Universidade de Aveiro Departamento de Matemática
2009

Inês Pereira de Matos

Problemas de Cobertura com Alcance Limitado
Limited Range Coverage Problems

Universidade de Aveiro Departamento de Matemática
2009

Inês Pereira de Matos

Problemas de Cobertura com Alcance Limitado
Limited Range Coverage Problems

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Doutor em Matemática, realizada sob a orientação científica do Doutor Antonio Leslie Bajuelos Domínguez, Professor Auxiliar do Departamento de Matemática da Universidade de Aveiro e do Doutor Manuel Abellanas Oar, Professor Titular do Departamento de Matemática Aplicada da Facultad de Informática da Universidad Politécnica de Madrid.

Apoio financeiro da FCT e do FSE no âmbito do III Quadro Comunitário de Apoio.

To everyone who makes me smile.

o júri

presidente

Doutor António Ferreira Pereira de Melo
Professor Catedrático do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro

Doutor Ferran Hurtado
Professor Catedrático do Departamento de Matemática Aplicada II da Universidade Politécnica da Catalunha, Espanha

Doutor Domingos Moreira Cardoso
Professor Catedrático do Departamento de Matemática da Universidade de Aveiro

Doutora Ana Paula Nunes Gomes Tomás
Professora Associada da Faculdade de Ciências da Universidade do Porto

Doutor Manuel Abellanas Oar
Professor Titular da Faculdade de Informática da Universidade Politécnica de Madrid, Espanha (Co-orientador)

Doutor Antonio Leslie Bajuelos Domínguez
Professor Auxiliar do Departamento de Matemática da Universidade de Aveiro (Orientador)

Acknowledgements

En primer lugar me gustaría agradecer a los Profesores Antonio L. Bajuelos y Manuel Abellanas por todo lo que me enseñaron sobre Matemática y Geometría Computacional, y además porque me recibieron siempre con una sonrisa! Eso ha sido muy importante para adaptarme. He aprendido mucho con vosotros y comprendo que todavía hay un mundo inmenso por descubrir en varias lenguas. En especial, muchas gracias por la amistad y confianza que depositaron en mi. Me gustaría agradecer también al Prof. Gregorio Hernández por su paciencia y por haberme traducido los nombres de los platos durante tantas comidas, al mismo tiempo que me enseñaba castellano e historia! No sería posible escribir esto hoy sin su ayuda. A todos los que conocí en España o que hablan castellano, ha sido estupendo compartir tantos momentos buenos con vosotros. Finalmente, a Mercè y Rodrigo por ayudarme con mi castellano mil y una veces.

E no nosso lindo português, queria agradecer à minha família, em particular à minha mãe pelo entusiasmo, ao meu pai pelo apoio “técnico” e não só, ao meu irmão pela paciência, ao meu avô pelo orgulho e ao Carlos pela incentivação quase diária. É realmente fantástico poder contar com o vosso apoio incondicional! Aos amigos que se ofereceram para ajudar e aos que foram obrigados (!). Não resisto a citar alguns que se envolveram mais directamente: André, Areal, Arturo, João, Mafalda, Pedro, Neto, Nuno e Sonia; e a todos os que me aturaram estoicamente nesta epopeia. Foi mais divertido porque estiveram por perto, obrigada ☺

To my fantastic family in London, I cannot thank you enough. Not only for your help, but also for your caring and genuine interest. You are such an important part of who I am and I love you all very much. Special thanks to Caroline, the best goddaughter ever!!
I’d also like to thank Calouste Gulbenkian Foundation whose short-term grant allowed me to start my studies in Madrid. Finally, I’d like to thank Nestlé for having created After Eights. This thesis just wouldn’t taste the same without them.

palavras-chave

Geometria Computacional, Problemas de Galeria de Arte, Cobertura, Alcance Limitado, Optimização Geométrica, Redes de Sensores, Vigilância, Itinerários Seguros, Diagramas de Voronoi.

resumo

Tal como o título indica, esta tese estuda problemas de cobertura com alcance limitado. Dado um conjunto de antenas (ou qualquer outro dispositivo sem fios capaz de receber ou transmitir sinais), o objectivo deste trabalho é calcular o alcance mínimo das antenas de modo a que estas cubram completamente um caminho entre dois pontos numa região. Um caminho que apresente estas características é um itinerário seguro. A definição de cobertura é variável e depende da aplicação a que se destina. No caso de situações críticas como o controlo de fogos ou cenários militares, a definição de cobertura recorre à utilização de mais do que uma antena para aumentar a eficácia deste tipo de vigilância. No entanto, o alcance das antenas deverá ser minimizado de modo a manter a vigilância activa o maior tempo possível. Consequentemente, esta tese está centrada na resolução deste problema de optimização e na obtenção de uma solução particular para cada caso.

Embora este problema de optimização tenha sido investigado como um problema de cobertura, é possível estabelecer um paralelismo entre problemas de cobertura e problemas de iluminação e vigilância, que são habitualmente designados como problemas da Galeria de Arte. Para converter um problema de cobertura num de iluminação basta considerar um conjunto de luzes em vez de um conjunto de antenas e submetê-lo a restrições idênticas. O principal tema do conjunto de problemas da Galeria de Arte abordado nesta tese é a 1-boa iluminação. Diz-se que um objecto está 1-bem iluminado por um conjunto de luzes se o invólucro convexo destas contém o objecto, tornando assim este conceito num tipo de iluminação de qualidade. O objectivo desta parte do trabalho é então minimizar o alcance das luzes de modo a manter uma iluminação de qualidade. São também apresentadas duas variantes da 1-boa iluminação: a iluminação ortogonal e a boa α-iluminação. Esta última tem aplicações em problemas de profundidade e visualização de dados, temas que são frequentemente abordados em estatística. A resolução destes problemas usando o diagrama de Voronoi Envolvente (uma variante do diagrama de Voronoi adaptada a problemas de boa iluminação) é também proposta nesta tese.

keywords

Computational Geometry, Art Gallery Problems, Coverage, Limited Range, Geometric Optimisation, Sensor Networks, Monitoring, Safe Routes, Voronoi Diagrams.

abstract

As the title implies, this thesis studies limited range coverage problems. Given a set of antennas (or any wireless device able to send or receive some sort of signal), the objective of the discussion that follows is to calculate the antennas' minimum range so that a path between two points within a region is covered by the antennas, a path known as a safe route. The definition of coverage is variable and depends on the applications. In some instances, for example, when monitoring is critical as in the case of fires or military, the definition of coverage necessarily involves the use of multiple antennas to increase the effectiveness of monitoring. However, it is also desirable to extend a network's lifespan, normally achieved by minimising the antennas' range. Therefore the focus of this thesis will be the resolution of this dual problem and an affective solution is offered for each case.

Although this question has been researched as an issue of coverage, it is also possible to establish a relation between coverage and illumination and visibility, known as Art Gallery problems. To conceptualise coverage problems as Art Gallery problems, all that is needed is to consider a set of lights instead of a set of antennas, which are subject to a similar set of restrictions. The main focus of the Art Gallery problems addressed in this thesis is 1-good illumination. An object is 1-well illuminated if it is fully contained by the convex hull of a set of lights, making this a type of quality illumination. The objective of the discussion that follows is therefore to minimise the lights' range whilst maintaining a quality illumination. Moreover, two variants of 1-good illumination are also presented: orthogonal good illumination and good α -illumination. The latter being related to data depth problems and data visualisation that are frequently used in statistics. The resolution of these problems using the Embracing Voronoi diagram (a variant of Voronoi diagrams adapted to good illumination) is also discussed in this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Coverage and Art Gallery Problems

On the 1^{st} of June 2009, Air France flight 447 was lost in the Atlantic Ocean en route to Paris (France) from Rio de Janeiro (Brazil). The alarm was raised when the plane left Brazilian-monitored airspace but failed to enter Senegalese-controlled airspace (see Figure 1.1).

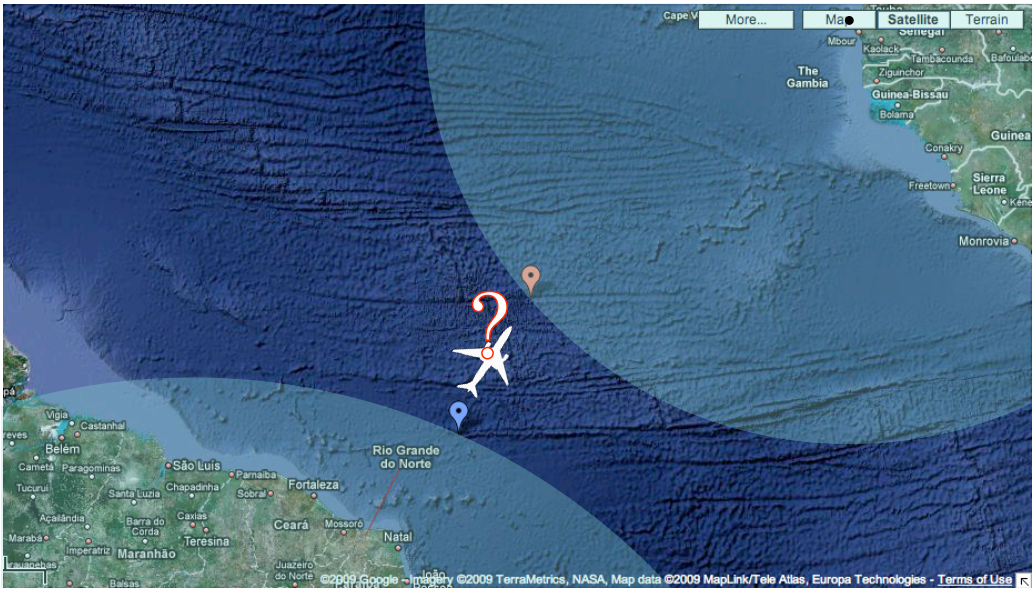


Figure 1.1: The blue marker showing the point where the plane left Brazilian-controlled airspace and the red where the plane should have entered Senegalese-controlled airspace. The monitored area is shown in light green.

The disappearance of the plane in an unmonitored zone, so called shadow areas, meant

that there was no way to know exactly where it fell, resulting in a potential search area that was clearly immense. Although in reality Brazilian radars are run by the military and their precise location is not easy to ascertain, for clarity in this example, the radars are assumed to be located at the airport in Rio de Janeiro and at the international airport in Dakar. The airspace controlled by both countries is shown in light green and the plane was to fly approximately 40 minutes in uncontrolled airspace between both markers. In a straight line, the blue and red markers are approximately 675km apart. Had the plane been monitored at the time, it would have been easier to locate its debris and concentrate the search for survivors in a particular area. This event makes it apparent that *coverage* has important applications. The problems presented in this thesis therefore aim to provide an effective coverage of a region or a path to be tracked, for instance, a flight route.

Although the previous example involves radars, this type of monitoring does not have to be necessarily performed by these devices. Coverage also includes sensors, antennas, routers and basically any device that is able to send or receive some sort of signal. Most of the applications in this field are associated with sensor networks (refer to Ning Xu [81] for a survey on the subject), resulting in an area of research that is under constant development as new technologies emerge. Apart from flight monitoring, coverage is used to solve a great diversity of problems; each presenting specific challenges that have to be addressed. Some applications do not need positioning information, that is, all that is needed to know is if someone or something has entered the network service area, for example, anti-burglary systems. The exact location of where a burglar entered a house is not necessarily important, what is crucial is the fact that the house has been invaded in order to take the appropriate measures. Similarly, weather forecast and habitat monitoring are other examples where positioning information is not needed. This type of coverage requires a network that monitors a region so that each point on such region is monitored at least by one device, also known as simple coverage. In relation to simple coverage, Abellanas et al. [18] and Mehta et al. [65] study routes on the plane that are always close to (or always far from) a given set of devices to achieve safe routes. These routes can be used in patient transport where the best route should never be too far from a medical centre. Also associated with simple coverage, Agnetis et al. [19] studied the monitoring of a river to avoid it being used by unauthorised boats, detect dangerous floating objects, etc.

Clearly the main coverage issue is centred around critical conditions that require reliable monitoring and immediate intervention like fires, disasters or leaking of toxic liquids/waste. The better the monitoring, the easier the location of the hazardous area and the rescuers' work. In such cases key questions are: how guaranteed is the detection of a critical event within a short-time interval? What is the extent of the region that is not monitored? Like the search for survivors in an emergency situation, these instances also require the tracking of people or

objects within a fully monitored area. These applications carry stronger coverage needs where failures cannot occur. What follows is therefore a discussion of multiple sensor coverage. If an area is monitored by multiple sensors, then it is secured that there are no shadow/breach areas even when a sensor fails. This is also the case for military surveillance, specifically, the ability to detect any intruder anywhere in the protected area (enemy tracking). In the case of flight monitoring, being covered by at least one radar is usually sufficient because the pilot is reporting the plane's exact position from time to time to the ground-based controllers. However, in the U.S. system, the airspace is often controlled by multiple radar systems since coverage may be inconsistent at lower altitudes due to high terrain or distance from radar facilities. A centre may require numerous radar systems to cover the airspace assigned to them, and may also rely on pilot position reports from aircrafts flying below the floor of radar coverage. The determination of an exact location is therefore a common issue.

The best known utility to solve this problem is the popular GPS (Global Positioning System), which was the first widely accessible positioning system to offer precise location data for any point on the planet. GPS gives the correct location of an object if it can be seen by at least three satellites. However, given that satellites operate using atomic clocks but that all commonly used terrestrial receivers do not, this strategy results in several issues including exact time measurement and the consequential need for error correction. For example, if the timing between emitter and receiver is out just by a thousandth of a second this translates into a ground distance error of almost 320km. There are a number of ways in which these errors can be corrected, for example, using an additional satellite (four satellites total) or, more commonly, using a variant of the original GPS known as the differential GPS that involves further input from two additional receivers. GPS has applications in agriculture, mass transit, urban deliveries, public safety, vessel and vehicle tracking, not to mention navigation and mapping. To summarise, it is clear that multiple coverage is used widely in our society but also raises a number of questions: what is the optimal number of devices that allow an accurate target detection? How accurate is this localisation in relation to the chosen number of devices? Further, an additional complication is that heterogeneous devices are needed for supervision and control applications each with distinct characteristics like processing power and communication properties. All these are challenging problems for any multiple network to cope with.

Each sensor can be placed almost anywhere within a specific region, depending on the region's topology, so coverage is really the discipline that measures the quality of the chosen device scheme. That is, coverage measures the quality of the solution obtained according to the characteristics of the application. The work by Meguerdichian et al. [64] is considered the breakthrough study that combined sensor networks and geometric tools. Several others,

including Boukerche et al. [26], Li et al. [57] and Mehta et al. [65], built on their ideas either by studying different versions of the coverage problem or by improving and proving previous results. Recently, Zhang et al. [82] focused on the decision problem and developed two localised algorithms to identify whether a sensor is on the boundary of the monitored area. Zhou et al. [85] addressed the problem of finding a minimum energy-cost coverage, where each sensor can vary its sensing and transmission radius. Das et al. [35] studied efficient location of base stations to cover a convex region when the base stations are inside the region. Although these studies are based on simple sensor coverage, there are also a few studies on multiple sensor coverage. For example, Efrat et al. [38] minimised the number of sensors in order to have each point "well covered". Good coverage means that each point is seen by three sensors that form a triangle containing the point or that each point is seen by two sensors that are separated by an angle of at least α. Furthermore, Zhou et al. [84] computed a small subset of sensors so that each point in the sensor network is at least covered by k sensors. To conclude, it can be seen that an effective solution found for a particular application is not necessarily good for another and previous research demonstrates that solutions that can cope with multiple issues are not easy to attain. The approach chosen in this thesis is known as best-case coverage. It is characterised as an attempt to locate the areas that are within reach of as many devices as possible. That is, to identify the "best" monitored areas of a given region or path whilst optimising the lifespan of the network by reducing the devices' sensing and transmission range as much as optimum [43, 57]. This optimisation problem will be further discussed in Chapters 4 and 5.

Even if Art Gallery problems seem very different from coverage problems, they are in fact intrinsically related. The connection dates back to 1973 when Victor Klee proposed the problem of calculating the minimum number of guards that are sufficient to cover the interior of an art gallery room with n walls. He posed this question after Vasek Chvátal asked him for an interesting geometric problem. Two years later, it was the same Chvátal who solved the problem in what has become known as the *Chvátal's Art Gallery Theorem*: $\lfloor \frac{n}{3} \rfloor$ guards are always sufficient and occasionally necessary to cover a simple polygon with n vertices [33]. However, the most popular proof of this theorem, mostly due to its simplicity, was proposed by Fisk [39] in 1978. This problem and its respective solution launched a whole new field of computational geometry: the Art Gallery problems. Interest in this area grew exponentially in the years that followed and this new branch of geometry became so popular that the first survey on the subject was written by J. O'Rourke in 1987 [69], less than ten years later. This keynote text was later complemented by Shermer [73], Asano et al. [20] and Urrutia [77] among others. To date, this area is still booming and it would appear that new variants continue to arise as technology advances.

As the concept of having an object monitored by a given sensor network is easily understood, the same happens with illumination or visibility. If a sensor is replaced by a light or a guard, that object is said to be illuminated or seen by such guard if it is within its range. Therefore, studies published under Art Gallery problems can be easily translated to coverage problems. For example, the main subject of Chapter 2 is 1-good illumination (also designated by $\triangle$-guarding [74] and well-covering [38]). A point q on the plane is said to be 1-well illuminated if it is within reach of at least three lights whose triangle contains q. If satellites are used instead of lights, it is easy to see that this concept is associated with multiple coverage and, particularly, to GPS applications. 1-good illumination has been generalised to t-good illumination [13, 14, 28] that can also be adapted to multiple coverage. The following works are just a few more examples of the relation between multiple coverage and Art Gallery problems. For Kaučič and Žalikm [51], k-guarding a surface patch is having it guarded by at least k guards. They proposed three heuristics (one of which is original) to find a minimum set of vertex guards that k-guard the whole terrain. On the other hand, Belleville et al. [23] consider that k-guarding a polygon P means that it is possible to find a set of guards on the edges of P (at most one guard per edge) such that every point on P is visible to at least k guards. To conclude and as in previous examples, the problems addressed in Chapter 2 are written as Art Gallery problems but can be easily converted to coverage as outlined above.

Most of the solutions found by the authors previously mentioned, either to solve coverage or Art Gallery problems, make use of Voronoi diagrams. As O'Rourke states [70]: "in a sense, the Voronoi diagram is a structure that records all the needed information on proximity to a set of points or other objects". Voronoi diagrams were considered as early as 1644 by René Descartes, but they were first formally introduced by Dirichlet [36] and Voronoi [78, 79] in the beginning of the XX century. Consequently, it has been called *Dirichlet tessellation* but the designation that prevails is *Voronoi diagram*. Although it is not by any means a new concept, it keeps inspiring researchers to this day. Not only there are conferences and symposiums based solely on Voronoi diagrams but even art exhibitions. An exhaustive and unified exposition of the mathematical and algorithmic properties of Voronoi diagrams can be found on a survey by Aurenhammer and Klein [21]. A typical example of a problem that is solved using Voronoi diagrams is facility location. Supposing a new shop is about to open in a place where there are other similar shops, the best place to locate this new shop is the point that is as far as possible to the other shops. In Voronoi terms, this would be expressed as the new shop being located where the distance to the closest of the existent shops is as large as possible. The resulting location being the centre of the largest empty circle (empty of other shops) that is a point on the Voronoi diagram. This structure will be used throughout this thesis and is shown to inspire three original variations: the *Embracing Voronoi diagram*, the *Orthogonal E-Voronoi*

diagram and the *Coverage Voronoi diagram*. The first two are introduced in Chapter 3 and the third in Chapter 5.

1.2 Summary of Published Work

The results obtained and presented in this thesis are divided into four chapters. Chapter 2 studies good illumination problems, in particular, the 1-good illumination concept. An object x is 1-well illuminated by a set of lights if, and only if, there is at least one light illuminating x in every open half-plane with x on its border. In this way, the larger the number of lights in every open half-plane with point q on its border, the better the illumination of the points on the plane that surround q. Consequently, this subject falls into the category of quality illumination. As previously mentioned, 1-good illumination can also be found under the designations of $\triangle$-guarding [74] and well-covering [38]. Chapter 2 presents a set of optimisation problems whose objective is to minimise the lights' range in order to maintain a given object 1-well illuminated. The results involving points and line segments were published in [1, 2, 7]. This chapter also presents two generalisations of this type of good illumination: the orthogonal good illumination and the good α-illumination. As before, these variants aim to optimise the lights' range to orthogonally well illuminate or well α-illuminate a given point. A new structure associated with good α-illumination called the *α-embracing contour* is also introduced in Chapter 2 and is associated with data visualisation. Results on both variants are also discussed in several publications [5, 6, 7, 8, 16, 17].

Chapter 3 is devoted to a new variant of Voronoi diagrams named the Embracing Voronoi diagram. This diagram arose from the concept of 1-good illumination and merges the notions of proximity and convex dependency. If a point q on the plane moves continuously or quickly changes from one location to another, there is the need to recompute the lights' minimum range in order to keep q 1-well illuminated. The Embracing Voronoi diagram is the structure that results from preprocessing the location of a set of lights in order to achieve a quicker solution. That is, a geometric structure that provides a basis to efficiently recalculate the lights' minimum range to keep any moving points 1-well illuminated. There are two papers on this subject, one was published in [3] and the other one in [4]. In this last paper there are also some results on the closest embracing number, that is, the number of lights needed to 1-well illuminate a given object, also associated with data depth problems.

As previously mentioned, results on 1-good illumination cannot be dissociated from coverage problems. Therefore, the final two chapters of this thesis, Chapters 4 and 5, study coverage problems and in particular multiple sensor coverage. Chapter 4 is devoted to coverage using

at least two devices (sensors, antennas, etc.) whereas Chapter 5 does not have any restriction on the number of devices. The geometric optimisation problem solved in Chapter 4 aims to minimise the power transmission range of the devices so that a path on a region or indeed the whole region, is within range of at least two devices. Different versions of this problem are obtained for different types of regions, for example, a line segment, a polygonal region or even the whole plane. The results on the variant of this problem that uses a planar graph (or street graph) were published in [12], whereas the results involving a path between two points on the plane were published in [9]. This type of coverage applied to a polygonal region was published in [11].

Chapter 5 is divided into two parts. The objective of the first part is to decide which is the maximum coverage of a path within a polygonal region. This problem is solved for four distinct regions: a line segment, a planar graph, a polygonal region and the whole plane. The solutions proposed for the four types of regions were published in [10]. The second part of this chapter is based on a more restrictive definition of coverage: a point q on the plane is said to be $\frac{\pi}{2}$-covered by two devices if the angle between q and those devices is at least $\frac{\pi}{2}$. This restriction ensures that the devices surround their service area uniformly. It is shown how to optimise the device's range to $\frac{\pi}{2}$-cover a point and how to construct the $\frac{\pi}{2}$-covered region and its contour. Moreover, the Coverage Voronoi diagram is also introduced in this second part as an important geometric tool to solve this type of optimisation problems. This structure is associated with Embracing Voronoi diagrams.

All the problems and solutions presented in this thesis use the Euclidean distance and it is assumed that any set of points on the plane is in general position, except where otherwise stated. The following chapters are intended to be self-contained, although the inter-relatedness of some of the subjects addressed in this thesis rendered it impossible to avoid cross-references altogether. The necessary notation and problem formulation are introduced in each chapter where they are required, and consequently there is no general introductory section filled with notations and previous results. Nevertheless, it is assumed that the reader is acquainted with the basic concepts of Computational Geometry. A list of the notations introduced in this thesis can be consulted in page 155.

Chapter 2

Good Illumination

This chapter focuses on good illumination problems. An object x is 1-well illuminated by a set of lights if, and only if, there is at least one light illuminating x in every open half-plane with x on its border. In this way, the larger the number of lights in every open half-plane with x on its border, the better the illumination of the surroundings of x. This chapter presents a set of optimisation algorithms whose objective is to minimise the lights' range in order to maintain a given object (point, line segment or polygonal line) 1-well illuminated. There follows two variations of 1-good illumination: the orthogonal good illumination and the good α-illumination. The algorithms involving these variants aim to optimise the lights' range to orthogonally well illuminate or well α-illuminate a given object. A new data visualisation tool called the α-embracing contour is also introduced in this chapter and is associated with good α-illumination. Finally, some of these algorithms were implemented in Java and those implementations are shown in dedicated sections.

2.1 Introduction

Since Art Gallery problems have been a very popular branch of Computational Geometry, new variants are constantly arising. As a result, some surveys on the subject (O'Rourke [69], Shermer [73], Urrutia [77], etc.) have become keynotes texts in Computational Geometry. This chapter is focused on illumination and visibility problems, which play a major role in Art Gallery problems [20]. According to Ghosh [41], a point p on the plane is said to be illuminated by a light s if the line segment connecting both, $\overline{sp}$, does not cross any obstacle (see Figure 2.1(a)). This concept has been adapted to different variants of illumination, for example, Urrutia [77] studied floodlights, which are lights that illuminate within a given angle

and can rotate around their apices. The following discussion considers another illumination concept whose definition is aimed at quality illumination: the good illumination, which is defined below.

Definition 2.1 ([28]) *Let S be a set of lights on the plane. A point q on the plane is t-well illuminated by S if and only if there are at least t lights of S illuminating q in every open half-plane with q on its border.*

This definition has been studied for $t = 1$, $t = 2$ and $t = 3$ by Canales et al. [13, 14, 28]. The motivation behind t-good illumination is the fact that, in some applications, it is not sufficient to have just a given object illuminated. It is also necessary that some of its neighbourhood is illuminated as well [38]. In this way, the larger the number of lights in every open half-plane with point q on its border, the better the illumination of the points that surround q (see Figure 2.1(b)). Consequently, this subject falls into the category of quality illumination. This chapter is based on 1-good illumination, which can also be found under the designations of $\triangle$-guarding [74] or good coverage [38].

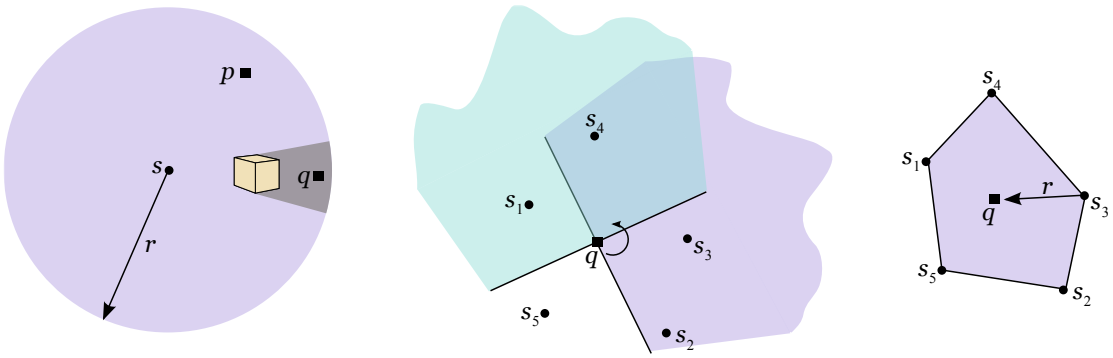


Figure 2.1: (a) Point p is illuminated by light s with illumination range r but q is not as the orange box casts a shadow on q. (b) Point q is 1-well illuminated since there is always a light in every open half-plane with q on its border. (c) Point q is inside CH(S) (shown in purple) and MER(q) = $d(s_3, q) = r$.

Given a set S of n lights, let CH(S) denote the convex hull of S, int(CH(S)) the interior of CH(S) and $d(q, s_i)$ the Euclidean distance between point q and light $s_i \in S$. The following designations were introduced by Chiu and Molchanov [32]. Given a set $S' \subseteq S$, if point q is inside CH(S') then S' is an *embracing set* for q. According to Definition 2.1, it is not difficult to see that if $q \in$ int(CH(S)), then q is 1-well illuminated and vice versa (see Figures 2.1(b) and 2.1(c)). Suppose that the minimum range of S that 1-well illuminates q is given by $d(s_c, q)$, then s_c is the closest light to q such that q is inside CH(S'), $S' = \{s_i \in S : d(s_i, q) \leq d(s_c, q)\}$.

Such light is called a *closest embracing site* for q. Any embracing set S' for q formed by s_c and other lights of S that are closer to q than s_c is a *closest embracing set* for q; and a closest embracing set formed by three lights defines a *closest embracing triangle* that is denoted by CET(q). In Figure 2.1(c), light s_3 is a closest embracing site for q and set $\{s_1, s_2, s_3, s_4, s_5\}$ is an embracing set for that point. The minimum illumination range needed to 1-well illuminate q is called the *minimum embracing range* and is calculated as the distance between q and its closest embracing site. Given an object x, the minimum embracing range of S that 1-well illuminates x is denoted by $\text{MER}_S(x)$ or $\text{MER}(x)$ if S is clear from the context.

Although Definition 2.1 adopts the classical notion of visibility that allows unlimited visibility along an unobstructed line of sight, this chapter is based on a more realistic and restrictive definition introduced by Ntafos [68]. Assuming there are no obstacles on the plane, two points p and q on the plane are *r-visible* to each other if the straight line distance between p and q is shorter than r. It does not seem necessary to consider a limited field of view since conventional rotating cameras can easily sweep the whole 360° field of view. However, video cameras and robot vision systems have severe visibility range restrictions. Such restrictions are largely a result of the low resolution images most panoramic cameras obtain, which in turn are a product of the method used to store each of the video's frames [52]. Therefore, adopting a restricted notion of visibility is a realistic and reasonable approach to this problem. Let $D(s_i, r)$ be the disc of radius r centred at light $s_i \in S$. If there are no obstacles on the plane, the r-visible area for s_i is precisely $D(s_i, r)$. In other words, only the points on the plane enclosed by $D(s_i, r)$ are illuminated by s_i. Given a set $S' \subseteq S$, the region on the plane that is illuminated by every light of S' with range r is given by $A_r(S') = \bigcap_{s_i \in S'} D(s_i, r)$.

Definition 2.2 *Let S be a set of lights with range $r \in \mathbb{R}^+$, then a point q on the plane is 1-well r-illuminated by S if there is a ternary set $S' \subseteq S$ such that $q \in \{A_r(S') \cap \text{int}(\text{CH}(S'))\}$.*

Definition 2.2 is a combination of 1-good illumination and limited illumination range (see Figure 2.2). Since this is the only definition studied in the following sections, 1-good r-illumination will be simply designated as 1-good illumination. Given a set S of n lights on the plane, the objective of this chapter is to *calculate the minimum illumination range needed to 1-well illuminate a given object*. Chan et al. [29] showed how to construct the Nearest Neighbour Embracing Graph [32] in optimal time. In order to do that, they had to find a closest embracing site for each of a set of points. Since the construction of the whole graph takes $\Theta(n^2)$ time, the embracing range for a single point on the plane can be calculated in $\mathcal{O}(n)$ time. This graph has interesting properties and it is even associated with coverage problems as demonstrated by Zhang et al. [82], though this will be further discussed in Chapter 4. The

work by Smith and Evans [74] was already referred as they presented an alternative designation to 1-good illumination. Given two simple polygons P and Q, and assuming Q is contained in P, the authors proposed a polynomial time algorithm to find a minimum set of vertex guards on P that 1-well illuminates Q. To this end, Q is assumed to have a transparent boundary, otherwise the problem becomes NP-hard (meaning there is an NP-complete problem that can be reduced to this problem in polynomial time). The following study is another example of the fact that 1-good illumination can be translated to coverage. Efrat et al. [38] minimised the number of sensors in order to have given region well covered. Good coverage means that each point is seen by three sensors that form a triangle containing the point or that each point is seen by two sensors that are separated by an angle of at least α. Translating back to illumination problems and using only their first definition, the authors showed how to minimise the number of lights to 1-well illuminate a given region.

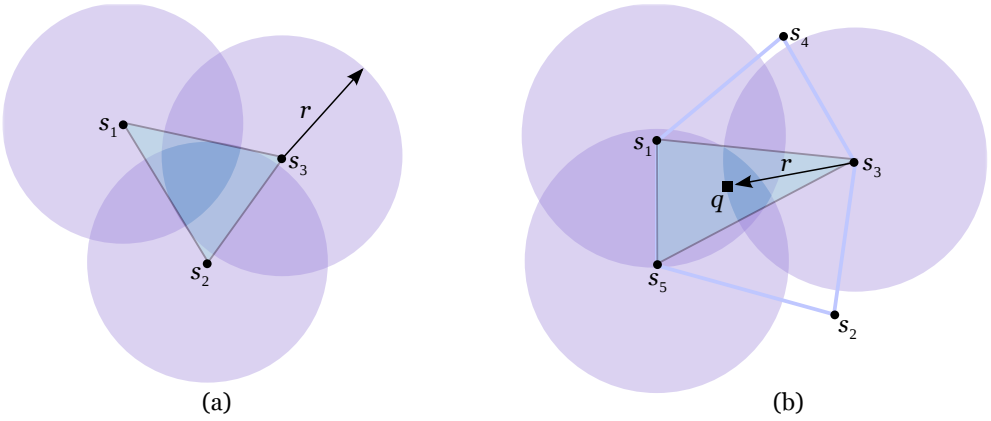


Figure 2.2: (a) Dark blue region is 1-well illuminated by lights s_1, s_2 and s_3 with range r. (b) Point q is 1-well illuminated by S with minimum embracing range $r = d(s_3, q)$.

Some of the algorithms introduced in this chapter were implemented in Java using the platform Netbeans 4.0 IDE (integrated development environment) developed by Sun Microsystems. The applications that resulted from these implementations are illustrated in Sections 2.2.2 and 2.5.2 that follow the respective algorithms. However, the images presented in those sections show a more recent version since the applications are currently running in version 6.5. There will not be a detailed analysis of any of the implementations since these are not relevant to the objectives of this thesis. All the algorithms were implemented mainly for visualisation purposes and debugging. The structure of this chapter is introduced below.

Throughout this chapter it is assumed that there are no obstacles on the plane and that every light has the same illumination range. Given a set of n lights, two algorithms to calculate the minimum embracing range to 1-well illuminate a point q are presented in Section 2.2. The first runs in $\mathcal{O}(n \log n)$ time and although this is not optimal, it automatically finds a closest

embracing triangle for q. Not aware of the work by Chan et al. [29], we developed a linear algorithm in the same year to calculate MER(q) [7]. Such algorithm is presented in Section 2.2.1 and it is interesting in its own right. In Section 2.3, the objective is to calculate the minimum embracing range to 1-well illuminate line segment $\overline{pq}$ that connects points p and q. The first approach computes MER($\overline{pq}$) in $\mathcal{O}(n^3)$ time and it can be generalised to polygonal lines. The second algorithm, introduced in Section 2.3.2, makes use of the Parametric Search [61, 62] and runs in $\mathcal{O}(n^2)$ time. Orthogonal good illumination is briefly introduced in Section 2.4 and good α-illumination is presented in Section 2.5. A linear algorithm is proposed to calculate the minimum embracing range to well α-illuminate a point and find a closest embracing set for it. Good α-illumination and 1-good illumination are related and this is demonstrated by a proposition in Section 2.5. Section 2.5.3 introduces the α-embracing contour, which is a data visualisation tool. Contours have applications in quality illumination, and in particular, they can be applied to good α-illumination.

2.2 Minimum Embracing Range to 1-Well Illuminate a Point

Let S be a set of n points on the plane that represent the location of n lights. This section has two main goals. The first is to calculate the minimum embracing range of S to 1-well illuminate a point q, MER(q). The second goal is to find a closest embracing triangle for q (see Figure 2.3(a)). As previously mentioned, Chan et al. accomplished the first goal in $\mathcal{O}(n)$ time [29]. However, we were not aware of their work while our research was being carried out. Therefore, this section is solely devoted to the algorithms we developed to achieve both goals, since we believe them to be interesting in their own right. The first calculates MER(q) and a CET(q) in $\mathcal{O}(n \log n)$ time. This complexity is not optimal, but it is worth a detailed description considering its geometrical approach based on a property of closest embracing triangles. The second algorithm runs in linear time and is discussed in Section 2.2.3.

2.2.1 An $\mathcal{O}(n \log n)$ Algorithm

The following algorithm calculates MER(q) as a result of its search for a closest embracing triangle for q, which is its main goal. Therefore its strategy is directed at finding ternary embracing sets for q, instead of working with all lights simultaneously. Nevertheless, this approach is hampered by the fact that usually there is more than one closest embracing triangle per point. This is simplified by the following proposition that states that the closest light to q is always a vertex of at least one closest embracing triangle for q.

Proposition 2.1 *Let S be a set of n lights and q a point on the plane. If s_n is the closest light of S to q, then s_n is a vertex of at least one closest embracing triangle for q.*

Proof: Let $\triangle(s_i, s_j, s_k)$ be a closest embracing triangle for q formed by lights s_i, s_j and s_k of S. Assume that $s_n \in S \backslash \{s_i, s_j, s_k\}$ is the closest light of S to q. This proof is split into two cases:

(a) If $s_n \in \triangle(s_i, s_j, s_k)$, then triangulate $\triangle(s_i, s_j, s_k)$ using s_n as a vertex (see Figure 2.3(b)). As a result, q must lie on one of the following: $\triangle(s_i, s_j, s_n)$, $\triangle(s_i, s_k, s_n)$ or $\triangle(s_j, s_k, s_n)$. Without loss of generality, suppose $q \in \triangle(s_j, s_k, s_n)$. This assumption implies that light s_i cannot be the closest embracing site for q, and so it is either light s_j or light s_k. Therefore, $\triangle(s_j, s_k, s_n)$ is a CET(q).

(b) If $s_n \notin \triangle(s_i, s_j, s_k)$, then without loss of generality suppose that $\overline{s_n s_j}$ intersects the edge $\overline{s_i s_k}$ of $\triangle(s_i, s_j, s_k)$ (see Figure 2.3(c)). Consequently, q either lies on $\triangle(s_i, s_j, s_n)$ or $\triangle(s_k, s_j, s_n)$. Since s_n is closer to q than any other light of S and $\triangle(s_i, s_j, s_k)$ is a CET(q), if q is inside $\triangle(s_i, s_j, s_n)$ then this triangle also is a CET(q). For the same reasons, if q is inside $\triangle(s_k, s_j, s_n)$, then $\triangle(s_k, s_j, s_n)$ is a CET(q).

In conclusion, s_n is a vertex of at least one closest embracing triangle for q. □

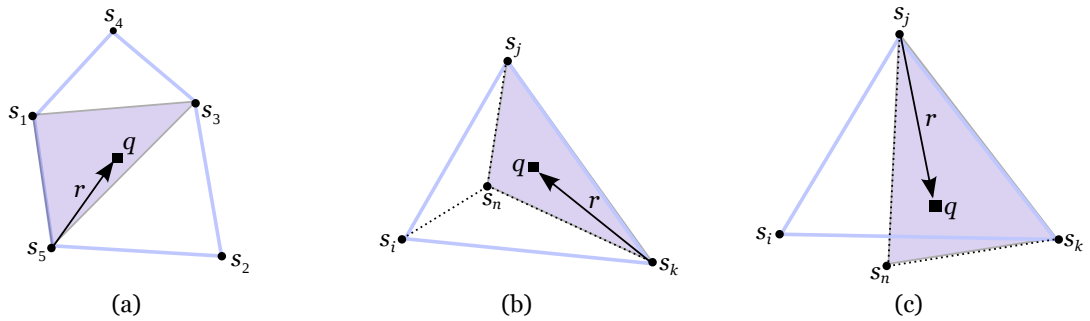


Figure 2.3: The minimum embracing range for point q is given by r. (a) Set $\{s_1, s_3, s_5\}$ is a closest embracing triangle for q. (b) Point q is inside $\triangle(s_i, s_j, s_k)$ and $\triangle(s_n, s_j, s_k)$. (c) Light s_n is not inside $\triangle(s_i, s_j, s_k)$ but $q \in \text{int}(\triangle(s_n, s_j, s_k))$.

This proposition is the basis for the following algorithm. Taking advantage of the fact that the closest light to q is a vertex of at least one closest embracing triangle for q, the algorithm only has to search the other two vertices of such triangle. In order to be 1-well illuminated, point q has to be inside the triangle, therefore each missing vertex is found on one side of the line that connects q to its closest light. Assuming that CCW order means anticlockwise, the following pseudo-code outlines this algorithm.

ALGORITHM **Minimum Embracing Range I**

INPUT: Set S of n lights, point q
OUTPUT: CET(q) and MER(q)

1. If $q \notin \text{int}(\text{CH}(S))$ then q cannot be 1-well illuminated by S;
2. Find $s_n \in S$, the closest light of S to q;
3. $L = \{s_i \in S : s_i \text{ is on the left of } \overrightarrow{s_n q}\}$, $R = S \backslash L$;
4. Sort the lights in sets L and R in CCW order around q;
5. For $i = 0$ to $|L| - 1$ do
   $flag(l_i) \leftarrow \{l_j : l_j \text{ is the closest light of } \{l_0, \ldots, l_i\} \text{ to } q\}$
6. Rotate $s_n q$ centred at q until $s_i \in S$ is found
   If $s_i \in L$ then $l_b \leftarrow flag(s_i)$
   Otherwise $d \leftarrow \max\{d(s_i, q), d(l_b, q)\}$ and if $d < \text{MER}(q)$
   Then MER(q) $\leftarrow d$, CET(q) $\leftarrow \{s_n, l_b, s_i\}$
   Repeat until the last light of R is found.

There is an example of the previous algorithm in Figure 2.4. Assume that $s_n \in S$ is the closest light to q. As line $s_n q$ rotates, the algorithm evaluates every triangle that has a vertex at s_n and contains q. The lights on the left of $\overrightarrow{s_n q}$ are flagged in the beginning of the algorithm to indicate the closest light of L to q found so far (see Figure 2.4(a)). The lights on the right of $\overrightarrow{s_n q}$ are tested as the line rotates and each new triangle only replaces the previous best if the range to 1-well illuminate q using the current triangle is shorter. For example, in Figure 2.4(c) triangle $\triangle(l_0, s_n, r_1)$ does not replace $\triangle(l_0, s_n, r_0)$ because $d(r_0, q) < d(r_1, q)$. On the contrary, in Figure 2.4(d) triangle $\triangle(l_0, s_n, r_0)$ is replaced by $\triangle(l_0, s_n, r_2)$ since $d(r_0, q) > d(r_2, q)$. The algorithm halts when the last light of R is found and then it outputs a closest embracing triangle for q and MER(q). The following theorem states the temporal complexity of this algorithm.

Theorem 2.1 *Given a set S of n lights and point q on the plane, the "Minimum Embracing Range I" algorithm calculates the minimum embracing range of q and a closest embracing triangle for q in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Proof: Although there are faster algorithms to decide if q is inside CH(S), this can be verified in $\mathcal{O}(n \log n)$ time by constructing CH(S) since it does not worsen the algorithm's

final complexity. Finding the closest light of S to q, s_n, and dividing all the lights in two sets takes linear time. Sorting the lights of each set in anticlockwise order takes $\mathcal{O}(n \log n)$ time. Rotating the line $s_n q$ around q and updating the current triangle (if necessary) requires linear time since the lights are already sorted. Therefore, a CET(q) and MER(q) are found in $\mathcal{O}(n \log n)$ time once the algorithm halts. Regarding space complexity, there is the need to store n lights and their order around q, which takes $\mathcal{O}(n)$ space. □

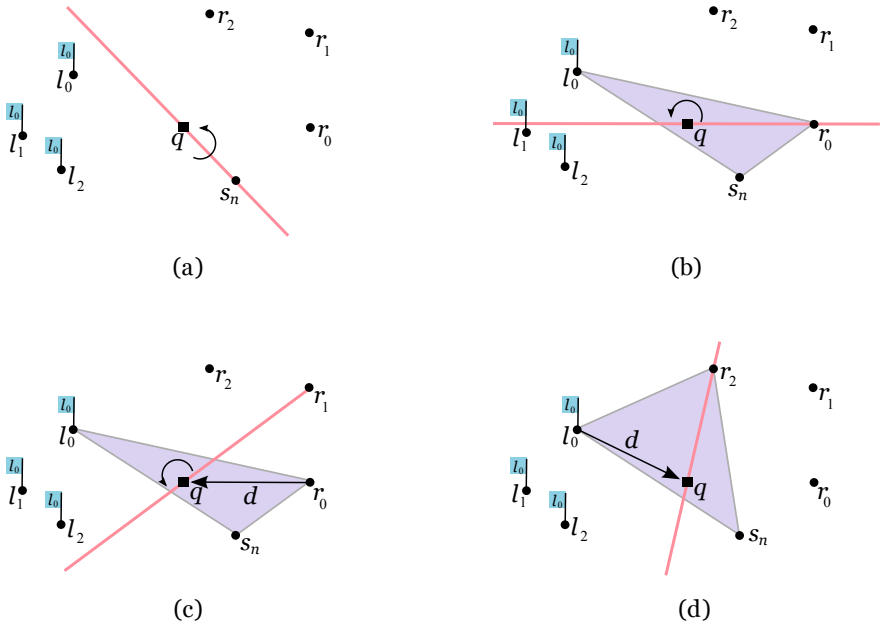


Figure 2.4: (a) Line $s_n q$ divides the lights in sets $L = \{l_0, l_1, l_2\}$ (whose lights are flagged) and $R = \{r_0, r_1, r_2\}$. (b) Light r_0 is reached and $\triangle(l_0, s_n, r_0)$ is a candidate to be a CET(q). (c) Light r_1 is reached but $\triangle(l_0, s_n, r_1)$ is not better than $\triangle(l_0, s_n, r_0)$. (d) The last light of R is reached, and so CET(q) = $\triangle(l_0, s_n, r_2)$ and MER(q) = d.

2.2.2 Implementation

The previous algorithm was implemented in Java and the resulting application was named "MER and CET". It starts with a white panel where the user can click to add lights (see Figure 2.5). Each light is represented by a blue dot and `Place point` is the only active button. After pressing that button, the next click on the panel will show a red dot that is the location of the query point (see Figure 2.6). Button `Divide` is enabled after the query point has been placed. Lights can be added at any time before button `Divide` is pressed.

When the user is satisfied with the location of both the lights and query point, button `Divide` should be pressed to start the algorithm. This button initiates a method that divides the lights on the left and right of the line that connects the query point to its closest light.

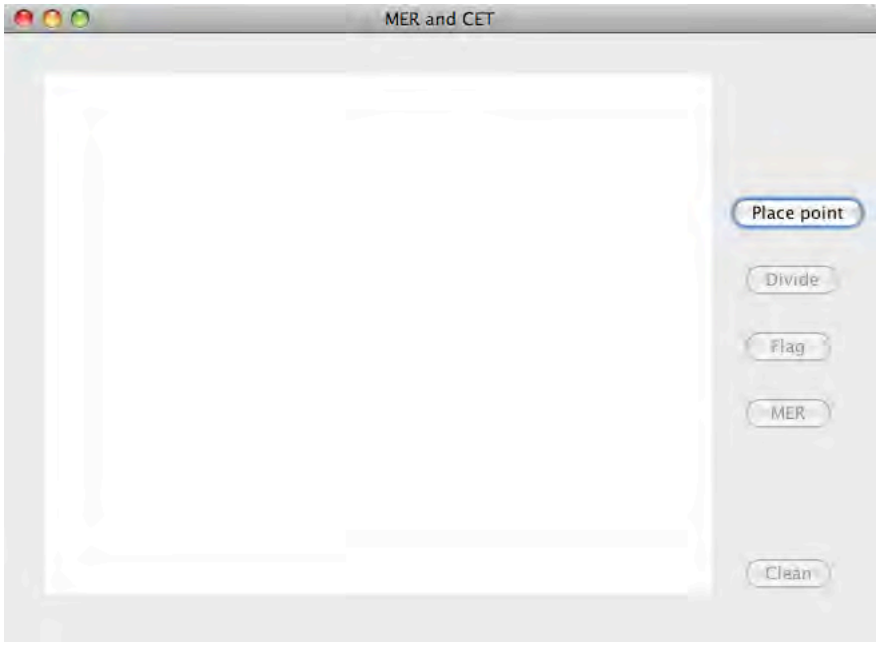


Figure 2.5: Aspect of the application once it is started.

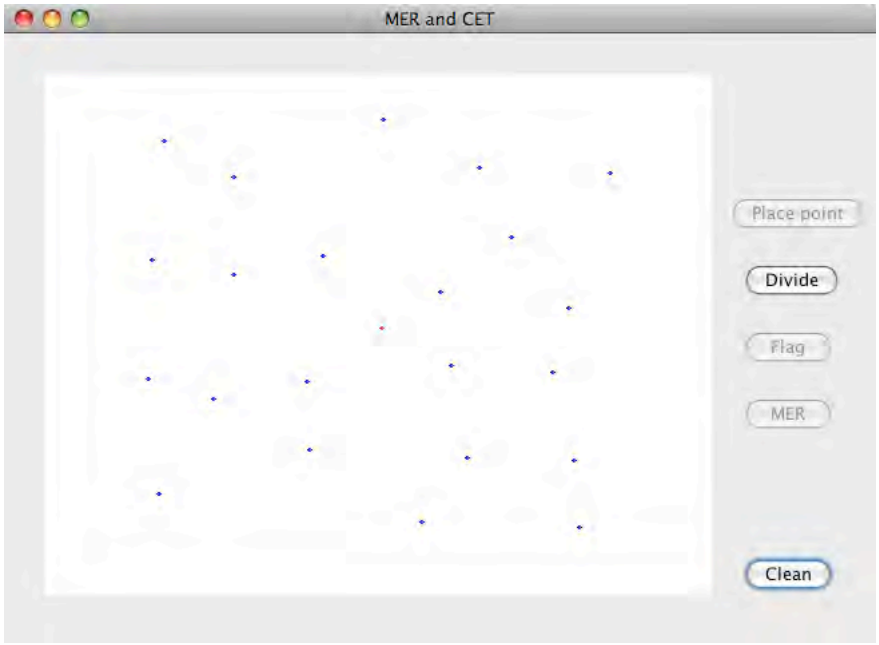


Figure 2.6: The blue dots are lights and the red dot is the query point.

Moreover, it also sorts the lights in anticlockwise order. Afterwards, button `Flag` becomes enabled (see Figure 2.7). Once button `Flag` is pressed, each light on one side of the line is assigned to the closest light on that side to the query point (see Figure 2.8). After this task is completed, the last button becomes enabled.

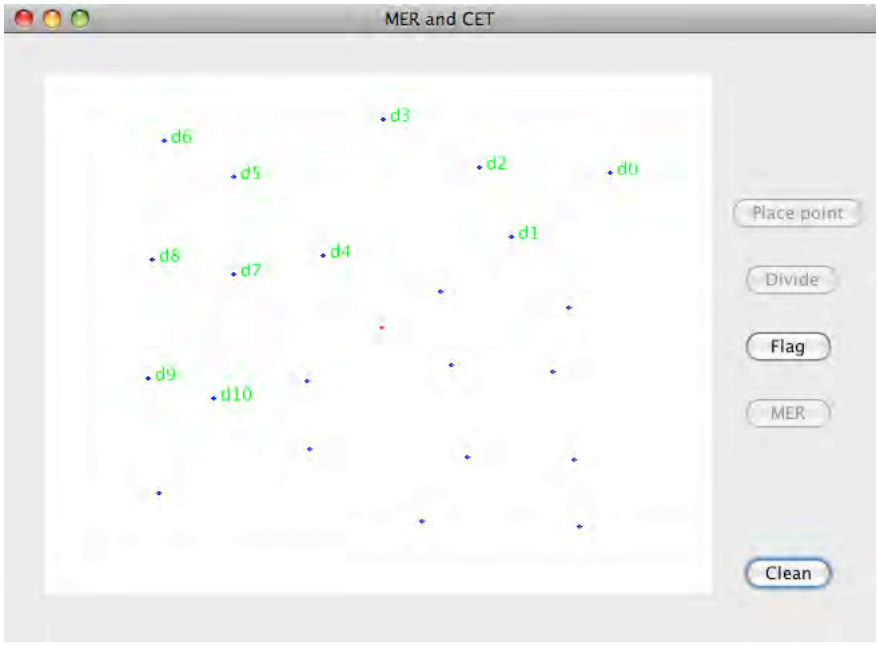


Figure 2.7: All the lights are sorted in anticlockwise order but the application only shows the ordering of the ones on the left of the line that connects the query point to its closest light.

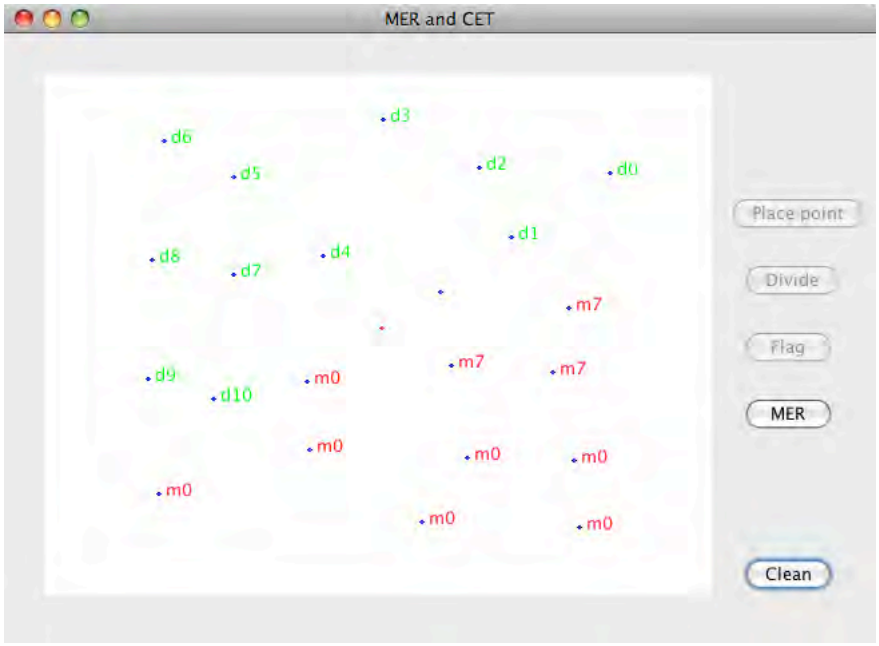


Figure 2.8: The lights on the right side are labelled in red to indicate the closest light on that side to the query point.

After button `MER` is pressed, the method to rotate the line centred at the query point is activated. Once the rotation is finished, the minimum embracing range is shown on the output window (see Figure 2.9) and a closest embracing triangle for the query point is represented in

a solid yellow line on the main panel (see Figure 2.10). A closest embracing site for the query point is labelled "fml".

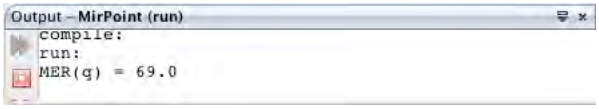


Figure 2.9: The minimum embracing range to 1-well illuminate the red dot is 69.

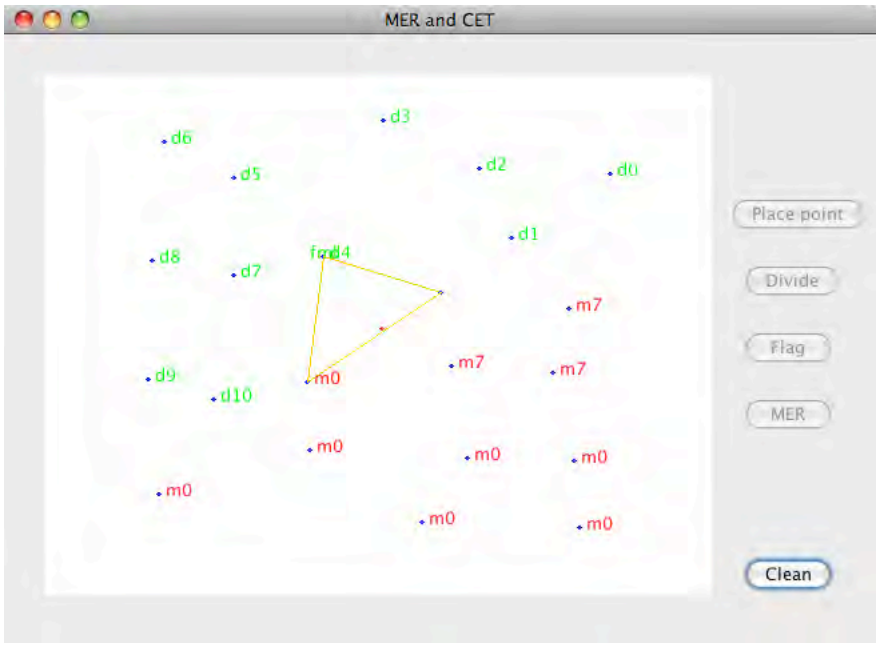


Figure 2.10: A closest embracing triangle for the red dot is represented in a solid yellow line. The light labelled "fml" is the closest embracing site for the red dot.

If the query point is not inside the convex hull of the lights, then the application outputs the message "The point is not inside CH(S)". Button `Clean` clears the white panel and restarts the algorithm. Since all the implement methods are pretty straightforward, the only one which is worth a remark is the method "Left". This method decides if a light is on the left of the line connecting the query point to its closest light and it was implemented using the coordinates of three points: the light to be tested, the query point and its closest light. These tree points form a triangle. Using the determinant that calculates double the area of such a triangle, the location of the light can be determined depending on whether the determinant is positive or negative [70]. If the triangle's vertices are read in anticlockwise order, then that determinant is positive. This method can also be used to detect if two line segments intersect each other or if they are collinear.

2.2.3 A Linear Algorithm

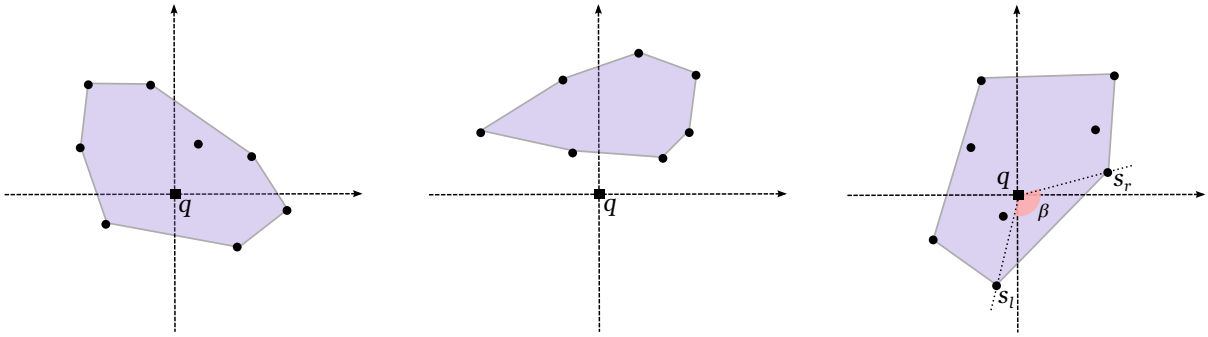
As previously explained, the closest embracing site for a point q can be obtained in linear time using the algorithm introduced by Chan et al. [29]. The following algorithm is similar to theirs and has the same temporal complexity, but it also has the advantage of finding a closest embracing triangle for q.

ALGORITHM **Minimum Embracing Range II**

INPUT: Set S of n lights, point q
OUTPUT: MER(q)

1. $R \leftarrow \{d(s_i, q) : s_i \in S\}$;
2. Perform a binary search on $R = \{r_0, \ldots, r_{n-1}\}$:
   For the median distance $r_i \in R$ do
   $S' \leftarrow \{s_i \in S : d(s_i, q) \leq r_i\}$
   If $q \in \text{int}(\text{CH}(S'))$
   Then proceed the search on $R \leftarrow \{r_j \in R : r_j \leq r_i\}$
   Otherwise proceed the search on $R \leftarrow \{r_j \in R : r_j > r_i\}$
3. The final distance is MER(q);
4. Find a CET(q) using the light that determines MER(q).

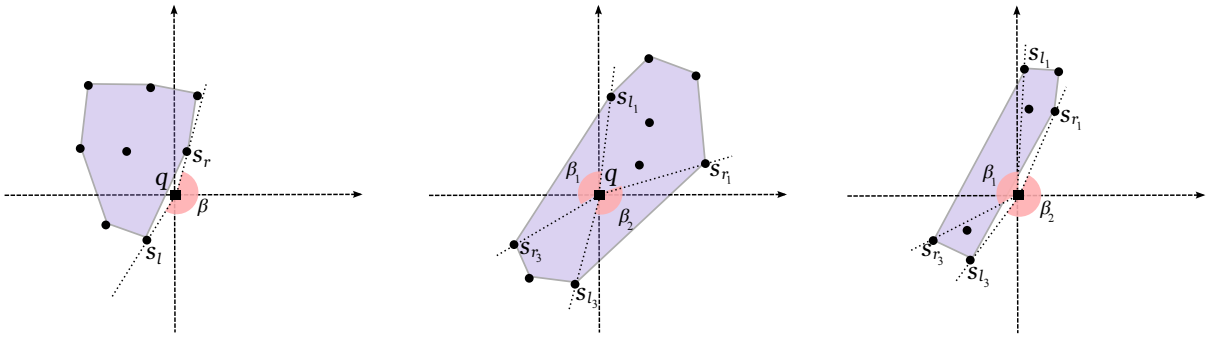
Given a set $S' \subseteq S$, the main difficulty to carry out this algorithm is to decide whether point q is in interior of CH(S') in linear time. There are several ways to do this, and obviously none of them constructs the convex hull. One solution explores the representation of the convex hull as an intersection of several half-planes. Sacristán [71] proved in her thesis that deciding if a point belongs to a convex hull defined by an intersection of half-planes takes $\Theta(n)$ time. The linear time solution chosen by Chan et al. [29] uses of a projection of the lights on a unit disc and then measures the angle of the wedge containing such projections. Our solution is close to the one suggested by Megiddo [63] and also employs angles. Suppose the lights of S' are divided through four quadrants with origin at point q. This decision problem can be split into four cases: there is a light on every quadrant, there are precisely two empty adjacent quadrants, there is only one empty quadrant or there are exactly two non-adjacent empty quadrants. The first and second cases are straightforward, q is inside CH(S') in the first case and outside in the second (see Figures 2.11(a) and 2.11(b)). In the other two, both situations can occur.



(a) (b) (c)

Figure 2.11: (a) Point q is inside the purple convex hull. (b) Point q is not inside the purple convex hull. (c) Point q is inside the purple convex hull since $\beta < \pi$.

If there is only one empty quadrant then there is the need to compute the outermost lights of set S'. Let s_l and s_r be the leftmost and rightmost lights of S', respectively. If the empty angle $\measuredangle(s_l, q, s_r) \geq \pi$ then q is not inside $\mathrm{CH}(S')$, otherwise S' is an embracing set for q (see Figures 2.11(c) and 2.12(a)). In the last case, there are precisely two empty quadrants that are opposite to each other (see Figures 2.12(b) and 2.12(c)). Therefore, two pairs of outermost lights have to be found on each empty quadrant. If both empty angles are smaller than π, then q is in the interior of $\mathrm{CH}(S')$. Otherwise q is outside $\mathrm{CH}(S')$. In any other situation, q is not inside $\mathrm{CH}(S')$.



(a) (b) (c)

Figure 2.12: (a) Point q is not inside the purple convex hull because $\beta \geq \pi$. (b) Point q is inside the purple convex hull since $\beta_1 < \pi$ and $\beta_2 < \pi$. (c) Point q is not inside the purple convex hull because $\beta_2 \geq \pi$, though $\beta_1 < \pi$.

Once the algorithm is over, $\mathrm{MER}(q)$ and a closest embracing site for q, $s_c \in S$, are found. According to the algorithm's pseudo-code, a closest embracing triangle for q can be found using light s_c. Such method is described in the following. Consider the disc centred at q of radius $\mathrm{MER}(q) = d(s_c, q)$ and the line segment connecting q to s_c. Observe that such disc has

a semicircle that only contains light s_c (see the first image in Figure 2.13). This is due to the fact that s_c is a closest embracing site for q. Line qs_c divides the lights inside the disc in two sets and any pair of lights, defined by one light from each set, together with s_c form a closest embracing triangle for q. Two examples of a closest embracing triangle for q constructed using this method can be seen in Figure 2.13.

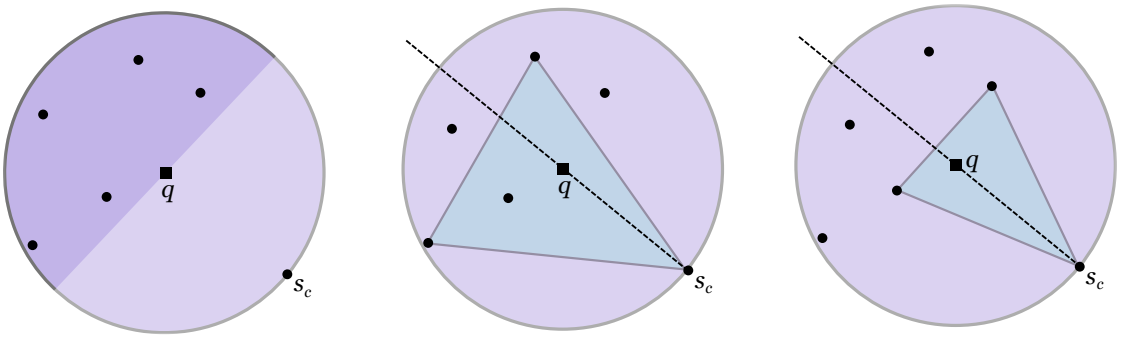


Figure 2.13: Disc $D(q, d(s_c, q))$ is shown in purple and $\text{MER}(q) = d(s_c, q)$. Any pair of lights, defined by one light on each side of qs_c, together with s_c form a $\text{CET}(q)$ (shown in blue).

The complexity of the algorithm proposed above is stated in the following result.

Theorem 2.2 *Given a set S of n lights and point q on the plane, the minimum embracing range of q and a closest embracing triangle for q can be found in $\mathcal{O}(n)$ time and space.*

Proof: Given a set S of n lights and a point q on the plane, set $R = \{d(s_i, q) : s_i \in S\}$ can be calculated in linear time, as well as its median [25]. In each step of the binary search there is the need to verify if a given set $S' \subseteq S$ is an embracing set for q. While the search is being carried out on the lowest half of R, every light of S' is processed. However, once there is the need to search the highest half, the lights on the lowest will not be studied again since the outermost lights of the previously failed verification are saved to the next step. Therefore, the binary search runs in $\mathcal{O}\left(n + \sum_{i=1}^{log_2(n)} \frac{n}{2^i}\right) = \mathcal{O}(n)$ time, which means $\text{MER}(q)$ is calculated in linear time. A closest embracing triangle for q can also be found in linear time once the closest embracing site for q is known, as previously explained. Set S is the only data that needs to be stored, consequently this algorithm takes $\mathcal{O}(n)$ space. □

In the worst case, the algorithm introduced above has to analyse every light of S to calculate $\text{MER}(q)$. Consequently, $\Omega(n)$ is a lower bound for the algorithm which, in addition to Theorem 2.2, implies that this algorithm's running time is optimal. To conclude, note that deciding if a given point is 1-well illuminated by a set of lights with range r is trivial after its minimum embracing range has been found. Such point is 1-well illuminated if $r \geq \text{MER}(q)$.

2.3 Minimum Embracing Range to 1-Well Illuminate a Line Segment

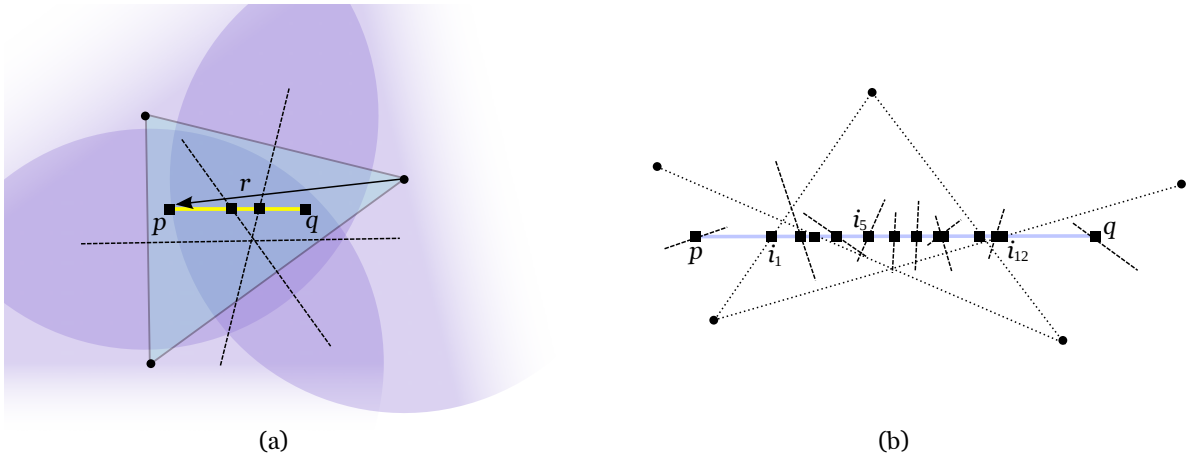
Given two points p and q on the plane, let $\overline{pq}$ be the line segment connecting both. This section proposes two algorithms to calculate the minimum embracing range to 1-well illuminate $\overline{pq}$. Such range is denoted by MER($\overline{pq}$). The first algorithm calculates MER($\overline{pq}$) in $\mathcal{O}(n^3)$ time using a basic approach to the solution. The second algorithm takes advantage of a more efficient technique called the Parametric Search [61, 62], which is applied to a parallelised decision algorithm. In this way, the algorithm is quicker and achieves a solution in quadratic time, dropping the extra n factor from the previous temporal complexity. The applicability of the Parametric Search to this problem and both algorithms will be analysed in detail in the discussion that follows.

2.3.1 A Plain Algorithm

The definition of 1-good illumination has to be slightly altered in order to achieve a correct solution using the algorithm proposed below. Instead of requiring that a point is inside a convex hull of lights, it suffices to consider that the point is 1-well illuminated if it is on such convex hull. In this way, a point on the boundary of a convex hull of lights is also 1-well illuminated. Without loss of generality, assume that $\overline{pq}$ is a horizontal line segment and that p is its leftmost point whilst q is the rightmost. Assume also that the initial set S of lights 1-well illuminates $\overline{pq}$. This can be easily decided in $\mathcal{O}(n)$ time since it suffices that points p and q are on CH(S).

If the problem were to find a set of three lights that 1-well illuminate $\overline{pq}$, then the solution would be quite simple to achieve. All the ternary sets of lights would be tested to find the ones that contain $\overline{pq}$, which would take $\mathcal{O}(n^3)$ time. Then the perpendicular bisectors between the lights of each of these sets should be intersected with $\overline{pq}$, which would generate at most three intersections (see Figure 2.14(a)). These intersections together with p and q would then be analysed to finally calculate MER($\overline{pq}$).

In contrast to the problem discussed in the last paragraph, the main problem in this section allows that $\overline{pq}$ is 1-well illuminated by a group of different sets of lights, as long as the line segment is fully 1-well illuminated. To calculate MER($\overline{pq}$), the algorithm starts by dividing $\overline{pq}$ into several pieces in a way that each piece is 1-well illuminated by the same set of lights. This partition is the plain part of the algorithm since it is not efficient, every point where the closest embracing set might change is analysed and most of the times this process is fruitless.



(a) (b)

Figure 2.14: (a) Dark blue region is 1-well illuminated with range $\text{MER}(\overline{pq}) = r$. (b) Line segment $\overline{pq}$ is broken into thirteen segments. Dotted lines represent the line segments of T and the dashed the perpendicular bisectors of B.

The points where a closest embracing set or closest embracing site might change are found using sets T and B as described in the following. First, divide the lights in sets S_t and S_b, the first is the set of lights above $\overline{pq}$ and the second the set of lights below $\overline{pq}$. Second, calculate the intersection points between $\overline{pq}$ and $T = \{\overline{s_i s_j} : s_i \in S_t, s_j \in S_b\}$. Set T simulates the edges of the convex hulls of possible closest embracing sets. Points where the closest embracing site may change are the intersection points between $\overline{pq}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$. Finally, there is also the need to analyse the outer points of $\overline{pq}$, p and q. Let I be the union of all these intersection points sorted from left to right, that is, according to their x-coordinate (see Figure 2.14(b)). Point p can be seen as the first point of set I, i_0, and q as the last, i_m. Each segment between two consecutive points of I, $\overline{i_{k-1} i_k} \subseteq \overline{pq}$, $k = 1, \ldots, m$, will be studied one at a time.

As soon as the previous tasks are finished, the next step of the algorithm is to analyse $\overline{i_0 i_1}$, which is the first segment of $\overline{pq}$, and calculate $r_0 = \text{MER}(i_0)$ and a $\text{CET}(i_0)$. This can be achieved by applying the "Minimum Embracing Range II" algorithm (described in Section 2.2.3) observing a slight modification: straight angles are allowed. It is not difficult to see that $\text{MER}(\overline{pq}) \geq \text{MER}(i_0)$. Once point i_0 is 1-well illuminated and a closest embracing set for it has been found, the algorithm proceeds to the next point of I, which is i_1. Such point can result from the intersection between $\overline{pq}$ and T or $\overline{pq}$ and B. As the first case, suppose $i_1 \in T$ and that it is a point on the boundary of $\text{CET}(i_0)$. Therefore, point i_1 is not 1-well illuminated by $\text{CET}(i_0)$, but before computing $\text{CET}(i_1)$ there is the need to secure that segment $\overline{i_0 i_1}$ is 1-well illuminated by $\text{CET}(i_0)$. For example, in Figure 2.15(a) the lights' range r_0 only 1-well illuminates i_0. Therefore, r_0 has to be updated to $d(s_i, i_1)$. The opposite situation is illustrated

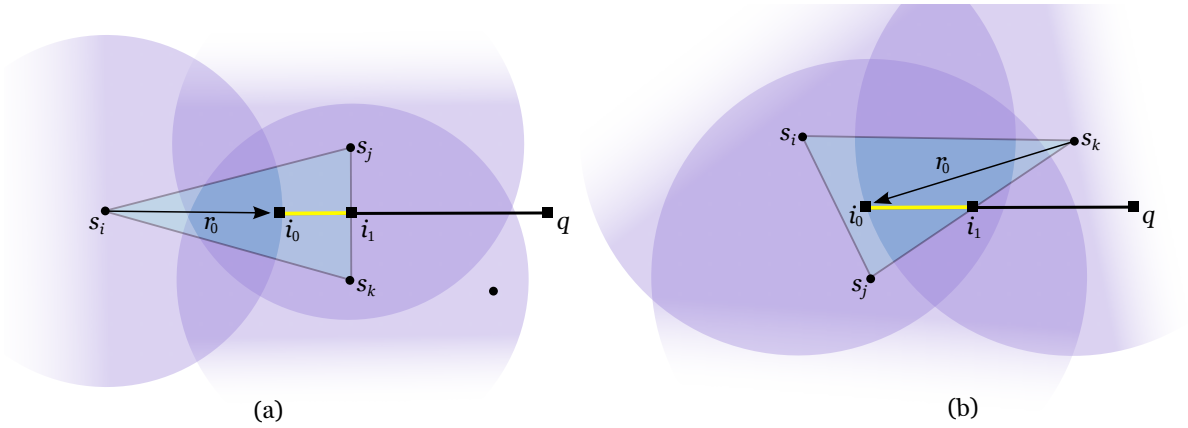


Figure 2.15: (a) Yellow segment $\overline{i_0 i_1}$ is not 1-well illuminated with $r_0 = d(s_i, i_0)$, but it is if $r_0 = d(s_i, i_1)$. (b) Yellow segment $\overline{i_0 i_1}$ is 1-well illuminated with minimum range $r_0 = d(s_k, i_0)$.

in Figure 2.15(b), where range $r_0 = d(s_k, i_0)$ is enough to 1-well illuminate segment $\overline{i_0 i_1}$. To conclude this case, and assuming that s_i is the farthest vertex of CET(i_0) to i_1, update the current range to $r_0 = \max\{r_0, d(s_i, i_1)\}$. Afterwards, a CET($i_1$) and the minimum embracing range $r_1 = \text{MER}(i_1)$ are calculated using the "Minimum Embracing Range II" algorithm. As the second case, suppose $i_1 \in T$ but it is not a point on the boundary of CET(i_0), as illustrated in Figure 2.16(a). If CET(i_0) is a closest embracing triangle for i_1, then such point is 1-well illuminated if the range r_0 is enough to reach it (meaning that r_0 needs to be revised). Otherwise, i_1 is not 1-well illuminated and there is the need to calculate another triangle and update the range just like it was done in the previous case (see Figure 2.16(b)).

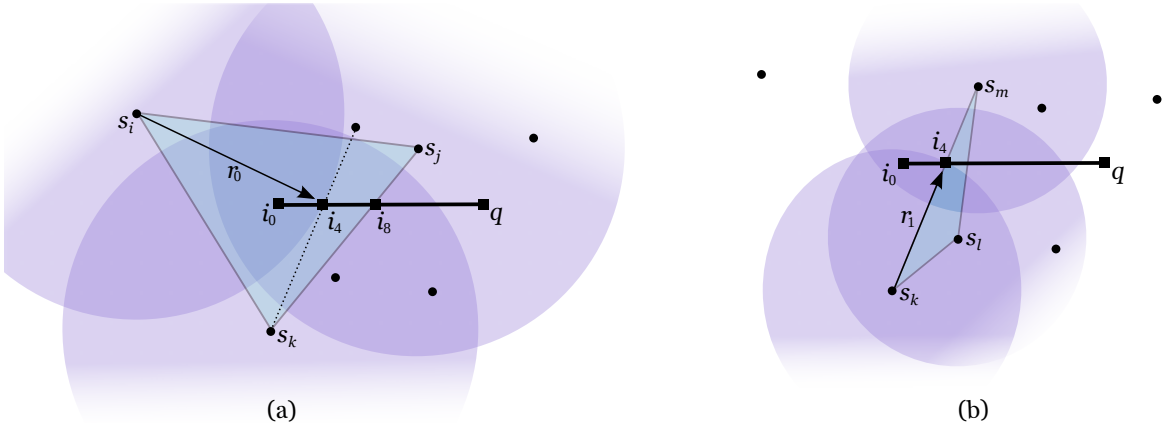


Figure 2.16: (a) Point i_4 is inside CET(i_0) but CET(i_0) $\neq$ CET(i_4). Range r_0 is updated to $r_0 = d(s_i, i_4)$. (b) Triangle $\triangle(s_k, s_l, s_m)$ is a closest embracing triangle for i_4 and MER(i_4) = $d(s_k, i_4) = r_1$.

As the last case, let i_1 be the intersection point between $\overline{pq}$ and PB(s_i, s_j). Although

$i_1 \in \text{CET}(i_0)$, it is possible that this is not a closest embracing triangle for the next segment on $\overline{pq}$, $\overline{i_1 i_2}$, if s_i or s_j is the closest embracing site for i_1 (see Figure 2.17). Without loss of generality, suppose s_i is the closest embracing site for i_1 and compute a $\text{CET}(i_1)$ using the lights of $S \backslash \{s_i\}$. If the new triangle 1-well illuminates i_1 with the same range as r_0 then $\text{CET}(i_1)$ prevails over $\text{CET}(i_0)$ since s_i no longer is the closest embracing site for segment $\overline{i_1 i_2}$. However, before swapping the triangles, r_0 needs to be revised. Afterwards, range r_1 is initiated with same value of r_0 since i_1 is a point on $\text{PB}(s_i, s_j)$.

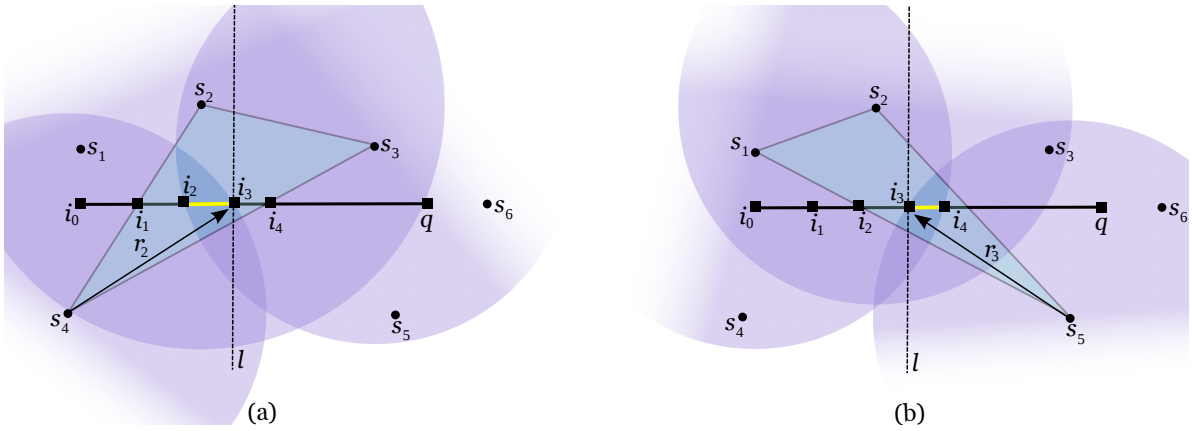


Figure 2.17: Line l is the $\text{PB}(s_4, s_5)$. (a) Segment $\overline{i_2 i_3}$ is 1-well illuminated by $\triangle(s_2, s_3, s_4)$ with range $r_2 = d(s_4, i_3)$. (b) Even though $i_3 \in \triangle(s_2, s_3, s_4)$, $\triangle(s_1, s_2, s_5)$ is the closest embracing triangle for i_3 and $\text{MER}(i_3) = r_3$.

The algorithm repeats this procedure for every intersection point of I until point $i_m = q$ is reached. When that happens, the algorithm outputs $\text{MER}(\overline{pq})$, which is the minimum embracing range that 1-well illuminates every segment of $\overline{pq}$ and consequently $\overline{pq}$. In addition, it is also known which is the minimum embracing range and closest embracing set that 1-well illuminates each segment. Therefore, it is possible to 1-well illuminate some segments of $\overline{pq}$ instead of the whole line segment. For example, when an artist is walking on stage, it is reasonable to only turn on the floodlights that 1-well illuminate the path that person is walking. Moreover, it is also possible to minimise the lights' range for that path while keeping it 1-well illuminated.

Theorem 2.3 *Given a set S of n lights and two points p and q on the plane, $\text{MER}(\overline{pq})$ and a set of closest embracing triangles that 1-well illuminate different segments of $\overline{pq}$ can be computed in $\mathcal{O}(n^3)$ time.*

Proof: Deciding if $\overline{pq}$ is on $\text{CH}(S)$ can be done in $\mathcal{O}(n)$ time since it suffices to verify if p and q are on $\text{CH}(S)$. Dividing the lights of S in sets S_t and S_b, S_t of the lights above $\overline{pq}$ and

S_b of the lights below, takes linear time too. Sorting the intersection points according to their x-coordinate takes $\mathcal{O}(n^2 \log n)$ time since sets T and B are defined for every pair of lights ($T = \{\overline{s_i s_j} : s_i \in S_t, s_j \in S_b\}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$). Therefore, intersecting T and B with $\overline{pq}$ results in a quadratic number of intersection points. For each intersection point, updating the range can be done in constant time but if the algorithm "Minimum Embracing Range II" (introduced in Section 2.2.3) is run, then the temporal complexity increases to $\mathcal{O}(n)$ according to Theorem 2.2. Therefore, in the worst case each segment defined by two consecutive intersection points is 1-well illuminated in linear time. Overall, the method described above computes the minimum embracing range and a closest embracing triangle for each segment of $\overline{pq}$ in $\mathcal{O}(n^3)$ time. The minimum embracing range to 1-well illuminate $\overline{pq}$ is the minimum range needed to 1-well illuminate every segment of $\overline{pq}$. In conclusion, if every segment of $\overline{pq}$ is 1-well illuminated in $\mathcal{O}(n^3)$ time, then $\overline{pq}$ also is 1-well illuminated in the same amount of time. □

The result below is a direct consequence of this theorem.

Corollary 2.1 *Given a set S of n lights and a polygonal line l with m segments,* MER(l) *and a set of closest embracing triangles that* 1*-well illuminate each line segment of l can be computed in $\mathcal{O}(mn^3)$ time.*

2.3.2 A Parallel Algorithm

Similarly to the previous section, assume that $\overline{pq}$ is a horizontal line segment connecting points p and q, where p is the leftmost endpoint and q the rightmost. The algorithm presented in this section calculates MER($\overline{pq}$) using the Parametric Search, which is a technique introduced by Megiddo [61, 62]. The main principle underpinning this technique is to calculate a value λ^*, which optimises a function g, by making use of an efficient algorithm to solve the decision problem. The search is particularly effective if g is a monotonic function ($g(x) \geq g(y)$ if $x > y$) and λ^* the largest root of g. If the decision algorithm runs in $A(n)$ time for an input of size n, then the parametric search finds λ^* in $\mathcal{O}(A(n)^2)$ time. Moreover, if the decision algorithm can be parallelised to run in $T(n)$ time using $P(n)$ processors, λ^* is found in $\mathcal{O}(A(n)T(n) \log P(n) + T(n)P(n))$ time. Therefore, it is best to develop a decision algorithm that can be parallelised in order to apply this technique as efficiently as possible.

Regarding the main subject of this section, the decision problem can be stated as: *does a given set S of n lights with range $r \in \mathbb{R}^+$ 1-well illuminate $\overline{pq}$?* An algorithm to solve this decision problem is proposed below. This algorithm will be later combined with the Parametric Search to solve the main optimisation problem. Further detail on the parallelisation of the

decision algorithm in order to improve the effectiveness of the main algorithm is also given in the following discussion.

Decision Algorithm

The algorithm introduced below decides whether a set S of lights with a given range $r \in \mathbb{R}^+$ 1-well illuminates $\overline{pq}$. The first step of the algorithm is to divide $\overline{pq}$ into several open segments, each of which will be analysed separately to decide if it is 1-well illuminated by S. In the following, it is shown that if two consecutive open segments are 1-well illuminated, then the point in between them, that is, the point that connects them also is 1-well illuminated. Therefore, if all segments and points in between are 1-well illuminated, so is $\overline{pq}$. The disc of radius r centred at $s_i \in S$, $D(s_i, r)$, bounds the points on the plane that are illuminated by s_i. Each of these discs intersects $\overline{pq}$ at most twice. Since S is formed by n lights, there are also n discs on the plane. This implies that the number of intersection points between such discs and $\overline{pq}$ is at most $2n$. Let $I = \{i_0, \ldots, i_m\}$ be the set of these intersection points sorted according to their x-coordinate (that is, from left to right). Assume that $p = i_0$ and $q = i_m$. Let $f : I \mapsto S$ be a function that associates an intersection point $i_k \in I$ to a light $s_i \in S$, $f(i_k) = s_i$, if i_k is the intersection point between $\overline{pq}$ and disc $D(s_i, r)$. If point i_k is on the left (right) of the disc's centre s_i, then i_k is called the leftmost (rightmost) intersection point between $\overline{pq}$ and $D(s_i, r)$. Let l_k denote the open segment with endpoints i_{k-1} and i_k, $l_k = (i_{k-1}, i_k)$, for $k = 1, \ldots, m$. Note that every point on l_k is illuminated by the same set of lights, which is consequential to the method used to divide $\overline{pq}$. Let $\mathcal{F}(l_k)$ be a function that returns the set of lights that illuminates segment l_k with range r. Assuming $\mathcal{F}(l_1) = \mathcal{F}(i_0)$, this function is defined recursively for $k = 1, \ldots, m-1$:

$$\mathcal{F}(l_{k+1}) = \begin{cases} \mathcal{F}(l_k) \cup \{f(i_k)\}, & \text{if } i_k \text{ is a leftmost intersection} \\ \mathcal{F}(l_k) \setminus \{f(i_k)\}, & \text{if } i_k \text{ is a rightmost intersection} \end{cases}$$

The relation between $\mathcal{F}$ and the notions of leftmost and rightmost intersection points are illustrated in Figure 2.18.

Lemma 2.1 *If two consecutive open segments $l_k = (i_{k-1}, i_k)$ and $l_{k+1} = (i_k, i_{k+1})$ of $\overline{pq}$ are 1-well illuminated by S, then point $i_k \in I$ also is 1-well illuminated by S.*

Proof: Suppose point $i_k \in I$ is not 1-well illuminated, that is, $i_k \notin \text{int}(\text{CH}(\mathcal{F}(l_k)) \cup \text{CH}(\mathcal{F}(l_{k+1})))$. Since both open segments are 1-well illuminated, i_k must lie on the boundary of both convex hulls. However, according to the definition of function $\mathcal{F}$, one of the convex

hulls is contained in the other and both segments are inside the largest because they are 1-well illuminated. Therefore, i_k must be inside of one of the convex hulls, which means it is 1-well illuminated as well. □

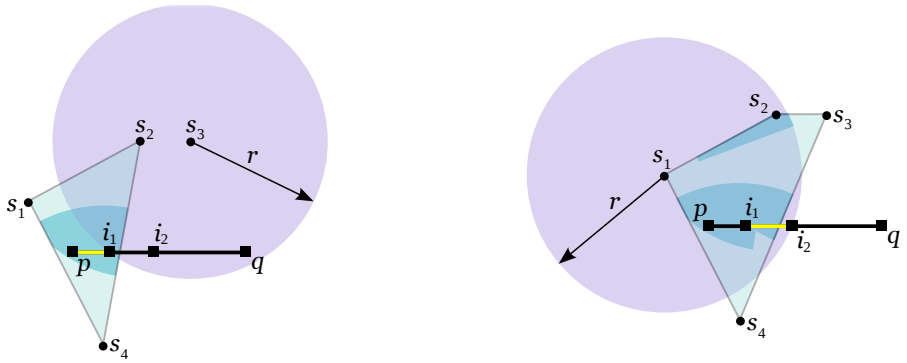


Figure 2.18: Blue regions are 1-well illuminated with range r. (a) Point $i_1 \in I$ is the leftmost intersection point between $\overline{pq}$ and $D(s_3, r)$. Segment $l_1 = (p, i_1)$ is 1-well illuminated by $\mathcal{F}(l_1) = \{s_1, s_2, s_4\}$. (b) Point $i_2 \in I$ is the rightmost intersection point between $\overline{pq}$ and $D(s_1, r)$, $\mathcal{F}(l_2) = \mathcal{F}(l_1) \cup \{s_3\}$.

Lemma 2.2 *If intersection points i_{k-1} and i_k are both inside* $\mathrm{CH}(\mathcal{F}(l_k))$*, then the open segment l_k is* 1*-well illuminated.*

Proof: Attending to the definition of function $\mathcal{F}$, segment l_k is 1-well illuminated if it is inside the convex hull of $\mathcal{F}(l_k)$. This is indeed the case as i_k and i_{k-1} are inside $\mathrm{CH}(\mathcal{F}(l_k))$ by hypothesis, which implies that the segment connecting both points also is. □

The following theorem is a direct consequence of the two previous lemmas.

Theorem 2.4 *Given two points p and q on the plane and $\overline{pq}$ partitioned in segments using the method described above, if p, q and all segments $l_k \subseteq \overline{pq}$ are* 1*-well illuminated, then $\overline{pq}$ also is* 1*-well illuminated.*

What follows is therefore an efficient algorithm to solve this decision problem. First, decide if p and q are 1-well illuminated using the "Minimum Embracing Range II" algorithm (introduced in Section 2.2.3). Second, for each segment l_k construct $\mathrm{CH}(\mathcal{F}(l_k))$, $k = 1, \ldots, m$, by simply inserting or removing one light of $\mathrm{CH}(\mathcal{F}(l_{k-1}))$. Then decide whether l_k is 1-well illuminated by $\mathcal{F}(l_k)$. Third, if all these segments are 1-well illuminated, then $\overline{pq}$ is 1-well illuminated too. Otherwise, the lights' range is not sufficient to fully 1-well illuminate the

line segment. In Figure 2.19 there is a detailed example of this algorithm, $\overline{pq}$ is not 1-well illuminated since $l_7 \notin \text{int}(\text{CH}(\mathcal{F}(l_7)))$. The theorem below states the temporal complexity of this decision algorithm.

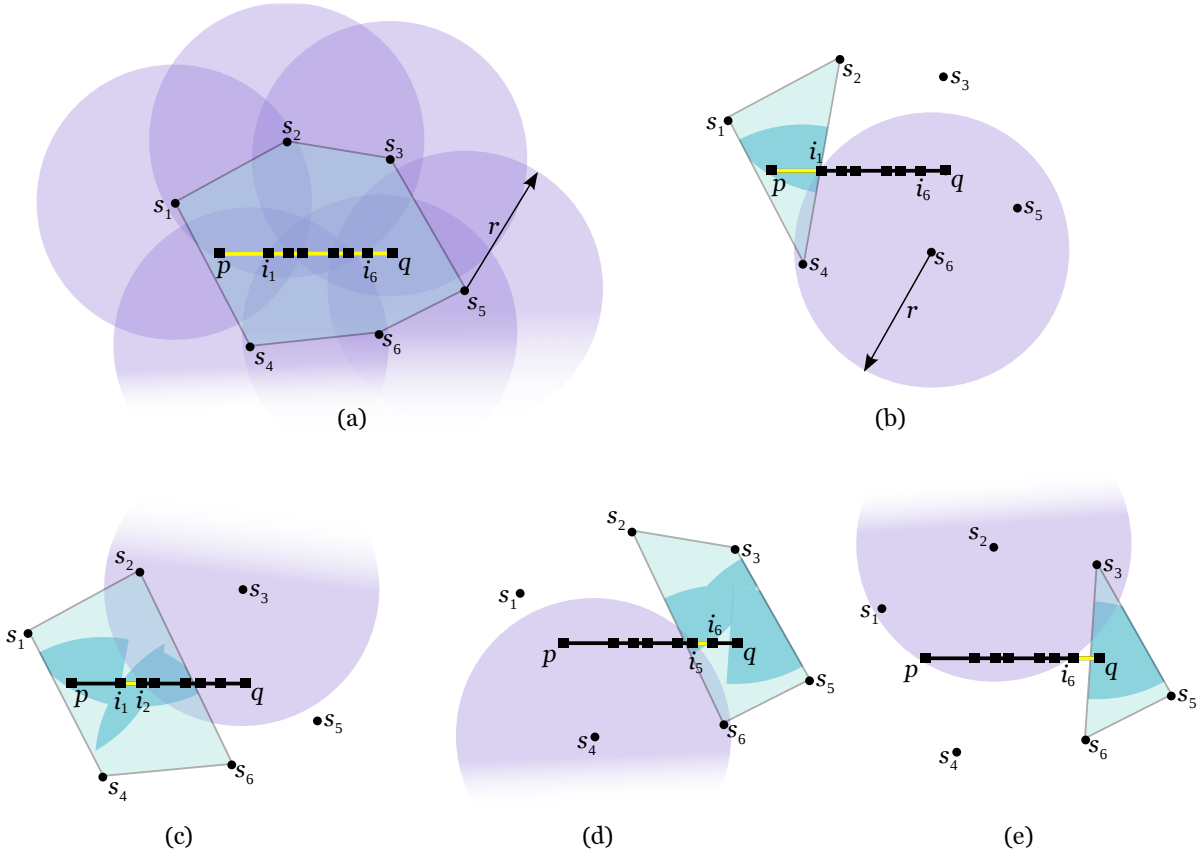


Figure 2.19: Blue regions are 1-well illuminated with range r. (a) There are six intersection points between $\overline{pq}$ and the discs of radius r centred at the lights. (b) Point i_1 is the leftmost intersection point between $\overline{pq}$ and $D(s_6, r)$. Segment $l_1 = (p, i_1)$ is 1-well illuminated by $\mathcal{F}(l_1) = \{s_1, s_2, s_4\}$. (c) Segment $l_2 = (i_1, i_2)$ is 1-well illuminated by $\mathcal{F}(l_2) = \{s_1, s_2, s_4, s_6\}$. (d) Segment $l_6 = (i_5, i_6)$ is 1-well illuminated by $\mathcal{F}(l_6) = \{s_3, s_4, s_5, s_6\}$. (e) Segment $l_7 = (i_6, q)$ is not 1-well illuminated since $l_7 \notin \text{int}(\text{CH}(\mathcal{F}(l_7)))$.

Theorem 2.5 *Given a set S of n lights with range $r \in \mathbb{R}^+$ and two points p and q on the plane, deciding if $\overline{pq}$ is 1-well illuminated by S takes $\mathcal{O}(n \log n)$ time.*

Proof: According to Theorem 2.2, deciding if p and q are 1-well illuminated using the "Minimum Embracing Range II" algorithm takes linear time. Let I be the set of the intersection points between $\overline{pq}$ and the discs of radius r centred at the lights of S. Since the cardinality of I is at most $2n$, these points can be sorted in $\mathcal{O}(n \log n)$ time. The convex hull of the lights that 1-well illuminate p is also constructed in $\mathcal{O}(n \log n)$ time. Since the convex hull can be

dynamically updated, each light can be inserted or removed from the structure in $\mathcal{O}(\log n)$ amortised time [27]. Similarly, deciding if points i_{k-1} and i_k of I are inside $\mathrm{CH}(\mathcal{F}(l_k))$ takes $\mathcal{O}(\log n)$ time. To conclude, and since each point of I is analysed in $\mathcal{O}(\log n)$ time, deciding if $\overline{pq}$ is 1-well illuminated by S takes $\mathcal{O}(n \log n)$ time. □

Observe that this algorithm also works when each light has a different illumination range. In this case, the discs centred at the lights have different radii. After computing the intersection points between the discs and $\overline{pq}$, the remaining procedure is exactly the same.

Final Algorithm

The algorithm above is an efficient decision algorithm that will be now combined with the Parametric Search [61, 62]. This combination results in an optimisation algorithm that calculates the minimum illumination range to 1-well illuminate $\overline{pq}$. However, in order to apply the Parametric Search technique, first there is the need to convert the decision algorithm in a monotonic root-finding problem. Let function g be defined as:

$$g(\lambda) = \begin{cases} 0, & \overline{pq} \text{ is 1-well illuminated by } S \text{ with range } r = \frac{1}{\lambda} \\ 1, & \overline{pq} \text{ is not 1-well illuminated by } S \text{ with range } r = \frac{1}{\lambda} \end{cases}$$

As previously explained, the objective of the Parametric Search is to find the largest root λ^* of function g, $\lambda^* = \max\{\lambda \in \mathbb{R}^+ : g(\lambda) = 0\}$. Once such root is found, $\mathrm{MER}(\overline{pq}) = \frac{1}{\lambda^*}$. According to Theorem 2.5, the decision algorithm runs in $\mathcal{O}(n \log n)$ time so λ^* can be found in $\mathcal{O}(n^2 \log^2 n)$ time. A small improvement on the performance of this algorithm can be attained if the decision algorithm is parallelised. To this end, the lights have to be lexicographic sorted using $\mathcal{O}(n)$ processors, which takes $\mathcal{O}(\log n)$ time. If each light is allocated to a different processor, then all the intersection points between the discs and $\overline{pq}$ can be found in constant time. As a result, each processor has to analyse at most two intersection points to decide if they are inside the convex hull of the lights illuminating them, that is, to decide if the intersection points are 1-well illuminated. Since the lights are lexicographic sorted, the convex hull is constructed in $\mathcal{O}(\log n)$ time if helped by $\mathcal{O}(\frac{n}{\log n})$ additional processors. Therefore, performing these decisions takes $\mathcal{O}(\log n)$ time. Finally, the parallelised decision algorithm runs in $\mathcal{O}(\log n)$ time if aided by $\mathcal{O}(\frac{n^2}{\log n})$ processors.

Theorem 2.6 *Given a set S of n lights and two points p and q on the plane, the Parametric Search calculates the minimum embracing range to 1-well illuminate $\overline{pq}$ in $\mathcal{O}(n^2)$ time.*

Proof: The sequential decision algorithm takes $A(n) \in \mathcal{O}(n \log n)$ time while the parallel one

requires $T(n) \in \mathcal{O}(\log n)$ time when using $P(n) \in \mathcal{O}(\frac{n^2}{\log n})$ processors. So the total time to evaluate function $g(\lambda)$ and calculate its largest root, as well as MER($\overline{pq}$), using the Parametric Search is $\mathcal{O}(A(n)T(n)\log P(n) + T(n)P(n)) \in \mathcal{O}(n\log n \times \log n \times \log(\frac{n^2}{\log n}) + \log n \times \frac{n^2}{\log n}) = \mathcal{O}(n^2)$ time. □

Corollary 2.2 *Given a set S of n lights and a polygonal line l with m segments, the minimum embracing range to 1-well illuminate l can be calculated in $\mathcal{O}(mn^2)$ time.*

2.4 Orthogonal Good Illumination

This section and the one that follows are dedicated to two variants of 1-good illumination. The variant introduced below is called the *orthogonal good illumination*.

Definition 2.3 *A point q on the plane is orthogonally well illuminated by a set S of lights if there is at least one light of S illuminating q on each quadrant with origin at q.*

This section's objective is to calculate the minimum illumination range of S to orthogonally well illuminate a point q on the plane. The key structure in this type of illumination is the *orthogonal convex hull*. The usual convex hull of a set of points is widely recognised and can be defined as the smallest convex region that contains such set. In the case of the orthogonal convex hull, the orthogonal prefix means that the convexity is defined by axis-parallel point connections. Consequently, two points can only be connected through vertical and horizontal line segments. If two points are inside the orthogonal convex hull of S, then the polygonal line connecting them also lies on the orthogonal convex hull of S (see Figure 2.20(a)). Karlsson and Overmars [50] constructed this structure in $\mathcal{O}(n\log n)$ time. The basic idea is to divide the lights into four quadrants and then sort them on each quadrant from left to right or top to bottom. The orthogonal convex hull is a useful geometric tool that will be further discussed in the following chapter.

In this variation of good illumination, the orthogonal convex hull plays the role of the usual convex hull in 1-good illumination. As previously mentioned, a point is 1-well illuminated by S if it is inside the convex hull of S. In comparison, a point inside the orthogonal convex hull of S is orthogonally well illuminated (see Figure 2.20(b)). It is not difficult to decide if a point q is orthogonally well illuminated by S. First, divide the lights of S into four quadrants with origin at q. Second, if any quadrant is empty then q is not orthogonally well illuminated by S. Assuming this is not the case, then there must be at least one light on each quadrant. Therefore, a closest embracing set for q is formed by the closest light to q on each quadrant.

The minimum illumination range to orthogonally well illuminate q is given by the largest distance between q and these four lights (see Figures 2.20(b) and 2.20(c)).

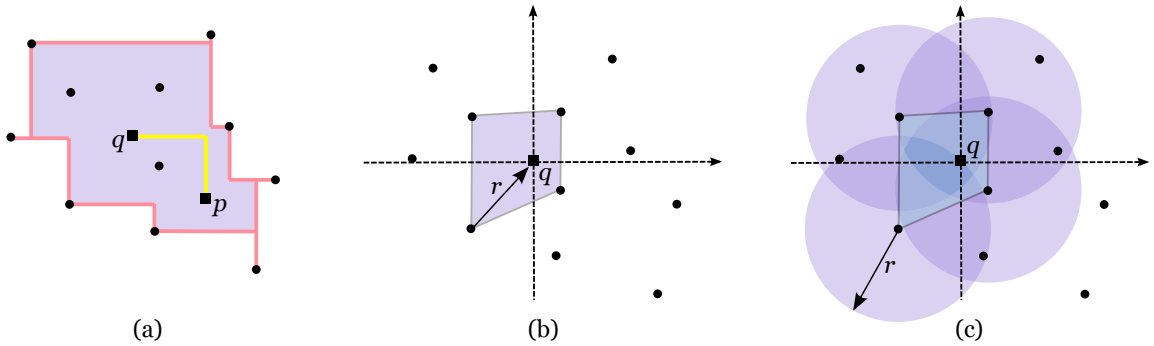


Figure 2.20: (a) Every polygonal line connecting points p and q using only vertical and horizontal line segments is contained in the orthogonal convex hull. (b) Point q is orthogonally well illuminated with minimum illumination range r. (c) Blue region is orthogonally well illuminated with range r.

Theorem 2.7 *Given a set S of n lights and a point q on the plane, a closest embracing set for q and the minimum illumination range of S that orthogonally well illuminates q can be calculated in $\mathcal{O}(n)$ time and space.*

Proof: Dividing the lights through four quadrants with origin at point q takes linear time since S is formed by n lights. Deciding if any of these quadrants is empty is easily verified at the same time that the closest light to q is being searched on each quadrant. This search is linear on the number of lights. The set of the four closest lights to q, one on each quadrant, forms a closest embracing set for q. The largest of the distances between q and the lights of this embracing set is the minimum illumination range of S to orthogonally well illuminate q. This range can be calculated in constant time. Therefore, finding a closest embracing set for q and the minimum illumination range of S that orthogonally well illuminates q takes $\mathcal{O}(n)$ time. Regarding space complexity, there is the need to store n lights and their placement on the quadrants, which takes $\mathcal{O}(n)$ space. □

2.5 Good α-Illumination

This section introduces another variation of 1-good illumination: the *good α-illumination*. The variable $\alpha \leq \pi$ represents an angle and is a given fixed parameter. This variant is particularly compelling since it can be seen as a generalisation of 1-good illumination.

Definition 2.4 *Let S be a set of lights on the plane. A point q is well α-illuminated by S if there is at least one light illuminating q inside every wedge of angle $\alpha \leq \pi$ with apex at q.*

This concept is illustrated in Figures 2.21(a) and 2.21(b). Well α-illuminated points are associated with dominance and maximal points as explained in the following. Given two points p and q on the plane, point $p = (p_x, p_y)$ dominates $q = (q_x, q_y)$ if $p_x > q_x$ and $p_y > q_y$. Therefore, a point is said to be maximal (or a maximum) if it is not dominated, that is, if the northeast quadrant with origin at p is empty (see Figure 2.21(c)). Avis et al. [22] extended this definition: a point p on the plane is said to be an unoriented α-maximum if there is an empty wedge of angle of at least α with apex at p. This definition is convenient for data visualisation, in particular, for boundary descriptions. The problem of finding all the maximal points of a set S is known as the *maxima problem* [54] and the problem of finding all the unoriented α-maximal points is known as the *unoriented α-maxima problem* [22]. The following proposition is a direct consequence of the definitions of good α-illumination and unoriented α-maxima.

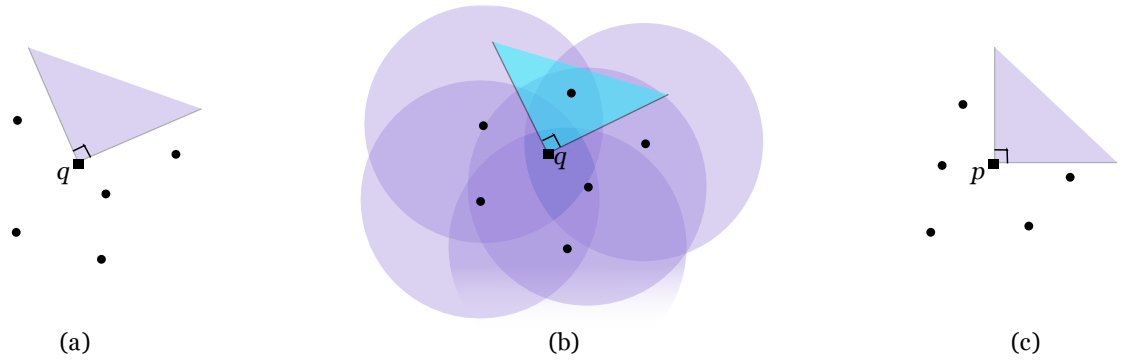


Figure 2.21: (a) Point q is not well $\frac{\pi}{2}$-illuminated because the purple wedge of angle $\frac{\pi}{2}$ with apex at q is empty. (b) Point q is $\frac{\pi}{2}$-illuminated, as well as the whole blue region. (c) Point p is maximal.

Proposition 2.2 *Let S be a set of lights and q a point on the plane. Point q is well α-illuminated by S if and only if q is not an unoriented α-maximum of the set $S \cup \{q\}$.*

Section 2.5.1 proposes an algorithm to calculate the minimum embracing range to α-illuminate a point on the plane and find a closest embracing set for that point. The implementation of such algorithm is illustrated in Section 2.5.2. Furthermore, it is shown how good α-illumination is intrinsically related to 1-good illumination. Section 2.5.3 introduces a data visualisation tool called the α-embracing contour. Contours have applications in quality illumination, and in particular, they can be applied to good α-illumination.

2.5.1 Minimum Embracing Range to Well α-Illuminate a Point

The algorithm that follows not only decides if a point q is well α-illuminated as it also calculates the minimum embracing range that makes it possible and a closest embracing set for q. Using a similar approach to the one used by the algorithm presented in Section 2.2.3, this optimisation algorithm performs a binary search on the distances between q and the lights. In each step of the search it decides whether q is well α-illuminated. This decision is the most important step of the algorithm and it is further discussed after the following pseudo-code. Let $\text{MER}^{\alpha}(q)$ denote the minimum embracing range to well α-illuminate point q.

ALGORITHM **Minimum Embracing Range III**

INPUT: Set S of n lights, angle $\alpha \leq \pi$, point q
OUTPUT: $\text{MER}^{\alpha}(q)$

1. Divide the lights of S into consecutive wedges of angle $\alpha/2$ with origin at q;
2. $R \leftarrow \{d(s_i, q) : s_i \in S\}$;
3. Perform a binary search on $R = \{r_0, \ldots, r_{n-1}\}$:

   For the median range $r_i \in R$ do

   $S' \leftarrow \{s_i \in S : d(s_i, q) \leq r_i\}$

   If q is well α-illuminated by S'

   Then proceed the search on $R \leftarrow \{r_j \in R : r_j \leq r_i\}$

   Otherwise proceed the search on $R \leftarrow \{r_j \in R : r_j > r_i\}$
4. The final range is $\text{MER}^{\alpha}(q)$.

Two steps of this algorithm are worth further discussion. The first is the division of lights into consecutive wedges of angle $\frac{\alpha}{2}$ with origin at q (see Figure 2.22(a)). Let w be the total number of such wedges. If 2π is divisible by α then $w = \frac{4\pi}{\alpha}$, otherwise $w = \lceil \frac{4\pi}{\alpha} \rceil$ since the last wedge has an angle smaller than $\frac{\alpha}{2}$ (see Figure 2.22(b)). As the angle α is fixed, the number of wedges is constant. Let i be an integer index of arithmetic mod w. For $i = 0, \ldots, w$, each ray i is defined by the set $\{q + (\cos(\frac{i\alpha}{2}), \sin(\frac{i\alpha}{2}))\lambda : \lambda > 0\}$, whilst each wedge is defined by q and two consecutive rays.

The second step worth of a detailed explanation, which was already mentioned, is the decision step on the binary search. If there are two adjacent empty wedges of angle $\frac{\alpha}{2}$, then

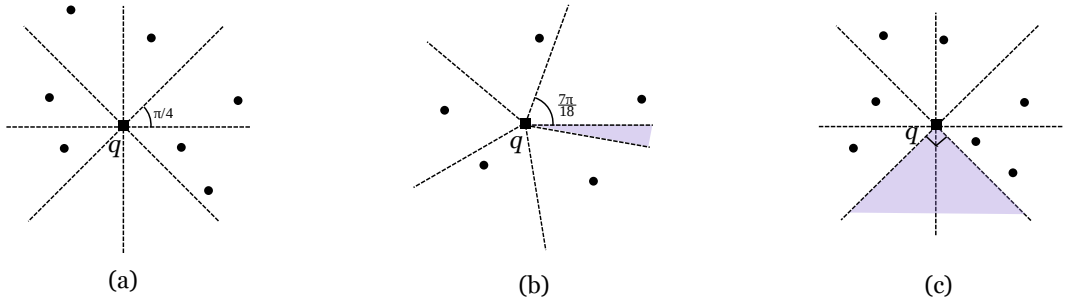


Figure 2.22: (a) To decide if q is well $\frac{\pi}{2}$-illuminated, the lights are divided into eight wedges of angle $\frac{\pi}{4}$. (b) If $\alpha = \frac{7}{9}\pi$, then the lights are divided into five wedges of angle $\frac{7}{18}\pi$ and one of $\frac{\pi}{18}$. (c) Point q is not well $\frac{\pi}{2}$-illuminated because the purple wedge of angle $\frac{\pi}{2}$ is empty.

q is not well α-illuminated (see Figure 2.22(c)). On the other hand, if there is at least one light on every wedge of angle $\frac{\alpha}{2}$ then q is well α-illuminated (see Figure 2.23(a)). As last case, suppose there is at least one empty wedge of angle $\frac{\alpha}{2}$ but no two adjacent empty ones (see Figure 2.23(b)). In this situation, there is the need to compute the outermost lights of each empty wedge. Let s_l be the outermost light found on the left of the wedge and s_l the one found on the right (see Figure 2.23(c)). If there is at least one empty angle $\measuredangle(s_l, q, s_r) \geq \alpha$, then q is not well α-illuminated by S. Otherwise, if every empty angle is less than α then q is well α-illuminated by S. In any other case, q is not well α-illuminated by S.

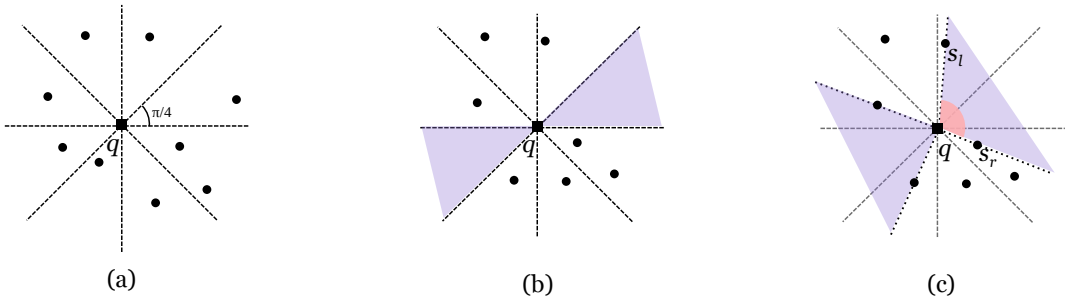


Figure 2.23: (a) Point q is well $\frac{\pi}{2}$-illuminated since there is a light on every wedge of angle $\frac{\pi}{4}$. (b) There are two non-adjacent empty wedges of angle $\frac{\pi}{2}$. (c) Point q is not well $\frac{\pi}{2}$-illuminated since $\measuredangle(s_l, q, s_r) \geq \frac{\pi}{2}$.

Once the algorithm is finished, it outputs $\text{MER}^{\alpha}(q)$ and a closest embracing site for q. A closest embracing set for q is formed by all the lights that are closer to q than its closest embracing site and the closest embracing site itself. Observe that this set only exists if S well α-illuminates q. The time and space complexities of the algorithm proposed above is stated in the theorem that follows.

Theorem 2.8 *Given a set S of n lights, a point q on the plane and an angle $\alpha \leq \pi$, deciding if q is well α-illuminated, calculating $\mathrm{MER}^{\alpha}(q)$ and finding a closest embracing set for q takes $\mathcal{O}(n)$ time and space.*

Proof: Given a set S of n lights, dividing them into consecutive wedges of angle $\frac{\alpha}{2}$ with origin at point q takes $\mathcal{O}(n)$ time. Set $R = \{d(s_i, q) : s_i \in S\}$ can be found in linear time, as also its median [25]. In each step of the binary search there is the need to decide if a given set $S' \subseteq S$ is an embracing set for q. While the binary search is performed on the lowest half of R, every light of S' is studied. However, once the search switches to the highest half, the lights on the lowest will not be studied again since the outermost lights of the previously failed verification are saved to the next step. Therefore, the binary search is performed in $\mathcal{O}\left(n + \sum_{i=1}^{log_2(n)} \frac{n}{2^i}\right) = \mathcal{O}(n)$ time. Consequently, $\mathrm{MER}^{\alpha}(q)$ is calculated in $\mathcal{O}(n)$ time. A closest embracing set for q can also be found in linear time once $\mathrm{MER}^{\alpha}(q)$ is known, as previously explained. Regarding the space complexity, the lights of S and set R need the largest storage and they both take $\mathcal{O}(n)$ space. □

In the previous algorithm, every light of S is a candidate to be a closest embracing site for point q. In the worst case, the closest embracing site for q is the furthest light of S to q and every light has to be processed. Therefore, the lower bound for this algorithm is $\Omega(n)$ time, which combined with Theorem 2.8 makes the linear complexity of this algorithm optimal. Furthermore, this algorithm not only finds a closest embracing set for q and $\mathrm{MER}^{\alpha}(q)$, as it also computes them for a t-well illuminated point (see Definition 2.1). The following proposition establishes a connection between t-good illumination and good α-illumination.

Proposition 2.3 *Let S be a set of lights, q a point on the plane and $\alpha \leq \pi$ a fixed angle. If point q is well α-illuminated by S, then q is t-well illuminated by S for $t = \lfloor \frac{\pi}{\alpha} \rfloor$.*

Proof: If q is well α-illuminated by S, then according to its definition there is at least one light on every wedge of angle α with origin at q. On the other hand, according to Definition 2.1, q is t-well illuminated if there are at least t lights of S illuminating q in every open half-plane with q on its border. Such a half-plane can be seen as a wedge of angle π with origin at q. Therefore, if there is at least one light on every wedge of angle α, then there are at least $\lfloor \frac{\pi}{\alpha} \rfloor$ lights in every half-plane. Consequently, q is $\lfloor \frac{\pi}{\alpha} \rfloor$-well illuminated and the minimum embracing range to well α-illuminate q also $\lfloor \frac{\pi}{\alpha} \rfloor$-well illuminates q. □

Corollary 2.3 *Let q be a well α-illuminated point on the plane and $\alpha \leq \pi$ a fixed angle. A closest embracing set for q also t-well illuminates q for $t = \lfloor \frac{\pi}{\alpha} \rfloor$.*

Proof: Assume that $t = \lfloor \frac{\pi}{\alpha} \rfloor$. According Proposition 2.3, the minimum embracing range to well α-illuminate q also t-well illuminates it. Therefore, a closest embracing site for a well α-illuminated point is at the same distance or farther than a closest embracing site for a t-well illuminated point. Consequently, a closest embracing set for q also t-well illuminates q. □

To conclude, note that the previous proposition is necessary but not sufficient. If point q is t-well illuminated then it is not necessarily true that q is well α-illuminated for $\alpha = \lfloor \frac{\pi}{t} \rfloor$. There is a counterexample in Figure 2.24: point q is 2-well illuminated but not well $\frac{\pi}{2}$-illuminated.

Figure 2.24: (a) Point q is 2-well illuminated. (b) Point q is not well $\frac{\pi}{2}$-illuminated.

2.5.2 Implementation

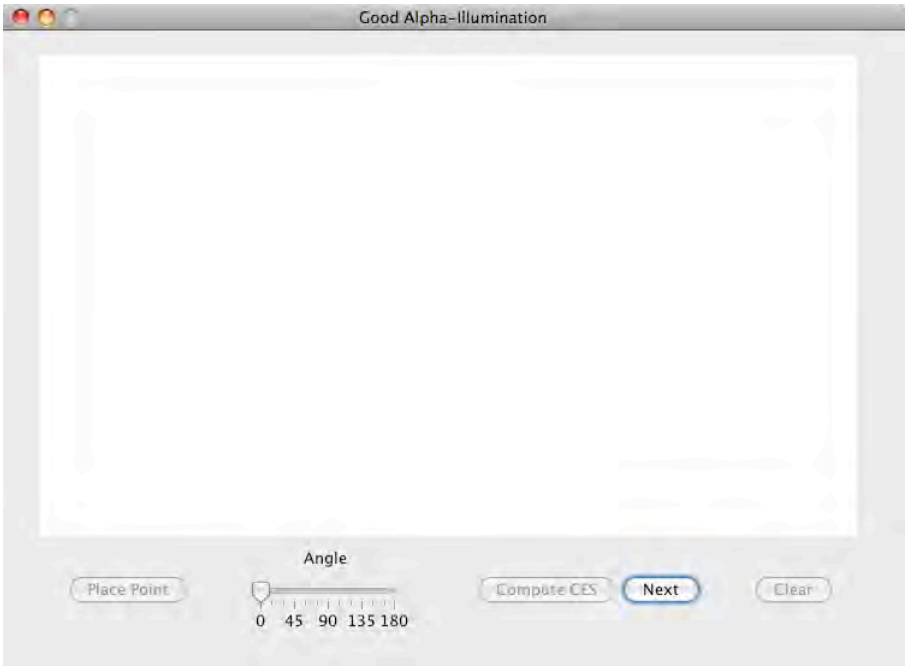


Figure 2.25: Aspect of the application once it is started.

The previous algorithm was implemented in Java and the resulting application was named "Good Alpha-Illumination". It starts with a white panel where the user can click to add lights (see Figure 2.25). Each light is represented by a yellow dot and button `Place Point` only is enabled after three lights have been placed. After pressing that button, the next click on the panel will show a red dot that represents the location of the query point (see Figure 2.26).

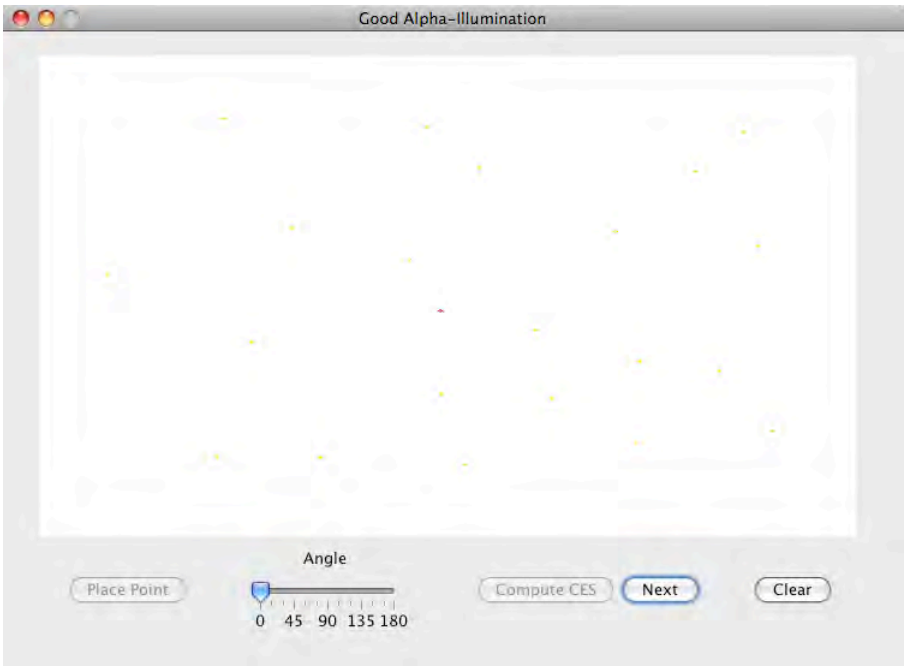


Figure 2.26: The yellow dots are lights and the red dot is the query point.

When the user is satisfied with the placement of the lights and query point, he can choose the angle α in degrees using the slider. For example, in Figure 2.27 there are wedges of angle $\frac{\pi}{8}$ because the angle chosen was $\frac{\pi}{4} = 45°$. Note that every wedge has the same angle. On the other hand, in Figure 2.28 there is an example where not all wedges have the same angle. The angle chosen was $\frac{3}{4}\pi = 135°$ and therefore, five wedges have angle $\frac{3}{8}\pi$ but the last wedge in anticlockwise order has an angle smaller than $\frac{3}{8}\pi$.

After the angle has been selected, button `Compute CES` initiates the main algorithm and the binary search begins. The median of the distances between the query point and the lights is calculated and the decision algorithm is applied to the half of the lights that are closer to the query point (see Figure 2.29). The green circle bounds the lights on the lowest half. In that same picture there are two blue lights, which means they were verified to decide if the angle they define with the query point is equal to or larger than α. If the query point is well α-illuminated, then the algorithm proceeds analysing the lowest half of the previous set of lights. Button `Next` should be pressed in order to run the algorithm step by step.

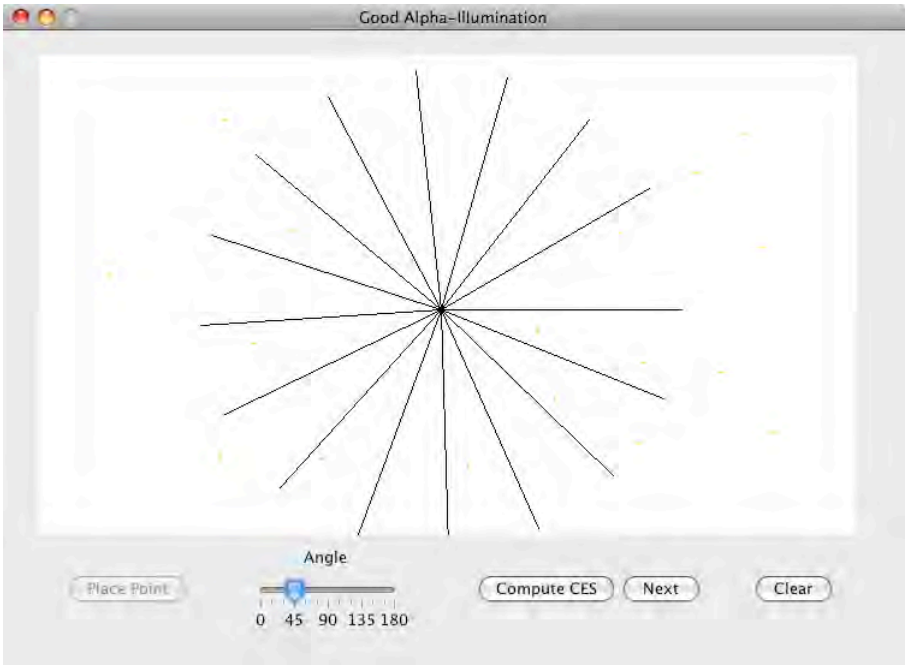


Figure 2.27: The plane is divided into wedges of angle $22,5°$.

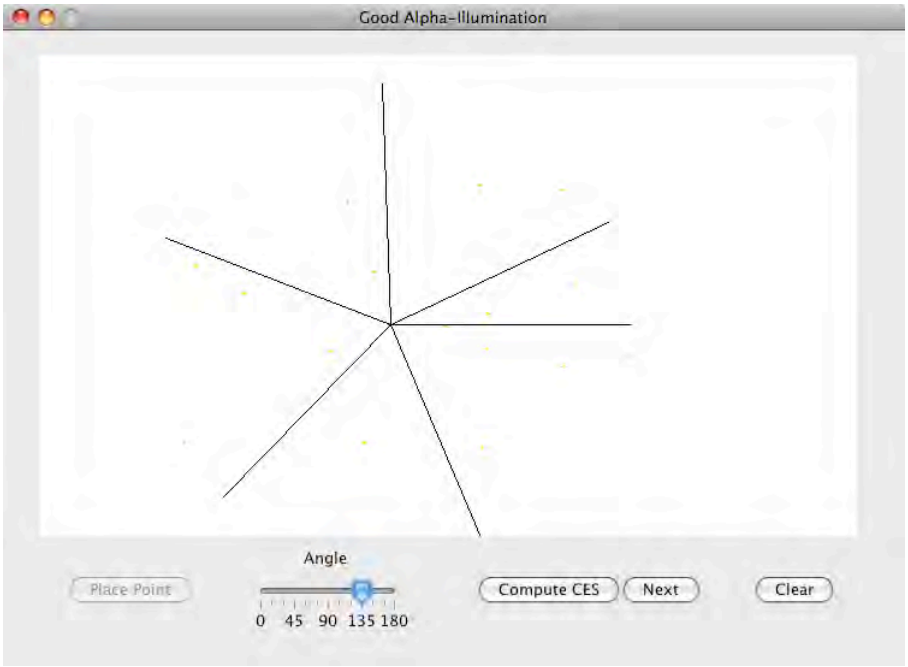


Figure 2.28: The plane is divided into five wedges of angle $67,5°$ and one of $22,5°$.

For a certain subset of lights, the query point no longer is well α-illuminated (see Figure 2.30) and so the algorithm draws the empty wedge of angle α where the verification failed. Afterwards, it proceeds searching the other half of the lights once button `Next` is pressed. As before, each pair of lights turns blue if they were verified to measure the empty angle they

define with the query point.

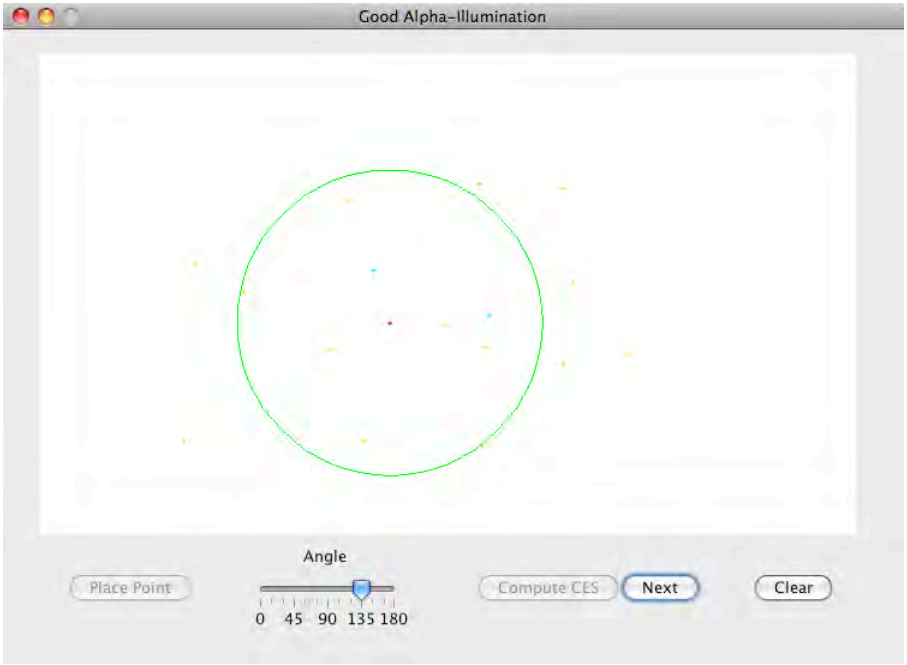


Figure 2.29: The green circle bounds the half of the closest lights to the red point.

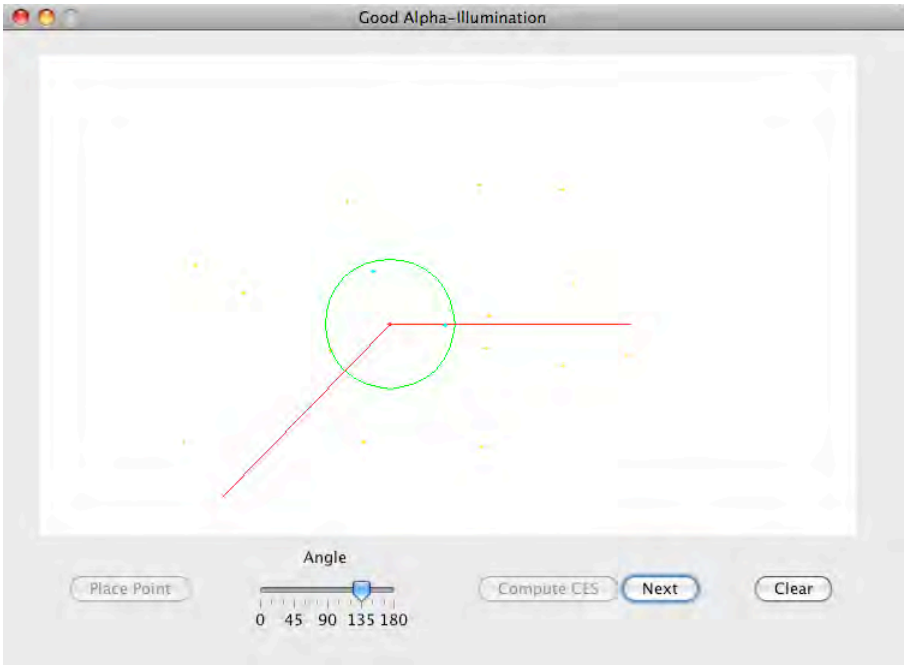


Figure 2.30: The red point is not well $\frac{3}{4}\pi$-illuminated by the lights inside the green circle since there is an empty wedge of angle $\frac{3}{4}\pi$.

If the query point is well α-illuminated, then the algorithm highlights its closest embracing site. Such light is shown in blue and labelled "ces" (see Figure 2.31). The algorithm

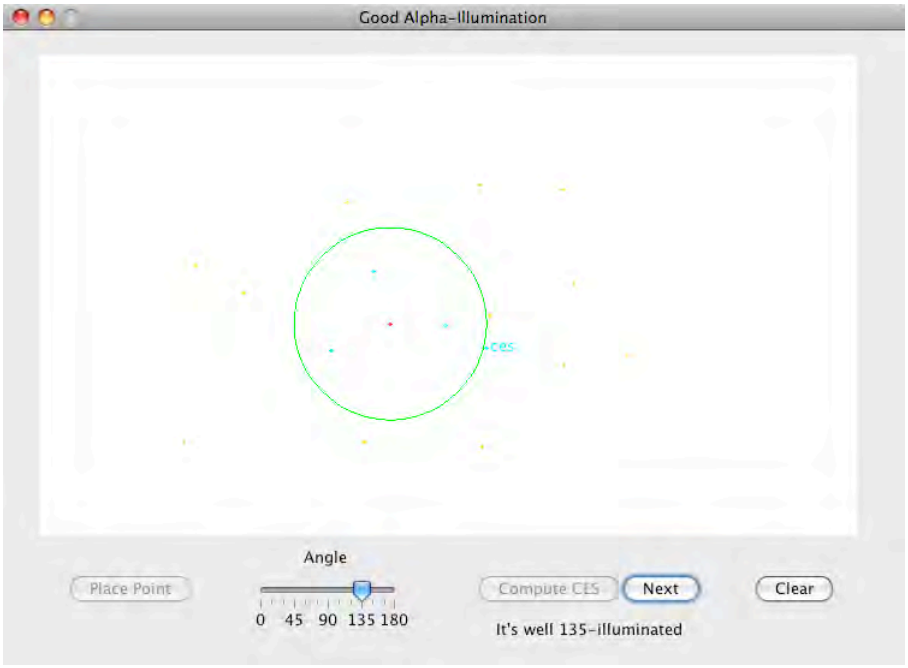


Figure 2.31: The red point is well $\frac{3}{4}\pi$-illuminated by the lights on the green circle. The closest embracing site is shown in blue and labelled "ces".

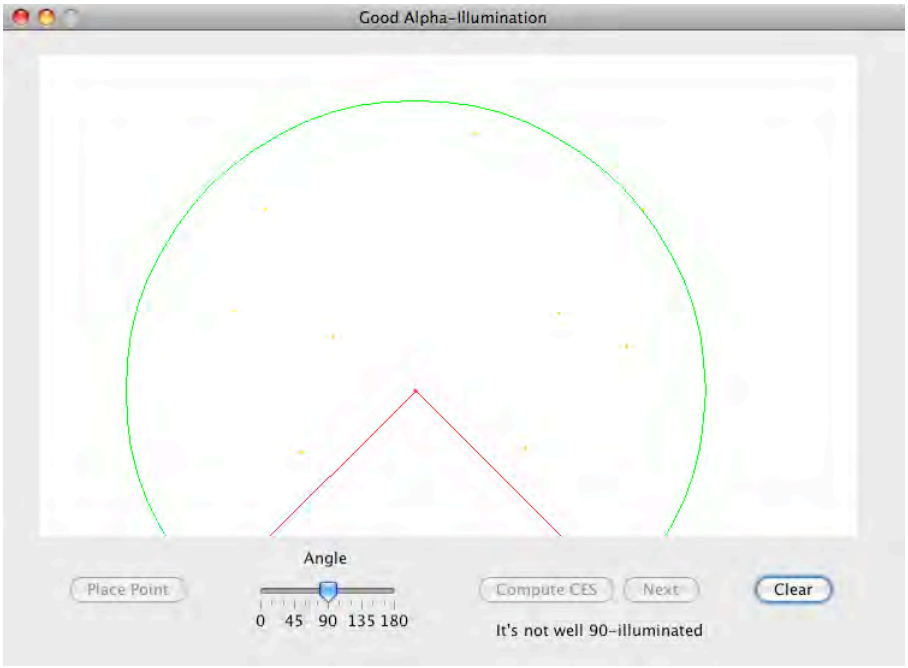


Figure 2.32: The red point is not well $\frac{\pi}{2}$-illuminated by the lights. The red wedge of angle $\frac{\pi}{2}$ is empty.

outputs the message "It's well α-illuminated" on the bottom of the panel, and the value of the minimum embracing range to well α-illuminate the query point on the output window.

Otherwise, the algorithm outputs the message "It's not well α-illuminated". As in previously failed verifications, it shows the final empty wedge of angle α in red (see Figure 2.32).

2.5.3 The α-Embracing Contour

The subject introduced in this section involves good illumination problems, as well as data visualisation and depth problems. Although they appear to be distinct subjects, it is interesting how they are more connected than it seems at first sight. What follows is therefore a short introductory summary on data depth problems. Afterwards, the relation between good α-illumination and data depth becomes apparent.

Data depth has been considered as a measure to evaluate how deep or central a given point is with respect to a multivariate distribution. The notion of depth induces a stratification of the plane into regions of points that share the same depth with respect to a given set of points. The boundaries of these regions are called *depth-contours* and are used as tools for data visualisation since they provide a quick and informative overview of the shape and properties of the point set. Several different notions of depth have been studied, for example, the location depth (also known by half-space depth or Tukey depth [76]) and Delaunay depth [44]. The Tukey depth measures the minimum number of points of a given set that can be separated from a point q by means of a half-plane. One can consider other ways to separate a point by choosing the best to fit a certain application. The depth used in this section is called the *α-depth* and is defined below.

Definition 2.5 *Let S be a set of points on the plane. A point q has α-depth k with respect to S if every open wedge of angle α with apex at q contains at least k points of S and there is, at least, one such wedge containing exactly k points.*

If the points on the border of the wedge were to be considered, then the α-depth would be a generalisation of the Tukey depth. If it were the case, the wedge would replace the half-plane and be seen as a way to separate q. For example, considering closed α-wedges for $\alpha = \pi$, the α-depth corresponds to the Tukey depth. This subject has been recently addressed by Miller et al. [66]. A main concern in current theoretical research on data depth is the construction of depth contours. Tukey depth contours have been studied and constructed by several authors [34, 59, 67], as well as Delaunay depth contours [15]. Contours also have applications in quality illumination, and in particular, they can be applied to good illumination. According to Definition 2.5, points well α-illuminated have α-depth greater than or equal to 1.

In 1-good illumination, the convex hull bounds the points on the plane that are 1-well illuminated. The boundary of the convex hull is therefore a contour that separates 1-well illuminated points from the rest. A similar structure exists for good α-illumination and is called the *α-embracing contour* or the *first α-depth contour*.

Definition 2.6 *Let S be a set of points on the plane and $\alpha \leq \pi$ a given angle. The α-embracing contour of S is the boundary of the region that encloses the points on the plane that are well α-illuminated.*

In Figure 2.33 there is an example of the $\frac{\pi}{2}$-embracing contour represented by a solid pink line. Only the points on the purple area are well $\frac{\pi}{2}$-illuminated. The fact that this embracing contour is not connected proves the following property.

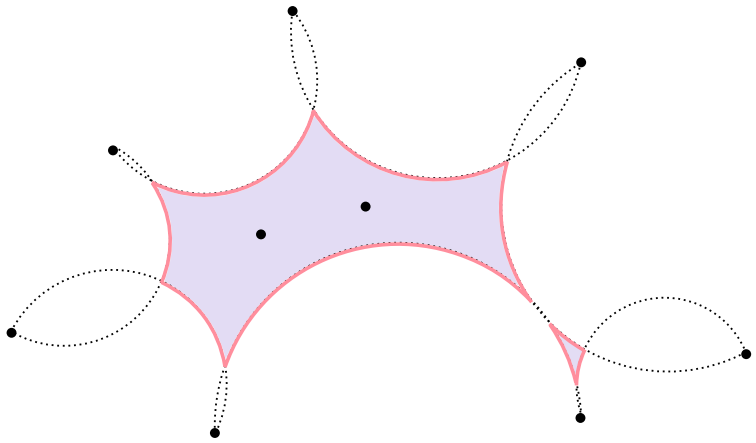


Figure 2.33: The $\frac{\pi}{2}$-embracing contour of the set S is not connected.

Proposition 2.4 *The α-embracing contour of a set S is not necessarily connected.*

The α-embracing contour was thought to have linear size for $0 < \alpha < \pi$ [16, 17]. However, this is not true as Matijevic and Osbild [58] proved. The contour has indeed linear size if $\frac{\pi}{2} \leq \alpha < \pi$ but for a constant $\delta > 0$, if $\delta < \alpha < \frac{\pi}{2}$ then its complexity is $\mathcal{O}(n^{1+\varepsilon})$ for any $\varepsilon > 0$. They also showed where our construction failed for smaller angles. Furthermore, they give upper bounds in the case that α is larger than π. This case is simpler to solve and the contour has complexity $\mathcal{O}(|\mathrm{CH}(S)|)$. They also proposed an algorithm to compute the α-embracing contour (which they call Θ-region) that runs in $\mathcal{O}(\frac{n^{\frac{3}{2}+\xi}}{\alpha} + \mu \log n)$ time, for any $\xi > 0$ and where μ denotes the complexity of the arrangement of arcs. Depth-contours will be further discussed in Section 5.2.3 in Chapter 5.

2.6 Closing Remarks and Future Research

The good illumination problems presented in this chapter cover three different types of quality illumination: 1-good illumination, orthogonal good illumination and good α-illumination. They all consider limited illumination range and therefore, the main objective in each section was to calculate the minimum embracing range to well illuminate a given object and find a closest embracing set for it. The final complexities of the algorithms proposed for each variant are shown in Table 2.1. In that table it is considered that S is a set of n lights, a polygonal line is formed by m line segments and the angle $\alpha \leq \pi$ is fixed. Moreover, minimum embracing range is denoted by M.E.R. and closest embracing set by C.E.S.

	Problem	Algorithm's Complexity
1-Good Illumination	M.E.R. for a point	$\mathcal{O}(n)$ time and space
	C.E.S. for a point	$\mathcal{O}(n)$ time and space
	M.E.R. for a line segment	$\mathcal{O}(n^2)$ time
	C.E.S. for a line segment	$\mathcal{O}(n^3)$ time
	M.E.R. for a polygonal line	$\mathcal{O}(mn^2)$ time
Orthogonal Good Illumination	M.E.R. for a point	$\mathcal{O}(n)$ time and space
	C.E.S. for a point	$\mathcal{O}(n)$ time and space
Good α-Illumination	M.E.R. for a point	$\mathcal{O}(n)$ time and space
	C.E.S. for a point	$\mathcal{O}(n)$ time and space
	α-Embracing Contour	$\mathcal{O}(\frac{n^{\frac{3}{2}+\epsilon}}{\alpha} + \mu \log n)$ time [58]

Table 2.1: Complexities of the algorithms proposed for each type of good illumination.

Section 2.2 had two main goals: calculate the minimum embracing range of S to 1-well illuminate a point q and find a closest embracing triangle for q. Although our algorithms do not improve the previous known bound [29], they have the advantage of finding a closest embracing triangle for q. Every problem presented in this chapter optimises the lights' range by calculating the minimum distance between an object and its closest embracing site. However, there are other possible optimisations, for example, minimising the sum of the distances between an object and the lights of a closest embracing set for it. In the case the object is a point q, a solution is attained if an algorithm identical to the one presented in Section 2.2.1 is applied. Therefore, minimising the sum of the distances between q and the lights of a closest embracing set for q takes $\mathcal{O}(n \log n)$ time, according to Theorem 2.1. For future research, this variant can be extended to other objects as line segments, polygons, etc. Other possible optimisation variants are: minimising the difference of the distances from an object to the farthest and closest light of a closest embracing triangle for such object, minimising the total area of a

closest embracing triangle, maximising the smallest angle of the closest embracing triangle or balancing the distance between an object and the three lights of its closest embracing triangle. The latter could be tackled using approximate methods and heuristics. All these variants pose interesting questions and each presents a new set of problems.

Given two points p and q on the plane, the problem of calculating the minimum embracing range to 1-well illuminate line segment $\overline{pq}$ was solved in Section 2.3. Two algorithms were introduced to this end. Although the first algorithm had a basic approach that worsens its time complexity, it has the advantage of dividing $\overline{pq}$ into several segments and outputting information on how to 1-well illuminate each segment. The method used to divide the line segment is actually the weakest point of the algorithm and it needs to be optimised. The second algorithm takes advantage of an efficient technique called the Parametric Search [61, 62], which makes it quicker than the first one. These optimisation problems were only studied for a point, a line segment and a polygonal line, so there is a large range of objects awaiting to be studied. If the object is a polygon, then this problem is semi-solved since the polygon's boundary can be seen as a closed polygonal line. Calculating the minimum embracing range to 1-well illuminate a whole polygon is harder to solve, though it is known that the minimum embracing range to 1-well illuminate a polygon's boundary is not sufficient to 1-well illuminate the whole polygon (see Figure 2.34). How to 1-well illuminate a polygon or a particular class of polygons remains an unresolved problem. Nevertheless, this is an important issue in situations like prison's security, where it is not sufficient to have just the prison's boundary guarded, but the prisoners inside need to be guarded as well.

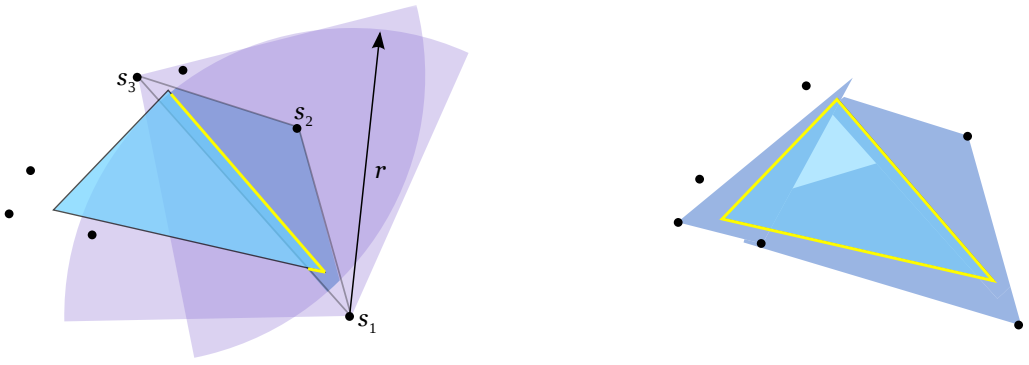


Figure 2.34: (a) Lights s_1, s_2 and s_3 with range r 1-well illuminate the yellow section of the polygon's boundary. (b) Dark blue region is 1-well illuminated and so the polygon's boundary is totally 1-well illuminated, although its interior is not.

This chapter concludes by introducing two variations of 1-good illumination. Orthogonal good illumination is briefly introduced in Section 2.4 and will be discussed again in Chapter

3. Good α-illumination is presented in Section 2.5 and is particularly compelling since it can be seen as a generalisation of 1-good illumination. Actually, good α-illumination and 1-good illumination are intrinsically related as Proposition 2.3 stated. Section 2.5.3 introduced the α-embracing contour, which is a data visualisation tool. This contour is associated with data depth problems but also with good α-illumination since it separates the points on the plane that are well α-illuminated from the rest. Contours will be mentioned again in Section 5.2.3 in Chapter 5.

In conclusion, this chapter presented three variations of good illumination, as well as several optimisation algorithms involving each variant. Two of these were implemented and the resulting applications were shown in Sections 2.2.2 and 2.5.2. There remain, however, a number of interesting questions associated with this topic and involving all three variants: how can every closest embracing set for any object be efficiently computed since it is known that it is not unique? Is there a direct relation between the number of closest embracing sets and the number of lights? How to use routes instead of fixed points to well illuminate a given object, for example, finding a route on a polygon to well illuminate it? All these questions remain unresolved and present compelling challenges for future research regarding good illumination.

Chapter 3

Embracing Voronoi Diagrams

The Embracing Voronoi diagram is a variation of the Voronoi diagram that arose from the concept of 1*-good illumination and merges the notions of proximity and convex dependency. It is a geometric structure that provides a basis to efficiently recalculate the lights' minimum range to keep any moving points* 1*-well illuminated. The Orthogonal E-Voronoi diagram is introduced for the same purpose, but is associated with orthogonal good illumination instead of* 1*-good illumination. This chapter also introduces the closest embracing number as an appropriate tool to merge convex dependency and data depth problems. The closest embracing number of a point is the number of elements in a closest embracing set for such point. Strictly for visualisation purposes, three brute-force approaches were implemented involving these concepts, each of which is illustrated in a dedicated section.*

3.1 Introduction

The Voronoi diagram is certainly a favourite among researchers of Art Gallery problems. In fact, roughly one in sixteen papers in Computational Geometry involves Voronoi Diagrams [21]. This structure has been applied to a surprisingly wide range of problems; there are conferences and symposiums devoted solely to Voronoi diagrams and even art exhibitions on the theme. For example, Kaplan [48] highlighted some particularities of the Voronoi diagrams that make them useful artistic tools. In 2005, the "First International Exhibition of Voronoi Art" took place in Korea, as part of the "2^{nd} International Symposium of Voronoi Diagrams in Science and Engineering". This event and several others paved the way for the so called *Voronoi Art*. Although the Voronoi diagram is not by any means a new concept (dates back to the XVII century), it keeps inspiring researchers to this day. Quoting O'Rourke [70]: "in a

sense, the Voronoi diagram is a structure that records all the needed information on proximity to a set of points or other objects". Given a set S of lights, the Voronoi diagram divides the plane into regions so that there is a light per region and the points on each region are closer to that light than to any other. An exhaustive and unified exposition of the mathematical and algorithmic properties of Voronoi diagrams can be found on a survey by Aurenhammer and Klein [21].

According to Definition 2.2 in Chapter 2, a point q on the plane is 1-well illuminated by a set S of lights if, and only if, there is at least one light of S illuminating q in every open half-plane with q on its border. Recall that the closest embracing site for q is the closest light $s_c \in S$ to q such that q is inside $\mathrm{CH}(S')$, $S' = \{s_i \in S : d(s_i, q) \leq d(s_c, q)\}$. And that the distance between q and its closest embracing site is called the minimum embracing range for q. If q moves continuously or quickly changes from one location to another, there is the need to recompute its minimum embracing range in order to keep the point 1-well illuminated. The *Embracing Voronoi diagram* or *E-Voronoi diagram* is the structure that results from preprocessing the location of the lights of S in order to achieve a quicker solution to this problem. That is, a geometric structure that provides a basis to efficiently recalculate the lights' minimum range to keep any moving points 1-well illuminated. A formal definition of this structure is given below.

Definition 3.1 *Let S be a set of lights on the plane. For every light $s_i \in S$, the Embracing Voronoi region of s_i with respect to S is defined as the following set*

$$\text{E-VR}(s_i, S) = \{x \in \mathbb{R}^2 : s_i \textit{ is a closest embracing site for } x\}.$$

Region E-VR(s_i, S) is also denoted by E-VR(s_i) if set S is clear from the context. The Embracing Voronoi diagram of S (or E-VD(S) for short) is then built up on the union of these regions, that is, $\text{E-VD}(S) = \bigcup_{s_i \in S} \text{E-VR}(s_i)$. While the Voronoi diagram of a set of lights divides the plane into regions associating each point on the plane with its closest light, the E-Voronoi diagram associates each point on the plane with its closest embracing site. Therefore, if $q \in \text{E-VR}(s_i), s_i \in S$, then the minimum embracing range for q is given by the distance between s_i and q. In Figure 3.1 there is an example of the E-Voronoi diagram of a set of four lights. Each light has a different colour associated and its E-Voronoi region is represented in that colour. In this example, the region that these four lights 1-well illuminate is a quadrangle enclosed by their convex hull. Note that the E-Voronoi regions are not necessarily convex or connected as illustrated by E-VR(s_1) and E-VR(s_2), which are represented in yellow and purple, respectively. It should also be noted that the edges of these regions are line segments.

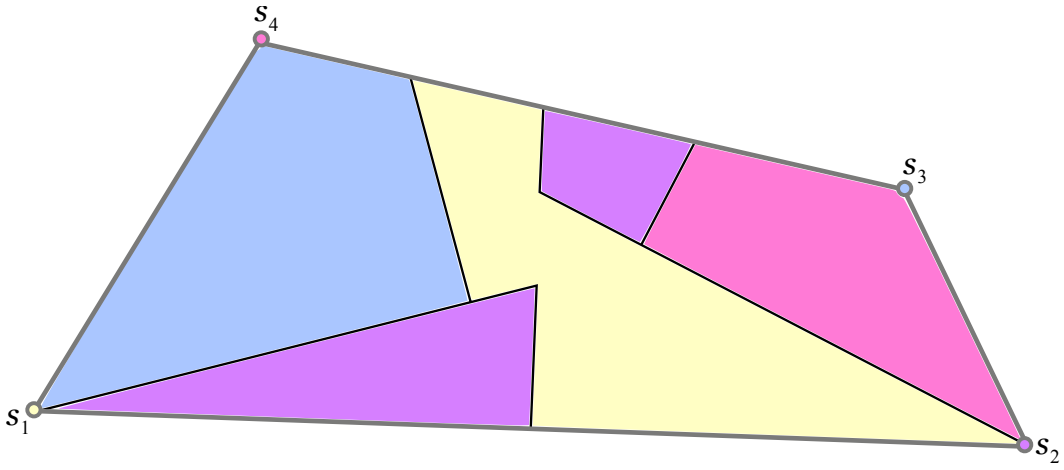


Figure 3.1: The E-Voronoi diagram of $S = \{s_1, s_2, s_3, s_4\}$.

Also mentioned in the previous chapter, given a point q and its closest embracing site $s_c \in S$, set $S' = \{s_i \in S : d(s_i, q) \leq d(s_c, q)\}$ is a closest embracing set for q. The number of elements in this particular set is called the *closest embracing number* of q. This chapter introduces the closest embracing number as an appropriate tool to merge convex dependency and data depth problems. This tool helps to visualise the distribution of the elements of S on the plane, for example, to decide whether the elements are clustered or scattered. Furthermore, the closest embracing number can be seen as notion of depth. If each point on the plane has as depth value its closest embracing number, then the plane can be divided into *embracing layers* and/or *embracing levels*, which help to evaluate how deep or central a given point is with respect to set S.

As in last the last chapter, but strictly for visualisation purposes, three brute-force approaches were implemented involving the concepts introduced in this chapter. Each implementation is further discussed and illustrated in its own section. The structure of this chapter is described below.

Even though not much is known about Embracing Voronoi Diagrams, some of its properties are presented in Section 3.2. These properties are particularly helpful to visualise this structure. Section 3.4 proposes two different strategies to construct the E-Voronoi diagram. Despite having the same running time, the algorithms take different approaches to the solution. The Orthogonal E-Voronoi diagram, which is a variation of the E-Voronoi diagram based on orthogonal good illumination is introduced in Section 3.5. An algorithm to construct such structure is also proposed in the same section. Finally, Section 3.6 introduces the closest embracing number and some of its properties. This concept can be seen as a depth notion, which divides the plane into embracing layers and embracing levels. Closing remarks and a brief

summary of the algorithms and concepts introduced in this chapter are discussed in Section 3.7.

3.2 Properties of the Embracing Voronoi Diagram

As previously mentioned, while the Voronoi diagram of a set of lights divides the plane into regions associating each point on the plane with its closest light, the E-Voronoi diagram associates each point on the plane with its closest embracing site. This section discusses the known properties of the Embracing Voronoi diagram. In the following, let set S be a set of n lights on the plane.

Lemma 3.1 *Given a set S of lights, let light s_c be a closest embracing site for $s_i \in S$. If s_i is inside the convex hull of S, then s_i is a reflex vertex of* E-VR(s_c, S).

Proof: Let s_i and s_c be two lights of S and assume that $s_i \in$ E-VR(s_c, S), that is, s_c is a closest embracing site for s_i. Given the disc $D(s_i, r)$ of radius $r = \text{MER}(s_i) = d(s_i, s_c)$ centred at s_i, suppose that S' is the subset of the lights of $S \backslash \{s_i\}$ that are inside $D(s_i, r)$ (see Figure 3.2(a)). Since s_c is the closest embracing site for s_i, $s_i \notin \text{CH}(S')$. Let s_l and s_r of S be the support lights of s_i regarding CH(S'). Suppose that $s_i^* \in S$ is the closest light of S to s_i outside $D(s_i, r)$. Given a point $x \in D(s_i, \varepsilon)$, for $\varepsilon < \frac{1}{2}(d(s_i, s_i^*) - d(s_i, s_c))$, if x is located on the convex sector determined by the lights s_l, s_i and s_r then x does not belong to E-VR(s_c, S) because set $\{s_i, s_l, s_r\}$ is an embracing set for x. However, a point $y \in D(s_i, \varepsilon)$ on the reflex sector determined by the same lights belongs to E-VR(s_c, S) since set $\{s_c, s_l, s_r\}$ is a closest embracing set for y. Therefore, light s_i is a reflex vertex of E-VR(s_c, S). □

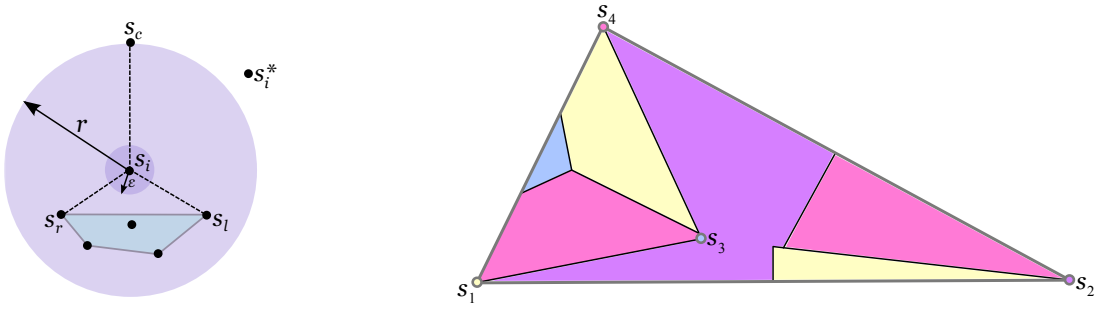


Figure 3.2: (a) CH(S') is shown in blue and $r = d(s_i, s_c)$. Light s_i inside CH(S) is a reflex vertex of E-VR(s_c, S). (b) Light s_3 is a reflex vertex of E-VR(s_2), which is shown in purple.

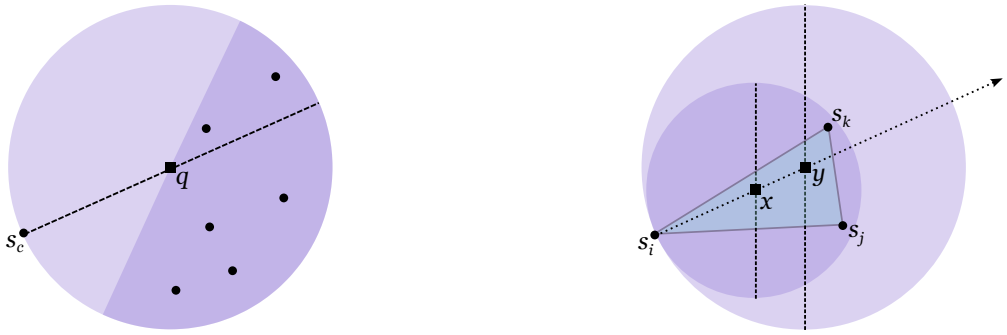
In Figure 3.2(b) there is an example of the previous lemma. The image shows the E-Voronoi diagram of a set of four lights and, as before, each light and its respective E-Voronoi

region are represented by the same colour. Light s_2 (shown in purple) is the closest embracing site for light s_3 (shown in blue), which together with the fact that s_3 is inside CH(S) implies that s_3 is a reflex vertex of E-VR(s_2). The following proposition is a consequence of this lemma.

Proposition 3.1 *The lights of S are vertices of the E-Voronoi diagram of S.*

Proof: The lights of S on the boundary of CH(S) are naturally vertices of the E-Voronoi diagram since such set bounds the diagram. According to Lemma 3.1, the lights of S inside CH(S) are reflex vertices of the diagram. Consequently, all the lights of S are vertices of the E-Voronoi diagram of S. □

Note that a light $s_i \in S$ never is a vertex of its own E-Voronoi region. If so, then s_i would be its own closest embracing site. This is impossible since at least two other lights must be part of a closest embracing set for s_i and they are farther from s_i than s_i itself.



(a) (b)

Figure 3.3: (a) Disc $D(q, d(s_c, q))$ has a semicircle empty of lights and there is at least one light on each side of qs_c. (b) Set $\{s_i, s_j, s_k\}$ is a closest embracing set for every point on $\overline{xy}$.

Proposition 3.2 *The E-Voronoi region of a light $s_i \in S$ is radially monotone (i.e. the intersection between* E-VR(s_i, S) *and any ray from s_i is a line segment or an empty set).*

Proof: Let s_c be a closest embracing site for a point q on the plane, that is, the light whose E-Voronoi region contains q. Disc $D(q, \text{MER}(q))$ has an empty semicircle that separates s_c from the lights that are in the interior of the disc (see Figure 3.3(a)). Observe that there must be at least one light on each side of the line qs_c inside the disc. Let x and y be two different points on a ray from s_i such that $d(s_i, x) < d(s_i, y)$ and assume that x and y are points of E-VR(s_i), that is, s_i is their closest embracing site. Consequently, $D(x, \text{MER}(x)) \subseteq D(y, \text{MER}(y))$ and both discs are tangent to s_i (see Figure 3.3(b)). As previously explained, two lights s_j and s_k

must exist inside $D(x, \text{MER}(x))$ such that $\{s_i, s_j, s_k\}$ is a closest embracing set for x. Lights s_j and s_k cannot be on the empty semicircle of $D(y, \text{MER}(y))$, otherwise s_i would not be a closest embracing site for y. This implies that set $\{s_i, s_j, s_k\}$ also is a closest embracing set for y and for any point on $\overline{xy}$. □

Let $S = \{s_1, \ldots, s_n\}$ be a set of n lights where each light $s_i \in S$ has coordinates $(\cos(\frac{\pi}{2^{i-1}}), \sin(\frac{\pi}{2^{i-1}}))$. It is not hard to verify that the lights of S are placed on the circumference of radius 1 centred at the origin (see Figure 3.4).

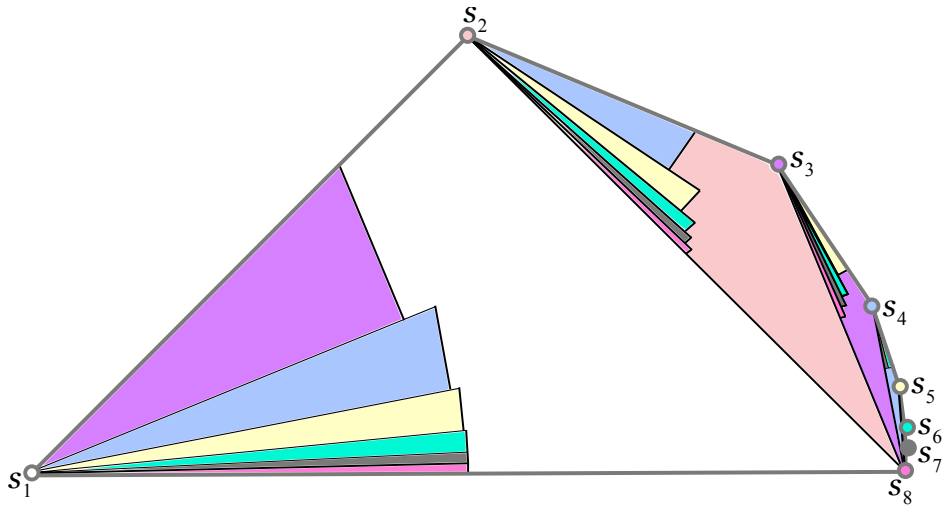


Figure 3.4: A set of triangle fans, each has a light of S as central vertex.

The following lemmas are a direct consequence of the configuration of this set S.

Lemma 3.2 *Using the configuration of set S described above, lights s_1, $s_2, \ldots, s_{k-1}$ are not necessary to construct the E-Voronoi diagram of S restricted to triangle $\triangle(s_k, s_{k+1}, s_n)$, for all $k = 2, \ldots, n-2$.*

Let $\text{PB}(\overline{s_i s_j})$ denote the perpendicular bisector between lights s_j and s_k. For each triangle $\triangle(s_k, s_{k+1}, s_n)$, consider the set of triangles T_{kj}, $j = k+1, \ldots, n$, defined by the line segments $\overline{s_k s_j}$, $\overline{s_k s_{j+1}}$ and $\text{PB}(\overline{s_k s_{j+1}})$. The triangles of T_{kj} form a triangle fan whose central vertex is s_k (see Figure 3.4).

Lemma 3.3 *Using the configuration of set S described above, the following properties are satisfied:*

(a) $T_{kj} \subseteq \text{E-VR}(s_{j+1}, S)$, *for all* $k = 1, \ldots, n-3$ *and for all* $j = k+1, \ldots, n-1$;

(b) *The points of* $\triangle(s_k, s_{k+1}, s_n)$ *that are not on the triangle fan* T_{km} *centred at* s_k *belong to* E-VR(s_k, S)*:*

$$\triangle(s_k, s_{k+1}, s_n) \setminus \bigcup_{m=k+1}^{n} T_{km} \subseteq \text{E-VR}(s_k, S).$$

Finally, the following proposition is built on the previous lemmas, which in turn are based on the particular configuration of set S described above.

Proposition 3.3 *Given a set* S *of* n *lights, the E-Voronoi diagram of* S *has complexity* $\Omega(n^2)$.

Proof: Using the configuration of set S described above, the E-Voronoi diagram of S has a linear number of regions of size $\mathcal{O}(n)$ and a linear number of regions with a linear number of connected components. □

According to the results presented in Section 2.2.3 in the last chapter, if there are n lights on the plane, then the best algorithm to calculate the minimum embracing range of a point q runs in $\Theta(n)$ time. This means that whenever q moves to another location on the plane, its minimum embracing range has to be recalculated in linear time so that q remains 1-well illuminated. The E-Voronoi diagram simplifies this task by calculating MER(q) in sublinear time, as stated in the following proposition.

Proposition 3.4 *Given a set* S *of* n *lights, a closest embracing site for any point on the plane with respect to* S *can be found on the E-Voronoi diagram of* S *in* $\mathcal{O}(\log n)$ *time.*

Proof: It suffices to locate the point on the E-Voronoi diagram of S, which can be done in $\mathcal{O}(\log m)$ time, where m is the size of the planar partition. Since the E-Voronoi diagram has polynomial size with respect to the number of lights of S, $\mathcal{O}(\log m) = \mathcal{O}(\log n)$. □

3.3 Implementation

This section illustrates an application that constructs the E-Voronoi diagram of a set of lights, which is based on a brute-force approach. Since the E-Voronoi diagram is difficult to visualise once there is a large number of lights, the following application was strictly implemented for visualisation purposes. The algorithm simply computes the closest embracing site for every pixel on the application's main panel, as long as such pixel is inside the convex hull of the lights. The application is called "E-Voronoi" and it starts with a white panel (see Figure 3.5). Before clicking on the white panel to add lights, there is the possibility of drawing a grid that

helps to place the lights for particular configurations. Such grid is drawn by pressing button `Grid` (see Figure 3.6).

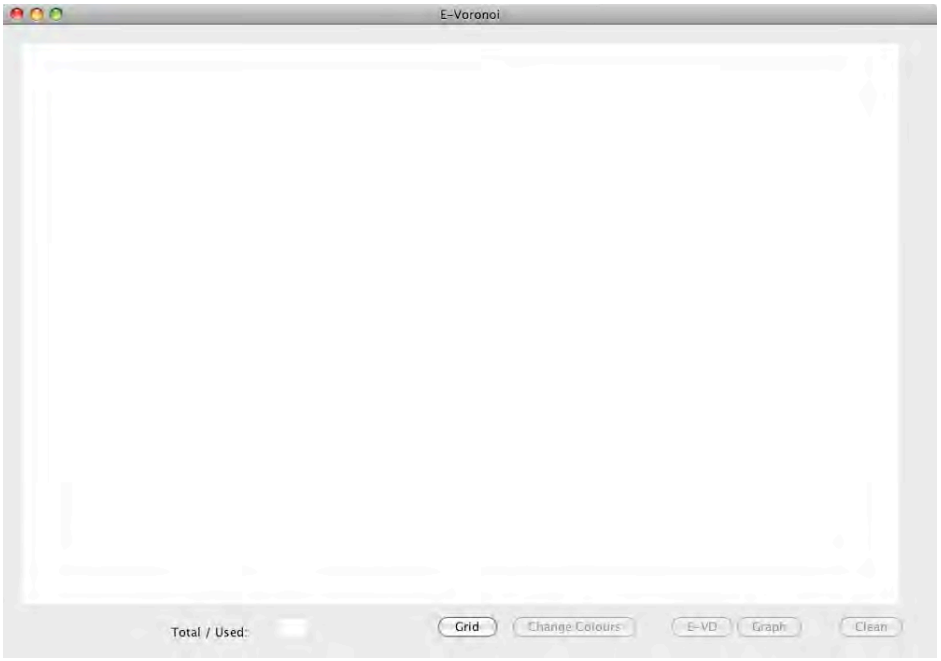


Figure 3.5: Aspect of the application once it is started.

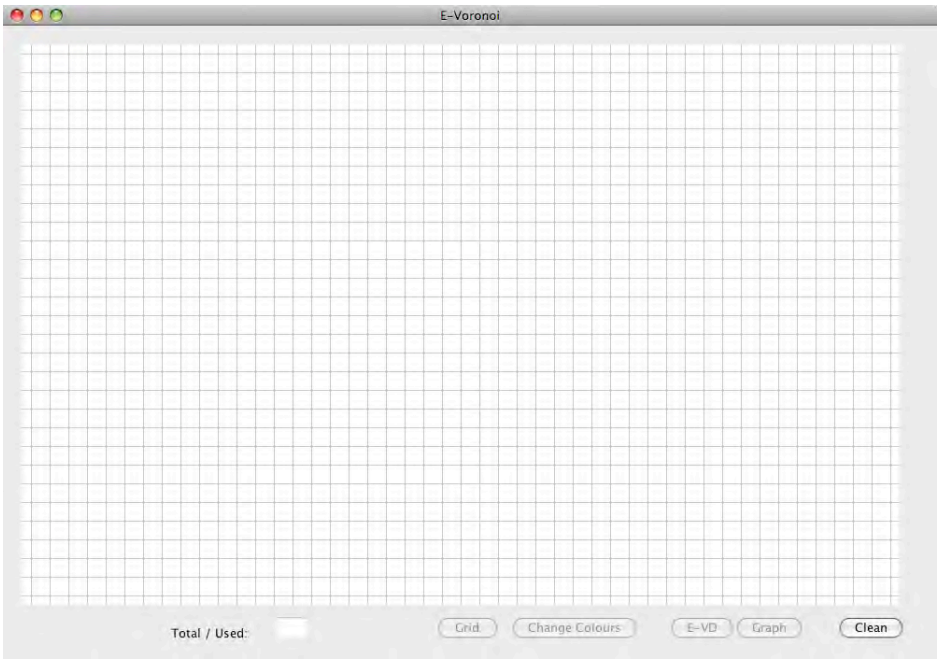


Figure 3.6: Aspect of the application with a grid.

Each light is placed where the user clicks and is represented by a red dot (see Figure 3.7). Button `E-VD` becomes enabled after the user has placed three lights and if pressed, the

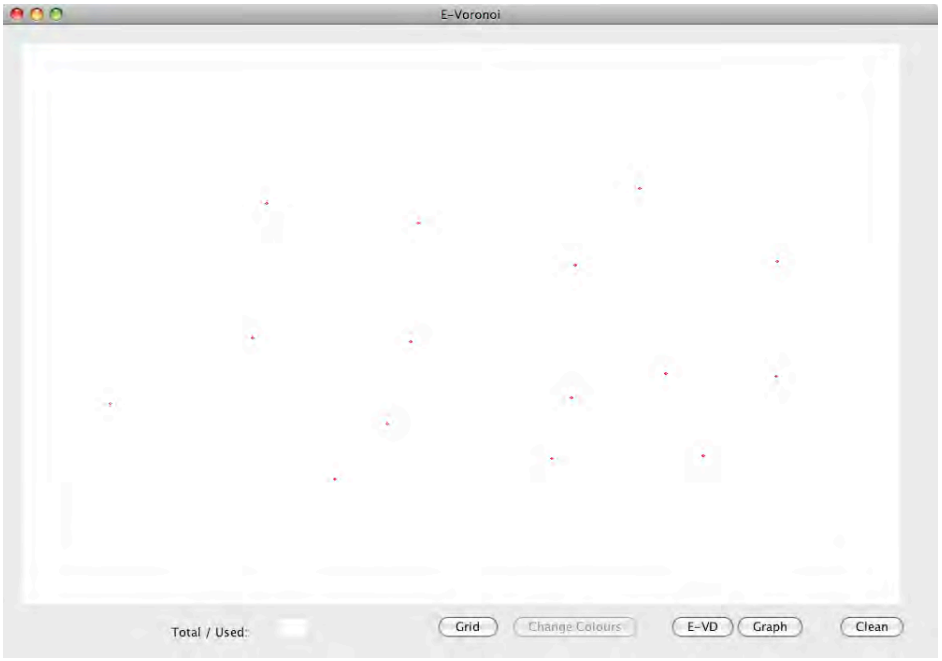


Figure 3.7: The red dots represent a set of lights.

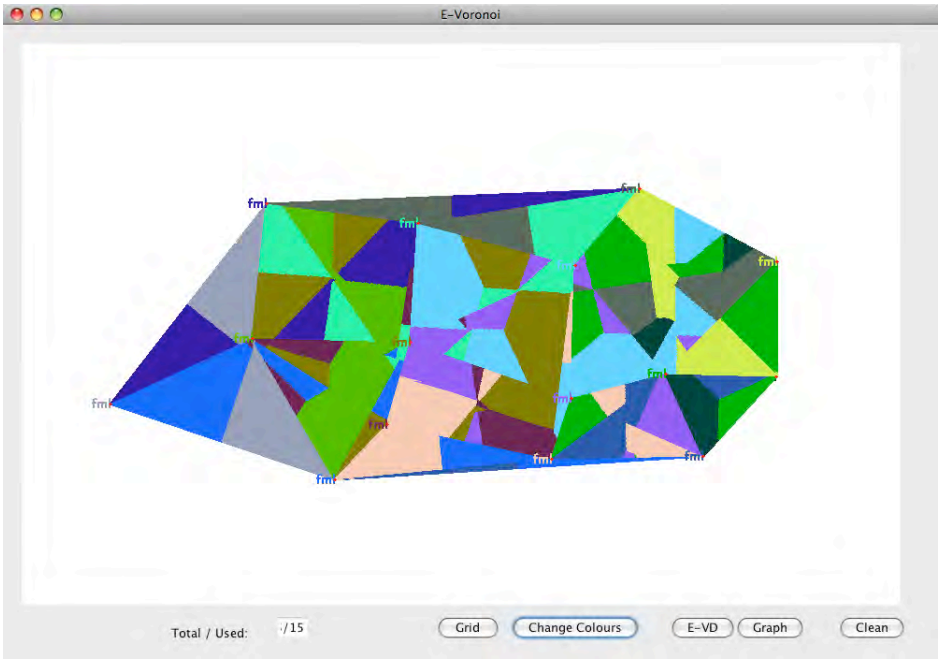


Figure 3.8: The E-Voronoi diagram of the chosen set of lights. Each pixel that is not a light has the colour of its closest embracing site.

algorithm sweeps the panel from left to right. During that sweep, the algorithm allocates a different colour to each light. Therefore, each pixel within the panel is represented in the colour of its closest embracing site (see Figure 3.8). If there is any problem with the colours,

for example, two colours too close to be differentiated, they can be changed using the `Change Colours` button (see Figure 3.9). Lights can be added at any time but in order for the diagram to be updated, a new sweep has to be carried out by pressing the button `E-VD` as before. The colours of the lights are preserved in each sweep.

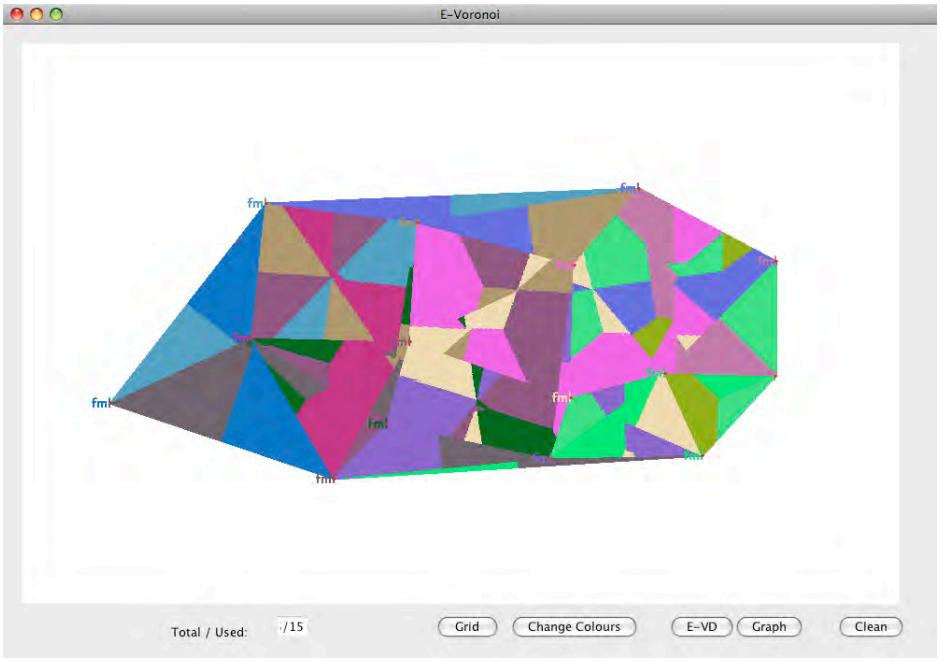


Figure 3.9: The E-Voronoi diagram of the chosen set of lights with different colours.

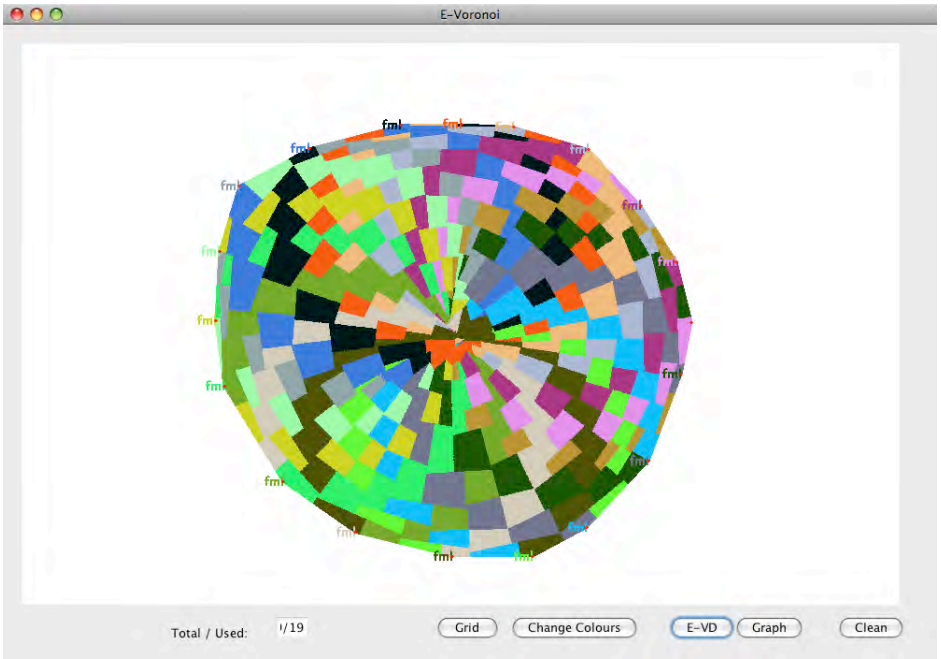


Figure 3.10: The E-Voronoi diagram of a set of lights placed in a circular position.

As in previous applications, button `Clean` restarts the algorithm and clears the panel (including the grid if there is one). There follow two more examples of the E-Voronoi diagram using this application. In Figure 3.10 the lights are placed in a degenerated circular position. In Figure 3.11 a central light shown in yellow was added to the previous set of lights. The E-Voronoi region of such light has a hole. In fact, each E-Voronoi region can only have one hole due to its radial monotonicity, which was stated in Proposition 3.2.

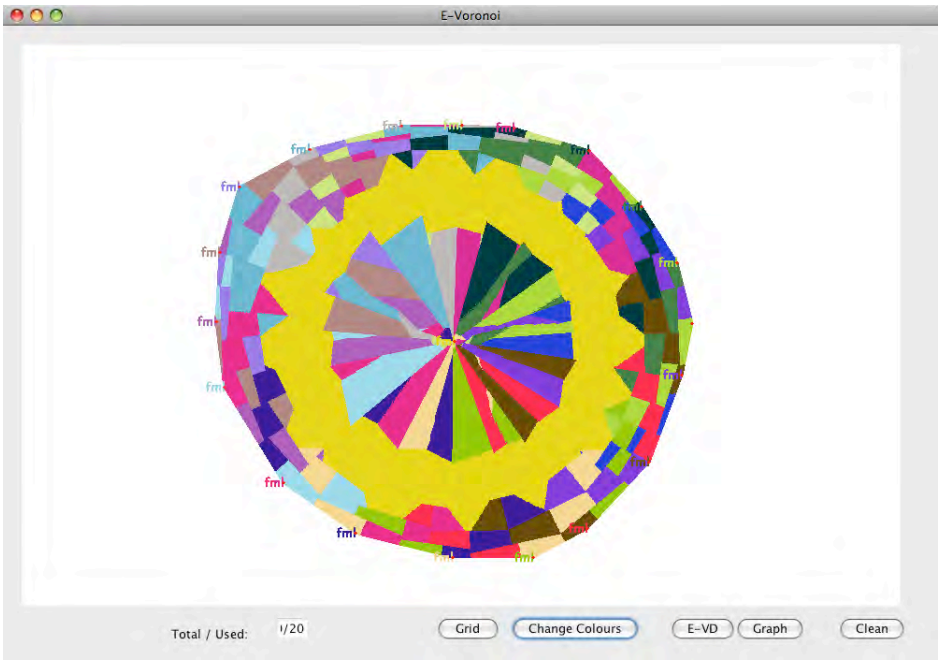


Figure 3.11: The E-Voronoi region of the yellow light has a hole.

3.4 Two Algorithms to Construct the E-Voronoi Diagram

This section proposes two distinct algorithms to construct the Embracing Voronoi diagram of a set S of lights. Despite having the same running time, each algorithm is based on a different strategy. The first algorithm takes a simpler approach: first it splits the interior of $\mathrm{CH}(S)$ into several regions so that all points on the same region have the same closest embracing site. Second, the algorithm computes the closest embracing site for the points on each of the previous regions. This strategy will be further discussed in Section 3.6, but with another purpose. The second algorithm constructs the E-Voronoi diagram by taking advantage of its similarities with higher order Voronoi diagrams.

3.4.1 First Construction of the E-Voronoi Diagram

The algorithm proposed in this section constructs the E-Voronoi diagram of S based on a similar strategy to the one presented in Section 2.3.1 in Chapter 2. First the algorithm divides the interior of CH(S) into several regions where all the points in the same region have the same closest embracing site. Second, the algorithm finds the closest embracing site for the points on each region. The following pseudo-code applies the "Minimum Embracing Range II" algorithm, which was presented in Section 2.2.3 in the last chapter.

ALGORITHM **First construction of the E-Voronoi diagram**

INPUT: Set S of n lights
OUTPUT: E-Voronoi diagram of S

1. $T \leftarrow \{\overline{s_i s_j} : s_i \neq s_j \in S\}$, $B \leftarrow \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$;
2. Compute arrangement $A \leftarrow \{\text{int}(\text{CH}(S)) \cap \{B \cup T\}\}$;
3. For each face a of A do
   (a) Select a random point q in a
   (b) Compute the closest embracing site $s_c \in S$ for q using the "Minimum Embracing Range II" algorithm
   (c) Assign every point on a to s_c
4. Sweep arrangement A and merge neighbouring faces that have the same closest embracing site; the final arrangement is the E-Voronoi diagram of S.

The previous algorithm is illustrated in Figure 3.12. The example shows three steps of the algorithm while it constructs the E-Voronoi diagram of a set S of four lights. First, the algorithm computes sets $T = \{\overline{s_i s_j} : s_i \neq s_j \in S\}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$. The resulting arrangement of lines can be seen in Figure 3.12(a). Consequent to the method used to compute the arrangement of lines, all the points on each of the resulting regions have the same closest embracing site. Second, the algorithm randomly chooses a point on each of these regions and computes its closest embracing site. Then all the other points on the same region are associated with that light (see Figure 3.12(b)). The resulting diagram is almost complete, but some regions need to be merged in order to fully construct the E-Voronoi diagram of S (see Figure 3.12(c)). To conclude, the plane is swept and neighbouring regions that have the same closest embracing site are merged. The following proposition states that the final structure

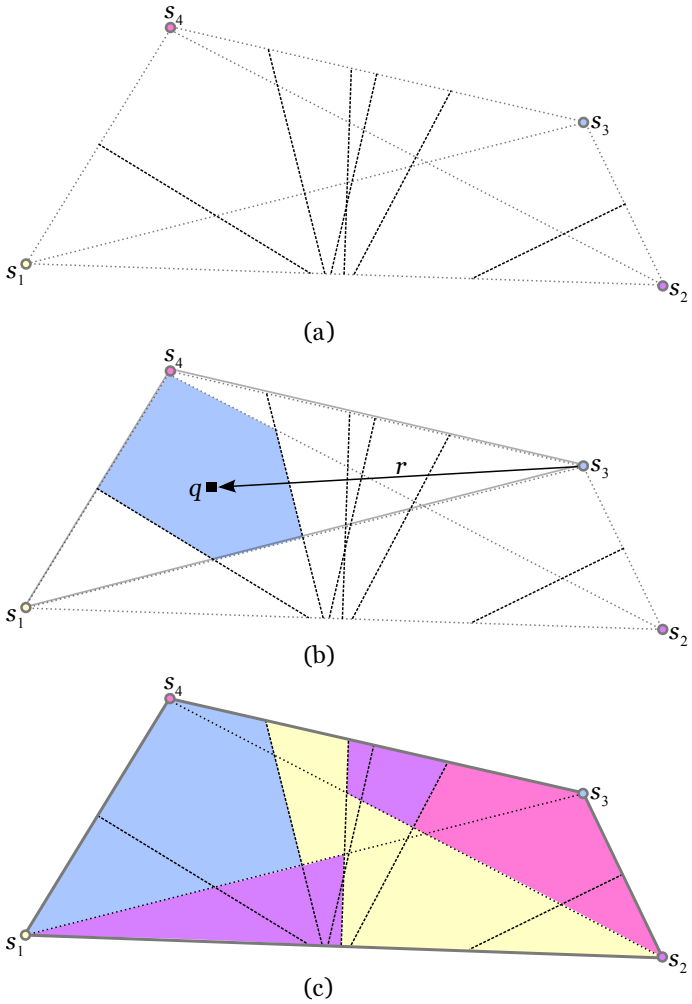


Figure 3.12: (a) Arrangement of lines formed by sets $T = \{\overline{s_i s_j} : s_i \neq s_j \in S\}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$. (b) The closest embracing site for point q is light s_3, $\text{MER}(q) = r$, and so every point on that region is associated with s_3. (c) Diagram before its refinement.

achieved by this algorithm is indeed the E-Voronoi diagram of S. The algorithm's temporal complexity is analysed below in Theorem 3.1.

Proposition 3.5 *Given a set S of lights, the previous algorithm constructs the E-Voronoi diagram of S.*

Proof: Assume that the lights of set S have been processed to form arrangement A by following the steps described above. Now suppose that all points on face $a \in A$ are associated with light $s_c \in S$ and there is a point q on such face whose closest embracing site is light $s_q \neq s_c$. Since a is assigned to light s_c, there must be a point on face a whose closest embracing site is s_c. Let p be such a point. Since p and q have different closest embracing sites, they are either

on opposite sides of PB($\overline{s_q s_c}$) or 1-well illuminated by different closest embracing sets. But any of these would imply that p and q were on different faces of A. Therefore, every point on a face of arrangement A has the same closest embracing site. After the final sweep that merges neighbouring faces that have the same closest embracing site, each face of the final arrangement is an E-Voronoi region and A is the E-Voronoi diagram of S. □

Theorem 3.1 *Given a set S of n lights on the plane, the E-Voronoi diagram of S can be constructed in $\mathcal{O}(n^5)$ time.*

Proof: Given a set S of n lights, computing sets T and B takes $\mathcal{O}(n^2)$ time since they are defined for every pair of lights ($T = \{\overline{s_i s_j} : s_i \neq s_j \in S\}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$). These lines generate up to $\mathcal{O}(n^4)$ intersection points, which means that arrangement A can be computed using a plane sweep that takes $\mathcal{O}(n^4 \log n^2)$ time. Applying the "Minimum Embracing Range II" algorithm to every face of A requires $\mathcal{O}(n^5)$ time, according to Theorem 2.2. To conclude and as stated in the previous proposition, arrangement A is the E-Voronoi diagram of S and so it can be constructed in $\mathcal{O}(n^5)$ time. □

3.4.2 Second Construction of the E-Voronoi Diagram

The following algorithm was developed when the only method known to calculate the minimum embracing range of a point was the method proposed in Section 2.2.1, which runs in $\mathcal{O}(n \log n)$ time. This implied that the running time of the previous algorithm was $\mathcal{O}(n^5 \log n)$. To improve this result, a second algorithm was developed with the clear intention of reducing the $\log n$ factor from the previous complexity. Such goal was achieved with the algorithm described below, but that slight improvement ceased to exist once it was proven that the minimum embracing range of a point could be calculated in linear time [29]. Notwithstanding these facts, the algorithm presented in this section is interesting in its own right since it takes full advantage of common ground between higher order Voronoi diagrams and the E-Voronoi diagram. As previously mentioned, the Voronoi diagram of S [21] divides the plane into regions such that there is one light per region and two points on the same region have the same closest light. This concept can be generalised for higher orders, for example, the third order Voronoi diagram divides the plane into regions so that all the points on the same region have the same three closest lights. Let $\text{VD}_k(S)$ denote the k^{th} order Voronoi diagram of S for $k = 2, \ldots, n-1$. Note that $\text{VD}_{n-1}(S)$ is also designated by Farthest Voronoi diagram.

The idea underpinning the following algorithm is to make the most of higher order Voronoi diagrams and use their information on proximity, that is, search for the points on the plane

that are 1-well illuminated by their k closest lights, $k = 3, \ldots, n-1$. The algorithm starts by constructing $\mathrm{VD}_3(S)$ since three lights form the smallest convex hull possible: a triangle. Points on $\mathrm{VD}_3(S)$ that lie inside the convex hull formed by their three closest lights clearly are 1-well illuminated. Therefore, these sets of points are saved to the next iteration, but before moving forward, each of these points has to be associated with its closest embracing site. To this end, each group of points that have the same closest embracing set is overlaid with the Farthest Voronoi diagram of their three closest lights. The resulting regions are pieces of the final E-Voronoi diagram of S. The iteration that follows constructs $\mathrm{VD}_4(S)$ and employs this diagram to find the points that are 1-well illuminated by their four closest lights. This procedure is repeated as many times as necessary, each for a diagram of higher order, until every point inside $\mathrm{CH}(S)$ is associated with its closest embracing site.

ALGORITHM **Second construction of the E-Voronoi diagram**

INPUT: Set S of n lights
OUTPUT: E-Voronoi diagram of S

1. Compute all the k^{th} order Voronoi diagrams for $k = 3, \ldots, n-1$;
2. For $k = 3$ to $n-1$ and while $A \neq \mathrm{int}(\mathrm{CH}(S))$ do
   $A_k \leftarrow \{\mathrm{VD}_k(S) \cap \mathrm{int}(\mathrm{CH}(S))\}$;
   For each face $a_j \in A_k$ do
   (a) Let $S_j \subseteq S$ be the set of the k closest lights to a_j
   (b) $a_j \leftarrow \{a_j \cap \mathrm{int}(\mathrm{CH}(S_j))\}$
   (c) Intersect a_j with the Farthest Voronoi diagram of S_j
   (d) $A' \leftarrow A' \cup \{a_j\}$
   $A \leftarrow A \cup \{A' \backslash A\}$;
   If $k = n-1$
   Then for each face a_i of A not assigned to a light do
   Assign points of a_i to their farthest light of S
3. Sweep arrangement A and merge neighbouring faces that have the same closest embracing site; the final arrangement is the E-Voronoi diagram of S.

Figure 3.13 illustrates the algorithm above constructing the E-Voronoi diagram of the

same set of four lights used in the example in Figure 3.12. It is not hard to see that the final arrangement constructed by this algorithm is the E-Voronoi diagram of a set S of lights. Each face a of the arrangement A is bounded by the convex hull of the closest embracing set for the points on a. Then, each point q on a is associated with the farthest light to it of its closest embracing set, which is its closest embracing site. Consequently, this algorithm constructs the E-Voronoi diagram of S.

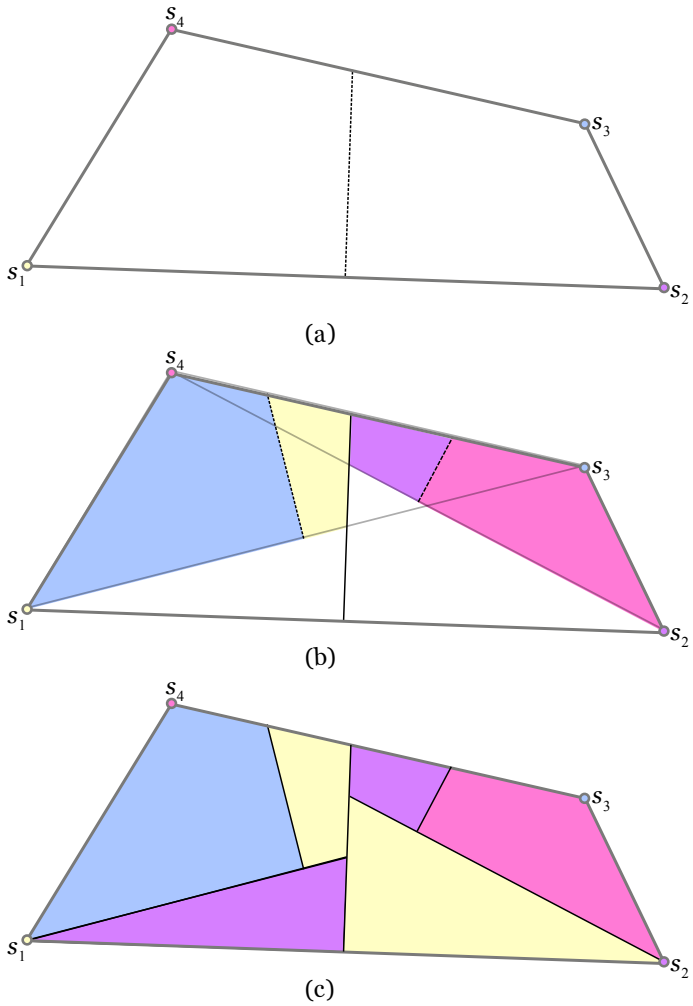


Figure 3.13: (a) Arrangemente A_3 has two faces since $\mathrm{VD}_3(S)$ restricted to the interior of $\mathrm{CH}(S)$ is line segment $\mathrm{PB}(\overline{s_1 s_2})$. (b) Diagram E-VD$(S)$ restricted to the pieces of the faces of A_3 that are inside the convex hull of their three closest lights. (c) Arrangement A before its refinement.

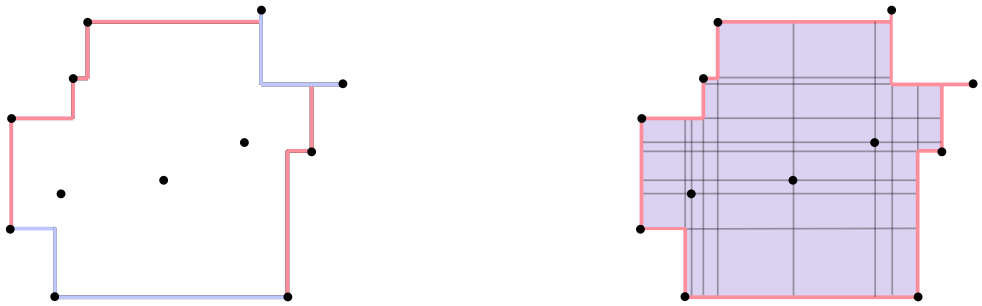
Theorem 3.2 *Given a set S of n lights on the plane, the E-Voronoi diagram of S can be constructed in $\mathcal{O}(n^5)$ time.*

Proof: Given a set S of n lights, all the higher order Voronoi diagrams can be constructed in

$\mathcal{O}(n^3)$ time [37]. Even though the original Voronoi diagram has linear size [21], a higher order Voronoi may have up to $\mathcal{O}(n^2)$ regions. And although A is initially formed only by convex regions, as the algorithm proceeds clipping regions of the Voronoi diagrams, the resulting regions may have holes. Therefore, the arrangement A constructed from the clipping of arrangement A' requires $\mathcal{O}(n^2)$ time to be constructed. In turn, the Farthest Voronoi diagram of a set of lights can be constructed in $\mathcal{O}(n \log n)$ time. Consequently, computing the arrangement A takes $\mathcal{O}(n^4)$ time for each order k, which means the final arrangement is constructed in $\mathcal{O}(n^5)$ time. To conclude, sweeping the final arrangement A to obtain the E-Voronoi diagram of S takes $\mathcal{O}(n^2 \log n)$ time. □

3.5 Orthogonal E-Voronoi Diagram

The orthogonal good illumination was introduced in Chapter 2. According to Definition 2.3, a point q on the plane is said to be orthogonally well illuminated if there is at least one light illuminating q on each quadrant with origin at q. As stated in Theorem 2.7, the minimum illumination range of a set S of lights that orthogonally well illuminates q can be calculated in linear time and space with respect to the number of lights of S. Following the previous line of reasoning, the Orthogonal E-Voronoi diagram, denoted by OE-VD(S), is a structure that helps to efficiently recalculate the lights' minimum range to keep any moving points orthogonally well illuminated. This section proposes an algorithm to construct such diagram.



(a) (b)

Figure 3.14: (a) The orthogonal convex hull of S is formed by four monotone chains. (b) The orthogonal convex hull of S is divided by axis-parallel lines through each light.

Let S be a set of n lights, the points on the plane that are orthogonally well illuminated by S are bounded by the orthogonal convex hull of S. As previously explained in Section 2.4, the prefix orthogonal means that the convexity is defined by axis-parallel point connections. Karlsson and Overmars [50] constructed this structure in $\mathcal{O}(n \log n)$ time by uniting at most four monotone chains (see Figure 3.14(a)). In this way, the first step of the algorithm to

construct the Orthogonal E-Voronoi diagram of S is to build the orthogonal convex hull of S. This polygon is then divided into rectangles by drawing two lines through each light, one vertical and another horizontal (see Figure 3.14(b)). This procedure generates a grid that can be scanned using the sweeping technique. Since the resulting arrangement has a linear number of lines, the grid is formed by a quadratic number of rectangles. The following steps of the algorithm are based on the proposition below.

Proposition 3.6 *Let S be a set of lights and consider the division of the orthogonal convex hull of S into rectangles as described above. If p and q are two points on a rectangle of such division, then the distribution of lights on the quadrants with origin at p is the same as on quadrants with origin at q.*

Proof: Considering the division of the orthogonal convex hull of S into rectangles as described above, let q be a point inside such a rectangle. Suppose that light $s_i \in S$ is on the northeast quadrant with origin at q (see Figure 3.15(a)). Without loss of generality, assume that a point p on the same rectangle has light s_i on the northwest quadrant with origin at p. Since the orthogonal convex hull of S is divided by vertical and horizontal lines, the vertical line through s_i must separate p and q. That is, p and q do not exist on the same rectangle of the division. Since this reasoning can be generalised to any quadrant or pair of points on any rectangle, any two points on the same rectangle have the same distribution of lights on the quadrants with origin at them. □

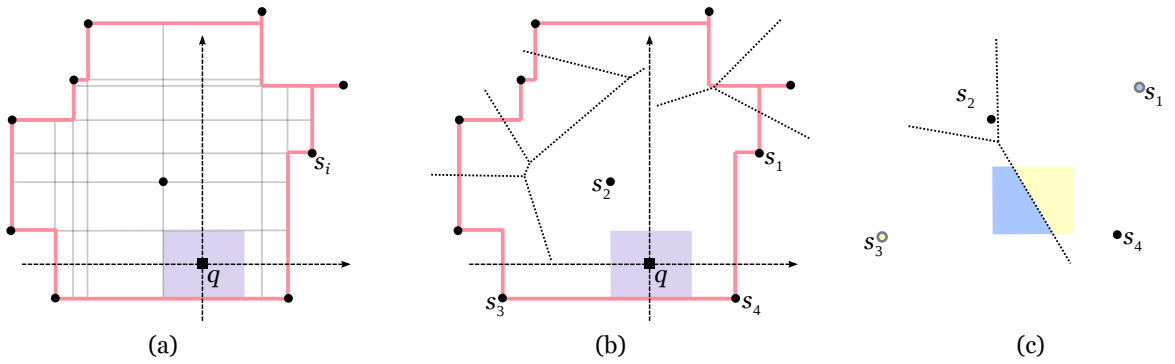


Figure 3.15: (a) The distribution of lights through quadrants with origin at q. (b) The Voronoi diagram of the lights on each quadrant is represented by a dotted line. All points inside the purple rectangle have the same closest embracing set: $\{s_1, s_2, s_3, s_4\}$. (c) The Farthest Voronoi diagram of $\{s_1, s_2, s_3, s_4\}$ is represented in a dotted line. The Orthogonal E-Voronoi diagram of S restricted to the rectangle has two regions, one associated with s_1 and the other with s_3.

In the following, assume that the points on the boundary of the rectangles have the same distribution of lights on quadrants as the interior points. This is true for every point

on the boundary of the rectangles that is not simultaneously a point on the boundary of the orthogonal convex hull. The approach taken by the algorithm below to construct the Orthogonal E-Voronoi diagram of S is similar to the one used by the algorithm described in Section 3.4.1. Therefore, the Orthogonal E-Voronoi diagram is first constructed as the union of several smaller diagrams restricted to the rectangles in which the orthogonal convex hull is divided. Afterwards, the plane is swept and the regions that belong to the same E-Voronoi region are merged. The pseudo-code that follows outlines this algorithm.

ALGORITHM **Construction of the Orthogonal E-Voronoi diagram**

INPUT: Set S of n lights
OUTPUT: Orthogonal E-Voronoi diagram of S

1. Compute the orthogonal convex hull of S;
2. For each light $s_i \in S$ draw a horizontal and vertical line through s_i;
3. For each rectangle of the division do
   (a) Select a random point q inside the rectangle
   (b) Divide the lights of S through the quadrants with origin at q
   (c) For the set of lights $S' \subseteq S$ on each quadrant compute the Voronoi diagram of S'
   (d) For each resulting region, compute the Farthest Voronoi diagram of the lights of the closest embracing set for the points on that region
4. Sweep the rectangles and merge the neighbouring regions that have the same closest embracing site; the final arrangement is the Orthogonal E-Voronoi diagram of S.

Let P be a rectangle of the division of the orthogonal convex hull (see Figure 3.15(a)). Not all the points on P have the same closest embracing set. Therefore, there is the need to compute the set of points on P that share such a set. This is done by intersecting P with four Voronoi diagrams, one per quadrant (see Figure 3.15(b)). Afterwards, the Farthest Voronoi diagram of the lights of the closest embracing set for each region creates a subdivision where all points have the same closest embracing site (see Figure 3.15(c)). The Orthogonal E-Voronoi diagram of S is constructed by repeating this procedure for all the rectangles of the division. To conclude, the plane is swept and the neighbouring subdivisions of the rectangles that have the same closest embracing site are merged.

In Figure 3.16 there is an example of the Orthogonal E-Voronoi diagram of six lights, each light that has a region on the diagram is shown in a different colour. As it is clear from the image, only four of these lights have an associated region on the diagram. Similarly to the E-Voronoi diagram, the regions of the Orthogonal E-Voronoi diagram are not necessarily connected or convex, as illustrated by the blue and yellow regions, respectively. Once the Orthogonal E-Voronoi diagram is constructed, the minimum embracing range to orthogonally well illuminate a point is given by the location of such point on the diagram.

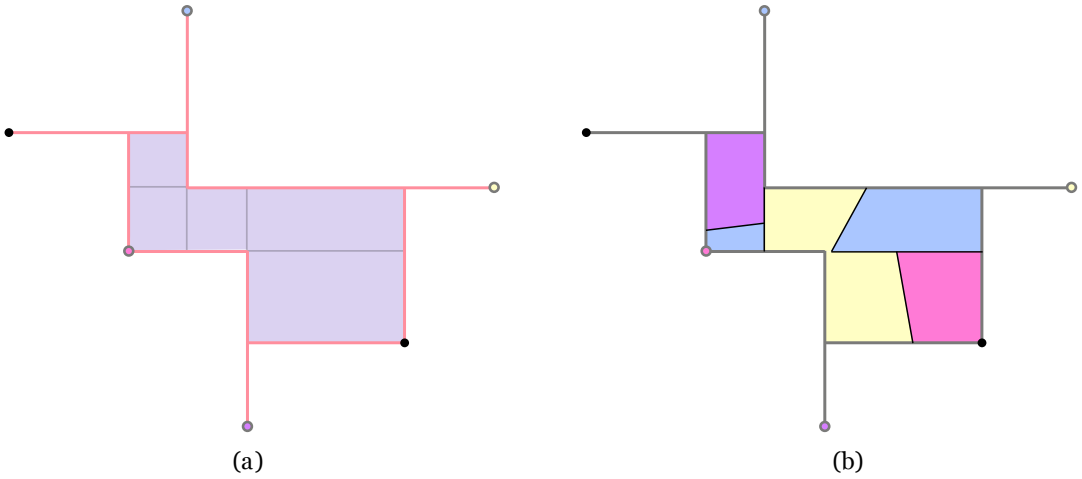


Figure 3.16: (a) A set of six lights and its orthogonal convex hull divided into rectangles. (b) The Orthogonal E-Voronoi diagram of six lights.

It is quite straightforward to see that the previous algorithm constructs the Orthogonal E-Voronoi diagram of S since the rectangles are divided into regions of points that have the same closest embracing set. Afterwards, the Farthest Voronoi diagram subdivides each of these regions by associating each point with its closest embracing site, that is, with the farthest light of its closest embracing set. The temporal complexity of this algorithm is given in the following theorem.

Theorem 3.3 *Given a set S of n lights, the algorithm described above constructs the Orthogonal E-Voronoi diagram of S in $\mathcal{O}(n^4)$ time.*

Proof: Given a set S of n lights, the orthogonal convex hull of S can be computed in $\mathcal{O}(n \log n)$ time [50] as the union of four monotone chains. This convex hull can be divided into rectangles using two plane sweeps (one vertical and another horizontal) in $\mathcal{O}(n \log n)$ time. Since there are a linear number of lines, the arrangement has a quadratic number of rectangles. Each of these rectangles can be analysed in $\mathcal{O}(n^2)$ time as it is explained in the following. Each rectangle is intersected with four Voronoi diagrams, which is a relatively simple process given

the convexity of both structures. As a result, all the points on the same division of the rectangle have the same closest embracing set. Afterwards, the Farthest Voronoi diagram subdivides each of these regions by associating each point with its closest embracing site. Therefore, a first version of the Orthogonal E-Voronoi diagram of S can be computed in $\mathcal{O}(n^4)$ time for every rectangle. A final plane sweep that merges neighbouring regions associated with the same closest embracing site takes $\mathcal{O}(n^2 \log n)$ time and finalises the construction of the Orthogonal E-Voronoi diagram of S. □

3.5.1 Implementation

The application implemented to visualise the Orthogonal E-Voronoi diagram is similar to the application presented previously to visualise the E-Voronoi diagram and it also takes on a brute-force approach. The application is called "Orthogonal E-Voronoi" and starts with a white panel where the user can click to choose the placement of the lights, which are represented by red dots (see Figure 3.17). Button `E-Voronoi` becomes enabled after four lights have been placed. Once that button is pressed, the algorithm constructs the diagram by sweeping the panel from left to right (see Figure 3.18).

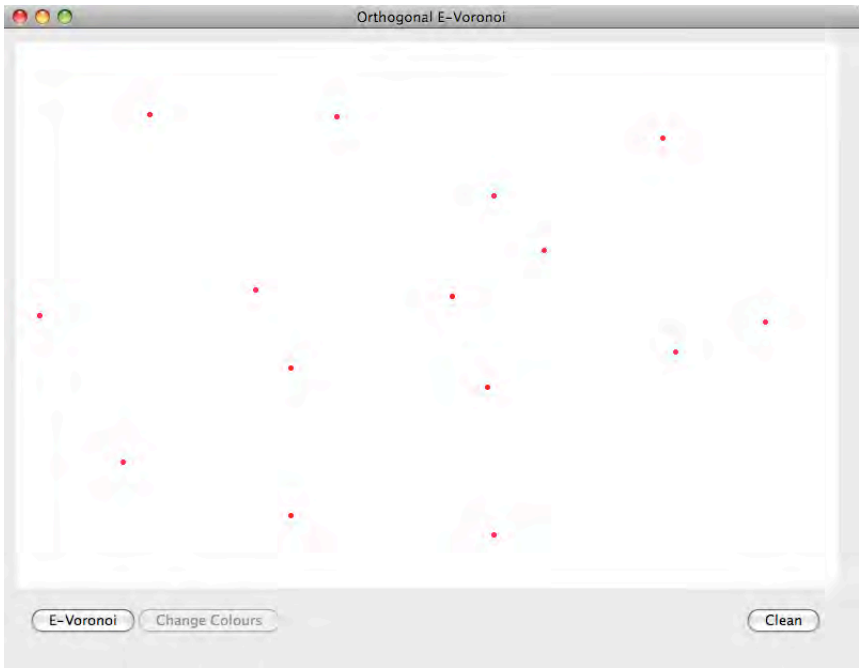


Figure 3.17: Lights are represented by red dots.

As in the previous application, the colours of the lights can be changed using the `Change Colours` button, but note that the colours are preserved in each sweep. Lights can be added at any time but in order for the diagram to be updated, a new sweep has to be carried out by

pressing the button `E-VD`. Also as before, the `Clean` button restarts the application and clears the main panel.

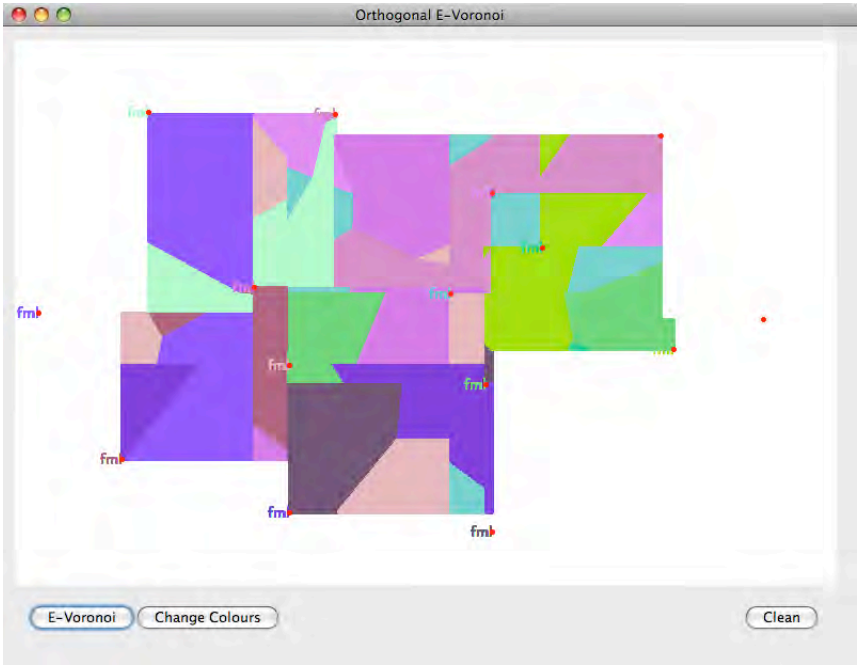


Figure 3.18: The Orthogonal Embracing Voronoi diagram of the set of red dots.

3.6 Closest Embracing Number

This section is not directly associated with quality illumination, unlike the previous sections in this chapter. Therefore, let S be a set of n sites on the plane instead of being a set of lights. The concept introduced in the following involves data visualisation and depth problems. This subject was previously addressed in Chapter 2. Tools associated with this discipline provide a quick and informative overview of the shape and properties of any point set. Given a point q and its closest embracing site $s_c \in S$, set $S' = \{s_i \in S : d(s_i, q) \leq d(s_c, q)\}$ is a closest embracing set for q. The number of elements in this particular set is called the *closest embracing number* of q. This section introduces the closest embracing number as a visualisation tool that merges convex dependency and data depth problems. This tool helps to visualise the distribution of the elements of S on the plane, for example, to decide whether the elements are clustered or scattered. Furthermore, the closest embracing number can be seen as notion of depth. If each point on the plane has as depth value its closest embracing number, then the plane can be divided into embracing layers and/or embracing levels, which help to evaluate how deep or central a given point is with respect to set S. The closest embracing number is formally introduced below.

Definition 3.2 *Given a S of n sites on the plane, a point q has closest embracing number k if q is inside the convex hull of its k closest sites but not inside the convex hull of its $k-1$ closest sites.*

There is an example of this definition in Figure 3.19(a). In that image, point q also is inside $\text{CH}(\{s_1, s_2, s_3, s_5\})$ but $\{s_1, s_2, s_3, s_5\}$ is not a closest embracing set for that point, whereas set $\{s_1, s_3, s_5\}$ is. Therefore, the closest embracing number of q is $|\{s_1, s_3, s_5\}| = 3$. The following property of the closest embracing number is a consequence of its definition.

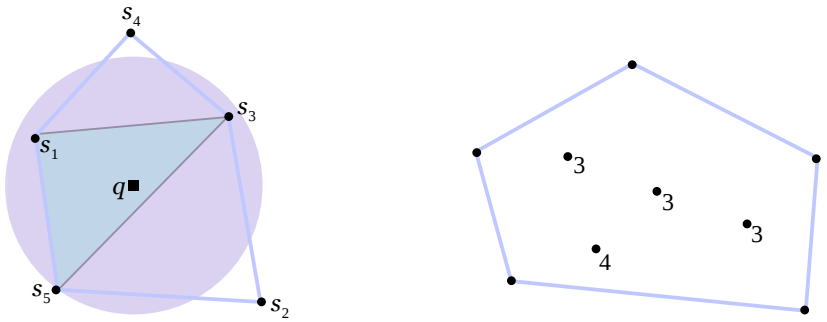


Figure 3.19: (a) Set $\{s_1, s_3, s_5\}$ is a closest embracing set for q and so its closest embracing number is 3. (b) There are only two embracing layers that are not empty: $L_3(S)$ and $L_4(S)$.

Lemma 3.4 *Let S be a set of $n > 3$ sites on the plane. If site $s_i \in S$ is inside $\text{CH}(S)$ then its closest embracing number k_i is larger than 2 and smaller than n, that is, $3 \leq k_i \leq n-1$.*

The closest embracing number of a point q is the number of sites of S whose distance to q is shorter than or equal to the distance between q and its closest embracing site. Therefore, the algorithms proposed in Chapter 2 to find the closest embracing site for a point can be used in this section to calculate the closest embracing number of such point. This observation is formally stated in following proposition.

Proposition 3.7 *Let S be a set of $n > 3$ sites on the plane. The closest embracing number of a point q with respect to S can be calculated in $\mathcal{O}(n)$ time and space.*

Proof: Given a set S of n sites on the plane, the "Minimum Embracing Range II" algorithm, which was presented in Section 2.2.3, finds a closest embracing site $s_c \in S$ for a given point q on the plane. In turn, the closest embracing number of q is given by the number of sites of S whose distance to q is shorter than or equal to the distance between q and s_c. That is, the closest embracing number of q is given by $k = |\{s_i \in S : d(s_i, q) \leq d(s_c, q)\}|$. According to

Theorem 2.2, the closest embracing site for q, and consequently the closest embracing number of q, can be calculated in $\mathcal{O}(n)$ time and space. □

The points of set S can be divided in disjoint subsets in a way that points of the same set have the same closest embracing number. These sets are used as depth tools to evaluate how deep or central a given point is with respect to set S.

Definition 3.3 *Let S be a set of sites on the plane. The set $S_k \subseteq S$ of sites whose closest embracing number is k is called the embracing layer k of S and is denoted by $L_k(S)$.*

Embracing layers classify the sites that are inside $\mathrm{CH}(S)$ in several disjoint subsets. For example, in Figure 3.19(b) each point inside $\mathrm{CH}(S)$ has its closest embracing number associated. In that image, only two embracing layers are not empty: $L_3(S)$ and $L_4(S)$.

Theorem 3.4 *Given a set S of n sites on the plane, the embracing layers of S can be computed in $\mathcal{O}(n^2)$ time and $\mathcal{O}(n)$ space.*

Proof: The embracing layers of S are sets of sites that are inside $\mathrm{CH}(S)$ and share the same closest embracing number. According to Proposition 3.7, the closest embracing number of each site with respect to S can be calculated in $\mathcal{O}(n)$ time and space. Since set S has at most $n-3$ sites inside its convex hull, the embracing layers of S can be computed in $\mathcal{O}(n^2)$ time and $\mathcal{O}(n)$ space. □

If site $s_i \in S$ has a large closest embracing number, then the closer sites to s_i do not surround it. In an ideal situation, most sites would have closest embracing number 3 since that would mean that the sites of S were uniformly distributed on the plane. Since embracing layers classify sites, embracing levels group points on the plane that have the same closest embracing number, as long as they are inside $\mathrm{CH}(S)$.

Definition 3.4 *Let S be a set of n sites on the plane. If a point q inside $\mathrm{CH}(S)$ has closest embracing number k, then q is a point on the embracing level k.*

As previously discussed for sites, if a point q has a small closest embracing number then the closest sites to q surround it. Embracing levels are harder to construct since they are regions on the plane and it is not possible to calculate the closest embracing number of every point on such regions. Therefore, the following theorem presents a strategy to construct embracing levels. This approach is similar to the one used by the first algorithm proposed to construct the E-Voronoi diagram (Section 3.4.1).

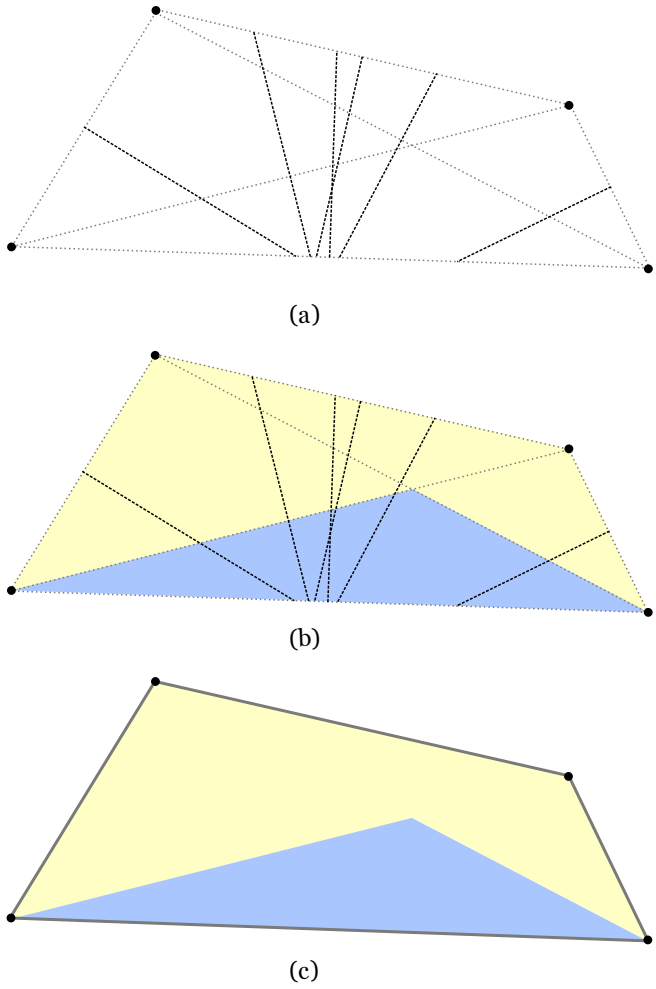


Figure 3.20: (a) The arrangement of lines T and B restricted to CH(S). (b) Yellow regions have closest embracing number 3 and blue have closest embracing number 4. (c) There are only two embracing levels of S.

Theorem 3.5 *Given a set S of n sites on the plane, the embracing levels of S can be constructed in $\mathcal{O}(n^5)$ time.*

Proof: Consider the line segments of sets $T = \{\overline{s_i s_j} : s_i \neq s_j \in S\}$ and $B = \{\text{PB}(s_i, s_j) : s_i \neq s_j \in S\}$ restricted to CH(S) (see Figure 3.20(a)). These lines generate up to $\mathcal{O}(n^4)$ intersection points, which means that this arrangement of lines can be computed using a plane sweep that takes $\mathcal{O}(n^4 \log n^2)$ time. The points on each face of such arrangement have the same closest embracing site and set, and so the same closest embracing number (see Figure 3.20(b)). According to Proposition 3.7, the closest embracing number of a point can be calculated in $\mathcal{O}(n)$ time. Therefore, the closest embracing number can be calculated in linear time for a random point on each face of the arrangement. Then all the other points on the same face are

given that number. This procedure calculates the closest embracing number of every face of the arrangement in $\mathcal{O}(n^5)$ time. Merging the neighbouring faces of the arrangement that are on the same embracing level takes $\mathcal{O}(n^4 \log n^2)$ time (see Figure 3.20(c)). In conclusion, the embracing levels of S can be constructed in $\mathcal{O}(n^5)$ time. □

3.6.1 Implementation

The application shown in Section 3.3 was extended to also include embracing layers and levels. The renewed application opens in two windows, the main one is titled "Closest Embracing Number" and the other one "Labels". This implementation takes once again a brute-force approach. In the case of embracing levels, the algorithm calculates the closest embracing number for every pixel on the main panel. As the user clicks on the white panel to add sites (shown as red dots), the window "Labels" outputs all the possible depth levels (see Figure 3.21). Depth levels start at 3 and each level has a different colour allocated.

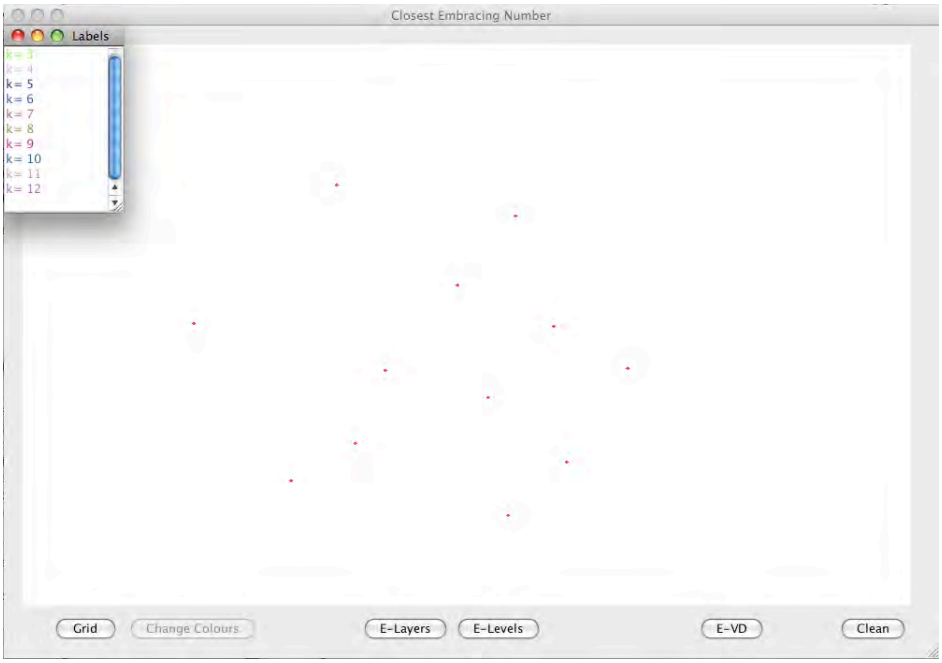


Figure 3.21: Sites are represented by red dots.

When pressed, button `E-Layers` runs the method to calculate the embracing layers of the set of sites. Each site inside the convex hull of sites is assigned its closest embracing number (see Figure 3.22). The "X" that appears on the "Labels" window marks which depth levels exist for that set of sites. The sentence "Used = ..." appears on the bottom of the main window and counts how many different depth levels exist for that set.

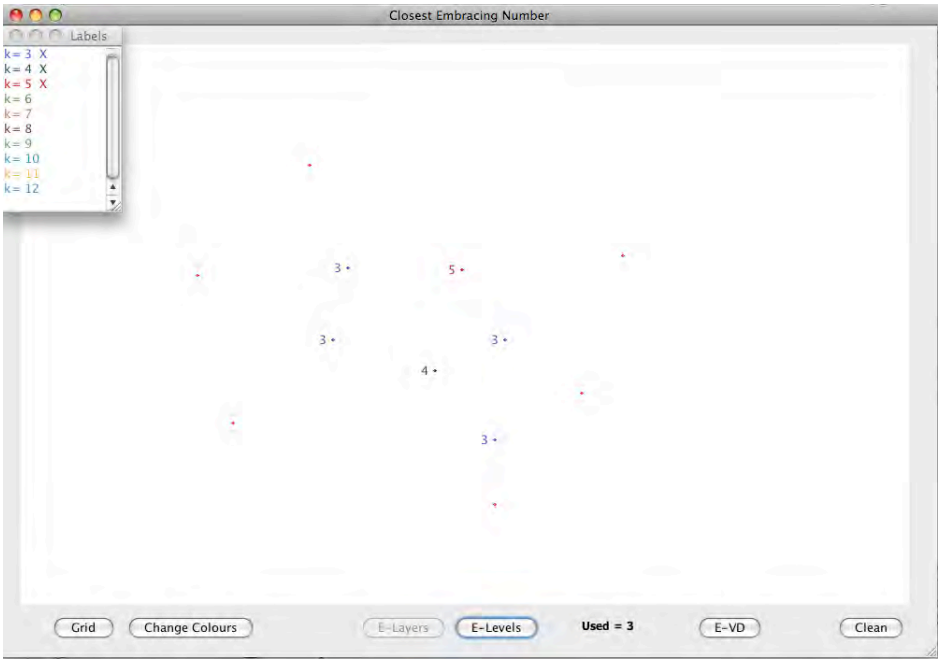


Figure 3.22: There are three non-empty embracing layers: $L_3(S)$, $L_4(S)$ and $L_5(S)$.

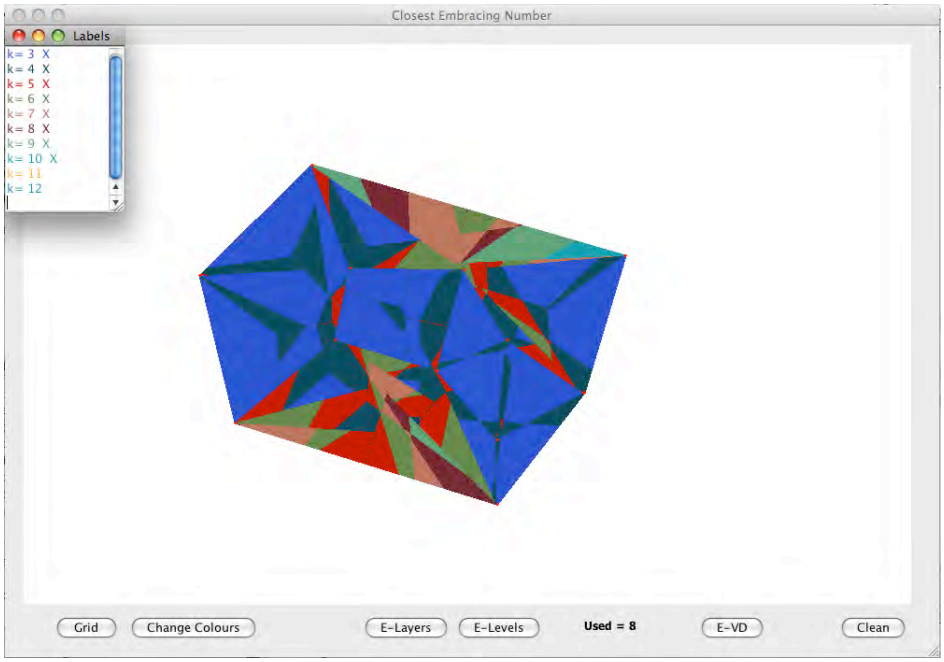


Figure 3.23: There are eight different embracing levels.

If pressed, button `E-Levels` constructs the embracing levels of the set of sites (see Figure 3.23). Note that window "Labels" changes because the points on the plane have a larger diversity of depth levels than sites. As in the application for the E-Voronoi diagram, it is possible to change the colours using the button `Change Colours`. Since it is the same application, but-

ton `E-VD` still constructs the E-Voronoi diagram of the set of sites (see Figure 3.24). Finally, button `Clean` restarts the application and clears both windows.

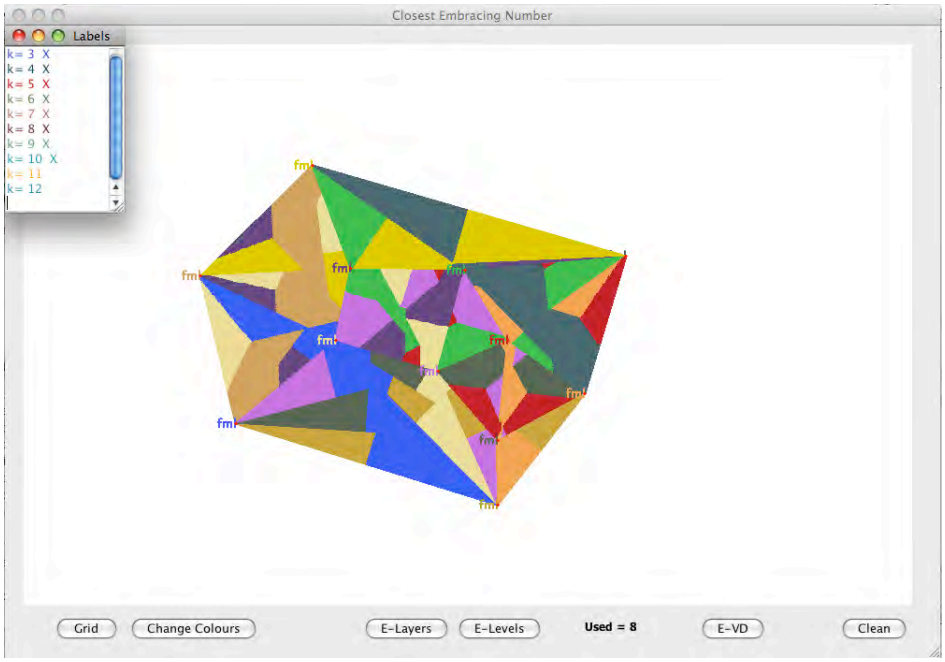


Figure 3.24: The Embracing Voronoi diagram of the red dots.

3.7 Closing Remarks and Future Research

The Embracing Voronoi diagram merges two different geometric concepts: convex dependency and proximity. Although it was first introduced in association with 1-good illumination of minimum range, it was then adapted to orthogonal good illumination. The Embracing Voronoi diagram of a set S of lights was developed by the necessity of a basis to efficiently recalculate the lights' minimum range to keep any moving points 1-well illuminated. This structure is a result of preprocessing the lights of S in order to find a closest embracing site for any point inside $\mathrm{CH}(S)$ in sublinear time. Several properties of the E-Voronoi diagram were presented throughout the chapter, as well as two algorithms to construct such structure. Despite having the same running time, the algorithms present rather different strategies to solve the problem. A variant of the E-Voronoi diagram associated with orthogonal good illumination was introduced in Section 3.5, as well as an algorithm to construct it. The final complexities of the algorithms associated with Embracing Voronoi diagrams that were proposed in this chapter are shown in Table 3.1. In that table it is considered that S is formed by n lights and C.E.S. denotes closest embracing site.

Problem	Complexity
Size of E-VD(S)	$\Omega(n^2)$
Construct E-VD(S)	$\mathcal{O}(n^5)$ time
C.E.S. for a point on E-VD(S)	$\mathcal{O}(\log n)$ time
Construct OE-VD(S)	$\mathcal{O}(n^4)$ time

Table 3.1: Size and final complexities of the algorithms associated with E-Voronoi diagrams.

Knowing that the closest embracing site for a point can be computed in linear time (according to Theorem 2.2 in Chapter 2), if the movement of q is made discrete using m fixed locations, then this problem is solved in $\mathcal{O}(mn)$ time. If the E-Voronoi diagram of S is constructed in a prior step, q can be located in $\mathcal{O}(\log n)$ time for each new position. Consequently, this problem is solved in $\mathcal{O}(m \log n)$ time. This complexity can be constant (except for the first step) if point q moves continuously following a previously known route. Even though calculating the minimum embracing range of a point was the only case studied, the diagram can also be used to compute such range for other objects like line segments, polygons, sets of points, etc. It has become apparent that this diagram can be an important tool regarding this type of problems, although much remains unknown. For example, is the diagram's size really quadratic? How to compute the diagram restricted to a line segment or even a polygon? Is this easier than constructing the whole diagram? Furthermore, the algorithms proposed to construct the diagram can probably be improved and their temporal complexity lowered. Further research is needed to solve these problems more effectively. Regarding the Orthogonal E-Voronoi diagram, even its real size is unknown. It is clear that most concepts that were introduced in this section have not been fully studied yet and that most approaches to them are unresolved problems and compelling future research. Therefore, there is still a lot to research regarding Embracing Voronoi diagrams.

The closest embracing number was introduced in Section 3.6 and is not directly associated with quality illumination. In fact, the closest embracing number involves data visualisation and depth problems, since it is a visualisation tool that provides a quick and informative overview of the shape and properties of any point set. Nevertheless, it is associated with convex dependency as a point on the plane has closest embracing number k if there are k elements on a closest embracing set for that point. Therefore, if a point q has a large closest embracing number, then the sites that are closer to q do not surround it. Embracing layers and levels were introduced as visualisation tools to classify elements of S and points on the plane, respectively. Although embracing layers are pretty straightforward to construct, embracing levels rise some interesting issues. For example, how to lower the algorithm's temporal complexity to construct them? How to efficiently use the information these structures provide to guarantee a well

illuminated route using lower embracing levels? The following table summarises the results of this particular section; closest embracing number is denoted by C.E.N.

Problem	Algorithm's Complexity
C.E.N. of a point	$\mathcal{O}(n)$ time and space
Construct the embracing layers of S	$\mathcal{O}(n^2)$ time and $\mathcal{O}(n)$ space
Construct the embracing levels of S	$\mathcal{O}(n^5)$ time

Table 3.2: Complexities of the algorithms associated with the closest embracing number.

To conclude, most of the problems associated with Embracing Voronoi Diagrams remain unresolved, even though these visually appealing structures are particularly compelling to research. The notions that define 1-good illumination - convexity and proximity - can be used in visualisation and depth tools such as the closest embracing number and its related structures. However, efficiently constructing embracing levels remains a challenge as complex as finding a lower complexity algorithm to construct Embracing Voronoi Diagrams. Thus each of these concepts merits a deeper analysis in future research, since it is clear that such studies will give rise to new and interesting results.

Chapter 4

Minimum 2-Coverage

Given a set of antennas, the geometric optimisation problem solved in this chapter is aimed at minimising the antennas' transmission range so that a path on a region and/or the whole region is within range of at least two antennas. If a region is fully covered by multiple antennas, then it is ensured that there are no shadow/breach areas even when an antenna fails. Different versions of this problem arise for different types of regions, such as a line segment, a planar graph, a polygonal region, a set of points and the whole plane.

4.1 Introduction

Problems involving wireless ad-hoc networks (or just sensor networks) have emerged in the last few years as a result of the fast development of the related technology, resulting in an area of research that is under constant development. These networks have great long-term economic potential and pose many new system-building challenges [64]. Sensor networks can be used to solve a great diversity of problems that range from enemy tracking to habitat monitoring, environment observation and forecast systems (systems that model and forecast physical processes, such as environmental pollution) and even health applications [81]. The problems solved in this chapter are associated with *coverage*. Although most of the applications in this field involve sensor networks, this type of monitoring does not have to be necessarily performed by sensors. Coverage also includes radars, antennas, routers and basically any device that is able to send or receive some sort of wireless signal. Each of these devices can be placed almost anywhere within a specific region, so coverage is really the discipline that measures the quality of the chosen device scheme. In other words, coverage measures the quality of the solution obtained, having in mind the specific challenges that characterise each

application. Let R be a region on the plane that is monitored by a given sensor network. Each solution found to cover R has to deal with several key questions, such as: how guaranteed is the detection of a critical event within a short-time interval? What is the extent of the region that is not monitored?

Coverage can be seen from two opposite perspectives. In the worst-case coverage there is an attempt to locate the regions of R that are hidden from the sensors. These areas are known as shadow/breach regions and determine the extent of the region that is not secured. On the other hand, the best-case coverage is characterised as an attempt to locate the areas that are within reach of as many sensors as possible, thus identifying the "best" monitored regions of R. The following optimisation problems aim to tune a sensor network so that a given region is within range of at least two sensors or to ensure the existence of a path on such region that stays as close to two sensors as possible. In other words, a best-coverage path. Let S be a set of n points on the plane that represent the location of n devices, which are able to send or receive some sort of signal, like antennas or sensors. The devices of S are homogeneous in the sense that they all have the same power transmission range $r \in \mathbb{R}^+$. Let R be a polygonal region that models a street network (see Figure 4.1, the background image was taken from Google Maps). People and vehicles can move within R to reach specific locations there.

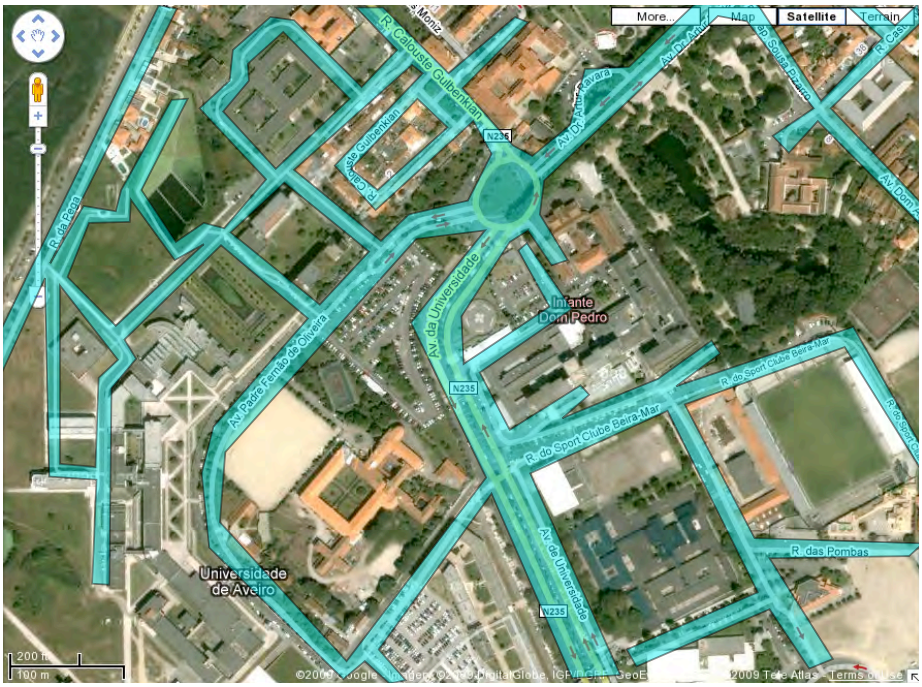


Figure 4.1: The blue polygonal region models a street network.

In the following images, the set S of antennas is located at the buildings' tops and not on the streets, so no antennas are found on R. Nevertheless, the algorithms proposed in this

chapter work for both cases. Regions within reach of at least two antennas are considered to be well covered. In Figures 4.2 and 4.3, there are two examples of a polygonal region, each with a set of homogeneous antennas (represented by black dots) whose coverage area is shown in white. The antennas' range in Figure 4.2 is shorter than in Figure 4.3. The best covered regions, that is, regions that are within reach of at least two antennas are shown in dark blue. In such areas, any traffic would get a signal transmitted from at least two antennas. If a region is fully covered by multiple antennas, then it is ensured that there are no shadow/breach areas even when an antenna fails.

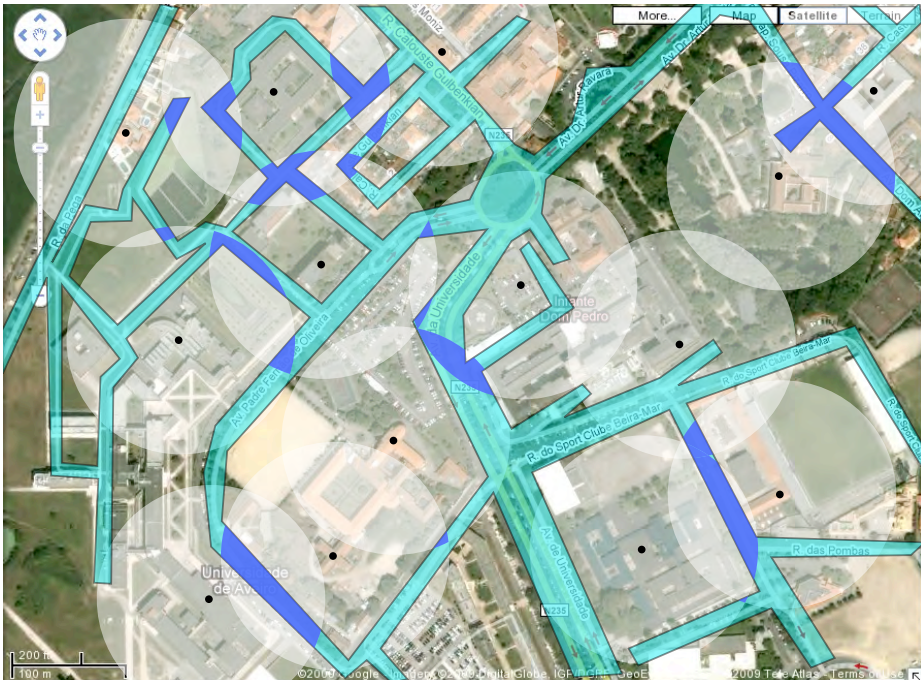


Figure 4.2: Regions within reach of at least two antennas are shown in dark blue.

Clearly the main coverage issue is centred around critical conditions that require reliable monitoring and immediate intervention like fires, disasters or leaking of toxic liquids/waste. Like the search for survivors in an emergency situation or enemy tracking, this type of instances carries stronger coverage needs where failures cannot occur. As illustrated in Figures 4.2 and 4.3, it can be seen that larger transmission ranges provide better coverage. However, as larger ranges also result in higher costs, it is appropriate to balance one against the other. Moreover, the lifespan of the network is extended by reducing the devices' sensing and transmission range as much as optimum, thus enhancing the overall user experience [43, 57, 85]. Consequently, the main problem in this chapter is to *minimise the antennas' range in order to provide a good coverage of region R and/or to ensure the existence of a well covered path within R.* What follows is therefore a formal definition of the problem. The distance between a point q on the plane and a set S of points is defined as the minimum distance between q and any one

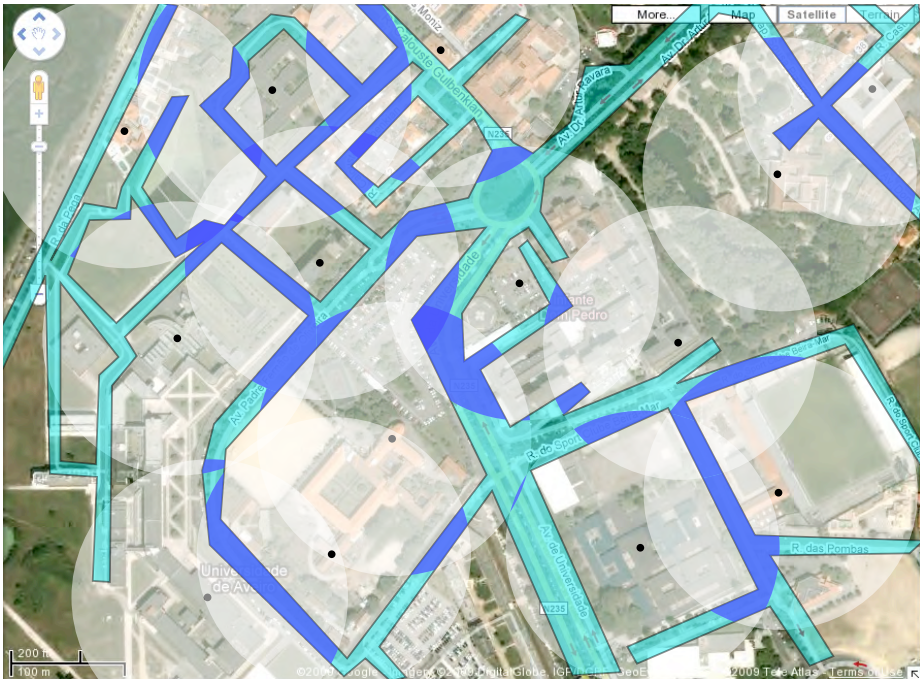
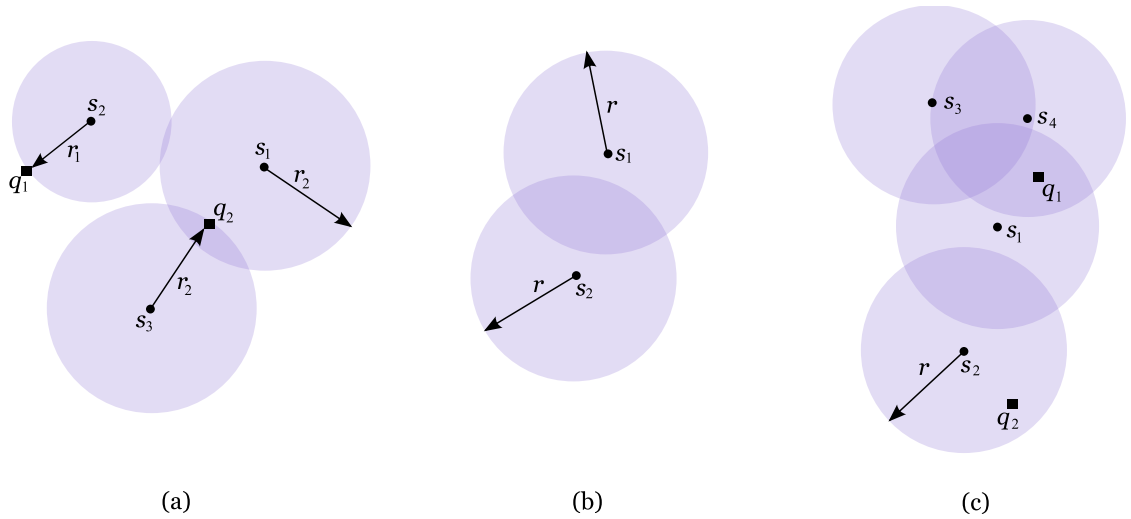


Figure 4.3: Regions covered by at least two antennas are shown in dark blue.

point of S. Point q is said to be *covered* by a set S of antennas with power transmission range r if the distance between q and S is less than or equal to r. Therefore, the antennas' minimum transmission range that covers q is the distance between q and S (see Figure 4.4(a)). A point covered by two or more antennas of S is said to be 2-*covered* by S (see Figure 4.4(a)). This definition aims towards reliable coverage, as previously explained. The minimum power transmission range of S that 2-covers an object x is denoted by $\mathrm{MR}_S(x)$. For example, if the object is a point q, $\mathrm{MR}_S(q)$ is the minimum power transmission range of S that 2-covers q.

Let $D = \{D(s_i, r) : s_i \in S\}$ be the set of discs of radius r each centred at an antenna of S. Each non-empty intersection between two discs of D is called a *lens* (see Figure 4.4(b)). The union of these lenses encloses all the points on the plane that are 2-covered by S, thus defining 2-*covered regions* (see Figure 4.4(c)). Every point within such regions clearly is 2-covered. Note that the minimum range of the antennas that covers a point can be easily calculated in $\mathcal{O}(n)$ time since such range is given by the distance between the point and its closest antenna. However, the Voronoi diagram of S [21] is a useful geometric structure that answers each query in $\mathcal{O}(\log n)$ time. This diagram can be constructed in $\mathcal{O}(n \log n)$ time, so its construction is only justified when the number of queries regarding S is larger than $\log n$. The two closest antennas to a given point can be found using the second order Voronoi diagram of S, denoted by $\mathrm{VD}_2(S)$. The second order Voronoi diagram of S divides the plane into several regions by grouping points that share the same two closest antennas [21] and has the same temporal



(a) (b) (c)

Figure 4.4: Set S is represented by dots. (a) Point q_1 is covered by antenna s_2 with minimum range $r_1 = d(s_2, q_1)$. Point q_2 is 2-covered by S with minimum range $r_2 = d(s_3, q_2)$. (b) The lens formed by $D(s_1, r)$ and $D(s_2, r)$ is shown in dark purple. (c) Regions 2-covered by S with range r are shown in dark purple. Point q_1 is 2-covered by S, whilst q_2 is not.

complexity as the ordinary Voronoi diagram [55]. This diagram is naturally associated with the optimisation problems studied in the present chapter since a point is 2-covered if it is within range of its two closest antennas. Combining Voronoi diagrams and coverage problems is not a novel approach and several other authors as Zhang et al. [82] or Das et al. [35] have made use of this geometric structure before.

Following the previous terminology, there are several works associated with geometric optimisation using 1-coverage. For example, Abellanas et al. [18] and Mehta et al. [65] study routes on the plane that are always close to (or always far from) a given set of antennas. Agnetis et al. [19] studied a problem known as the *disc covering problem on a line*. They provided an exact solution and a heuristic to compute a subset of a given set of discs in order to keep a given line segment covered at minimum cost. They studied discs with variable radii with two different approaches: first the costs of the discs depended on their radii and second, the costs of the discs were fixed. Meguerdichian et al. [64] were the first to combine ad-hoc sensor networks and computational geometry. Several others, including Boukerche et al. [26], Li et al. [57] and Mehta et al. [65], built on their ideas either by studying different versions of the coverage problem or by improving and proving previous results. Recently, Zhang et al. [82] focused on the decision problem and developed two localised algorithms to identify whether a sensor is on the boundary of the monitored area. To this end, they based their algorithms on two novel geometric tools: localised Voronoi diagram and neighbour embracing polygons. Using variable radii sensors, Zhou et al. [85] proposed several centralised

and distributed algorithms to compute a minimum energy-cost coverage, where each sensor can vary its sensing and transmission radius. Das et al. [35] studied efficient location of base stations to cover a convex region when the base stations are inside the region. Making use of Voronoi diagrams, they developed a fast iterative algorithm to solve this problem. Consult their paper for additional bibliography on the coverage of a square or an equilateral triangle. Also employing Voronoi diagrams, Stoyan and Patsuk [75] considered the problem of covering a compact polygonal set using identical discs of minimum radius. Boukerche et al. [26] addressed sensor networks by studying how well a large wireless sensor network can be monitored or tracked while keeping a long lifetime. Their solution outperformed previously known techniques. Bezdek and Kuperberg [24] researched an intermediate problem between 1-coverage and 2-coverage. They studied a minimal coverage of the plane using discs of the same radius so that the region remains covered even when the radius of one of the discs decreases. This concept provides a continuous transition from 1-coverage to 2-coverage.

The various optimisation problems studied in this chapter aim to minimise the power transmission range of a set S of antennas to 2-cover a given region R and/or to ensure the existence of a 2-covered path on that region. Distinct versions of this problem arise for different types of regions, so each of the following sections studies the relevant version of the problem for a particular region. In Section 4.2 it is shown how to calculate the minimum range of S such that the line segment connecting points p and q on the plane is 2-covered. Such algorithm will then be used in Section 4.3 to calculate the antennas' minimum range in order to obtain a 2-covered path between two nodes of a graph. In the process, it is also explained how to minimise the antennas' range to 2-cover the whole graph. Assuming R is the whole plane, the minimum range that allows a 2-covered path between two points on the plane is calculated in Section 4.4. Furthermore, it is also shown how to compute the shortest 2-covered path between those points. The minimum transmission range of S to 2-cover a polygonal region R (with or without holes) is calculated in Section 4.5. Section 4.6 calculates the antennas' minimum range so that a 2-covered path exists within a polygonal region R. Although comparable, this problem is more complex than the one presented in Section 4.4. Section 4.7 addresses the minimisation of the antennas' range to ensure the existence of a 2-covered path connecting a set of points. Section 4.8 deals with a similar problem: minimise the range of S to allow a 2-covered path connecting two line segments. Finally, this chapter concludes with a discussion of the results and future research in Section 4.9.

4.2 Minimum Transmission Range to 2-Cover a Line Segment

Given a set S of n antennas and two points p and q on the plane, the following algorithm calculates the minimum power transmission range of S so that the line segment connecting both points, $\overline{pq}$, is 2-covered. Such range is denoted by $\mathrm{MR}_S(\overline{pq})$ and is calculated as being the largest distance between a point of $\overline{pq}$ and its second closest antenna. To find this point, assume that the antennas' range is sufficiently large to fully 2-cover $\overline{pq}$. Now suppose that this range is slowly reduced until there is one point i on $\overline{pq}$ that is no longer 2-covered. As a result, $\overline{pq}$ is not 2-covered until point i is, which means that the antennas' minimum range to 2-cover i is precisely the minimum range to fully 2-cover $\overline{pq}$. This point i can be either an interior point of $\overline{pq}$ (see Figure 4.5(a)) or an outer point (see Figure 4.5(b)). Given the point's definition, it becomes apparent that if i is not p or q then it has to be an intersection point between $\overline{pq}$ and the second order Voronoi diagram of S, $\mathrm{VD}_2(S)$. The following pseudo-code calculates the minimum power transmission range of S to 2-cover $\overline{pq}$.

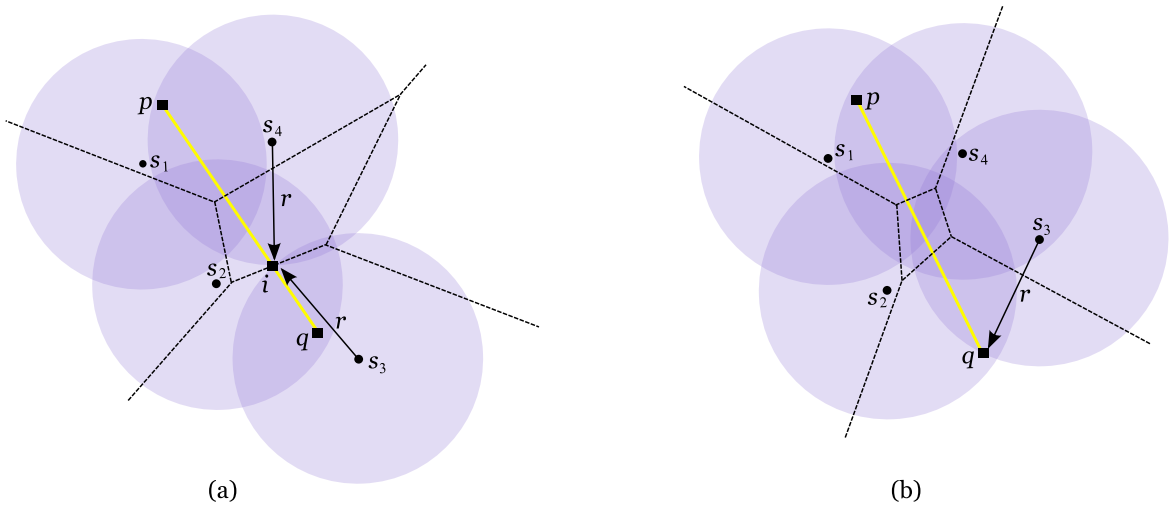


Figure 4.5: Line segment $\overline{pq}$ is 2-covered by S with minimum range r, $\mathrm{VD}_2(S)$ is represented by a dashed line. (a) $\mathrm{MR}_S(\overline{pq}) = d(i, s_3) = d(i, s_4) = r$. (b) $\mathrm{MR}_S(\overline{pq}) = d(q, s_3) = r$.

ALGORITHM **Minimum 2-Coverage of a Line Segment**

INPUT: Set S of antennas, points p and q on the plane
OUTPUT: $\mathrm{MR}_S(\overline{pq})$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
2. $I \leftarrow (\{\overline{pq}\} \cap \mathrm{VD}_2(S)) \cup \{p, q\}$;
3. $\mathrm{MR}_S(\overline{pq}) \leftarrow \max\{\mathrm{MR}_S(i) : i \in I\}$.

The following theorem states the complexity of the algorithm proposed above.

Theorem 4.1 *Given a set S of n antennas and two points p and q on the plane, $\mathrm{MR}_S(\overline{pq})$ can be calculated in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is a set of n antennas on the plane [55]. The cardinality of the set $I = \{\overline{pq}\} \cap \mathrm{VD}_2(S)$ is at most n since $\mathrm{VD}_2(S)$ has a linear number of regions. Inserting points p and q to set I can be done in constant time. For each intersection point $i \in I$, $\mathrm{MR}_S(i)$ is calculated in constant time using $\mathrm{VD}_2(S)$ since it is the distance between i and its second closest antenna (recall that i had been previously located in $\mathrm{VD}_2(S)$). The largest of these distances is $\mathrm{MR}_S(\overline{pq})$. Consequently, the temporal complexity of this procedure is $\mathcal{O}(n \log n)$. Regarding the space complexity, $\mathrm{VD}_2(S)$ is the largest data structure of this algorithm and it can be stored in $\mathcal{O}(n)$ space [55]. □

4.3 Minimum 2-Covered Path on a Planar Graph

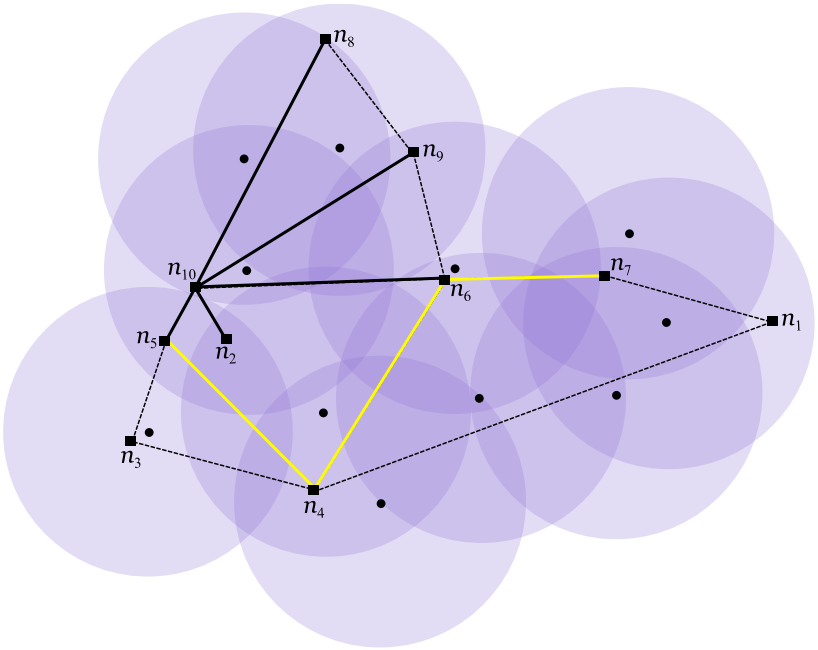


Figure 4.6: The set of antennas is represented by dots and the nodes of G by squares. The 2-covered edges of G are shown in a solid line. The yellow path from n_5 to n_7 is a 2-covered path on G.

Let $G = (N, E)$ be a connected geometric planar graph whose nodes represent locations and its edges model roads or streets that connect such locations. As before, let S be a set of

n antennas on the plane and assume that $|E| > \log n$, without loss of generality. A path on G connecting two nodes using only the edges of G that are 2-covered by S is called a 2-covered path or a 2-path (see Figure 4.6). Given two nodes n_i and n_j of G, let $P(n_i, n_j)$ denote a path between those nodes. The minimum range of S that ensures the existence of a 2-path between n_i and n_j on G is denoted by $\mathrm{MR}_{S,G}(P(n_i, n_j))$. To simplify this notation, such range can also be denoted by $\mathrm{MR}_S(P(n_i, n_j))$ if G is clear from the context. The following sections propose an algorithm to calculate $\mathrm{MR}_S(P(n_i, n_j))$ and a 2-covered path on G between nodes n_i and n_j. Said algorithm is divided into two phases: preprocess and solution.

4.3.1 First Phase: Preprocess

The procedure explained in this first part acts as a preprocess that transforms graph G into edge-weighted graph G_w. To make this transformation, each edge e of G is assigned the weight $\mathrm{MR}_S(e)$, which is calculated using the algorithm described in Section 4.2. For example, in Figure 4.7(a) the weight of the edge $\overline{n_1 n_7}$ is given by the distance $d(s_8, n_1) = d(s_{10}, n_1) = 38$, and consequently $\mathrm{MR}_S(\overline{n_1 n_7}) = 38$. In the same example, the graph is transformed into an edge-weighted graph in Figure 4.7(b).

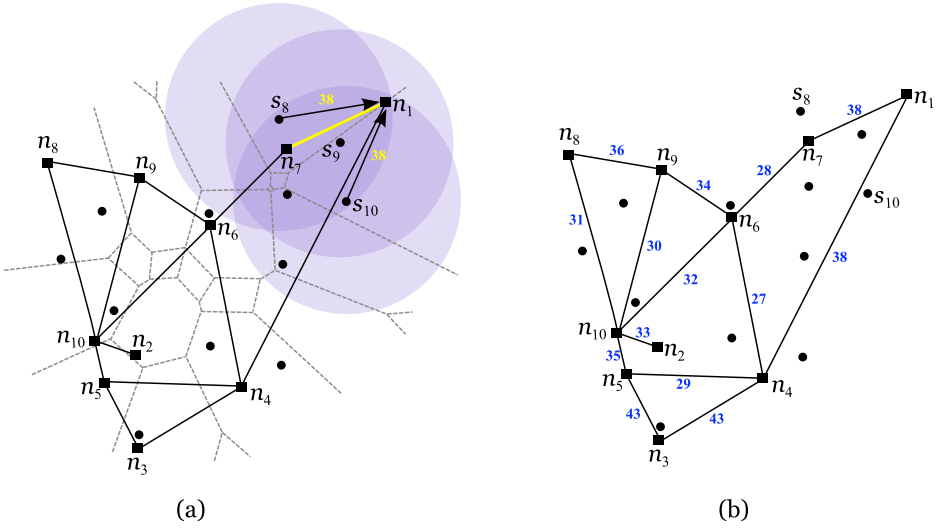


Figure 4.7: (a) $\mathrm{VD}_2(S)$ is shown in a dashed line. The weight of edge $\overline{n_1 n_7}$ is given by $d(s_8, n_1) = 38$. (b) Edge-weighted graph G_w.

Note that the minimum range of S to fully 2-cover graph G is given by the largest weight of the graph's edges, that is, $\mathrm{MR}_S(G) = \max\{\mathrm{MR}_S(e) : e \in G\}$. These weights can be easily calculated using the following proposition, which is a consequence of Theorem 4.1.

Proposition 4.1 *Let S be a set of n antennas and $G = (N, E)$ a connected geometric planar*

graph with m edges. The weight of every edge of G and $\mathrm{MR}_S(G)$ can be calculated in $\mathcal{O}(mn)$ time, $m > \log n$.

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is a set of n antennas on the plane [55]. Set $I_e = \{e\} \cap \mathrm{VD}_2(S)$ united with the endpoints of e is found in $\mathcal{O}(n)$ time for each edge $e \in E$. Calculating $\mathrm{MR}_S(e) = \max\{\mathrm{MR}_S(i), i \in I_e\}$ is linear on n for each edge e. Therefore, finding $\mathrm{MR}_S(e)$ for every edge $e \in G$ takes $\mathcal{O}(mn)$ time since G has m edges. Finally, as it was assumed that $m > \log n$, $\mathrm{MR}_S(e)$ for every edge of G and consequently $\mathrm{MR}_S(G)$ can be calculated in $\mathcal{O}(mn)$ time. □

4.3.2 Second Phase: Solution

Similarly to above, let n_i and n_j be two nodes of G_w. It is apparent that the minimum power transmission range that ensures the existence of $P(n_i, n_j)$ is given by the weight of its heaviest edge. The following proposition shows that it is only necessary to consider the edges of a minimum spanning tree (MST) of G_w to find a 2-path between two nodes of G_w.

Proposition 4.2 *Let G_w be an edge-weighted connected graph. For each path on G_w, assume the path's weight is given by the weight of its heaviest edge. Then the path on an* MST *of G_w connecting any pair of nodes of G_w is a minimum weight path between such pair.*

Proof: Let G_w be an edge-weighted connected graph, n_i and n_j two of its nodes and T_w an MST of G_w. Suppose that $P(n_i, n_j)$ is the only path on T_w connecting nodes n_i and n_j and e is its heaviest edge. Consequently, $P(n_i, n_j)$ has weight $w(e)$. Now suppose that path $P^*(n_i, n_j)$ on G_w is a minimum weight path connecting n_i and n_j. Its weight is given by e^*, its heaviest edge, and so $w(e^*) < w(e)$. Since $P(n_i, n_j)$ is heavier than $P^*(n_i, n_j)$, the edge e is not an edge of $P^*(n_i, n_j)$. If the two paths are united, then a cycle is created. Such cycle contains e, which clearly is its heaviest edge. But this contradicts the hypothesis, since the heaviest edge of a cycle in G_w cannot be an edge of an MST of G_w. Therefore, a minimum weight path between two nodes of G_w is the path on T_w connecting those nodes. □

Since G_w is a planar graph, there is an algorithm by Matsui [60] that finds an MST of G_w in time proportional to the number of edges of G_w. According to the proposition above, a 2-path between two nodes on a graph with minimum power transmission range can be computed using an MST of such graph (see Figure 4.8(a)). As a result, it is easier to work with an MST because paths between any two nodes of such tree are unique. What follows is therefore the core of the algorithm to find a 2-path connecting two nodes of G_w with minimum transmission

range, taking advantage of an MST of G_w and of the Depth-First Search (DFS) algorithm [42].

ALGORITHM **Minimum 2-Covered Path**

INPUT: Graph G_w, nodes n_i and n_j
OUTPUT: Path $P(n_i, n_j)$ and $\mathrm{MR}_{S,G_w}(P(n_i, n_j))$

1. Find T_w, an MST of G_w;
2. Compute $P(n_i, n_j)$ on T_w using the DFS algorithm [42];
3. $\mathrm{MR}_{S,G_w}(P(n_i, n_j)) \leftarrow \max\{w(e) : e \in P(n_i, n_j)\}$.

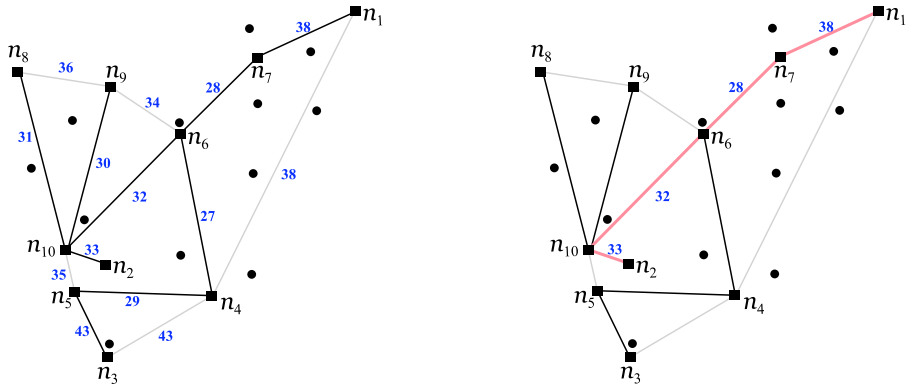


Figure 4.8: (a) An MST of the edge-weighted graph is shown in a dark line. (b) The pink path connecting n_1 to n_2 on the tree only exists if the antennas' transmission range is at least $\max\{33, 32, 28, 38\} = 38$.

In Figure 4.8(b) there is an example of a path on an MST connecting nodes n_1 and n_2, $P(n_1, n_2)$. Note that this path also traverses nodes n_{10}, n_6 and n_7. If the antennas' transmission range is larger than $38 = \max\{33, 32, 28, 38\}$, then $P(n_1, n_2)$ is a 2-covered path. If this range is exactly 38 then $P(n_1, n_2)$ is 2-covered with minimum transmission range, that is, a 2-path between n_1 and n_2 exists and the antennas' range is minimised. Therefore, $\mathrm{MR}_S(P(n_1, n_2)) = 38$. The following result states the temporal complexity of the previous algorithm.

Theorem 4.2 *Let S be a set of n antennas, G_w an edge-weighted planar graph with m edges and T_w an* MST *of G_w. Given two nodes n_i and n_j of G_w, $P(n_i, n_j)$ and $\mathrm{MR}_{S,G_w}(P(n_i, n_j))$ can be found on T_w in $\mathcal{O}(m)$ time.*

Proof: As previously mentioned, the algorithm by Matsui [60] can be used to find an MST of G_w, $T_w = (N, B)$, in $\mathcal{O}(m)$ time since G_w is a planar graph. Given two nodes n_i and n_j of T_w, path $P(n_i, n_j)$ on T_w can be found using the Depth-First Search algorithm [42], which runs in $\mathcal{O}(|B|)$ time. According to Proposition 4.2, $P(n_i, n_j)$ is a minimum weight path between n_i and n_j on G_w. Furthermore, it is also a 2-covered path connecting nodes n_i and n_j. Since $\mathrm{MR}_{S,G_w}(P(n_i, n_j))$ is the weight of $P(n_i, n_j)$, it can be calculated in time proportional to the number of edges of $P(n_i, n_j)$. Overall, this procedure takes $\mathcal{O}(m)$ time. □

4.4 Minimum 2-Covered Path on the Plane

In the previous sections, region R was seen as a line segment or a connected graph. In this section, R is considered to be the whole plane, that is, $R = \mathbb{R}^2$. Given a set S of n antennas and two points p and q on the plane, the objective of the following discussion is to calculate the minimum range of the antennas such that there is a 2-covered path connecting p and q. In the example in Figure 4.9(a), the black path connecting p and q is not a 2-path since some of its points are only covered by one antenna. On the other hand, the yellow path represents a 2-covered path between p and q.

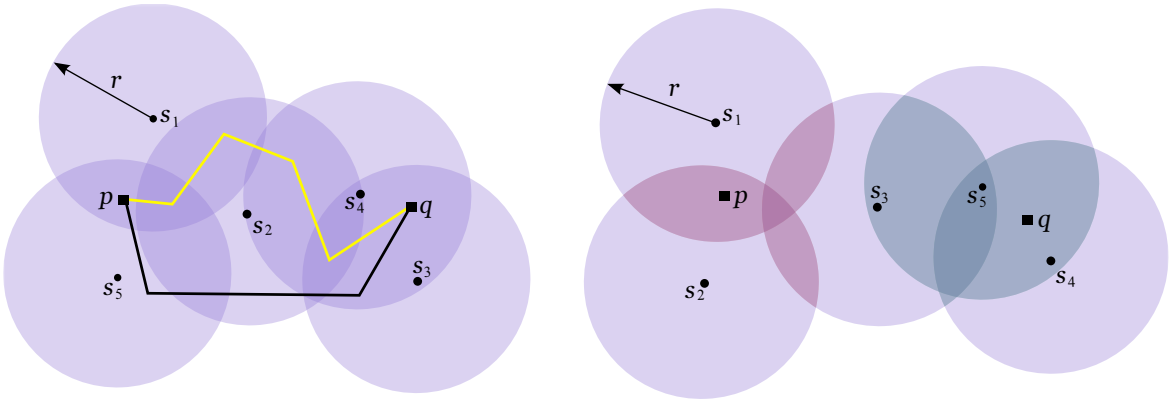


Figure 4.9: A set of five antennas with range r. (a) The yellow path is a 2-path connecting p to q, whilst the black is not. (b) One connected component of the union of lenses is shown in red and the other in green. It is not possible to find a 2-covered path between p and q.

The following subsection proposes a solution to the following decision problem: given a set S of n antennas with power transmission range $r \in \mathbb{R}^+$ and two points p and q on the plane, decide if there is a 2-covered path connecting p to q. The solution to this problem is the basis of the algorithm presented in Section 4.4.2, which solves the main problem. In the latter, it is also explained how to obtain a 2-covered polygonal path between p and q employing the

method previously used to find the antennas' minimum range. The shortest 2-covered path between two points on the plane is computed in the Section 4.4.3.

4.4.1 Decision Problem: Is r large enough?

In the following, it is shown how to decide if a path between two points p and q on the plane, $P(p, q)$, is 2-covered by S. Note that a 2-path from p to q exists if and only if p and q lie in the same connected component of the union of lenses (see Figure 4.9(b)). Therefore, the overlapping of lenses is the key to find a 2-path between p and q. Consider the intersection graph of the set of lenses, that is, the graph where each node represents a lens and two nodes are connected if the respective lenses intersect each other. This graph has $\mathcal{O}(n^4)$ edges and this upper bound is tight because, if r is large enough, every pair of overlapping discs of radius r centred at the antennas defines a lens and so the intersection graph is a complete graph. Regarding this graph, assign point p to node n_p that corresponds to the lens containing p. In a similar way, assign point q to node n_q. Then the existence of a path between p and q depends on whether n_p and n_q lie in the same connected component of this graph.

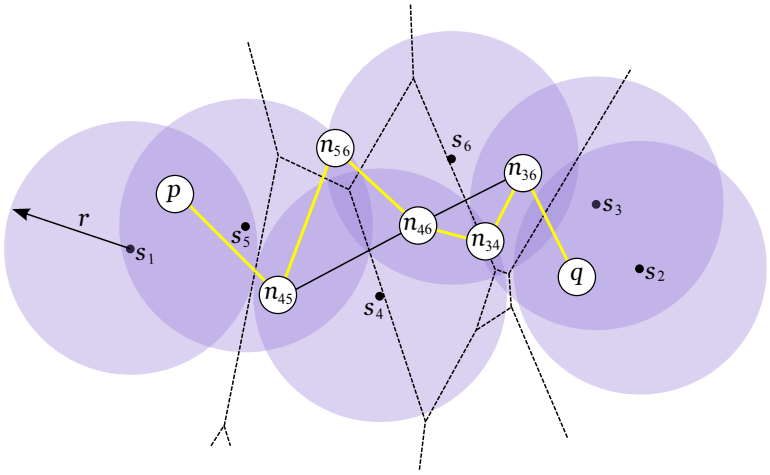


Figure 4.10: Set S is represented by dots, graph G by a solid line and $\mathrm{VD}_2(S)$ by a dashed line. The yellow edges of G form a path between p and q.

However, given the high complexity of the intersection graph described above, the subgraph $G = (N, E)$ will be used instead. Graph G plays the same role as the intersection graph, but has linear size and can be constructed in $\mathcal{O}(n \log n)$ time by means of the second order Voronoi diagram of S. Let $\mathrm{VR}_2(s_i, s_j)$ denote the second order Voronoi region of the antennas s_i and s_j. For each second order Voronoi region $\mathrm{VR}_2(s_i, s_j)$, such that the discs of radius r centred at s_i and s_j define a lens, add a node to graph G (see Figure 4.10). Two nodes of G are connected by an edge if they correspond to two Voronoi regions that are neighbours

in $\mathrm{VD}_2(S)$ and the corresponding lenses have a non-empty intersection. Let $l_r(s_i, s_j)$ denote the lens resulting from the intersection of discs $D(s_i, r)$ and $D(s_j, r)$. The general strategy to solve the previously described decision problem is presented in the following pseudo-code.

ALGORITHM **Power Transmission Range Test (TRT)**

INPUT: Set S of antennas with transmission range r, points p and q
OUTPUT: YES or NO

1. If p and q are not within the lenses defined by their two closest antennas then return NO;
2. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
3. Construct graph G:

   For each $\mathrm{VR}_2(s_i, s_j)$ do

   (a) If $l_r(s_i, s_j) \neq \emptyset$ then add node n_{ij} to G

   (b) For each $\mathrm{VR}_2(s_l, s_k)$ neighbour of $\mathrm{VR}_2(s_i, s_j)$ do
   If $l_r(s_i, s_j) \cap l_r(s_l, s_k) \neq \emptyset$ then add edge $\overline{n_{ij}n_{lk}}$ to G
4. Find nodes n_p and n_q of G that correspond to the lenses containing p and q, respectively;
5. Traverse G starting at n_p using the DFS algorithm [42];
6. If node n_q is found then return YES, otherwise return NO.

In Figure 4.10 there is an example of graph G and a path between p and q. For that set of antennas with transmission range r, a 2-covered path connecting p to q exists since both points lie in the same connected component of the union of lenses. The following result proves the temporal and space complexities of this decision algorithm.

Theorem 4.3 *Let S be a set of n antennas with power transmission range $r \in \mathbb{R}^+$ and p and q two points on the plane. Deciding if there is a 2-covered path between p and q takes $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is formed by n antennas on the plane [55]. Graph G is a subgraph of the dual graph of $\mathrm{VD}_2(S)$ and therefore has linear size [21]. The construction of G takes $\mathcal{O}(n)$ time because the algorithm performs a constant

number of operations per edge of the second order Voronoi diagram. In the worst case, the Depth-First Search (DFS) algorithm has to visit every node of G twice, and so its complexity depends on the number of edges and nodes of G. Since G has linear size, the DFS algorithm runs in linear time as well [42]. Consequently, the decision algorithm runs in $\mathcal{O}(n \log n)$ time. Regarding the space complexity, G has a linear size and $\mathrm{VD}_2(S)$ can be stored in $\mathcal{O}(n)$ space [55], which means the algorithm takes $\mathcal{O}(n)$ space. □

This decision algorithm is a key method to solve several problems associated with 2-coverage. As a result, it will be often applied throughout this chapter.

4.4.2 Minimum 2-Covered Path on the Plane

The following algorithm calculates the minimum power transmission range of S that ensures the existence of a 2-path between points p and q on the plane. Such range is denoted by $\mathrm{MR}_S(P(p,q))$. Note that for $|S| \geq 2$, this problem always has a solution because, if the range r is large enough, then all discs of radius r centred at the antennas contain points p and q. In this case, the line segment connecting p to q is a 2-covered path. Both points have to be 2-covered in order for a 2-path to exist between them. Consequently, if s_p is the second closest antenna of S to p and s_q the second closest antenna of S to q, then $\mathrm{MR}_S(P(p,q)) \geq \max\{d(s_p,p), d(s_q,q)\}$. Observe that if p or q is an antenna of S, then it is considered that they cover themselves. Furthermore, if a 2-path exists between p and q when the antennas' range is $r = \max\{d(p,s_p), d(q,s_q)\}$, then $\mathrm{MR}_S(P(p,q)) = r$.

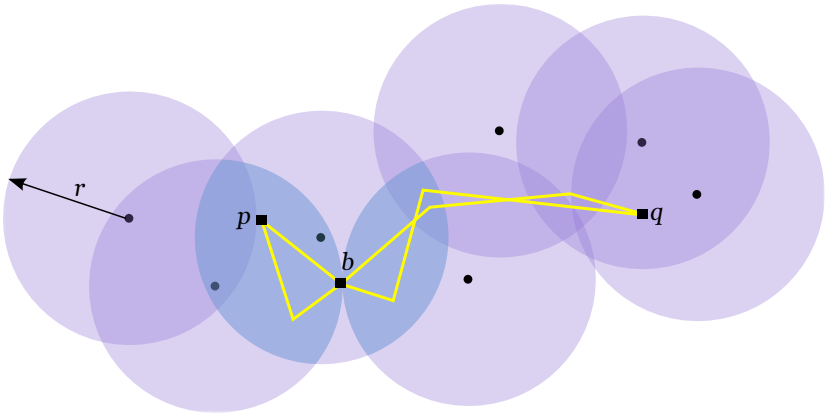


Figure 4.11: The antennas' range is $r = \mathrm{MR}_S(P(p,q))$. The connected component of lenses containing p meets the component containing q at point b. Two possible 2-paths connecting p to q are represented by a solid yellow line.

Consider the case $\mathrm{MR}_S(P(p,q)) > \max\{d(s_p,p), d(s_q,q)\}$. If the power transmission range of S is $r = \max\{d(s_p,p), d(s_q,q)\}$, then p and q lie in different connected components

of the union of lenses. As the range r increases, the discs of radius r centred at the antennas grow larger. Eventually, the range reaches the value of $\mathrm{MR}_S(P(p,q))$ and the components that contain p and q unite (see Figure 4.11). The intersection points between the components of lenses at this specific range are called *bottleneck-points* for 2-paths between p and q. Note that in degenerated cases there might be more than one bottleneck-point for 2-paths between p and q. For example, four collinear antennas placed at the same distance generate up to two bottleneck-points (see Figure 4.12). This subject will be further discussed in this section. For now, and without loss of generality, degenerate input data is not considered. Let $b_S(p,q)$ denote the bottleneck-point for 2-paths between p and q. If the antennas' range is $\mathrm{MR}_S(P(p,q))$ then every 2-path connecting p and q crosses $b_S(p,q)$. This holds for this range because $b_S(p,q)$ is the only point connecting the component of lenses that contains p to the one containing q. Therefore, every 2-path connecting these points has to cross both components and therefore, cross $b_S(p,q)$. Bottleneck-points may be classified in two categories according to the following definition.

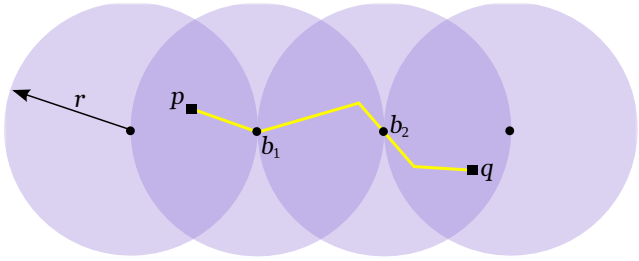


Figure 4.12: A 2-path connecting p to q is represented by a solid yellow line. Points b_1 and b_2 are two bottleneck-points for 2-covered paths between p and q.

Definition 4.1 *Let p and q be two points on the plane and S a set of antennas with range $r = \mathrm{MR}_S(P(p,q))$.*

(a) *A point $b = b_S(p,q)$ is a type I bottleneck-point if there is only one antenna $s_i \in S$ such that $b \in \mathrm{int}(D(s_i,r))$ and there are two other antennas $s_j, s_k \in S$ such that $D(s_j,r) \cap D(s_k,r) = \{b\}$.*

(b) *A point $b = b_S(p,q)$ is a type II bottleneck-point if there are exactly three antennas $s_i, s_j, s_k \in S$ such that $D(s_i,r) \cap D(s_j,r) \cap D(s_k,r) = \{b\}$ and there is no antenna $s_l \in S$ such that $b \in \mathrm{int}(D(s_l,r))$.*

An example of both types of bottleneck-points can be seen in Figure 4.13. According to Definition 4.1, it becomes apparent that a type I bottleneck-point for 2-paths between p and q is the midpoint of the segment $\overline{s_j s_k}$ (see Figure 4.13(a)).

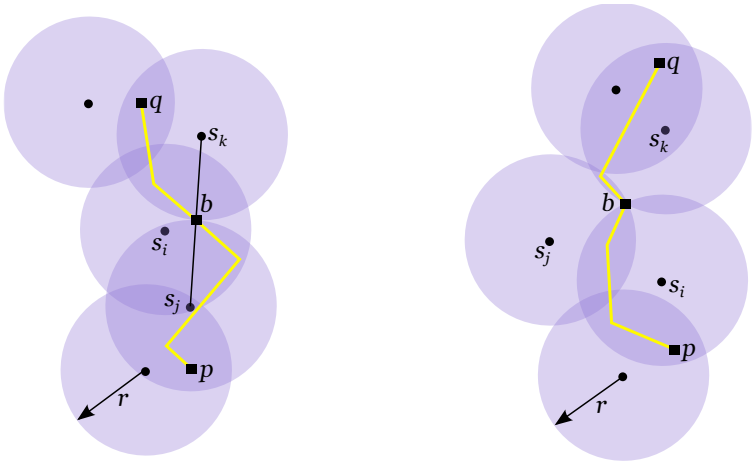


Figure 4.13: The antennas' range is $r = \mathrm{MR}_S(P(p,q))$ and a 2-path connecting p to q is represented by a solid yellow line. (a) Point $b = b_S(p,q)$ is a type I bottleneck-point. (b) Point $b = b_S(p,q)$ is a type II bottleneck-point.

Proposition 4.3 *Let p and q be two points on the plane and S a set of antennas with range $r = \mathrm{MR}_S(P(p,q))$. Point $b_S(p,q)$ is either type I or II.*

Proof: As the antennas' range increases, the two connected components of the union of lenses (one containing p and the other containing q) eventually unite at a bottleneck-point. If this union is made through the intersection of only two lenses, then the intersection point is a type I bottleneck-point. Otherwise, the union is made through the intersection of three lenses at a time and that intersection point is a type II bottleneck-point. Recall that no degenerate cases are considered. □

Let $\triangle(s_i, s_j, s_k)$ be the triangle formed by the antennas s_i, s_j and s_k. The following proposition establishes a connection between bottleneck-points and the second order Voronoi diagram of S.

Proposition 4.4 *Let p and q be two points on the plane and S a set of antennas with range $r = \mathrm{MR}_S(P(p,q))$.*

(a) *If $b = b_S(p,q)$ is a type I bottleneck-point covered by $s_i, s_j, s_k \in S$ and such that $b \in \mathrm{int}(D(s_i, r))$, then b is the intersection point between $\overline{s_j s_k}$ and the second order Voronoi edge that separates $\mathrm{VR}_2(s_i, s_j)$ from $\mathrm{VR}_2(s_i, s_k)$;*

(b) *If $b = b_S(p,q)$ is a type II bottleneck-point covered by $s_i, s_j, s_k \in S$, then b is a vertex of $\mathrm{VD}_2(S)$ and $b \in \mathrm{int}(\triangle(s_i, s_j, s_k))$.*

Proof:

(a) According to the definition of type I bottleneck-points, b is a point on $\mathrm{PB}(\overline{s_j s_k})$, the perpendicular bisector between s_j and s_k (see Figure 4.14). Moreover, b is the intersection point between $\mathrm{PB}(\overline{s_j s_k})$ and $\overline{s_j s_k}$ since the midpoint of $\overline{s_j s_k}$ is b. As b is a bottleneck-point and $d(b, s_i) < r$, $\mathrm{PB}(\overline{s_j s_k})$ separates $\mathrm{VR}_2(s_i, s_j)$ from $\mathrm{VR}_2(s_i, s_k)$. If there were other antennas interfering with these two second order Voronoi regions, then $b \neq b_S(p, q)$ because it would be 2-covered with a range smaller than r. Consequently b is the intersection point between $\overline{s_j s_k}$ and the second order Voronoi edge that separates $\mathrm{VR}_2(s_i, s_j)$ from $\mathrm{VR}_2(s_i, s_k)$.

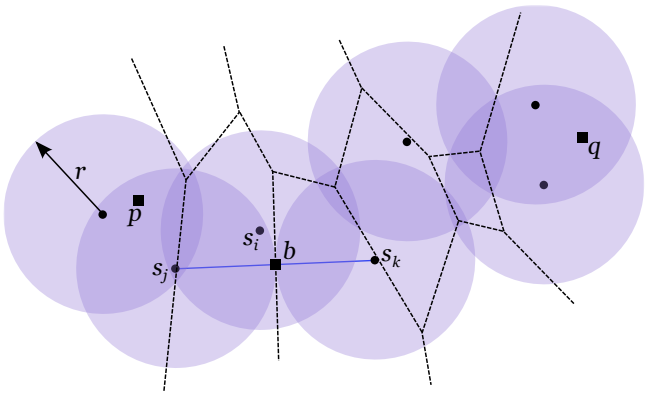


Figure 4.14: The antennas' range is $r = \mathrm{MR}_S(P(p, q))$ and $\mathrm{VD}_2(S)$ is shown in a dashed line. Point $b = b_S(p, q)$ is a type I bottleneck-point and a point of $\mathrm{VD}_2(S)$: $\{b\} = \mathrm{PB}(\overline{s_j s_k}) \cap \{\overline{s_j s_k}\}$.

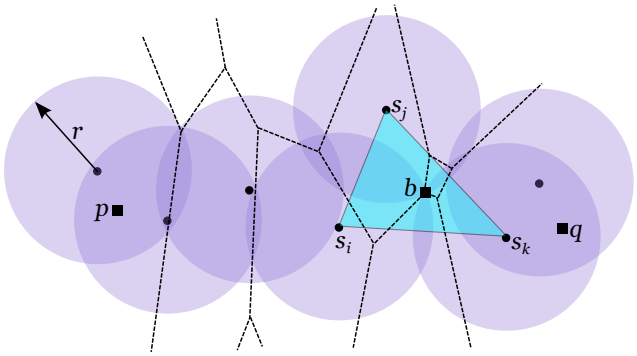


Figure 4.15: The antennas' range is $r = \mathrm{MR}_S(P(p, q))$ and $\mathrm{VD}_2(S)$ is represented by a dashed line. Point $b = b_S(p, q)$ is a type II bottleneck-point and a vertex of $\mathrm{VD}_2(S)$: $\{b\} = \mathrm{PB}(\overline{s_i s_j}) \cap \mathrm{PB}(\overline{s_j s_k}) \cap \mathrm{PB}(\overline{s_k s_i})$. Moreover, $b \in \mathrm{int}(\triangle(s_i, s_j, s_k))$.

(b) According to the definition of type II bottleneck-points, b is equally distant from s_i, s_j and s_k (see Figure 4.15). This means that point b is the intersection point between the

perpendicular bisectors PB($\overline{s_i s_j}$), PB($\overline{s_i s_k}$) and PB($\overline{s_j s_k}$). Moreover, these perpendicular bisectors contain edges of $\mathrm{VD}_2(S)$ since there are no other antennas of S interfering with the surroundings of b (otherwise $b \neq b_S(p,q)$). Perpendicular bisector PB($\overline{s_i s_j}$) separates $\mathrm{VR}_2(s_i, s_k)$ from $\mathrm{VR}_2(s_j, s_k)$, PB($\overline{s_i s_k}$) separates $\mathrm{VR}_2(s_i, s_j)$ from $\mathrm{VR}_2(s_j, s_k)$ and PB($\overline{s_j s_k}$) separates $\mathrm{VR}_2(s_i, s_j)$ from $\mathrm{VR}_2(s_i, s_k)$. Thus, b is a vertex of $\mathrm{VD}_2(S)$. Furthermore, since the discs $D(s_i, r)$, $D(s_j, r)$ and $D(s_k, r)$ intersect only once (precisely at b), b is in the interior of $\triangle(s_i, s_j, s_k)$. □

Given the direct relation between the second order Voronoi diagram and bottleneck-points, a type I bottleneck-point is also referred to as *edge bottleneck-point* and a type II bottleneck-point as *vertex bottleneck-point*. As previously mentioned, a point x in $\mathrm{VR}_2(s_i, s_j)$ is 2-covered by S with range r if and only if $x \in l_r(s_i, s_j)$. With this in mind, as well as Proposition 4.4, the search for bottleneck-points must be focused on edges and vertices of $\mathrm{VD}_2(S)$. For that purpose, first search for edges intersected by the line segment joining the two antennas defining such edge. The resulting intersection point is a candidate for edge bottleneck-point (see Figure 4.14). Second, search every vertex of $\mathrm{VD}_2(S)$ that is inside the triangle of the antennas responsible for it since it is a candidate for vertex bottleneck-point (see Figure 4.15). For every such candidate calculate the minimum range needed to 2-cover it. Then sort these ranges into a list. Third, determine s_p and s_q, the second closest antenna of S to p and q respectively, to calculate $r_0 = \max\{d(s_p, p), d(s_q, q)\}$. Range r_0 is the starting point for the list of ranges. All ranges below r_0 cannot be $\mathrm{MR}_S(P(p,q))$. Note that if r_0 is the largest element of said list then $\mathrm{MR}_S(P(p,q)) = r_0$. Otherwise perform a binary search on the list of ranges to calculate $\mathrm{MR}_S(P(p,q))$. In every step of the binary search, decide if the corresponding range is large enough to allow a 2-covered path connecting p to q. To this end, use the TRT decision algorithm described in Section 4.4.1.

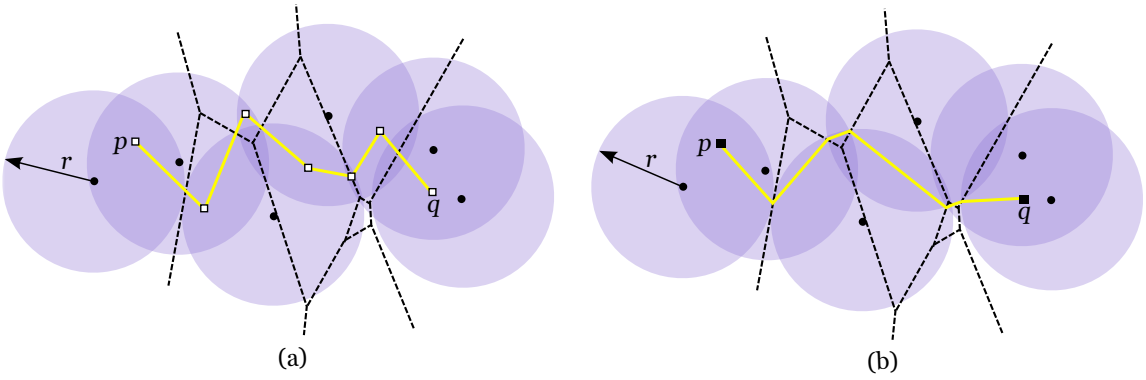


Figure 4.16: The antennas' range is $r = \mathrm{MR}_S(P(p,q))$ and $\mathrm{VD}_2(S)$ is represented by a dashed line. (a) Path between nodes p and q on graph G. (b) A 2-path between points p and q.

Once the binary search is over, the final range is $\mathrm{MR}_S(P(p,q))$. Consider graph G constructed by the TRT algorithm for the range $\mathrm{MR}_S(P(p,q))$ (see Figure 4.16(a)). A polygonal 2-path between p and q can be computed using a path on graph G that connects nodes p and q. For every edge of the path from node p to q on G compute a point on its dual Voronoi edge such that it lies in the intersection of the two lenses associated with this edge. As lenses are convex sets, the straight line segment connecting two consecutive such points is entirely contained in one lens. If needed, this path hast to be completed by adding the line segments that connect p to the first node and q to the last node of the 2-path. This ensures that these line segments form a 2-covered path from p to q (see Figure 4.16(b)).

ALGORITHM **Minimum 2-Covered Path on the Plane**

INPUT: Set S of antennas, points p and q
OUTPUT: $\mathrm{MR}_S(P(p,q))$, $P(p,q)$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
2. $E \leftarrow \{i : \overline{s_i s_j} \cap \mathrm{PB}(s_i, s_j) = \{i\}, \forall s_i, s_j \in S\}$;
3. $V \leftarrow \{i : \mathrm{PB}(s_i, s_j) \cap \mathrm{PB}(s_i, s_k) \cap \mathrm{PB}(s_j, s_k) = \{i\}$
   and $i \in \triangle(s_i, s_j, s_k), \forall s_i, s_j, s_k \in S\}$;
4. $I' \leftarrow \{\mathrm{MR}_S(i) : \forall i \in \{E \cup V\}\}$;
5. Calculate $r_0 = \max\{\mathrm{MR}_S(p), \mathrm{MR}_S(q)\}$;
6. Perform a binary search on $I' = \{r_i \in I' : r_i \geq r_0\} \cup \{r_0\}$:
   For the median range $r_i \in I'$ do
   If $\mathrm{TRT}(S, r_i, p, q) = \mathrm{YES}$
   Then proceed the search on $I' \leftarrow \{r_j \in I' : r_j \leq r_i\}$
   Otherwise proceed the search on $I' \leftarrow \{r_j \in I' : r_j > r_i\}$
7. The final range is $\mathrm{MR}_S(P(p,q))$, compute $P(p,q)$ using the graph constructed by the algorithm TRT.

The description of this optimisation algorithm concludes with the following theorem.

Theorem 4.4 *Let S be a set of n antennas and p and q two points on the plane. Calculating $\mathrm{MR}_S(P(p,q))$ and a 2-covered path between p and q can be determined in $\Theta(n \log n)$ time.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is a set of n antennas on the plane

[55]. Since $\mathrm{VD}_2(S)$ has a linear number of vertices and edges [21], searching for possible bottleneck-points is also linear. Locating p and q in $\mathrm{VD}_2(S)$ can be done in $\mathcal{O}(\log n)$ time. According to Theorem 4.3, and having constructed $\mathrm{VD}_2(S)$, constructing graph G and finding a 2-path (if it exists) takes linear time. Since each step of the binary search is performed in linear time (the median of the list of ranges can be found in linear time regarding the number of ranges [25]), the overall time complexity to calculate $\mathrm{MR}_S(P(p,q))$ and find a 2-path between p and q is $\mathcal{O}(\log n) \times \mathcal{O}(n) = \mathcal{O}(n \log n)$.

The lower bound is achieved by a reduction to the Max-Gap problem: given a set of real numbers regarded as points on the x-axis, find the maximum distance (gap) between any two consecutive points once sorted. Lee and Wu [56] proved that the lower bound for this problem is $\Omega(n \log n)$ time. The lower bound for calculating $\mathrm{MR}_S(P(p,q))$ is the same since this problem can reduced to the Max-Gap. Let S be a set of n real numbers for the Max-Gap problem. For each number s_i, consider the point $(s_i, 0)$ on the x-axis. Duplicate every point in order to have $2n$ points on the x-axis. Consider $p = (\min\{s_0, \ldots, s_{n-1}\}, 0)$ and $q = (\max\{s_0, \ldots, s_{n-1}\}, 0)$, that is, p is the leftmost point on the x-axis and q is the rightmost. If $r = \mathrm{MR}_S(P(p,q))$, then the maximum gap for S is $2r$. As a result, the previous algorithm solves the Max-Gap problem, which results in both sharing the same lower bound: $\Omega(n \log n)$ time. □

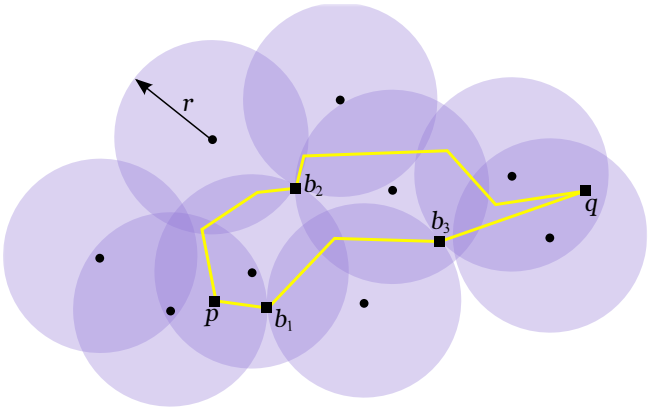


Figure 4.17: There are three bottleneck-points for 2-covered paths between p and q: b_1, b_2 and b_3. Two 2-covered paths connecting p to q are shown in yellow.

Degenerate cases were already briefly mentioned. The antennas' distribution might result in degenerate cases, which means that there can be more than one bottleneck-point for 2-paths between two points. Notwithstanding the fact that bottleneck-points are not unique, they can still be found using the previously described technique and they remain classified into two categories. The major difference is that, in the case of degenerate input data, 2-covered paths between two points do not have to cross every single bottleneck-point, although they must

cross at least one of them (see Figure 4.17). Observe that even though the classification of bottleneck-points does not change in the presence of degenerate cases, vertex bottleneck-points can correspond to Voronoi vertices with degree higher than three (in cases where more than three lenses meet at once).

4.4.3 Shortest 2-Covered Path with Minimum Transmission Range

Given a set S of n antennas and two points p and q on the plane, the previous algorithm calculates $\mathrm{MR}_S(P(p,q))$ and finds a 2-path on the plane between p and q in $\Theta(n \log n)$ time. However, that 2-path may not be optimal and, in that case, it can be shortened. What follows is therefore a method to find the shortest geometric 2-path between two points on the plane, once the minimum power transmission range that makes its existence possible is known. Such range clearly is independent of the chosen path, so $\mathrm{MR}_S(P(p,q))$ is assumed to be a known parameter and the antennas' range r is also assumed to be at least $\mathrm{MR}_S(P(p,q))$. The path to be computed can only traverse 2-covered regions in order to be a 2-path. In other words, such path has to exist within the union of lenses. Knowing how to find the shortest path between two points on a polygon [49], the natural approach is to convert the union of lenses into a polygon (with or without holes). In this way, the shortest 2-path between p and q can be easily computed within that polygon. To this end, the following algorithm transforms the union of lenses into a polygon (with or without holes).

ALGORITHM **Converting Lenses into a Polygon**

INPUT: Set S of n antennas with range r, $\mathrm{VD}_2(S)$, points p and q
OUTPUT: Polygon P

1. For each $\mathrm{VR}_2(s_i, s_j)$ do
   (a) Compute $I \leftarrow l_r(s_i, s_j) \cap \mathrm{VR}_2(s_i, s_j)$
   (b) If an apex of $l_r(s_i, s_j)$ is inside $\mathrm{VR}_2(s_i, s_j)$ then add it to I
   (c) For every pair $v_i, v_j \in I$ do
   If v_i, v_j lie on the same arc of $l_r(s_i, s_j)$ and $\overline{v_i v_j} \in \mathrm{int}(\mathrm{VR}_2(s_i, s_j))$
   Then connect v_i and v_j
2. Construct P using the chain(s) of line segments;
3. If p (or q) is not in P, find the closest edge $\overline{v_i v_j} \in P$ to p (or q) and replace $\overline{v_i v_j}$ with $\overline{v_i p}$ and $\overline{p v_j}$ (or $\overline{v_i q}$ and $\overline{q v_j}$).

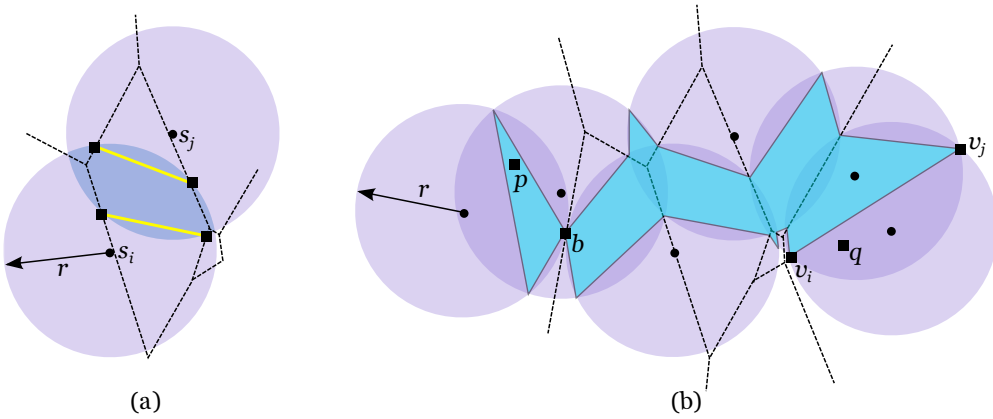


Figure 4.18: Set of antennas with range $r = \mathrm{MR}_S(P(p,q))$, $\mathrm{VD}_2(S)$ is represented by a dashed line. (a) There are four intersection points between $l_r(s_i, s_j)$ and $\mathrm{VR}_2(s_i, s_j)$. The yellow line segments connect intersection points that lie on the same arc of $l_r(s_i, s_j)$. (b) The union of lenses is converted into the blue polygon.

In Figure 4.18(a) there is an example of a second order Voronoi region defined by the antennas s_i and s_j. Lens $l_r(s_i, s_j)$ and $\mathrm{VR}_2(s_i, s_j)$ intersect in four points (represented by squares). Every pair of points lying on the same arc of $l_r(s_i, s_j)$ are connected through a line segment. Figure 4.18(b) shows an early stage of the algorithm. All the intersection points between the second order Voronoi regions and respective lenses are computed, as well as the line segments connecting pairs of intersection points that lie on the same arc of a lens. In the case the antennas' transmission range is $\mathrm{MR}_S(P(p,q))$, there is the need to be particularly cautious when connecting the bottleneck-point to the rest of the intersection points. If it is not correctly connected, then the resulting polygon will not be simple. This situation is further discussed below.

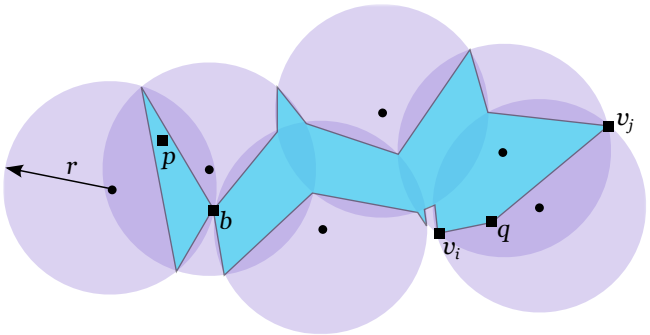


Figure 4.19: The union of lenses is converted into a blue polygon that contains both p and q.

In Figure 4.18(b), point q is not inside the polygon but this situation is corrected by replacing the closest edge of the polygon to q, $\overline{v_i v_j}$, with edges $\overline{v_i q}$ and $\overline{q v_j}$ (see Figure 4.19). This correction, if needed, is the last step of the algorithm. Finally, the union of lenses is

converted into a polygon (with holes in the case the arrangement of lenses has them as well).

Theorem 4.5 *Let S be a set of n antennas with range $r \geq \mathrm{MR}_S(P(p,q))$. Given two points p and q on the plane and having constructed $\mathrm{VD}_2(S)$, the union of lenses can be converted into a polygon containing p and q in $\mathcal{O}(n)$ time.*

Proof: For each $\mathrm{VR}_2(s_i, s_j)$, computing the set $I = l_r(s_i, s_j) \cap \mathrm{VR}_2(s_i, s_j)$ takes constant time since there are at most four intersection points between a second order Voronoi region and its corresponding lens (the Voronoi diagram has amortised linear complexity [21]). Consequently, set I is formed by at most n intersection points. Constructing a polygon P using the line segments connecting these intersection points takes linear time. Once P is constructed, locating p and q regarding P can be done in linear time since the polygon has a linear number of edges. In the case that p or q is outside P, the polygon can be rearranged to contain both points in $\mathcal{O}(n)$ time. □

It is now left to prove that the shortest 2-covered path between points p and q is contained in the polygon constructed by the previous algorithm.

Proposition 4.5 *Let S be a set of antennas with range $r \geq \mathrm{MR}_S(P(p,q))$. Given two points p and q on the plane, the shortest 2-covered path between p and q exists within the polygon constructed by the algorithm above.*

Proof: Since an optimal path can only be formed by optimal subpaths, a shortest 2-covered path between p and q is necessarily polygonal since it has to be formed by a sequence of shortest paths between each two consecutive vertices. And it is not difficult to see that the shortest path between two consecutive vertices is the line segment connecting them. Furthermore, this polygonal path has to lie within the union of lenses in order to be 2-covered by S. Let P be the polygon constructed from the union of lenses. There are only a few regions of this union that do not belong to P. Such regions are isolated 2-covered regions and are effectively "dead-ends". As a consequence, any path entering these regions is forced to return to P creating a much longer than necessary path. Therefore, a shortest 2-path between p and q exists within in P. □

If the antennas' transmission range is larger than $\mathrm{MR}_S(P(p,q))$, then a shortest path between p and q within polygon P can be found applying an algorithm by Kapoor et al. [49]. Otherwise, if the antennas' transmission range is precisely $\mathrm{MR}_S(P(p,q))$, then P has an unusual shape since four edges meet at a single vertex $b = b_S(p,q)$. In this case, in order to apply the algorithm by Kapoor et al. [49], P has to be split into two polygons, P_1 and P_2,

at vertex b (see Figure 4.20). In this case, and without loss of generality, suppose $p \in P_1$ and $q \in P_2$. A shortest path between p and q on P is found as the union of two shortest paths: a shortest path from p to b within P_1 and another one from b to q within P_2. Recall that every path between p and q has to cross b for this particular range.

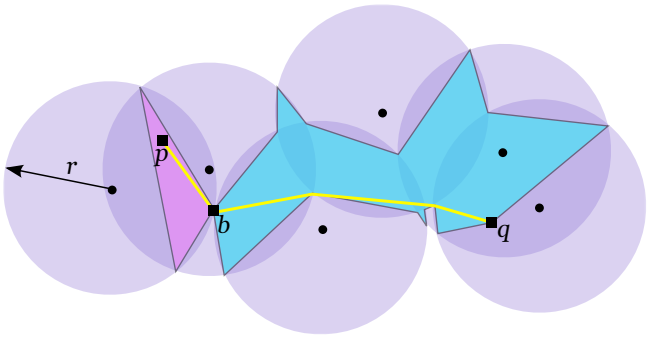


Figure 4.20: Set of antennas with range $r = \mathrm{MR}_S(P(p,q))$ and $b = b_S(p,q)$. Polygon P_1 is shown in pink and P_2 in blue. The shortest 2-path between p and q is represented by a solid yellow line.

In conclusion, given a set S of n antennas with range $r \geq \mathrm{MR}_S(P(p,q))$, a shortest 2-covered path between points p and q can be found in $\mathcal{O}((n+h^2)\log n)$ time [49], where h is the number of holes of polygon P.

4.5 Minimum Transmission Range to 2-Cover a Polygonal Region

Let R be a polygonal region (with or without holes) and S a set of n antennas with the same power transmission range. In the following, the boundary of R, denoted by $B(R)$, is considered to be part of R. A region is said to be 2-covered by S if every point on such region is 2-covered. The algorithm introduced in this section calculates the minimum power transmission range of S to 2-cover R, which is denoted by $\mathrm{MR}_S(R)$. Such range is the largest distance between a particular point on R and its second closest antenna. To find this point, assume that the antennas' range is sufficiently large to fully 2-cover R. Now suppose that this range is slowly reduced until there is one point q on R that no longer is 2-covered. As a result, R is not 2-covered until point q is, which means that the antennas' minimum range to 2-cover q is precisely the minimum range to fully 2-cover R, that is, $\mathrm{MR}_S(R) = \mathrm{MR}_S(q)$. The following proposition characterises the location of such point in relation to the polygonal region.

Proposition 4.6 *Let R be a polygonal region, q a point on R and S a set of antennas. If*

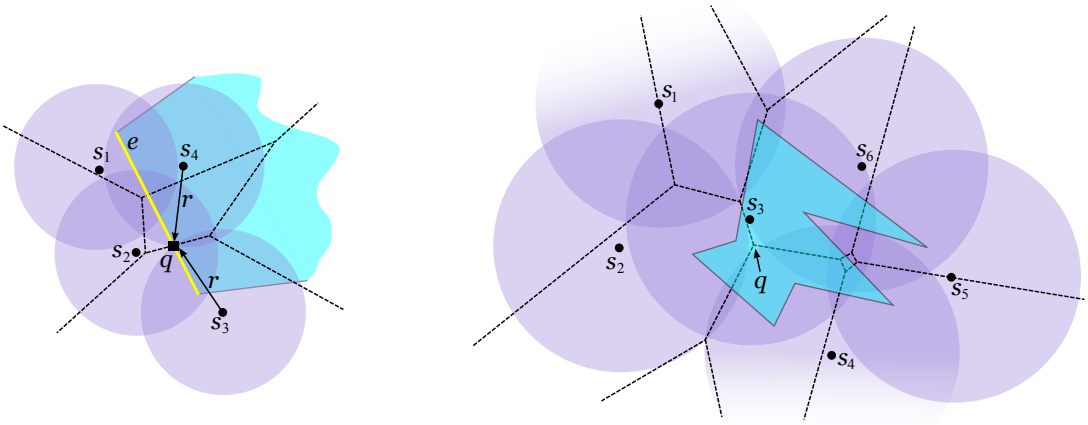
$\mathrm{MR}_S(R) = d(q, s_i)$, *for some antenna* $s_i \in S$, *then* q *can only be one of the following:*

(a) *A vertex of* R*;*

(b) *An intersection point between* $B(R)$ *and* $\mathrm{VD}_2(S)$*;*

(c) *A vertex of* $\mathrm{VD}_2(S)$ *inside* R*.*

Proof: Let $q \in R$ be such that $\mathrm{MR}_S(R) = d(q, s_i)$ for some antenna $s_i \in S$. Given an edge e of R, it was seen in Section 4.2 that $\mathrm{MR}_S(e)$ is calculated using the intersection points between e and $\mathrm{VD}_2(S)$ plus the endpoints of e (see Figure 4.21(a)). Consequently, if $q \in B(R)$ then q must be a vertex of R or an intersection point between $B(R)$ and $\mathrm{VD}_2(S)$. If q is inside R then the situation gets trickier since it does not depend on the shape of R but on the way lenses interact with each other. As the antennas' range increases, the lenses grow larger and fill the interior of R. The last interior point to be 2-covered must be a point lying in the last "hole" (meaning a region of R not yet 2-covered). These holes are filled when three lenses intersect at a time, since not only the intersection point has to be 2-covered but also its whole neighbourhood (see Figure 4.21(b)). According to Proposition 4.4, q is a type II bottleneck-point and therefore it has to be a vertex of $\mathrm{VD}_2(S)$. □



(a) (b)

Figure 4.21: Region R is shown in light blue and $\mathrm{VD}_2(S)$ is represented by a dashed line. (a) The yellow edge e of R is 2-covered with minimum range $r = \mathrm{MR}_S(e)$. (b) Point q is the vertex of $\mathrm{VD}_2(S)$ defined by s_2, s_4 and s_6, $\mathrm{MR}_S(R) = \mathrm{MR}_S(q)$.

According to this proposition, there are several candidates on R to be the point defining $\mathrm{MR}_S(R)$. Since every point on R has to be 2-covered, $\mathrm{MR}_S(R)$ can be calculated as the minimum range to 2-cover every such candidate. Therefore, there is the need to analyse every

vertex of R, intersection points between $B(R)$ and $\mathrm{VD}_2(S)$ and vertices of $\mathrm{VD}_2(S)$ inside R (see Figure 4.22(a)). In Figure 4.22(b) there is an example of a 2-covered polygonal region R with minimum power transmission range. Point $q \in B(R)$ defines $\mathrm{MR}_S(R)$ and consequently R is 2-covered if the antennas' transmission range is at least $\mathrm{MR}_S(q) = d(q, s_4) = d(q, s_9)$.

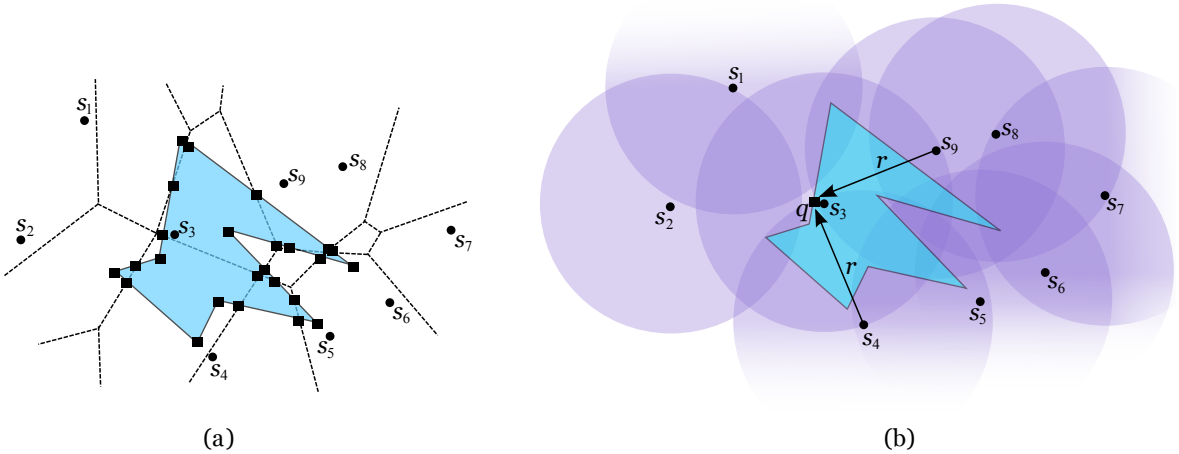


Figure 4.22: (a) The candidates are represented by squares: vertices of R, vertices of $\mathrm{VD}_2(S)$ inside R and points of $\{B(R) \cap \mathrm{VD}_2(S)\}$. (b) $\mathrm{MR}_S(R)$ is given by $\mathrm{MR}_S(q) = r$.

Given a polygonal region R, the following algorithm calculates $\mathrm{MR}_S(R)$.

ALGORITHM **Minimum Range to 2-Cover a Region**

INPUT: Set S of n antennas, polygonal region R
OUTPUT: $\mathrm{MR}_S(R)$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
2. Compute the intersection set $I \leftarrow B(R) \cap \mathrm{VD}_2(S)$;
3. Add the vertices of $\mathrm{VD}_2(S)$ that are inside R to set I;
4. Add the vertices of R to set I;
5. $\mathrm{MR}_S(R) \leftarrow \max\{\mathrm{MR}_S(q) : q \in I\}$.

The following theorem is a direct consequence of the method employed to calculate the antennas' minimum range to 2-cover a polygonal region.

Theorem 4.6 *Given a set S of n antennas and a polygonal region R with m vertices, $\mathrm{MR}_S(R)$ can be calculated in $\mathcal{O}(mn + n \log n)$ time and $\mathcal{O}(mn)$ space.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is formed by n antennas on the plane [55]. The cardinality of set $I = B(R) \cap \mathrm{VD}_2(S)$ is at most mn since $\mathrm{VD}_2(S)$ has n faces and R has m edges. Adding all the vertices of $\mathrm{VD}_2(S)$ that are inside R to set I takes $\mathcal{O}(mn)$ time. The region's vertices can be added to I in $\mathcal{O}(m)$ time. For each intersection point $q \in I$, $\mathrm{MR}_S(q)$ is calculated in constant time using $\mathrm{VD}_2(S)$ since it is given by the distance between q and its second closest antenna. The largest of these distances is $\mathrm{MR}_S(R)$. Overall, the time complexity of this procedure is $\mathcal{O}(mn)$. Regarding the space complexity, there is the need to store at most mn intersection points while $\mathrm{VD}_2(S)$ can be stored in $\mathcal{O}(n)$ space [55]. Consequently, this algorithm runs in $\mathcal{O}(mn + n \log n)$ time and $\mathcal{O}(mn)$ space. □

Corollary 4.1 *Given a set S of n antennas and a convex polygonal region R with m vertices, $\mathrm{MR}_S(R)$ can be calculated in $\mathcal{O}(m + n \log n)$ time and $\mathcal{O}(m + n)$ space.*

Observe that if $\mathrm{VD}_2(S)$ is previously constructed, $\mathrm{MR}_S(R)$ can be calculated in $\mathcal{O}(m+n)$ time and space for a convex polygonal region R.

4.6 Minimum Transmission Range to 2-Cover a Path on a Polygonal Region

Given two points p and q on a polygonal region R, the objective of this section is to calculate the minimum transmission range of the antennas such that there is a 2-covered path within R connecting p to q. This problem is more restrictive than the previous ones since the space where this path lies is enclosed by R. In Figure 4.23(a), the black path connecting p and q is not a 2-path since some of its points are only covered by one antenna. On the other hand, the yellow path in the same figure is a 2-path between p and q. Not only it is 2-covered by S as it exists within R, which makes it the interesting type of 2-path for this section. In the following, whenever 2-path is mentioned, it is understood as a 2-path within R. Similarly to what happened in Section 4.4, the solution to this problem is achieved by employing the associated decision problem, which is presented below.

4.6.1 Decision Problem: Is r large enough?

The present subsection proposes an algorithm to solve the following problem: given a set S of n antennas with power transmission range $r \in \mathbb{R}^+$ and two points p and q on R, decide if there is a 2-path connecting p and q. This decision algorithm will then be used to solve the main problem in Section 4.6.2. As previously mentioned, a 2-path from p to q exists if and only if

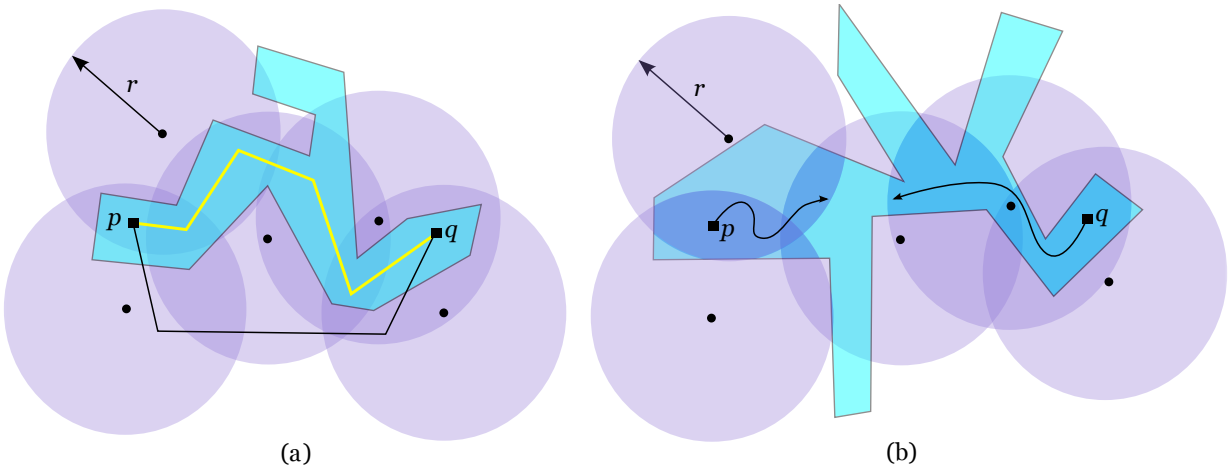


Figure 4.23: Region R is shown in blue. (a) The yellow path is a 2-path within R connecting p to q, whilst the black is not. (b) It is not possible to find a 2-path within R between p and q because they lie in different connected components of the union of lenses.

p and q lie in the same connected component of the union of lenses (see Figure 4.23(b)). If it exists, such 2-path only crosses the regions of R that are 2-covered. With this in mind, let A be the arrangement of the union of lenses confined to region R and intersected by $\mathrm{VD}_2(S)$ (see Figure 4.24). If there is a 2-path between p and q on R, then it exists within A. If R is not convex, then there can be more than one face of A per Voronoi region. For example, there are two non-connected faces of A defined by the spikes of R on the lower leftmost Voronoi region in Figure 4.24.

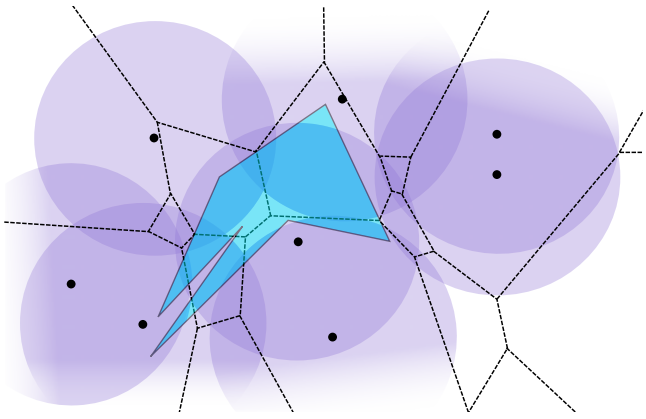


Figure 4.24: Polygonal region R is shown in blue, $\mathrm{VD}_2(S)$ is represented by a dashed line and the arrangement of the union of lenses confined to R and intersected by $\mathrm{VD}_2(S)$ is shown in dark blue.

According to Corollary 4.1, it is easier to work with convex regions since the number of intersection points decreases sharply. Moreover, there is only one face of A per Voronoi region. Consequently, the first step to solve this problem is to divide R into convex pieces (see Figure 4.25(a)). This is easily obtained using Steiner points: for each reflex vertex $v_r \in R$ extend a ray from v_r, which bisects the internal angle of R at v_r, until it reaches $B(R)$ or a previous ray. It can be shown that if R has k reflex vertices, then this set of rays divides R into $k+1$ convex pieces. There are some studies on the optimisation of the final number of convex pieces, either using Steiner points [30] or diagonals [53]. Notwithstanding these results and since it does not worsen the final complexity, the partition technique chosen in the following does not optimise the resulting number of convex pieces.

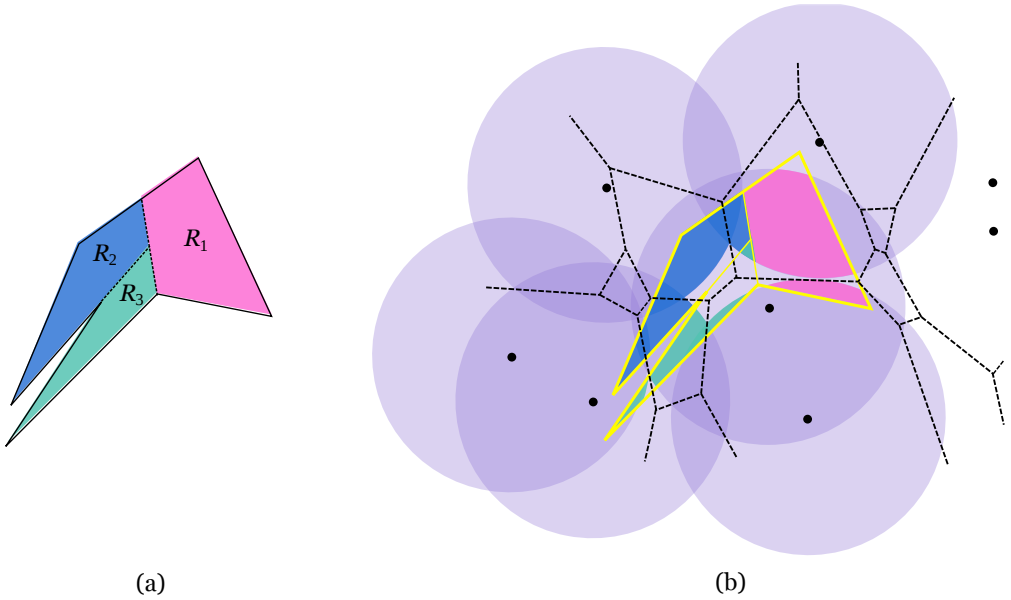


Figure 4.25: (a) Polygonal region divided into three convex pieces: R_1, R_2 and R_3. (b) The arrangements A_1, A_2 and A_3 are shown in pink, blue and green, respectively.

Th following algorithm constructs an intersection graph G_i of the arrangement A_i, which is in fact the arrangement A restricted to convex piece $R_i \subseteq R$ (see Figure 4.25(b)). This algorithm will then be applied to every convex piece of R in order to construct the intersection graph of the whole arrangement A. Recall that $\mathrm{VR}_2(s_i, s_j)$ is the second order Voronoi region of antennas s_i and s_j and $l_r(s_i, s_j)$ is the lens resulting from the intersection of discs $D(s_i, r)$ and $D(s_j, r)$.

ALGORITHM **Preprocess (PRE)**

INPUT: Set S of n antennas, range r, $\mathrm{VD}_2(S)$, convex piece R_i
OUTPUT: Arrangement A_i, graph G_i

1. Compute arrangement A_i:

   For each $\mathrm{VR}_2(s_i, s_j)$ do

   (a) Compute $l_r(s_i, s_j)$

   (b) The region resulting from the non-empty intersection between $\mathrm{VR}_2(s_i, s_j)$, $l_r(s_i, s_j)$ and R_i is a face of the arrangement A_i

2. Construct graph G_i:

   For each face $a_k \in A_i$ do

   (a) Add node n_k to G_i

   (b) For each neighbouring face $a_j \in A_i$ of a_k do

   If $a_k \cap a_j \neq \emptyset$ then add edge $\overline{n_k n_j}$ to G_i

The temporal complexity to construct the restricted arrangement and corresponding intersection graph is given in the following result.

Theorem 4.7 *Let S be a set of n antennas with transmission range r and R_i a convex region with m_i vertices. If $\mathrm{VD}_2(S)$ is preprocessed, then both arrangement A_i and graph G_i can be constructed in $\mathcal{O}(nm_i)$ time.*

Proof: Each lens $l_r(s_i, s_j)$ is intersected at most four times by $\mathrm{VR}_2(s_i, s_j)$ and the resulting region is convex. Consequently, the intersection between that convex region and R_i also is convex and can be found in $\mathcal{O}(m_i + n_i)$ time, n_i being the complexity of $\mathrm{VR}_2(s_i, s_j)$. As the Voronoi diagram has amortised linear complexity [21], the arrangement A_i can be found in $\mathcal{O}(n + nm_i) = \mathcal{O}(nm_i)$ time. Graph G_i, the intersection graph of A_i, has at most n nodes since it may have one node per Voronoi region that intersects R_i (see Figure 4.26(a)). Two nodes of G_i are connected if their corresponding faces of A_i intersect. The vertices of each face of A_i can be used to check if it intersects other faces of A_i. It suffices that each vertex is identified as an apex of a lens, a point on $B(R)$ or a point on a Voronoi edge. The latter is the important type of vertex for this procedure. Suppose face $a_k \in A_i$ has a vertex v that is a point on a Voronoi edge that separates $\mathrm{VR}_2(s_k, s_l)$ from $\mathrm{VR}_2(s_l, s_j)$. Then v belongs to lenses $l_r(s_k, s_l)$ and $l_r(s_l, s_j)$, which means faces a_k and a_j intersect each other. Therefore, to construct G_i it suffices to connect the nodes corresponding to the faces of A_i that share

at least one vertex on the same Voronoi edge. Overall, this arrangement and corresponding intersection graph can be constructed in $\mathcal{O}(nm_i)$ time. □

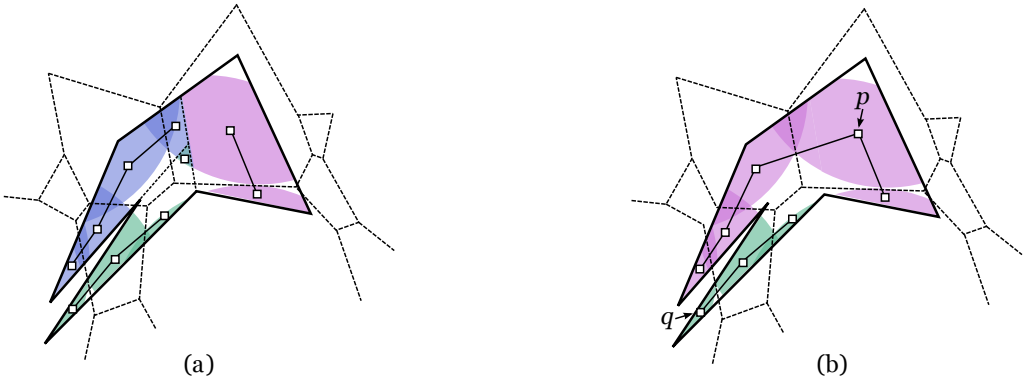


Figure 4.26: (a) Each subgraph corresponds to a coloured convex piece. The subgraph on the green region is disconnected. (b) There is not a 2-path between p and q because they lie in different connected components of the graph.

Note that G_i has as many edges as the Voronoi diagram and that it may not be connected (see the subgraph on the green region in Figure 4.26(a)). The strategy to decide if there is a 2-path between p and q is based on the construction of an intersection graph G of the whole arrangement A, which is in fact the union of the subgraphs constructed for each convex piece of R. It is not hard to see that a node of G_i is merged with another node of G_j if the faces of A they correspond to intersect each other (see Figures 4.26(a) and 4.26(b)). The general strategy to solve the decision problem proposed in this section is presented in the following pseudo-code, which applies the preprocess algorithm described above.

ALGORITHM **Power Transmission Range Test 2 (TRT2)**

INPUT: Set S of antennas, range r, $\mathrm{VD}_2(S)$, region R, points p and q
OUTPUT: YES or NO

1. Divide R in convex pieces by adding a set L of rays;
2. For each convex piece $R_i \subseteq R$ do
   $\{A_i, G_i\} \leftarrow \mathrm{PRE}(S, r, \mathrm{VD}_2(S), R_i)$
   $G \leftarrow G \cup G_i$
   $A \leftarrow A \cup A_i$

...

ALGORITHM **Power Transmission Range Test 2 (TRT2)** (cont.)

3. For every ray $l_i \in L$ do

   $F_i \leftarrow \{a \in A : \text{an edge of face } a \text{ is on } l_i\}$

   Merge the nodes of G whose corresponding faces of F_i lie on the same lens

4. Assign p to node n_p and q to node n_q;

5. Traverse G starting at p using the DFS algorithm [42];

6. If node q is found then return YES, otherwise return NO.

Point p is assigned to node n_p since n_p is assumed to be the node of G whose corresponding face of A contains p. In a similar way, point q is assigned to node n_q. Consequently, the existence of a 2-path connecting p to q depends on n_p and n_q being within the same connected component of G (see Figure 4.26(b)). The method chosen to traverse G is the Depth-First Search since its complexity is linear on the number of edges of the graph [42].

Theorem 4.8 *Let S be a set of n antennas with transmission range r and R a polygonal region divided into convex pieces by adding k rays. Let M be the largest complexity of the convex pieces. If $\mathrm{VD}_2(S)$ is preprocessed, then deciding if a 2-covered path between points p and q exists takes $\mathcal{O}(knM)$ time.*

Proof: Dividing region R with m vertices into convex pieces by adding a set L of k rays takes $\mathcal{O}(m+k^2 \log(\frac{m}{k}))$ time [30]. Consequently, R is divided into $k+1$ convex pieces. Supposing each convex piece R_i has m_i vertices, assume $M = \max\{m_1, \ldots, m_{k+1}\}$. According to Theorem 4.7, constructing G_i for each convex piece takes $\mathcal{O}(nm_i)$ time, so constructing a first version of G takes $\mathcal{O}(knM)$ time. For the same reason, A is also constructed in $\mathcal{O}(knM)$ time. Analysing the vertices of every face of A on a convex piece takes $\mathcal{O}(nM)$ time. Consequently, finding the sets of faces that have an edge on the same ray of L takes $\mathcal{O}(knM)$ time. Finding the lens that contains a face of A can be done in constant time. Every ray of L can intersect n Voronoi regions and consequently n faces of A. Therefore, each set of faces F_i has cardinality at most n. The vertices of the rays of L can have degree 3 at most [30], so the number of nodes of G to be merged at a time is at most 3, which can be done in constant time. Consequently, the construction of G is concluded in $\mathcal{O}(kn)$ time. Locating p and q on A can be done in $\mathcal{O}(nM)$ time. In the worst case, the Depth-First Search algorithm has to visit every node of G twice [42]. Therefore, traversing G to decide if there is a 2-path between points p and q takes $\mathcal{O}(kn)$ time. Overall, this decision problem can be solved in $\mathcal{O}(knM)$ time. □

4.6.2 Minimum Transmission Range to 2-Cover a Path on a Polygonal Region

The optimisation algorithm presented in this section calculates the minimum power transmission range of S that ensures the existence of a 2-path on R between p and q. Such range is denoted by $\mathrm{MR}_{S,R}(P(p,q))$. Observe that for $n \geq 2$, this problem always has a solution because, if the range r is large enough, all discs of radius r centred at points of S contain R. In this case, any path connecting p to q on R is a 2-path. Following the idea presented in Section 4.5, there is the need to locate point b that is the first intersection between the connected component of lenses containing p and the one containing q. As before, this point can either be on $B(R)$ or inside R. The first case was discussed in Proposition 4.6 and so b can be a vertex of R or an intersection point between $\mathrm{VD}_2(S)$ and $B(R)$. The second case is more complicated since now there is no need to completely cover R, in fact, all that is needed is that p and q lie in the same connected component of lenses. Therefore, b is a bottleneck-point for 2-paths between p and q, $b_S(p,q)$. Without loss of generality, degenerate input data is not considered, so for all purposes, bottleneck-points are regarded as unique for every pair of points. According to Proposition 4.4, candidates to bottleneck-points are found on the edges of $\mathrm{VD}_2(S)$ that are intersected by the line segment joining the two antennas defining such edge and vertices of $\mathrm{VD}_2(S)$ that are inside the triangle of the corresponding antennas. To conclude, $\mathrm{MR}_{S,R}(P(p,q))$ is calculated using a binary search on a list of ranges. Each range of the list is the minimum transmission range needed to 2-cover a candidate and each candidate is either p, q, a vertex of R, a point of $\{B(R) \cap \mathrm{VD}_2(S)\}$ or a bottleneck-point. In every step of the binary search, the corresponding range is evaluated to decide if it is large enough to allow a 2-path between p and q. The algorithm *Power Transmission Range Test 2* (TRT2) introduced in Section 4.6.1 will be used to this end. If the range is indeed large enough then the search proceeds on the lowest half of the ranges. Otherwise, the search continues on the highest half of the ranges. This method is summarised in the following pseudo-code, note that it is assumed that p and q are points inside R.

ALGORITHM **Minimum Range that allows a 2-Path on R**

INPUT: Set S of n antennas, region R, points p and q
OUTPUT: $\mathrm{MR}_{S,R}(P(p,q))$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S; ...

ALGORITHM **Minimum Range that allows a 2-Path on** R (cont.)

3. Compute set $I \leftarrow \{p, q\} \cup \{ B(R) \cap \mathrm{VD}_2(S)\}$;
   Add the vertices of R to set I
4. Find set B of candidates for bottleneck-points using Proposition 4.4;
5. $I' \leftarrow \{\mathrm{MR}_S(x) : x \in \{I \cup B\}\}$;
6. Calculate $r_0 = \max\{\mathrm{MR}_S(p), \mathrm{MR}_S(q)\}$;
7. Perform a binary search on $I' = \{r_i \in I' : r_i \geq r_0\} \cup \{ r_0\}$:
   For the median range $r_i \in I'$ do
   If $\mathrm{TRT2}(S, \mathrm{VD}_2(S), r_i, R, p, q) =$ YES
   Then proceed the search on $I' \leftarrow \{r_j \in I' : r_j \leq r_i\}$
   Otherwise proceed the search on $I' \leftarrow \{r_j \in I' : r_j > r_i\}$
8. The final range is $\mathrm{MR}_{S,R}(P(p, q))$.

In Figure 4.27 there is an example of a 2-path between p and q within a polygonal region. Any 2-path between these points only exists if the antennas' transmission range is at least $\mathrm{MR}_{S,R}(P(p, q))$. This range is calculated as the minimum range to 2-cover point $b = b_S(p, q)$, which is given by $d(s_2, b) = d(s_4, b)$. The following theorem concludes this subject.

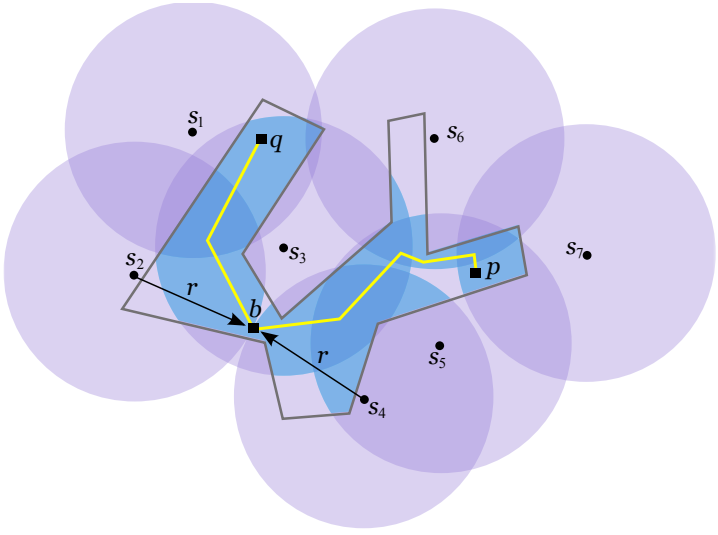


Figure 4.27: The yellow 2-path between p and q on R only exists if the antennas' transmission range is at least $\mathrm{MR}_{S,R}(P(p, q)) = \mathrm{MR}_S(b) = r$. The 2-covered regions of R are shown in light blue.

Theorem 4.9 *Let S be a set of n antennas and R a polygonal region with m vertices divided into convex pieces by adding k rays. Let M be the largest complexity of the convex pieces. Given two points p and q on R, $\mathrm{MR}_{S,R}(P(p,q))$ can be calculated in $\mathcal{O}(knM \log mn)$ time.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is a set of n antennas on the plane [55]. Set $I = \{p, q\} \cup \{B(R) \cap \mathrm{VD}_2(S)\}$ has cardinality mn since $\mathrm{VD}_2(S)$ is linear on n and R has m vertices. Adding m vertices to I plus the n candidates to bottleneck-points for 2-paths between p and q takes $\mathcal{O}(m+n)$ time. Therefore, set I can be found in $\mathcal{O}(mn)$ total time. This is also the temporal complexity of calculating $\mathrm{MR}_S(x)$ for every $x \in I$. Supposing each convex piece has m_i vertices, assume that $M = \max\{m_1, \ldots, m_{k+1}\}$. Then according to Theorem 4.8, each step of the binary search runs in $\mathcal{O}(knM)$ time. Consequently, the binary search runs in $\mathcal{O}(knM \log mn)$ time, which is the overall complexity to calculate $\mathrm{MR}_{S,R}(P(p,q))$. □

4.7 Minimum Transmission Range to 2-Cover a Path between any Two Points of a Set of Points

Let S be a set of n antennas and Q a set of m points on the plane (see Figure 4.28). The goal of this section is not to calculate the minimum range of S that 2-covers every point of Q, but to find the minimum range that ensures that all points of Q are within the same connected component of the union of lenses. In other words, the minimum range that allows the existence of a 2-path connecting any pair of points of Q. Such range, denoted by $\mathrm{MR}_S(Q)$, allows a user to move within Q using 2-paths exclusively. The following algorithm proposes a solution to this problem.

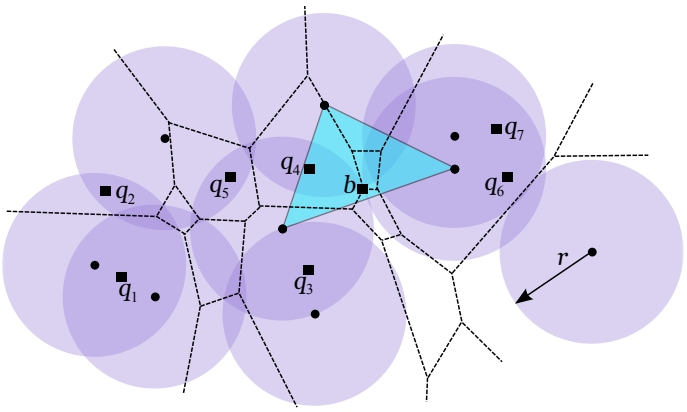


Figure 4.28: Set Q is represented by seven squares and set S by nine dots with range $r = \mathrm{MR}_S(b)$. Point b is a type II bottleneck-point and $\mathrm{VD}_2(S)$ is shown in a dashed line.

ALGORITHM **Minimum 2-Coverage of a Path between a Set of Points**

INPUT: Set S of antennas, set Q of points
OUTPUT: $\mathrm{MR}_S(Q)$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
2. Find set B of candidates for bottleneck-points using Proposition 4.4;
3. $I \leftarrow \{\mathrm{MR}_S(b) : b \in B\}$, $r_0 \leftarrow \max\{\mathrm{MR}_S(q) : q \in Q\}$;
4. Perform a binary search on $I = \{r_i \in I : r_i \geq r_0\} \cup \{r_0\}$:

   For the median range $r_i \in I$ do

   (a) Construct graph G as explained in step 3 of the algorithm TRT in Section 4.4.1

   (b) Associate each point $q \in Q$ with the node of G representing the lens defined by the two closest antennas to q

   (c) Traverse G starting at a node associated with a point of Q

   (d) If every point of Q is found

   Then proceed the search on $I \leftarrow \{r_j \in I : r_j \leq r_i\}$

   Otherwise proceed the search on $I \leftarrow \{r_j \in I : r_j > r_i\}$
5. The final range is $\mathrm{MR}_S(Q)$.

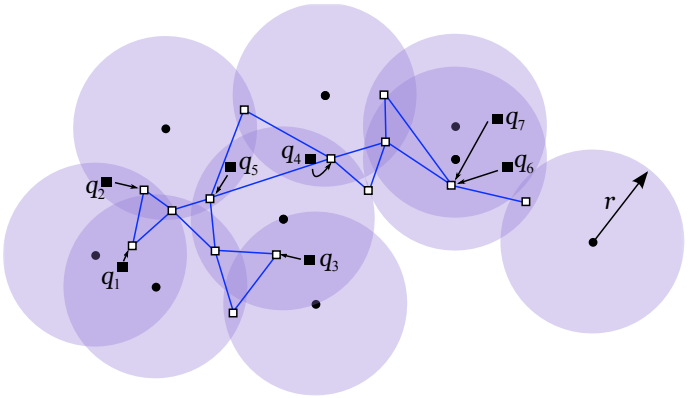


Figure 4.29: Each point of Q is assigned to its respective node of G (shown in blue). All points are in the same connected component of G.

Figures 4.28 and 4.29 illustrate this algorithm. In Figure 4.28, the antennas' range is given by $\mathrm{MR}_S(b)$ and b is a type II bottleneck-point since it is a vertex of $\mathrm{VD}_2(S)$ inside the triangle formed by the antennas that cover b. Graph G constructed for the range $\mathrm{MR}_S(b)$ is

represented by a solid blue line in Figure 4.29. Each point $q \in Q$ is assigned to the node of G that represents the lens defined by the two closest antennas to q. For this range, every point of Q is in the same connected component of the union of lenses and so the binary search proceeds on the lowest half of the remaining ranges. Once the binary search is over, the final range is $\mathrm{MR}_S(Q)$. In the example in Figures 4.28 and 4.29, $\mathrm{MR}_S(Q)$ is given precisely by $\mathrm{MR}_S(b)$.

Theorem 4.10 *Let S be a set of n antennas and Q a set of m points on the plane. The minimum power transmission range of S that ensures the existence of a 2-path connecting any pair of points of Q can be calculated in $\mathcal{O}((n+m)\log n)$ time.*

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is a set of n antennas on the plane [55]. Calculating $\mathrm{MR}_S(q)$ for each $q \in Q$ takes $\mathcal{O}(\log n)$ time since that is the temporal complexity of locating q on $\mathrm{VD}_2(S)$. Therefore, computing these ranges takes $\mathcal{O}(m \log n)$ time and range $r_0 = \max\{\mathrm{MR}_S(q) : q \in Q\}$ is calculated in $\mathcal{O}(m)$ time. Since $\mathrm{VD}_2(S)$ has n vertices and edges [21], searching for possible bottleneck-points takes $\mathcal{O}(n)$ time. The median of the list of ranges can be found in linear time regarding the number of ranges [25]. The intersection graph G of the set of lenses is constructed for each range of the binary search. Having constructed $\mathrm{VD}_2(S)$, this graph can be constructed in $\mathcal{O}(n)$ time since it is a subgraph of the dual graph of $\mathrm{VD}_2(S)$ and therefore has linear size [21]. The points of Q were already located in $\mathrm{VD}_2(S)$, so associating each with the corresponding node of G takes $\mathcal{O}(m)$ time. Graph G is traversed in $\mathcal{O}(n)$ time using the DFS algorithm [42]. Consequently, the binary search is performed in $\max\{\mathcal{O}(n), \mathcal{O}(m)\}$ time for each range. In conclusion, the final temporal complexity to calculate the minimum power transmission range of S that ensures the existence of a 2-path connecting any pair of points of Q is $\max\{\mathcal{O}(n \log n), \mathcal{O}(m \log n)\}$, that is, $\mathcal{O}((n+m)\log n)$. □

4.8 Minimum 2-Covered Path between two Line Segments

Let e_1 and e_2 be two line segments on the plane. Continuing the discussion of the last section, the objective of the present section is to calculate the minimum transmission range of S so that a 2-covered path between e_1 and e_2 exists. This implies that at least two points of both line segments have to be within the same connected component of the union of lenses. Let $\mathrm{MR}_S(P(e_1, e_2))$ denote the minimum range that ensures the existence of a 2-path between e_1 and e_2. In the following, it is shown how to calculate $\mathrm{MR}_S(P(e_1, e_2))$ and find a 2-path between the two line segments. The following pseudo-code applies the algorithm TRT (described in Section 4.4.1) to decide if there is a path 2-covered by S with range r between points p and q.

ALGORITHM **Minimum 2-Covered Path between two Segments**

INPUT: Set S of n antennas, line segments e_1 and e_2
OUTPUT: $\mathrm{MR}_S(P(e_1, e_2))$

1. Compute $\mathrm{VD}_2(S)$, the second order Voronoi diagram of S;
2. $I \leftarrow \{\{e_1\} \cap \mathrm{VD}_2(S)\} \cup \{\{e_2\} \cap \mathrm{VD}_2(S)\}$;
3. Add the endpoints of e_1 and e_2 to set I;
4. Find set B of candidates for bottleneck-points using Proposition 4.4;
5. $I' \leftarrow \{\mathrm{MR}_S(q) : q \in I \cup B\}$;
6. Perform a binary search on I':

   For the median range $r_i \in I'$ do

   (a) Compute set $P_1 \subseteq e_1$ of midpoints of pieces of e_1 that are 2-covered by S with range r_i, similarly, compute set $P_2 \subseteq e_2$

   (b) For every pair (p_i, q_i), $p_i \in P_1$ and $q_i \in P_2$, do

   If $\mathrm{TRT}(S, r_i, p_i, q_i) =$ YES

   Then continue the search on $I' \leftarrow \{r_j \in I' : r_j \leq r_i\}$

   Otherwise continue the search on $I' \leftarrow \{r_j \in I' : r_j > r_i\}$

7. The final range is $\mathrm{MR}_S(P(e_1, e_2))$.

The algorithm TRT presented in Section 4.4.1 constructs a subgraph of the intersection graph of the union of lenses for each range r_i and it can be used to compute a 2-path from e_1 to e_2. Once the binary search is finished, the pair of points (p_i, q_i), $p_i \in e_1$ and $q_i \in e_2$, which is in the same connected component of the union of lenses for the range $\mathrm{MR}_S(P(e_1, e_2))$ is known (see Figure 4.30(a)). Therefore, a polygonal 2-path between p_i and q_i can be found applying the method introduced in Section 4.4.2 to the same end. For every edge of the path from vertex p_i to q_i on the graph, compute a point on its dual Voronoi edge such that it lies in the intersection of the two lenses associated with this edge. The straight line segment connecting two consecutive such points is entirely contained in one lens. If needed, this path is completed by adding the line segments that connect p_i to the first node and q_i to the last node of the 2-path. This ensures that these line segments form a 2-covered path from p_i to q_i (see Figure 4.30(b)).

Theorem 4.11 *Let S be a set of n antennas and e_1 and e_2 two line segments on the plane,*

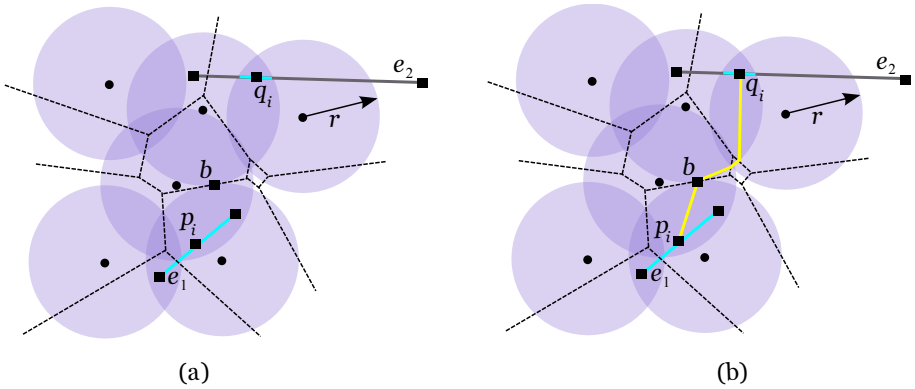


Figure 4.30: Set of six antennas with range $r = \mathrm{MR}_S(b)$. (a) The 2-covered pieces of e_1 and e_2 are shown in blue. (b) The yellow path is a 2-path between e_1 and e_2 with $r = \mathrm{MR}_S(P(e_1, e_2))$.

$\mathrm{MR}_S(P(e_1, e_2))$ and a 2-path between e_1 and e_2 can both be found in $\mathcal{O}(n^3 \log n)$ time.

Proof: Computing $\mathrm{VD}_2(S)$ takes $\mathcal{O}(n \log n)$ time since S is formed by n antennas on the plane [55]. Set $I = \{\{e_1\} \cap \mathrm{VD}_2(S)\} \cup \{\{e_2\} \cap \mathrm{VD}_2(S)\}$ is formed by a linear number of intersection points since $\mathrm{VD}_2(S)$ has a linear number of regions [21]. Given the diagram's linear complexity, searching for possible bottleneck-points takes $\mathcal{O}(n)$ time. The median of the list of minimum ranges to 2-cover each candidate can be also found in linear time regarding the number of ranges [25]. Since the intersection points between each line segment and $\mathrm{VD}_2(S)$ are already computed, finding which pieces of e_1 and e_2 are 2-covered by S takes linear time. Finding sets P_1 and P_2 takes the same amount of time because there are at most n midpoints. However, this generates up to n^2 pairs of midpoints (p_i, q_i), $p_i \in P_1$ and $q_i \in P_2$. Since the algorithm TRT is linear for each pair of points (according to Theorem 4.3 and having $\mathrm{VD}_2(S)$ already constructed), $\mathrm{MR}_S(P(e_1, e_2))$ and a 2-path between e_1 and e_2 can be found in $\mathcal{O}(n^3 \log n)$ time. □

4.9 Closing Remarks and Future Research

This chapter was dedicated to the following optimisation geometric problem: minimise the power transmission range of a set S of antennas to 2-cover a given region R or to ensure the existence of a path on that region. The final complexities of the algorithms proposed to solve this problem, regarding different types of regions, are shown in Table 4.1. In that table it is considered that S is a set of n antennas, R a polygonal region with m vertices and M the largest complexity of the convex pieces in which R is divided by adding k segments.

Analysing all the complexities of the algorithms proposed in this chapter, only one is

Minimum 2-Coverage of	Algorithm's Complexity
Line segment	$\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space
Planar graph with m edges	$\mathcal{O}(mn)$ time, $m > \log n$
Path on a planar edge-weighted graph with m edges	$\mathcal{O}(m)$ time, $m > \log n$
Path on the plane (decision)	$\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space
Path on the plane	$\Theta(n \log n)$ time
Polygonal region with m vertices	$\mathcal{O}(mn + n \log n)$ time and $\mathcal{O}(mn)$ space
Convex polygonal region with m vertices	$\mathcal{O}(m + n \log n)$ time and $\mathcal{O}(m + n)$ space
Path on a polygonal region R (decision)	$\mathcal{O}(knM)$ time
Path on a polygonal region R	$\mathcal{O}(knM \log mn)$ time
Path between any two points of a set of m points	$\mathcal{O}((m + n) \log n)$ time
Path between two line segments	$\mathcal{O}(n^3 \log n)$ time
Other Problems	**Algorithm's Complexity**
Converting the union of lenses into a polygon P	$\mathcal{O}(n \log n)$ time
Constructing the shortest 2-path on P	$\mathcal{O}((n + h^2) \log n)$ time [49]

Table 4.1: Complexities of the algorithms proposed in this chapter.

proven to be optimal (Theorem 4.4 in Section 4.4.2). The key question to reduce the remaining complexities is whether there is another way to directly compute the point of the region or path that needs the largest transmission range in order to be 2-covered. Such question is this chapter's main unresolved problem.

Given two points p and q on the plane, the algorithm introduced in Section 4.2 minimises the antennas' range to 2-cover a line segment connecting p to q. Since there is only a linear number of intersection points to be studied, it is natural that this complexity can be lowered. Nevertheless, that implies that the second order Voronoi diagram of S cannot be used in the solution. There follows another problem where region R is still degenerated: R is a planar graph G with m edges in Section 4.3. The most interesting aspect in this problem is the computation of a minimum 2-covered path between two nodes of G. Since G can be regarded as a model of a street network, a path on that graph may be seen as journey from one location to another, where the user is always within reach of two antennas. As a result, any user on that path is eligible to receive the service provided by the antennas, even if one of them fails. This type of path and the minimum range of S that allows its existence can be found in $\mathcal{O}(m)$ time (after a preprocess that runs in $\mathcal{O}(mn)$ time, assuming $m > \log n$). In Section 4.4, the minimum transmission range to ensure the existence of a path on the plane is solved

using a decision algorithm. Such algorithm (presented in Section 4.4.1) revealed to be quite a breakthrough not only to solve the main problem in this section but also to solve other problems in this chapter. Optimising the antennas' range such that there is a 2-path between two points on the plane can be solved in $\Theta(n \log n)$ time. The lower bound of this complexity is achieved by reducing the original problem to the Max-Gap problem [56]. Moreover, in that section it is also shown how to compute the shortest 2-covered path between two points on the plane using a polygon enclosed by the union of lenses.

Some other problems associated with this subject remain for future research. Consider a complete edge-weighted graph whose nodes are points of S. The weight of each edge of such graph is given by the Euclidean distance between the nodes it connects. The Euclidean Minimum Spanning Tree, EMST, is a minimum spanning tree for this graph such that the total length of its edges is minimised. For future research, how should p and q be chosen in Section 4.4.2 to compute the largest edge of an EMST? It is known that an EMST of S can be computed in $\mathcal{O}(n \log n)$ time [72] and its largest edge can be found in linear time. Moreover, the same question can be asked of the Gabriel graph [40]. Two points p and q are connected through an edge of the Gabriel graph if the disc of diameter $\overline{pq}$ is empty (this graph is a subgraph of the Delaunay triangulation). Can the problem solved in Section 4.4.2 compute the largest edge of the Gabriel graph as well? This graph has the Euclidean Minimum Spanning Tree and the Nearest Neighbour Graph as subgraphs.

Finally in Section 4.5, R is regarded as a polygonal region with m vertices. It is shown how to minimise the range of S to 2-cover the whole region in $\mathcal{O}(mn + n \log n)$ time. This upper bound is probably tight because there is the need to study mn intersection points. However, this complexity is lowered to $\mathcal{O}(m + n \log n)$ time if R is convex. With this in mind, the minimisation of the antennas' range to ensure the existence of a 2-path between two points on R was solved using a division of R into convex pieces. Although the antennas' range is optimised to ensure the existence of a 2-path on R, it is never shown how to construct such a path. Future research obviously passes through explaining how to construct a 2-path within R and even how to construct the shortest 2-path. Currently, it is not clear how to do this efficiently because the union of lenses restricted to R has an uncommon shape, some edges are arcs while others are line segments.

The algorithms presented in Sections 4.7 and 4.8 solve a problem directed at optimising the range of S to ensure the existence of a path between any pair of points of a set of points and the existence of a path between two line segments, respectively. Future research associated with the first problem passes through generalising points to other geometric objects, for example, simple polygons. The algorithm that solves the second problem has the most challenging

temporal complexity of all the algorithms proposed in this chapter: $\mathcal{O}(n^3 \log n)$. The reason behind this is the fact that each step of the binary search takes $\mathcal{O}(n^3)$ time. Decreasing this complexity remains for future research.

To summarise the future research involving the problems discussed in this chapter, the following studies are naturally associated with expanding these ideas and optimisation algorithms to k-coverage, $k > 2$. As the signal decays with the distance from the antennas, it is reasonable to consider that the energy needed by each antenna is an increasing function of a power of the transmission range. Therefore, further research on minimising the energy spent by a given network should take this model into account. Moreover, the antennas' transmission range may also be variable while time is passing by or even not fixed for all the antennas (this subject has already been addressed for 1-coverage [57, 85]). Since the main reason to study 2-coverage is associated with the assumption that devices fail, a future approach could also take this into account using fallibility probability. Each device is assigned a probability of failing and the 2-covered path should be found on the areas where the probability of both devices failing is lower.

Chapter 5

Other Problems Involving Coverage

Given a set S of antennas with a given transmission range $r \in \mathbb{R}^+$, the objective of the first part of this chapter is to decide which is the maximum coverage of a path within a polygonal region R. Similarly to the previous chapter, distinct versions of this problem arise for different types of regions: a line segment, a planar graph, a polygonal region and, to conclude, the whole plane. The second part of this chapter introduces a more restrictive definition of coverage. Given a point q on the plane, q is said to be $\frac{\pi}{2}$-covered by antennas s_i and s_j of S if the angle $\measuredangle(s_i, q, s_j) \geq \frac{\pi}{2}$. This restriction ensures that the antennas surround their service area uniformly. It is shown how to minimise the transmission range of S to $\frac{\pi}{2}$-cover a point and how to construct the $\frac{\pi}{2}$-covered region and its contour. The Coverage Voronoi diagram, which is associated with Embracing Voronoi diagrams, is also introduced in this chapter as an important geometric tool to solve the latter problem.

5.1 Maximum Coverage of a Path within a Region

5.1.1 Introduction

This chapter is divided into two parts and the first part extends last chapter's discussion: coverage problems. Therefore, the bibliography regarding coverage presented in Chapter 4 is still an important background for this section. For instance, the applications presented before for sensor networks, such as habitat monitoring or weather forecast [81], remain pertinent; this is also true for studies that optimise the coverage of the service area provided by a particular sensor network [26, 57, 64, 65]. However, the most important related work is by Huang and Tseng [46]. They proposed an algorithm to decide if every point on a polygonal region or

in the service area of the sensor network is monitored by at least k sensors. Following their approach to coverage, the objective of the next group of problems is to decide whether there is a covered path within a polygonal region. And if so, calculate the maximum coverage of such path. As the previous chapter, the type of coverage studied in the following discussion falls into the category of best-case coverage, that is, it is intended to identify the "best" monitored zones of a particular region.

Let S be a set of n points on the plane that represent the location of n devices that are able to send or receive some sort of wireless signal, like antennas. Also assume that the devices of S are homogeneous, that is, they all have the same power transmission range $r \in \mathbb{R}^+$, which is fixed. An object is said to be *k-covered* by S if every point of such object is within range of at least k antennas of S. It is apparent that this definition is a generalisation of the concept of 2-coverage introduced in Chapter 4. Let $\mathrm{MC}_S(x)$ denote the maximum coverage that S provides to an object x. If D is the set of discs of radius r each centred at an antenna of S, $D = \{D(s_i, r) : s_i \in S\}$, then $\mathrm{MC}_S(x) = |\{D(s_i, r) : x \in D(s_i, r), s_i \in S\}|$. This definition is illustrated in Figure 5.1. In that example, point q_1 is 3-covered since it is within reach of antennas s_1, s_2 and s_5, and consequently $\mathrm{MC}_S(q_1) = 3$. A path between two points is called a k-covered path or a k-path if every point of such path is k-covered by S. Since the antennas' transmission range is a given parameter, the algorithms proposed in the following aim to calculate the maximum $k \in \mathbb{N}$ so that a k-path between two points on a polygonal region R exists. This goal can be restated in another way: calculate the maximum number of discs of D so that every point of a path within R is covered by the same number of discs.

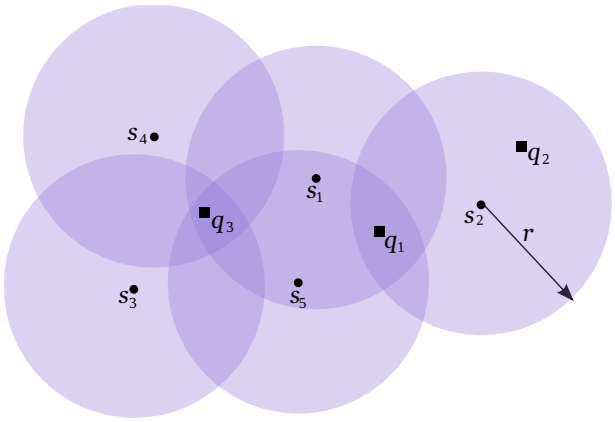


Figure 5.1: Set S with range r is represented by dots. Point q_1 has $\mathrm{MC}_S(q_1) = 3$ since it is a point on $D(s_1, r) \cap D(s_2, r) \cap D(s_5, r)$. Point q_2 is 1-covered by S since it is only inside $D(s_2, r)$ and $\mathrm{MC}_S(q_3) = 4$ since q_3 is a point on the intersection of four discs.

There are several works involving this subject. For Kaučič and Žalikm [51], k-guarding a surface patch is having it guarded by at least k guards. They proposed three heuristics (one

of which is original) to find a minimum set of vertex guards that k-guard a given terrain. On the other hand, Belleville et al. [23] consider that k-guarding a polygon P means that it is possible to find a set of guards on the edges of P (at most one guard per edge) such that every point on P is visible to at least k guards. They proved that such set exists to 1-guard every polygon with holes and 2-guard every polygon with just one hole. Furthermore, they also proved that not every polygon with holes is 2-guardable. Although already mentioned in Chapter 2, the work by Efrat et al. [38] was mainly written under the topic of k-coverage. They solve two distinct problems: where to locate a base station in order to optimise a given network's lifespan and how to minimise the number of sensors required to have every point in a particular area "well covered". The second problem is associated with multiple coverage because of their definition of "good coverage": a point on the plane is well covered if it is seen by three sensors that form a triangle containing the point or if it is seen by two sensors that are separated by an angle of at least α. The first definition is a variant of 3-coverage that was already addressed in Chapter 2, whilst the second will be further discussed in the second part of this chapter (Section 5.2). To conclude, Zhou et al. [84] proposed a centralised greedy algorithm to compute a minimum energy-cost k-coverage (this algorithm extended an earlier study by Wang et al. [80]).

As previously stated, Huang and Tseng [46] studied a similar problem to the one presented in this section. They proposed an algorithm to decide if every point on a polygonal region R or in the service area of the antennas is k-covered. The solution they found is based on the boundary of the area covered by each antenna, that is, on the boundary of $D(s_i, r)$ for $s_i \in S$. They define s_i as *k-perimeter-covered* if every point on the boundary of $D(s_i, r)$ is k-covered by $S \backslash \{s_i\}$. According to that definition, they calculate the pieces of the boundary of $D(s_i, r)$ that are within reach of other antennas. For each antenna that captures the signal of s_i, they compute the starting and ending angle of the arc of $D(s_i, r)$ that is within reach of that particular antenna. Afterwards, they sort all these intervals into a list ranging from 0 to 2π. Finally, every element on said list is studied to determine the perimeter-coverage of s_i. Assuming d is the number of antennas that are within reach of s_i, the perimeter-coverage of s_i can be calculated in $\mathcal{O}(d \log d)$ time. Overall, the perimeter-coverage of every antenna is found in $\mathcal{O}(nd \log d)$ time, which may add up to $\mathcal{O}(n^2 \log n)$ time if there are n antennas and most of them are within range of the others. In the example in Figure 5.2, region R is 1-covered by S. Huang and Tseng also extended their decision problem to three dimensions and solved it in $\mathcal{O}(nd^2 \log d)$ time [47]. In this situation, the region covered by each antenna is modelled by a ball. This algorithm also handles the case where each antenna has a different transmission range, that is, each ball has a different radius. To conclude, perimeter-coverage can be improved if a crossing-coverage approach is used instead [45, 83].

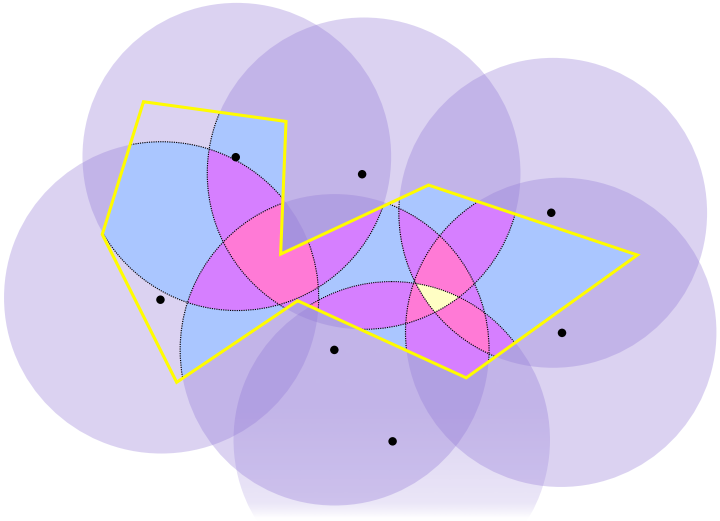


Figure 5.2: The regions of the polygon are coloured according to their coverage: colourless faces are 1-covered, blue are 2-covered, purple are 3-covered, pink are 4-covered and the yellow face is 5-covered.

The structure of the first part of this chapter is introduced in the following. Given a set S of antennas with transmission range $r \in \mathbb{R}^+$, Section 5.1.2 decides which is the maximum $k \in \mathbb{N}$ such that the line segment connecting points p and q on the plane is k-covered by S. This algorithm will then be used in Section 5.1.3 to calculate the maximum coverage of a path between two nodes of a geometric planar graph. It is also shown how to find such a path. In turn, the idea introduced in Section 5.1.3 contributes to the resolution of the problems presented in Sections 5.1.4 and 5.1.5. The problem of deciding which is the maximum coverage of a path between two points within a polygonal region is analysed in Section 5.1.4 and the case where the path exists on the plane is discussed in Section 5.1.5. Section 5.3 presents a summary of the algorithms proposed in this chapter, as well as a discussion of the results along with future research on the topic.

5.1.2 Maximum Coverage of a Line Segment

Given two points p and q on the plane, the following algorithm decides which is the maximum $k \in \mathbb{N}$ so that the line segment connecting both points, $\overline{pq}$, is k-covered. Its strategy is similar to the one presented in Section 4.2 in the last chapter. Line segment $\overline{pq}$ is first divided into several pieces and then each piece is analysed separately. Let I be the sorted set of the intersection points between $\overline{pq}$ and $D = \{D(s_i, r) : s_i \in S\}$. Each pair of consecutive points of I defines a piece of $\overline{pq}$ (see Figure 5.3). Consequent to the method used to find I, each of these pieces is covered by the same antennas. In other words, the number of discs covering each

piece of $\overline{pq}$ is constant. Therefore, the maximum coverage of $\overline{pq}$ can be calculated using only the points of the line segment that are also points of I. Moreover, the maximum coverage of $i \in I$ can be calculated from the previous intersection point since the number of discs covering two adjacent pieces of $\overline{pq}$ only differs by 1. As a result, it suffices to calculate the maximum coverage of the first point of I, the rest follows either by adding or subtracting 1.

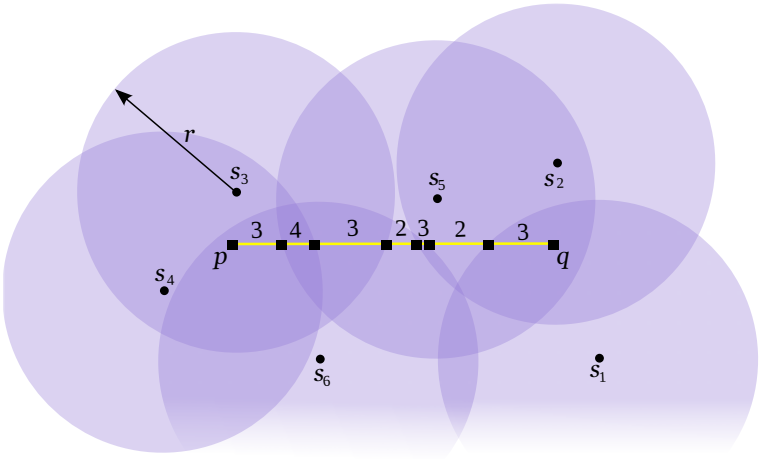


Figure 5.3: Set $S = \{s_1, \ldots, s_6\}$ with range r is represented by dots. Yellow line segment $\overline{pq}$ is divided into seven pieces and above each piece there is the number of discs that cover it; $\mathrm{MC}_S(\overline{pq}) = \min\{3, 4, 3, 2, 3, 2, 3\} = 2$.

The following pseudo-code outlines the algorithm to calculate the maximum coverage of $\overline{pq}$, which is denoted by $\mathrm{MC}_S(\overline{pq})$.

ALGORITHM **Maximum Coverage of a Line Segment**

INPUT: Set S of antennas with range r, points p and q
OUTPUT: $\mathrm{MC}_S(\overline{pq})$

1. $I \leftarrow \{\{\overline{pq}\} \cap \{ D(s_i, r) : s_i \in S\}\} \cup \{p, q\}$;
2. Sort $I = \{i_0, \ldots, i_m\}$ along $\overline{pq}$;
3. $\mathrm{MC}_S(i_0) \leftarrow |\{D(s_i, r) : i_0 \in D(s_i, r), s_i \in S\}|$;
4. From $j = 1$ to m do
   If a new disc covers $i_j \in I$
   Then $\mathrm{MC}_S(i_j) \leftarrow \mathrm{MC}_S(i_{j-1}) + 1$
   Otherwise $\mathrm{MC}_S(i_j) \leftarrow \mathrm{MC}_S(i_{j-1}) - 1$
5. $\mathrm{MC}_S(\overline{pq}) \leftarrow \min\{\mathrm{MC}_S(i_0), \ldots, \mathrm{MC}_S(i_m)\}$.

The temporal complexity of the algorithm above is stated in the following result.

Theorem 5.1 *Given a set S of n antennas with transmission range $r \in \mathbb{R}^+$ and two points p and q on the plane, $\mathrm{MC}_S(\overline{pq})$ can be calculated in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Proof: Since set S is formed by n antennas on the plane, set D also has n discs. There is a linear number of intersection points between the arrangement of discs and $\overline{pq}$ since each disc can only intersect $\overline{pq}$ twice. Let I be that set of intersection points plus p and q. Consequently, sorting I takes $\mathcal{O}(n \log n)$ time. While being computed, each intersection point $i_j \in I$ is flagged to indicate if a new disc is covering the next piece of $\overline{pq}$ or not. The maximum coverage of $i_0 \in I$, the first point of I, is calculated in linear time. In turn, the maximum coverage of each of the remaining intersection points can be easily calculated in constant time by adding 1 to or subtracting 1 from the maximum coverage of the previous point. Therefore, $\mathrm{MC}_S(\overline{pq}) = \min\{\mathrm{MC}_S(i_0), \ldots, \mathrm{MC}_S(i_m)\}$ can be calculated in linear time. In conclusion, the temporal complexity for this procedure is $\mathcal{O}(n \log n)$. Regarding space complexity, there is a linear number of antennas and therefore a linear number of intersection points between the discs of D and the line segment. Therefore, the algorithm takes $\mathcal{O}(n)$ space. □

Note that if there are many queries regarding the same set of antennas, there is another solution that is more efficient. In this case, it is convenient to preprocess the arrangement of discs D, which can be done in $\mathcal{O}(n^2)$ time [31] if all discs have the same radius. That way, each line segment connecting two points can only cross a linear number of discs (at most, it crosses the same disc twice). The number of discs that cover each intersection point is given by the faces of the arrangement. Consequently, the number of discs covering every piece of the line segment is controllable in $\mathcal{O}(n)$ time. Finally, the minimum number of discs covering every piece of the line segment is the maximum coverage of such line segment.

5.1.3 Maximum Covered Path on a Planar Graph

Let $G = (N, E)$ be a connected geometric planar graph. The algorithm proposed in this section calculates the maximum coverage of a path between two nodes of G. The maximum coverage of the whole graph is not difficult to calculate as the following proposition shows. Observe that this result is a generalisation of Theorem 5.1.

Proposition 5.1 *Let S be a set of n antennas with transmission range $r \in \mathbb{R}^+$ and G a connected geometric planar graph with m edges. The maximum coverage of G can be calculated in $\mathcal{O}(mn \log n)$ time and $\mathcal{O}(mn)$ space.*

Let n_i and n_j be two nodes of G, then a path connecting these nodes using only the edges of G that are k-covered by S is a k-path (see Figure 5.4(a)). The following algorithm calculates the maximum $k \in \mathbb{N}$ such that a k-path between nodes n_i and n_j, $P(n_i, n_j)$, exists on G. Such k is denoted by $\mathrm{MC}_{S,G}(P(n_i, n_j))$ or $\mathrm{MC}_S(P(n_i, n_j))$ if graph G is clear from the context.

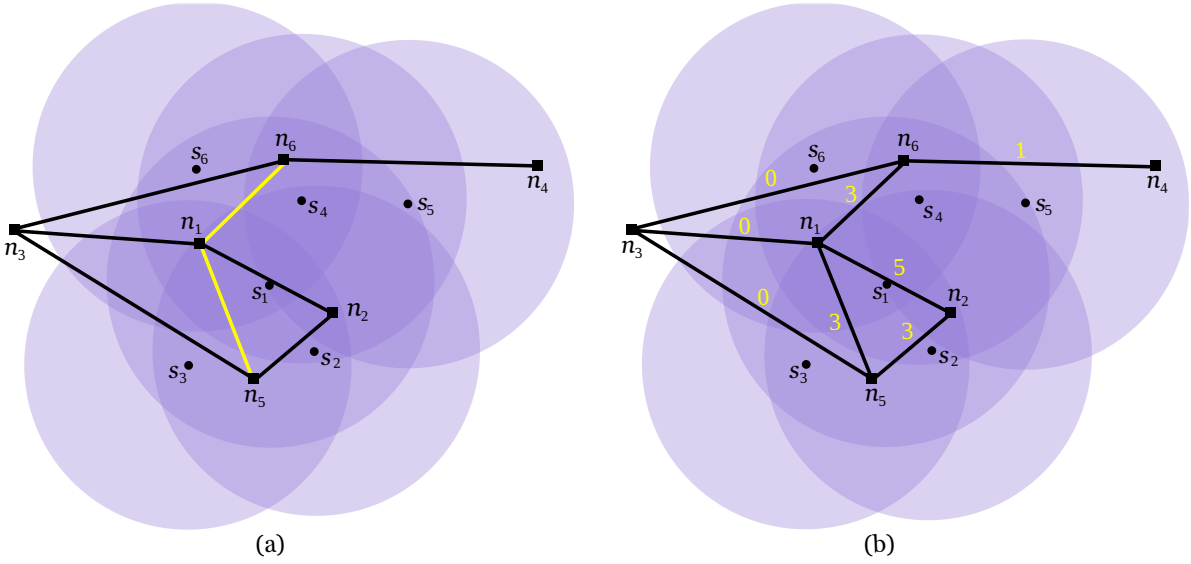


Figure 5.4: Nodes of graph G are represented by squares. (a) The yellow path connecting n_5 to n_6 is a 3-covered path on G. (b) Each edge e of G_w has weight $\mathrm{MC}_S(e)$.

In the following there is a description of the algorithm to calculate $\mathrm{MC}_S(P(n_i, n_j))$, as well as to find a maximum covered path between n_i and n_j. First graph $G = (N, E)$ is transformed into edge-weighted graph G_w by assigning the weight $w(e) = \mathrm{MC}_S(e)$ to each edge e of the graph (see Figure 5.4(b)). Calculating the weights of the edges of E has the same time complexity as calculating the maximum coverage of G. Therefore, and according to Proposition 5.1, such procedure takes $\mathcal{O}(|E| \times n \log n)$ time and $\mathcal{O}(|E| \times n)$ space. A k-path on G_w between two nodes of N can only traverse edges whose weight is greater than or equal to k. Consequently, a k-path between n_i and n_j that maximises the value of k is a path whose lightest edge has the maximum weight possible. If the problem was the opposite, that is, find a k-path connecting two nodes of G_w that minimises the weight of its heaviest edge, then it could be solved by computing a minimum-weight spanning tree (MST). In contrast, for the original problem there is the need to compute a maximum-weight spanning tree (MaxST). There is a well known algorithm to find such tree: invert the weights of the original graph and compute an MST of this new graph, which is in fact a MaxST of the original one. A MaxST of the weighted graph shown in Figure 5.4(b) is represented in Figure 5.5.

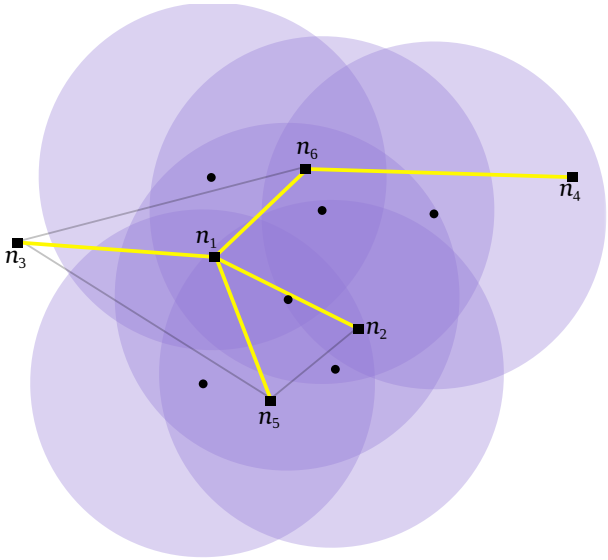


Figure 5.5: A maximum-weight spanning tree of G_w is represented by a solid yellow line.

If T_w is a MaxST of G_w, then it is known that a path on T_w between two of its nodes is unique. Let the weight of a path on G_w be determined by the weight of its lightest edge. The following proposition shows that the weight of a path on a MaxST of G_w is larger than or equal to the weight of any path on G_w that connects the same nodes.

Proposition 5.2 *Let G_w be an edge-weighted connected graph. For each path on G_w, assume the path's weight is given by the weight of its lightest edge. Then the path on a* MaxST *of G_w connecting any pair of nodes of G_w is a maximum weight path between such pair.*

Proof: Let G_w be an edge-weighted connected graph, n_i and n_j two of its nodes and T_w a maximum-weight spanning tree (MaxST) of G_w. Assume that $P(n_i, n_j)$ is the only path on T_w connecting nodes n_i to n_j and e is its lightest edge. Consequently, $P(n_i, n_j)$ has weight $w(e)$. Now suppose that path $P^*(n_i, n_j)$ on G_w is a maximum weight path between n_i and n_j. Its weight is given by e^*, its lightest edge, and so $w(e^*) > w(e)$. Since $P(n_i, n_j)$ is lighter than $P^*(n_i, n_j)$, the edge e is not an edge of $P^*(n_i, n_j)$. If paths $P(n_i, n_j)$ and $P^*(n_i, n_j)$ are united, then a cycle is created. Such cycle contains e, which clearly is its lightest edge. But this contradicts the hypothesis, since the lightest edge of a cycle in G_w cannot be an edge of a MaxST of G_w. Therefore, a maximum weight path between two nodes of G_w is the path on T_w connecting those nodes. □

A MaxST of G_w found by the algorithm described above can now be used to solve the original problem. The maximum $k \in \mathbb{N}$ such that there is a k-path between two nodes of G_w can be calculated as the weight of the only path that exists on a MaxST of G_w between those

nodes. What follows is therefore the core of the algorithm to calculate $\mathrm{MC}_{S,G_w}(P(n_i,n_j))$ and $P(n_i,n_j)$, taking advantage of a MaxST of G_w and of the Depth-First Search (DFS) algorithm [42].

ALGORITHM **Maximum Covered Path on a Graph**

INPUT: Graph G_w, nodes n_i and n_j
OUTPUT: $P(n_i,n_j)$, $\mathrm{MC}_{S,G_w}(P(n_i,n_j))$

1. Find T_w, a MaxST of G_w;
2. Compute $P(n_i,n_j)$ on T_w using the DFS algorithm [42];
3. $\mathrm{MC}_{S,G_w}(P(n_i,n_j)) \leftarrow \min\{w(e) : e \in P(n_i,n_j)\}$.

The following result states the temporal complexity of the previous algorithm.

Theorem 5.2 *Let S be a set of n antennas, G_w an edge-weighted planar graph with m edges and T_w a* MaxST *of G_w. Given two nodes n_i and n_j of G_w, $P(n_i,n_j)$ and $\mathrm{MC}_{S,G_w}(P(n_i,n_j))$ can be found on T_w in $\mathcal{O}(m)$ time and space.*

Proof: Constructing a MaxST of G_w by computing an MST of the same graph with inverted weights takes $\mathcal{O}(m)$ time using an algorithm by Matsui [60] since G_w is a planar graph. According to Proposition 5.2, the path connecting n_i and n_j on a MaxST of G_w, $T_w = (N,B)$, is a maximum-weight path connecting those nodes on G_w. Such path can be found by traversing T_w using the Depth-First Search technique [42], which takes $\mathcal{O}(|B|)$ time. Therefore, the weight of the path just computed on T_w between n_i and n_j is $\mathrm{MC}_{S,G_w}(P(n_i,n_j))$. Regarding the space complexity, graph G_w can be stored using $\mathcal{O}(m)$ space since it is a planar graph with m edges. The MaxST of G_w is a subgraph of G_w, so it can be stored in the same amount of space, as well as the maximum weight path found on such tree. □

5.1.4 Maximum Coverage of a Path on a Polygonal Region

As previously mentioned, the maximum coverage of a polygonal region was previously solved by Huang and Tseng [46]. A related but different problem is solved in the following: given a set S of antennas with transmission range $r \in \mathbb{R}^+$, a polygonal region R and two points p and q on R, calculate the maximum integer k such that a k-path connecting p and q exists within R. This problem can be solved using a similar idea to the one presented in the last section. To this end, there is the need to construct an edge-weighted graph. The arrangement of discs

of D divides the plane into regions of points that are covered by the same discs. Let D_R be the arrangement of discs confined to R, since that is the relevant part of the arrangement to this problem (see Figure 5.6).

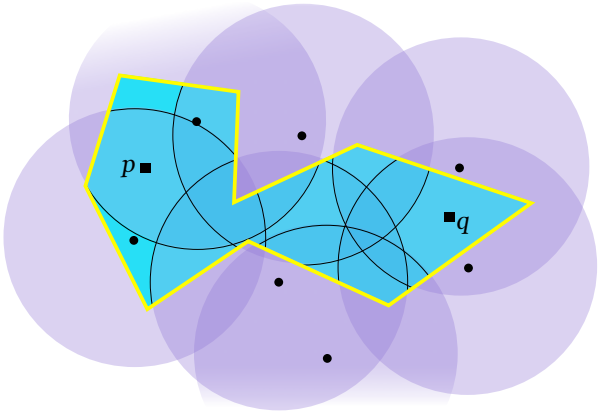


Figure 5.6: The arrangement D_R is defined by the discs enclosed by R and is shown in blue.

For each face of D_R, calculate its weight that is determined by the number of discs covering such face. Next construct the dual graph of D_R, denoted by D'_R, whose nodes are the faces of D_R and there is an edge between two of these nodes if the corresponding primal faces share an edge. Points p and q act as nodes of D'_R by representing the faces of D_R that contain them. Since each face of D_R is weighted, the dual nodes of D'_R are also weighted. Consequently, graph D'_R can be transformed into an edge-weighted graph by assigning each edge the minimum weight of the nodes it connects (see Figure 5.7). As a result, it is now possible to compute a MaxST of D'_R. Such tree helps to compute the maximum coverage of a path between p and q within R, which is denoted by $\mathrm{MC}_{S,R}(P(p,q))$.

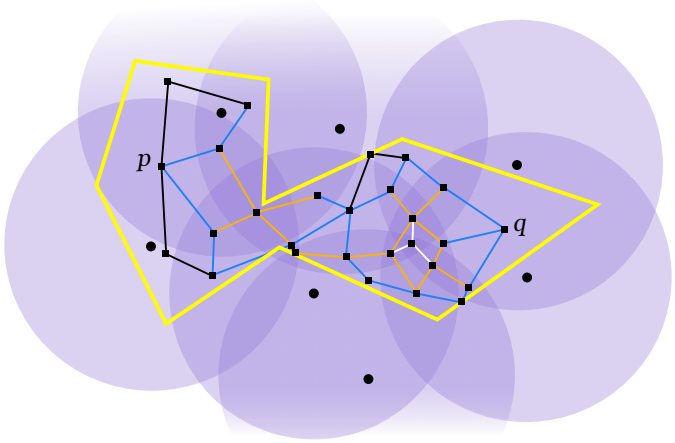


Figure 5.7: The nodes of the dual graph D'_R are represented by squares. Black edges weight 1, blue weight 2, orange weight 3 and white weight 4.

In the example in Figure 5.7, the maximum coverage of $P(p,q)$ is 2, therefore a k-path between p and q within R for $k > 2$ does not exist. Although the algorithm below does not compute a maximum covered path between p and q, it outputs a sequence of faces of D_R that can be traversed in order to construct such a path. In the following, assume that i and j are faces of D_R and i' and j' are the respective dual nodes of D'_R. Assume also that the weight of face i is represented by $w(i)$ and $w(\overline{i'j'})$ denotes the weight of the edge connecting the dual nodes i' and j'.

ALGORITHM **Maximum Coverage of a Path on a Region**

INPUT: Set S with range r, region R, points p and q
OUTPUT: $\mathrm{MC}_{S,R}(P(p,q))$

1. Construct D_R, the arrangement of D restricted to R;
2. For each face i of D_R do
   $w(i) \leftarrow |\{D(s_i,r) : i \in D(s_i,r), s_i \in S\}|$
3. Construct the graph D'_R, dual of D_R;
4. For every dual edge $\overline{i'j'} \in D'_R$ do
   $w(\overline{i'j'}) \leftarrow \min\{w(i), w(j)\}$
5. Compute T, a MaxST of D'_R and find $P(p,q)$ on T;
6. $\mathrm{MC}_{S,R}(P(p,q)) \leftarrow \min\{w(e) : e \in P(p,q)\}$.

Theorem 5.3 *Let S be a set of n antennas with transmission range $r \in \mathbb{R}^+$. Given two points p and q on a polygonal region R with m edges, the maximum coverage of a path between p and q within R can be calculated in $\mathcal{O}(n(n+m)\log(n+m))$ time.*

Proof: Each disc of D may intersect all the other discs of the same set, so the number of intersection points between them is $\mathcal{O}(n^2)$. Further, observe that each edge of R can only intersect each disc twice, so at most it intersects n discs. Therefore, R intersects D at most nm times. Consequently, the total number of intersection points of the arrangements involving D and R is $\mathcal{O}(n^2+nm)$. Arrangement D_R can be found by sweeping the plane, as long as the discs are segmented in a way that each piece is monotone. Such sweep takes $\mathcal{O}(n(n+m)\log(n+m))$ time and it can be used to calculate the weight of each face of D_R. Since D_R has $\mathcal{O}(n(n+m))$ faces and its dual graph is planar, the edge-weighted dual graph D'_R can be constructed in $\mathcal{O}(n(n+m))$ time, which is its number of edges. For the same reason and according to Theorem

5.2, calculating $\mathrm{MC}_{S,R}(P(p,q))$ using a MaxST of D'_R takes $\mathcal{O}(n(n+m))$ time. In conclusion, $\mathrm{MC}_{S,R}(P(p,q))$ can be calculated in $\mathcal{O}(n(n+m)\log(n+m))$ time. □

5.1.5 Maximum Coverage of a Path on the Plane

Let p and q be two points on the plane, the following algorithm is the last of this part of the chapter and it is proposed to calculate the maximum $k \in \mathbb{N}$ so that there is a k-path on the plane between p and q. Such value is denoted by $\mathrm{MC}_S(P(p,q))$. Since a path between p and q is not restricted to a graph or a polygonal region, this problem can be seen as a generalisation of the problems presented in Sections 5.1.3 and 5.1.4. The following strategy is very similar to the one discussed above and so a pseudo-code of the algorithm is omitted. First consider the arrangement of discs of D, where the weight of each face of the arrangement is determined by the number of discs that cover such face. Second, construct the dual D' of D: each face of D becomes a node of D' and there is an edge between two of these nodes if the corresponding primal faces share an edge. Consequently, every dual node has a weight associated. Graph D' is then transformed into an edge-weighted graph by assigning each edge the weight given by the minimum weight of the nodes it connects. In Figure 5.8 there is an example of an arrangement of discs of D and its dual graph D'. Third, find a maximum-weight spanning tree (MaxST) of D' and compute a path between p and q on that tree. Finally, assuming the weight of the path just computed is the weight of its lightest edge, $\mathrm{MC}_S(P(p,q))$ is precisely the weight of such path.

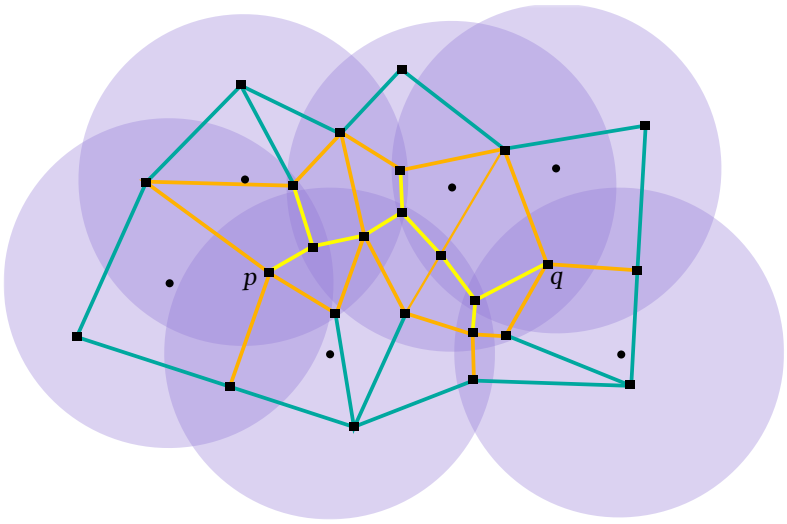


Figure 5.8: Arrangement of discs of D and its dual graph D'. The green edges of D' weight 1, the orange weight 2 and the yellow weight 3.

In the example in Figure 5.8, the weight of the maximum weight path between p and

q on the dual graph is 3, and so $\mathrm{MC}_S(P(p,q)) = 3$. The edges of D' supporting such path are represented by a solid yellow line in Figure 5.9. Neither the algorithm introduced in the previous subsection nor this one actually compute a maximum covered path between p and q. Notwithstanding this flaw, they both output a sequence of faces of the arrangement of discs that can be traversed in order to construct a maximum covered path between p and q. This subject is further discussed in Section 5.3.

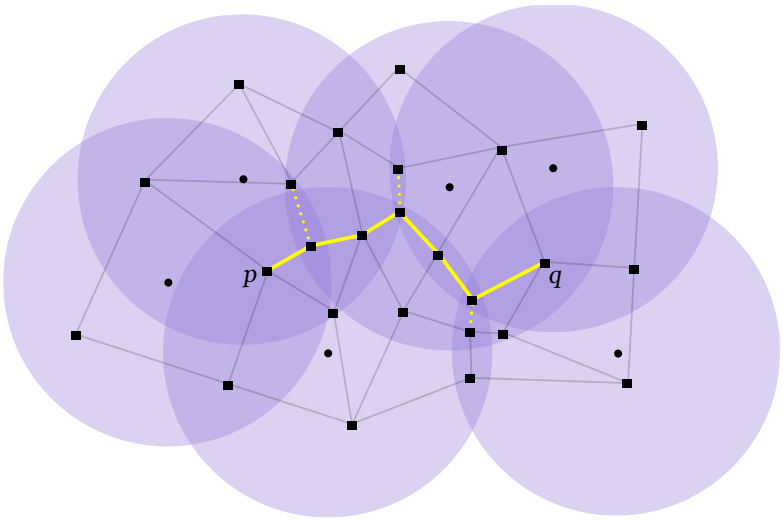


Figure 5.9: A 3-path between p and q is represented by a solid yellow line. The edges of D' that weight 3 but are not part of the path are represented by a dotted line.

The temporal complexity of the algorithm previously described is stated in the following result.

Theorem 5.4 *Let S be a set of n antennas with transmission range $r \in \mathbb{R}^+$ and p and q two points on the plane. The maximum $k \in \mathbb{N}$ so that a k-covered path between p and q exists can be calculated in $\mathcal{O}(n^2)$ time.*

Proof: Since set S is formed by n antennas with transmission range $r \in \mathbb{R}^+$, set D also has n discs. As a result, the arrangement of discs can be computed in $\mathcal{O}(n^2)$ time [31] because all discs have the same radius r. The weight of each face of the arrangement of discs can be calculated while the arrangement is being constructed. Therefore, the weights of the edges of D' are directly calculated as the minimum weight of their endpoints. Since the arrangement of discs may have a quadratic number of faces, the number of edges of D' may also add up to $\mathcal{O}(n^2)$ as they are both planar graphs. Consequently, computing a MaxST of D' and calculating $\mathrm{MC}_S(P(p,q))$ takes $\mathcal{O}(n^2)$ time, as stated by Theorem 5.2. In conclusion, the maximum $k \in \mathbb{N}$ so that a k-covered path between p and q exists can be calculated in $\mathcal{O}(n^2)$

time. □

5.2 Coverage Restricted to an Angle

5.2.1 Introduction

The second part of this chapter is devoted to a variation of 2-coverage. As previously discussed, a point q is 2-covered by a set S of n antennas if q is within range of at least two antennas. The concept introduced in the following is more restrictive than 2-coverage. A point q is said to be α-*covered* if for two antennas of S, s_i and s_j, the angle $\measuredangle(s_i, q, s_j) \geq \alpha$. In other words, point q is α-covered if it sees $\overline{s_i s_j}$ with an angle of at least α. This definition ensures that the antennas surround their service area uniformly, instead of favouring a particular side. Furthermore, this definition can also be used to localise an object within a terrain, where two sensors are needed to determine the object's position. Observe that the object and the two sensors should not be nearly collinear as this translates into positioning errors. In the discussion that follows, quality parameter α is assumed to be constant and equal to $\frac{\pi}{2}$ (see Figure 5.10(a)). Therefore, $\frac{\pi}{2}$-coverage will be referred to simply as coverage throughout this section.

Figure 5.10: (a) Point q is $\frac{\pi}{2}$-covered because $\measuredangle(s_1, q, s_2) \geq \frac{\pi}{2}$. (b) Blue wedge passing through s_1 and s_2 whose apex is located at the boundary of $D_d(s_1, s_2)$ has a right angle.

Given two antennas s_i and s_j, let $D_d(s_i, s_j)$ denote the disc of diameter $\overline{s_i s_j}$. It is known that any wedge passing through s_i and s_j whose apex is located at the boundary of $D_d(s_i, s_j)$ has a right angle (see Figure 5.10(b)). Furthermore, any wedge passing through s_i and s_j whose apex is inside $D_d(s_i, s_j)$ has an angle greater than $\frac{\pi}{2}$. As a result, the area covered by two antennas s_i and s_j is bounded by $D_d(s_i, s_j)$. A set of antennas that covers q is called a *coverage set* for q. Using the same notation that was introduced in Chapter 4, let $\mathrm{MR}_S(q)$ denote the minimum power transmission range of S that covers point q. Assume that the

antennas of $S = \{s_1, \ldots, s_n\}$ are sorted in ascending order of their distance to q; that is, s_1 is the closest antenna to q, s_2 is the second closest antenna to q and so on. Now suppose $\{s_1, \ldots, s_i\}$ is a coverage set for q whilst $\{s_1, \ldots, s_{i-1}\}$ is not. This supposition implies that s_i is the closest antenna to q that together with the other antennas of S that are closer to q than s_i form a coverage set for q. Consequently, $\mathrm{MR}_S(q) = d(s_i, q)$.

Efrat et al. [38] studied how to minimise the number of sensors required to have every point on a given region R covered by two sensors at angle α. That is, each point on R is separated from the sensors with an angle within the range $[\alpha, \pi - \alpha]$. They assume that the set of sensors exists within a simple polygon P, which in turn contains region R, and find a small subset of sensors on P that α-covers R. Contrary to their work, the problems presented in this section do not aim to minimise the number of sensors but to minimise the devices' range while keeping a given object $\frac{\pi}{2}$-covered.

The problems presented in the following are devoted to $\frac{\pi}{2}$-coverage. Section 5.2.2 shows how to minimise the antennas' range so that a given point on the plane is covered. The locus of all points covered by S and its boundary are constructed in Section 5.2.3. Some properties of both structures are also established in that section. The variation of Embracing Voronoi diagrams associated with this type of coverage, which is called the Coverage Voronoi diagram, is introduced in Section 5.2.4 where an algorithm to construct it is also proposed. This subject is concluded in Section 5.3 with a brief summary of the presented algorithms and future research.

5.2.2 Minimum Transmission Range to Cover a Point

Let S be a set of n antennas and q a point on the plane. To calculate $\mathrm{MR}_S(q)$, there is the need to locate the closest antenna $s_c \in S$ to q that together with some other antenna s_j, such that $d(s_j, q) < d(s_c, q)$, form a coverage set for q. The first reasonable idea to find such an antenna is to search the smallest diametral disc containing q and verify if s_c defines its diameter. Unfortunately, this is not true. As Figure 5.11(a) illustrates, the smallest disc containing q is $D_d(s_1, s_3)$ (shown in blue) but $\mathrm{MR}_S(q)$ is given by the distance between q and s_2. Moreover, it is not even true that the closest antenna to q is part of a coverage set for q (see antenna s_1 in Figure 5.11(b)). Therefore, the straightforward usefulness of the closest antenna to q or of the smallest diametral disc containing q is not clear when it comes to calculate $\mathrm{MR}_S(q)$. However, applying a similar strategy to the one used in the "Minimum Embracing Range III" algorithm (presented in Section 2.5.1 in Chapter 2) solves this problem, as it is shown in the following pseudo-code.

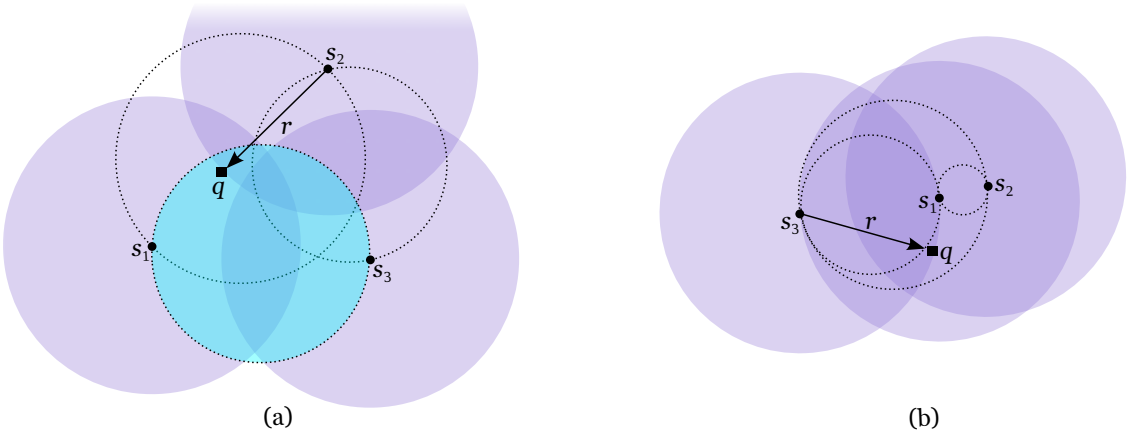


Figure 5.11: (a) The smallest disc containing q is $D_d(s_1, s_3)$ (shown in blue) and $\mathrm{MR}_S(q) = d(s_2, q) = r$. (b) The closest antenna to q is s_1, which is not part of any coverage set for q, $\mathrm{MR}_S(q) = d(s_3, q) = r$.

ALGORITHM **Minimum Coverage of a Point**

INPUT: Set S of n antennas, point q
OUTPUT: $\mathrm{MR}_S(q)$

1. Divide the lights of S into four quadrants with origin at q;
2. $R \leftarrow \{d(s_i, q) : s_i \in S\}$;
3. Perform a binary search on $R = \{r_1, \ldots, r_n\}$:

   For the median range $r_i \in R$ do

   $S' \leftarrow \{s_i \in S : d(s_i, q) \leq r_i\}$

   If q is covered by S'

   Then proceed the search on $R \leftarrow \{r_j \in R : r_j \leq r_i\}$

   Otherwise proceed the search on $R \leftarrow \{r_j \in R : r_j > r_i\}$
4. The final range is $\mathrm{MR}_S(q)$.

To decide if point q is covered by a set of antennas S' there is the need to study how the antennas are distributed through the quadrants. If there is an antenna on every quadrant, then q is clearly covered by S'. If there are precisely two empty quadrants which are opposite to each other, then q is covered by S' because a diametral disc defined by an antenna on each quadrant contains q (see Figure 5.12(a)). On the other hand, if the two empty quadrants are adjacent, then there is the need to calculate the angle β between q and the outermost antennas

the non-empty quadrants. If $\beta \geq \frac{\pi}{2}$ then q is covered by S', otherwise q is not covered. The first case is illustrated in Figure 5.12(b) and the second in Figure 5.12(c).

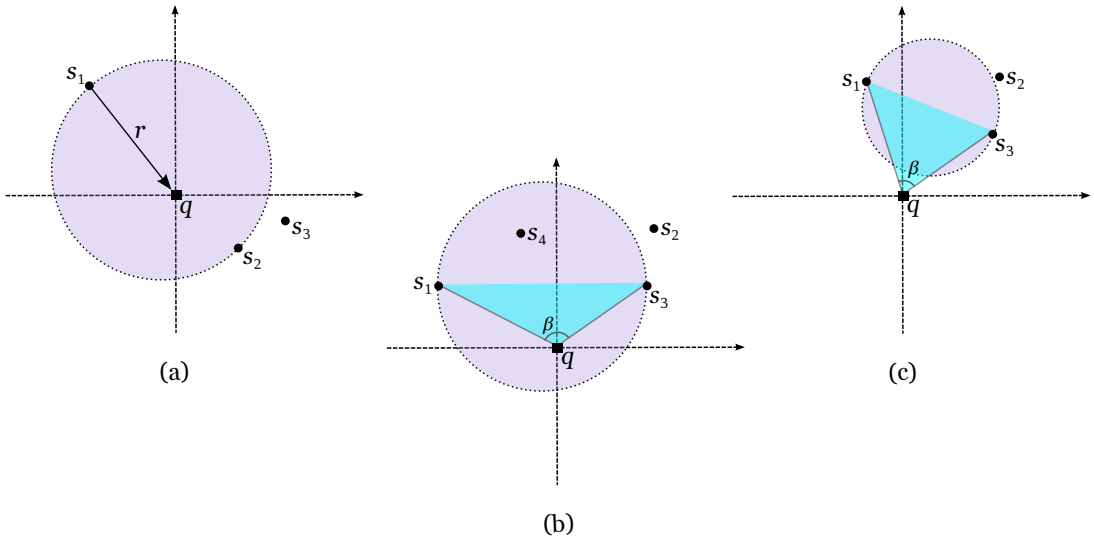


Figure 5.12: (a) Point q is covered and $\text{MR}_S(q) = d(s_1, q) = r$. (b) Point q is covered since $\beta \geq \frac{\pi}{2}$, and therefore $q \in D_d(s_1, s_3)$. (c) Point q is not covered since $\beta < \frac{\pi}{2}$.

The following theorem states the complexity of the algorithm proposed above.

Theorem 5.5 *Let S be a set of n antennas and q a point on the plane. The minimum transmission range of S that covers q can be calculated in $\mathcal{O}(n)$ time and space.*

Proof: Since S is formed by n antennas, dividing the antennas into quadrants and find set $R = \{d(s_i, q) : s_i \in S\}$ can be done in linear time. Calculating the median of the distances of R also takes linear time [25]. In each step of the binary search there is the need to decide if a given set $S' \subseteq S$ is a coverage set for q. While the binary search is being performed on the lowest half of R, every antenna of S' is studied. However, once the search switches to the highest half, the antennas on the lowest will not be studied again since the outermost antennas of the previously failed verification are saved to the next step. Therefore, the binary search is performed in $\mathcal{O}\left(n + \sum_{i=1}^{log_2(n)} \frac{n}{2^i}\right) = \mathcal{O}(n)$ time. Consequently, the minimum transmission range of S that covers q can be calculated in linear time. Regarding space complexity, the list R of distances needs $\mathcal{O}(n)$ space to be stored, as well as set S. □

Observe that all distances on list R are candidates to be the minimum transmission range of S that covers q. In the worst case, the "Minimum Coverage of a Point" algorithm has to analyse every antenna of S. Therefore, a lower bound for this algorithm is $\Omega(n)$ time, which combined with the previous theorem makes the linear complexity of this algorithm optimal.

5.2.3 Covered Region and its Contour

As previously mentioned, given two antennas s_i and s_j of S, the diametral disc $D_d(s_i, s_j)$ bounds the points on the plane that are covered by both antennas. Therefore, it is not hard to realise that the region on the plane that is covered by S is enclosed by the union of diametral discs defined by each pair of antennas of S. This observation leads to the following result.

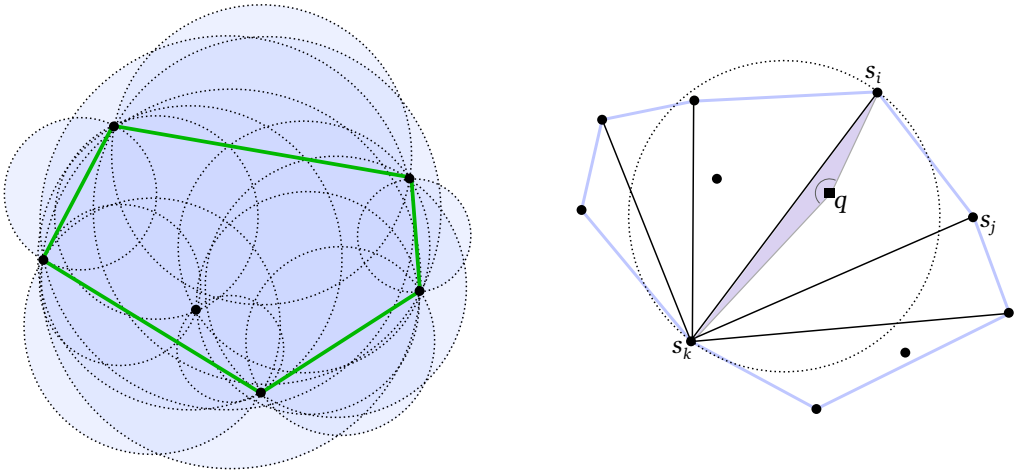
Proposition 5.3 *Given a set S of n antennas, the region on the plane that is covered by S can be found in $\mathcal{O}(n^3 \log n)$ time.*

Proof: As set S is formed by n antennas, there are $\mathcal{O}(n^2)$ pairs of antennas each defining a distinct diametral disc. Since each of these discs can intersect all the others, there are up to $\mathcal{O}(n^3)$ intersection points in this arrangement. Consequently, sweeping the plane to compute the arrangement of diametral discs takes $\mathcal{O}(n^3 \log n)$ time. As previously mentioned, this arrangement of discs encloses the region on the plane that is covered by S. □

In Figure 5.13(a) there is an example of a region covered by six antennas. Such region is shown in blue and the diametral discs defined by each pair of antennas are represented by a dotted line. Also clear from that image is the fact that the region covered by S contains the convex hull of S, which is represented by a solid green line. This is not a coincidence as the following lemma shows.

Lemma 5.1 *Given a set S of antennas, the region on the plane covered by S, defined by the union of all the diametral discs between every pair of antennas, contains the convex hull of S.*

Proof: Points on the boundary of CH(S) are clearly contained in the diametral discs formed by the antennas that define such edges. Let q be point inside CH(S) and assume that the convex hull is triangulated (see Figure 5.13(b)). Suppose q is a point on triangle $\triangle(s_i, s_j, s_k)$. If $\triangle(s_i, s_j, s_k)$ is an obtuse triangle, meaning it has an angle larger than $\frac{\pi}{2}$, then q is inside the diametral disc formed by the antennas located at the acute vertices. This is straightforward to see since the obtuse angle is also inside such diametral disc (otherwise the angle would be smaller than $\frac{\pi}{2}$). If $\triangle(s_i, s_j, s_k)$ is an acute or right triangle, then q is inside the diametral disc formed by the antennas that define the closer edge of the triangle to q. Let such antennas be s_i and s_k (see Figure 5.13(b)), then this holds because $\measuredangle(s_i, q, s_k) > \frac{\pi}{2}$. Since point q is always inside some diametral disc defined by two antennas of S and can be any point on CH(S), CH(S) is contained in the union of these diametral discs. In other words, CH(S) is contained in the region covered by S and this is precisely the assertion of the lemma. □



(a) (b)

Figure 5.13: Set S is represented by dots. (a) The region covered by S is shown in blue, the diametral discs in a dotted line and the convex hull of S in a solid green line. (b) Point q belongs to $\triangle(s_i, s_j, s_k)$ that is part of the triangulation of $\mathrm{CH}(S)$, $\measuredangle(s_k, q, s_i) > \frac{\pi}{2}$.

Note that the boundary of the union of the diametral discs defined by every pair of antennas of S encloses the points on the plane that are covered by S. As previously mentioned in Section 2.5.3 in Chapter 2, the boundary of this type of regions, so called depth contours, is an appropriate tool for data visualisation. What follows is therefore a method to construct the contour of the region covered by S, which is denoted by $\mathrm{C}(S)$. It is straightforward to see that the diametral discs formed by antennas that are also vertices of $\mathrm{CH}(S)$ are part of this contour (see Figure 5.13(a)). In fact, these antennas are the key to construct $\mathrm{C}(S)$ as stated in the following theorem.

Lemma 5.2 *Only the antennas of S that are vertices of $\mathrm{CH}(S)$ are relevant to construct the contour of the region covered by S.*

Proof: Let s_i be an antenna of S inside $\mathrm{CH}(S)$ and $\triangle(s_j, s_k, s_l)$ a triangle formed by three other antennas such that s_i is inside $\triangle(s_j, s_k, s_l)$ and s_j, s_k and s_l are vertices of $\mathrm{CH}(S)$. Given the set $D_\triangle = \{D_d(s_j, s_k), D_d(s_j, s_l), D_d(s_k, s_l)\}$, in the following it is proven that any diametral disc defined by s_i and some other antenna of S is contained in $D_\triangle$. Consider the lines tangent to $D_d(s_i, s_j)$ at s_j and $D_d(s_i, s_l)$ at s_l (see Figure 5.14(a)). Disc $D_d(s_i, s_j)$ only is part of the contour of the region covered by S if the tangent line at s_j is not inside $D_d(s_j, s_k)$ or $D_d(s_j, s_l)$ in some neighbourhood of s_j. On the same basis, $D_d(s_i, s_l)$ only is part of $\mathrm{C}(S)$ if the tangent line at s_l is not inside $D_d(s_l, s_k)$ or $D_d(s_l, s_j)$ in some neighbourhood of s_l. However, this only happens if s_i is outside $\triangle(s_j, s_k, s_l)$ (see Figure 5.14(b)). The tangent line

at s_j is perpendicular to the edge $\overline{s_j s_l}$ if s_i is a point on that edge. If s_i is outside $\triangle(s_j, s_k, s_l)$, then the tangent line at s_j is not inside $D_d(s_j, s_k)$ or $D_d(s_j, s_l)$ in some neighbourhood of s_j, meaning that $D_d(s_i, s_j)$ and $D_d(s_i, s_l)$ are part of the contour. However, this is impossible since s_i is assumed to be inside $\mathrm{CH}(S)$. Therefore, the antennas inside $\mathrm{CH}(S)$ are not relevant to construct the contour of the region covered by S. □

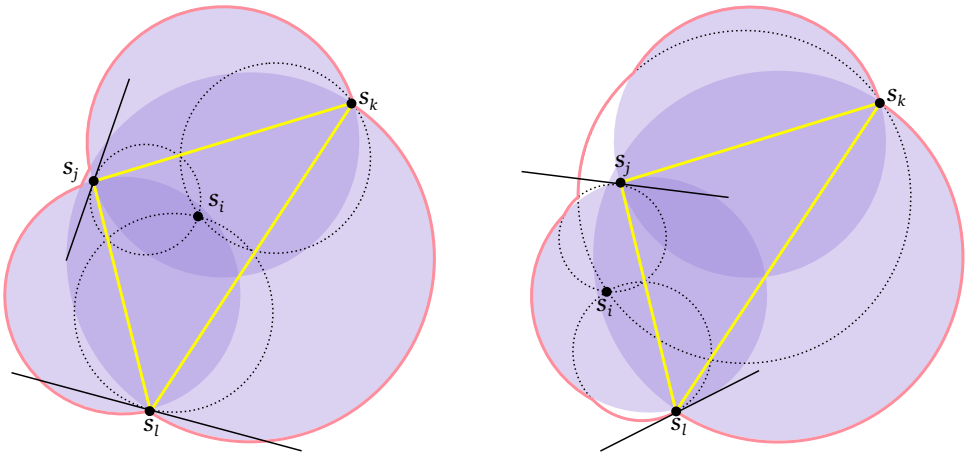


Figure 5.14: (a) None of the diametral discs defined by s_i appear on the pink contour. (b) Discs $D_d(s_i, s_j)$ and $D_d(s_i, s_l)$ are part of the pink contour since $s_i \notin \triangle(s_j, s_k, s_l)$.

The following lemma paves the way to start the construction of $\mathrm{C}(S)$ by proving that a piece of the diametral disc defined by the leftmost and topmost vertices of $\mathrm{CH}(S)$ is an arc of $\mathrm{C}(S)$.

Lemma 5.3 *Let s_i and s_j of S be the leftmost and topmost vertices of $\mathrm{CH}(S)$, respectively. A piece of $D_d(s_i, s_j)$ is an arc of the contour of the region covered by S.*

Proof: Suppose $D_d(s_i, s_j)$ is not part of $\mathrm{C}(S)$, that is, every point on $D_d(s_i, s_j)$ is contained in other diametral discs defined by antennas of S. Consider the vertical line l_i that passes through s_i and the horizontal line l_j that passes through s_j. Let q be the intersection point between l_i and l_j. Point q is on $D_d(s_i, s_j)$ because $\measuredangle(s_i, q, s_j)$ is a right angle. If q was inside another diametral disc, for example, $D_d(s_k, s_l)$ then $\measuredangle(s_k, q, s_l) > \frac{\pi}{2}$. But that is impossible since it implies that either s_l is on the left of l_i or s_k is above l_j. Therefore, q is a point of $\mathrm{C}(S)$, as well as some of its neighbouring points on $D_d(s_i, s_j)$. □

What follows is therefore an algorithm to construct $\mathrm{C}(S)$, starting with the diametral disc defined by the leftmost and topmost vertices of $\mathrm{CH}(S)$. The algorithm works with two pointers, p and q, that always point at vertices of $\mathrm{CH}(S)$. Since computing $\mathrm{CH}(S)$ induces a

convex ordering of the antennas of S, function $p_n \leftarrow next(p)$ points p_n at the next vertex of $\mathrm{CH}(S)$ in that same order. A ray from s_i to infinity passing through s_j is denoted by $\overrightarrow{s_i s_j}$ and the boundary of a diametral semicircle defined by s_i and s_j is denoted by $\mathrm{SC}(s_i, s_j)$. Let Q be a queue that only stores two points and $top(Q)$ and $bottom(Q)$ the functions that retrieve the element at the top of Q and bottom of Q, respectively. Function $insert(i, Q)$ inserts point i at the top of Q and $remove(bottom(Q))$ removes $bottom(Q)$ from Q.

ALGORITHM **Covered Region Contour**

INPUT: Set S of antennas

OUTPUT: $\mathrm{C}(S)$, the contour of the region covered by S

1. Compute $\mathrm{CH}(S)$, the convex hull of S;
2. Let p and q point at the leftmost and topmost vertices of $\mathrm{CH}(S)$, respectively;
3. Initialise $p_n \leftarrow next(p)$, $q_n \leftarrow next(q)$, $\mathrm{C}(S) \leftarrow \emptyset$ and $Q \leftarrow \emptyset$;
4. While an arc on $\mathrm{SC}(p, q) \nsubseteq \mathrm{C}(S)$ do
   (a) $I_p \leftarrow \mathrm{SC}(p, q) \cap \overrightarrow{pp_n}$, $I_q \leftarrow \mathrm{SC}(p, q) \cap \overrightarrow{q_n q}$
   (b) If $I_p = I_q = \{i\}$ then
   $insert(i, Q)$
   $p \leftarrow p_n$, $p_n \leftarrow next(p)$, $q \leftarrow q_n$, $q_n \leftarrow next(q)$
   Else choose the first intersection $i \in \{I_p \cup I_q\}$ from p to q
   If $i \in I_p$ then
   $insert(i, Q)$, $p \leftarrow p_n$, $p_n \leftarrow next(p)$
   Else
   $insert(i, Q)$, $q \leftarrow q_n$, $q_n \leftarrow next(q)$
   (c) If $|Q| > 1$ then
   $\mathrm{C}(S) \leftarrow \mathrm{C}(S) \cup \{\text{arc on } \mathrm{SC}(p, q) \text{ from } bottom(Q) \text{ to } top(Q)\}$
   $remove(bottom(Q))$

In Figures 5.15 and 5.16 there is a step by step example of the previous algorithm constructing the contour of a region covered by six antennas. Pointers p and q are initialised by pointing at the leftmost and topmost vertices of $\mathrm{CH}(S)$, respectively (see Figure 5.15(a)). The algorithm starts at these particular points since, according to Lemma 5.3, a piece of the diametral disc defined by p and q is part of $\mathrm{C}(S)$. There are two intersection points between

$SC(p, q)$ and rays $\overrightarrow{pp_n}$ and $\overrightarrow{q_n q}$. Since the first intersection (point A) is found on ray $\overrightarrow{q_n q}$, pointer q (followed by q_n) moves to the next vertex of $CH(S)$. Figures 5.15(b) and 5.15(c) show the construction of the first piece of $C(S)$. In Figure 5.15(d) there is an example of what happens when both rays have the same intersection point. In that case, the four pointers move to the next vertex of $CH(S)$. The rest of the algorithm's steps until $C(S)$ is completely constructed are shown in Figure 5.16.

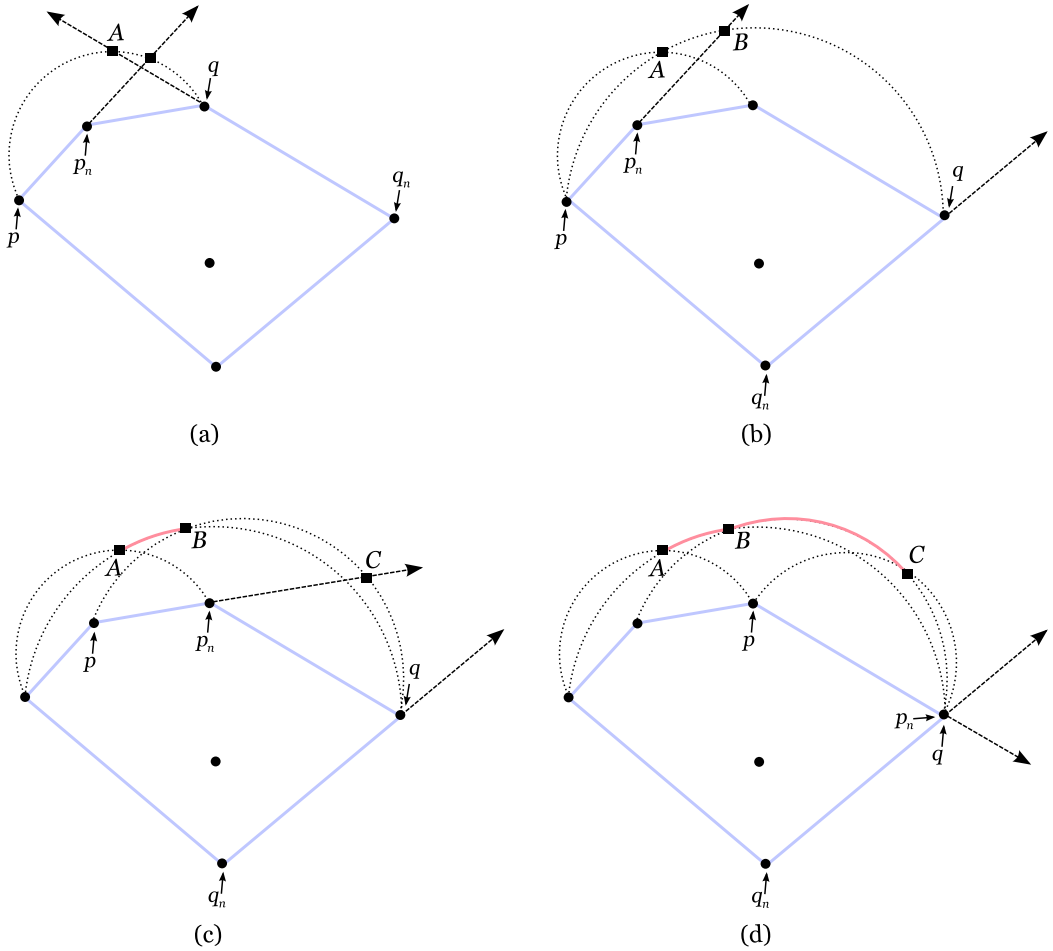


Figure 5.15: (a) Point A is the first intersection point between the rays and $SC(p, q)$. (b) The arc on $SC(p, q)$ from A to B belongs to $C(S)$. (c) The arc on $SC(p, q)$ from B to C belongs to $C(S)$. (d) Both rays have the same intersection point, which is a vertex of $C(S)$. Then each pointer moves to the next vertex.

The following proposition states that the union of arcs constructed by the algorithm described above is indeed the contour of the region covered by S.

Proposition 5.4 *Given a set S of antennas on the plane, the union of arcs constructed by the previous algorithm is the contour of the region covered by S.*

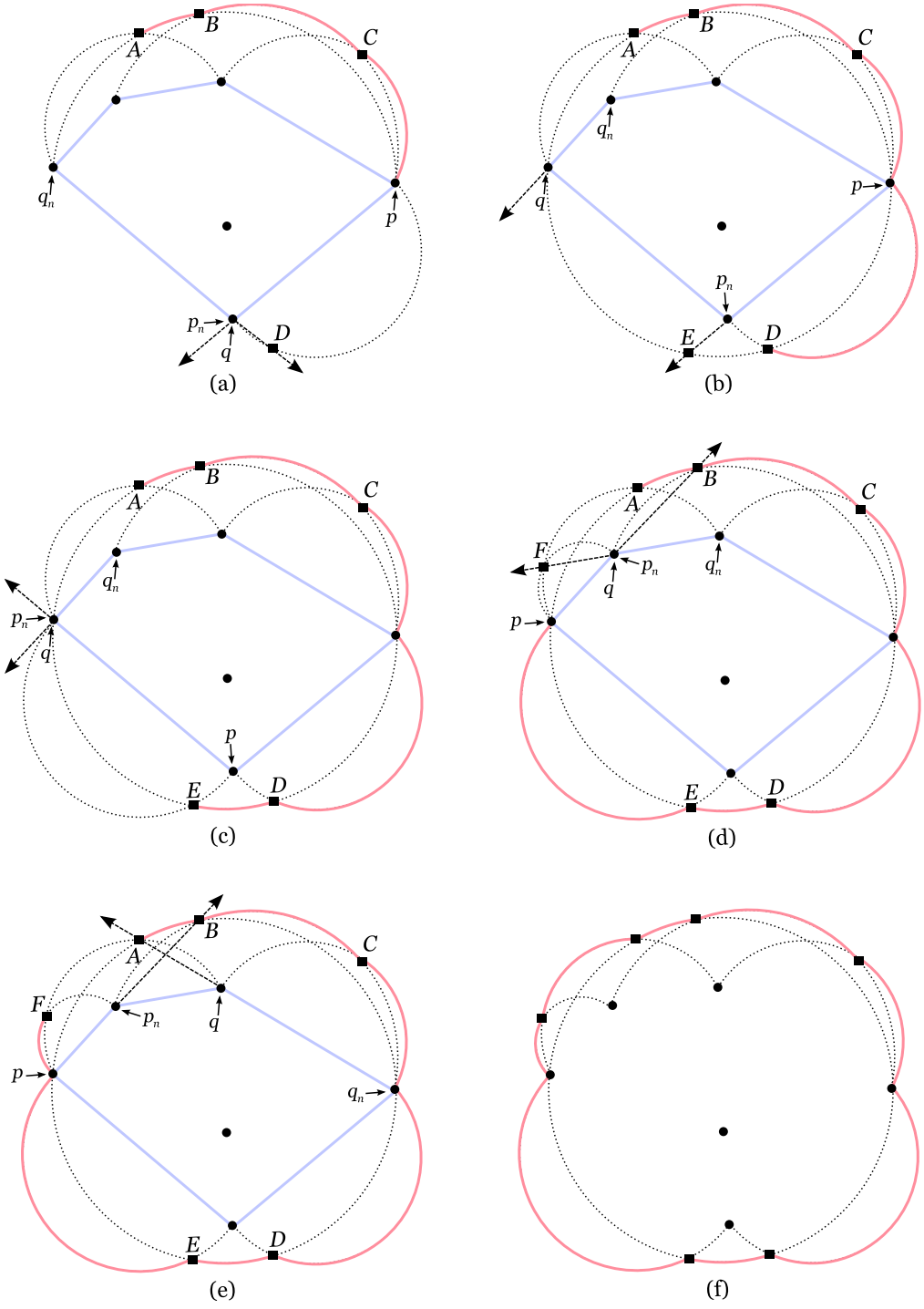


Figure 5.16: The antennas of S are represented by dots and C(S) by a solid pink line. Last steps to construct the contour of the region covered by S, which is complete in the last image.

Proof: A point i on the boundary of a semicircle defined by antennas s_j and s_k, SC(s_j, s_k), is a point of C(S) if $\measuredangle(s_j, i, s_k)$ is a right angle. Moreover, all the other antennas of S have to be inside the wedge formed by the angle $\measuredangle(s_j, i, s_k)$. This is a necessary condition since it

ensures that i is not inside any other diametral disc formed by the rest of the antennas (see Figure 5.17(a)). Therefore, the neighbouring points of i outside $\mathrm{SC}(s_j, s_k)$ are not covered by S, contrary to what happens to the ones that are inside such semicircle. Consequently, i is a point of $\mathrm{C}(S)$.

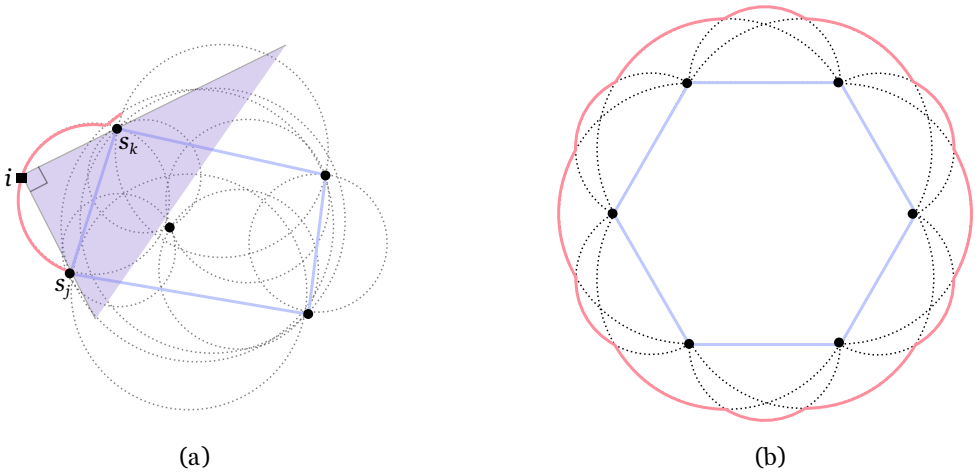


Figure 5.17: (a) Point i is a point of $\mathrm{C}(S)$ since all the antennas of $S \backslash \{s_j, s_k\}$ are inside the purple wedge defined by the right angle $\measuredangle(s_j, i, s_k)$. (b) The pink contour of the region covered by six antennas arranged to form an hexagon is the union of twelve arcs.

According to Lemma 5.3, the first semicircle considered, $\mathrm{SC}(p, q)$, has an arc that is part of $\mathrm{C}(S)$. That arc ends at the first intersection point i between $\mathrm{SC}(p, q)$ and $\overrightarrow{pp_n}$ or $\overrightarrow{q_n q}$. If point i is found on $\overrightarrow{q_n q}$, that means the angles $\measuredangle(p, i, q)$ and $\measuredangle(p, i, q_n)$ are both right angles (see Figure 5.16(d)). Therefore, the vertex pointed by q_n is on the wedge formed by the angle $\measuredangle(p, i, q)$. Consequently, the points on $\mathrm{SC}(p, q)$ after i no longer belong to $\mathrm{C}(S)$ because the angles between each of these points and pointers p and q_n are larger than $\frac{\pi}{2}$. The algorithm handles this situation by moving pointers q and q_n to the following vertices of $\mathrm{CH}(S)$ (see Figure 5.16(e)). As a result, $\mathrm{SC}(p, q)$ is updated and the points along this new arc are part of $\mathrm{C}(S)$ from i to the next intersection with $\overrightarrow{pp_n}$ or $\overrightarrow{q_n q}$. If the intersection i is an intersection point between $\mathrm{SC}(p, q)$ and $\overrightarrow{pp_n}$, then $\measuredangle(p, i, q)$ and $\measuredangle(p_n, i, q)$ are both right angles. Exactly like the previous situation, these pointers are actualised so that $\mathrm{SC}(p, q)$ is updated to $\mathrm{SC}(p_n, q)$. A piece of the latter is then the new arc of $\mathrm{C}(S)$. In the case the intersection point i occurs between $\mathrm{SC}(p, q)$ and both rays (see Figures 5.15(d) and 5.16(c)), then i is a vertex of $\mathrm{C}(S)$. The wedge formed by the right angle $\measuredangle(p, i, q)$ contains all the antennas of S and so it is a point of $\mathrm{C}(S)$. Since there are no more points to follow on that semicircle, each pointer has to move to the next vertex of $\mathrm{CH}(S)$. The new arc starts on vertex i and is part of $\mathrm{C}(S)$ until the next intersection with $\overrightarrow{pp_n}$ or $\overrightarrow{q_n q}$ is found.

The algorithm halts when the current semicircle has been previously analysed, that is, a piece of such semicircle is already part of $\mathrm{C}(S)$. In conclusion, the union of arcs constructed by the previous algorithm is the contour of the region covered by S. □

The following proposition determines the complexity of $\mathrm{C}(S)$.

Proposition 5.5 *Given a set S of n antennas on the plane, assume that $\mathrm{CH}(S)$ has $m \leq n$ vertices. The contour of the region covered by S consists of $2m$ arcs at most.*

Proof: Let $\mathrm{CH}(S)$ be the convex hull of S formed by $m \leq n$ vertices. Pointers p and q start pointing at $\mathrm{SC}(p, q)$ but the first arc that is really part of $\mathrm{C}(S)$ is the one considered by the algorithm in the second iteration. Besides the endpoints of these two arcs, the pointers never stop twice at the same vertices of $\mathrm{CH}(S)$ because the algorithm halts when an arc that is already part of $\mathrm{C}(S)$ is found. Once the algorithm is finished, every vertex of $\mathrm{CH}(S)$ has been analysed. Since the pointers cannot move backwards and q is always ahead of p, each pointer stops at most m times before reaching the first arc of the contour. In conclusion, both pointers stop at most $2m$ times, which means that $\mathrm{C}(S)$ consists of $2m$ arcs at most. □

This proposition proves that $\mathrm{C}(S)$ has linear complexity. Note that $\mathrm{C}(S)$ is only formed by a number of arcs that is twice the number of vertices of $\mathrm{CH}(S)$ if the pointers on the algorithm never move simultaneously. For example, this situation occurs when the antennas of S are in convex position since all of them contribute to $\mathrm{C}(S)$. Figure 5.17(b) illustrates this with six antennas arranged to form an hexagon. Therefore, the contour of the region covered by them is the union of twelve arcs. The complexity of the algorithm that constructs this contour is analysed in the following.

Theorem 5.6 *Given a set S of n antennas on the plane, the contour of the region covered by S can be constructed in $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space.*

Proof: Constructing the convex hull of S, $\mathrm{CH}(S)$, takes $\mathcal{O}(n \log n)$ time and induces a convex ordering of the antennas of S that are simultaneously vertices of $\mathrm{CH}(S)$. Therefore, each pointer moves to the next vertex of $\mathrm{CH}(S)$ in constant time. The intersection points between the current semicircle $\mathrm{SC}(p, q)$ and rays $\overrightarrow{pp_n}$ and $\overrightarrow{q_n q}$ are also computed in constant time since each ray only intersects $\mathrm{SC}(p, q)$ once. According to Proposition 5.5, $\mathrm{C}(S)$ is the union of a linear number of arcs. Since each arc of the contour can be found in constant time, the whole contour can be constructed in $\mathcal{O}(n)$ time. Regarding space complexity, there is the need to store the antennas of S, the ordering of the antennas that form the convex hull and the arcs of $\mathrm{C}(S)$, which can all be done using $\mathcal{O}(n)$ space. □

Corollary 5.1 *Let S' be the subset of the antennas of S that are also vertices of* $\mathrm{CH}(S)$. *Then for every antenna $s_i \in S'$ there is a piece of* $\mathrm{SC}(s_i, s_j)$, *for some antenna $s_j \in S'$, that is part of* $\mathrm{C}(S)$.

Proof: Whenever pointers p and q stop, a piece of $\mathrm{SC}(p, q)$ is used to construct $\mathrm{C}(S)$. According to the algorithm's construction of $\mathrm{C}(S)$, pointers p and q stop at every antenna of S'. Therefore, for every $s_i \in S'$ there is a piece of $\mathrm{SC}(s_i, s_j)$, for some antenna $s_j \in S'$, that is part of $\mathrm{C}(S)$. □

5.2.4 The Coverage Voronoi Diagram

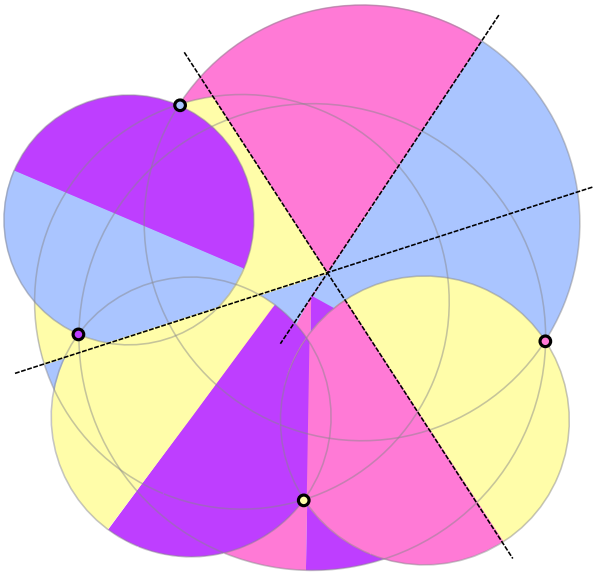


Figure 5.18: Coverage Voronoi diagram of four antennas (blue, purple, yellow and pink). Some of the perpendicular bisectors between the antennas are represented by a dashed line.

The last subject discussed in this section merges coverage and Embracing Voronoi diagrams, which were discussed in Chapter 3. Such structure is called the *Coverage Voronoi diagram* and is introduced below. Following the line of reasoning of Chapter 3, the Coverage Voronoi diagram is a structure that helps to efficiently recalculate the antennas' minimum range to keep any moving points covered. As before, let S be a set of n antennas on the plane. The minimum transmission range to cover a point q, $\mathrm{MR}_S(q)$, is given by the distance between an antenna $s_c \in S$ and q. Therefore, such antenna s_c is the closest antenna to q so that q is inside $D_d(s_c, s_j)$ for some antenna $s_j \in S$ that is closer to q than s_c. Now consider the partition of the plane into sets of points that share their closest antenna according to the previous conditions. The resulting diagram can be seen in Figure 5.18. The Coverage

Voronoi diagram can be constructed using the following algorithm, which takes advantage of the *Minimum Coverage of a Point* algorithm presented in Section 5.2.2.

Algorithm **Coverage Voronoi diagram**

Input: Set S of n antennas
Output: The Coverage Voronoi diagram of S

1. Compute all the perpendicular bisectors between the antennas;
2. $D_d \leftarrow \{D_d(s_i, s_j) : s_i \neq s_j \in S\}$;
3. For each face f of arrangement of discs D_d do
   (a) Intersect f with the perpendicular bisectors between the antennas whose diametral discs cover f
   (b) For each new subdivision of f do
      i. Select an interior random point q
      ii. Find the antenna $s_c \in S$ defining $\mathrm{MR}_S(q)$ using the *Minimum Coverage of a Point* algorithm
      iii. Assign all the other points on the subdivision to s_c

Figure 5.18 illustrates a Coverage Voronoi diagram of a set of four antennas. Each antenna has a different colour associated and its Coverage Voronoi region is represented by that colour. The temporal complexity of the previous algorithm is stated in the following theorem.

Theorem 5.7 *Given a set S of n antennas, the Coverage Voronoi diagram of S can be constructed in $\mathcal{O}(n^5)$ time.*

Proof: Since set S is formed by n antennas, there are up to $\mathcal{O}(n^2)$ diametral discs and perpendicular bisectors between the antennas. The arrangement of diametral discs can be found using a plane sweep that takes $\mathcal{O}(n^3 \log n)$ time. Each face f of the arrangement is divided into $\mathcal{O}(n^2)$ regions when intersected with the perpendicular bisectors previously computed. For each of these regions, select a random point there and find the antenna defining its minimum transmission range using the algorithm *Minimum Coverage of a Point* (proposed in Section 5.2.2). Every point on this region is then assigned to the antenna outputted by the algorithm. According to Theorem 5.5, such antenna can be found in $\mathcal{O}(n)$ time. Consequently, constructing the Coverage Voronoi diagram restricted to a face of the arrangement of discs

takes $\mathcal{O}(n^3)$ time. Since there are $\mathcal{O}(n^2)$ faces, the whole diagram can be constructed in $\mathcal{O}(n^5)$ time. □

The Coverage Voronoi diagram helps to calculate the minimum transmission range needed to cover several objects and not only points. For example, a given line segment e can be overlaid on the diagram and divided into several pieces. The minimum transmission range that covers each piece can be easily calculated using the diagram and the minimum transmission range that covers all these pieces is the minimum transmission range that covers e. This method can also be applied to any type of polygons. However, not much is known about this diagram and further research is needed to better understand this structure. This will be further discussed in the following section.

5.3 Closing Remarks and Future Research

Given a set S of n antennas with transmission range $r \in \mathbb{R}^+$, the problems presented in the first part of this chapter aimed to calculate the maximum $k \in \mathbb{N}$ so that a k-covered path between two points on a given region R exists. The final complexities of the algorithms proposed to solve this problem regarding different types of regions are shown in Table 5.1. Two of the following results are by Huang and Tseng [46, 47] but can also be consulted in the table below since they are relevant to this subject.

Maximum Coverage of	Algorithm's Complexity
Line segment	$\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space
Planar graph with m edges	$\mathcal{O}(mn \log n)$ time and $\mathcal{O}(mn)$ space
Path on an edge-weighted graph with m edges	$\mathcal{O}(m)$ time and space
Polygonal region R	$\mathcal{O}(n^2 \log n)$ time [46]
Cuboidal region R	$\mathcal{O}(n^3 \log n)$ time [47]
Path on polygonal region R with m edges	$\mathcal{O}(n(n+m)\log(n+m))$ time
Path on the plane	$\mathcal{O}(n^2)$ time

Table 5.1: Complexities of the algorithms proposed to maximise the coverage of a path within a region.

The first two sections studied the problem where the region is degenerated into a line segment and a planar graph. In the case of the planar graph G with m edges, the maximum covered path between two arbitrary nodes of G can be found in $\mathcal{O}(m)$ time after a preprocess that runs in $\mathcal{O}(mn \log n)$ time. Although the last two algorithms finish with a sequence of regions over which a maximum covered path can be found, it is not known how to find such

path. It is clear that the path found on a MaxST of the dual graph cannot be used because its edges may cross undesirable regions. The fact that the regions of the arrangement of discs are not convex does not help either. It is not clear what points of those regions should be selected in order to form a path. Future research involving this topic is naturally associated with expanding these ideas to other types of regions. Moreover, since these problems are based in real-life situations, it is reasonable to consider the generalisation of these results to higher dimensions. For example, when using three dimensions all the results can be directly applied to positioning problems.

The second part of this chapter, which starts in Section 5.2, introduced the α-coverage that is a concept associated with 2-coverage but more restrictive. Quality parameter α was considered to be fixed and equal to $\frac{\pi}{2}$ for all problems involving this subject. Given a set S of n antennas, Table 5.2 summarises the results associated with $\frac{\pi}{2}$-coverage that were achieved in this chapter. In that table, "M.T.R." means minimum transmission range.

Problem	Algorithm's Complexity
M.T.R. to $\frac{\pi}{2}$-cover a point	$\mathcal{O}(n)$ time and space
Compute region $\frac{\pi}{2}$-covered by S	$\mathcal{O}(n^3 \log n)$ time
Size of the contour of the region $\frac{\pi}{2}$-covered by S	$\mathcal{O}(n)$
Construct the contour of the region $\frac{\pi}{2}$-covered by S	$\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ space
Construct the Coverage Voronoi diagram of S	$\mathcal{O}(n^5)$ time

Table 5.2: Complexities of the algorithms proposed to solve problems involving $\frac{\pi}{2}$-coverage.

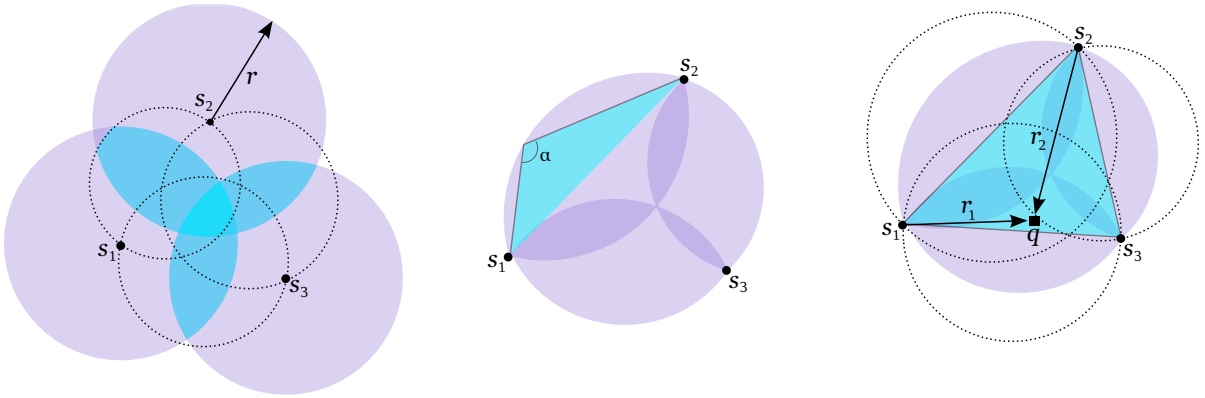
In Section 5.2.3 it was shown how to construct the region $\frac{\pi}{2}$-covered by S. The contour of this region, C(S), was also introduced in this section as well as an algorithm to construct it. Some properties of these structures were analysed, as is the case of the linearity of the contour. This is an important property since it highlights the future possibility of achieving a linear time algorithm to construct C(S). The Coverage Voronoi diagram was introduced and constructed in Section 5.2.4. This diagram helps to calculate the minimum transmission range needed to $\frac{\pi}{2}$-cover several geometrical objects, for example, line segments or polygons. It is also a basis to efficiently recalculate the antennas' minimum range to keep any moving points $\frac{\pi}{2}$-covered. However, not much is known about this diagram and further research is needed to better understand this structure. For example, it is lacking a zone theorem: how many regions of the diagram are crossed by a line? According to the algorithm proposed to construct the Coverage Voronoi diagram, a line crosses $\mathcal{O}(n^4)$ regions since each face of the arrangement of diametral discs is divided into $\mathcal{O}(n^2)$ smaller pieces. Nevertheless, this is probably excessive and it seems reasonable to assume that a line crosses fewer regions of the diagram. Such a theorem would probably help to significantly improve the complexity

of this type of optimisation problems. Moreover, the study of the diagram's properties poses an interesting challenge and opens way to improve the temporal complexity of the current algorithm as well.

Using a similar approach to the one presented in the beginning of this chapter, future research can address $\frac{\pi}{2}$-coverage with limited transmission range. In this variation, each antenna has limited transmission range $r \in \mathbb{R}^+$. Consequently, the region $\frac{\pi}{2}$-covered by antennas s_i and s_j of S no longer is defined as the diametral disc $D_d(s_i, s_j)$. Instead, this region is given by $D(s_i, r) \cap D(s_j, r) \cap D_d(s_i, s_j)$. Therefore, as the example in Figure 5.19(a) illustrates, the region $\frac{\pi}{2}$-covered by S with range r is given by

$$\bigcup_{s_i, s_j \in S} (D(s_i, r) \cap D(s_j, r) \cap D_d(s_i, s_j)).$$

There are $\mathcal{O}(n^2)$ diametral discs, and the lens between the discs centred at the antennas of radius r can be computed in constant time. Since each lens intersects at most four times the diametral disc that contains it, the region $\frac{\pi}{2}$-covered by each pair of antennas with range r can be found in constant time. If every disc is divided in monotone pieces, such arrangement can be found in $\mathcal{O}(n^3 \log n)$ time using a plane sweep. Several problems identical to the ones presented in Sections 5.1.2, 5.1.4 and 5.1.5 can be addressed using this variant.



(a) (b) (c)

Figure 5.19: (a) Blue region is $\frac{\pi}{2}$-covered by S with range r. (b) Purple region is α-covered for $\alpha = \frac{2}{3}\pi$. (c) Distance r_2 is the minimum embracing range to 1-well illuminate q and it is larger than r_1, which is the minimum transmission range to α-cover q, for $\frac{\pi}{2} \leq \alpha \leq \frac{2}{3}\pi$.

Future research concerning this definition of coverage also includes the study of cases where the angle α is different from $\frac{\pi}{2}$ (see Figure 5.19(b)). For obtuse angles, the region α-covered by S may not be connected and the probable existence of holes will surely rise the complexity of the α-covered region. Note that if the angle α is large enough, the arcs become the line segments that connect the antennas. This subject also involves 1-good illumination

(introduced in Chapter 2): if $\frac{\pi}{2} \leq \alpha \leq \frac{2}{3}\pi$, then the minimum transmission range to α-cover a point is less than or equal to the minimum embracing range to 1-well illuminate that point (see Figure 5.19(c)). For this interval, every closest embracing triangle is contained in the α-covered region and as some points on the plane are α-covered but not 1-well illuminated, the α-coverage is less restrictive than 1-good illumination.

To conclude, the algorithms presented to calculate the maximum $k \in \mathbb{N}$ so that a k-covered path between two points on a given region exists were studied for four different types of regions. Future research is aimed at studying this problem applied to other types of regions, as well as computing the k-covered path within every type of region. Moreover, translating the previous results to higher dimensions is also an interesting challenge given its practical applications. Although the second part of this chapter presented compelling problems, most of them remain unresolved. This is particularly evident in the case of the Coverage Voronoi diagram. Each of these concepts merits a deeper analysis in future research, since it is clear that such studies will give rise to new and interesting results.

List of Notations

The following list details the notations introduced in this thesis in order of appearance.

Notation	Description
$\mathrm{CH}(S)$	Convex hull of set S
$\mathrm{int}(x)$	Interior of object x
$d(p, q)$	Euclidean distance between points p and q
$\mathrm{CET}(q)$	Closest embracing triangle for point q
$\mathrm{MER}_S(x)$	Minimum Embracing Range of S that 1-well illuminates x
$D(s_i, r)$	Disk of radius r centred at s_i
$A_r(S)$	Region illuminated by all lights of set S with range r
$\triangle(s_i, s_j, s_k)$	Triangle formed by s_i, s_j and s_k
$\mathrm{MER}_S(\overline{pq})$	Minimum Embracing Range of S that 1-well illuminates $\overline{pq}$
$\mathrm{PB}(\overline{s_i s_j})$	Perpendicular bisector between s_j and s_k
$\mathrm{MER}^{\alpha}(q)$	Minimum embracing range that well α-illuminates q
$\text{E-VR}(s_i, S)$	Embracing Voronoi region of s_i with respect to S

Notation	Description
E-VD(S)	Embracing Voronoi diagram of set S
$\mathrm{VD}_k(S)$	k^{th} order Voronoi diagram of S
$L_k(S)$	Set of sites of S whose closest embracing number is equal to k
OE-VD(S)	Orthogonal Embracing Voronoi diagram of S
$\mathrm{MR}_S(x)$	Minimum Power Transmission Range of S that 2-covers an object x
$\mathrm{VD}_2(S)$	Second Order Voronoi Diagram of S
$P(p,q)$	Path between p and q
$\mathrm{MR}_{S,G}(P(p,q))$	Minimum Power Transmission Range of S that assures the existence of a 2-covered path between nodes p and q on graph G
$\mathrm{VR}_2(s_i,s_j)$	Second Order Voronoi Region of the antennas s_i and s_j
$l_r(s_i,s_j)$	Lens resulting from the intersection of discs $D(s_i,r)$ and $D(s_j,r)$
$\mathrm{MR}_S(P(p,q))$	Minimum Power Transmission Range of S that assures the existence of a 2-covered path between p and q
$b_S(p,q)$	Bottleneck-point for 2-covered paths between p and q
$\mathrm{SP}(p,q)$	A shortest 2-path between p and q
$B(R)$	Boundary of region R
$\mathrm{MC}_s(x)$	Maximum Coverage of object x by S

Notation	Description
$\mathrm{MC}_{S,G}(P(p,q))$	Maximum Coverage of a path between p and q on G
$\mathrm{MC}_{S}(P(p,q))$	Maximum Coverage of a path between p and q
$D_d(s_i, s_j)$	Disk of diameter $\overline{s_i s_j}$
$\mathrm{C}(S)$	Contour of the region covered by S
$\mathrm{SC}(s_i, s_j)$	Boundary of the semicircle of diameter $\overline{s_i s_j}$

Bibliography

[1] M. Abellanas, A.L. Bajuelos, G. Hernández and I. Matos. *Good Illumination of Points and Line Segments with Limited Range Lights*. In XI Encuentros de Geometría Computacional, pp. 231–238. Santander, Spain, 2005.

[2] M. Abellanas, A.L. Bajuelos, G. Hernández and I. Matos. *Good Illumination with Limited Visibility*. In T.E. Simos, editor, International Conference on Numerical Analysis and Applied Mathematics, pp. 35–38. Wiley-VCH Verlag, 2005.

[3] M. Abellanas, A.L. Bajuelos, G. Hernández, I. Matos and B. Palop. *Minimum Illumination Range Voronoi Diagrams*. In 2^{nd} International Symposium on Voronoi Diagrams in Science and Engineering, pp. 317–324. Seoul, Korea, 2005.

[4] M. Abellanas, A.L. Bajuelos, G. Hernández, I. Matos and B. Palop. *Embracing Voronoi Diagram and Closest Embracing Number*. Journal of Mathematical Sciences, volume 161, no. 6, pp. 909–918, 2009. doi:10.1007/s10958-009-9610-0.

[5] M. Abellanas, A.L. Bajuelos and I. Matos. *The Angular Escape*. Cadernos de Matemática, 2007.

[6] M. Abellanas, A.L. Bajuelos and I. Matos. *Good Θ-Illumination of Points*. In 23^{rd} European Workshop on Computational Geometry, pp. 61–64. Graz, Austria, 2007.

[7] M. Abellanas, A.L. Bajuelos and I. Matos. *Some Problems Related to Good Illumination*. In International Conferece on Computational Science and Its Applications, volume 4705/2007, Part I, pp. 1–14. Springer Berlin/Heidelberg, 2007. doi:10.1007/978-3-540-74472-6_1.

[8] M. Abellanas, A.L. Bajuelos and I. Matos. *Variations of Good Illumination*. In XII Encuentros de Geometría Computacional, pp. 265–272. Valladolid, Spain, 2007.

[9] M. Abellanas, A.L. Bajuelos and I. Matos. 2-*Covered Paths by a Set of Antennas with Minimum Power Transmission Range*. Information Processing Letters, volume 109, no. 14, pp. 768–773, 2009. doi:10.1016/j.ipl.2009.03.016.

[10] M. Abellanas, A.L. Bajuelos and I. Matos. *Maximum Covered Path within a Region*. In XIII Encuentros de Geometría Computacional, pp. 277–284. Saragossa, Spain, 2009.

[11] M. Abellanas, A.L. Bajuelos and I. Matos. *Optimal 2-Coverage of a Polygonal Region in a Sensor Network*. Algorithms, volume 2, no. 3, pp. 1137–1154, 2009. doi:10.3390/a2031137.

[12] M. Abellanas, A.L. Bajuelos and I. Matos. *Safe Routes on a Street Graph with Minimum Power Transmission Range*. In 25^{th} European Workshop on Computational Geometry, pp. 85–88. Brussels, Belgium, 2009.

[13] M. Abellanas, S. Canales and G. Hernández. *Buena Iluminación*. In Actas de las IV Jornadas de Matemática Discreta y Algorítmica, pp. 236–246. 2004.

[14] M. Abellanas, S. Canales and G. Hernández. *Más Resultados sobre Buena Iluminación*. In XI Encuentros de Geometría Computacional, pp. 239–246. Santander, Spain, 2005.

[15] M. Abellanas, M. Claverol and F. Hurtado. *Point Set Stratification and Delaunay Depth*. Computational Statistics & Data Analysis, volume 51, no. 5, pp. 2513–2530, 2007. doi: 10.1016/j.csda.2006.09.004.

[16] M. Abellanas, M. Claverol and I. Matos. *The α-Embracing Contour*. In A. González-Escribano, D. Orden and B. Palop, editors, XII Encuentros de Geometría Computacional, pp. 257–264. Valladolid, Spain, 2007.

[17] M. Abellanas, M. Claverol and I. Matos. *The α-Embracing Contour*. In O. Gervasi and M. Gavrilova, editors, International Conference on Sciences and Its Applications, pp. 365–372. IEEE Computer Society, Washington, DC, USA, 2008. doi:10.1109/ICCSA.2008.36.

[18] M. Abellanas and G. Hernández. *Optimización de Rutas de Evacuación*. In XII Encuentros de Geometría Computacional, pp. 273–280. Valladolid, Spain, 2007.

[19] A. Agnetis, E. Grande, P.B. Mirchandani and A. Pacifici. *Covering a Line Segment with Variable Radius Discs*. Computers and Operations Research, volume 36, no. 5, pp. 1423–1436, 2009. doi:10.1016/j.cor.2008.02.013.

[20] T. Asano, S.K. Ghosh and T.C. Shermer. *Visibility in the Plane*. In J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry, pp. 829–876. North-Holland, Amsterdam, the Netherlands, 2000.

[21] F. Aurenhammer and R. Klein. *Voronoi Diagrams.* In J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry, pp. 201–290. North-Holland, Amsterdam, the Netherlands, 2000.

[22] D. Avis, B. Beresford-smith, L. Devroye, H. Elgindy, E. Guévremont, F. and B. Zhu. *Unoriented Θ-Maxima In The Plane: Complexity And Algorithms.* Siam J. Computation, volume 28, pp. 278–296, 1998. doi:10.1137/S0097539794277871.

[23] P. Belleville, P. Bose, J. Czyzowicz, J. Urrutia and J. Zaks. *K-Guarding Polygons on the Plane.* In 6^{th} Canadian Conference in Computational Geometry, pp. 381–386. Saskatchewan, Canada, 1994.

[24] A. Bezdek and W. Kuperberg. *Circle Covering With a Margin.* Periodica Mathematica Hungarica, volume 34, no. 1–2, pp. 3–16, 1997. doi:10.1023/A:1004275121878.

[25] M. Blum, R.W. Floyd, V.R. Pratt, R.L. Rivest and R.E. Tarjan. *Time Bounds for Selection.* Journal of Computer and System Sciences, volume 7, no. 4, pp. 448–461, 1973.

[26] A. Boukerche, X. Fei and R.B. Araujo. *An Energy Aware Coverage-Preserving Scheme for Wireless Sensor Networks.* In 2^{nd} ACM International Workshop on Performance Evaluation of Wireless ad hoc, Sensor, and Ubiquitous Networks, pp. 205–213. ACM, New York, USA, 2005. doi:10.1145/1089803.1089987.

[27] G.S. Brodal and R. Jacob. *Dynamic Planar Convex Hull.* In 43^{rd} Annual IEEE Symposium on Foundations of Computer Science, pp. 617–626. 2002.

[28] S. Canales. Métodos Heurísticos en Problemas Geométricos. Visibilidad, iluminación y vigilancia. Ph.D. thesis, Universidad Politécnica de Madrid, Madrid, Spain, 2004.

[29] M.Y. Chan, D.Z. Chen, F.Y.L. Chin and C.A. Wang. *Construction of the Nearest Neighbor Embracing Graph of a Point Set.* Journal of Combinatorial Optimization, volume 11, no. 4, pp. 435–443, 2006. doi:10.1145/355541.355562.

[30] B. Chazelle. Computational Geometry and Convexity. Ph.D. thesis, Yale University, 1980.

[31] B.M. Chazelle and D.T. Lee. *On a Circle Placement Problem.* Computing, volume 36, no. 1-2, pp. 1–16, 1986. doi:10.1007/BF02238188.

[32] S.N. Chiu and I.S. Molchanov. *A New Graph Related to the Directions of Nearest Neighbours in a Point Process.* Advances in Applied Probability, volume 35, no. 1, pp. 47–55, 2003. doi:0.1239/aap/1046366098.

[33] V. Chvátal. *A Combinatorial Theorem in Plane Geometry.* Journal of Combinatorial Theory, Series B, volume 18, pp. 39–41, 1975. doi:10.1016/0095-8956(75)90061-1.

[34] M. Claverol. Geometric Problems on Computational Morphology. Ph.D. thesis, Universitat Politécnica de Catalunya, Barcelona, Spain, 2004.

[35] G.K. Das, S. Das, S.C. Nandy and B.P. Sinha. *Efficient Algorithm for Placing a Given Number of Base Stations to Cover a Convex Region.* Journal of Parallel and Distributed Computing, volume 66, no. 11, pp. 1353–1358, 2006. doi:10.1016/j.jpdc.2006.05.004.

[36] G.L. Dirichlet. *Uber die Reduktion der positiven quadratischen Formen mit drei unbestimmten ganzen Zahlen.* Journal fur die Reine und Angewandte Mathematik, volume 40, pp. 209–227, 1850.

[37] H. Edelsbrunner. Algorithms in Combinatorial Geometry. Springer-Verlag New York, Inc., New York, USA, 1987.

[38] A. Efrat, S. Har-Peled and J.S.B. Mitchell. *Approximation Algorithms for Two Optimal Location Problems in Sensor Networks.* In 2^{nd} International Conference on Broadband Networks, volume I, pp. 714–723. 2005. doi:10.1109/ICBN.2005.1589677.

[39] S. Fisk. *A Short Proof of Chvátal's Watchman Theorem.* Journal of Combinatorial Theory, Series B, volume 24, no. 3, p. 374, 1978.

[40] K.R. Gabriel and R.R. Sokal. *A New Statistical Approach to Geographic Variation Analysis.* Systematic Zoology, volume 18, no. 3, pp. 259–278, 1969. doi:10.2307/2412323.

[41] S.K. Ghosh. Visibility Algorithms in the Plane. Cambridge University Press, New York, USA, 2007.

[42] S.W. Golomb and L.D. Baumert. *Backtrack Programming.* Journal of ACM, volume 12, no. 4, pp. 516–524, 1965.

[43] J. Gomez, A.T. Campbell, M. Naghshineh and C. Bisdikian. *Conserving Transmission Power in Wireless Ad Hoc Networks.* In 9^{th} International Conference on Network Protocols, pp. 24–34. IEEE Computer Society, Washington, DC, USA, 2001.

[44] P.J. Green. *Peeling Bivariate Data.* Interpreting Multivariate Data, 1981. doi:10.1002/0471667196.ess1941.pub2.

[45] P. Hall. Introduction to the Theory of Coverage Processes. John Wiley & Sons Inc, New York, USA, 1988.

[46] Chi-Fu Huang and Yu-Chee Tseng. *The Coverage Problem in a Wireless Sensor Network.* Mobile Networks and Applications, volume 10, no. 4, pp. 519–528, 2005. doi:10.1145/1160162.1160175.

[47] Chi-Fu Huang, Yu-Chee Tseng and Li-Chu Lo. *The Coverage Problem in three-dimensional Wireless Sensor Networks.* Global Telecommunications Conference, volume 5, pp. 3182–3186, 2004. doi:10.1109/GLOCOM.2004.1378938.

[48] C.S. Kaplan. *Voronoi Diagrams and Ornamental Design.* In First Annual Symposium of the International Society for the Arts, Mathematics, and Architecture, pp. 277–283. 1999.

[49] S. Kapoor, S.N. Maheshwari and J.S.B. Mitchell. *An Efficient Algorithm for Euclidean Shortest Paths Among Polygonal Obstacles in the Plane.* Discrete Computational Geometry, volume 18, no. 4, pp. 377–383, 1997. doi:10.1145/160985.161156.

[50] R. Karlsson and M.H. Overmars. *Scanline Algorithms on a Grid.* BIT Numerical Mathematics, volume 28, no. 2, pp. 227–241, 1988. doi:10.1007/BF01934088.

[51] B. Kaučič and B. Žalikm. *K-guarding of Polyhedral Terrain.* International Journal of Geographical Information Science, volume 18, no. 7, pp. 709–718, 2004. doi:10.1080/13658810410001705299.

[52] G.D. Kazazakis and A.A. Argyros. *Fast Positioning of Limited-Visibility Guards for the Inspection of 2D Workspaces.* In IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2843–2848. 2002.

[53] J.M. Keil. *Decomposing a Polygon into Simpler Components.* SIAM Journal on Computing, volume 14, no. 4, pp. 799–817, 1985.

[54] H.T. Kung, F. Luccio and F.P. Preparata. *On Finding the Maxima of a Set of Vectors.* Journal of the ACM, volume 22, no. 4, pp. 469–476, 1975. doi:10.1145/321906.321910.

[55] D.T. Lee. *On k-Nearest Neighbor Voronoi Diagrams in the Plane.* IEEE Transactions on Computers, volume 31, no. 6, pp. 478–487, 1982. doi:10.1109/TC.1982.1676031.

[56] D.T. Lee and Y.F. Wu. *Geometric Complexity of some Location Problem.* Algorithmica, volume 1, no. 1–4, pp. 193–211, 1986. doi:10.1007/BF01840442.

[57] X.-Y. Li, P.-J. Wan and O. Frieder. *Coverage in Wireless Ad Hoc Sensor Networks.* IEEE Transactions on Computers, volume 52, no. 6, pp. 753–763, 2003. doi:10.1109/TC.2003.1204831.

[58] D. Matijevic and R. Osbild. *Finding the Θ-Guarded Region.* Computational Geometry: Theory and Applications, volume 43, no. 2, pp. 207–218, 2009. doi:10.1016/j.comgeo.2009.07.001.

[59] J. Matousek. *Computing the Center of Planar Point Sets.* Computational Geometry: Papers from the DIMACS Special Year, pp. 221–230, 1991.

[60] T. Matsui. *The Minimum Spanning Tree Problem on a Planar Graph.* Discrete Applied Mathematics, volume 58, no. 1, pp. 91–94, 1995. doi:10.1016/0166-218X(94)00095-U.

[61] N. Megiddo. *Combinatorial Optimization with Rational Objective Functions.* Mathematics of Operations Research, volume 4, no. 4, pp. 414–424, 1979. doi:10.1287/moor.4.4.414.

[62] N. Megiddo. *Applying Parallel Computation Algorithms in the Design of Serial Algorithms.* Journal of the ACM, volume 30, no. 4, pp. 852–865, 1983. doi:10.1145/2157.322410.

[63] N. Megiddo. *Linear-time Algorithms for Linear Programming in $\mathbb{R}^3$ and Related Problems.* SIAM Journal on Computing, volume 12, no. 4, pp. 759–776, 1983. doi:10.1137/0212052.

[64] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M.B. Srivastava. *Coverage Problems in Wireless Ad-hoc Sensor Networks.* In Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, volume 3, pp. 1380–1387. 2001.

[65] D.P. Mehta, M.A. Lopez and L. Lin. *Optimal Coverage Paths in Ad-hoc Sensor Networks.* In IEEE International Conference on Communications, volume 1, pp. 507–511. ACM, New York, USA, 2003. doi:10.1109/ICC.2003.1204228.

[66] G.L. Miller, T. Phillips and D. Sheehy. *The Centervertex Theorem for Wedges.* In CCCG: Canadian Conference in Computational Geometry, pp. 17–19. 2009.

[67] H.K. Miller, S. Ramaswami, P. Rousseeuw, T. Sellarès, D. Souvaine, I. Streinu and A. Struyf. *Efficient Computation of Location Depth Contours by Methods of Computational Geometry.* Statistics and Computing, volume 13, no. 2, pp. 153–162, 2003. doi:10.1023/A:1023208625954.

[68] S.C. Ntafos. *Watchman Routes under Limited Visibility.* Computational Geometry: Theory and Applications, volume 1, no. 3, pp. 149–170, 1992. doi:10.1016/S0925-7721(99)00022-X.

[69] J. O'Rourke. Art Gallery Theorems and Algorithms. The International Series of Monographs on Computer Science. Oxford University Press, New York, USA, 1987.

[70] J. O'Rourke. Computational Geometry in C. Cambridge University Press, 1998.

[71] V. Sacristán. Geometric Optimization and Applications in Visibility. Ph.D. thesis, Universitat Politécnica de Catalunya, Barcelona, Spain, 1997.

[72] M.I. Shamos and D.J. Hoey. *Closest-point Problems*. In 16^{th} Annual IEEE Symposium on Foundations of Computer Science, pp. 151–162. 1975.

[73] T. Shermer. *Recent Results in Art Galleries*. In Proceedings of the IEEE, volume 80, no. 9, pp. 1384–1399. 1992. doi:10.1109/5.163407.

[74] J. Smith and W. Evans. *Triangle Guarding*. In 15^{th} Canadian Conference on Computational Geometry, pp. 76–80. 2003.

[75] Y.G. Stoyan and V.M. Patsuk. *Covering a Compact Polygonal Set by Identical Circles*. Computational Optimization and Applications, 2008. doi:10.1007/s10589-008-9191-8.

[76] J.W. Tukey. *Mathematics and the Picturing of Data*. In R.D. James, editor, International Congress of Mathematicians, volume 2, pp. 523–531. 1975.

[77] J. Urrutia. *Art Gallery and Illumination Problems*. In J.-R. Sack and J. Urrutia, editors, Handbook of Computational Geometry, pp. 973–1027. North-Holland, Amsterdam, the Netherlands, 2000.

[78] G. Voronoi. *Nouvelles Applications des Paramètres Continus à la Théorie des Formes Quadratiques*. Journal fur die reine und angewandte Mathematik, volume 134, pp. 198–287, 1908.

[79] G. Voronoi. *Deuxième Mémoire: Recherches sur les Parallélloedres Primitifs*. Journal fur die reine und angewandte Mathematik, volume 136, pp. 67–181, 1909.

[80] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless and C. Gill. *Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks*. In Proceedings of the ACM SenSys, pp. 28–39. 2003.

[81] Ning Xu. *A Survey of Sensor Network Applications*, 2003.

[82] C. Zhang, Y. Zhang and Y. Fang. *Localized Algorithms for Coverage Boundary Detection in Wireless Sensor Networks*. Wireless Networks, volume 15, no. 1, pp. 3–20, 2009. doi: 10.1007/s11276-007-0021-1.

[83] H. Zhang and J. Hou. *Maintaining Sensing Coverage and Connectivity in Large Sensor Networks*. Ad Hoc & Sensor Wireless Networks, volume 1, no. 1–2, pp. 89–124, 2005.

[84] Zongheng Zhou, S.R. Das and H. Gupta. *Connected K-coverage Problem in Sensor Networks*. In R.D. James, editor, 13^{th} International Conference on Computer Communications and Networks, volume 2, pp. 373–378. 2004. doi:10.1109/ICCCN.2004.1401672.

[85] Zongheng Zhou, S.R. Das and H. Gupta. *Variable Radii Connected Sensor Cover in Sensor Networks*. ACM Transactions on Sensor Networks, volume 5, no. 1, pp. 1–36, 2009. doi:10.1145/1464420.1464428.