A Pseudo-Polynomial Time $O(\log^2 n)$-Approximation Algorithm for Art Gallery Problems

by

Ajay A. Deshpande

B.Tech., M.Tech., Mechanical Engineering
Indian Institute of Technology Bombay, 2001

Submitted to the Department of Mechanical Engineering and the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of

Master of Science in Mechanical Engineering
and
Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2006

Signature redacted

Author .....
Department of Mechanical Engineering
May 16, 2006

Signature redacted

Certified by .....
Sanjay E. Sarma (Thesis Supervisor)
Associate Professor of Mechanical Engineering

Signature redacted

Certified by .....
Erik Demaine (Thesis Reader)
Associate Professor of Electrical Engineering and Computer Science

Signature redacted

Accepted by .....
Lallit Anand
Chairman, Department Committee on Graduate Students
Department of Mechanical Engineering

Signature redacted

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students
Department of Electrical Engineering and Computer Science

A Pseudo-Polynomial Time $O(\log^2 n)$-Approximation Algorithm for Art Gallery Problems

by

Ajay A. Deshpande

Submitted to the Department of Mechanical Engineering and
the Department of Electrical Engineering and Computer Science
on May 16, 2006, in partial fulfillment of the requirements for the degrees of

Master of Science in Mechanical Engineering
and
Master of Science in Electrical Engineering and Computer Science

Abstract

In this thesis, we give a pseudo-polynomial time $O(\log^2 n)$-approximation algorithm for a variant of the art gallery problem — the *point-guard problem.* The point-guard problem involves finding the minimum number of points and their positions so that guards located at these points cover the interior of the art gallery. Our algorithm is pseudo-polynomial in the sense that it is polynomial in the number of walls of the art gallery but is possibly exponential in the number of bits required to represent the positions of the vertices of the art gallery. Our approach involves reducing the point-guard problem to a new problem of choosing a minimum number of guard-locations from a finite set obtained by a special subdivision procedure. The new problem has the optimal solution at most three times the optimal solution of the point-guard problem. We further reduce the new problem to the set cover problem and obtain an approximate solution to the set cover problem.

Thesis Supervisor: Sanjay E. Sarma
Title: Associate Professor of Mechanical Engineering

Thesis Reader: Erik Demaine
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

First of all, I would like to thank my advisor, Sanjay Sarma for his invaluable guidance and support. Sanjay gave me complete freedom in terms of the classes that I took and the problems I chose to work on and I value that a lot. I have learned a lot from him over these past few years. Besides being an excellent academic advisor, he has given me invaluable advice on various other aspects of life for which I will be forever indebted to him. I am amazed by his energy and enthusiasm. At the end of our every meeting, I feel charged with new vigor.

I would like to thank Erik Demaine for his guidance and agreeing to be my thesis reader. During the course of this project, he gave several useful suggestions and pointed to appropriate references. I am grateful to him for introducing me to algorithms in my first year through 6.046.

I am thankful to Taejung Kim for several useful discussions. I will never forget those three days before the conference deadline when Taejung and I sat together and finished writing the paper, most parts of which appear in this thesis. I am thankful to him for being really patient with me. I learned a lot from him about paper writing.

I would like to thank Prahladh Harsha for patiently listening to my rambling on the problem. He made several useful comments on the paper I wrote with Taejung.

I would like to thank my current and former office mates, Kashif Khan, Sriram Krishnan, Kirti Mansukhani, Seung-Kil Son and Marty Vona for creating wonderful environment in the office. I was fortunate enough to have many friends at MIT. I would like to thank Sreekar Bhaviripudi, Shashibhushan Borade, Amit Deshpande, Sriram Krishnan, Dhanushkodi Mariappan, Sayan Mitra, Ashish Shah, Vijay Shilpiekandula, Amit Surana, Kripa Varanasi and his wife Manasa, Murtaza Zafer. In my first two and half year, Dhanush and Kripa were my afternoon coffee buddies and we had several interesting discussions. Vijay and two Amit's (Deshpande and Surana) are my current coffee and fun time buddies. I have spent real quality time with these guys and have learned a lot from each one. Kripa is like a 'big brother' to me. I will forever cherish the time I spent with him at MIT. I have learned a lot from

him. I would also like to thank my friends in other parts of the world including my school friends and my IIT friends. I would also like to thank Pavithra and Prahladh's parents for a truly memorable summer of 2004. I apologize in advance to anybody whose name I missed unintentionally.

I would like to thank all my relatives back in India for their support and affection. Finally I would like to thank Pavithra Harsha, Prahladh Harsha, my brother Nitin and my parents Alka and Ashok for always being there. Since the time I got to know Pavithra and Prahladh at MIT, they have become inseparable part of my life. My brother, Nitin has been always my true friend, philosopher and role model in life. Most of what I am today is due to my brother and my parents. I cannot express in words my gratitude towards them.

Contents

List of Figures

Chapter 1

Introduction

The field of computational geometry is around 30 years old. It largely involves computations with basic geometric elements such as points, lines, curves, surfaces, etc. Since its emergence, this ever growing field has had huge impact on many areas of science and engineering including computer graphics, robotics, design, manufacturing, molecular biology, etc. The Art Gallery Problem is one of the early problems that contributed to the growth of some of the central notions such as visibility, triangulation, partitioning, etc. in this field. In this thesis, we propose a pseudo-polynomial time approximation algorithm for this problem.

We begin this chapter by giving brief history of the problem. Then we describe the problem of our interest and give brief account of the related work. We then outline our contributions and summarize the applications. We conclude the chapter by presenting the outline of the remaining thesis.

1.1 Art Gallery Theorem — A Brief Introduction

The Art Gallery Problem is one of the classic problems in the field of computational geometry. The problem was first posed by Victor Klee extemporaneously in response to request for an interesting geometric problem from Vasek Chvátal at a conference at Stanford in August 1973 [13, 15]. It addresses the following question [13, 15]: How many guards are required

to guard the interior of an art gallery with n walls? Soon Chvátal showed that $\lfloor \frac{n}{3} \rfloor$ guards are always sufficient and occasionally necessary to guard an art gallery [4]. This well-known result is now known as "Chvátal's Art Gallery Theorem".

Since then, numerous variations of this problem have been studied. We refer to the category of all these problems as art gallery problems. Sometimes these problems are also referred to as watchman problems or illumination problems in different contexts. Some of these include mobile guards, guards with limited visibility, illumination of families of convex sets on the plane, guarding rectilinear polygons and others. For more details, readers are referred to a book by O'Rourke [13] and survey articles by Shermer [14] and Urrutia [15]. O'Rourke's book was published in 1987 and is the first monograph entirely devoted to this subject. Shermer's paper [14] was published in 1992 and summarizes the results since the publication of O'Rourke's book. Urrutia's paper [15] is most recent in the series and contains the latest results and open problems in the subject. Below we describe summary of the basic results on different variations of art gallery problems. In the results below, *point guards* are referred to as guards that can be located anywhere inside the art gallery; *vertex guards* are those that can be located only at the corners or *vertices* of the art gallery; *line guards* are those that are allowed to move on the line segment(s) obtained by intersecting a line with the art gallery.

- As mentioned above, $\lfloor \frac{n}{3} \rfloor$ vertex and point guards are always sufficient and occasionally necessary to guard an art gallery [4].

- A rectilinear or orthogonal art gallery is one in which all walls are either *horizontal* or *vertical*. $\lfloor \frac{n}{4} \rfloor$ vertex and point guards are always sufficient and occasionally necessary to guard a rectilinear art gallery with n walls [13].

- $\lfloor \frac{n}{4} \rfloor$ line guards that are allowed to move on line segment wholly contained in an art gallery are always sufficient and occasionally necessary to guard a gallery. For an orthogonal art gallery, $\lfloor \frac{(3n+4)}{16} \rfloor$ line guards are necessary and sufficient [13].

- $\lceil \frac{(n+h)}{3} \rceil$ point guards are necessary and sufficient to guard an art gallery with n vertices

and h *holes* [15]. Shermer conjectured that $\lfloor \frac{(n+h)}{3} \rfloor$ vertex guards are sufficient to guard an art gallery with n vertices and h *holes* [13].

Most of the results on art gallery problems are of the type: X number of particular type of guards are always sufficient and occasionally necessary to guard a particular type of art gallery. An approach usually followed is to find an upper bound on the number of guards required using partitioning arguments. The next step involves finding the *worst-case* configuration of the art gallery for which those many guards are necessary concluding the tightness of the bound. However the algorithmic question of finding exact locations of particular types of guards for any given configuration of the art gallery has been often left unanswered. In fact, in his survey, Urrutia states, "One approach that has been neglected in the study of art gallery problems is that of finding algorithms that obtain approximate solutions in terms of optimal ones." The emphasis on approximate solutions is due to the fact that several art gallery problems of finding minimum number of guards have been proved to be *NP-hard*. In this thesis, we focus on the problem of finding an approximate solution to one type of art gallery problems.

1.2 The Problem

We study one of the versions of art gallery problems, also known as the *point-guard problem*. As mentioned before, a *point guard* refers to the guard that can be located anywhere inside the art gallery. The point-guard problem involves finding the minimum number of points and their positions so that guards located at these points *cover* (i.e., *see* or guard) every point in the interior of the art gallery with n walls. We represent the art gallery as a simple polygon with n vertices and n edges. We also allow the art gallery to have *holes*. A *hole* is nothing but a simple polygon that is fully contained within the art gallery. In other words, holes represent obstacles in the art gallery. We wish to guard remaining portions of the interior of the art gallery including walls of the holes. Below we briefly survey related work on the point-guard problem. A detailed account of this is given in Chapter 2.

1.2.1 Related Work

The *vertex-guard problem* refers to finding minimum number of vertex guards that guard the art gallery. The vertex-guard problem as well as the point-guard problem, both have been shown to be *NP-hard* [11, 13, 15]. Thus, we can only hope for an approximate solution to the problem. However, as mentioned in [15], there is not much work done in the area of approximation algorithms for art gallery problems. In the first paper in this area [8], Ghosh proposes an $O(\log n)$-approximation algorithm for the minimum vertex-guard problem. Banos *et al.* [9] consider another version of the art gallery problem in which the guards are required to cover only walls of the art gallery. Their algorithm works even when the guards have range and incidence constraints. They propose a randomized algorithm which computes with *high probability* a solution whose size is at most a factor $O(\log n \cdot \log(c \log n))$ times the size c of the optimal solution. We explain the notion of *high probability* in this context in the next chapter where we review this work in detail. In both these papers [8, 9], the basic idea is to choose a finite set of points as potential candidates for locating guards and subdivide the art gallery by constructing the visibility polygons of all these candidate points. The problem of covering all the cells in the subdivision generated by constructing all the visibility polygons is formulated as a set cover problem, which is then solved to yield an approximate solution to the original problem. In [8], by definition of the problem, the set of potential candidates consists of the vertices of the art gallery and is finite. In [9], a set of candidate guard-locations is chosen randomly. In the same paper, the authors show that the VC-dimension of the discretized problem is bounded above by $O(\log n)$ and use a VC-dimension-based algorithm to solve the set cover problem to obtain with high probability a near-optimal solution whose quality does not depend on the number of random samples.

Recently, Eidenbenz *et al.* [7] proved inapproximability results for several versions of the art gallery problem. They show that the art gallery problem for the case of art gallery without holes is *APX-hard*, i.e., no polynomial time algorithm can achieve an approximation ratio of $1+\delta$ for a constant $\delta > 0$ for the art gallery without holes unless $P = NP$. They also prove that the problem of art gallery with holes can not be approximated by a polynomial

time algorithm with ratio $(\frac{1-\epsilon}{12})\ln n$ for any $\epsilon > 0$, unless $NP \subseteq TIME(n^{O(\log\log n)})$. In [2], the authors show that the art gallery problem restricted even to a special class of art galleries represented as *2-link polygons* is also *APX-hard.*

In this thesis, we study the point-guard version of the art gallery problem (with holes), however without any range or incidence constraints on the guards. It is to be noted that the vertex-guard version of the problem can be easily reduced to a purely combinatorial problem, which can be solved exactly using an exponential time algorithm. However, in case of the point-guard problem, we are not aware of any such exponential time algorithm. In fact, we even do not know of any exponential time algorithm that produces an approximate solution to our problem with sublinear, i.e., $o(n)$ approximation ratio.

1.3 Our Contribution

In this thesis, we propose a pseudo-polynomial time $O(\log^2 n)$-approximation algorithm for the point-guard problem. Our algorithm is pseudo-polynomial in the sense that it is polynomial in the number of walls of the art gallery but is possibly exponential in the number of bits required to represent the positions of the vertices of the art gallery.

Our basic approach involves choosing only a finite set of guard-locations as potential candidates and reducing the point-guard problem to a new problem of choosing a minimum number of guards from this finite set. We devise our algorithm such that the new problem has an optimal solution at most three times the optimal solution to the original point-guard problem. Then we further reduce the new problem to a set cover problem which can be solved approximately.

Our overall algorithm can be summarized in the following 3 steps:

- **Step 1:** Generate an initial triangulation of the art gallery.
- **Step 2:** Subdivide the initial triangulation and choose vertices of the triangulation as potential candidates for locating guards.

- **Step 3:** Formulate the set cover problem and solve it to obtain a near-optimal set of guard-locations.

We choose vertices of the triangle obtained at the end of Step 2 as potential candidates for locating point guards. Our subdivision procedure guarantees that had the guard-locations been restricted to these points only, the minimum number of guards required is at most three time the minimum number of guards for the point-guard problem. In Step 3, we construct visibility sets of potential guard-locations and formulate the set cover problem. We solve the set cover problem approximately using VC-dimension-based algorithm.

1.4 Applications

The main application of the art gallery problem is in surveillance. The problem is posed using the same language — how many security cameras, guards are required to guard an art gallery with valuables?

The recent interest in the problem stems from applications in the area of wireless communications [7, 12]. The signal coverage of an antenna is modeled as a sphere [7]. The problem of interest is then to place minimal number of antennae in a terrain such that each point receives minimal signal strength. Similarly the sensing coverage of a wireless sensor is modeled as a disc [12]. The optimal coverage problem involves finding minimum number of sensors and their locations in a region such that each *interesting* event within the region is detected by at least one sensor. Both these problems can be seen as variants of the art gallery problem which involve *guards* with range constraints.

In [9], Banos and Latombe describe an application in the field of robotics. Their problem involves capturing texture data of the surrounding environment in a workspace in autonomous fashion. Their set up involves mobile robots each with a mounted camera having range and incidence constraints. They represent the workspace as a 2-D art gallery and based on the visibility constraints of cameras, they derive solution of the art gallery problem. The mobile robots are then sent to locations obtained as a solution to the art gallery problem

where they capture the texture data of the surrounding. We describe their formulation in detail in the next chapter.

1.5 Outline of Thesis

In Chapter 2, we define the basic terminology and formally define the problem. Then we review the related work on the problem in detail. In Chapter 3, we describe our overall algorithm in more detail. We also briefly mention running time and approximation ratio analysis in the same chapter. In Chapter 4, we describe Step 1 of generating initial triangulation of our algorithm and associated correctness results. In Chapter 5, we describe Step 2 in detail. In this step, we further subdivide the initial triangulation such that each triangle in the final triangulation satisfies a special property. In Chapter 6, we explain Step 3 of the algorithm. In this step, we formulate the set cover problem by choosing the vertices of the triangulation obtained in Step 2 as potential guard-locations and solve it using the VC-dimension-based approximation algorithm. We conclude in Chapter 7 by summarizing our results and suggesting possible topics for future work.

Chapter 2

Problem Statement and Related Work

We begin this chapter by defining basic terminology and formulating our problem formally. Then we review related work on our problem. We explain the related work in terms of the terminology we develop here so that it may help readers to understand similarities and differences between our algorithm and algorithms proposed before.

2.1 Basic Terminology and Problem Statement

We represent an art gallery without holes as a simple polygon. Let P be a simple polygon with n vertices. Let $bd(P)$ denote the boundary of P and $int(P)$ denote interior of P excluding the boundary. Thus, $P = bd(P) \cup int(P)$. Let $Gallery(P)$ denote the set P itself. We also use $Gallery(P)$ to indicate an art gallery without holes. We say that two points in $Gallery(P)$ see each other if the line segment joining these two points is fully contained in $Gallery(P)$. In other words, it does not intersect with the exterior of P. We formally state our problem for the case of art galley without holes.

Problem: *Point-guard Problem* for an art gallery without holes, $Gallery(P)$, involves finding minimum number of points and their positions in $Gallery(P)$ such that every point in $Gallery(P)$ sees at least one point among these select points.

Now we consider the case of an art gallery with holes. Let P denote a simple poly-

gon. A hole is also nothing but a simple polygon. Suppose P contains h number of holes, $P_1, P_2, \cdots, P_h$, such that any P_i does not intersect with P or other holes. Holes can be viewed as obstacles in the art gallery. In this case, let n be the total number of vertices of P as wells as $P_1, P_2, \cdots, P_h$. The set $P \setminus \{int(P_1) \cup int(P_2) \cup \cdots \cup int(P_h)\}$ is referred to as $Gallery(P, P_1, P_2, \cdots, P_h)$. From now onwards we use $Gallery(P, P_1, P_2, \cdots, P_h)$ to denote an art gallery with holes. Two points in $Gallery(P, P_1, P_2, \cdots, P_h)$ see each other if the line segment joining these two points lies entirely within $Gallery(P, P_1, P_2, \cdots, P_h)$. In other words, the line segment does not intersect with any $int(P_i)$ or exterior of P. We formally state our problem for this case below.

Problem: *Point-guard Problem* for $Gallery(P, P_1, P_2, \cdots, P_h)$, an art gallery with holes, involves finding minimum number of points and their positions in it such that every point in $Gallery(P, P_1, P_2, \cdots, P_h)$ sees at least one point among these select points.

As mentioned in Chapter 1, several other variants of this problem have been defined. Note that we do not impose any range constraints on visibility of a point. One particular variant, *vertex-guard problem*, is defined in the similar way except that the select points are chosen only from vertices of P or holes. In the case of art gallery without holes, formally it refers to finding minimum number of vertices of P such that every point in $Gallery(P)$ sees at least one of these select vertices. For the sake of simplicity in understanding, throughout this thesis, we restrict our discussion to the case of art gallery without holes. The algorithm for this case can be easily extended to the case of art gallery with holes. At the end of the thesis in Chapter 7, we comment about this case. In the case of art gallery without holes, $Gallery(P)$ is same as P. Hence, from now onwards, instead of $Gallery(P)$ we refer to P itself as an art gallery. Now we further develop basic terminology that would be used throughout the thesis. Most of the definitions and notation we present in this section have been borrowed from [1, 10]; however, we reformulate some of these and define new ones for our convenience. Most of the notions we describe below are illustrated in Figure 2.1.

Consider any point $x \in P$. As mentioned above x sees a point $y \in P$ if the line segment xy lies entirely within P. P has n vertices. Some of these are reflex vertices that subtend

an angle greater than 180^0 inside P. The visibility polygon $V(x)$ is the polygon consisting of all the points in P that are visible from x. Note that some of the edges of $V(x)$ coincide with those of the original polygon P and some are newly introduced as shown in Figure 4.1(a). A new edge is introduced at a reflex vertex of P that blocks the view of x. We call this reflex vertex a blocking reflex vertex. The other end-point of the new edge which lies on the boundary of P is referred to as an image of x through the blocking reflex vertex. To remove any ambiguities, we assume that for P and $V(x)$, no two consecutive edges are collinear. Note that, in the case of a convex P, $V(x)$ of any point $x \in P$ is P itself. We can also reformulate our problem in terms the definition of a visibility polygon. We wish to find minimum number of points, $x_1, x_2, \cdots, x_k \in P$ such that $\{V(x_1) \cup V(x_2) \cdots \cup V(x_k)\} = P$

Now we give a series of definitions related to visibility of an edge from a point. For any point $x \in P$, we say that x sees an edge of P, if it sees a point on the edge. If x cannot see either of the end-points of a visible edge of P, we say that x sees the edge partially. We call the corresponding edge of P a partial edge with respect to x. We say that x sees a visible edge of P non-partially, if it sees at least one of its end-points. We call the corresponding edge of P a non-partial edge with respect to x. If we join every vertex of $V(x)$ to x, we get a triangulation of $V(x)$. We call each triangle as a visibility sector of x. The edge of a visibility sector that is a part of an edge of P is referred to as a base of the visibility sector. Depending upon the type of the edge of P corresponding to the base of a visibility sector, we classify the visibility sector into non-partial-edge sector or partial-edge sector.

Having defined basic terms and stated our problem formally, in the next section we review in detail related work on this problem.

2.2 Related Work

We begin by reviewing the classic result — "Chvátal's Art Gallery Theorem". Chvátal's result provides mere upper bound on the number of guards required for any kind of art gallery in terms of the number of vertices of the art gallery. However, it does not provide an algorithmic solution for a given art gallery. The vertex-guard problem and our problem

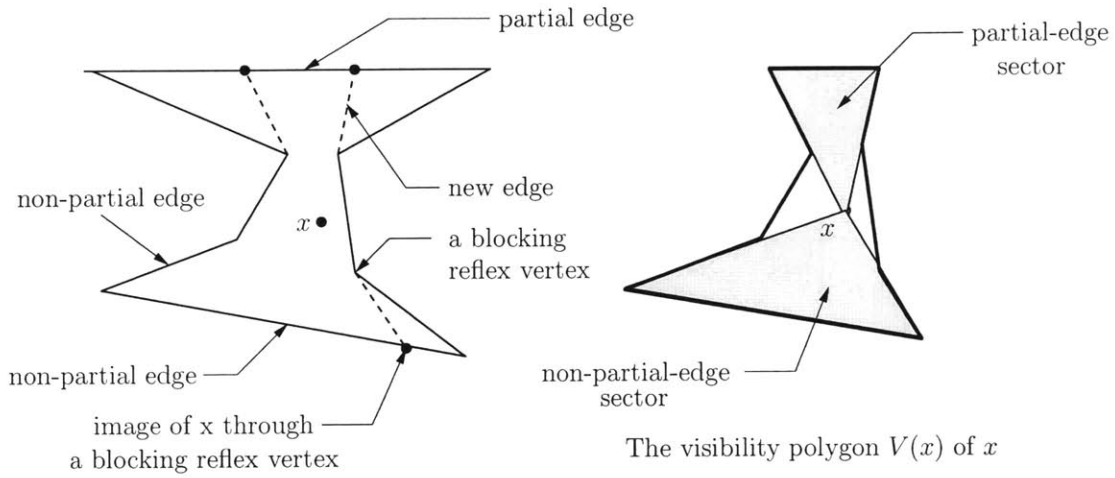

Figure 2.1: Visibility polygon and visibility sectors

– the point-guard problem – both have been shown to be *NP-hard* [11, 13, 15]. Thus, we resort to finding approximate solution to the problem. After reviewing Chvátal's result, we describe Ghosh's $O(\log n)$-approximation algorithm for the vertex-guard problem [8]. Next we explain heuristic proposed by Banos and Latombe in [9] for a variation of the point-guard problem with range and incidence constraints. Finally, we review inapproximability results proved by Eidenbenz *et al.* in [7].

2.2.1 Chvátal's Art Gallery Theorem

We provide proof suggested by Fisk in 1978 based on the graph coloring argument [13, 15]. Chvátal's original proof is based on induction arguments and is not as concise as Fisk's proof. Nevertheless, it is full of insights and we urge interested readers to refer to [13].

Theorem 2.2.1. $\lfloor \frac{n}{3} \rfloor$ *point guards are always sufficient and occasionally necessary to guard an art gallery P with n vertices.*

Proof. Here we just give an outline of the proof. We refer readers to [13, 6, 15] for details. First we obtain triangulation of P by adding $n-2$ interior diagonals. This implies that $n-2$ guards, one guard per triangle are sufficient. However, this seems an overkill. The next step

is to *3-color* vertices of this triangulation such that any two vertices sharing an edge have different colors. Any triangulated planar graph admits a *3-coloring*. This partitions vertices of P into three chromatic classes. We locate guards at each vertex of the smallest chromatic class. The size of the smallest chromatic class is at most $\lfloor \frac{n}{3} \rfloor$. This proves the sufficiency part of the theorem. Figure 2.2 indicates a *comb* example where $\lfloor \frac{n}{3} \rfloor$ guards are necessary. This proves the result. □

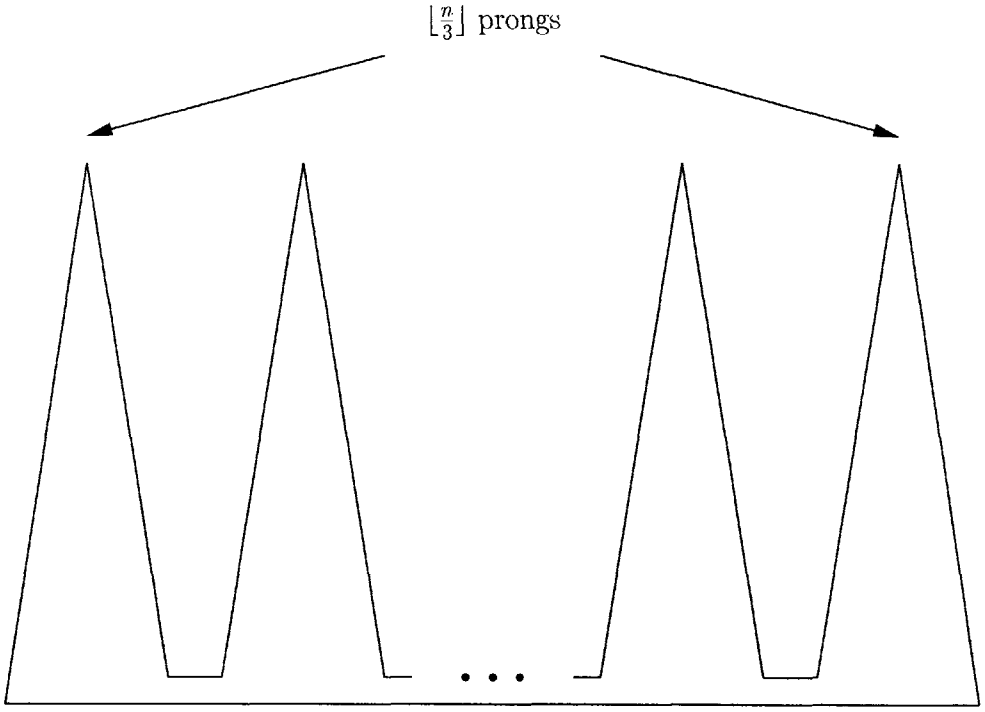

Figure 2.2: *Comb* shaped art gallery requires $\lfloor \frac{n}{3} \rfloor$ guards

2.2.2 Ghosh's $O(\log n)$-Approximation Algorithm for the Vertex-Guard Problem

Ghosh's paper [8] is the first known work in the area of approximation algorithm for art gallery problems. He proposes an $O(\log n)$-approximation algorithm for the vertex-guard problem. Recall that in the vertex-guard problem, guards are restricted to be located only

at the vertices of P. Our description of Ghosh's algorithm is different from his original paper. We explain his algorithm in terms of a concept – *visibility cell decomposition* – described independently by Bose *et al.* [1] and Guibas *et al.* [10]. We use this notion in our algorithm and throughout the thesis. We think it is appropriate to introduce it here. Moreover, the treatment of Ghosh's algorithm using this notion is clear and concise. Also, this will help readers to understand similarities and differences between algorithms reviewed in this section and our algorithm.

Definition 2.2.1. *The* visibility cell decomposition *of P is a subdivision induced by visibility polygons of all the vertices of P. We call each component of the subdivision a* visibility cell.

Below we state without proofs two properties of the visibility cell decomposition that are useful in the analysis of the algorithm. These are proved in [1, 10]. We provide their proofs in Chapter 4.

- Each visibility cell is a convex polygon.
- The total number of visibility cells in the visibility cell decomposition is $O(n^3)$.

Below we summarize steps of the algorithm proposed by Ghosh using the terminology described above.

- **Step 1:** Construct visibility polygon of each vertex and generate visibility cell decomposition of P.
- **Step 2:** Enumerate visibility cells. For each vertex v of P, construct a set C_v which contains indices of visibility cells that are contained in the visibility polygon of v, $V(v)$.
- **Step 3:** Formulate the set cover problem of finding minimum number of vertices $v_1, v_2, ..., v_k$ such that the set $C_{v_1} \cup C_{v_2} ... \cup C_{v_k}$ includes all the visibility cells. In other words, find optimal cover of all visibility cells. Solve this problem approximately using greedy approach as in [5].

In his original paper [8], Ghosh prove that the total number of visibility cells is $O(n^4)$. However, as mentioned above Bose *et al.* [1] and Guibas *et al.* [10] prove that the number of visibility cells is $O(n^3)$. The running time of Step 1 is $O(n^3)$. Since there are n vertices and the size of set cover problem is $O(n^4)$. The set cover problem is solved approximately in $O(n^4 \log n)$ time with the approximation ratio $O(\log n)$. The running time in the original paper is $O(n^5 \log n)$.

2.2.3 Heuristic proposed by Banos and Latombe

Banos and Latombe consider a variant of the point-guard problem [9]. Their problem is inspired by a real-world application of capturing texture data of the surrounding environment in a workspace. They assume that the workspace is represented as a 2-D art gallery P. Their set up involves mobile robots each with a mounted camera having range and incidence constraints. According to range constraints, a camera is able to capture image of an object clearly only when it is at least distance d_{min} away and at most within distance d_{max}. According to incidence constraints, a point in the walls is captured by a camera only if the angle between normal at the point and line between the camera and the point is within certain range. Since they use these cameras to capture texture data of the surrounding, they focus on the problem of covering only the walls of the art gallery unlike our problem. In their set up, they calculate potential locations of the cameras such that every point on the walls of the art gallery is captured by at least one camera with range and incidence constraints. Then the mobile robots are sent to these locations and the cameras mounted on them capture 3D texture data of their surroundings. In this case, the definition of the *visibility set* can be modified as follows. The *visibility set* of a point $x \in P$ is defined as the set of points on $bd(P)$ visible to x that are within range and incidence constraints imposed. Below we outline algorithm proposed by Banos and Latombe. We refer readers to [9] for additional details.

- **Step 1:** Sample P uniformly at random m times. The sample m points are potential candidates for locating cameras.

- **Step 2:** For every sampled points, construct visibility set by taking range and incidence constraints into account. This induces subdivision of $bd(P)$. Enumerate all the segments of the subdivision.

- **Step 3:** For each sampled point construct a set of segments visible from it. Formulate the set cover problem of covering all the segments of the subdivision. Solve the set cover problem approximately using *VC-dimension* based algorithm.

Now we describe running time analysis of the algorithm without details. The running time of Step 1 of the algorithm is $O(n \log n + m)$. The sampling procedure is carried out by triangulating P and then sampling each triangle depending upon its relative area. The visibility set of each point is constructed in $O(n \log n)$ time. Step 2 is carried out in $O(mn \log n)$ time and the total number of segments induced by the subdivision is $O(mn)$. The set cover problem if solved using greedy approach as in [5], it requires the running time $O(mn \log mn)$. However, the approximation ratio is $O(\log mn)$ which depends on the size of the sampling. Hence, Banos and Latombe suggest an alternate *VC-dimension* based approach to solve the set cover problem approximately. They prove that the *VC-dimension* of the dual of the set cover problem, the *hitting set problem*, is $O(\log n)$. Thus, the algorithm computes with *high probability* the solution with the approximation ratio $O(\log n \log (c \log n))$ where c is the optimal number of guards required. We define VC-dimension and hitting set problem in Chapter 6. We also prove a similar bound on VC-dimension in the context of our problem.

Note that the algorithm suggested by Banos and Latombe is probabilistic in nature and does not provide any guarantees on the solution. In fact, they assume that for most problems, the optimal solution has certain *elasticity*, i.e., the solution does not change if location of each guard in the optimal solution is perturbed slightly. They indicate σ as the normalized area of the largest disc around each optimal guard location that characterizes the elasticity. If c indicates the optimal number of guards required, then the probability that at least one sample falls in a disc around each optimal guard location is $(1 - (1 - \sigma)^m)^c$ [9]. If number of samples m is large enough, then this probability quickly approaches 1. Thus, for the class of art galleries which admit such *elasticity* in the solution, the algorithm returns an approximate

solution with high probability. However, this assumption needs rigorous verification. Many art galleries do not admit such elasticity. The authors dismiss such situations which require perfect positioning, because from practical standpoint, it is not possible to achieve perfect positioning due to sensor limitations.

2.2.4 Inapproximability Results for Art Gallery Problems

In this section, we outline results proposed by Eidenbenz *et al.* in [7] and Brodén *et al.* in [2] on inapproximability results for the point-guard problem. We do not produce proofs of those results here and refer interested readers to [7, 2] for details. Eidenbenz *et al.* in [7] show that the point-guard problem for art galley without holes is *APX-hard*, i.e., no polynomial time algorithm can achieve an approximation ratio of $1 + \delta$ for a constant $\delta > 0$ for the art gallery without holes unless $P = NP$. In other words, this result implies that if P is not equal to NP, it is not possible to produce a polynomial time algorithm that yields constant approximation ratio. They also prove that the problem of art gallery with holes can not be approximated by a polynomial time algorithm with ratio $(\frac{1-\epsilon}{12}) \ln n$ for any $\epsilon > 0$, unless $NP \subseteq TIME(n^{O(\log \log n)})$. This result implies that if NP is not subset of $TIME(n^{O(\log \log n)})$, no polynomial time algorithm exists that can approximate the solution with the approximation ratio better that $O(\log n)$. In [2] Brodén *et al.* prove that the point-guard problem even for special class of art galleries, which are *2-link polygons*, is *APX-hard*. *k-link polygon* is a polygon in which the shortest path restricted to the polygon between any two points in the polygon consists of at most k line segments. The result implies that even for this special class of polygons, the solution cannot be approximated beyond a constant factor using any polynomial time algorithm unless $P = NP$.

Summary: In this chapter, we developed basic terminology and formally stated our problem. We mentioned that throughout this thesis we will focus on the case of the art gallery without holes and will comment about the case of art gallery with holes at the end. Then we reviewed related work in detail. In particular, we stated two related algorithms by Ghosh

and Banos and Latombe using the terminology developed earlier. This will help readers to understand similarities and differences between our and their algorithms. In the next chapter, we describe our algorithm.

Chapter 3

Our Algorithm

In this chapter, we outline our algorithm and describe each step of our algorithm in little more detail. Here we stress on giving intuition behind each step rather than precise mathematical details. Mathematical formulations follow in the later chapters. The basic idea behind our algorithm is to triangulate the art gallery such that each triangle in the triangulation satisfies a special visibility property -- *vertex-visibility property.* According to this property, for any triangle in the triangulation, a region that is visible to any point in the triangle is always *seen* or *covered* by three vertices of the triangle. We select the vertices of the triangulation to be the potential candidates for locating guards and formulate the new problem of finding minimum number of guards only from these selected points. *Vertex-visibility property* guarantees that the new problem has an optimal solution at most three times the optimal solution to the original point-guard problem. We further reduce the new problem to a set cover problem which can be solved approximately. We begin by defining the *vertex-visibility property.*

3.1 Vertex-Visibility Property

First two steps of our algorithm involve partitioning P into triangles such that each triangle in the triangulation satisfies a special visibility property — vertex-visibility property.

Definition 3.1.1. *Let $\triangle abc$ be a triangle in the polygon P. We say $\triangle abc$ satisfies the* vertex

visibility property, *if for any point* $x \in \triangle abc$, $V(x) \subseteq V(a) \cup V(b) \cup V(c)$.

In other words, a triangle satisfies the vertex-visibility property if the region visible from any point in the triangle is a subset of the total region visible from three vertices of the triangle.

3.2 Our Algorithm

Our algorithm can be summarized as follows.

Input : Input to the algorithm is a representation of P. We assume that P is represented as an array of x and y coordinates of n vertices of P.

- **Step 1:** Construct *visibility cell decomposition* of P and obtain *initial triangulation* of P by triangulating each visibility cell.
- **Step 2:** Check if each triangle in the initial triangulation satisfies the vertex-visibility property. If it does not, then subdivide it using a recursive *special subdivision procedure* to obtain the *final triangulation.* Each triangle in the final triangulation satisfies the vertex-visibility property.
- **Step 3:** Construct visibility polygon of each vertex $v_1, v_2, ..., v_N$in the *final triangulation.* This induced a subdivision of P. Enumerate each element of the subdivision of P. For each vertex v, construct the set C_v of elements of subdivision in its visibility polygon $V(v)$. Formulate the set cover problem of finding minimum number of vertices $t_1, t_2, ..., t_k$ such that the set $C_{t_1} \cup C_{t_2} ... \cup C_{t_k}$ includes all the elements of subdivision of P. Solve the set cover problem approximately using *VC-dimension* based algorithm.

Output: A near-optimal set of guard locations in terms of x and y coordinates.

Now we describe each step of algorithm in more detail with its running time.

- **Step 1:** As mentioned in the previous chapter, *visibility cell decomposition* is induced by constructing visibility polygon of each vertex of P. Each visibility cell is a convex

polygon. We triangulate each visibility cell simply by adding all diagonals from one particular vertex of the visibility cell. Thus, we obtain *initial triangulation* of P. The visibility cell decomposition consists of $O(n^3)$ cells that can be constructed in $O(n^3)$ time [1, 10]. We prove this result in the next chapter. Since a visibility cell has at most $2n$ sides in the worst case, the initial triangulation can be generated in $O(n^4)$ time and consists of $O(n^4)$ triangles.

- **Step 2:** Some triangles in the initial triangulation may not satisfy the vertex-visibility property defined earlier. For each such triangle $\triangle$ that does not satisfy this property, we partition $\triangle$ using *special subdivision procedure* into at most six smaller triangles. The special subdivision procedure guarantees that at most one triangle among these does not satisfy the vertex-visibility property and the rest do. We invoke the special subdivision procedure on the triangle that does not satisfy the property and proceed in the recursive fashion. Thus we obtain *final triangulation* at the end of Step 2 such that each triangle satisfies the vertex-visibility property. We formally describe this procedure in Chapter 5. We show that on each triangle that does not satisfy the vertex visibility property, the special subdivision procedure runs recursively in $O(K)$ time. It induces $O(n^2K^2)$ triangles in a triangle in the initial triangulation. Thus the final triangulation consists of $O(n^6K^2)$ triangles. We later show that K depends on the representation of P and in the worst case it is *pseudo-polynomial.* Suppose the maximum number of bits required to represent a particular coordinate in the representation of P is $\log(\frac{1}{\epsilon})$, then we show that K is a polynomial function of n and $(\frac{1}{\epsilon})$. Thus the running time of this step is *pseudo-polynomial*, i.e. polynomial in the number of vertices of P but exponential in the bit-representation of coordinates of the vertices.

- **Step 3:** In this step, we essentially solve a new problem where guard locations are restricted to the vertices of the final triangulation only. We construct visibility polygon of each vertex of the final triangulation. This induces a new subdivision of P. We enumerate each element of this subdivision and label the vertices. For each vertex,

we construct a set of elements of subdivision that form the visibility polygon of the vertex. We then formulate the set cover problem of covering P using sets of vertices of the final triangulation. The number of triangles and in turn the number of vertices in the final triangulation is $O(n^6K^2)$. To construct the visibility polygon of a point, we may have to introduce $O(n)$ new edges. Therefore, the final subdivision obtained by constructing visibility polygons of all the vertices of the final triangulation may consist of $O(n^{14}K^4)$ cells. If we use greedy approach to solve the set cover problem as in [5], it runs in $O(n^{14}K^4 \log K)$ time. But it gives the approximation ratio $O(\log K)$ which does not depend on n alone. Hence, we use a *VC-dimension* based algorithm to solve the set cover problem approximately which yields the approximation ratio $O(\log n \cdot \log(c \log n))$, where c is the optimal number of guards and $c = O(n)$ in the worst case. Therefore, overall we get a solution with the approximation ratio $O(\log^2 n)$, which depends only on the number of points in the art gallery and does not depend on the parameter K. We formally prove this in Chapter 6. The result depends on a theorem where we show that the VC-dimension of the set cover problem is $O(\log n)$.

Summary: In this chapter, we explained three steps of our algorithm. We also stated running time bounds on each state and approximation ratio analysis without proof. In the following three chapters, we formally state each step of the algorithm, prove the correctness results wherever necessary and show running time bounds.

Chapter 4

Visibility Cell Decomposition

The first step of our algorithm involves obtaining the visibility cell decomposition of P and triangulating each visibility cell. In this chapter, we formally prove that each visibility cell is a convex polygon and the number of visibility cells is $O(n^3)$. Moreover, we mention without proof additional visibility properties of a visibility cell which are important in proving correctness of the algorithm. The results that we describe here are proved independently in [1, 10] in different contexts. We repeat proofs of relevant results using the terminology we developed earlier. We refer readers to these papers for additional details.

4.1 Visibility Cell Decomposition

Earlier, in Chapter 2, we defined some basic terminology. We repeat some of it here for the sake of simplicity. Recall that for $x \in P$, the visibility polygon $V(x)$ is the polygon consisting of all the points in P that are visible from x. Note that some of the edges of $V(x)$ coincide with those of the original polygon P and some are newly introduced. A new edge is introduced at a reflex vertex of P that blocks the view of x. We call this reflex vertex a blocking reflex vertex. To remove any ambiguities, we assume that for P and $V(x)$, no two consecutive edges are collinear. We repeat definition of *visibility cell decomposition* below.

Definition 4.1.1. *The* visibility cell decomposition *of P is a subdivision induced by visibility*

polygons of all the vertices of P. We denote it as $VCD(P)$ We call each component of the subdivision a visibility cell.

In $VCD(P)$, we construct visibility polygon of each vertex. As mentioned above, some of the edges of visibility polygons are part of edges of P, while some are newly introduced. We prove following lemma related to number of such newly introduced edges.

Lemma 4.1.1. *The number of newly introduced edges in $VCD(P)$ is $O(n^2)$.*

Proof. In a visibility polygon of a vertex, a new edge is introduced at a blocking reflex vertex. There are $O(n)$ reflex vertices in the worst case. P has n vertices. Hence the bound follows. □

Now we prove a theorem about convexity of a visibility cell [10].

Theorem 4.1.1. *Each visibility cell in $VCD(P)$ is a convex polygon with at most $2n$ sides.*

Proof. First we provide an indirect proof for the fact that a visibility cell is a convex polygon. Suppose C is a visibility cell in *VCD(P)*. Since all the edges in *VCD(P)* are straight line segments, C is a polygon. Now suppose C is a non-convex polygon and let v be a reflex vertex on $bd(C)$. Note that v cannot be a vertex of P, because if that is the case, then the edges incident at v will be extended into interior lines and that would subdivide C. Also note that v cannot be on $bd(P)$. So v has to be in $int(P)$. However, this is not possible, because v has to be formed by intersection of two newly introduced edges, each of which has endpoints on $bd(P)$. This gives the contradiction.

Now we prove second part of the theorem that C has at most $2n$ sides. C has some edges that are part of edges of P or some edges that are newly introduced. Each vertex of P is either visible or invisible from $int(C)$. Moreover each vertex introduces at most two new edges that can contribute to form sides of C because the art gallery does not contain holes. Thus C can have at most $2n$ sides. □

Next we prove a bound on a number of visibility cells in *VCD(P)* [10].

Theorem 4.1.2. *The number of visibility cells in $VCD(P)$ is $O(n^3)$.*

Proof. According to Lemma 4.1.1 above, the number of newly introduced edges in $VCD(P)$ is $O(n^2)$. Consider one such edge with endpoints a and b on $bd(P)$. We intend to identify number of other newly introduced edges intersecting ab. Imagine that we walk starting from a towards b along ab and observe set of vertices of P visible from each point on ab. During our walk, there can be at most $2n$ changes in the set of visible vertices. This is because, during the walk, once a vertex becomes invisible, it never becomes visible again or vice-versa. Otherwise it implies that P contains holes. This transition of a vertex being visible or invisible occurs due to a newly introduced edge by the visibility polygon of the vertex. Therefore, at most n newly introduced edges intersect ab. Since there are $O(n^2)$ newly introduced edges, total number of intersection points in $VCD(P)$ is $O(n^3)$. Since $VCD(P)$ is a planar graph, by Euler's theorem for a planar graph, the number of faces i.e. the number of visibility cells is $O(n^3)$. □

Notice that the bounds on the number of visibility cells, $O(n^3)$ is tight. An example of a star shaped polygon where this bound holds tight is shown in [1]. An algorithm is presented in [1] to construct and store $VCD(P)$ in $O(n^3)$ time.

4.2 Visibility Properties of a Visibility Cell

In this section, we describe visibility properties from a visibility cell. We mention these properties without proofs. We refer readers to [1, 10] for details.

First we repeat some basic terminology we defined in Chapter 2. For any point $x \in P$, we say that x sees an edge of P, if it sees a point on the edge. If x cannot see either of the end-points of a visible edge of P, we say that x sees the edge partially. We call the corresponding edge of P a partial edge with respect to x. We say that x sees a visible edge of P non-partially, if it sees at least one of its end-points. We call the corresponding edge of P a non-partial edge with respect to x. If we join every vertex of $V(x)$ to x, we get a triangulation of $V(x)$. We call each triangle as a visibility sector of x. The edge of a visibility sector that is a part of an edge of P is referred to as a base of the visibility sector. Depending upon the

type of the edge of P corresponding to the base of a visibility sector, we classify the visibility sector into non-partial-edge sector or partial-edge sector.

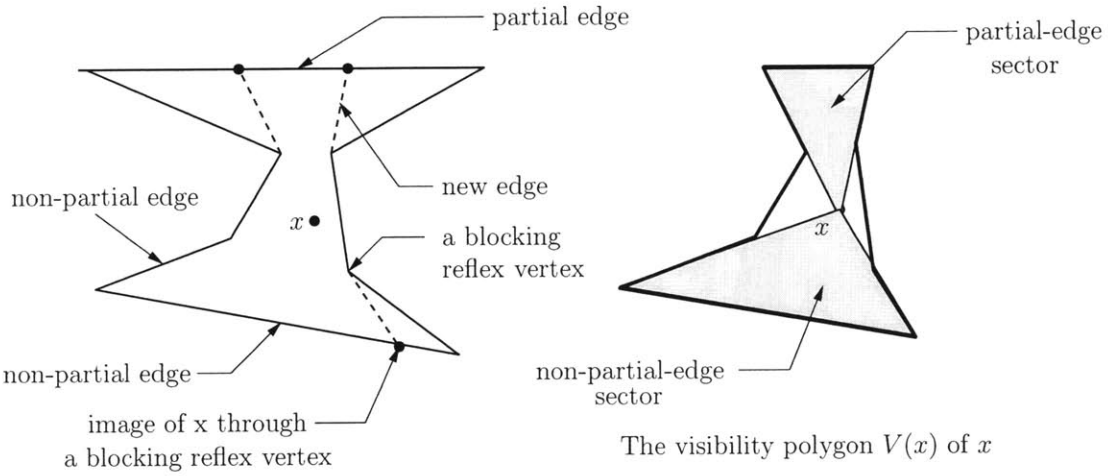

Figure 4.1: Visibility polygon and visibility sectors

Next we mention visibility properties from a visibility cell without proof.

Theorem 4.2.1. *By definition, any two points in a visibility cell see the same set of vertices of P. Any two points in the same visibility cell see the same set of non-partial edges and the same set of partial edges of P.*

Summary: In this chapter, we proved that the number of visibility cells in $VCD(P)$ is $O(n^3)$ and we also showed that each visibility cell is a convex polygon. In addition, we summarized visibility properties of a visibility cell. Step 1 of our algorithm involves just constructing visibility cell decomposition and triangulating each visibility cell by adding all diagonals from one particular vertex of the cell. In general, each triangle in the triangulation obtained this way need not satisfy the vertex-visibility property. In the next chapter, we describe Step 2 of our algorithm which further subdivides such triangles to obtain final triangulation where each triangle satisfies the vertex-visibility property.

Chapter 5

Further Subdivision

In the previous chapter, we described how we obtain initial triangulation from the visibility cell decomposition of P. In this chapter, we explain Step 2 of our algorithm which is a subdivision procedure that further subdivides triangles from the initial triangulation which do not satisfy the vertex-visibility property. The vertex-visibility property is not directly useful in the construction of our algorithm. Hence, first we define another visibility property — *vertex-pair-visibility property*. Then we prove various results related to this property that are useful in proving correctness of the algorithm. Next we give the algorithm for Step 2 and prove the correctness of the algorithm. Finally we identify a special structure – *pinhole* – in the art gallery that leads to the pseudo-polynomial behavior of the algorithm.

5.1 Further Subdivision of the Initial Triangulation

In this section, we describe Step 2 of our algorithm. We give a procedure to subdivide the initial triangulation in such a way that each triangle in the final triangulation satisfies the vertex-visibility property. In subsection 5.1 we define another visibility property and prove related theorems that are useful in the construction of Step 2 which we describe in subsection 5.2.

5.1.1 Vertex-Pair-Visibility Property

Each triangle in the final triangulation is required to satisfy the vertex-visibility property — the region that is visible to any point in a triangle is covered by the visibility polygons of the three vertices of the triangle. Covering the visibility polygon of a point is equivalent to covering every visibility sector of the point. This motivates the following definition.

Definition 5.1.1. *A triangle in a visibility cell satisfies the* vertex-visibility property with respect to a particular edge *of the polygon, if the corresponding visibility sector of any point in the triangle is a subset of the union of the visibility polygons of the vertices of the triangle.*

The vertex-visibility property is not directly useful in the construction of our algorithm. We define a more convenient property.

Definition 5.1.2. *A triangle in a visibility cell satisfies the* vertex-pair-visibility property with respect to a particular edge *of the polygon, if the visibility sectors of any two vertices of the triangle overlap on the edge.*

Below we prove two theorems related to the vertex-visibility and vertex-pair-visibility property of a triangle in a visibility cell. We first give a few definitions and prove a lemma that is useful in the proofs of the theorems.

Consider the images of two points in a visibility cell through a blocking reflex vertex on an edge of the polygon. We call the portion of the edge between the two images as a span of the two points corresponding to the blocking reflex vertex. Now consider the images of the three vertices of a triangle in a visibility cell through a blocking reflex vertex on an edge of the polygon. One of the three images lies between the other two. We call the portion of the edge between the two extreme images as a span of the triangle through the blocking reflex vertex. Figure 5.1 illustrates this notion. The images of the vertices of $\triangle abc$ through the blocking reflex vertex r are shown in the figure. The span of the triangle in this case is the line segment a_1c_1.

Lemma 5.1.1. *For any point in a triangle in a visibility cell, its image through a blocking reflex vertex always lies in the span of the triangle through the blocking reflex vertex.*

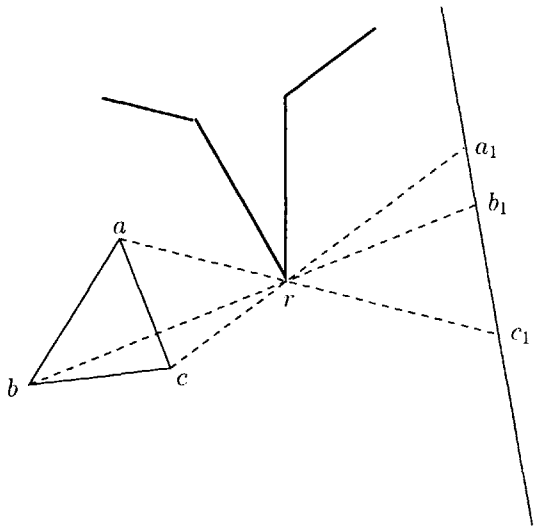

Figure 5.1: Span of $\triangle abc$ through the blocking reflex vertex r is the line segment a_1c_1.

Proof. Let $\triangle abc$ be a triangle in a visibility cell and let r be the blocking reflex vertex. Consider a point x in $\triangle abc$. Let x' be its image through r. x', r and x are collinear. Suppose that x' does not lie in the span of $\triangle abc$ through r. Then, if we draw a line through x', r and x, it does not intersect $\triangle abc$, which is contradictory to our assumption that x lies in $\triangle abc$. Therefore, for any x in $\triangle abc$, the image of x through r always lies within the span of $\triangle abc$ through r. □

Theorem 5.1.1. *A triangle in a visibility cell satisfies the vertex-pair-visibility as well as the vertex-visibility property with respect to a non-partial edge.*

Proof. Let $\triangle abc$ be a triangle in a visibility cell C. Let e be a non-partial edge. As we have already seen, at least one of the end-points of a non-partial edge is visible from any point in a visibility cell. Depending on whether one or both the end-points of a non-partial edge are visible, we make two cases and deal with each case separately.

- Case 1: Both the end-points of e are visible from any point in C. In this case, by definition. $\triangle abc$ satisfies the vertex-pair-visibility property. Let u and v be the end-points of e. Consider the convex hull of a, b, c, u and v. Since $\triangle abc$ is on one side of

e, line segment uv must be one of the edges of the convex hull. Therefore, the convex hull can also be formed by considering the union of $\triangle abc$ and the visibility sectors of a, b and c. Note that the convex hull is a subset of $V(a) \cup V(b) \cup V(c)$ and the visibility sector of any point $x \in \triangle abc$ is a subset of this convex hull. Therefore, $\triangle abc$ also satisfies the vertex-visibility property with respect to e.

- Case 2: In this case, only one end-point of e is visible from any point in C. Let u be the visible end-point. Let r be a blocking reflex vertex. Again by definition $\triangle abc$ satisfies the vertex-pair-visibility property because u is a common visible point. Now, consider any point x in $\triangle abc$. The visibility sector of x with respect to e consists of two triangles, $\triangle xur$ and $\triangle urx'$, where x' is the image of x through r. By similar arguments as in the first case, we can prove that $\triangle xur$ is a subset of $V(a) \cup V(b) \cup V(c)$. By lemma 5.1.1, x' lies in the span of the image of $\triangle abc$ through r. Thus, at least one of a, b or c cover $\triangle rux'$. Therefore, $\triangle abc$ satisfies the vertex-visibility property with respect to e.

□

Theorem 5.1.2. *If a triangle in a visibility cell satisfies the vertex-pair-visibility property with respect to a partial edge, then it also satisfies the vertex-visibility property with respect to the partial edge.*

Proof. Let $\triangle abc$ be a triangle in a visibility cell C such that it satisfies the vertex-pair-visibility property with respect to a partial edge. Let e be the partial edge and r_1 and r_2 be the two blocking reflex vertices. Since the visibility sectors of any two vertices of $\triangle abc$ overlap on e, the spans of any two vertices of triangle through r_1 and r_2 do not overlap on e. The spans of $\triangle abc$ also do not overlap on e. The portion of e that is simultaneously visible to a, b and c consists of the spans of $\triangle abc$ through r_1 and r_2 and the patch between the two spans. By lemma 5.1.1, for any point x in $\triangle abc$, the two images of x through r_1 and r_2 lie in the spans of $\triangle abc$ through r_1 and r_2 respectively. Thus, the portion of e that is visible to x is a subset of the portion that is visible to a, b and c. Therefore, the visibility sector of x is a subset of $V(a) \cup V(b) \cup V(c)$. □

The theorem we prove below is useful in the analysis of the algorithm.

Theorem 5.1.3. *If a triangle in a visibility cell satisfies the vertex-pair-visibility property with respect to a partial edge, then any subtriangle also satisfies the vertex-pair-visibility property with respect to the partial edge.*

Proof. Let $\triangle abc$ be a triangle in a visibility cell C such that it satisfies the vertex-pair-visibility property with respect to a partial edge. Let e be the partial edge and r_1 and r_2 be the two blocking reflex vertices. We already proved in the proof of theorem 5.1.2 that the spans of $\triangle abc$ through r_1 and r_2 do not overlap on e because it satisfies the vertex-pair-visibility property. For any two points x and y in $\triangle abc$, the spans of x and y through r_1 and r_2 do not overlap on e because they are contained in the spans of $\triangle abc$ through r_1 and r_2. Therefore, the visibility sectors of x and y overlap on e. Therefore, any $\triangle xyz$ in $\triangle abc$ satisfies the vertex-pair-visibility property. □

The above theorem allows us to further subdivide the visibility cell without affecting already existent vertex-pair visibility property with respect to a partial-edge visibility sector. This will become clear after the description of our subdivision procedure in the next subsection.

5.1.2 Further Subdivision

In this subsection, we give a procedure to further subdivide the initial triangulation obtained in Step 1 of our algorithm. The subdivision procedure described below generates the final triangulation where every triangle satisfies the vertex-visibility property. This property is required so that we can reduce the art gallery problem to a problem with guaranteed approximation ratio. By virtue of Theorem 5.1.1 and Theorem 5.1.2, we achieve this by developing a subdivision procedure which is based on a stronger condition, the vertex-pair-visibility property.

First we define a notion that is useful in the description of our algorithm. Let a and b be two points in a visibility cell such that the visibility sectors of a and b do not overlap

on a partial edge. Let r_1 and r_2 be the corresponding blocking reflex vertices. Consider the convex hull of a, b, r_1 and r_2. We call a triangle obtained by taking set difference between the convex hull and the union of the visibility sectors of a and b as a dark triangle of segment ab. An example of a dark triangle is shown in figure 5.2(a).

Step 2 of our algorithm can be summarized as follows.

For every $\triangle abc$ in the initial triangulation obtained in Step 1, repeat the following procedure:

1. Construct a set S of partial edges for which $\triangle abc$ does not satisfy the vertex-pair-visibility property. Repeat the following procedure for every edge $e \in S$:

   (a) Construct a dark triangle of every edge of $\triangle abc$.

   (b) For each dark triangle whose interior is not disjoint with $\triangle abc$, invoke *SUBDIVIDE-DARK-TRIANGLE*.

   (c) Intersect with $\triangle abc$, the subdivisions of all such dark triangles on which the function *SUBDIVIDE-DARK-TRIANGLE* is invoked in the above step to generate a new subdivision of $\triangle abc$.

2. Intersect all the subdivisions of $\triangle abc$ corresponding to every edge $e \in S$ to generate the final subdivision. Triangulate the final subdivision in the similar way as in Step 1 of our algorithm and return the final triangulation of $\triangle abc$.

Function *SUBDIVIDE-DARK-TRIANGLE*:

Input: A dark triangle $\triangle aob$ corresponding to the two blocking reflex vertices r_1 and r_2

Procedure: Let a_1b_1 and a_2b_2 be the two spans of ab through r_1 and r_2 respectively on the partial edge. Construct a line joining the reflex vertex r_2 and the image a_1 of a through r_1 and another line joining the reflex vertex r_1 and the image b_2 of b through r_2. Depending upon whether the two lines intersect inside or outside $\triangle aob$, choose one of the following two steps.

(**Case 1**) The two lines meet outside $\triangle aob$: Return the new subdivision of $\triangle aob$ induced by the two lines (figure 5.2(a)). Terminate the function.

(**Case 2**) The two lines meet in $\triangle aob$: Return the new subdivision of $\triangle aob$ without $\triangle a'o'b'$, where o' is the point of intersection of the two lines, and a' and b' are the points of intersection of the two lines with the segment ab. Check if $\triangle a'o'b'$ satisfies the vertex-pair-visibility property. If it does not, invoke *SUBDIVIDE-DARK-TRIANGLE* on $\triangle a'o'b'$. (figure 5.2(b))

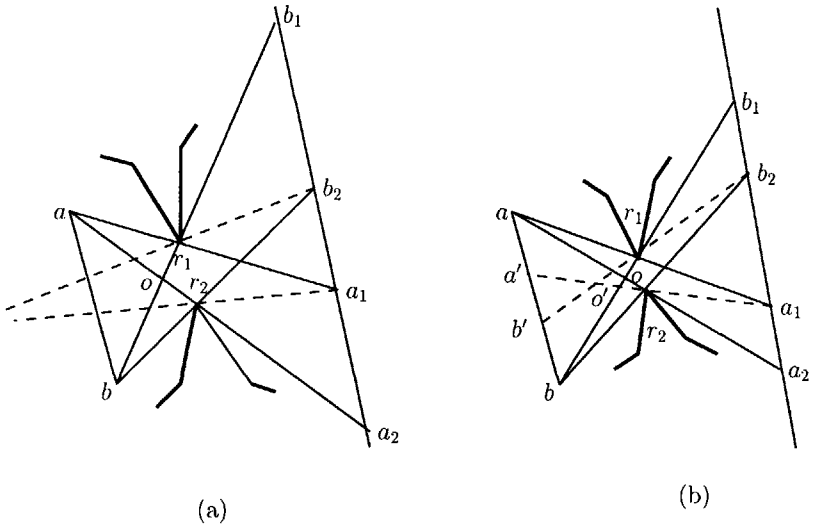

Figure 5.2: $\triangle aob$ is a dark triangle. Two cases in *SUBDIVIDE-DARK-TRIANGLE*: (a)lines a_1r_2 and b_2r_1 meet outside $\triangle aob$ (b) lines a_1r_2 and b_2r_1 meet in $\triangle aob$

As a result of Theorem 5.1.1 and Theorem 5.1.2, in our subdivision procedure, we need to subdivide a triangle only if it does not satisfy the vertex-pair-visibility property with respect to a partial edge. The result of our subdivision procedure is the final triangulation where every triangle satisfies the vertex-pair visibility property and in turn, the vertex-visibility property. We prove the correctness of this result in the next section.

5.2 The Correctness of Step 2 of the Algorithm

In this section, we show that every triangle in the final triangulation obtained at the end of step 2 satisfies the vertex-pair visibility property. As we have already mentioned, by virtue of Theorem 5.1.1 and Theorem 5.1.2, we check whether a triangle in the initial triangulation

satisfies the vertex-pair-visibility with respect to partial edges only. Now we prove the following lemma. By the virtue of Theorem 5.1.3, the subdivision procedure of a triangle with respect to one edge is 'independent' of the subdivision procedure with respect to another edge. This allows us to subdivide a triangle in the edge-by-edge fashion. Now we prove the correctness of the subdivision procedure with respect to one edge.

Lemma 5.2.1. *Consider the partial-edge visibility sector of a point in a visibility cell. Any triangle that lies in the visibility sector as well as the visibility cell always satisfies the vertex-pair-visibility property.*

Proof. Let x be a point in a visibility cell C. Let r_1 and r_2 be the blocking reflex vertices corresponding to the partial edge. Let x_1 and x_2 be the images of point x through r_1 and r_2 respectively. Any point a that lies in the visibility sectors of x as well as in the same visibility cell C, sees the line segment x_1x_2. Therefore, by definition, any triangle that lies in the visibility sector of x as well as in C satisfies the vertex-pair-visibility property. □

Let $\triangle abc$ be a triangle in the initial triangulation. Suppose that it does not satisfy the vertex-pair-visibility property with respect to a partial edge. Consider the convex hull of a, b, c, r_1 and r_2. The convex hull can also be obtained by taking union of the visibility sectors of a, b and c and the dark triangles of all the edges of $\triangle abc$. By Lemma 5.2.1, the portions of $\triangle abc$ that lie in the visibility sector of any of the vertices satisfies the vertex-pair-visibility property. The remaining part of $\triangle abc$ is a subset of the union of the dark triangles. Therefore, in our subdivision procedure in step 2, we just subdivide the dark triangles.

Now we prove the correctness of the function *SUBDIVIDE-DARK-TRIANGLE* with reference to figure 5.2

Theorem 5.2.1. *In the first case, the subdivision of $\triangle aob$ satisfies the vertex-pair-visibility property.*

Proof. Consider line a_1r_2. It subdivides $\triangle aob$ into two part. a_1 is always visible from any point in one part. Therefore that always satisfies the vertex-pair visibility property. Similarly

line b_2r_1 subdivides $\triangle aob$ in two parts out of which one part always satisfies the vertex-pair-visibility property because b_2 is the common visible point from that part. In the first case lines a_1r_2 and b_2r_1 meet outside $\triangle abc$. Both the parts of mentioned above that satisfy the vertex-pair-visibility property cover $\triangle aob$ in the first case. Therefore, the subdivision of $\triangle aob$ satisfies the vertex-pair-visibility property. □

Theorem 5.2.2. *In the second case, the subdivision of $\triangle aob$ except $\triangle a'o'b'$ satisfies the vertex-pair-visibility property.*

The proof of the above Lemma is similar to the proof of Lemma 5.2.1. $\triangle a'o'b'$ may not satisfy the vertex-pair-visibility property. In that case, we subdivide $\triangle a'o'b'$ by again invoking the function *SUBDIVIDE-DARK-TRIANGLE*. The first case is the termination case for the recursion in *SUBDIVIDE-DARK-TRIANGLE*. In the next section, we show that *SUBDIVIDE-DARK-TRIANGLE* indeed terminates. Thus, subdivision generated by *SUBDIVIDE-DARK-TRIANGLE* always satisfies the vertex-pair-visibility property.

5.3 Pinholes in the Art Gallery

The function *SUBDIVIDE-DARK-TRIANGLE* in the subdivision procedure described above is recursive. Here, we address the question after how many steps this recursion ends. It turns out that the number of steps of the recursion depends upon a parameter K which is polynomial in the size of the unary encoding of the art gallery. Sometime K can be exponential in the binary encoding of the art gallery, that is in the input size. Below we discuss an example of the art gallery, where the number of steps of the recursion is arbitrarily large. We identify such cases as the *pinholes* in the art gallery. We also discuss how to identify parameter K. We use this parameter later to analyze the running time of our algorithm.

We explain the notion of a *pinhole* with the help of an example. Consider the art gallery as shown in figure 5.3. Let p and q be the two blocking vertices corresponding to the partial edge e. The arrangement is symmetric with respect to X and Y-axis as shown in the figure. ab, pq and the edge e are parallel to Y-axis. By similarity of triangles, we can easily show

that $\frac{L_1}{L_2} = \frac{D_1}{D_2}$. Suppose that we apply our algorithm to this art gallery to get a solution to the point-guard problem. In figure 5.3, we have specifically chosen the dimensions of the art gallery in such a way that, in Step 2 of our algorithm, we have to recursively subdivide the dark triangle of segment ab with respect to p and q. In figure 5.3, the dark triangle of ab is not shown explicitly, but the lines generated in the subdivision procedure are shown. Again by similarity of triangles, we can show that the number of steps of the recursion in the subdivision procedure for the dark triangle of ab is $O((\frac{L_1 L_2}{L_1+L_2})\epsilon)$. Consider a simple case where we choose L_1=L_2=1 and $\frac{1}{\epsilon} = 2^{10}$. In this case, the binary encoding of the art gallery requires roughly 10 bits. But the number of subdivisions is exponential in the size of the input. The number of subdivisions are of the order of the actual number 2^{10}. In other words, the number of subdivisions is of the order of the size of the unary encoding of 2^{10}. On the other hand, if we choose $L_1 = L_2 = 1$ and $\frac{1}{\epsilon} = 2^3$, then the number of bits in the input and the number of subdivisions are comparable. Note that the art gallery consists of 6 points. The unary encoding of the art gallery in this case is of the order of the number of points in the art gallery. In general, the number of subdivisions ($O((\frac{L_1 L_2}{L_1+L_2})\epsilon)$) is of the order of the sum of the sizes of the unary encodings of L_1, L_2 and $\frac{1}{\epsilon}$. Depending upon the choices of the values of L_1, L_2 and $\frac{1}{\epsilon}$, the number of subdivisions could be exponential or of the order of the size of the binary encoding of the art gallery.

Now, consider the general case of the art gallery with n walls. Note that the size of the binary encoding of the art gallery contains information about n implicitly. n is always less than the size of the binary encoding. In our algorithm, sometime we have to subdivide the dark triangles recursively. The number of subdivisions in the subdivision of a dark triangle depends upon the positions of the vertices of the dark triangle, the blocking reflex vertices and the end-points of the partial edge. In this general case also, we can show that the number of subdivisions is of the order of the sum of the sizes of the unary encodings of the positions of these points. In our approach, the vertices of the dark triangle are the vertices of the visibility cell decomposition and the blocking reflex vertices and the end-points of the partial-edge are the vertices of the art gallery. The visibility cell decomposition is obtained

in polynomial time of the input. Therefore, the positions of all these points is polynomial in the size of the unary encoding of the art gallery. Hence, the number of subdivisions of any dark triangle is polynomial in the unary encoding of the input. As discussed above, the number of subdivisions of a dark triangle could be arbitrarily large compared to the size of the input. We identify these cases as 'pinholes' in the art gallery. Essentially in this case, a point on the edge of the dark triangle sees only a tiny portion of the partial edge through the 'hole' between the two blocking reflex vertices. Thus, in our earlier example where, we chose $L1{=}L_2{=}1$ and $\frac{1}{\epsilon} = 2^{10}$, pq forms a pinhole with respect to ab and the partial edge e.

The number of the subdivisions for some of the dark triangles in the art gallery could be small, but for some it could be arbitrarily large. We need a good bound to analyze the running time of our algorithm. For that purpose we define a parameter K. For a given art gallery, we can loosely define $K = (\frac{L}{\epsilon})^4$, where L is distance between two farthest vertices in the art gallery and ϵ is the distance between two closest vertices in the art gallery. The power 4 comes from the running time of the visibility cell decomposition. In most of the cases, the number of subdivisions may be very small compared to K, but for the worst case analysis of our algorithm K is sufficient. Note that K is polynomial in the unary encoding of the input. K can also be defined in a slightly different fashion. Consider a vertex whose position requires the maximum number of bits among all the vertices of the art gallery. Let $\log k$ represent that number. Then, the size of the binary encoding of the art gallery is $O(n \log k)$, whereas the size of the unary encoding is $O(nk)$. We can define $K = k^4$. Note that in the case when k is a polynomial function of n, K is still polynomial in the size of the input. But when k is arbitrarily large compared to n, K could be exponential in the size of the input.

Summary: In this chapter, we described Step 2 of our algorithm. The vertex-visibility property is not directly useful in the construction of the algorithm. We defined another special visibility property - vertex-pair-visibility property - which guarantees the vertex-visibility property and also helps in the construction of the algorithm. After proving relevant results for this property, we proved the correctness of the algorithm. Finally we identified *pinholes* in the art gallery that lead to the pseudo-polynomial behavior of the algorithm.

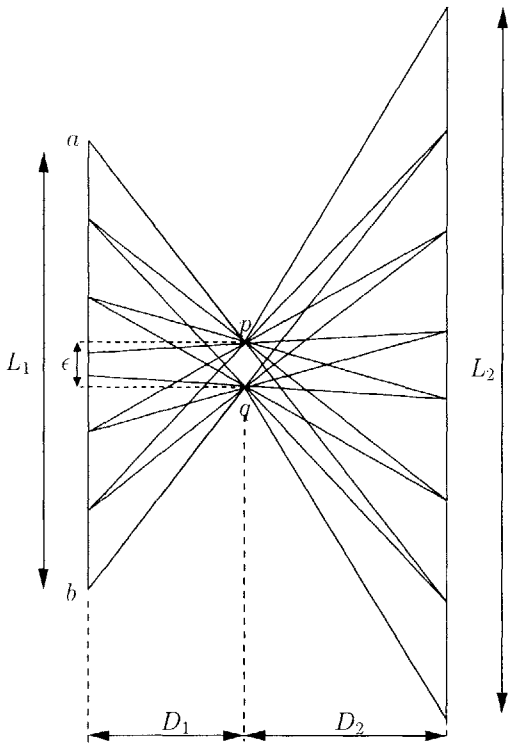

Figure 5.3: Pinhole pq

Chapter 6

Set Cover and VC-Dimension

In this chapter, we describe Step 3 of our algorithm. Each triangle in the final triangulation obtained in Step 2 of our algorithm satisfies the vertex-visibility property. We choose the vertices of this triangulation as the potential candidates for locating guards. The problem of finding minimum number of guards from these selected points can be formulated as a pure combinatorial problem — the set cover problem. If the set cover problem is solved optimally using exponential algorithm, the vertex-visibility property guarantees that the solution to the new problem is at most three times the optimal solution to the original point-guard problem. We use a *VC-dimension* based algorithm to solve the set cover problem approximately.

6.1 Set Cover Formulation and Approximate Solution

In this section, we describe Step 3 of our algorithm formally. We choose all the vertices of the final triangulation obtained in Step 2 as the potential guard-locations and formulate the set cover problem. The set cover problem is then solved approximately using a VC-dimension-based algorithm.

Step 3 of our algorithm can be summarized in the following way:

1. Construct a set G consisting of all the vertices of the final triangulation obtained in Step 2 of our algorithm. Let $|G| = m$.

2. Construct the visibility polygon for every $g_i \in G$ and generate the new subdivision of the polygon. Enumerate all the cells in the new subdivision and group them in the set $X = \{1, 2, \cdots, l\}$. For each $g_i \in G$, construct a set R_i of cells visible from g_i, that is, $R_i = \{x \in X | x \in V(g_i)\}$. Build the set family, $R = \{R_1, R_2, \cdots, R_m\}$. Group X and R together to form the set system (X, R).

3. Invoke the function *SET-COVER* on the set system Σ to obtain a near-optimal covering of X from the set family R. Return the set of guard-locations corresponding to the sets in the covering solution.

The function *SET-COVER* used in the above procedure is based on the algorithm proposed by Brönnimann and Goodrich [3] for finding set covers for set systems with finite VC-dimension. In the next section, we prove a bound on the VC-dimension of the set system generated in the above procedure. In this paper, we do not give details of the function *SET-COVER*. We refer interested readers to [9] for further details.

6.2 Bound on the Approximation Ratio of the Algorithm

The final triangulation in Step 2 of our algorithm satisfies the vertex-visibility property. We choose the vertices of the final triangulation as the potential candidates for locating guards. The optimal solution of the problem of choosing the minimum number of guards from the set of candidate guard-locations is at most three times the optimal solution of the original point-guard problem. We solve the new problem in Step 3 of our algorithm by solving the set cover problem. Instead of using the special function *SET-COVER* that is based on the VC-dimension of the problem, if we use Chvátal's greedy approach [5], we get the solution with the approximation ratio $O(\log n + \log K)$. As long as $K = O(n^{c_1})$ for some constant c_1, we get $O(\log n)$ approximation ratio, but when K is arbitrarily large compared with n, then the approximation ratio is $O(\log K)$ which linear in the size of the input and that is undesirable. Therefore, instead of using the greedy approach, we use *SET-COVER*.

SET-COVER uses the VC-dimension-based algorithm as proposed by Brönnimann and Goodrich [3]. The main result from [3] is that, for set systems with VC-dimension d, it is possible to compute in polynomial time a hitting set of size $O(dc\ log(dc))$, where c is the size of the smallest hitting set. First, we define the hitting set and VC-dimension. Then, we prove an important theorem about the VC-dimension of our problem. Most of the discussion in this section is inspired from [9].

Definition 6.2.1. *Let (X, R) be a set system such that the set family $R = \{R_1, R_2, ..., R_m\}$ covers $X = \{1, 2, .., l\}$. Let T_x, where $x \in X$, be a set consisting of all the sets in R that contain x. The* dual set system *(Y, S) of (X, R) is defined by $Y = R$ and $S = \{T_x | x \in X\}$. A* hitting set *is a set $H \subseteq Y$ such that $H \cap T \neq \phi$, $\forall\ T \in S$.*

Definition 6.2.2. VC-dimension *: Let (Y, S) denote a set system. We say a set $A \subseteq Y$ is* shattered *by S if for any subset $B \subseteq A$, there exists some $T \in S$ such that $B = A \cap T$. The* VC-dimension *of the set system (Y, S) is the cardinality of the largest shattered subset of Y.*

Let (X, R) be the set system of our problem as generated in Step 3 of our algorithm. Let (Y, S) be the corresponding dual set system. The set Y represents the set of potential choices for locating guards and a set T_x in S consists of guards each of which can cover a particular cell $x \in X$ in the subdivision. The problem of finding the optimal set cover for (X, R) is equivalent to that of finding the smallest hitting set for (Y, S). Now, we prove the following theorem related to the VC-dimension of the dual set system of our problem.

Theorem 6.2.1. *The VC-dimension of the dual set system (Y, S) of the set system (X, R) of our problem as generated in Step 3 of our algorithm is $O(\log n)$.*

Proof. Select $A \subseteq Y$ such that $|A| = d$. As we have already seen, the visibility polygon of a guard can introduce at most n new edges. Therefore, the visibility polygons of the guards in A induce a sub-arrangement of at most n^2d^2 components. This sub-arrangement contains some of the cells from X. Suppose that cells i and j from X are present in a common component in the new sub-arrangement. Let S_i and S_j be their corresponding sets in S.

Since, i and j are in the same component of the sub-arrangement, $S_i \cap A = S_j \cap A$. Thus, all cells in a single component of the new sub-arrangement induce only one subset of A. Therefore, to induce all possible subsets of A, there must be at least 2^d components in the sub-arrangement; that is $2^d < n^2 d^2$. Therefore, $\frac{2^d}{d^2} < n^2$. For $d \geq 4$, $d < 2 \log n$. Therefore, the VC-dimension of our problem is $O(\log n)$. □

Above Theorem implies that the function *SET-COVER* produces a solution that has the approximation ratio $O(\log n \cdot \log(c \log n))$, where $c = O(n)$ in the worst case. Therefore, overall we get a solution with the approximation ratio $O(\log^2 n)$, which depends only on the number of points in the art gallery and does not depend on the parameter K.

Summary: In this chapter, we formally explained Step 3 of our algorithm. The pseudo-polynomial behavior of Step 2 of our algorithm leads to exponential size of the set cover problem. The usual greedy approach does not give a good approximation ratio. Hence we used a VC-dimension based algorithm to solve the set cover problem. We proved a bound on the size of the VC-dimension of our problem. This leads to the overall approximation ratio of $O(\log^2 n)$ with pseudo-polynomial running time.

Chapter 7

Extensions And Conclusions

In the previous four chapters, we described our algorithm in detail for the case of art gallery without holes including its correctness and running time and approximation ratio analysis. The algorithm is also applicable to the case of art gallery with holes with slight modifications. In this chapter, we describe details for this case. We then conclude the thesis by outlining topics for future work.

7.1 Art Gallery with Holes

Our algorithm can still be used for the case of art gallery with holes. The algorithm involves same steps. Guibas et. al. [10] extend the visibility cell decomposition to a polygon with holes; except that in this case, the vertices of the holes also act as the blocking vertices and their visibility polygons take part in construction of the visibility cell decomposition. Step 1 of the algorithm for this case involves construction of the visibility cell decomposition and initial triangulation by triangulating each visibility cell. The subdivision procedure in Step 2 of our algorithm still remains valid. The overall running time of the algorithm is still a polynomial function of n and K as in the previous case. However, note that in this case n is the total number of vertices including vertices of the holes as well. The result about the bound on the VC-dimension of the problem still holds true and we get a solution with

guaranteed $O(\log^2 n)$-approximation ratio.

7.2 Conclusions and Extensions

In this paper, we have considered a variant of the art gallery problem — the point guard problem and have presented a pseudo-polynomial time algorithm with guaranteed $O(\log^2 n)$ approximation ratio. Our basic approach involves reducing the art gallery problem to the problem of choosing the minimum number of guard-locations from a finite set obtained by a special subdivision procedure. The optimal solution of the new problem is at most three times the optimal solution to the art gallery problem. We further reduce the new problem to the set cover problem and obtain an approximate solution to the set cover problem.

An interesting topic for future research is to investigate whether our subdivision procedure can be applied to other variants of the art gallery problems such as the limited-range version. There are a number of questions that remain open on the theoretical side. Can we get better bounds on the approximation ratio of our algorithm? Can we improve the running time of our algorithm? Is it possible to reduce the point-guard problem to a purely combinatorial problem?

Bibliography

[1] P. Bose, A. Lubiw, and J. I. Munro. Efficient visibility queries in simple polygons. In *Proc. 4th Canad. Conf. Comput. Geom.*, pages 23–28, 1992.

[2] Björn Brodén, Mikael Hammar, and Bengt J. Nilsson. Guarding lines and 2-link polygons is apx-hard. In *Proc. 13th Canad. Conf. Comput. Geom.*, pages 45–48, 2001.

[3] Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite vc-dimension: (preliminary version). In *Proceedings of the tenth annual symposium on Computational geometry*, pages 293–302. ACM Press, 1994.

[4] V. Chvátal. A combinatorial theorem in plane geometry. *Journal of combinatorial theory*, 18:39–41, 1975.

[5] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Cliff Stein. *Introduction to Algorithms.* MIT Press, Cambridge, MA, USA, second edition, 2001.

[6] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications.* Springer-Verlag, Heidelberg, 1997.

[7] S. Eidenbenz, C. Stamm, and P. Widmayer. Inapproximability results for guarding polygons and terrains. *Algorithmica*, 31:79–113, 2001.

[8] S. Ghosh. Approximation algorithm for art gallery problems. In *Canad. Information Processing Soc. Congress*, 1987.

[9] H. González-Banos and J. Latombe. A randomized art-gallery algorithm for sensor placement. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pages 232–240. ACM Press, 2001.

[10] L. J. Guibas, R. Motwani, and P. Raghavan. The robot localization problem. *SIAM J. Comput.*, 26(4):1120–38, 1997.

[11] D. T. Lee and A. K. Lin. Computational complexity of art gallery problems. *IEEE Transactions on Information Theory.*, IT-32:276–282, 1986.

[12] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society*, vol.3:1380–1387, 2001.

[13] Joseph O'Rourke. *Art Gallery Theorems and Algorithms.* Oxford University Press, 1987.

[14] T. Shermer. Recent results in art galleries. In *Proceedings of the IEEE*, volume 80, pages 1384–1399, 1992.

[15] Jorge Urrutia. *Art Gallery and Illumination Problems.* North-Holland, Amsterdam, 2000.