connected, and it has exactly one unbounded face, as shown in Lemma 4.3, we can easily infer the following.

Corollary 4.14. *$\mathrm{RVD}(\mathcal{R}_d^r)$ is a tree of $\Theta(\mathcal{R}_d^r)$ complexity.*

It is worth noting that the original set $\mathcal{R}_\mathcal{P}$ need not satisfy the AVD axioms; hence the necessity for partitioning into the four subsets and rotating. An example of a diagram that satisfies axiom (A2), only after the clockwise $\pi/2$ rotation is shown in Fig. 4.17.

The intuition behind the rotation of the rays comes from the fact that only circular parts of bisectors appear in $\mathrm{PRVD}(\mathcal{R}_\mathcal{P})$, and that the bisecting circles remain the same under a uniform rotation. A clockwise $\pi/2$-rotation by itself is not sufficient and this can be justified by the example in Fig. 4.19: a set $\mathcal{R}_\mathcal{P}$ of 5 rays is given, and there exists a subset of rays which need more rotation in order

to have each region connected (see Fig. 4.19b), and a subset of rays which needs less rotation (Fig. 4.19c).

Remark 4.15. *There are sets of rays $\mathcal{R}_\mathcal{P}$ for which there exists no unique angle to rotate the rays in $\mathcal{R}_\mathcal{P}$, so that axiom (A2) is satisfied.*

The results of Klein [1989] on abstract Voronoi diagrams, directly imply an $O(n \log n)$-time algorithm to construct RVD$(\mathcal{R}_d^r)$. We can further improve upon this time complexity, to $O(n)$-time, by showing that the system of bisectors of $\mathcal{R}_d^r$ falls under the, more restricted, *Hamiltonian abstract Voronoi diagram* framework of Klein and Lingas [1994]. In addition to satisfying (A1)-(A3), the following axiom should also be satisfied.

(A4) There exists a Jordan curve $\mathcal{H}$ of constant complexity such that, $\mathcal{H}$ visits the region $r_\angle reg(r)$ in RVD$(\mathcal{R}')$, $\forall \mathcal{R}' \subseteq \mathcal{R}$ and $\forall r \in \mathcal{R}'$, exactly once.

If a Voronoi diagram satisfies axioms (A1)-(A4) and the ordering of the regions of RVD$(\mathcal{R}')$ along $\mathcal{H}$ is given, then RVD$(\mathcal{R})$ can be computed in deterministic $\Theta(n)$-time. Hence for our problem, it suffices to prove that we can find a curve $\mathcal{H}$ satisfying these properties. We show this in the following.

Lemma 4.16. *RVD$(\mathcal{R}_d^r)$ can be constructed in deterministic $\Theta(|\mathcal{R}_d|)$ time.*

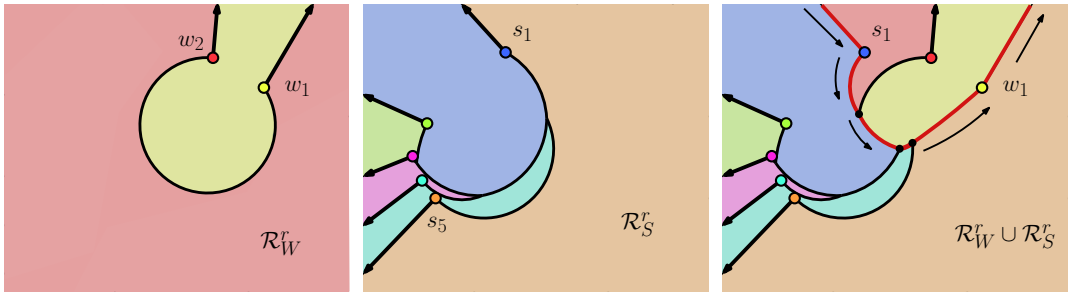
Proof. We show that there exists a curve $\mathcal{H}$ satisfying axiom (A4), and the ordering of the regions of RVD$(\mathcal{R}')$ along $\mathcal{H}$ is known, for every $\mathcal{R}' \subseteq \mathcal{R}_d^r$. Then, the deterministic linear time algorithm directly follows the existing results of Klein and Lingas [1994].

Let $\mathcal{H}$ be a circle of sufficient large radius, such that all bisecting circles lie entirely in the interior of $\mathcal{H}$. $\mathcal{H}$ is obviously a simple and closed curve of constant complexity. For any $\mathcal{R}' \subseteq \mathcal{R}_d^r$, the diagram RVD$(\mathcal{R}')$ is a tree, so it has only unbounded faces. By definition, $\mathcal{H}$ does not intersect any bisecting circle, hence $\mathcal{H}$ visits each region of RVD$(\mathcal{R}')$ exactly once, with a change on the visited region happening when $\mathcal{H}$ intersects a ray.

The ordering of the unbounded faces of RVD$(\mathcal{R}_d^r)$ corresponds to the ordering of the respective vertices along the polygon $\mathcal{P}$, and this is maintained for any $\mathcal{R}' \subset \mathcal{R}_d^r$. The ordering of the vertices of $\mathcal{P}$ is given, so this concludes the proof. □

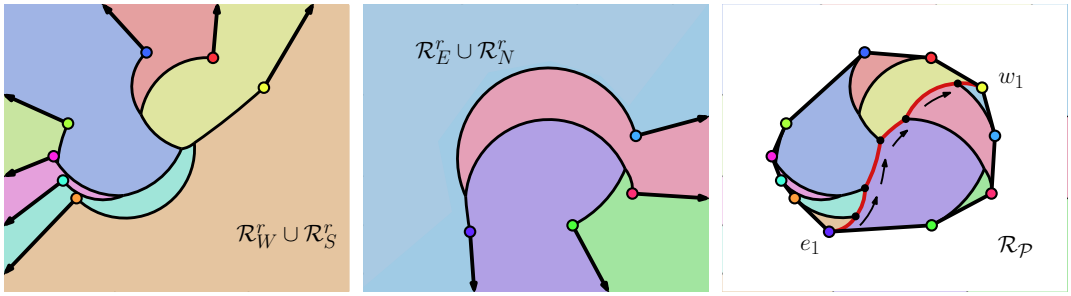
4.3.5 $O(n)$-time algorithm: Merging the 4 partial diagrams

We now merge all diagrams to obtain PRVD$(\mathcal{R}_\mathcal{P})$. Our merging procedure consists of two steps. In an initial step we merge RVD$(\mathcal{R}_W^r)$ with RVD$(\mathcal{R}_S^r)$ to obtain



(a) RVD($\mathcal{R}^r_W$). (b) RVD($\mathcal{R}^r_S$). (c) RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$).

Figure 4.20. 1st merging phase: merging RVD($\mathcal{R}^r_W$) and RVD($\mathcal{R}^r_S$). The red edges correspond to the merge curve, and the arrows schematize tracing.



(a) RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$). (b) RVD($\mathcal{R}^r_E \cup \mathcal{R}^r_N$). (c) PRVD($\mathcal{R}_\mathcal{P}$).

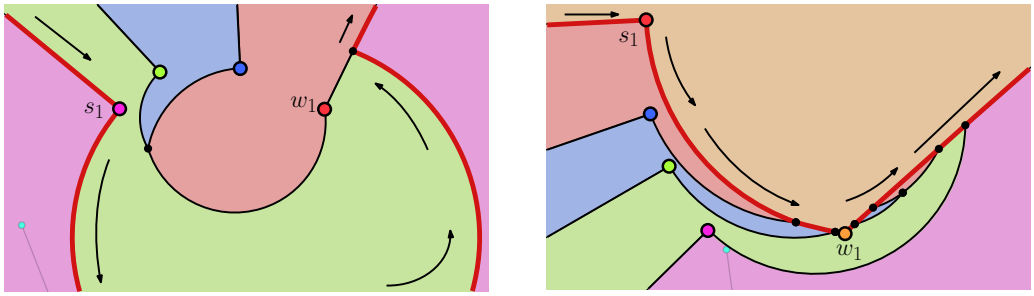
Figure 4.21. 2nd merging phase: merging RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$) and RVD($\mathcal{R}^r_E \cup \mathcal{R}^r_N$) restricted to $\mathcal{P}$.

RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$), and respectively we obtain RVD($\mathcal{R}^r_E \cup \mathcal{R}^r_N$); see Fig. 4.20. Then, in a final step we merge the diagrams RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$) and RVD($\mathcal{R}^r_E \cup \mathcal{R}^r_N$), restricted to the interior of $\mathcal{P}$, to obtain PRVD($\mathcal{R}_\mathcal{P}$); see Fig. 4.21.

We first describe the merging procedure briefly, and then give more details and prove its correctness. We will ultimately show the following.

Lemma 4.17. *Given RVD($\mathcal{R}^r_d$) for all $d \in$ {N,W,S,E}, we can merge the 4 diagrams to obtain PRVD($\mathcal{R}_\mathcal{P}$) in $\Theta(n)$ time.*

Outline of the merging phase. Regarding the first merging step, we describe how to merge RVD($\mathcal{R}^r_W$) with RVD($\mathcal{R}^r_S$). Merging RVD($\mathcal{R}^r_E$) with RVD($\mathcal{R}^r_N$) is done analogously. Let $w_1, \ldots, w_k$ be the rays in $\mathcal{R}^r_W$ and let $s_1, \ldots, s_l$ be the rays in $\mathcal{R}^r_S$, as they appear ordered on $\mathcal{P}$. We need to construct the *merge curve* of the two diagrams, which partitions $\mathbb{R}^2$ in two, and keep from one side the diagram



(a) The curve E_C does not end at $p(w_1)$. (b) Ray w_1 intersects RVD($\mathcal{R}_S^r$).

Figure 4.22. Special cases of merging two diagrams RVD($\mathcal{R}_W^r$) and RVD($\mathcal{R}_S^r$).

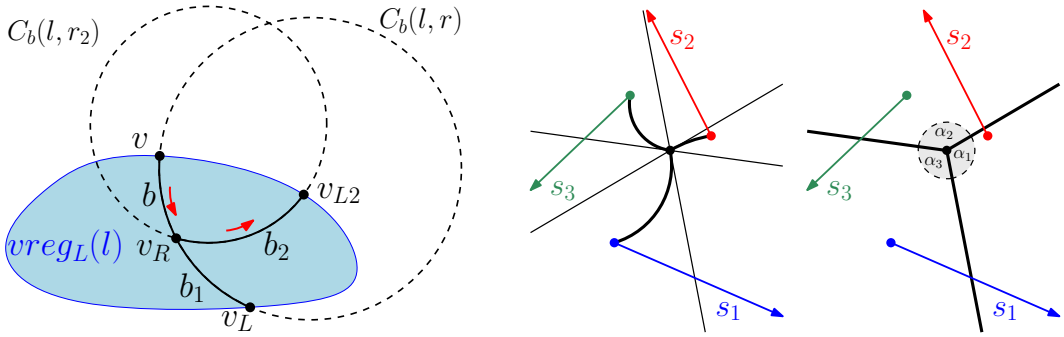
RVD($\mathcal{R}_W^r$) and from the other side we keep RVD($\mathcal{R}_S^r$); see Fig. 4.20c. The merge curve consists of the two rays s_1 and w_1, and the set of circular edges of RVD($\mathcal{R}_W^r \cup \mathcal{R}_S^r$) equidistant to sites $w \in \mathcal{R}_W^r$ and $s \in \mathcal{R}_S^r$, which forms a single connected chain bounded by $p(s_1)$ and $p(w_1)$. We denote the set of circular edges by E_C.

Then, we merge RVD($\mathcal{R}_W^r \cup \mathcal{R}_S^r$) with RVD($\mathcal{R}_E^r \cup \mathcal{R}_N^r$) restricted into $\mathcal{P}$. In this case, since we are merging restricted into $\mathcal{P}$, the merge curve is simpler and it consists only of the set E_C, which is again a single connected chain, bounded by $p(e_1)$ and $p(w_1)$; see Fig. 4.21c.

Tracing along the rays (initial step). Tracing along the ray s_1 can be done easily as the ray lies entirely in $r_\angle reg(w_k)$. To see that, consider the set $\mathcal{R}_W$ (before rotation) and continuously clockwise rotate all rays by an angle of $\pi/2$. During this process, w_k does not intersect any of the rays in $\mathcal{R}_W$, hence $s_1 \in r_\angle reg(w_k)$. Thus, s_1 does not intersect RVD($\mathcal{R}_W^r$) and it can be trivially traced in $\Theta(1)$ time, as there is no vertex to identify.

Tracing along the ray w_1 is done in a different way. In contrast to s_1, the ray w_1 may intersect many circular edges of RVD($\mathcal{R}_S^r$), each inducing a vertex on w_1; see e.g., Fig. 4.22b. To identify such vertices, we intersect w_1 with RVD($\mathcal{R}_S^r$). This can be easily done in $O(|\mathcal{R}_S|)$ time, as RVD($\mathcal{R}_S^r$) is proved to be a tree; see Corollary 4.14. Note that, the curve E_C might not intersect w_1 at $p(w_1)$ but at some other point; see e.g., Fig. 4.22a. In this case the aforementioned search for intersections RVD($\mathcal{R}_S^r$) should start from that point.

Tracing the sequence of circular edges E_C. We describe how to trace E_C in $\Theta(|E_C|)$, for both merging steps. The procedure is similar to the standard Voronoi diagrams of points (see e.g., Aurenhammer et al. [2013]), adapted to angular bisectors. Following, to show that E_C can be traced in $\Theta(|E_C|)$ time, we need to



(a) Tracing the merge curve (following the arrows) in a region $r_{\angle}reg_L(l)$ (shown shaded).

(b) Tangents to the circular edges incident to a proper vertex, and the respective curvilinear angles $(\alpha_1, \alpha_2, \alpha_3)$.

Figure 4.23. Illustration of the tracing process while merging E_C and the proof that it is not necessary to backtrack.

prove that there is no need to *backtrack*, i.e., that we do not have to scan parts of a Voronoi region more that once; we prove this in Lemma 4.18. Following, we give a detail description of the tracing procedure.

Assume we want to merge two diagrams along one merge curve. Let L be the set of rays defining the diagram to the left of the curve E_C and R the set to the right hand side. Initially a starting point for the tracing needs to be found, which is an apex, as already described.

Without loss of generality assume that we are tracing E_C from top to bottom; refer to Fig. 4.23a. Suppose that the current edge b of E_C has just entered the region $r_{\angle}reg_L(l)$, the Voronoi region of site $l \in L$ within RVD(L), at point v. Let r be the site of the right diagram, in whose region $r_{\angle}reg_R(r)$ the edge b lies. We determine the points v_L (resp. v_R) where b leaves the region $r_{\angle}reg_L(l)$ (resp. $r_{\angle}reg_R(r)$). The point v_L is found by scanning the boundary of $r_{\angle}reg_L(l)$ clockwise starting from v. The point v_R is found by scanning the boundary of $r_{\angle}reg_R(r)$ counterclockwise starting from v. Without loss of generality assume that vertex v_R is reached first, which then describes the endpoint of edge b.

E_C continues from v_R with another edge b_2 along the bisector $b_{\angle}(l, r_2)$, where r_2 is another site in R. We again determine the points v_{L2} and v_{R2}, where the edge b_2 leaves the regions of l and r_2. We will prove in Lemma 4.18 that v_{L2} cannot be on the boundary of $r_{\angle}reg_L(l)$, which was already scanned, i.e., we can start scanning the boundary starting from v_L in clockwise direction. Since there is no backtracking required to trace E_C, the tracing takes $\Theta(|E_C|)$ time.

Lemma 4.18. *There is no need to backtrack while tracing the curve E_C.*

Proof. Let v, v_L and v_{L2} be the points as defined in the above description of tracing E_C. To prove that there is no need for backtracking, it suffices to prove that the points v, v_L and v_{L2} appear in counterclockwise order along the boundary of face $r_{\angle}reg_L(l)$; see Fig. 4.23a.

The *curvilinear angle* between two intersecting curves is the angle between their two tangents at the point of intersection. Each proper vertex of the diagram, is of degree 3, so, the three edges incident to a vertex induce three curvilinear angles; see Fig. 4.23b. Each such curvilinear angle can be seen as the angle at the intersection of two halfplanes. Hence, each such angle is less than π.

The point v_R is a proper vertex in the merged diagram. Therefore, the curvilinear angle $\angle(v_{L2}, v_R, v)$ between the edges b_2 and b is less than π. On the other hand, the angle $\angle(v_L, v_R, v)$ between the edges b_1 and b_2 is exactly π as both edges lie on the same bisector. Within the polygon, two related bisectors intersect at most once. Thus, the edge b_2 has to hit the boundary of $r_{\angle}reg_L(l)$ after v_L but before v in counterclockwise order. □

Correctness of the construction of E_C. We first argue why in the first merging phase (assuming that we merge RVD($\mathcal{R}^r_W$) with RVD($\mathcal{R}^r_S$), E_C which starts at $p(s_1)$ ends at ray w_1. To see that, consider the distance of any point on the chain to its nearest ray. The distance when E_C, at $p(s_1)$, is exactly $\pi/2$, and it is monotonically increasing. Moreover, consider the polygonal chain P^* consisting of the line segments $\overline{p(w_1)p(w_2)}$, $\overline{p(w_2)p(w_3)}$, …, $\overline{p(w_k)p(s_1)}$, $\overline{p(s_1)p(s_2)}$, …, $\overline{p(s_{l-1})p(s_l)}$ and the ray s_l. The distance of any point on P^* to its nearest ray, is exactly $\pi/2$. Hence, since the distance along the chain of circular edges is increasing, the only possibility for this chain to end up is at w_1 (but not necessarily at its apex, see Fig. 4.22a). For the second merging phase it is obvious that E_C is bounded by $p(e_1)$ and $p(w_1)$.

We need to show that (i) the chain which we traced is the complete set E_C, i.e., there are no other connected components which we have to identify, and that (ii) E_C does not induce any bounded faces in RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$). To prove both statements we use the disk diagram.

(i) By Lemma 4.3, each region has one unbounded face, so if there exists another connected component in E_C, it has to be bounded. Suppose that E_C has another connected component which is bounded only by circular edges. This implies that the bisecting circles of the bisectors create bounded faces. Since the bisecting circles of the disk diagrams are supersets of the circular arcs appearing in RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$), RVD($\mathcal{R}^r_E \cup \mathcal{R}^r_N$) and PRVD($\mathcal{R}_\mathcal{P}$), such a bounded face would also appear in the respective disk diagrams, a contradiction to Lemma 4.7.

(ii) Suppose now that the merge curve has a component which is bounded from one side by a ray. In the final merging step this is clearly not possible, as on PRVD($\mathcal{R}_\mathcal{P}$) all points on an edge/ray of $\mathcal{P}$ belong to the region of the respective ray. For the initial merging step, assume w.l.o.g. that we are merging RVD($\mathcal{R}^r_W$) with RVD($\mathcal{R}^r_S$). Apart from the ray w_1, for every other ray, say w.l.o.g. s_i, the complete right side of the ray is incident to $r_\angle reg(s_{i-1})$. This can be proved with the same argument, used to show that $s_1 \in r_\angle reg(w_k)$ (where s_1 is the first ray of $\mathcal{R}^r_S$, and w_k is the last ray of $\mathcal{R}^r_W$ respectively). As a result no bounded component of E_C could be be incident to a ray. Hence, E_C is a single unbounded chain.

Analogously, we can derive that while tracing E_C no other bounded faces are created, as these bounded faces should also appear in the respective disk diagrams. As a result in the initial merged diagram RVD($\mathcal{R}^r_W \cup \mathcal{R}^r_S$), each ray $s_i \in \mathcal{R}^r_S$ can have at most two faces (the unbounded one and the one incident to w_1) and $r_\angle reg(w_i)$ is connected $\forall w_i \in \mathcal{R}^r_W$. On the contrary in PRVD($\mathcal{R}_\mathcal{P}$), $r_\angle reg(r_i)$ is connected $\forall r_i \in \mathcal{R}_\mathcal{P}$.

Overall time complexity. In the first step, tracing the rays s_1 and n_1 takes $\Theta(1)$ time, and tracing the rays w_1 and e_1, takes $\Theta(|\mathcal{R}_S|)$ time and $\Theta(|\mathcal{R}_N|)$ time respectively. Tracing the curve E_C takes $O(|\mathcal{R}_W| + |\mathcal{R}_S|)$ time, and $O(|\mathcal{R}_E| + |\mathcal{R}_N|)$ time respectively, so in total the first step requires $O(n)$ time. The final step requires $O(n)$ time to trace E_C and $\Theta(n)$ to restrict the diagram into $\mathcal{P}$. So, the overall merging of the four diagrams takes $\Theta(n)$ time.

Putting everything together, we can trivially split, $\mathcal{R}_\mathcal{P}$ into 4 sets in $\Theta(n)$ time, we can construct the 4 diagrams in $\Theta(n)$ time, as shown in Lemma 4.16, and we can merge them in $\Theta(n)$ time, as shown in Lemma 4.17. So, we can summarize (and re-state) the main result of this section as follows.

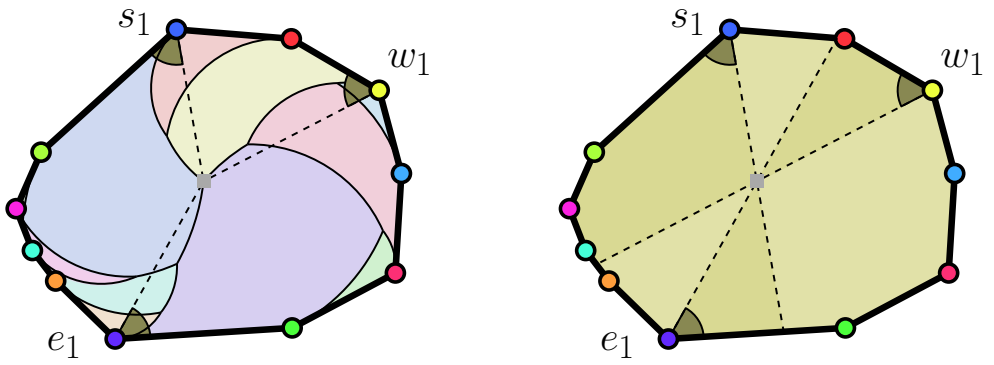
Theorem 4.5. *Given a convex polygon $\mathcal{P}$, we can construct PRVD($\mathcal{R}_\mathcal{P}$) in optimal deterministic $\Theta(n)$ time.*

4.3.6 Brocard illumination of a convex polygon

We now turn to the Brocard illumination problem of a convex polygon $\mathcal{P}$. Our goal is to find the Brocard angle of $\mathcal{P}$, which is

$$\alpha^* = \max_{x \in \mathcal{P}} \min_{r \in \mathcal{R}} d_\angle(x, r).$$

Observe that diagram PRVD($\mathcal{R}_\mathcal{P}$) is a subset of RVD($\mathcal{R}_\mathcal{P}$), hence Proposition 4.6 applies also in this setting, and so α^* is realized on PRVD($\mathcal{R}_\mathcal{P}$). However, since



(a) PRVD($\mathcal{R}_\mathcal{P}$) and the rays realizing α^*. (b) The 3 α^*-floodlights illuminating $\mathcal{P}$.

Figure 4.24. The Brocard angle ($\measuredangle$) of a polygon $\mathcal{P}$, realized by (e_1, w_1, s_1).

the diagram is strictly confined into $\mathcal{P}$, the point x^* realizing the Brocard angle, can only lie on Voronoi vertex equidistant to 3 rays; see an example in Fig. 4.24a.

Similarly to the setting in $\mathbb{R}^2$, to find α^* we can first construct PRVD($\mathcal{R}_\mathcal{P}$) and then we can traverse it to find the Voronoi vertex of maximum distance. Both steps can be done in $O(n)$ time resulting in the following.

Theorem 4.6. *The Brocard angle of a convex polygon $\mathcal{P}$ can be found in $\Theta(n)$ time.*

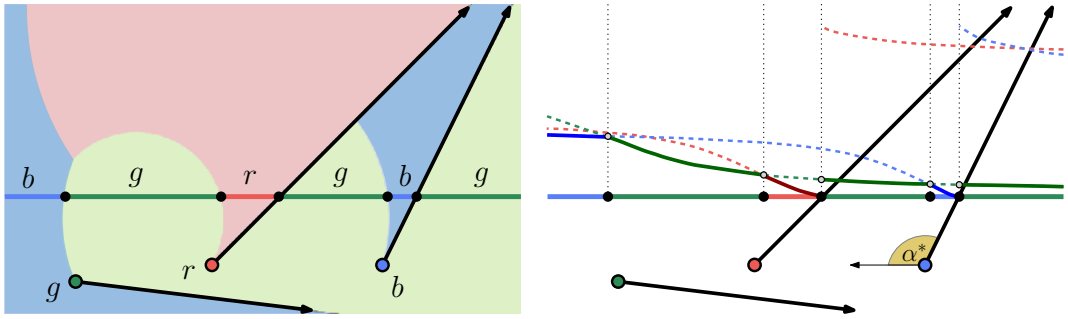
Following, we give tight bounds on the value of the Brocard angle.

Proposition 4.19. *The set of values that the Brocard angle of a convex polygon can achieve is* $[0, \pi/2 - \pi/n]$.

Proof. A $\pi/2 - \pi/n$ upper bound on the Brocard angle is given by Dmitriev and Dynkin [1946], as pointed out (and translated) by Besenyei [2015]. Such an angle is realized by regular polygons, which are Brocard polygons. The last illuminated point is the center of the polygon, which is simultaneously illuminated by all the floodlights at an angle of $\pi/2 - \pi/n$.

The lower bound is realized by a polygon whose bounding box has width w, height h, and an aspect ratio h/w arbitrarily close to zero, so that α^* is also arbitrarily close to zero. Further, while preserving convexity, we can smoothly transform a regular polygon into such an arbitrarily thin polygon. Hence it is possible to get any Brocard angle in the range $(0, \pi/2 - \pi/n]$. □

Floodlight illumination of a convex polygon by 3 floodlights. Note that the three floodlights which realize α^* suffice to illuminate $\mathcal{P}$. This can be observed in the illustrated example of Fig. 4.24b, and it implies the following.



(a) $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ as the intersection of $\mathrm{RVD}(\mathcal{R})$ in $\mathbb{R}^2$ with $\mathcal{C}$.

(b) $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ as the lower envelope (highlighted) of distance functions (dashed).

Figure 4.25. The curve $\mathcal{C}$ is the horizontal line $x_2 = 0$, and $\mathcal{R}$ is a set of 3 rays.

Remark 4.20. *A convex polygon $\mathcal{P}$ can be entirely illuminated be 3 α^*-floodlights.*

We conclude by discussing an implication of the results on the Brocard angle. Consider the following question posed by O'Rourke et al. [1995]: given a convex polygon $\mathcal{P}$ with n vertices, what is the minimum angle α_3, such that 3 vertex α_3-floodlights (not necessarily aligned with the edges) illuminate $\mathcal{P}$. Contreras et al. [1998a] gave an $\alpha_3 = \pi/6$ solution for $n = 3$ and an $\alpha_3 = \pi/4$ solution for $n = 4$. Urrutia [2000] gave an $\alpha_3 = \pi/3$ solution for arbitrary n.

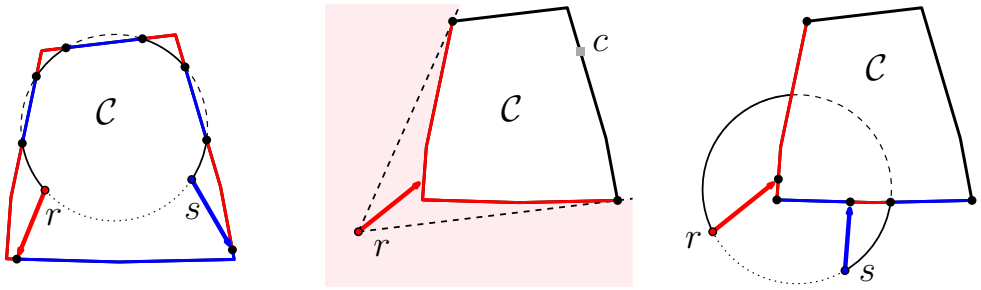
Our results directly imply an $\alpha_3 = \alpha^*$ solution for arbitrary n and, as proved in Proposition 4.19, $\alpha^* \leqslant \pi/2 - \pi/n$. Hence, our results subsume the aforementioned solutions for $n = 3, 4, 6$ and also improve the case of $n = 5$, to $\alpha_3 = 3\pi/10$.

4.4 Diagrams on curves

As we discussed in Section 2.3.1, floodlight illumination problems have also been considered restricted to curves. Motivated by such problems, in this section, we assume that $\mathcal{R}$ is a set of n rays in $\mathbb{R}^2$, and the domain of interest is a curve $\mathcal{C}$. We denote by $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ the rotating rays Voronoi diagram of $\mathcal{R}$ restricted to $\mathcal{C}$. We show how $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ can be viewed as the lower envelope of distance functions in 2-space; see Fig. 4.25. We first consider the curve $\mathcal{C}$ to be a line and then discuss how this approach can be extended to other curves.

Theorem 4.7. *Given a line $\mathcal{C}$, $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ has complexity $O(n2^{\alpha(n)})$ and it can be constructed in $O(n\alpha(n)\log n)$ time.*

Proof. Without loss of generality, let $\mathcal{C}$ be the horizontal line $x_2 = 0$. Each site $r \in \mathcal{R}$ induces a distance function in 2-space which maps a point $x = (x_1, 0) \in \mathcal{C}$



(a) Two rays inside $\mathcal{C}$. (b) The visible part of a ray. (c) Two rays outside $\mathcal{C}$.

Figure 4.26. A convex polygon $\mathcal{C}$; dominance regions along $\mathcal{C}$ are highlighted.

to point $x_{map} = (x_1, d_\angle(x,r))$; see Fig. 4.25b. Observe that if a ray r intersects $\mathcal{C}$ at point $(i,0)$, then there is a point of discontinuity, and the distance function is split into two partially defined functions, one with the domain up to i and one with the domain starting at i.

$\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ is the lower envelope of all these distance functions. The lower envelope of n partially defined functions, where each pair of functions intersects at most s times, has $O(\lambda_{s+2}(n))$ complexity (see Hart and Sharir [1986]) and it can be constructed in $O(\lambda_{s+1}(n)\log n)$ time (see Hershberger [1989]), where $\lambda_s(n)$ is the length of the longest (n,s) *Davenport-Schinzel sequence*.

Observe that the number of intersections of two distance functions is the same as the number of intersection of their bisecting circle with $\mathcal{C}$. In our case, a pair of functions intersects at most twice, as $\mathcal{C}$ may intersect twice the bisecting circle of the two respective rays, so $s = 2$. Further, we have at most $2n$ partially defined functions. Thus, $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ has complexity $O(n2^{\alpha(n)})$ and it can be constructed in $O(n\alpha(n)\log n)$ time, where $\alpha(n)$ is the inverse Ackermann function. □

Observe that the Brocard angle α^* needed to illuminate the line $\mathcal{C}$ is realized either at an intersection point between $\mathrm{RVD}(\mathcal{R})$, i.e., a vertex of $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$, or at a point of $\mathcal{C}$ at infinity; see for example the point $(-\infty, 0)$ in Fig. 4.25, which first illuminated by ray b. Hence, after constructing $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$, the angle α^* is revealed by a simple traversal of $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ in time linear in its size.

The aforementioned approach can be generalized to arbitrary curves in a straightforward way. Regarding the Brocard angle, note though, that if $\mathcal{C}$ is a bounded curve, α^* cannot be realized at infinity.

Let us first consider $\mathcal{C}$ to be a closed convex curve. Using a similar approach, and depending on whether we illuminate the interior or the exterior of $\mathcal{C}$, i.e., whether the apices of the rays inside or outside $\mathcal{C}$, we obtain the following results.

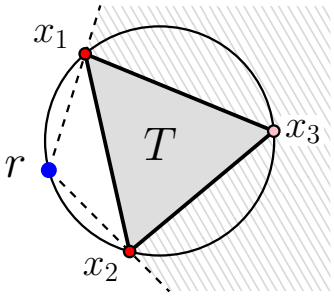


Figure 4.27. Illustration for the proof of Theorem 4.9. Point x_3 is not visbile from ray r. Triangle T is a subset of $\mathcal{C}$.

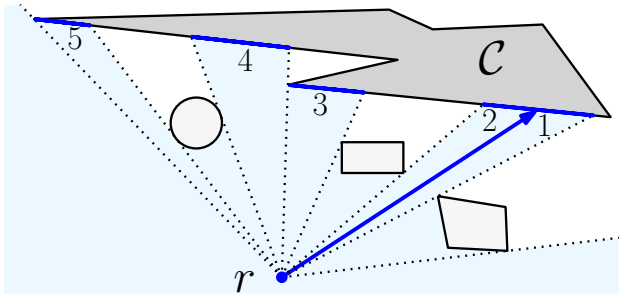


Figure 4.28. Illumination of $\mathcal{C}$ and the parameter k. The distance function of a ray r is split into 5 partial functions (k=5) by different types of breakpoints.

Theorem 4.8. *Let $\mathcal{C}$ be a closed convex curve, and let the apices of the rays in $\mathcal{R}$ lie inside $\mathcal{C}$. Then, $RVD_{\mathcal{C}}(\mathcal{R})$ has complexity $O(\lambda_{s+2}(n))$ and it can be constructed in $O(\lambda_{s+1}(n)\log n)$ time, where s is the maximum number of times $\mathcal{C}$ intersects a circle.*

Proof. Assume that the curve $\mathcal{C}$ is parametrized in the following form, $\mathcal{C} : [0,1] \to \mathbb{R}^2$ with $\mathcal{C}(0) = \mathcal{C}(1)$. Analogously to the approach of Theorem 4.7, each site r induces a distance function on the curve $\mathcal{C}$, which maps a value $t \in [0,1]$ to the point $(t, d_{\angle}(\mathcal{C}(t), r))$, and the result immediately follows the results of the envelopes of distance functions in 2-space. □

As a corollary of Theorem 4.8, if $\mathcal{C}$ is a circle, then $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ has complexity $O(n2^{\alpha(n)})$ and it can be constructed in $O(n\alpha(n)\log n)$ time, since $\mathcal{C}$ intersects a bisecting circle at most twice; hence, $s = 2$. Conversely, if $\mathcal{C}$ is an m-sided convex polygon, then $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ has complexity $O(\lambda_{2m+2}(n))$ and it can be constructed in $O(\lambda_{2m+1}(n)\log n)$ time. This is because $\mathcal{C}$ may intersect a bisecting circle $2m$ times, hence $s = 2m$; see, e.g., Fig. 4.26a.

We can obtain better results if the apices are lying outside a closed convex curve $\mathcal{C}$. In this case, only a part of the curve $\mathcal{C}$ is *visible* by each input ray, where a point $x \in \mathcal{C}$ is visible by a ray r if the open segment $\overline{p(r)x}$ does not intersect $\mathcal{C}$. If a point x is not visible by a ray $r \in \mathcal{R}$, we set $d_{\angle}(x,r) = +\infty$. For instance, in Fig. 4.26b, the visible portion of a ray r is shown; point c is not visible by r.

Theorem 4.9. *Let $\mathcal{C}$ be a closed convex curve, and let the apices of the rays in $\mathcal{R}$ lie outside $\mathcal{C}$. Then, $RVD_{\mathcal{C}}(\mathcal{R})$ with visibility restrictions has complexity $O(n2^{\alpha(n)})$ and it can be constructed in $O(n\alpha(n)\log n)$ time.*

Proof. We apply the same approach used as in Theorems 4.7 and 4.8. To get the

claimed results we show the part of the curve visible by any two rays is intersected at most twice ($s = 2$) by a bisecting circle.

Given a ray $r \in \mathcal{R}$ consider the portion of $\mathcal{C}$ visible by r. Suppose, for the sake of contradiction, that a bisecting circle defined by r intersects the visible part of $\mathcal{C}$ in at least 3 points x_1, x_2, x_3; refer also to Fig. 4.27. By definition, $p(r)$ lies on the bisecting circle, so $p(r)$ lies on one of the circular arcs $\overline{x_1x_2}$, $\overline{x_2x_3}$, or $\overline{x_3x_1}$; without loss of generality, suppose $p(r) \in \overline{x_1x_2}$. By the assumption $p(r)$ lies outside $\mathcal{C}$, so obviously points $x_1, x_2 \in C$ obstruct the visibility of r, and x_3 is not visible by r. But x_3 was the intersection point of the bisecting circle with the visible portion of $\mathcal{C}$, a contradiction.

The part of $\mathcal{C}$ visible by two rays is a subset of the part visible by each of the rays independently, so it is intersected by a bisecting circle at most twice. Hence, $s = 2$ and as in Theorems 4.7 and 4.8 the claim follows. □

Our approach can be generalized to other classes of curves as well, although the visibility restrictions should be taken into account. Given a ray r, the portion of $\mathcal{C}$ visible by r can be split into many maximal connected components; see Fig. 4.28. A split might be induced by the curve $\mathcal{C}$ itself (breakpoint (3-4) in Fig. 4.28), it can be induced by other curves (breakpoint (2-3) in Fig. 4.28), or it can be induced by the ray r itself (breakpoint (1-2) in Fig. 4.28).

If the part of $\mathcal{C}$ visible by a ray r_i is split in k_i connected components, this implies that the corresponding distance function of r is split into k_i partially defined functions. Let $K = \sum_{i \in n} k_i$ be the total number of partially defined functions, then from the results on lower envelopes of distance functions in 2-space, of Hart and Sharir [1986] and Hershberger [1989], we derive that $\mathrm{RVD}_{\mathcal{C}}(\mathcal{R})$ has complexity $O(\lambda_{s+2}(K))$ and it can be constructed in $O(\lambda_{s+1}(K) \log K)$ time.

4.5 Conclusion

In this chapter we presented our work related to the rotating rays Voronoi diagram and the Brocard illumination problem. Our motivation to study the rotating rays Voronoi diagram was because we showed that solving the Brocard illumination problem in different domains can be reduced to the construction of such a diagram. We briefly describe our main results.

We first considered the domain to be the entire plane, and we presented an $\Omega(n^2)$ worst-case lower bound on the combinatorial complexity. We also gave an $O(n^{2+\epsilon})$ upper bound which was complemented by a construction algorithm with the same time complexity. Using this Voronoi diagram we showed how the Brocard illumination problem in $\mathbb{R}^2$ can be solved in $O(n^{2+\epsilon})$ time.

Motivated by the Brocard illumination problem we considered convex polygonal domains; the diagram of a convex polygon is a tree of $\Theta(n)$ complexity. We presented a simple $\Theta(n \log n)$-time construction algorithm and we also gave an algorithm which takes optimal deterministic $\Theta(n)$ time. This is of particular interest as it enables us to also solve the Brocard illumination problem of a convex polygon in optimal $\Theta(n)$ time.

Finally, we considered the domain to be a curve, and we provided a generic approach to obtain combinatorial and algorithmic results for any given curve.

There are many interesting future directions related to both the rotating rays Voronoi diagram but also to the Brocard illumination problems. We discuss such future directions and open question in Section 6.2.

The implementation used to create many of the figures, is based on a simple *pixel-based* approach (developed by Marko Savić). So, we did not have to deal with possible algebraic issues stemming from the implementation of our algorithms. Despite being a Voronoi diagram with non-linear features (circular edges) we believe that an efficient exact implementation would be possible. This is because the bisectors of the Voronoi diagram consist only of rays and circles which are relatively simple objects, in contrast to other studied non-linear Voronoi diagrams.

As a final remark, parts of this chapter were the result of collaboration; in particular, Section 4.3 was joint work with Martin Suderland.

Chapter 5

Towards linear-time construction algorithms

This chapter presents our results on linear-time construction algorithms for planar Voronoi diagrams, and more specifically on a generalization of a combinatorial result, part of the algorithmic scheme of Aggarwal et al. [1989]. The chapter is based on the following publication:

> *K. Junginger, I. Mantas, and E. Papadopoulou.* On selecting a fraction of leaves with disjoint neighborhoods in a plane tree. *Discrete Applied Mathematics*. Elsevier, 2021.

In Section 5.1 we describe the algorithm of Aggarwal et al. [1989], and the basic definitions and notions which are useful for the rest of the chapter. In Section 5.2 we generalize the existence part of the original combinatorial result, and in Section 5.3 we generalize the algorithmic part of the original result. Section 5.4 concludes the chapter.

5.1 Preliminaries

In this chapter we consider planar trees embedded in $\mathbb{R}^2$. The trees we consider are *proper* without further mention, i.e., they contain no vertex of degree 2, but only of degree 1 or 3. Observe that each embedded tree induces a topological ordering of the leaves; see for example the tree in Fig. 5.2a and its topological ordering in Fig. 5.2c. We will be referring to leaves which are consecutive in the topological ordering simply as *(topologically) consecutive*.

We first overview the algorithm of Aggarwal et al. [1989] to compute the Voronoi diagram of points in convex position, given the ordering of these points

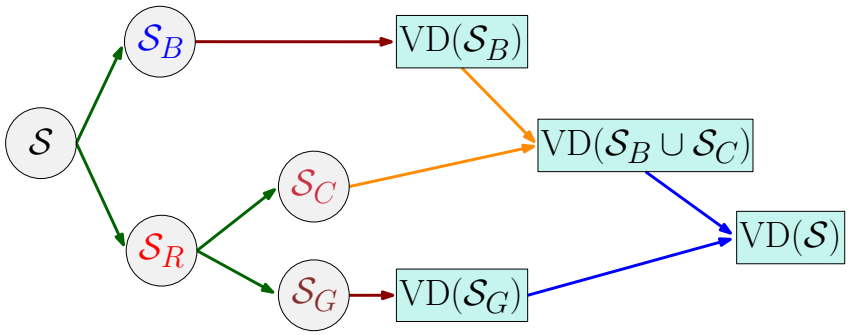


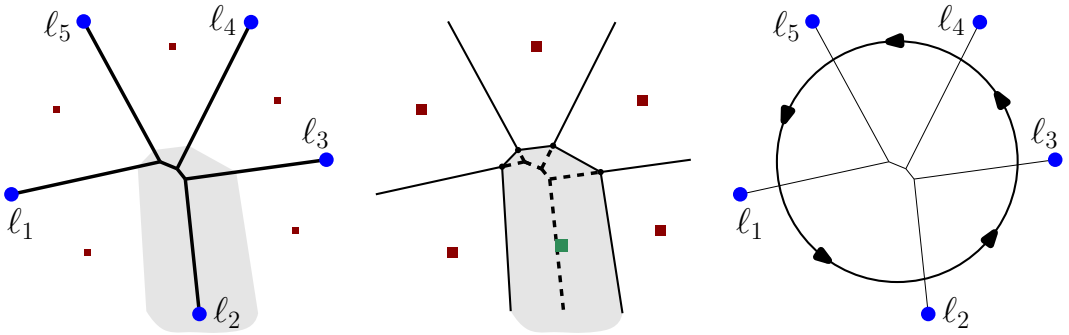
Figure 5.1. A sketch of the divide & conquer algorithm of Aggarwal et al. [1989]. (⟶) indicates the two different divide phases; (⟶) indicates the recursive constructions; (⟶) indicates the first "incremental" merge phase; (⟶) indicates the second "standard" merge phase.

along the boundary of the convex hull. Note that in this case the $\Omega(n \log n)$ does not apply and recall that such a diagram has a tree structure.

Algorithm of Aggarwal et al. [1989]. The algorithm takes as input a set of points $\mathcal{S}$, the ordering of the points of $\mathcal{S}$ along the boundary of the convex hull of $\mathcal{S}$, and returns the diagram $\mathrm{VD}(\mathcal{S})$. It is a recursive divide & conquer algorithm where each step is split in two distinct phases. An outline of the algorithm is also sketched in Fig. 5.1. It is briefly described as follows.

In an initial divide phase, the set $\mathcal{S}$ is split in two sets $\mathcal{S}_R$ (red) and $\mathcal{S}_B$ (blue) of roughly equal size, i.e., $|\mathcal{S}_R| = \Theta(|\mathcal{S}_B|)$, with the property that every two consecutive red sites in $\mathcal{S}_R$ have disjoint Voronoi regions in the Voronoi diagram $\mathrm{VD}(\mathcal{S})$. In a second divide phase, the set $\mathcal{S}_R$ is split further in sets $\mathcal{S}_C$ (crimson) and $\mathcal{S}_G$ (garnet), so that any two sites in $\mathcal{S}_C$ have pairwise disjoint Voronoi regions in the diagram $\mathrm{VD}(\mathcal{S}_B \cup \mathcal{S}_C)$, and the cardinality of $\mathcal{S}_C$ is a constant fraction of the cardinality of $\mathcal{S}_R$, i.e., $|\mathcal{S}_C| = \Theta(|\mathcal{S}_R|)$. In the first merge phase, the sites of $\mathcal{S}_C$ are inserted one by one in the recursively computed diagram $\mathrm{VD}(\mathcal{S}_B)$, deriving the diagram $\mathrm{VD}(\mathcal{S}_C \cup \mathcal{S}_B)$. Then, in a second merge phase the resulting $\mathrm{VD}(\mathcal{S}_C \cup \mathcal{S}_B)$ is merged with the recursively computed diagram $\mathrm{VD}(\mathcal{S}_G)$, yielding $\mathrm{VD}(\mathcal{S}_G \cup \mathcal{S}_C \cup \mathcal{S}_B) = \mathrm{VD}(\mathcal{S})$ and concluding the algorithm.

The key factor in obtaining the deterministic linear-time complexity is that the cardinality of the set $\mathcal{S}_C$ is a constant fraction of $\mathcal{S}_R$, which in turn is $\Theta(|\mathcal{S}|)$, and that $\mathcal{S}_C$ can be obtained in linear time. The above is possible due to the following combinatorial result on a geometric binary tree embedded in the plane.



(a) $\mathcal{T}$ corresponds to the Voronoi diagram of the 5 points (■). The neighborhood of ℓ_2 is shown shaded.

(b) The neighborhood of ℓ_2 shows the part of the diagram (dashed) that will get deleted if point (■) is inserted.

(c) The topological ordering of the leaves in $\mathcal{T}$.

Figure 5.2. An embedded binary tree $\mathcal{T}$ in the setting of Aggarwal et al. [1989].

Theorem 5.1. *Let $\mathcal{T}$ be a binary tree embedded in $\mathbb{R}^2$ with n leaves. Each leaf is associated with a* neighborhood*, which is a subtree of T rooted at that leaf, and topologically consecutive leaves have disjoint neighborhoods. Then:*

(1) At least $1/10$ of the leaves have pairwise disjoint neighborhoods such that no tree edge has its endpoints in two different neighborhoods.

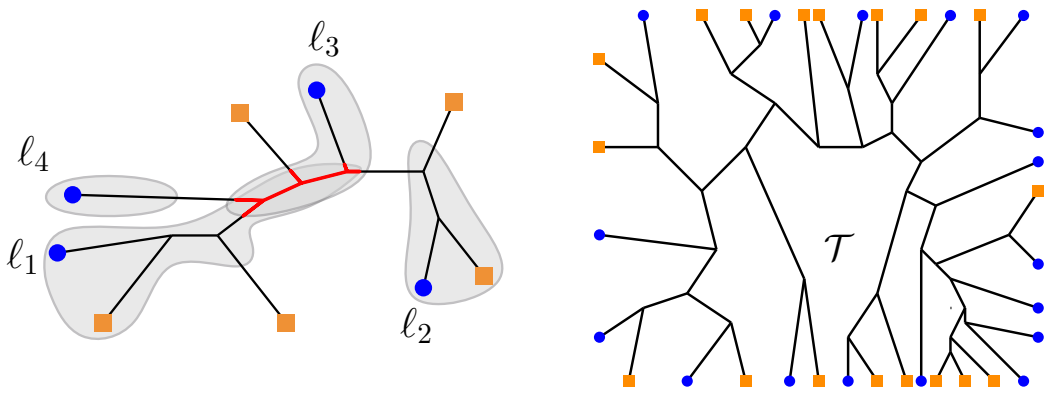
(2) Such leaves with disjoint neighborhoods can be found in $O(n)$ time.

Overall, the time complexity of the algorithm is described by the following recursive equation and can be proved to be $\Theta(n)$.

$$\begin{aligned} T(n) &= T(|\mathcal{S}_B|) + T(|\mathcal{S}_G|) + \Theta(|\mathcal{S}_R|) + |\mathcal{S}_C| \cdot \Theta(1) + \Theta(n) \\ &= T(|\mathcal{S}_B|) + T(|\mathcal{S}_G|) + \Theta(n) \\ &= \Theta(n) \qquad \text{(Because } |\mathcal{S}_C| = \Theta(n)\text{)} \end{aligned}$$

It is worth understanding what Theorem 5.1 represents, in order to have a spherical perspective of its connection to Voronoi diagrams. An embedded tree corresponds to the graph structure of a Voronoi diagram, and leaves are the endpoints of unbounded Voronoi edges "*at infinity*"; see, e.g., Fig. 5.2a. The neighborhood of a leaf corresponds to part of the diagram (of $\mathcal{S}_B$) that gets deleted if a (*missing*) point-site is inserted there; see Fig. 5.2b. Hence, Theorem 5.1 aims to select leaves with pairwise disjoint neighborhoods ($\mathcal{S}_C$), as they can easily, and independently from one another, be inserted in the Voronoi diagram.

Note that the result of Theorem 5.1 is inherently used by any other algorithm which is based on this linear-time framework.



(a) Neighborhoods of the marked leaves are shown shaded. The intersection of neighborhoods is highlighted with red.

(b) A marked tree $\mathcal{T}$, on which the rest of the notions of Section 5.1 are illustrated (in Figs. 5.4a, 5.4b, 5.5 and 5.6).

Figure 5.3. Two marked trees, where marked leaves are shown with (●) and unmarked leaves are shown with (■).

Generalized setting. In the rest of the chapter, we generalize the structures considered, to so-called *marked trees*. As discussed earlier in Section 1.2.3, such trees are inspired by the Voronoi-like structures of Junginger and Papadopoulou [2018], and they are defined as follows.

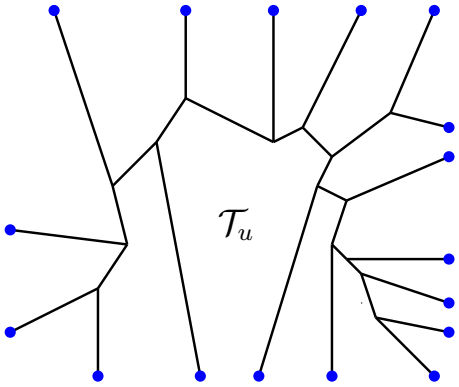
Definition 5.1. A binary tree $\mathcal{T}$ of n leaves embedded in $\mathbb{R}^2$, is called a *marked tree* if it satisfies the following properties.

- m out of the n leaves of $\mathcal{T}$ have been *marked* and the remaining $r := n-m$ leaves are *unmarked*.
- Every marked leaf ℓ is associated with a *neighborhood*, denoted $nh(\ell)$, which is a subtree of $\mathcal{T}$ rooted at ℓ.
- Every two topologically consecutive marked leaves have disjoint neighborhoods.

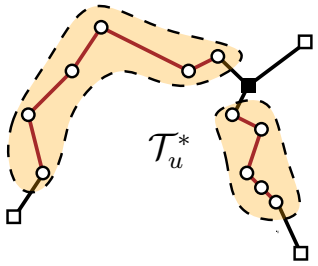
Refer to Fig. 5.3a for an illustration of the generalized setting: out of $n = 9$ leaves only $m = 4$ of them are marked. A neighborhood of marked leaf (shown shaded) might contain also unmarked leaves (see $nh(\ell_1)$), and neighborhoods are not defined for unmarked leaves.

Note that we do not make any assumptions on the ratio between r and m, and we also do not make any assumptions on the distribution of the unmarked leaves among the marked leaves (in the topological ordering).

The main result of this chapter is the following generalized theorem.



(a) Tree $\mathcal{T}_u$.



(b) Tree $\mathcal{T}_u^*$: L-nodes are shown with (□), C-nodes with (○), J-nodes with (■), and spines are highlighted (colored).

Figure 5.4. Illustration of Definition 5.2 applies to the tree $\mathcal{T}$ of Fig. 5.3b.

Theorem 5.2. *Let $\mathcal{T}$ be a marked tree of n total leaves and m marked leaves.*

(1) Then there exist at least $\frac{1}{10}m$ leaves in $\mathcal{T}$ with pairwise disjoint neighborhoods such that no tree edge has its endpoints in two different neighborhoods.

(2) We can select at least a fraction p of these $\frac{1}{10}m$ marked leaves in time $O(\frac{1}{1-p}n)$, for any $p \in (0,1)$.

Given a marked tree $\mathcal{T}$, let $\mathcal{T}_u$ denote the *unmarked tree* obtained by deleting all the unmarked leaves of $\mathcal{T}$ and contracting the resulting degree-2 nodes; see Fig. 5.4a. We apply to $\mathcal{T}_u$ the following definition, which is extracted from the proof of Theorem 5.1 in Aggarwal et al. [1989]; see also Fig. 5.4b.

Definition 5.2. Let T be an embeded binary tree and let T^* be the tree obtained from T after deleting all its leaves. A node u in T^* is called:

- *Leaf* or *L-node* if $deg(u) = 1$ in T^*, i.e., u neighbors two leaves in T.
- *Comb* or *C-node* if $deg(u) = 2$ in T^*, i.e., u neighbors one leaf in T.
- *Junction* or *J-node* if $deg(u) = 3$ in T^*, i.e., u neighbors no leaves in T.

A *spine* is a maximal sequence of consecutive C-nodes, which is delimited by J- or L-nodes. Each spine has two *sides* and marked leaves may lie in either side of a spine.

Let $\mathcal{T}_u^*$, be the tree obtained by applying Definition 5.2 to the unmarked tree $\mathcal{T}_u$. The nodes $\mathcal{T}_u^*$ are labeled as L-, C- and J-nodes; see, e.g., Figs. 5.4a and 5.4b. The labeling of nodes in $\mathcal{T}_u^*$ is then carried back to their corresponding nodes in

the original marked tree $\mathcal{T}$ obtaining a *marked tree $\mathcal{T}$ with labels*, see Fig. 5.5. Some nodes in $\mathcal{T}$ remain unlabeled; see, e.g., node u in Fig. 5.5.

Definition 5.3. Given a marked tree $\mathcal{T}$ with labels we define the following two types of *components*:

a) *L-component*: an L-node λ defines an L-component that consists of λ union the two subtrees of $\mathcal{T}$ that are incident to λ and contain no labeled node (see K_2 in Fig. 5.6). The L-component contains exactly the two marked leaves that labeled λ.

b) 5-*component*: a group of five successive C-nodes $c_i, \ldots, c_{i+4}$ on a spine defines a 5-component that consists of the path $P_{c_i:c_{i+4}}$ from c_i to c_{i+4} (which may contain unlabeled nodes) union the subtrees of $\mathcal{T}$, which are incident to the nodes of $P_{c_i:c_{i+4}}$ and contain no labeled node, (see K_1 in Fig. 5.6). Nodes c_i and c_{i+4} are referred to as the *extreme nodes* of K. The 5-component contains exactly the five marked leaves, which labeled the five C-nodes.

Each spine is partitioned into consecutive groups of 5-components and at most four remaining *ungrouped C-nodes*.

These definitions are illustrated in the instance of Fig. 5.6. The tree $\mathcal{T}$ has three L-components and two 5-components which are indicated shaded. The 5-component K_1 contains the path $P_{c_1:c_5}$ from c_1 to c_5, which is shown in thick black lines, and contains one unlabeled node (node u in Fig. 5.5). Node c_6 is an ungrouped C-node. Further, a spine consisting of the C-nodes $c_1, c_2, c_3, c_4, c_5, c_6$ is highlighted. The spine is delimited by the L-node λ' and the J-node ι; it has five marked leaves from one side and one marked leaf from the other.

Remark 5.1. *The components of $\mathcal{T}$ are pairwise vertex disjoint. Every L-component contains exactly two marked leaves and every* 5*-component contains exactly five marked leaves.*

Among the components of $\mathcal{T}$ there may be subtrees of $\mathcal{T}$ consisting of unlabeled nodes and unmarked leaves that may be arbitrarily large. These subtrees hang off any unlabeled nodes and ungrouped C-nodes. For example, in Fig. 5.6, node u' is unlabeled and the gray dotted subtree incident to it consists solely of unmarked leaves and unlabeled nodes that do not belong to any component.

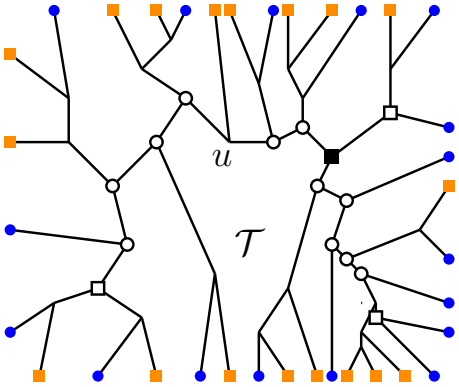


Figure 5.5. The marked tree $\mathcal{T}$ of Fig. 5.3b with labels (○, □, ■). Node u is not labeled.

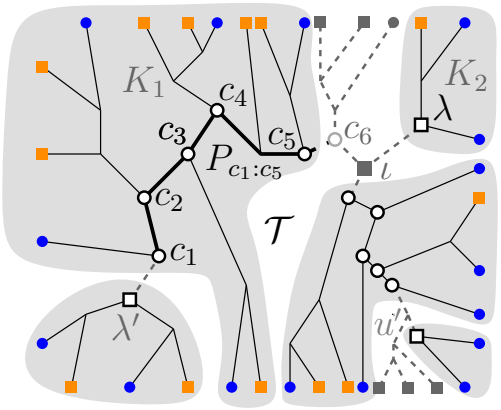


Figure 5.6. The components of $\mathcal{T}$ shown shaded. The dashed parts do not belong to any component.

5.2 Existence of leaves with pairwise disjoint neighborhoods

In this section, we generalize statement (1) of Theorem 5.1. Aggarwal et al. [1989] showed that for every eight ungrouped C-nodes in $\mathcal{T}_u$ there exists at least one L-node. Their argument holds for the marked tree $\mathcal{T}$ as well, which is described in the following lemma for completeness.

Lemma 5.2. *For every eight ungrouped C-nodes in $\mathcal{T}$ there exists at least one L-component.*

Proof. We count the L-nodes of $\mathcal{T}$ using the tree $\mathcal{T}_u^*$ following the argument of Aggarwal et al. [1989]. Let k be the number of leaves in $\mathcal{T}_u^*$, which also equals the number of L-nodes in $\mathcal{T}$. Contracting all degree-2 vertices in $\mathcal{T}_u^*$ yields a binary tree $\mathcal{T}_b^*$, which has the same leaves as $\mathcal{T}_u^*$. Since $\mathcal{T}_b^*$ is an unrooted binary tree with k leaves, it has $2k-2$ nodes and $2k-3$ edges. Every edge in $\mathcal{T}_b^*$ corresponds to at most one spine in $\mathcal{T}_u^*$ and in every spine there are at most four ungrouped C-nodes. Thus,

$$\begin{aligned} |\{\text{ungrouped } C\text{-nodes}\}| &\leqslant 4|\{\text{spines}\}| \\ &\leqslant 4 \cdot (2k-3) \\ &< 8|\{L\text{-nodes}\}|. \end{aligned}$$

So, there exists at least one L-node for every eight ungrouped C-nodes, and an L-node corresponds to exactly one L-component. □

The following lemmas establish that there exists a constant fraction of the marked leaves, which have pairwise disjoint neighborhoods. The counting arguments follow those in Aggarwal et al. [1989] while they are further enhanced to account for the unmarked leaves, which are arbitrarily distributed among the marked leaves. We say that the neighborhood $nh(\ell)$ of a marked leaf ℓ is *confined to a component* K if it is a subtree of K.

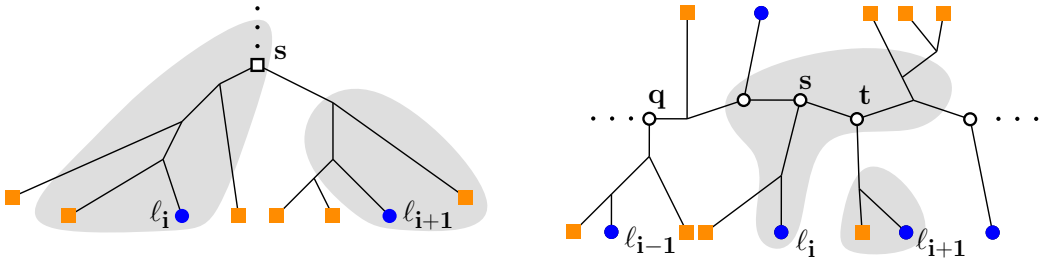
Lemma 5.3. *In every component K, there exists a marked leaf $\ell \in K$ whose neighborhood is confined to K. This neighborhood may contain no L-node and no extreme C-node.*

Proof. Let K be an L-component and let s be the L-node that defines K. Let ℓ_i and ℓ_{i+1} be the two marked leaves of K. Since the neighborhoods $nh(\ell_i)$ and $nh(\ell_{i+1})$ are disjoint, at least one of them cannot contain s. This neighborhood is, thus, entirely contained in the relevant subtree rooted at s and contains no labeled node; see Fig. 5.7a

Let K be a 5-component. Since a 5-component has two sides, at least three out of the five marked leaves of the component must lie on the same side of K, call them ℓ_{i-1}, ℓ_i and ℓ_{i+1}. Let q, s, and t be their corresponding C-nodes, i.e., the first C-nodes in K reachable from ℓ_{i-1}, ℓ_i, and ℓ_{i+1}, respectively; see Fig. 5.7b. There are three cases. If $t \in nh(\ell_i)$, then $t \notin nh(\ell_{i+1})$ (since the two neighborhoods are disjoint), and thus, $nh(\ell_{i+1})$ is confined to the subtree of t that contains ℓ_{i+1}. Similarly, if $q \in nh(\ell_i)$, then $q \notin nh(\ell_{i-1})$, so $nh(\ell_{i-1})$ is confined to the subtree of q containing ℓ_{i-1}. If neither q nor t are in $nh(\ell_i)$, then clearly $nh(\ell_i)$ is confined to K. In all cases the confined neighborhood cannot contain neither q nor t. So, at least one of the five marked leaves must have a neighborhood confined to K and this neighborhood cannot contain the extreme C-nodes in K. □

Lemma 5.4. *Let $\mathcal{T}$ be a marked tree with m marked leaves. At least $\frac{1}{10}m$ marked leaves have pairwise disjoint neighborhoods such that no tree edge has its endpoints in two different neighborhoods.*

Proof. Every spine of $\mathcal{T}$ has up to four ungrouped C-nodes. By Lemma 5.2, there exists at least one L-component for every eight ungrouped C-nodes. By Lemma 5.3, every component of $\mathcal{T}$ has at least one marked leaf whose neighborhood is confined to the component. So, overall, at least $\frac{1}{5}$ of the marked leaves from each 5-component and at least $\frac{1}{10}$ marked leaves of the remaining nodes, which label ungrouped C-nodes or L-nodes, have a confined neighborhood. The components are pairwise disjoint, so at least $\frac{1}{10}$ marked leaves have pairwise disjoint neighborhoods. Furthermore, confined neighborhoods do not contain an



(a) An L-component (b) A 5-component

Figure 5.7. Marked leaves with their neighborhoods shaded. The neighborhood $nh(\ell_i)$ is confined to the component in both cases.

L-node or extreme C-node, as shown in Lemma 5.3. Thus, no tree edge may have its endpoints in two different neighborhoods. □

We remark that the neighborhoods implied by Lemma 5.4 may not contain any L-node nor any extreme C-node. We also remark that these neighborhoods need not be of constant complexity as their counterparts in Aggarwal et al. [1989] are. These neighborhoods may have complexity $\Theta(r)$, where $r = n - m$ is the number of unmarked leaves. Since r may be $\Theta(n)$, this poses a challenge on how we can select these leaves efficiently.

5.3 Selecting leaves with pairwise disjoint neighborhoods

In this section, we generalize statement (2) of Theorem 5.1. Given a marked tree $\mathcal{T}$ with m marked leaves, we have already established the existence of $\frac{1}{10}m$ marked leaves that have pairwise disjoint neighborhoods. In this section, we present an algorithm to select a fraction p of these leaves, i.e., $\frac{p}{10}m$ marked leaves with pairwise disjoint neighborhoods, in time $O(\frac{1}{1-p}n)$, where $0 < p < 1$.

The main challenge over the algorithm of Aggarwal et al. [1989] is that the r unmarked leaves are arbitrarily distributed among the m marked leaves, and thus, the components of $\mathcal{T}$ and the neighborhoods of the marked leaves may have complexity $\Theta(r)$. If for each component we spend time proportional to its size, then the time complexity of the algorithm will be $\Theta(mr)$, i.e., $\Theta(n^2)$ if $r, m \in \Theta(n)$.

To keep the complexity of the algorithm linear, we spend time up to a predefined number of steps in each component depending on the ratio $c = \lceil \frac{r}{m} \rceil$ and the trade-off parameter $p \in (0, 1)$. Our algorithm guarantees to find at least a

fraction p of the possible $\frac{1}{10}m$ marked leaves in time $O(\frac{1}{1-p}n)$. We first present a series of results necessary to establish the correctness of the approach and then describe the algorithm.

Let $\ell_1, \ldots, \ell_m$ be the marked leaves in $\mathcal{T}$ ordered in a counterclockwise topological ordering. Let the *interval* (ℓ_i, ℓ_{i+1}) denote the set of unmarked leaves between ℓ_i and ℓ_{i+1} in the same order. The *interval tree* of (ℓ_i, ℓ_{i+1}), denoted $T_{(\ell_i,\ell_{i+1})}$, is the minimal subtree of $\mathcal{T}$ that contains the marked leaves ℓ_i and ℓ_{i+1}, including the unmarked leaves in (ℓ_i, ℓ_{i+1}), see Fig. 5.8b. We show the following *pigeonhole lemma* involving unmarked leaves and intervals.

Lemma 5.5. *Suppose that r items (unmarked leaves) are distributed in $k \geqslant m$ containers (intervals), and let $c = \lceil \frac{r}{m} \rceil$. For any natural number $x \leqslant r$, let k_x denote the number of containers that contain more than x items. Then, $k_x \leqslant \frac{cm}{x+1}$.*

Proof. Each of the k_x containers contains at least $x+1$ items. Thus,

$$k_x(x+1) \leqslant r \;\Rightarrow\; k_x \;\leqslant\; \frac{r}{x+1}. \tag{5.1}$$

$$c \;=\; \left\lceil \frac{r}{m} \right\rceil \;\Rightarrow\; c \geqslant \frac{r}{m} \;\Rightarrow\; r \;\leqslant\; cm \tag{5.2}$$

$$(5.1) \overset{(5.2)}{\Longrightarrow} k_x \;\leqslant\; \frac{cm}{x+1} \tag{5.3}$$

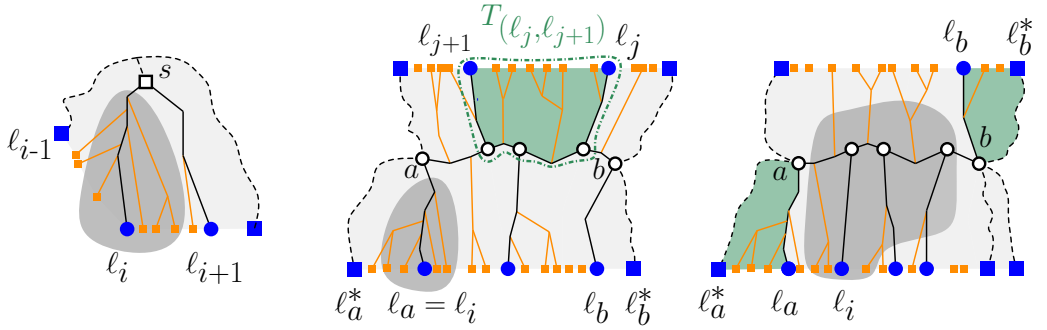
For a component K, let δ_K denote the maximum number of topologically consecutive unmarked leaves in K. The unmarked leaves counted in δ_K belong to some interval (ℓ_i, ℓ_{i+1}).

Lemma 5.6. *Let K be a component of $\mathcal{T}$ and let ℓ_i be a marked leaf whose neighborhood $nh(\ell_i)$ is confined to K.*

a) If K is an L-component, then $nh(\ell_i)$ has at most $4\delta_K$ nodes.
b) If K is a 5-component, then $nh(\ell_i)$ has at most $10\delta_K$ nodes.

Proof. Let K be an L-component whose L-node is s, see Fig. 5.8a. Since $nh(\ell_i)$ is confined to K then $s \notin nh(\ell_i)$. Thus, s disconnects $nh(\ell_i)$ from the rest of $\mathcal{T}$, making $nh(\ell_i)$ disjoint from any interval tree, other than $T_{(\ell_{i-1},\ell_i)}$ and $T_{(\ell_i,\ell_{i+1})}$. Hence, $nh(\ell_i)$ contains at most $2\delta_K + 1$ leaves, and since it is a proper binary tree, it can have at most $4\delta_K$ nodes in total.

Suppose K is a 5-component. Since K contains exacly five marked leaves, there can be at most seven interval trees that may share a node with K. Let a and b be the two extreme C-nodes of K and let ℓ_a and ℓ_b be their corresponding marked leaves, which labeled a and b as C-nodes. Let ℓ_a^* (resp. ℓ_b^*) be the



(a) An L-component.

(b) A 5-component with $\ell_i = \ell_a$. The interval tree $T_{(\ell_j,\ell_{j+1})}$ is highlighted.

(c) A 5-component with $\ell_i \notin \{\ell_a, \ell_b\}$. The interval trees $T_{(\ell_a,\ell_a^*)}$ and $T_{(\ell_b,\ell_b^*)}$ are highlighted

Figure 5.8. Illustration of a component K in different settings for the proof of Lemma 5.6. The neighborhood $nh(\ell_i)$ is shaded gray. Marked leaves of K are indicated with (•) and the other marked leaves with (▪).

neighboring marked leaf of ℓ_a (resp. ℓ_b) in the topological ordering of the marked leaves, which does not belong to K. Refer to Figs. 5.8b and 5.8c. Neighborhood $nh(\ell_i)$ is confined to K, thus, $a, b \notin nh(\ell_i)$. If $\ell_i = \ell_a$ (resp. $\ell_i = \ell_b$) the C-node a (resp. b), disconnects $nh(\ell_i)$ from the rest of $\mathcal{T}$. Thus, $nh(\ell_i)$ has a node in common with only two interval trees, $T_{(\ell_{i-1},\ell_i)}$ and $T_{(\ell_i,\ell_{i+1})}$, see Fig. 5.8b. If $\ell_i \notin \{\ell_a, \ell_b\}$, then nodes a and b disconnect $nh(\ell_i)$ from the rest of $\mathcal{T}$, thus, $nh(\ell_i)$ is disjoint from both $T_{(\ell_a,\ell_a^*)}$ and $T_{(\ell_b,\ell_b^*)}$, see Fig. 5.8c. Then $nh(\ell_i)$ may have a node in common with at most five out of the seven interval trees that could be related to K. Concluding, $nh(\ell_i)$ has at most $5\delta_K + 1$ leaves, and since it is a proper binary tree, it has at most $10\delta_K$ nodes overall. □

For each component K we define a so-called *representative leaf* and at most two *delimiting nodes*. These are used by our algorithm to identify a confined neighborhood within the component.

Definition 5.4. For a component K, we define its *representative leaf* and *delimiting nodes* as follows:

a) If K is an L-component, there is one *delimiting node*, which is its L-node. The *representative leaf* is the first marked leaf of K in the topological ordering of leaves (see leaf ℓ_i and node s in Fig. 5.7a).

b) If K is a 5-component, consider the side of K containing at least three marked leaves. The *representative leaf* is the second leaf among these three

leaves in the topological ordering. The *delimiting nodes* are the C-nodes defined by the other two leaves in the same side (see leaf ℓ_i and nodes q, t in Fig. 5.7b).

Algorithm outline. Our algorithm takes as input a marked tree $\mathcal{T}$ and a parameter $p \in (0,1)$, and returns $\frac{p}{10}m$ marked leaves that have pairwise disjoint neighborhoods. A pseudocode description is given in Algorithm 3. The algorithm iterates over all the components of $\mathcal{T}$, and selects at most one marked leaf for each component.

For each component K, the algorithm first identifies its representative leaf and delimiting nodes (lines 6,14), and then traverses the neighborhood of the representative leaf performing a depth-first-search in the component up to a predefined number of steps (lines 7,15). If, while traversing the neighborhood, a delimiting node is detected (lines 8,16,19), then a marked leaf is selected (lines 9,17,20), following the case analysis of Lemma 5.3. If the entire neighborhood is traversed within the allowed number of steps without detecting a delimiting node (lines 11,22), then the representative leaf is selected (lines 12,23). Otherwise, K is abandoned and the algorithm proceeds to the next component.

Algorithm 3: Selecting leaves with pairwise disjoint neighborhoods.

Input : A marked tree $\mathcal{T}$ of $n = r + m$ leaves and a parameter $p \in (0,1)$.
Output: A set *sol* of marked leaves.

```
1  Obtain the labeling of T;
2  Partition T into components as indicated in Definition 5.3 ;
3  sol ← ∅;   c ← ⌈r/m⌉;   z ← ⌈10c/(1−p)⌉ − 1;
4  for each component K of T do
5      if K is an L-component then
6          ℓ_i ← representative leaf;   s ← delimiting node ;
7          for at most 4z steps traverse nh(ℓ_i)
8              if s is visited then
9                  sol ← sol ∪ {ℓ_{i+1}} ;
10                 break;
11         if nh(ℓ_i) is traversed and s is not visited then
12             sol ← sol ∪ {ℓ_i} ;
13 ...
```

```
12 ...
13 |  else if K is a 5-component then
14 |  |  ℓ_i ← representative leaf,   q, t ← delimiting nodes ;
15 |  |  for at most 10z steps traverse nh(ℓ_i)
16 |  |  |  if q is visited then
17 |  |  |  |  sol ← sol ∪ {ℓ_{i−1}} ;
18 |  |  |  |  break;
19 |  |  |  if t is visited then
20 |  |  |  |  sol ← sol ∪ {ℓ_{i+1}} ;
21 |  |  |  |  break;
22 |  |  if nh(ℓ_i) is traversed and q, t are not visited then
23 |  |  |  sol ← sol ∪ {ℓ_i} ;
24 return sol ;
```

Lemma 5.7. *Algorithm 3 returns at least $\frac{p}{10}m$ marked leaves with pairwise disjoint neighborhoods such that no tree edge has its endpoints in different neighborhoods.*

Proof. Let K be a component. The algorithm traverses the neighborhood of the representative leaf ℓ_i and takes a decision after at most $4z$, or $10z$, steps. In Lemma 5.6, we proved that if $nh(\ell_i)$ is confined, $nh(\ell_i)$ has at most $4\delta_K$, or $10\delta_K$, nodes. Hence, if $\delta_K \leqslant z$, the algorithm will succeed to select a marked leaf from K, because either $nh(\ell_i)$ is confined to K, and thus, the entire $nh(\ell_i)$ is traversed (lines 11-12,22-23), or else a delimiting node gets visited, and thus, the corresponding marked leaf is selected (lines 7-10,15-21). In all cases, we follow the proof of Lemma 5.3 and the neighborhood of the selected leaf is confined to K. Thus the selected leaf is among those counted in Lemma 5.4.

If on the other hand $\delta_k > z$, then the algorithm may fail to identify a marked leaf of K. We use the pigeonhole Lemma 5.5 to bound the number of these components. To this aim, we consider the set I of all intervals induced by the marked leaves and the component of $\mathcal{T}$. For an interval (ℓ_i, ℓ_{i+1}), which is not disjoint from K, let $(\ell_i, \ell_{i+1})_K := (\ell_i, \ell_{i+1}) \cap K$ denote its sub-interval of unmarked leaves that belong to K; see an example in Fig. 5.9. Let I_z be the intervals in I that contain more than z unmarked leaves. Then the algorithm may fail in at most $|I_z|$ components.

To bound $|I_z|$, we use Lemma 5.5 for $x = z = \left\lceil \frac{10c}{1-p} \right\rceil - 1$. Then,

$$|I_z| \overset{(5.3)}{\leqslant} \frac{cm}{z+1} = \frac{cm}{\left\lceil \frac{10c}{1-p} \right\rceil - 1 + 1} \leqslant \frac{cm}{\frac{10c}{1-p}} = \frac{1-p}{10}m \tag{5.4}$$

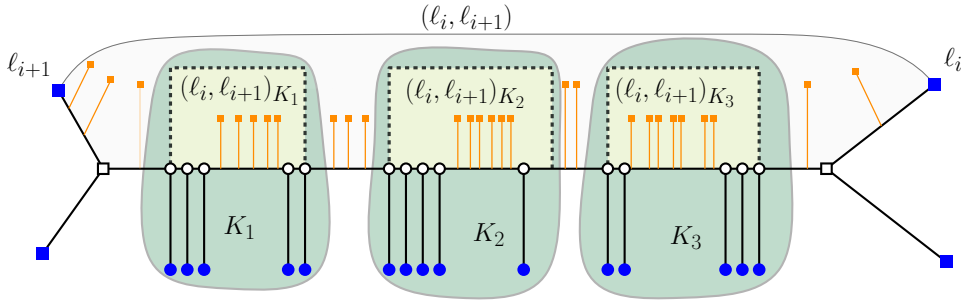


Figure 5.9. An interval (ℓ_i, ℓ_{i+1}) related to three components K_1,K_2 and K_3. Interval (ℓ_i, ℓ_{i+1}) is further subdivided into three intervals $(\ell_i, \ell_{i+1})_{K_1}$,$(\ell_i, \ell_{i+1})_{K_2}$ and $(\ell_i, \ell_{i+1})_{K_3}$.

Thus, the algorithm may fail for at most $\frac{1-p}{10}m$ components. By Lemma 5.2, there exist at least $\frac{1}{10}m$ components in $\mathcal{T}$, thus, the algorithm will succeed in selecting a marked leaf from at least

$$\frac{1}{10}m - |I_z| \overset{(5.4)}{\geqslant} \frac{1}{10}m - \frac{1-p}{10}m = m\frac{p}{10} \tag{5.5}$$

components, concluding the proof. □

Lemma 5.8. *Algorithm 3 has time complexity* $O\left(\frac{1}{1-p}n\right)$.

Proof. Labeling and partitioning the tree $\mathcal{T}$ into components can be done in $\Theta(n)$ time. Then, for each component the algorithm traverses a neighborhood performing at most $10z = \Theta(\frac{c}{1-p})$ steps. There are $\Theta(m)$ components, so we have $O\left(\frac{c}{1-p} \cdot m\right)$ time complexity. Recall that $c = \left\lceil \frac{r}{m} \right\rceil$. If $m = \Theta(n)$, then $c = \Theta(1)$, so $cm = \Theta(n)$. Else if $m = o(n)$, then $cm = \Theta(r) = \Theta(n)$. In all cases, the time complexity of the algorithm is $O\left(\frac{1}{1-p}n\right)$. □

By combining Lemmas 5.4, 5.7 and 5.8 we establish (and re-state) Theorem 5.2.

Theorem 5.2. *Let $\mathcal{T}$ be a marked tree of n total leaves and m marked leaves.*

(1) Then there exist at least $\frac{1}{10}m$ leaves in $\mathcal{T}$ with pairwise disjoint neighborhoods such that no tree edge has its endpoints in two different neighborhoods.

(2) We can select at least a fraction p of these $\frac{1}{10}m$ marked leaves in time $O\left(\frac{1}{1-p}n\right)$, for any $p \in (0, 1)$.

Note that if the parameter $p \in (0, 1)$ is a constant, then the algorithm returns a constant fraction of the marked leaves and the time complexity of the algorithm is $O(n)$. This matches the results of the general setting described in Theorem 5.1.

5.4 Conclusion

In this section we focused on generalizing a combinatorial result on embedded binary trees to, so-called, marked trees. Marked trees were motivated by the Voronoi-like structures introduced by Junginger and Papadopoulou [2018], as an object useful for the generalization of the $O(n)$-time scheme of Chew [1990] to abstract Voronoi diagram. The generalization of the combinatorial result which we presented, was motivated by the potential generalization of the $O(n)$-time scheme of Aggarwal et al. [1989] using Voronoi-like structures.

The combinatorial result in the original setting, consists of two parts, the existence, and the algorithmic one. We managed to generalized both parts to the setting of marked trees. This is important, as the combinatorial result is inherently used by any other algorithm which is based on the $O(n)$-time scheme.

Still, our work is only a stepping stone to the generalization of the algorithmic scheme of Aggarwal et al. [1989] to larger classes of planar Voronoi diagrams. There are still several open questions that need to be addressed. Refer to the recent thesis of [Junginger, 2020, Chapter 7] for a detailed discussion of these open questions.

Chapter 6

Conclusion

In this dissertation, we aimed to advance the state of the art in some problems related to planar Voronoi diagrams. Our research revolved around three topics.

In Chapter 3, we considered color Voronoi diagrams, and more specifically the farthest such diagram. We managed to get a better understanding of the diagram, by studying its structural properties. We refined the combinatorial complexity bound, we identified conditions that induce diagrams of linear complexity, and we studied the case of linearly separable clusters showing a quadratic worst-case lower bound. Our research was complemented by a construction algorithm.

In Chapter 4, motivated by the Brocard illumination problem, we studied the rotating rays Voronoi diagram, a newly defined Voronoi structure. We studied the diagram in $\mathbb{R}^2$, giving combinatorial complexity results, and an accompanying construction algorithm which can be used to obtain the Brocard angle of a set of rays in $\mathbb{R}^2$. We also considered convex polygonal domains bounded by the input rays. By exploiting the properties of the diagram in such domains we managed to solve the Brocard illumination problem in optimal linear time.

In Chapter 5, we considered a problem related to a general deterministic linear-time algorithmic scheme for Voronoi diagrams. We generalized a main component of this linear-time scheme, a combinatorial result on embedded binary trees, aiming to make the scheme applicable to larger classes of planar Voronoi diagrams.

We conclude this dissertation by discussing a few future directions and open questions related to the topics that we looked into. In Section 6.1, we discuss open problems related to color Voronoi diagrams, and in Section 6.2, we discuss open problems related to the rotating rays Voronoi diagram and the Brocard illumination problem.

6.1 Future directions related to color Voronoi diagrams

We now discuss some open problems related to color Voronoi diagrams. We first describe two open questions regarding the farthest color Voronoi diagram of linearly separable clusters. Then, we discuss future directions related to order-k color Voronoi diagrams and the application of color Voronoi diagram in constructing approximate Voronoi diagrams of arbitrary input sites.

Farthest color Voronoi diagram of linearly separable clusters

A first open question is related to the combinatorial complexity of the farthest color Voronoi diagram of linearly separable clusters. We proved that FCVD($\mathcal{P}$) has complexity $\Omega(m^2 + n)$ in the worst case, and we also showed that FCVD($\mathcal{P}$) is upper bounded by $O(n + s(\mathcal{P}))$, where $s(\mathcal{P}) = O(mn)$. If $m = \Theta(n)$, our lower bound is tight, but if $m = o(n)$, a small gap remains.

Open question 1. Given a set $\mathcal{P}$ of linearly separable clusters, how large is $s(\mathcal{P})$ in the worst case? What is the worst-case complexity of FCVD($\mathcal{P}$)?

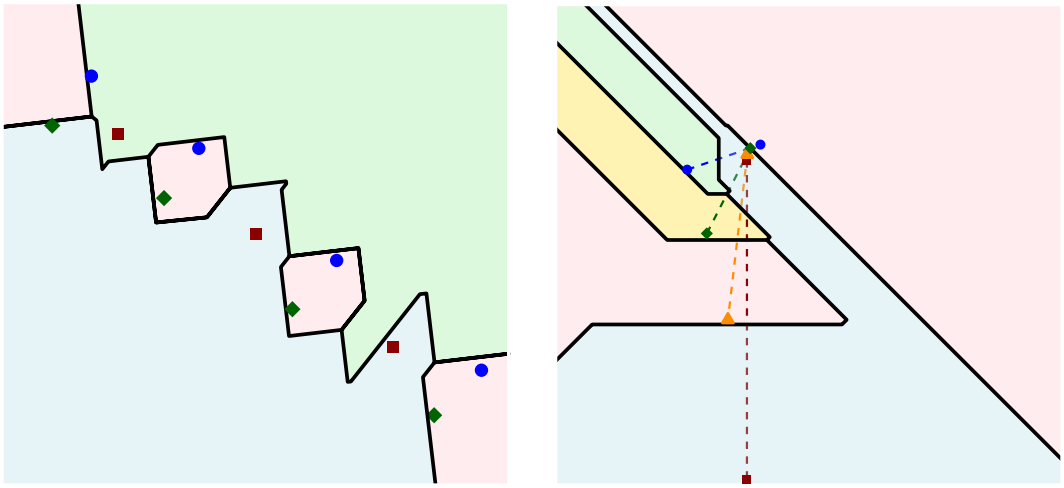
We believe that the diagram has complexity $o(mn)$. In fact, we believe that the straddling number of linearly separable clusters can be proved to be $s(\mathcal{P}) = o(mn)$, thus, immediately lowering the bound on the complexity of FCVD($\mathcal{P}$) (as the diagram has $O(n + s(\mathcal{P}))$ complexity). Our intuition comes from the *degeneracy* of the presented construction that achieves the $\Theta(m^2)$ complexity. This construction is degenerate in the sense that, if we slightly perturb the clusters, then the complexity of the diagram would be reduced.

Farthest color Voronoi diagrams in the L_∞ metric

Oftentimes Voronoi diagrams in the L_∞ metric present a simpler structure than their L_2 counterparts. Regarding the farthest color Voronoi diagram under the L_∞ metric, it has been proved that the diagram has complexity $O(mn)$, and this is complemented by a matching $\Omega(mn)$ lower bound construction.

The $\Omega(mn)$ lower bound construction is given for clusters which have pairwise intersecting convex hulls, and clusters which are linearly separable have not been considered. We are interested in examining whether the $\Omega(mn)$ can also be achieved by linearly separable clusters, or if a tighter upper bound, for example $O(n)$, can be proved in such cases.

Open question 2. Given a set $\mathcal{P}$ of linearly separable clusters, what is the worst-case complexity of FCVD($\mathcal{P}$) under the L_∞ metric?



(a) $\mathcal{P}$ is a set of 3 clusters (■, ◆, ●). The farthest color region of (■) in FCVD($\mathcal{P}$) has $\Theta(n)$ complexity.

(b) $\mathcal{P}$ is a set of 4 clusters (■, ◆, ●, ▲) of size 2. FCVD($\mathcal{P}$) has complexity $\Theta(n)$. Dashed segments indicate the clusters.

Figure 6.1. Farthest color Voronoi diagrams of linearly separable clusters using the L_∞ distance.

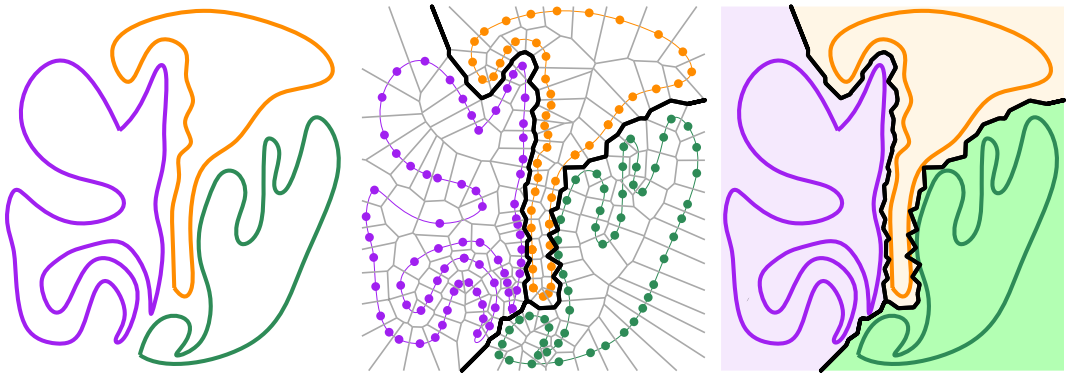
Two examples of farthest color Voronoi diagrams of linearly separable clusters under the L_∞ metric are illustrated in Fig. 6.1. Observe in the instance of Fig. 6.1a, that a single farthest color region (■) can have $\Theta(n)$ complexity. Still generalizing to a construction where all m clusters have complexity $\Theta(n)$ does not seem easy, if possible at all.

In fact, a possible approach to generalize our lower bound construction for the L_2 metric, to the L_∞ metric, does not seem to work. To verify this refer to the instance of Fig. 6.1b. The illustrated clusters of cardinality 2 under the L_2 metric would yield a diagram with $\Theta(m^2) = \Theta(n^2)$ bounded faces, and complexity. On the contrary, under L_∞ not even a single bounded face appears in the diagram.

Application of color Voronoi diagrams in approximating Voronoi diagrams of arbitrary sites

We describe how a well known method for approximating nearest Voronoi diagrams can be extended to Voronoi diagrams of higher order, with the use of color Voronoi diagrams, and the open questions related to this generalization.

Suppose we are given as input sites some curves which are "*not very simple*", as for example the curves shown in Fig. 6.2a, and we want to compute the nearest



(a) The 3 input sites, which are "not very simple" curves.

(b) The nearest Voronoi diagram of all sample-points. Thin edges are induced by points of the same site.

(c) The resulting approximated nearest Voronoi diagram of the 3 curves.

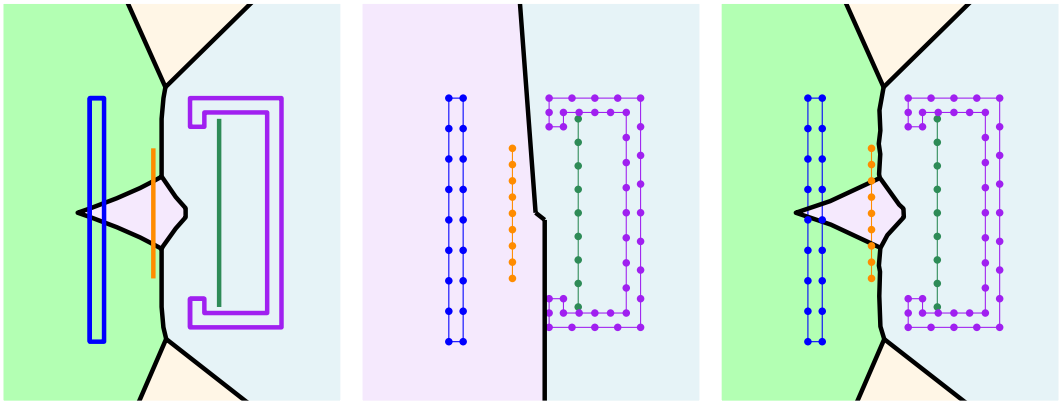
Figure 6.2. An example of approximating a nearest Voronoi diagram of arbitrary sites using the nearest point Voronoi diagram of sample points.

Voronoi diagram of such curves. An exact computation of such a Voronoi diagram might be too complicated or there might not be any available algorithms at all. Hence, depending on the input sites, it might make sense to aim for an *approximate Voronoi diagram*. A standard method is to approximate nearest site Voronoi diagrams using sampled points. This method can be described as follows.

1. For each site s_i in $\mathcal{S} = \{s_1, s_2, \ldots s_m\}$ create a (sample) set of points P_i that approximates the boundary of s_i.
2. Construct the Voronoi diagram $\mathrm{VD}(\{P_1 \cup P_2 \cup \ldots P_m\})$, i.e., the nearest Voronoi diagram of the set of all points.
3. Keep only the Voronoi edges (and the incident Voronoi vertices) which are induced by points belonging to different clusters.
4. Return the remaining diagram which approximates $\mathrm{VD}(\{s_1, s_2, \ldots, s_m\})$.

Refer to Fig. 6.2 where this method is illustrated on an instance of 3 input curves. It is not hard to see that the better the sample points approximate the input sites, the better will the quality of the resulting diagram be. The above method is quite natural, and is being extensively used. To the best of our knowledge, it was first formalized in the literature by Sugihara [1993].

Generalization to higher order Voronoi diagrams. Suppose now that instead of the nearest Voronoi diagram the goal was to construct a higher order Voronoi diagram of some *not very simple* sites, e.g., a farthest Voronoi diagram.



(a) The exact farthest site Voronoi diagram.

(b) "Approximate" diagram using the farthest point Voronoi diagram.

(c) Approximate diagram using the farthest color Voronoi diagram.

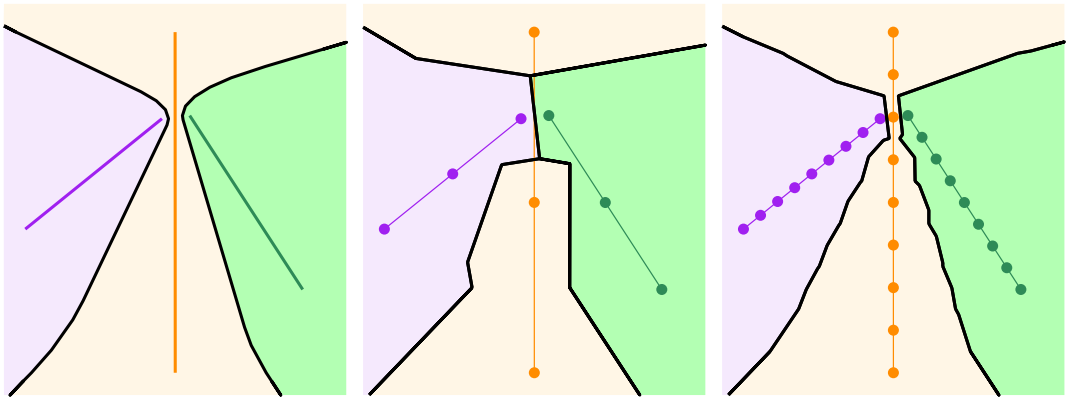
Figure 6.3. An example of approximating the farthest Voronoi diagram of 4 rectangular sites. Illustrated is a comparison of different approaches.

Looking at the simplicity of the aforementioned method, one might be tempted to generalize it as follows: (1) sample the boundaries of the sites, (2) construct the farthest Voronoi of all points, and (3) keep only the edges which are induced by sample-points belonging to different clusters. Such an approach will in general fail. To see that, observe Fig. 6.3, where given is a set of 4 axis aligned polygons and the goal is to construct the farthest Voronoi diagram. In Fig. 6.3a, the exact Voronoi diagram of the 4 sites is shown. In Fig. 6.3b, the aforementioned (bad) generalization is illustrated; observe how different the output is from the exact diagram, despite the densely sampled points.

The problem with the above generalization, is that it fails to capture the nature of the sites and treat all the points corresponding to a sampled curve as a single object. Instead, the way to effectively encode this, is by considering color Voronoi diagrams which inherently deal with clusters of points as input sites. Following we formalize a method which generalizes correctly the method to higher order Voronoi diagrams.

1. For each site s_i in $\mathcal{S} = \{s_1, s_2, \ldots s_m\}$ obtain a (sample) cluster of points P_i that approximates the boundary of s_i. This yields a set of clusters $\mathcal{P}$.
2. Construct the order-k color Voronoi diagram of $\mathcal{P}$.
3. Return the order-k color Voronoi diagram of $\mathcal{P}$ which approximates the order-k Voronoi diagram of $\mathcal{S}$.

For an illustration, refer to the Voronoi diagram of Fig. 6.3c, and observe how



(a) The exact Voronoi diagram. All regions are connected.

(b) The Voronoi diagram of a "bad" sample (3 points per site). The region of (●) is disconnected.

(c) The Voronoi diagram of a "good" sample (9 points per site). All regions are connected.

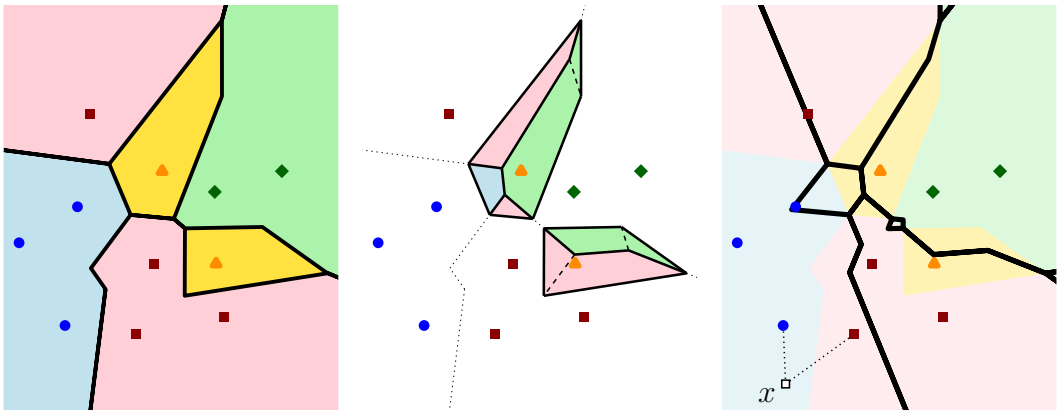
Figure 6.4. Exact and approximate nearest Voronoi diagrams of a set of 3 linearly separable sites (line segments).

well it approximates the exact Voronoi diagram of Fig. 6.3a.

The above approach is very interesting, as it reduces the construction of higher order Voronoi diagram of any input sites, to a simple process of sampling the sites and to the construction of a higher order color Voronoi diagram. There are several open question which involve getting a better understanding between the *quality* of the sampling and the *quality* of the approximation.

It would be meaningful to ask for sampled clusters which yield approximate diagrams that are combinatorially equivalent to the exact ones. As an example, the approximate diagram of Fig. 6.4c is combinatorially equivalent to the exact diagram of Fig. 6.4a whereas the diagram of Fig. 6.4b is not. Similarly, it would be relevant to quantify the difference between an exact and an approximate diagram, e.g., in terms of Hausdorff distance, and ask for sampled clusters which bound this measure. The above questions are very much related to the notion of *local feature size* and ϵ*-samples*, that have been used in Voronoi-based surface reconstruction; see Amenta et al. [1998a] and Amenta and Bern [1999]

From an application viewpoint, we think that this perspective of color Voronoi diagrams will broaden their applicability. It would be interesting to find applications that can benefit from the existence of these approximate Voronoi structures.



(a) Diagram NCVD($\mathcal{P}$), or equivalently 1-CVD($\mathcal{P}$).

(b) Diagram NCVD($\mathcal{P}\setminus$▲) restricted to the region of cluster (▲).

(c) Diagram 2-CVD($\mathcal{P}$). Point x belongs to the region of {■, ●}.

Figure 6.5. Color Voronoi diagrams of a set $\mathcal{P}$ of 4 clusters (■, ◆, ●, ▲) and the illustration of an iterative construction algorithm

Higher order color Voronoi diagrams

Order-k color Voronoi diagrams, except the farthest, have not been considered in the literature, Studying them, both from a combinatorial and an algorithmic perspective, becomes more interesting, due to the aforementioned application of color Voronoi diagrams in constructing approximate Voronoi diagrams.

Open question 3. What is the combinatorial complexity of the order-k color Voronoi diagram k-CVD($\mathcal{P}$)? How fast can we construct k-CVD($\mathcal{P}$)?

An order-2 Voronoi diagram of a set of 4 clusters is illustrated in Fig. 6.5c. Each region is assigned to the two clusters-colors which are present in the region; for example point p belongs to the region of clusters (■) and (●)

On the algorithmic side, one may consider the iterative construction scheme of Lee [1982]. In the context of color Voronoi diagrams, it can be briefly described as follows. The algorithm starts from 1-CVD($\mathcal{P}$), i.e., NCVD($\mathcal{P}$), and at each iteration, the diagram i-CVD($\mathcal{P}$) is constructed using (i-1)-CVD($\mathcal{P}$), until it reaches the target diagram k-CVD($\mathcal{P}$). Given the diagram (i-1)-CVD($\mathcal{P}$) to construct i-CVD($\mathcal{P}$), each region R of (i-1)-CVD($\mathcal{P}$) is considered, and the 1-CVD($\mathcal{P}$) of the sites neighboring R is constructed restricted to R. The union of all these partial diagrams yields i-CVD($\mathcal{P}$).

Refer to Fig. 6.5 for an illustration of the algorithm. In Fig. 6.5b, the nearest color Voronoi diagram of $\{\mathcal{P}\setminus$▲$\}$ is constructed confined into the region of (▲).

This reveals the second nearest cluster for each point, and since the first nearest cluster is already known, this yields the 2-CVD($\mathcal{P}$) confined into that part of $\mathbb{R}^2$.

This iterative scheme is very nice due to its simplicity. It reduces the construction of the construction of any order-k color Voronoi diagram, to the construction of order-1 color Voronoi diagrams, which are simply nearest Voronoi diagrams of points. The complexity of the algorithm, though, depends on the complexity of k-CVD($\mathcal{P}$) which is currently an open question.

6.2 Future directions related to the rotating rays Voronoi diagram

In this section we list some open problems regarding the rotating rays Voronoi diagram and the Brocard illumination problem. We first discuss an open problem related to the diagram in $\mathbb{R}^2$, and another related to the diagram restricted to simple polygonal domains. We conclude by suggesting future directions regarding related Voronoi structures.

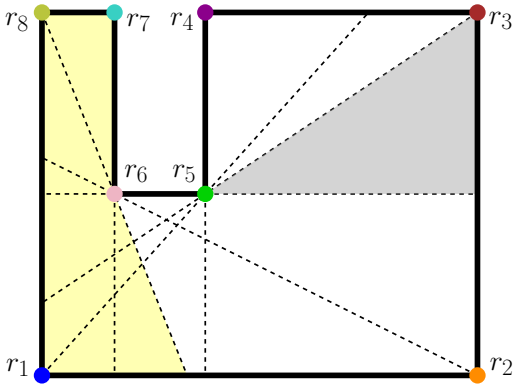
Rotating rays Voronoi diagram in the plane

Using the general approach of envelopes in 3-space, we proved that RVD($\mathcal{R}$) has complexity $O(n^{2+\epsilon})$ for any arbitrarily small $\epsilon > 0$. At the same time we gave a worst-case lower bound of $\Omega(n^2)$. Hence, a gap of $O(n^{\epsilon})$ remains to settle the complexity of the diagram in the worst case.

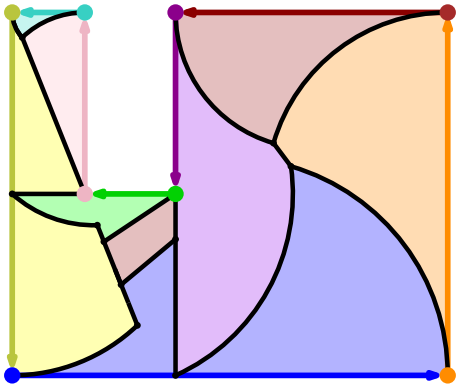
Open question 4. What is the combinatorial complexity of RVD($\mathcal{R}$) in the worst case? Is it $o(n^{2+\epsilon})$? Is it $\Theta(n^2)$?

To close this gap, one could study the different vertices of RVD($\mathcal{R}$) and to bound their number. Out of the 4 different types, the intersection vertices are trivially $O(n^2)$, and the apex vertices are trivially $\Theta(n)$; hence it remains to bound the number of real and mixed vertices. From our preliminary research in this direction we believe that the diagram has $o(n^{2+\epsilon})$ worst-case complexity, and more specifically $\Theta(n^2)$.

Given the current $O(n^{2+\epsilon})$ upper bound, the $O(n^{2+\epsilon})$-time algorithm is worst-case optimal. Assuming that an $o(n^{2+\epsilon})$ bound is given for the complexity, then the question for faster construction algorithms will become imminent. Such an algorithm could deploy any of the standard techniques described, like randomized incremental construction or divide & conquer.



(a) Partition of $\mathcal{P}$ into visibility cells (dashed). The gray region is visible by $\{r_1, \ldots, r_5\}$. The visibility region of r_8 is yellow.



(b) The rotating rays Voronoi diagram of $\mathcal{R}_\mathcal{P}$ restricted to $\mathcal{P}$.

Figure 6.6. A simple polygon $\mathcal{P}$ of 8 vertices and the diagram PRVD($\mathcal{R}_\mathcal{P}$).

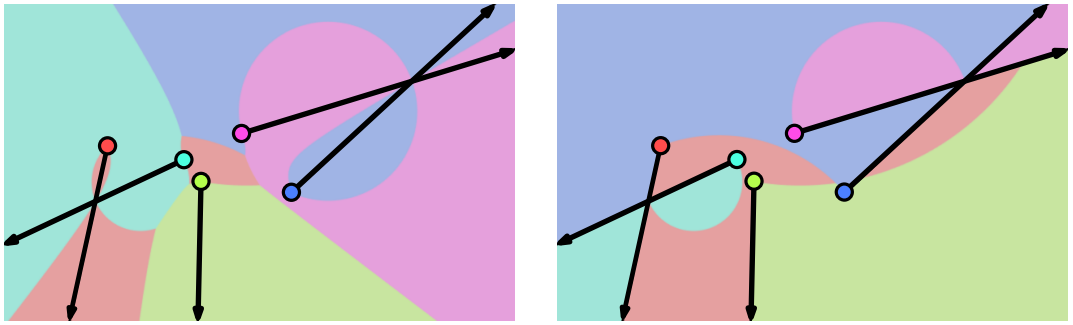
Finally, as with any Voronoi diagram, there are many standard questions to ask. For example what are some necessary or sufficient conditions for the diagram to have linear complexity? Or what are the conditions for the diagram to fall under the abstract Voronoi diagram framework?

Rotating rays Voronoi diagram of non-convex polygons

Regarding the Brocard illumination of polygons, so far we only considered convex polygons. Our approach can be naturally extended to arbitrary simple polygons, although there are some issues that need to be tackled. The main complication in non-convex polygons is that not all points in $\mathcal{P}$ are visible by all rays. Instead, each ray $r \in \mathcal{R}_\mathcal{P}$ has a *visibility region* and the distance to any non-visible point $x \in \mathcal{P}$ is $d_\angle(x, r) = +\infty$. See for example the yellow region in Fig. 6.6a.

Open question 5. Given a simple polygon $\mathcal{P}$ what is the worst-case complexity of PRVD($\mathcal{R}_\mathcal{P}$)? How much time do we need to construct it?

Using the visibility region of each ray in $\mathcal{R}_\mathcal{P}$, a possible approach is to decompose the polygon in *visibility cells*, these are maximal connected components such that any two points p, q in a single visibility cell are visible by the same set of vertices in $\mathcal{P}$. See an example of such a decomposition in Fig. 6.6a. Given a decomposition in visibility cells, we can simply construct RVD($\mathcal{R}_\mathcal{P}$) in each of the cells independently, and then concatenate all these partial Voronoi diagrams



(a) Unoriented angular distance. (b) Oriented angular distance.

Figure 6.7. A set of 5 rays and the corresponding Voronoi diagrams considering different angular distances functions.

to obtain PRVD($\mathcal{R}_\mathcal{P}$). The Brocard angle is simply the maximum over all the vertices of maximum angular distance of all visibility cells. An example of a diagram PRVD($\mathcal{R}_\mathcal{P}$) of a non-convex polygon is illustrated in Fig. 6.6b. Observe that in contrast to the diagram of convex polygons, regions can be disconnected.

Such a visibility cell decomposition has been studied by Guibas et al. [1997] and Bose et al. [2002] It has $O(nr)$ cells, where $r = O(n)$ is the number of reflex vertices in $\mathcal{P}$, it can be computed in $O(n^3)$ time, and each cell is a convex polygon. The above discussion already yields a first construction algorithm for PRVD($\mathcal{R}_\mathcal{P}$): we can simply construct the decomposition in $O(n^3)$ time, and then construct in time $O(n^{2+\epsilon})$ the diagram in each cell of the decomposition. This complexity calls for improvement but it can serve as a framework to start with. Improving upon this will require to delve deeper into the literature of polygons.

As a final remark, when the goal is to find the Brocard angle, one might argue that the construction of the entire diagram is an excessive effort; in the end, the Brocard angle is realized in just one of the many Voronoi vertices of PRVD($\mathcal{R}_\mathcal{P}$). Indeed, this is something to consider, especially given the $O(n^3 \log^2 n)$-time algorithm to find the Brocard angle by Alegría-Galicia et al. [2017]. This is why it would be of particular interest to construct PRVD($\mathcal{R}_\mathcal{P}$) in time $O(n^3 \log^2 n)$.

Rotating rays Voronoi diagram with unoriented angular distance

Our work on this diagram was motivated by the Brocard illumination polygons. In order to model the Brocard illumination problem we defined the analogous distance measure, i.e., oriented angular distance. An alternative meaningful distance measure would be the *unoriented (bidirectional) angular distance*. Intuitively, each ray instead of rotating only counterclockwise, now rotates in both

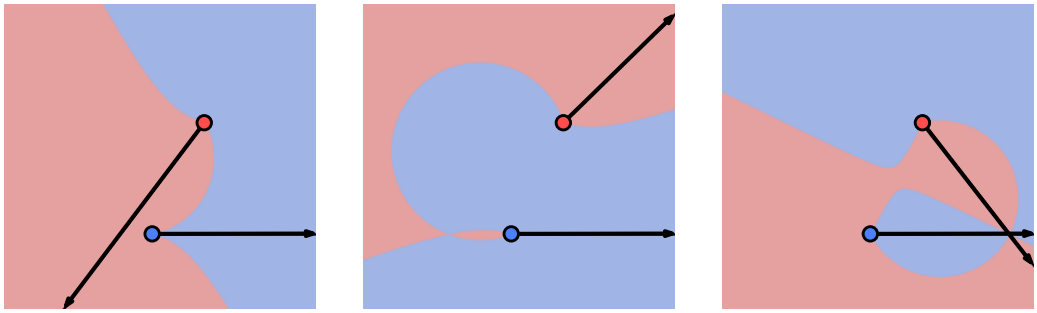


Figure 6.8. Examples of bisectors under the unoriented angular distance.

directions. The Voronoi diagrams, with both distance measures, restricted to a polygon are equivalent, as the right side of each ray is blocked by the corresponding edge. In $\mathbb{R}^2$ though, the two diagrams demonstrate significant differences.

This distance measure was considered by de Berg et al. [2017]. It would be interesting to further investigate this diagram, its connection to the (oriented) rotating rays Voronoi diagram and its application to floodlight illumination.

When comparing the two diagrams, we observe that bisectors under the unoriented distance demonstrate a more complicated structure. Refer to Fig. 6.8 for some examples of angular bisectors using the unoriented distance. More specifically, the two rays are not part of the bisector anymore, but additionally to the circular arc (part of the bisecting circle), the bisector now contains hyperbolic arcs. On the other hand, using the unoriented distance, the discontinuity of the oriented angular distance (at the rays) is eliminated. As a result, no mixed vertices are present and the diagram might be easier to study. Refer to Fig. 6.7 for a set of 5 rays illustrating the differences between the two diagrams.

Finally, note that many of our results for the oriented diagram in $\mathbb{R}^2$ can be easily adapted to the unoriented diagram, as for example the $\Omega(n^2)$ complexity lower bound for pairwise non-interecting rays, the $\Theta(n^2)$ worst-case complexity for a Voronoi region, and the envelopes approach of Sharir [1994] which yields the $O(n^{2+\epsilon})$ complexity upper bound and the $O(n^{2+\epsilon})$-time algorithm.

Disk Voronoi diagram of convex polygons

Another interesting direction would be to explore the disk (Voronoi) diagram, which we defined as an auxiliary structure, in order to study the Voronoi diagram of a convex polygon. Recall that the two diagrams restricted into the polygon coincide, thus constructing the disk diagram also yields the Brocard angle.

The disk diagram of a convex polygon is interesting to study on its own, and to

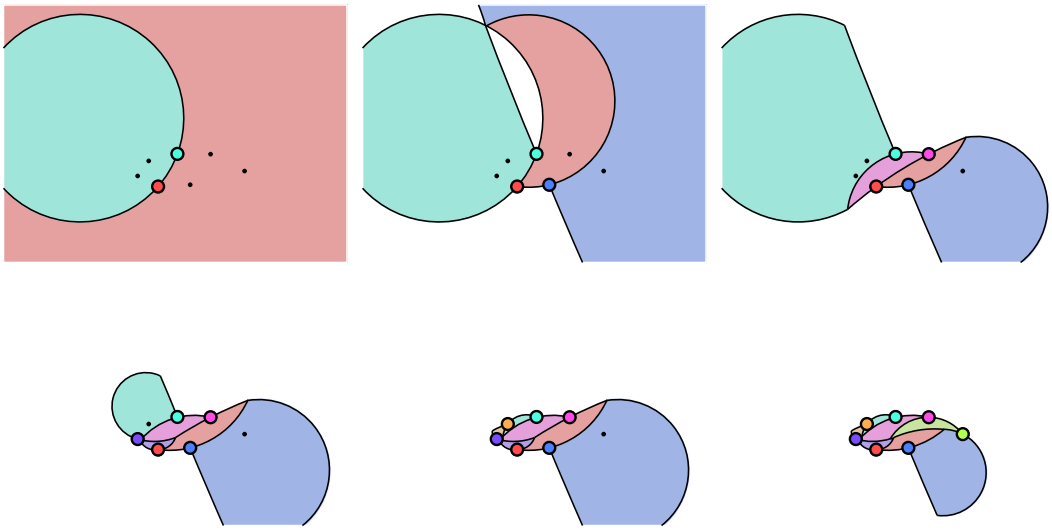


Figure 6.9. An illustration of an incremental construction of the disk diagram of a convex polygon with 7 vertices. In each figure, the inserted sites are highlighted and the remaining are represented by a small dot.

perhaps design a randomized incremental $O(n)$-time algorithm. Refer to Fig. 6.9 for an illustration of an incremental construction, and observe at each insertion the shrinking domain (union of the regions). There seems to be some connection with the recent work of Junginger and Papadopoulou [2018], who presented a randomized incremental algorithm for *tree-like* diagrams, where given is a circular list of the input sites and the domain is shrinking every time a site is inserted.

Concluding, we would like to point out a particularity of this diagram. The dominance relation of the sites is not transitive, and as a result, there are points not belonging to any Voronoi region; see the white surrounding region in Fig. 6.9. From the viewpoint of abstract Voronoi diagrams, this property implies that the disk diagram does not satisfy axiom (A3), which requires that the union of all Voronoi regions covers $\mathbb{R}^2$. This seems to be quite rare; another example of a Voronoi structure exhibiting such a behavior is the *zone (Voronoi) diagram* of Asano et al. [2007]

Bibliography

Ahmed Abdelkader, Ahmed Saeed, Khaled A Harras, and Amr Mohamed. The inapproximability of illuminating polygons by α-floodlights. In *Proceedings of the 27th Canadian Conference on Computational Geometry (CCCG'15)*. Citeseer, 2015.

Manuel Abellanas, Ferran Hurtado, Christian Icking, Rolf Klein, Elmar Langetepe, Lihong Ma, Belén Palop, and Vera Sacristán. Smallest color-spanning objects. In *Proceedings of the 9th Annual European Symposium on Algorithms (ESA 2001)*, pages 278–289. Springer, 2001a.

Manuel Abellanas, Ferran Hurtado, Christian Icking, Rolf Klein, Elmar Langetepe, Lihong Ma, Belén Palop, and Vera Sacristán. The farthest color Voronoi diagram and related problems. In *Proceedings of the 17th European Workshop on Computational Geometry (EuroCG 2001)*, pages 113–116, 2001b.

James Abello, Vladimir Estivill-Castro, Thomas Shermer, and Jorge Urrutia. Illumination of orthogonal polygons with orthogonal floodlights. *International Journal of Computational Geometry & Applications*, 8(01):25–38, 1998.

Ankush Acharyya, Subhas C Nandy, and Sasanka Roy. Minimum width color spanning annulus. *Theoretical Computer Science*, 725:16–30, 2018.

Alok Aggarwal, Leonidas J. Guibas, James Saxe, and Peter W. Shor. A linear-time algorithm for computing the Voronoi diagram of a convex polygon. *Discrete & Computational Geometry*, 4(6):591–604, 1989.

Carlos Alegría, Ioannis Mantas, Evanthia Papadopoulou, Marko Savić, Hendrik Schrezenmaier, Carlos Seara, and Martin Suderland. The Voronoi diagram of rotating rays with applications to floodlight illumination. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

Carlos Alegría-Galicia, David Orden, Carlos Seara, and Jorge Urrutia. Illuminating polygons by edge-aligned floodlights of uniform angle (Brocard illumination). In *Proceedings of the 33rd European Workshop on Computational Geometry (EuroCG 2017)*, pages 281–284, 2017.

Helmut Alt, Otfried Cheong, and Antoine Vigneron. The Voronoi diagram of curved objects. *Discrete & Computational Geometry*, 34(3):439–453, 2005.

Nina Amenta and Marshall Bern. Surface reconstruction by Voronoi filtering. *Discrete & Computational Geometry*, 22(4):481–504, 1999.

Nina Amenta, Marshall Bern, and David Eppstein. The crust and the β-skeleton: Combinatorial curve reconstruction. *Graphical models and image processing*, 60(2):125–135, 1998a.

Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *Proceedings of the 25th Annual conference on Computer graphics and interactive techniques (SIGGRAPH '98)*, pages 415–421, 1998b.

Esther M Arkin, José Miguel Díaz-Báñez, Ferran Hurtado, Piyush Kumar, Joseph SB Mitchell, Belén Palop, Pablo Pérez-Lantero, Maria Saumell, and Rodrigo I Silveira. Bichromatic 2-center of pairs of points. *Computational Geometry*, 48(2):94–107, 2015.

Elena Arseneva and Evanthia Papadopoulou. Randomized incremental construction for the Hausdorff Voronoi diagram revisited and extended. *Journal of Combinatorial Optimization*, 37(2):579–600, 2019.

Tetsuo Asano, Hisao Tamaki, Naoki Katoh, and Takeshi Tokuyama. Angular Voronoi diagram with applications. In *Proceedings of the 3rd International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2006)*, pages 18–24. IEEE, 2006.

Tetsuo Asano, JiřÍ Matoušek, and Takeshi Tokuyama. Zone diagrams: Existence, uniqueness, and algorithmic challenge. *SIAM Journal on Computing*, 37(4): 1182–1198, 2007.

Franz Aurenhammer. Power diagrams: properties, algorithms and applications. *SIAM Journal on Computing*, 16(1):78–96, 1987.

Franz Aurenhammer and Herbert Edelsbrunner. An optimal algorithm for constructing the weighted Voronoi diagram in the plane. *Pattern recognition*, 17 (2):251–257, 1984.

Franz Aurenhammer, Robert L Scot Drysdale, and Hannes Krasser. Farthest line segment Voronoi diagrams. *Information Processing Letters*, 100(6):220–225, 2006.

Franz Aurenhammer, Rolf Klein, and Der-Tsai Lee. *Voronoi Diagrams and Delaunay Triangulations*. World Scientific, 2013. ISBN 978-981-4447-63-8.

Franz Aurenhammer, Evanthia Papadopoulou, and Martin Suderland. Piecewise-linear farthest-site Voronoi diagrams. In *Proceedings of the 32nd International Symposium on Algorithms and Computation (ISAAC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2021.

Sang Won Bae. On linear-sized farthest-color Voronoi diagrams. *IEICE TRANSACTIONS on Information and Systems*, 95(3):731–736, 2012.

Sang Won Bae. Tight bound and improved algorithm for farthest-color Voronoi diagrams of line segments. *Computational Geometry*, 47(8):779–788, 2014.

Sang Won Bae. An almost optimal algorithm for Voronoi diagrams of non-disjoint line segments. *Computational Geometry*, 52:34–43, 2016.

Sang Won Bae, Chunseok Lee, and Sunghee Choi. On exact solutions to the Euclidean bottleneck Steiner tree problem. *Information Processing Letters*, 110 (16):672–678, 2010.

Jay Bagga, Laxmi Gewali, and David Glasser. The complexity of illumination polygons by α-flood-lights. In *Proceedings of the 8th Canadian Conference on Computational Geometry (CCCG'96)*, pages 337–342, 1996.

Gill Barequet, Matthew T Dickerson, and Robert L Scot Drysdale. 2-point site Voronoi diagrams. *Discrete Applied Mathematics*, 122(1-3):37–54, 2002.

Gill Barequet, Matthew Dickerson, David Eppstein, David Hodorkovsky, and Kira Vyatkina. On 2-site Voronoi diagrams under geometric distance functions. *Journal of Computer Science and Technology*, 28(2):267–277, 2013.

Piotr Berman, Jieun Jeong, Shiva P. Kasiviswanathan, and Bhuvan Urgaonkar. Packing to angles and sectors. In *Proceedings of the 19th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '07)*, pages 171–180, 2007.

Arthur Bernhart. Polygons of pursuit. *Scripta Mathematica*, 24(1):23–50, 1959.

Ádám Besenyei. The Brocard angle and a geometrical gem from Dmitriev and Dynkin. *The American Mathematical Monthly*, 122(5):495–499, 2015.

Cecilia Bohler and Rolf Klein. Abstract Voronoi diagrams with disconnected regions. *International Journal of Computational Geometry & Applications*, 24(04): 347–372, 2014.

Cecilia Bohler, Rolf Klein, and Chih-Hung Liu. Forest-like abstract Voronoi diagrams in linear time. In *Proceedings of the 26th Canadian Conference on Computational Geometry (CCCG'14)*, 2014.

Cecilia Bohler, Panagiotis Cheilaris, Rolf Klein, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. On the complexity of higher order abstract Voronoi diagrams. *Computational Geometry*, 48(8):539–551, 2015.

Cecilia Bohler, Chih-Hung Liu, Evanthia Papadopoulou, and Maksym Zavershynskyi. A randomized divide and conquer algorithm for higher-order abstract Voronoi diagrams. *Computational Geometry*, 59:26–38, 2016.

Cecilia Bohler, Rolf Klein, and Chih-Hung Liu. Abstract Voronoi diagrams from closed bisecting curves. *International Journal of Computational Geometry & Applications*, 27(03):221–240, 2017.

Cecilia Bohler, Rolf Klein, and Chih-Hung Liu. An efficient randomized algorithm for higher-order abstract Voronoi diagrams. *Algorithmica*, 81(6):2317–2345, 2019.

J-D Boissonnat, Micha Sharir, Boaz Tagansky, and Mariette Yvinec. Voronoi diagrams in higher dimensions under certain polyhedral distance functions. *Discrete & Computational Geometry*, 19(4):485–519, 1998.

Prosenjit Bose, Leonidas J. Guibas, Anna Lubiw, Mark Overmars, Diane Souvaine, and Jorge Urrutia. The floodlight problem. *International Journal of Computational Geometry & Applications*, 7(1-2):153–163, 1997.

Prosenjit Bose, Anna Lubiw, and J Ian Munro. Efficient visibility queries in simple polygons. *Computational Geometry*, 23(3):313–335, 2002.

David Bremner, Erik Demaine, Jeff Erickson, John Iacono, Stefan Langerman, Pat Morin, and Godfried Toussaint. Output-sensitive algorithms for computing nearest-neighbour decision boundaries. *Discrete & Computational Geometry*, 33(4):593–604, 2005.

Matthew Cary, Atri Rudra, Ashish Sabharwal, and Erik Vee. Floodlight illumination of infinite wedges. *Computational Geometry*, 43(1):23–34, 2010.

John Casey. *A sequel to the first six books of the elements of Euclid*. Dublin University Press, 1888.

Bernard Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(4):377–409, 1993.

Danny Z Chen, Ziyun Huang, Yangwei Liu, and Jinhui Xu. On clustering induced Voronoi diagrams. *SIAM Journal on Computing*, 46(6):1679–1711, 2017.

Lisi Chen, Shuo Shang, Chengcheng Yang, and Jing Li. Spatial keyword search: a survey. *GeoInformatica*, 24(1):85–106, 2020.

Otfried Cheong, Hazel Everett, Marc Glisse, Joachim Gudmundsson, Samuel Hornus, Sylvain Lazard, Mira Lee, and Hyeon-Suk Na. Farthest-polygon Voronoi diagrams. *Computational Geometry*, 44(4):234–247, 2011.

L. Paul Chew. Building Voronoi diagrams for convex polygons in linear expected time. Technical report, Dartmouth College, 1990.

Francis Chin, Jack Snoeyink, and Cao An Wang. Finding the medial axis of a simple polygon in linear time. In *Proceedings of the 6th International Symposium on Algorithms and Computation (ISAAC 1995)*, pages 382–391. Springer, 1995.

Kenneth L Clarkson and Peter W Shor. Applications of random sampling in computational geometry, ii. *Discrete & Computational Geometry*, 4(5):387–421, 1989.

Mercè Claverol, Elena Khramtcova, Evanthia Papadopoulou, Maria Saumell, and Carlos Seara. Stabbing circles for sets of segments in the plane. *Algorithmica*, 80(3):849–884, 2018.

Felipe Contreras, Jurek Czyzowicz, Nicolas Fraiji, and Jorge Urrutia. Illuminating triangles and quadrilaterals with vertex floodlights. In *Proceedings of the 10th Canadian Conference on Computational Geometry (CCCG '98)*, 1998a.

Felipe Contreras, Jurek Czyzowicz, Eduardo Rivera-Campo, and Jorge Urrutia. Optimal floodlight illumination of stages. In *Proceedings of the 14th annual symposium on Computational geometry (SoCG 1998)*, pages 409–410, 1998b.

Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

Jurek Czyzowicz, Stefan Dobrev, Benson Joeris, Evangelos Kranakis, Danny Krizanc, Jan Maňuch, Oscar Morales-Ponce, Jaroslav Opatrny, Ladislav Stacho, and Jorge Urrutia. Monitoring the plane with rotating radars. *Graphs and Combinatorics*, 31(2):393–405, 2015.

Sandip Das, Partha P Goswami, and Subhas C Nandy. Smallest color-spanning object revisited. *International Journal of Computational Geometry & Applications*, 19(05):457–478, 2009.

Mark de Berg, Joachim Gudmundsson, Herman Haverkort, and Michael Horton. Voronoi diagrams with rotational distance costs. In *Abstracts of the Computational Geometry Week: Young Researchers Forum (CG:YRF)*, 2017.

Frank Dehne, Anil Maheshwari, and Ryan Taylor. A coarse grained parallel algorithm for Hausdorff Voronoi diagrams. In *Proceedings of the 35th International Conference on Parallel Processing (ICPP'06)*, pages 497–504. IEEE, 2006.

Boris Delaunay. Sur la sphère vide. a la mémoire de Georges Voronoï. (6):793–800, 1934.

René Descartes. *Principia philosophiae*. Apud Ludovicum Elzevirium, 1644.

Jana Dietel, Hans-Dietrich Hecker, and Andreas Spillner. A note on optimal floodlight illumination of stages. *Information Processing Letters*, 105(4):121–123, 2008.

Hu Ding and Jinhui Xu. Solving the chromatic cone clustering problem via minimum spanning sphere. In *Proceedings of the 38th International Colloquium on Automata, Languages, and Programming (ICALP 2011)*, pages 773–784. Springer, 2011.

G. Lejeune Dirichlet. Über die reduction der positiven quadratischen formen mit drei unbestimmten ganzen zahlen. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1850(40):209–227, 1850.

Hristo Djidjev and Andrzej Lingas. On computing the Voronoi diagram for restricted planar figures. In *Proceedings of the 2nd Workshop on Algorithms and Data Structures (WADS 1991)*, pages 54–64. Springer, 1991.

Nikolai A. Dmitriev and E Dynkin. On characteristic roots of stochastic matrices. *Izvestiya Rossiiskoi Akademii Nauk. Seriya Matematicheskaya*, 10(2):167–184, 1946.

Herbert Edelsbrunner and Raimund Seidel. Voronoi diagrams and arrangements. *Discrete & Computational Geometry*, 1(1):25–44, 1986.

Herbert Edelsbrunner, Leonidas J Guibas, and Micha Sharir. The upper envelope of piecewise linear functions: algorithms and applications. *Discrete & Computational Geometry*, 4(1):311–336, 1989.

Friedrich Eisenbrand, Stefan Funke, Andreas Karrenbauer, and Domagoj Matijevic. Energy-aware stage illumination. *International Journal of Computational Geometry & Applications*, 18(01n02):107–129, 2008.

Ioannis Z Emiris, Elias P Tsigaridas, and George M Tzoumas. The predicates for the Voronoi diagram of ellipses. In *Proceedings of the 22nd Annual Symposium on Computational geometry (SoCG 2006)*, pages 227–236, 2006.

David Eppstein. Finding relevant points for nearest-neighbor classification. In *Proceedings of the 5th Symposium on Simplicity in Algorithms (SOSA22)*, pages 68–78. SIAM, 2022.

Vladimir Estivill-Castro, Joseph O'Rourke, Jorge Urrutia, and Dianna Xu. Illumination of polygons with vertex lights. *Information Processing Letters*, 56(1): 9–13, 1995.

Rudolf Fleischer and Xiaoming Xu. Computing minimum diameter color-spanning sets. In *Proceedings of the 4th International Workshop on Frontiers in Algorithmics (FAW 2010)*, pages 285–292. Springer, 2010.

Peter J Green and Robin Sibson. Computing Dirichlet tessellations in the plane. *The computer journal*, 21(2):168–173, 1978.

Laura Guggenbuhl. Henri Brocard and the geometry of the triangle. *The Mathematical Gazette*, 37(322):241–243, 1953.

Leonidas J Guibas, Donald E Knuth, and Micha Sharir. Randomized incremental construction of Delaunay and Voronoi diagrams. *Algorithmica*, 7(1):381–413, 1992.

Leonidas J Guibas, Rajeev Motwani, and Prabhakar Raghavan. The robot localization problem. *SIAM Journal on Computing*, 26(4):1120–1138, 1997.

Tao Guo, Xin Cao, and Gao Cong. Efficient algorithms for answering the m-closest keywords query. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data (PODS'15)*, pages 405–418, 2015.

Sergiu Hart and Micha Sharir. Nonlinearity of Davenport—Schinzel sequences and of generalized path compression schemes. *Combinatorica*, 6(2):151–177, 1986.

John Hershberger. Finding the upper envelope of n line segments in $O(n \log n)$ time. *Information Processing Letters*, 33(4):169–174, 1989.

Ziyun Huang, Danny Z Chen, and Jinhui Xu. Influence-based Voronoi diagrams of clusters. *Computational Geometry*, 96:101746, 2021.

Daniel P Huttenlocher, Klara Kedem, and Jon M Kleinberg. On dynamic Voronoi diagrams and the minimum Hausdorff distance for point sets under euclidean motion in the plane. In *Proceedings of the 8th annual symposium on Computational geometry (SoCG 1992)*, pages 110–119, 1992.

Daniel P Huttenlocher, Klara Kedem, and Micha Sharir. The upper envelope of Voronoi surfaces and its applications. *Discrete & Computational Geometry*, 9 (3):267–291, 1993.

John Iacono, Elena Khramtcova, and Stefan Langerman. Searching edges in the overlap of two plane graphs. In *Proceedings of the 15th Algorithms and Data Structures Symposium (WADS 2017)*, pages 473–484. Springer, 2017.

Dan Ismailescu. Illuminating a convex polygon with vertex lights. *Periodica Mathematica Hungarica*, 57(2):177–184, 2008.

Hiro Ito, Hideyuki Uehara, and Mitsuo Yokoyama. NP-completeness of stage illumination problems. In *Proceedings of the 2nd Japanese Conference on Discrete and Computational Geometry (JCDCG '98:)*, pages 158–165. Springer, 1998.

Allan Jørgensen, Maarten Löffler, and Jeff M Phillips. Geometric computations on indecisive points. In *Proceedings of the 12th Algorithms and Data Structures Symposium (WADS 2011)*, pages 536–547. Springer, 2011.

Kolja Junginger. *Voronoi-like diagrams: towards linear-time algorithms for tree-like abstract Voronoi diagrams*. PhD thesis, Università della Svizzera italiana, 2020.

Kolja Junginger and Evanthia Papadopoulou. Deletion in abstract Voronoi diagrams in expected linear time and related problems. *arXiv preprint arXiv:1803.05372*, 2018.

Kolja Junginger, Ioannis Mantas, and Evanthia Papadopoulou. On selecting a fraction of leaves with disjoint neighborhoods in a plane tree. *Discrete Applied Mathematics*, 2021a.

Kolja Junginger, Ioannis Mantas, Evanthia Papadopoulou, Martin Suderland, and Chee Yap. Certified approximation algorithms for the Fermat point and n-ellipses. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021b.

Payam Khanteimouri, Ali Mohades, Mohammad Ali Abam, and Mohammad Reza Kazemi. Computing the smallest color-spanning axis-parallel square. In *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC 2013)*, pages 634–643. Springer, 2013.

Elena Khramtcova and Evanthia Papadopoulou. An expected linear-time algorithm for the farthest-segment Voronoi diagram. *arXiv preprint arXiv:1411.2816*, 2017.

David Kirkpatrick. Optimal search in planar subdivisions. *SIAM Journal on Computing*, 12(1):28–35, 1983.

Victor Klee. On the complexity of d-dimensional Voronoi diagrams. *Archiv der Mathematik*, 34(1):75–80, 1980.

Rolf Klein. *Concrete and abstract Voronoi diagrams*, volume 400. Springer Science & Business Media, 1989.

Rolf Klein and Andrzej Lingas. Hamiltonian abstract Voronoi diagrams in linear time. In *Proceedings of the 5th International Symposium on Algorithms and Computation (ISAAC 1994)*, pages 11–19. Springer, 1994.

Rolf Klein, Kurt Mehlhorn, and Stefan Meiser. Randomized incremental construction of abstract Voronoi diagrams. *Computational Geometry*, 3(3):157–184, 1993.

Rolf Klein, Elmar Langetepe, and Zahra Nilforoushan. Abstract Voronoi diagrams revisited. *Computational Geometry*, 42(9):885–902, 2009.

Evangelos Kranakis, Danny Krizanc, and Oscar Morales. Maintaining connectivity in sensor networks using directional antennae. In *Theoretical Aspects of Distributed Computing in Sensor Networks*, pages 59–84. Springer, 2011.

Chunseok Lee, Donghoon Shin, Sang Won Bae, and Sunghee Choi. Best and worst-case coverage problems for arbitrary paths in wireless sensor networks. *Ad hoc networks*, 11(6):1699–1714, 2013.

Der-Tsai Lee. On k-nearest neighbor Voronoi diagrams in the plane. *IEEE transactions on computers*, 100(6):478–487, 1982.

Der-Tsai Lee and CK Wong. Voronoi diagrams in $L_1(L_\infty)$ metrics with 2-dimensional storage applications. *SIAM Journal on computing*, 9(1):200–211, 1980.

Ioannis Mantas, Evanthia Papadopoulou, Vera Sacristán, and Rodrigo I Silveira. Farthest color Voronoi diagrams: Complexity and algorithms. In *Proceedings of the 14th Latin American Symposium on Theoretical Informatics (LATIN 2020)*, pages 283–295. Springer, 2021a.

Ioannis Mantas, Marko Savić, and Hendrik Schrezenmaier. New variants of perfect non-crossing matchings. In *Proceedings of the 7th Annual International Conference on Algorithms and Discrete Applied Mathematics (CALDAM 2021)*, pages 151–164. Springer, 2021b.

Kurt Mehlhorn, Stefan Meiser, and Ronald Rasch. Furthest site abstract Voronoi diagrams. *International Journal of Computational Geometry & Applications*, 11 (06):583–616, 2001.

Hyeon-Suk Na, Chung-Nim Lee, and Otfried Cheong. Voronoi diagrams on the sphere. *Computational Geometry*, 23(2):183–194, 2002.

Azin Neishaboori, Ahmed Saeed, Khaled A. Harras, and Amr Mohamed. On target coverage in mobile visual sensor networks. In *Proceedings of the 12th ACM International Symposium on Mobility Management and Wireless Access (MobiWac '15)*, pages 39–46, 2014.

Eunjin Oh. Optimal algorithm for geodesic nearest-point Voronoi diagrams in simple polygons. In *Proceedings of the 39th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA19)*, pages 391–409. SIAM, 2019.

Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons, 2009.

Joseph O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc., 1987.

Joseph O'Rourke. Visibility. In *Handbook of discrete and computational geometry*, pages 875–896. CRC Press, 2017.

Joseph O'Rourke, Thomas Shermer, and Ileana Streinu. Illuminating convex polygons with vertex floodlights. In *Proceedings of the 7th Canadian Conference on Computational Geometry (CCCG'95)*, pages 151–156, 1995.

Evantha Papadopoulou and DT Lee. The L_∞ Voronoi diagram of segments and vlsi applications. *International Journal of Computational Geometry & Applications*, 11(05):503–528, 2001.

Evanthia Papadopoulou. The Hausdorff Voronoi diagram of point clusters in the plane. *Algorithmica*, 40(2):63–82, 2004.

Evanthia Papadopoulou and Sandeep Kumar Dey. On the farthest line-segment voronoi diagram. *International Journal of Computational Geometry & Applications*, 23(06):443–459, 2013.

Evanthia Papadopoulou and Der-Tsai Lee. A new approach for the geodesic Voronoi diagram of points in a simple polygon and other restricted polygonal domains. *Algorithmica*, 20(4):319–352, 1998.

Evanthia Papadopoulou and Der-Tsai Lee. The Hausdorff Voronoi diagram of polygonal objects: A divide and conquer approach. *International Journal of Computational Geometry & Applications*, 14(06):421–452, 2004.

Evanthia Papadopoulou and Maksym Zavershynskyi. The higher-order Voronoi diagram of line segments. *Algorithmica*, 74(1):415–439, 2016.

Franco P Preparata and Michael I Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

Ophir Setter, Micha Sharir, and Dan Halperin. Constructing two-dimensional Voronoi diagrams via divide-and-conquer of envelopes in space. In *Transactions on computational science IX*, pages 1–27. Springer, 2010.

Michael Ian Shamos. *Computational geometry*. Yale University, 1978.

Michael Ian Shamos. The early years of computational geometry—a personal memoir. In *Advances in Discrete and Computational Geometry: Proceedings of the 1996 AMS-IMS-SIAM Joint Summer Research Conference, Discrete and Computational Geometry–Ten Years Later*, volume 223, page 313. American Mathematical Society, 1999.

Michael Ian Shamos and Dan Hoey. Closest-point problems. In *Proceedings of the 16th Annual Symposium on Foundations of Computer Science (FOCS 1975)*, pages 151–162. IEEE, 1975.

Micha Sharir. Almost tight upper bounds for lower envelopes in higher dimensions. *Discrete & Computational Geometry*, 12(3):327–345, 1994.

Sven Skyum. A simple algorithm for computing the smallest enclosing circle. *Information Processing Letters*, 37(3):121–125, 1991.

Andreas Spillner and Hans-Dietrich Hecker. Minimizing the size of vertexlights in simple polygons. *Mathematical Logic Quarterly: Mathematical Logic Quarterly*, 48(3):447–458, 2002.

William Steiger and Ileana Streinu. Illumination by floodlights. *Computational Geometry*, 10(1):57–70, 1998.

Kokichi Sugihara. Approximation of generalized Voronoi diagrams by ordinary Voronoi diagrams. *CVGIP: Graphical Models and Image Processing*, 55(6):522–531, 1993.

Tsuyoshi Taki, Jun-ichi Hasegawa, and Teruo Fukumura. Development of motion analysis system for quantitative evaluation of teamwork in soccer games. In *Proceedings of the 3rd IEEE international conference on image processing (ICIP 1996)*, volume 3, pages 815–818. IEEE, 1996.

Csaba D Tóth. Art gallery problem with guards whose range of vision is 180°. *Computational Geometry*, 17(3-4):121–134, 2000.

Csaba D. Tóth. Art galleries with guards of uniform range of vision. *Computational Geometry*, 21(3):185–192, 2002.

Csaba D. Tóth. Illumination of polygons by 45-floodlights. *Discrete Mathematics*, 265(1):251–260, 2003.

Jorge Urrutia. Art gallery and illumination problems. In *Handbook of Computational Geometry*, pages 973–1027. Elsevier, 2000.

Remco C Veltkamp and Michiel Hagedoorn. State of the art in shape matching. In *Principles of visual information retrieval*, pages 87–119. Springer, 2001.

Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. premier mémoire. sur quelques propriétés des formes quadratiques positives parfaites. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1908(133):97–102, 1908a.

Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1908(134):198–287, 1908b.

Chee K Yap. An O(nlogn) algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete & Computational Geometry*, 2(4):365–393, 1987.

Yongding Zhu and Jinhui Xu. Improved algorithms for the farthest colored Voronoi diagram of segments. *Theoretical Computer Science*, 497:20–30, 2013.