Optimal Placement of Light Fixtures for Energy Saving

by

Huamei Tian
B.Sc., Southeast University, 2010
M.Sc., Southeast University, 2013

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

in the Department of Computer Science
University of Victoria
Victoria, BC Canada

Optimal Placement of Light Fixtures for Energy Saving

by

Huamei Tian
B.Sc., Southeast University, 2010
M.Sc., Southeast University, 2013

Supervisory Committee

Prof. Kui Wu, Co-Supervisor
(Department of Computer Science)

Prof. Sue Whitesides, Co-Supervisor
(Department of Computer Science)

Supervisory Committee

Prof. Kui Wu, Co-Supervisor
(Department of Computer Science)

Prof. Sue Whitesides, Co-Supervisor
(Department of Computer Science)

ABSTRACT

Energy consumption of large commercial buildings has become higher than before, and a major part of the energy is on their lighting systems. This thesis aims at reducing the energy consumption of a building's lighting system. Our solution is to minimize the total number of necessary light fixtures in a commercial building, and thus we formulate the Constrained Light Deployment Problem (CLDP). The CLDP problem is tightly related to the Art Gallery Problem (AGP), a classical problem in computational geometry that finds the minimum number of guards to monitor a polygon area. Unlike the traditional AGP, however, our problem poses a new challenge that the illuminance of any spot in the building must be higher than a required threshold. To address the new challenge, we first propose an algorithm based on polygon partition and iteratively remove redundant light fixtures to obtain a tighter upper bound on the necessary number of light fixtures. We further improve the algorithm with clustering and binary search to reduce the number of light fixtures. Our algorithm can return the locations of resulted light fixtures, which are not necessarily the vertices of the orthogonal polygon. Simulation results demonstrate that our algorithm is fast and effective.

Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **iv**

List of Tables **vii**

List of Figures **viii**

List of Abbreviations **ix**

Acknowledgements **x**

1 Introduction **1**

1.1 Minimizing Lighting Energy Consumption without Sacrificing Visual Comfort 1

1.2 Contributions of the Thesis 3

1.3 Thesis Organization 3

2 Problem Formulation and Related Work **5**

2.1 Background 5

2.2 Formulation of Constrained Light Deployment Problem 6

2.2.1 Illumination Model 7

2.2.2 Unobstructed Distance 8

2.2.3 Objective Functions and Constraints 9

2.3 Related Work 10

2.3.1 Significant Results Related to AGP 10

2.3.2 Algorithms Based on Linear Programming 12

2.3.3 Algorithms Based on Set Covering Problem 14

2.3.4 Algorithms Based on Hitting Set Problem 14
2.3.5 Algorithms Based on Genetic Models 15
2.3.6 Algorithms Based on Simulated Annealing Approach 16
2.3.7 Other Algorithms 17
2.4 Problem Analysis and Solution 18
2.4.1 The Lower Bound 18
2.4.2 A Tighter Upper Bound 19
2.4.3 Obtain A Minimum N 20
2.4.4 A Bird View of the Solution 20

3 A Tighter Upper Bound for CLDP in the case of SOP 22
3.1 Overview 22
3.2 The SOP-UL Problem 23
3.2.1 Definitions and Propositions 24
3.2.2 A Heuristic Algorithm 26
3.2.3 Redundancy Removal Algorithm 27
3.3 Performance Evaluation 27
3.3.1 The Choices for Candidate Light Sets 27
3.3.2 Time Complexity Analysis 30
3.4 The SOP-L Problem 31

4 A Tighter Upper Bound for CLDP in the case of OPH 34
4.1 Overview 34
4.2 Algorithm Improvement for OPH 35
4.3 Experimental Evaluation 36
4.3.1 OPH Generation 36
4.3.2 Evaluation Results for the OPH-UL Problem 38
4.3.3 Evaluation Results for the OPH-L Problem 42

5 Solving CLDP with Binary Search and Clustering 46
5.1 Overview 46
5.2 Algorithm 46
5.3 Determine Whether k Lights Are Enough 47
5.3.1 Clustering with Unobstructed Distance 47
5.3.2 Hierarchical Clustering and Adjustment 47
5.4 Experimental Evaluation 49

5.4.1 Considerations on the Observation Points 50
5.4.2 Influence of the Effective Illumination Radius 56

6 Conclusions and Future Work **58**
6.1 Summary 58
6.2 Future Work 58

Bibliography **60**

List of Tables

Table 2.1 Table of notations 7

Table 3.1 Average number of light fixtures 31
Table 3.2 Min/Max number of light fixtures 31

Table 4.1 Number of vertices vs. number of light fixtures 41
Table 4.2 Number of holes vs. number of lights 43

List of Figures

Figure 1.1 Electric demand in commercial buildings 2

Figure 2.1 The main work of this theis 21

Figure 3.1 Block visibility 26
Figure 3.2 Results obtained from different candidate light sets 30
Figure 3.3 Average number of light fixtures for different candidate light sets 32
Figure 3.4 The number of light fixtures for SOPs vs. the effective illumination radius 33

Figure 4.1 Special case in OPH 36
Figure 4.2 Figure for the proof of proposition 5 37
Figure 4.3 Inflate-Cut example 1 38
Figure 4.4 Inflate-Cut example 2 40
Figure 4.5 Orthogonal polygon with one hole 41
Figure 4.6 Orthogonal polygon with two holes 42
Figure 4.7 The number of hole vertices vs. the number of lights 43
Figure 4.8 The number of holes vs. the number of lights 44
Figure 4.9 The number of light fixtures for OPHs vs. the effective illumination radius 45

Figure 5.1 Search for the necessary number of lights 48
Figure 5.2 The upper bound results 51
Figure 5.3 Clustering results, k=8 52
Figure 5.4 Clustering results, k=6 53
Figure 5.5 Clustering results, k=7 54
Figure 5.6 Special cases at observation points 55
Figure 5.7 Clustering results, radius = 4 57

List of Abbreviations

Compact Fluorescent Lamp CFL

Light Emitting Diode LED

CLDP Constrained Light Deployment Problem

AGP Art Gallery Problem

SOP Simple Orthogonal Polygon

OPH Orthogonal Polygon with Holes

LP Linear Programming

ACKNOWLEDGEMENTS

I would like to thank all those who gave me support and help during the writing of this thesis.

I gratefully acknowledge the help of my Co-Supervisors, Dr. Kui Wu and Dr. Sue Whitesides, who offered me a valuable opportunity to study in University of Victoria. They gave me lots of help and guidance throughout the graduate study. When I encounter difficult problems, they always consider them with importance and give me inspiring suggestions. I can hardly finish this thesis without their patient instruction, expert guidance and insightful criticism.

I would like to thank my family for their company through my high and low points. I thank my friends for their support. I especially want to thank River and Nishat for the discussions on graph theories and art gallery problems. Thanks to my friend Lan for encouraging me to overcome the difficulties.

Chapter 1

Introduction

1.1 Minimizing Lighting Energy Consumption without Sacrificing Visual Comfort

Lighting systems, critical to our daily life in modern society, consume a tremendous amount of energy and financial resources. A survey [16] published by Schneider Electric discloses that the annual energy consumption in lighting systems of commercial buildings is over 336 billion Kwh. Figure 1.1 [16] shows the electric demand in commercial buildings, from which we can see that lighting consumption amounts to 44% of the electricity bill. Therefore, reducing energy consumption of lighting systems has a significant impact on the sustainable development of our society.

The energy comsumption of lighting system is influenced by a number of factors: the power of the light fixture, the lighting time, and the number of light fixtures. Thus there are many ways to reduce lighting cost. The most common method is to use energy-saving light bulbs [46], such as Compact Fluorescent Lamps (CFLs) and Light Emitting Diodes (LEDs). Secondly, we can take advantage of the daylight in commercial buildings. Thirdly, we can use the minimum number of light fixtures and turn off the redundant ones.

In this thesis, we work on the third idea mentioned above. Suppose the light fixtures in a commercial building are of the same specification, which means the light fixtures provide the same illuminance at the sources. In this case, less number of light fixtures lower the illuminance level (or brightness). However, lighting systems should make the tenants comfortable and thus require a desirable illuminance level in the rooms. The two requirements are contradicting. The challenging problem

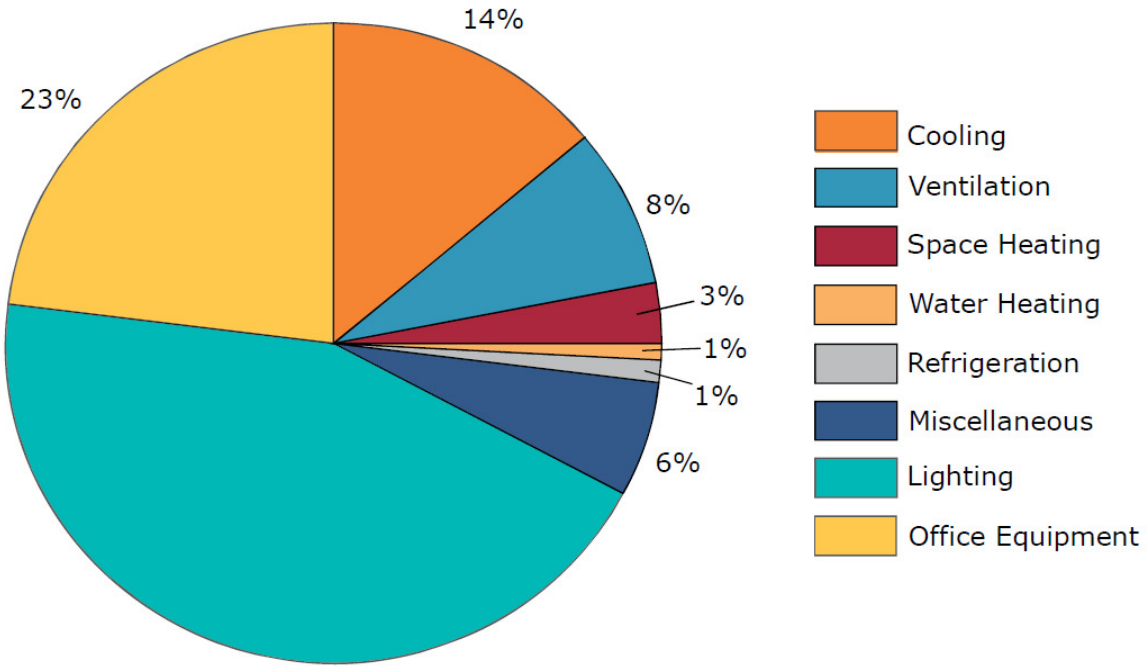

Figure 1.1: Electric demand in commercial buildings

we are faced with is: without changing the building infrastructure such as installing skylights, can we reduce the number of light fixtures and in the meantime maintain the comfort of tenants?

To tackle the above problem, a solution is to use the minimum number of light fixtures in a building such that a given level of illuminance can be satisfied in the building. In the problem, we suppose the illuminance requirement in the whole building is the same. We call this problem Constrained Light Deployment Problem (CLDP). This problem is similar to the well-known "Art Gallery Problem (AGP)", in which we need to find the minimum number of guards to fully monitor an art gallery. While a light fixture is analogous to a guard, our problem is much harder than AGP due to two reasons: (1) the illuminance of a light fixture decreases over distance and (2) the aggregated illuminance at any spot[1] of the building needs to reach a given threshold.

This thesis aims at designing effective algorithms to tackle the challenging CLDP problem.

[1]This requirement is to simplify the problem. It does not necessarily mean in practice every corner of the building needs to reach the same illuminance level.

1.2 Contributions of the Thesis

This thesis studies a new problem of using coverage-limited light fixtures to light up buildings whose floor plans are shaped as orthogonal polygons. The main contributions of the thesis are as follows:

1. In this thesis, we formulate the CLDP problem, which is a new variant of the classical art gallery problem. Different from the classical art gallery problem, which only considers the visibility of the target area when choosing the locations for the guards, the new problem tries to find the minimum number of light fixtures to light up the target area. The light fixtures are assumed to be point light sources. The illuminance requirement is also given as part of the input.

2. To solve the challenging CLDP problem, we calculate the upper bound and the lower bound for the number of the needed light fixtures and then we use binary search to find the exact number of light fixtures. When determining if a certain number satisfies the constraints, we use a clustering algorithm to divide the observation points into clusters according to the unobstructed distance between the points and the light fixtures. In addition, we reduce the illuminance difference between the observation points.

3. We propose a simple algorithm to obtain a tighter upper bound on the number of the light fixtures for simple orthogonal polygons and an enhanced algorithm for orthogonal polygons with holes. Comprehensive experiments show that the algorithms can effectively solve the problem.

4. Different from most existing algorithms for the art gallery problem, we not only consider the vertices of the target area as the candidate locations for the light fixtures, but also select points inside the target area as possible candidates, such as the intersections of edge extensions.

5. In order to test our algorithm, especially for the case that the target area is an orthogonal polygon with holes, we proposed an algorithm to generate orthogonal polygons with arbitrary number of holes.

1.3 Thesis Organization

This thesis is organized as follows:

- In Chapter 2, the CLDP problem is formulated and analysed. A solution based on binary search is introduced. The chapter also gives a simple method to obtain the lower bound for the number of the light fixtures, and formulates a problem for obtaining the upper bound. We also review the previous research related to art gallery problem.

- In Chapter 3, we get the upper bound for the number of necessary light fixtures with limited coverage for simple orthogonal polygons. We set the illuminance requirement to 0 at first, meaning that the light fixtures have unlimited coverage. We formulate the problem of determining the minimum number of light fixtures in a simple orthogonal polygon area without illuminance limits as an optimization problem. An algorithm based on partitioning and set covering is proposed to obtain an upper bound for the minimum number. Based on the algorithm, we impose the illuminance requirement and achieve the upper bound on the necessary number of light fixtures for simple orthogonal polygons.

- In Chapter 4, we ignore the illuminance constraints first when the target area is an orthogonal polygon with a number of holes. Several cases are deeply researched: (a) orthogonal polygons with various numbers of vertices, (b) orthogonal polygons with various numbers of holes but with the same number of hole vertices, (c) orthogonal polygons with various numbers of hole vertices. At the end, we obtain an upper bound for the necessary number of light fixtures for orthogonal polygons with holes when illuminance constraints are considered.

- In Chapter 5, we propose an algorithm to calculate the necessary light fixtures using binary search and a clustering algorithm based on the lower bound and the upper bound. Instead of only placing the light fixtures at vertices or the intersections of the edge extensions, we obtain better locations for the light fixtures to minimize the illuminance variance at different observation points.

- In Chapter 6, we summarize the thesis and point out future work.

Chapter 2

Problem Formulation and Related Work

In this chapter, we formulate the problem of minimizing the number of necessary light fixtures in a commercial building. Our goal is to save lighting energy consumption without sacrificing visual comfort, which means the illuminance of the target area should satisfy the given requirements.

2.1 Background

Our problem is a variation of the Art Gallery Problem (AGP), which is a classical problem in Computational Geometry proposed by Victor Klee in 1973. The classical AGP is to find the minimum number of guards for an arbitrary art gallery with n walls [37]. There are many applications for the Art Gallery Problem. For example, the home owners may want to illuminate their houses with the fewest number of lighting fixtures to save electric costs; museum or gallery managers may want to use the minimum number of cameras to monitor an entire building; mobile telecom carriers may seek to use the minimum number of mobile communication base stations to reduce operation costs while still guaranteeing good quality of service in the service area.

Many variations of AGP were proposed and studied by adding variant restrictions. The related research can be classified by the target monitoring areas, the locations of the guards, and the shapes of the polygons. For classification by target monitoring areas, the objective is mainly to cover the interior of the polygon [37, 45]. Some

applications only have interest in observing the boundary, such as inspection or image based rendering, which inspire the researches on edge covering problem [33, 39]. There is also research on vertex covering problem, which only requires every vertex of the polygon is monitored [27]. For the locations of the guards, the research mainly focuses on three cases: (1) vertex guards: the guards can only be placed at the vertices; (2) edge guards: the guards can be placed on the boundary; (3) point guards: the guards can be anywhere in the interior of the polygon. Based on the shape, the target region is usually modeled as a 2D polygon according to the region's shape, which could be: (1) a simple polygon without holes; (2) a simple polygon with holes; (3) an orthogonal polygon without holes, also known as Simple Orthogonal Polygon (SOP); (4) an Orthogonal Polygon with Holes (OPH).

However, the classical AGP or its variations is hard to solve. Lee and Lin [34] discussed the computational complexity of the minimum vertex guard problem, and they proved that determining the minimum number of vertex guards or point guards in a n-edge simple polygon is NP-hard. A few years later, Schuchardt and Hecker [38] worked on orthogonal polygons and also showed that minimizing the number of guards on the vertices or in the interior of an orthogonal region is NP-hard. Katz et al. focused their work on guarding the vertices of the orthogonal polygons, and they proved that guarding vertices of an orthogonal polygon is NP-hard [27], no matter where the guards are placed (i.e., at the vertices, on the boundary, or anywhere in the interior of the orthogonal polygon).

2.2 Formulation of Constrained Light Deployment Problem

In this thesis, we consider commercial buildings whose floor plans are shaped as orthogonal polygons. The floor plan of a building can be represented as an orthogonal polygon P with vertex set $V = v_1, v_2, ..., v_n$, where $|V| = n$. The goal of the classical AGP is to find a minimum guard set G for the polygon P so that all points inside the polygon P are visible to at least one guard in G. In order to save lighting energy consumption, we want to place as few lights in the polygon as possible. This is another description of the classic AGP if the illuminance at the target region is not taken into account. But in this thesis, our goal is to satisfy the illuminance requirements at every point in the polygon with the minimum number of light fixtures. We have to

consider the illuminance attenuation model. This problem is called Constrained Light Deployment Problem (CLDP) and is significantly different from AGP.

The main notations used in this thesis are listed in Table 2.1.

Table 2.1: Table of notations

Symbol	Meaning
P	The orthogonal polygon
V	The vertex set of P: $V = v_1, v_2, ..., v_n$, where $\|V\| = n$
G	The light fixtures set: $G = g_1, g_2, ..., g_m$, where $\|G\| = m$
g	A light fixture and $g \in G$
R	The radius of the effective illumination area of the light fixture
R_g	The illumination area of the light fixture g, which is the set of all points that are visible to g in P
I_p	The total illuminance at the point p. The illuminance at a single light source is denoted as I_0
I_{pg}	The illuminance received from light fixture g at the point p
d_{pg}	The unobstructed distance between point p and light fixture g
δ	The minimum illuminance requirement/threshold
N	The number of light fixtures that satisfy the requirements
N_{lb}	The lower bound on N
N_{ub}	The upper bound on N

2.2.1 Illumination Model

Illuminance [47] is a measure of how much the incident light illuminates the surface. It is the intensity of luminous flux per unit area, and it can be expressed in lux (lumens/square meter, where lumens is the unit of luminous flux).

According to the inverse-square law [48], the intensity (or illuminance) of light radiating from a point source is inversely proportional to the square of the distance from the source. Suppose that the illuminance at a single light source is I_0. The illuminance at point p received from light g, denoted by I_{pg}, usually can be calculated by the following function:

$$I_{pg} = \frac{I_0}{a + b \cdot d_{pg} + c \cdot d_{pg}^2},$$

where d_{pg} represents the line-of-sight distance between the point p and the light g, and a, b, and c are coefficients that can be set according to empiracle measurements.

To simplify the model, we assume $a = c = 1$ and $b = 0$. Hence, we have

$$I_{pg} = \frac{I_0}{1 + d_{pg}^2}.$$

Disregarding the refraction, reflection and coherence of light waves, all of which depend on the setup of interior furniture and are hard to model, we assume that the total illuminance at the point p is the sum of the illumination intensities received from all lights. Given the light set $G = g_1, g_2, ..., g_m$ in the polygon P. The total illuminance at the point p can be calculated by Equation (2.1):

$$I_p = \sum_{i=1}^{N} \frac{I_0}{1 + d_{ip}^2} \tag{2.1}$$

where N is the number of lights that can illuminate the point p directly and d_{ip} represents the distance between the $i-$th light and point p.

The effective illumination area of a light fixture is the area where the illuminance received from that light fixture is greater than the illumination requirement. Suppose the threshold of the illumination requirement is δ, the effective illumination radius R of the effective illumination area of the light fixture must satisfy the following equation:

$$I_p = \frac{I_0}{1 + R^2} = \delta.$$

Thus we can calculate the effective illumination radius R of the light fixture by:

$$R = \sqrt{\frac{I_0}{\delta} - 1} \tag{2.2}$$

2.2.2 Unobstructed Distance

In the above illumination model, we can calculate the illuminance at one observation point according to the distance between the point and the light fixture. However, the light could be blocked by a wall. In our context, if the line connecting the observation point p and the light fixture g is blocked by some walls, then I_{pg} should be 0, which means d_{pg} should be infinity. Obviously, the Euclidean distance between the point p and the light g cannot satisfy this constraint. Therefore, we define the notion of *Unobstructed Distance* between point $p = \{p_x, p_y\}$ and $g = \{g_x, g_y\}$ as:

$$d_{pg} = \sqrt{(p_x - g_x)^2 + (p_y - g_y)^2}\mathcal{I}, \tag{2.3}$$

where $\mathcal{I}$ is the indicator function:

$$\mathcal{I} = \begin{cases} 1, & \text{if } p \text{ and } g \text{ have a line-of-sight} \\ \infty, & otherwise \end{cases} \tag{2.4}$$

Equation (2.3) shows that if the observation point p is visible to one light g, the unobstructed distance between the point and the light is the Euclidean distance, otherwise, the unobstructed distance is bound onto infinity.

In the rest part of this thesis, the distance d_{pg} means the unobstructed distance between p and g.

2.2.3 Objective Functions and Constraints

Our first and main objective is to minimize the number of light fixtures for a given orthogonal polygon.

Objective:

$$\text{minimize } N, \text{where N is the number of light fixtures} \tag{2.5}$$

To guarantee visual comfort, we set the required illuminance to a threshold δ and require that the illuminance at every observation point to be above the threshold δ. In other words,

Constraint:

$$I_p = \sum_{i=1}^{N} \frac{I_0}{1 + d_{pi}^2} \geq \delta, \text{for every point } p \text{ in the polygon} \tag{2.6}$$

where δ is the illuminance threshold, and d_{pi} represents the lighting distance between the point p and the $i-$th light.

The CLDP problem can now be formulated as follows:

$$
\begin{array}{ll}
\text{input} & \text{The illuminance at the source: } I_0. \\
& \text{The illuminance requirement threshold: } \delta. \\
& \text{The vertex set of polygon } P\text{: } V = v_1, v_2, ..., v_n. \\
& \text{The observation point } j \text{ in polygon } P, \text{ for } j = 1, \ldots, m. \\
\text{output} & \text{The locations of the } N \text{ light fixtures.} \\
\text{minimize} & N \\
\text{subject to} & I_j = \sum_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2},\ j = 1, \ldots, m. \\
& I_j \geq \delta,\ j = 1, \ldots, m.
\end{array}
\tag{2.7}
$$

2.3 Related Work

2.3.1 Significant Results Related to AGP

O'Rouke [37] proved that a simple polygon that has $r \geq 1$ reflex vertices always can be covered by r guards at reflex vertices.

Chvátal [10] showed that $\lfloor \frac{n}{3} \rfloor$ point guards are always sufficient and sometimes necessary to guard a simple polygon. This theorem was later proved by a $3-$*coloring* method [17].

For a simple polygon with c convex vertices, Addario-Berry et al. [1] claimed that $2c - 4$ point guards are sufficient and sometimes necessary to cover the polygon.

Iwerks and Mitchell further researched the bound on the number of guards for simple polygons with r reflex vertices and c convex vertices [24]. They proved that when $\lfloor \frac{c}{2} \rfloor < r < 5c - 12$, $\lfloor \frac{n}{3} \rfloor$ point guards are always sufficient and sometimes necessary to cover a simple polygon. They defined function $G(r, c)$ to represent the maximum necessary number of guards over all simple polygons with r reflex vertices and c convex vertices. Combining the results of O'Rourke and Addaric et al., they

showed $G(r, c)$ satisfies the following equation:

$$G(r,c) = \begin{cases} 1, & \text{if } r = 0; \\ r, & \text{if } r \leq \lfloor \frac{c}{2} \rfloor; \\ \lfloor \frac{n}{3} \rfloor, & \text{if } \lfloor \frac{c}{2} \rfloor < r < 5c - 12; \\ 2c - 4, & \text{if } r \geq 5c - 12. \end{cases} \tag{2.8}$$

For a polygon with h holes and a total of n vertices, O'Rourke [37] proved that at most $\lfloor \frac{n+2h}{3} \rfloor$ vertex guards are needed to guard the polygon.

If the guards can be placed at any point, $\lceil \frac{n+h}{3} \rceil$ point guards are always sufficient for guarding the polygon with h holes and n vertices, which has been proved by Hoffmann et al. [21] and Bjorling-Sachs et al. [8].

Kahn et al. [26] proved that $\lfloor \frac{n}{4} \rfloor$ guards are always sufficient and sometimes necessary to cover a simple orthogonal polygon of n vertices.

Edelsbrunner et al. [15] developed an $O(n \log(\log n))$ time algorithm using $\lfloor \frac{n}{4} \rfloor$ guards to cover arbitrary orthogonal polygons with n vertices. However, $\lfloor \frac{n}{4} \rfloor$ is the upper bound on the required number of guards and is often much greater than the necessary number. Edelsbrunner's algorithm did not deal with the guarding set minimization problem, which is more challenging.

O'Rourke [37] showed that $\lfloor \frac{n+2h}{4} \rfloor$ vertex guards are always sufficient to guard an $n-$vertex orthogonal polygon with an arbitrary number h of holes.

Hoffmann and Kriegel [22] proved that at most $\frac{n+h}{3}$ vertex guards are needed for an orthogonal polygon with n vertices and h holes. Żyliński [50] proved that the bounds can be improved to $\lfloor \frac{n+h}{4} \rfloor$. Żyliński 4-colored the quadrilateralization graph that is obtained by partitioning the orthogonal polygon into a number of convex quadrilaterals with pairwise disjoint interiors, and then positioned the guards at those vertices with the colour that has the minimum number of occurrences. Michael and Pinciu [35] improved the bound to $\frac{17n-8}{52}$.

Hliněnỳ [20] researched polygons with n vertices and m interior diagonals. He proved that $min\{ \lfloor \frac{n+2m}{3} \rfloor, \lfloor \frac{2n-3}{3} \rfloor, \lfloor \frac{2n+m-2}{4} \rfloor\}$ guards are sufficient to guard these kinds of polygons.

Hutchinson and Kundgen [23] focused on orthogonal polygons with horizontal or vertical interior walls and gave a result for the minimum number of guards: $min\ \{\lfloor \frac{n+2m}{4} \rfloor, \lfloor \frac{3 \lfloor n/2 \rfloor n+m-2}{8} \rfloor\}$.

Bajuelos et al. [4] studied the problem of guarding orthogonal galleries with n

rectangular rooms. For any two adjacent rooms, there is one door connecting them, and the two rooms are visible to the guard placed at this door. They proved that L-shape polygons can be guarded with $\lceil \frac{n}{2} \rceil$ guards, and an orthogonal staircase with r reflex vertices and n rooms can be guarded with $\lceil \frac{n+\lfloor \frac{r}{2} \rfloor}{2} \rceil$ guards. They also gave an upper bound on the number of guards for arbitrary orthogonal polygons with h holes, r reflexes and n rooms.

2.3.2 Algorithms Based on Linear Programming

Couto et al. [13][12][11] proposed an approximation algorithm based on polygon discretization and Linear Programming (LP) for orthogonal polygons with vertex guards. They discretized the polygon P by generating a finite set $D(P)$ of representative points in the polygon, and formulated the problem as the following LP problem:

$$\begin{aligned} \min \quad & \sum_{j \in V} x_j \\ \text{s.t.} \quad & \sum_{j \in V} a_{ij} x_j \ \geq \ 1, \ \forall \ p_i \in D(P) \\ & x_j \in \{0,1\}, \ \forall \ j \in P \end{aligned} \tag{2.9}$$

In the above formulation, x_j is a binary variable and a_{ij} is a binary value, where i corresponds to the $i-$th point in $D(P)$ and j corresponds to the $j-$th vertex. If point p_i is visible to the vertex j, then a_{ij} is set to be 1. If vertex j is a guard, then the variable x_j is set to be 1. The objective function minimizes the number of guards, and the first constraint guarantees that every point in $D(P)$ is guarded by at least one guard. Since $D(P)$ is a discretized instance of polygon P, the solution of the above IP problem may not be a guard set for the polygon P, implying that there might be some uncovered regions in P. To overcome this problem, Couto et al. choose a new point inside every uncovered region and add these points to $D(P)$, and then a new IP problem is created. By iteratively adding uncovered points to $D(P)$ and solving the IP problem, their algorithm can obtain a complete guard set for the polygon P. In their algorithm, the initialization of the discretized points has a big influence on the processing time, and the number of iterations depends on the number of uncovered regions.

Tozoni et al. [44] developed an integer linear programming based algorithm to iteratively generate upper and lower bounds for the AGP through the solution of

discretized versions of the AGP. In the paper, the visibility polygon $Vis(g)$ of point g is a coverage set consisting of all points that are visible to g inside P. The objective of AGP is to determine a smallest set of points $G \subset P$ such that $\bigcup_{g \in G} Vis(g) = P$. Thus, any point in P is an element of at least one coverage set. Accordingly, Tozoni et al. formulated the AGP as follows:

$$\begin{aligned} \min \quad & \sum_{c \in P} x_c \\ \text{s.t.} \quad & \sum_{c \in P, w \in Vis(c)} x_c \ \geq \ 1, \ \forall \ w \in P \\ & x_c \in \{0, 1\}, \ \forall \ c \in P \end{aligned} \tag{2.10}$$

In the above formulation, the guard candidate c or the witness w is an arbitrary point in the polygon P. Thus, the problem formulation has an infinite number of constraints and an infinite number of variables x_c. The authors formulated the following discretized versions of the AGP: (1) Art Gallery Problem with Fixed Guard Candidates (AGPFC), by fixing the guard candidates to be finite; (2) Art Gallery Problem with Witness (AGPW), by fixing the witness set to be finite. With a fixed number of witnesses, the solution of the AGPW is a lower bound for the original AGP. With a fixed number of guard candidates, the solution of the AGPFC always yields an upper bound for the original AGP. Therefore, their algorithm initializes the upper bound as the vertex set and the lower bound as empty, and then iterates the following two steps: fix the witness set, solve the AGPW problem and use the solution to update the lower bound; fix the guard candidate set, solve the AGPFC problem and use the solution to update the upper bound. The gap between the upper bound and the lower bound decreases monotonically through the iterations and the algorithm stops when the gap reaches zero or the running time exceeds a given duration.

Baumgartner et al. [7] proposed a primal-dual algorithm based on linear programming for guarding a polygon with or without holes by point guards. Their algorithm is similar to Tononi's. The linear programming based algorithm calculates lower bounds on the necessary number of guards in each iteration step until the optimal solution is obtained.

2.3.3 Algorithms Based on Set Covering Problem

Set Covering Problem [49] is a NP-complete problem. Given a set of elements U and a collection S of subsets of U. The Set Covering Problem is to obtain a minimum subset C of S so that $\bigcup_{e \in C} e = U$.

Ghosh proposed an approximation algorithm for vertex guarding polygons with n vertices with or without holes [19]. Firstly, the algorithm partitions the polygon into m convex components by drawing lines between each pair of vertices of the polygon. It initializes the guard set with an empty set and determine which convex components are totally visible to each vertex and construct the visibility set for the vertex. If the visibility set of vertex A includes the visibility set of another vertex, vertex A is added to the guard set. It then deletes the components covered by the guard set from the component set. The algorithm iterates the last step until the component set is empty and finally output the guard set. The solution size is at most $O(OPT \cdot log\ n)$ for both simple polygons and polygons with holes, where OPT is the number of vertices in the optimal solution. The time complexity is $O(n^4)$ and $O(n^5)$ for simple polygons and polygons with holes, respectively.

Jang and Kwon [25] improved the $O(n^4)$ algorithm of Ghosh [19] for the minimum vertex guard problems in simple polygons. They claimed that there are $O(n^3)$ visibility regions for an arbitrary simple polygon with n vertices, and the approximation ratio is $log(n)$ limited by an approximation algorithm for the set covering problem. In their paper, a sink is defined as a visibility region whose visible vertex set is contained in the visible vertex set of every neighboring region. They proved that if the $O(n^2)$ sinks are covered, then all points in the polygon are visible to the guards. Based on this lemma, they proposed an $O(n^3)$ time algorithm. In their algorithm, they compute the visibility polygons for every vertex and construct the planar subdivision, and then compute all the visibility regions and construct a dual graph of the visibility regions. Later, they compute the set of sinks and the set of visible vertices for each sink. Finally, they compute the set of visible sinks for each vertex and solve the set covering problem. The solution is the set of guards, which is at most $O(log\ n)$ times the optimal solution.

2.3.4 Algorithms Based on Hitting Set Problem

Given a set of elements U and a collection S of subsets $S_1, S_2, ..., S_m$ of U. The Hitting Set Problem [49] to obtain a minimum subset$H$ of U so that $S \bigcap H \neq \varnothing$ for

any $S_i \in S$, $i = 1, 2, ..., m$.

King and Kirkpatrick [31] modeled the problem of guarding a simple polygon with guards on the vertices or on the perimeter as an instance of the Hitting Set problem. In their problem formulation, U represents the finite set of potential guards, and P is the set of points that must be guarded and is possibly infinite, and S_p represents the set of potential guards that can see the point $p \in P$ and $S = \{S_p | p \in P\}$. When P is infinite, it must be discretized into a finite representative subset $P' \subset P$, such that any subset of U that guards P' also guards P. Their algorithm defines a weight function ω for the guard set U and initializes the weight of each element in U with 1. To relax the Hitting Set problem, the algorithm finds an $\varepsilon -$ net H repeatedly. An $\varepsilon -$ net is a subset of U that hits every set in S having weight at least $\varepsilon \cdot \omega(U)$. When H is not a hitting set, for example, when H cannot see every point in P, choose a point $p \in P$ that is not seen by H and then find the guards that can see the point p and double these guards' weights. The authors build an $\varepsilon -$ net using $O(1/\varepsilon^2)$ guards and then use hierarchical fragmentation technique [30] to reduce the number of guards to $O(\frac{1}{\varepsilon} \log\log \frac{1}{\varepsilon})$. In their algorithm, the total weight of U increases in each iteration. They claim that their algorithm can give an approximation result of size $O(OPT \cdot \log\log OPT)$, which they justify by proving that the total weight never exceeds $\frac{|U|^4}{OPT^3}$, where where OPT is the size of the minimum subset of U that guards P'.

King analyzed the time complexity of the above algorithm in [30]. He proved that the algorithm runs in time $O(n^2 \log\log \frac{1}{\varepsilon})$ for the case of vertex guards. Additionally, King developed the approximation algorithm for vertex guarding polygons with h holes in [30]. The first step is to decompose the polygon into a number of cells. King proved that for a polygon with h holes, it can be decomposed into $O((h+1)n^3)$ cells and $O((h+1)^2 n^2)$ cells of minimal visibility. Later, the problem was modeled as a Hitting Set problem based on $\varepsilon -$ nets. The algorithm gives an approximation result of size $O((1 + \log(h+1)) \log OPT)$ for polygons with h holes, where OPT is the size of the minimum subset of U that guards P'.

2.3.5 Algorithms Based on Genetic Models

Genetic Algorithms can simulate the processes of the natural evolution to solve an optimization problem. Usually, "individuals" or "chromosomes" are genetic representation of the potential solutions; "initial population" ia a possible solution created at

the initialization; "selection", "crossover", and "mutation" are genetic operators to alter the solutions; and the "objective" or "fitness" function is a function to influence natural selection and evaluate the solutions.

Bajuelos Domínguez et al. [5] proposed an algorithm based on the general metaheuristic genetic approach to obtain a small vertex guard set for a simple orthogonal polygon. In their algorithm, they use a chain $I = g_0 g_1 ... g_{n-1}$ to represent an individual (or chromosome). Element g_i, called a gene, represents the vertex v_i of the polygon, and store a binary value. The vertex v_i is a vertex guard, if $g_i = 1$; otherwise, $g_i = 0$. The initial population is created based on the theory that r guards on the reflex vertices are always sufficient and sometimes necessary to guard a polygon that has a set $R = \{u_0, u_1, ..., u_{r-1}\}$ of reflex vertices. For arbitrary $i \in \{0, ..., r-1\}$, if $R \backslash \{u_i\}$ is a vertex-guard set, then the $i-$th individual is generated by setting all the vertices of $R \backslash \{u_i\}$ as guards; otherwise, the individual is generated by setting all vertices in R as guards. The authors defined the fitness function $f(I)$ as the number of $1's$ in the chain I. In "selection" step, the algorithm selects two best individuals of each method to be parents. In "crossover" step, the algorithm generates one child from the two parents with a given probability p_c. Then in the mutation step, each binary digit flips from 0 to 1 or vice versa, with a probability p_m. In each iteration, a new population is generated by replacing the worst individual of the population with the child obtained at the crossover. Since the objective is to find the best individuals that have the minimum $f(I)$, the algorithm terminates when the value of $f(I)$ remains unchanged for a number of generations (e.g., 500).

2.3.6 Algorithms Based on Simulated Annealing Approach

Bajuelos Domínguez et al. [6] described a metaheuristic method based on the simulated annealing approach for the vertex-guarding problem for simple polygons. For this method, the solution space is defined as the set of all vertex-guard sets of the simple polygon. Each candidate solution is a chain of length n and is denoted as $S_i = v_0^i v_1^i ... v_{n-1}^i$, where $i \in \{1, ..., size(\text{solution space})\}$. v_j represents a vertex of the polygon, and $v_j^i = 1$ for $j \in \{1, ..., n-1\}$, if only if v_j is a vertex-guard; otherwise $v_j^i = 0$. The objective function is defined as $f(S_i) = \sum_{k=0}^{n-1} v_j^i$. The initial solution is initialized with the set of all reflex vertices of the polygon. In each iteration, a neighbourhood of the solution S_i is generated by changing the value of one bit of S_i. The algorithm accepts the change with a probability, depending on a control

parameter called temperature. The temperature in the $k-$th iteration can be defined as one of the following three formulas:

- $T_{k+1} = \frac{T_0}{k+1}$.
- $T_{k+1} = \frac{T_0}{e^k}$.
- $T_{k+1} = \alpha T_0$, where $0 < \alpha < 1$.

For each temperature, the number of iterations is $N(T_k) = \lceil T_k \rceil$. The algorithm terminates when the temperature is less than one specified constant value.

2.3.7 Other Algorithms

Amit et al. [2] proposed a heuristic algorithm for guarding a polygon with point guards. Their algorithm generates a guard set by adding guards one by one until the whole polygon is covered by the guard set. In each step, only the guard that covers the largest previously uncovered area by the existing guard set can be added. They also defined a number of candidate sets to select necessary guards. The candidate sets could be the vertex set, some significant points inside the polygon, and so on. After the guard set is generated from its candidate set, the redundant guards are removed iteratively. The algorithm compares the final guard sets generated from different candidate sets and finally outputs the smallest guard set.

Deshpande et al. [14] developed a sophisticated discretization method that gives an $O(logOPT)$ approximation result for guarding art galleries with point guards, where OPT is the optimal number of guards. The algorithm runs in pseudo polynomial time and could be linear in the ratio between the longest and shortest distances between two vertices.

Bottino and Laurentini studied an incremental algorithm for the edge covering problem and then proposed an algorithm for covering the interior of a simple polygon by modifying the edge covering algorithm [9].

An orthogonal polygon is vertically convex if its intersection with every vertical line is a line segment or empty. The same rule goes for horizontally convex orthogonal polygons. Franzblau et al. [18] focused on vertically convex orthogonal polygons and developed an $O(n^2)$ time algorithm for the minimum covering problem. Keil [29] solved the covering problem for horizontally convex orthogonal polygons with an $O(n^2)$ time algorithm.

Tomás et al. developed an approximation algorithm to partition the polygon into pieces and use Constraint Programming techniques to model the optimization problem [43].

Krohn and Nilsson [36, 32] proved that vertex guarding a monotone polygon is NP-hard and developed an approximation algorithm for monotone polygon guarding. Their algorithm also works for interior guarding rectilinear polygons and produces a guard set of size $O(OPT^2)$, where OPT is the size of the smallest interior guard set.

2.4 Problem Analysis and Solution

Taking a look at the CLDP problem (2.7), we can see that if the number of the candidate light fixtures is N_c, the dimension of the solution space of problem is N_c. To find a minimal number of light fixtures for an arbitrary orthogonal polygon, the computational complexity is

$$O(N_c) \cdot O(C_{N_c}^{N}) = O(N_c^{N_c-N+1}),$$

which strongly depends on N_c.

In order to minimize the search scope, we decide to determine the lower bound N_{lb} and upper bound N_{ub} for the number N first and then to find the solution that satisfies the illuminance requirement by binary search.

2.4.1 The Lower Bound

For a given orthogonal polygon, we denote its area by S_{op}. According to Equation (2.2), we can obtain the effective illumination radius R of one light fixture. Thus, the effective illumination area of the light fixture is:

$$S_{lf} = \pi R^2.$$

Obviously, the number of light fixtures for the orthogonal polygon must be no less than $\lceil \frac{S_{op}}{S_{lf}} \rceil$. Therefore, we can easily get a lower bound N_{lb} for the minimum number of the necessary light fixtures:

$$N_{lb} = \lceil \frac{S_{op}}{S_{lf}} \rceil = \lceil \frac{S_{op}}{\pi R^2} \rceil \tag{2.11}$$

2.4.2 A Tighter Upper Bound

Our ultimate goal is to find a minimal number of light fixtures that can guarantee the illuminance at every observation point satisfies the illuminance threshold requirement. The illuminance is contributed by all light fixtures that can directly illuminate the point. Hence, we calculate the illuminance at each observation point j by adding the illuminance provided by all light fixtures:

$$I_j = \sum_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \ for\ j = 1, \ldots, m.$$

Nevertheless, the above requirement makes CLDP hard to solve. Since the illuminance is inversely proportional to the square of unobstructed distance between the light fixture and the observation point, the nearest light fixture supplies the biggest portion to the sum of the illuminance. If the illuminance supplied by the nearest light fixture is greater than the threshold at all observation points, the required number of light fixtures is an upper bound on the solution of the problem (2.7). In addition, the model is simpler if we only consider the nearest light fixture when calculating the illuminance. Therefore, we solve the following problem as an approximate solution to the original CLDP.

Based on this assumption, for the observation point j, when we only consider the nearest light fixture, which provides point j with the maximum illuminance, the illuminance at point j would be:

$$I_j = \max_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \ for\ j = 1, \ldots, m$$

To find a minimal number of light fixtures satisfying that for each observation point, considering that the illuminance from the nearest light fixture is greater than the threshold value, we formulate the problem (2.12) below.

A tighter upper bound for CLDP:

$$\begin{aligned} \text{minimize} \quad & N \\ \text{subject to} \quad & I_j = \max_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \ j = 1, \ldots, m. \\ & I_j \geq \delta, \ j = 1, \ldots, m. \end{aligned} \tag{2.12}$$

Clearly, the solution of Problem (2.12) is sufficient for solving Problem (2.7). In

other words, we can obtain an upper bound for the solution of Problem (2.7) by solving Problem (2.12),

2.4.3 Obtain A Minimum N

The main problem of this thesis is the CLDP problem (2.7). After we get the lower bound and the upper bound for the number of light fixtures N, we use binary search to find a minimum N that satisfis the constraints of the CLDP problem. We use the Hierarchical Clustering algorithm to judge if N is the number we want, since we don't have the initial locations of the light fixtures. Thus, we finally solve the CLDP problem.

2.4.4 A Bird View of the Solution

The flow of our approach is shown Figure 2.1. The SOP-L problem and the OPH-L problem are two cases of Problem (2.12), and they are formulated and solved in Chapter 3 and Chapter 4, respectively. The solution of the SOP-L/OPH-L problem provides an upper bound for the CLDP problem. With the lower bound and upper bound, we solve the CLDP problem in Chapter 5.

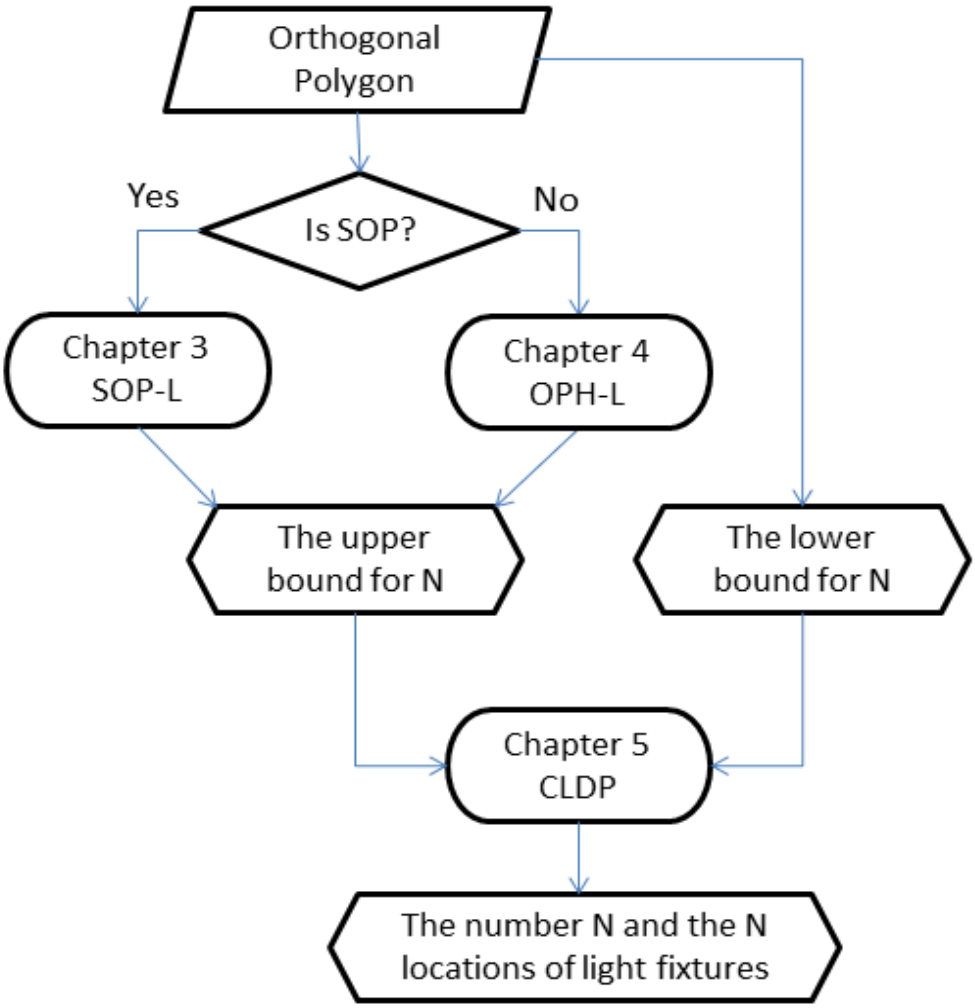

Figure 2.1: The main work of this theis

Chapter 3

A Tighter Upper Bound for CLDP in the case of SOP

3.1 Overview

In this chapter we will achieve a tighter upper bound for CLDP in the case of SOP by solving the following problem:

$$
\begin{array}{ll}
\text{input} & \text{The illuminance at the source: } I_0. \\
 & \text{The illuminance requirement threshold: } \delta. \\
 & \text{The vertex set of the SOP } P\text{: } V = v_1, v_2, ..., v_n. \\
 & \text{The observation point } j \text{ in the SOP, for } j = 1, \ldots, m. \\
\text{output} & \text{The locations of the } N \text{ light fixtures.} \\
\text{minimize} & N \\
\text{subject to} & I_j = \max_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \text{ for every observation point } j \text{ in the SOP.} \\
 & I_j \geq \delta, \text{ for every point } j \text{ in the SOP.}
\end{array}
\tag{3.1}
$$

There is a requirement for the illuminance, which means every light fixture has a limited coverage. The problem (3.1) is denoted as "SOP-L" since that its motivation is to light up a simple orthogonal polygon with light fixtures having limited coverage. We propose a heuristic algorithm based on a polygon partitioning technique and an iterative set redundancy removal algorithm to solve the SOP-L problem.

3.2 The SOP-UL Problem

Given a simple orthogonal polygon P with its vertex set $V = \{v_1, v_2, ..., v_n\}$, the solution for Problem (3.1) could vary depending on the illumination requirement. A lower requirement would offer more flexibility and more potential solutions. If we relax the requirement by setting δ to zero, then the goal is similar to light up a SOP with light fixtures having unlimited coverage. The relaxed problem is called "SOP-UL" problem:

$$\begin{array}{ll} \text{input} & \text{The illuminance at the source: } I_0. \\ & \text{The vertex set of the SOP } P\text{: } V = v_1, v_2, ..., v_n. \\ & \text{The observation point } j \text{ in the SOP, for } j = 1, \ldots, m. \\ \text{output} & \text{The locations of the } N \text{ light fixtures.} \\ \text{minimize} & N \\ \text{subject to} & I_j = \max_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \text{for every observation point } j \text{ in the SOP.} \\ & I_j > 0, \text{for every observation point } j \text{ in the SOP.} \end{array} \tag{3.2}$$

Obviously, the SOP-UL problem (3.2) only requires that each observation point be illuminated by at least one light fixture. Because there is no other requirement for illuminance, there is no need to calculate the illuminance I_j at the observation point j. In the other words, the light fixtures in the model have unlimited coverage. Actually, the SOP-UL problem (3.2) is the same as the classical Art Gallery Problem, which is formulated as follows:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{i=k} x_i \\ \text{subject to} & \bigcup_{i=1}^{i=k} x_i R_{c_i} = P, \ \forall \ x_i \in X, \ c_i \in C \\ & x_i \in \{0, 1\}, \ \forall \ i = 1, \ldots, k. \end{array} \tag{3.3}$$

where C is the set of candidate locations which has k elements and X is a binary vector whose size is k. c_i is the $i - th$ element of the candidate set C, and R_{c_i} represents the visibility region of the candidate c_i. If c_i is chosen to be a light, then the corresponding variable x_i of X is set to be 1; otherwise, x_i is set to be 0.

There are two kinds of algorithms: heuristic algorithms and approximation algorithms to solve the classical AGP. The details of these algorithms can be referred to in Chapter 2. We will develop a heuristic algorithm to solve the SOP-UL problem in this chapter.

3.2.1 Definitions and Propositions

We first explain several basic concepts.

- **Reflex Vertex**: If the internal angle of a vertex is greater than π, this vertex is called a reflex vertex.
- **Edge extension**: The two edges incident to a reflex vertex can be extended until they intersect the boundary of the orthogonal polygon.
- **Rectangle block**: With edge extensions, the whole orthogonal polygon can be divided into rectangle blocks.
- **Block number**: We assign an integer to each rectangle block as a block number. Every block number is unique.
- **Visibility**: An observation point p and a light g are visible to each other if point p can be illuminated directly by light g, in other words, the line segment between p and g is entirely included in the simple orthogonal polygon.
- **Coverage set**: The coverage set of a light fixture g, denoted by $CS(g)$, is a set containing all the block numbers of the rectangle blocks that are visible to the light fixture g.

We have the following propositions:

Proposition 1. *If all the four vertices of one block are visible to a light fixture g, then the entire block is visible to the light fixture g.*

Proof. We can prove this proposition by contradiction. Suppose there is a block that is not entirely visible to the light g, while all its four vertices are visible to the light g. Then there must be some other points or edges that obstruct the visibility from the light g. Since it is a simple orthogonal polygon, the obstacle must be part of the boundary of the orthogonal polygon, not a separate hole. Hence, the boundary

must obstruct the visibility from the light g to at least one vertex of the block, which contradicts to the condition. Hence, the proposition is correct for simple orthogonal polygons. □

Proposition 2. *Every rectangular block is entirely visible to at least one reflex vertex.*

Proof. Since the blocks are generated by edge extensions, we have four kinds of blocks:

1. The block has only one edge on an extended edge.
2. The block has two edges on the extended edges.
3. The block has three edges on the extended edges.
4. The block has four edges on the extended edges.

For the first case, the block has two possible kinds of shapes, which are shown by (a) and (b) in Figure 3.1. The solid edges are line segments on the polygon's boundary and the dotted edge is on the extended edge. Since the extended edge is a ray starting from a reflex vertex, at least one endpoint of the dotted edge is reflex vertex and the whole block is visible to this reflex vertex.

For the second case, the block has five kinds of shapes, shown by (c) to (g) in Figure 3.1. In (c), although no vertex of the block is reflex, the block must be visible to the reflex vertices that emit the block's dotted edges, because in a simple orthogonal polygon, if some other points obstruct the visibility between the reflex vertex and the block, there must be another reflex vertex that divides the block into smaller ones. In (d) to (k), at least one vertex of the polygon is reflex, and hence, the whole block is visible to the vertex.

For the third case, (h) and (i) give the two possible shapes. For the shape in (h), the block must be visible to three reflex vertices and the reason is similar to that for (c). In (i), the block is visible to at least two reflex vertices, which are its own vertices.

For the fourth case, similar to the shape in (c) and (h), the block is entirely visible to at least four reflex vertices.

Therefore, each rectangular block must be entirely visible to at least one reflex vertex. □

Proposition 3. *If the orthogonal polygon has reflex vertices, then the reflex set is a light set that can cover the whole polygon but may not be the minimum set.*

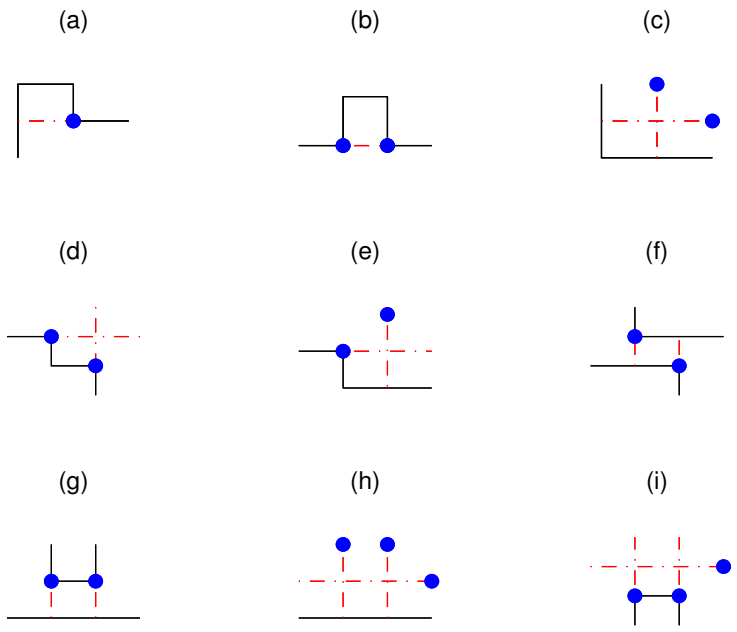

Figure 3.1: Block visibility

Proposition 4. *If every element in the coverage set of a light A is also an element in the coverage set of another light B, in other words,* $CS(A) \subseteq CS(B)$*, then the light A can be removed from the light set. Similarly, if* $CS(A_1 \bigcup A_2 \bigcup ... \bigcup A_m) \subseteq CS(B_1 \bigcup B_2 \bigcup ... \bigcup B_n)$*, the lights* A_1*,* A_2*, ... , and* A_m*, can be removed from the light set.*

3.2.2 A Heuristic Algorithm

According to Proposition 3, if we put a light fixture at every reflex vertex, the light set covers the whole polygon and the number of the light fixtures is $r = \frac{n-4}{2}$, where n is the vertices number of the orthogonal polygon. In many cases, r is greater than the minimum necessary number. One possibility is to remove the redundant reflexes from the light set, which means that if we remove one light from the light set, the remaining light set still can cover the whole polygon. Hence we can initialize the candidate light set with the reflex vertex set, and then obtain an improved light set by removing all the redundant light fixtures.

Another possible choice for the candidate light set is the set of all vertices of the

polygon. In this case, the initial number of the candidate light fixtures is n, almost double the size of the reflex vertex set, which potentially consumes more time when removing redundant lights.

The extended edges from the reflex vertices intersect with each other. The intersections also can be viewed as candidate locations for light fixtures. The number of intersections depends on the shape of the orthogonal polygon. While the number of intersections is normally smaller than r, it could be $O(4r^2) = O(n^2)$ in the worst case. If we add the intersections into the candidate light set, the size of the candidate light set is $O(n^2)$.

Therefore, the candidate light set has the following four choices:

- Type 1: the vertex set
- Type 2: the reflex set
- Type 3: the vertex set $\bigcup$ the intersection set
- Type 4: the reflex set $\bigcup$ the intersection set

We proposed Algorithm 1 which obtains a minimal light set by removing redundant lights from the candidate light set. The choice of the candidate light set must be provided for Algorithm 1 because of its great influence on the final result.

3.2.3 Redundancy Removal Algorithm

In our method, we need to reduce the redundancy of the coverage sets. In fact, this is a special case of the Set Covering Problem. The Set Covering Problem was proved to be NP-complete in 1972, and it was one of Karp's 21 NP-complete problems.

Our objective is to remove redundant light fixtures according to the relationships between these coverage sets. Algorithm 2 is proposed for this aim based on Proposition 4.

3.3 Performance Evaluation

3.3.1 The Choices for Candidate Light Sets

The initial candidate light set greatly influences the performance of the algorithm. A larger number of candidates will cost more time. We compare the time consumption

Algorithm 1 Simple Orthogonal Polygon Lighting Algorithm.

Input:

The vertex set V of a simple orthogonal polygon;
The type of the candidate light set;

Output:

An improved light set

1: **if** The number of vertices is 4 **then**
2: **return** the centre of the polygon and end the algorithm
3: **else**
4: Determine the set of reflex vertices
5: **end if**
6: Construct edge extensions at reflex vertices. The extended edges divide the whole orthogonal polygon into orthogonal blocks.
7: Label the blocks with integers.
8: Initialize the candidate light set according to the selected type.
9: Calculate the coverage set for every candidate light fixture according to Proposition 1.
10: Use Algorithm 2 to remove redundant lights.
11: **return** the results of Algorithm 2

and the final solutions of the four different candidate light sets: (1) the vertices, (2) the reflex vertices, (3) the reflex vertices $\bigcup$ the intersections of the edge extensions, (4) the vertices $\bigcup$ the intersections of the edge extensions, respectively. Figure 3.2 shows the results. The blue solid line shows the boundary of the orthogonal polygon. The red solid points represent the necessary light fixtures, which is the final solution obtained by Algorithm 1. The two rules apply to all figures in this thesis.

Usually, a larger candidate light set provides more candidates than a smaller candidate set, which gives it higher probability to get a better solution. From Figure 3.2 we can see that the last cases require fewer light fixtures than the first two cases.

We apply our algorithm on the SOPs with various number of vertices. For each case, we generate 100 SOP instances randomly and run the algorithm to obtain the light set for the four different candidate light sets: (1) the vertices, (2) the reflex vertices, (3) the reflex vertices $\bigcup$ the intersections of the edge extensions, (4) the vertices $\bigcup$ the intersections of the edge extensions, respectively. The average numbers of light fixtures "Ave1", "Ave2", "Ave3", and "Ave4" corresponding to the above four candidate light sets are shown in Table 3.1. In Table 3.2, "Min1", "Min2", "Min3", and "Min4" give the minimum number of light fixtures in the 100 SOP instances for the four candidate light sets, respectively. Similarly, "Max1", "Max2", "Max3", and

Algorithm 2 Redundancy Removal Algorithm.

Input:

The coverage sets $C = C(1), C(2), ..., C(r)$ of a simple orthogonal polygon.

Output:

A minimal light set

1: Construct a *candidate matrix*, denoted by $M(C)$, for the coverage sets. We use row vectors to denote the coverage sets. The candidate matrix is the combination of these row vectors. We use zeros to pad each row vector so that each row has the same length as the longest row vector.
2: **while** $M(C)$ is NOT empty **do**
3: Count the frequency of each element.
4: Store the rows containing elements that only appear once in the *necessary matrix*, denoted by $M(N)$.
5: **if** All elements in the candidate matrix $M(C)$ appear more than once **then**
6: Store the longest row in the necessary matrix $M(N)$
7: Remove these rows from the candidate matrix $M(C)$.
8: **end if**
9: Define an *elements summary vector*, denoted by $ESV(N)$, as the aggregation of all elements in the necessary matrix. Check the elements in the rest of the candidate matrix.
10: **if** One element (leave out the zeros) is included in the elements summary vector $ESV(N)$ **then**
11: Remove this element from the candidate matrix $M(C)$.
12: **end if**
13: The remaining rows form the new candidate matrix $M(C)$.
14: Update $M(N)$ and $ESV(N)$.
15: **end while**
16: **return** all the light fixtures corresponding to $M(N)$ as a minimal set of light fixtures.

"Max4" gives the maximum number of light fixtures. We can see that the maximum number of light fixtures is not greater than the well known upper bound $\lfloor \frac{n}{4} \rfloor$, while the average number of light fixtures is smaller than $\lfloor \frac{n}{5} \rfloor$, where n is the number of vertices.

Figure 3.3 gives a more intuitive result for Table 3.1. In the figure, "Ave3" is always smaller than "Ave1", "Ave2" and "Ave4", meaning that the algorithm obtains better solution when the candidate set is the union of the reflex vertices and the intersections of the edge extensions.

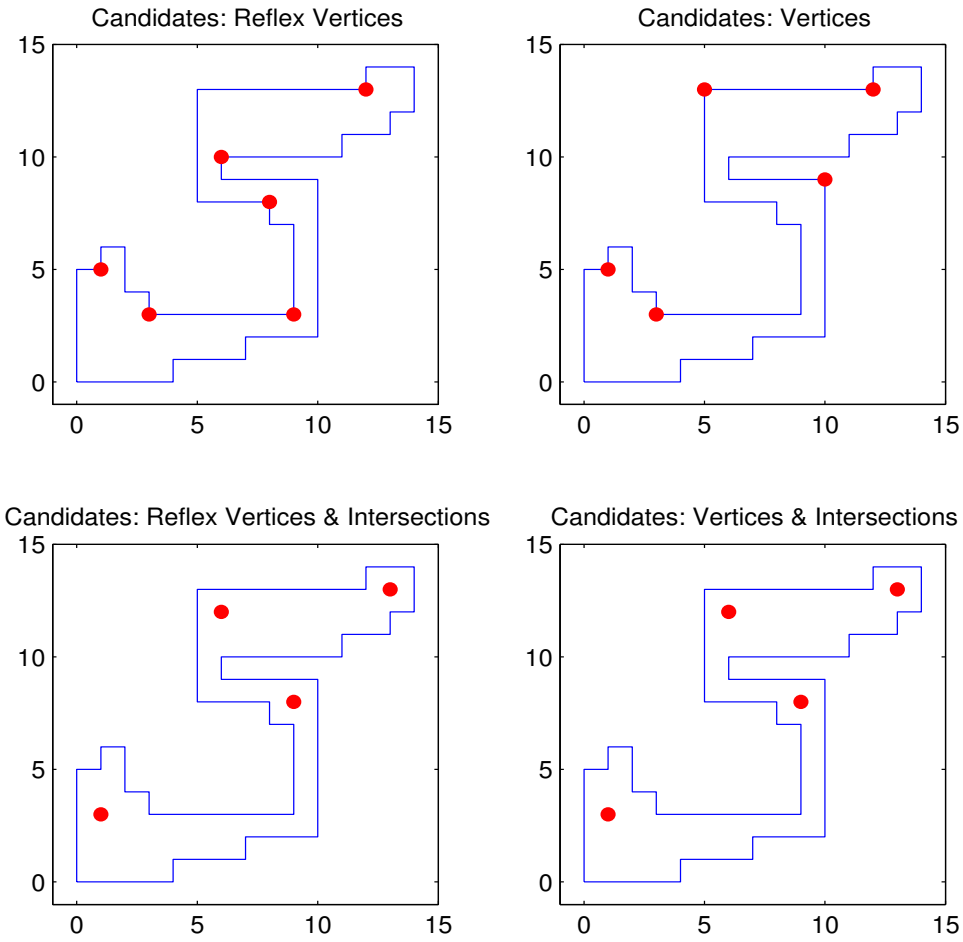

Figure 3.2: Results obtained from different candidate light sets

3.3.2 Time Complexity Analysis

Suppose the number of vertices of the simple orthogonal polygon is n, and the number of reflex vertices is r, which satisfies $n = 2r + 4$. The time complexity of determining the reflex set is $O(n)$; constructing the edge extension needs $O(r)$ time; determining the intersections of edge extensions costs $O(r^2)$; labeling the blocks costs $O(n^2)$; calculating the coverage costs $O(n^3)$; the redundancy removal costs $O(nr)$. Hence, the time complexity of this algorithm is $O(n^3)$.

Table 3.1: Average number of light fixtures

Vertices Num	Ave1	Ave2	Ave3	Ave4
20	3.96	3.98	3.58	3.66
30	5.74	5.66	5.20	5.22
40	7.68	7.60	6.90	6.94
50	9.48	9.22	8.36	8.46
60	11.26	11.16	9.96	10.16
70	13.28	13.12	12.04	12.14
80	15	14.90	13.60	13.74
90	17.40	17.08	15.54	15.58
100	18.80	18.62	17.16	17.24

Table 3.2: Min/Max number of light fixtures

Vertices Num	Min1	Min2	Min3	Min4	Max1	Max2	Max3	Max4
20	3	3	2	2	5	5	4	5
30	4	3	3	3	7	7	7	7
40	6	6	4	4	10	10	9	10
50	7	7	7	7	12	12	10	11
60	9	9	8	8	14	14	12	12
70	10	10	9	9	16	17	15	15
80	12	12	10	10	18	18	17	17
90	14	14	12	12	22	20	19	19
100	16	16	14	14	22	22	19	20

3.4 The SOP-L Problem

In the above section, we discussed how to get a minimal number of light fixtures for simple orthogonal polygons without considering the illuminance value. In this section we solve the problem when there is a higher illumination requirement, i.e., the illumination threshold $\delta > 0$.

According to Equation (2.2) we introduced in Chapter 2, we can get the effective illumination radius R of one light fixture:

$$R = \sqrt{\frac{I_0}{\delta} - 1}.$$

Taking advantage of Algorithm 1, we make the following changes:

- If the diagonal of one rectangle block is greater than the radius R of the coverage area of one light fixture, we split this rectangle block into several pieces until

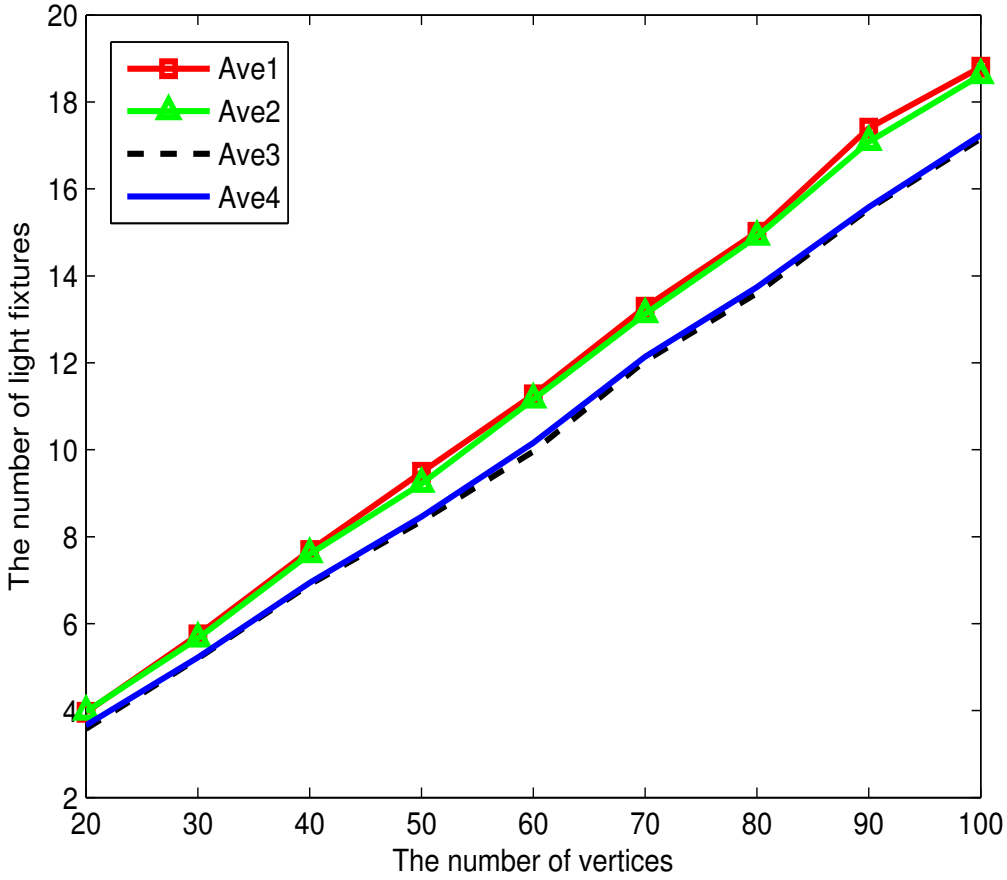

Figure 3.3: Average number of light fixtures for different candidate light sets

all rectangle blocks can be covered by one light fixture. When we split one rectangle block, we introduce another important set: the secondary-point set. A secondary point is a vertex shared by two new adjacent blocks generated by splitting and it is a good candidate location for a light fixture.

- We use the combination of the reflex vertex set and the secondary-point set as the candidate light set.

With the above two changes, Algorithm 1 also works when the illumination threshold $\delta > 0$.

After the light set is determined, we need to choose a good location for each light fixture in order to minimize the range differences.

Figure 3.4 shows the number of light fixtures required for 50 SOP instances with

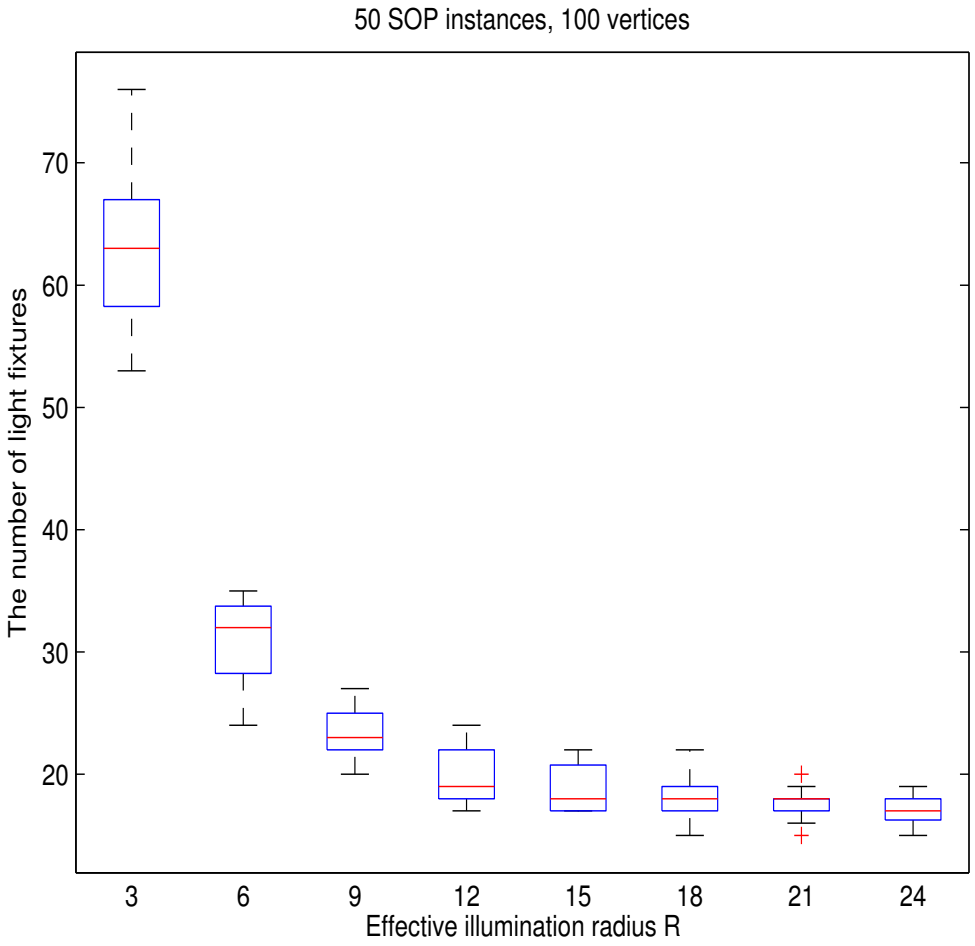

Figure 3.4: The number of light fixtures for SOPs vs. the effective illumination radius

100 vertices for each when we use different illuminance requirement. From the figure we can see that the number of light fixtures decreases when the effective illuminance radius R increases.

Chapter 4

A Tighter Upper Bound for CLDP in the case of OPH

4.1 Overview

An orthogonal polygon with h holes (OPH) is an orthogonal outer boundary P with h disjoint simple orthogonal polygons $H_1, ..., H_h$ in its interior. We denote this orthogonal polygon with h holes as $P - H_1 \bigcup ... \bigcup H_h$.

In this chapter we give the CLDP in the case of OPH. In particular, we consider the following problem:

$$
\begin{array}{ll}
\text{input} & \text{The illuminance at the source: } I_0. \\
 & \text{The illuminance requirement threshold: } \delta. \\
 & \text{The vertex set of the SOP } P\text{: } V = v_1, v_2, ..., v_n. \\
 & \text{The observation point } j \text{ in the SOP, for } j = 1, \ldots, m. \\
\text{output} & \text{The locations of the } N \text{ light fixtures.} \\
\text{minimize} & N \\
\text{subject to} & I_j = \max_{i=1}^{N} \frac{I_0}{1 + d_{ji}^2}, \text{for every point } j \text{ in the OPH.} \\
 & I_j \geq \delta, \text{for every point } j \text{ in the OPH.}
\end{array}
\tag{4.1}
$$

The above problem is called "OPH-L" problem since it is to light an OPH with light fixtures of limited coverage range.

O'Rourke [37] showed that $\lfloor \frac{n+2h}{3} \rfloor$ vertex light fixtures are always sufficient to light any orthogonal polygon with n vertices and h holes. Hoffmann and Kriegel [22] proved that at most $\frac{n}{3}$ vertex light fixtures are needed for an orthogonal polygon with n vertices and arbitrary number of holes. Michael and Pinciu [35] improved the bound to $\frac{17n-8}{52}$.

We will apply the approach for the "SOP-L" problem 3.1 to the "OPH-L" problem 4.1. In the second section, we will analysis the different conditions caused by holes and discuss the feasibility of the proposed method for OPH. In order to evaluate the algorithm, a great number of OPHs must be generated, and as such a simple OPH generation algorithm is developed in the third section. The last section will evaluate the performance of the algorithm for OPHs.

4.2 Algorithm Improvement for OPH

Proposition 1 in Chapter 3 is no longer applicable for OPH. Figure 4.1 shows a special case of Proposition 1 when the orthogonal polygon has one hole. In the figure, the outer boundary of the orthogonal polygon and the hole are plotted with a black line and the blue dotted lines are the extended edges. Since the four points A, B, C, D are visible to the reflex vertex g. According to Proposition 1, the rectangular block $ABCD$ should be visible to g. But this is not the case here because of the hole.

Fortunately, we only need to revise Proposition 1 by adding a constraint, so that the revised proposition can be applicable for OPHs. The proposition is revised as follows:

Proposition 5. *If all four vertices of one block are visible to light fixture g, and there is no other vertex inside the maximum convex polygon shaped by the light fixture g and some vertices of that block, then the entire block is visible to light fixture g.*

Proof. This proposition can be proved by contradiction. Suppose block $ABCD$ is a small block generated by the edge extensions of an orthogonal polygon P, and all four vertices of block $ABCD$ in Figure 4.2 are visible to light fixture G_1 but the block is not entirely visible to light fixture G_1. The maximum convex polygon shaped by light G_1 and some vertices of block $ABCD$ is G_1ABCD. Since block $ABCD$ is not entirely visible to light fixture G_1, there must be some vertices of the polygon P inside the polygon G_1ABCD blocking the light from light fixture G_1. This contradicts with the

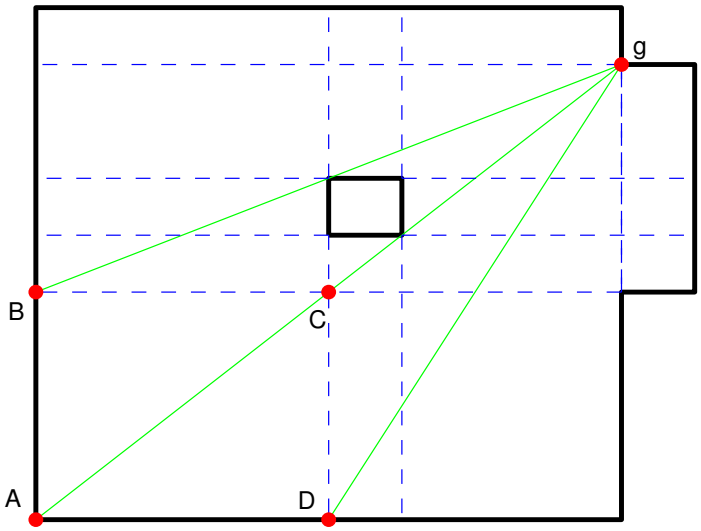

Figure 4.1: Special case in OPH

condition "there is no other vertex inside the maximum convex polygon". Therefore, the proposition is correct for orthogonal polygons. □

4.3 Experimental Evaluation

4.3.1 OPH Generation

In order to evaluate the performance of our algorithm, we need to generate different OPH instances. Tomás et al. [40, 42, 41] developed an Inflate-Cut algorithm to generate random orthogonal polygons without holes. Currently, there are very few papers or documents that give algorithms for randomly generating OPHs. In this section, we develop an algorithm based on the "Inflate-Cut" technique for this purpose.

The "Inflate" operation on a cell c, which is a unit grid, transforms cell c into a four-cell square and transforms all other cells in the same row into two rows of cells, and transforms cell c and all other cells in the same column into two columns

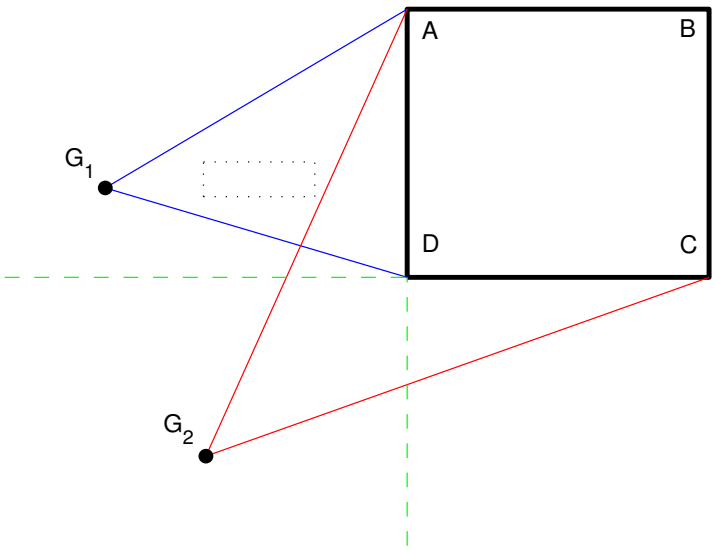

Figure 4.2: Figure for the proof of proposition 5

of cells. In Figure 4.3, the cell in subfigure (a) is inflated into four cells in subfigure (b). In Figure 4.4, the red-filled cells in subfigure (a), (c)and (e) are inflated into the red-filled squares in subfigure (b), (d) and (f), respectively. All other cells in the same row are inflated vertically and all other cells in the same column are inflated horizontally.

The "Cut" operation works on a inflated square, and it cuts one cell from the four-cell square to obtain one more reflex vertex for the polygon. In Figure 4.3, the four-cell square in subfigure (b) has four random results in subfigure (c), (d), (e) and (f) for the "Cut" operation.

Algorithm 3 can generate an OPH that has *num_vtc_plg* vertices on its outer boundary and *num_holes* holes. The numbers of the vertices of the holes are stored in the vector *num_vtc_holes*. We generate a big square that includes at least $9 \cdot$ *num_holes* cells, and then pick *num_holes* nonadjacent cells that don't share any edge with the boundary of the big square. These cells are the initializations of the *num_holes* holes and we will convert them into the *num_holes* holes with required number of vertices by "Inflate" and "Cut" operations. After all holes are generated,

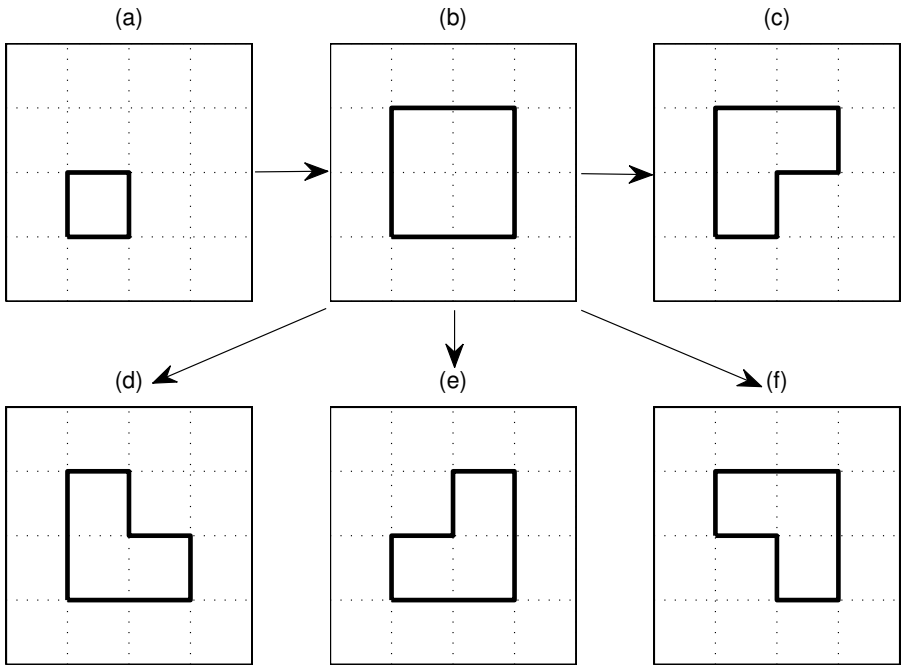

Figure 4.3: Inflate-Cut example 1

we will convert the big square into an orthogonal polygon with num_vtc_plg vertices. The operation is given in detail in Algorithm 3.

4.3.2 Evaluation Results for the OPH-UL Problem

We relax the "OPH-L" problem to "OPH-UL" problem by ignoring the illumination constraint.

Figure 4.5 and Figure 4.6 show the results of Algorithm 3 for an orthogonal polygon with one hole and two holes, respectively. In the figures, the candidate set is initialized by the reflex vertices, the vertices, the reflex vertices and the intersections, and the vertices and the intersections, respectively. The red points represent the light fixtures.

Table 4.1 shows the results for several cases. We generate 100 orthogonal polygons with holes for each case. In the table, "NVplg" gives the number of polygon vertices, "NVhole" gives the number of hole vertices, "AVEgds" gives the average number of light fixtures, "MAXgds" gives the max number of light fixtures, and "AVEtime" gives the average running time.

Algorithm 3 OPH Generation

Input:

The number of vertices on the outer boundary num_vtc_plg;
The number of holes: num_holes;
The vector of the numbers of the holes' vertices: num_vtc_holes.

Output:

The list of vertices on the outer boundary and the vertex lists of all holes.

Generate a square, whose side length is $\lceil\sqrt{9 \cdot num\_holes}\rceil + 1$. The boundary of this square is represented by its four vertices ordered clockwise on the square.
Pick num_holes cells in the square and store the vertices of each cell in clockwise order around the block respectively.
for each hole **do**
  while its current number of vertices is less than the required number **do**
    Pick one cell c randomly in that hole.
    Apply the "Inflate" operation on cell c.
    Apply the "Cut" operation on the four-cell square inflated from cell c.
    Update the vertex list of the outer boundary and the vertex lists of all holes.
  end while
end for
while the current number of vertices of the outer boundary is less than the required number num_vtc_plg **do**
  Inside the outer boundary, pick one cell c outside all holes randomly.
  Apply the "Inflate" operation on cell c.
  Apply the "Cut" operation on the four-cell square inflated from cell c.
  Update the vertex list of the outer boundary and the vertex lists of all holes.
end while
return The vertex list of the outer boundary and the vertex lists of all holes.

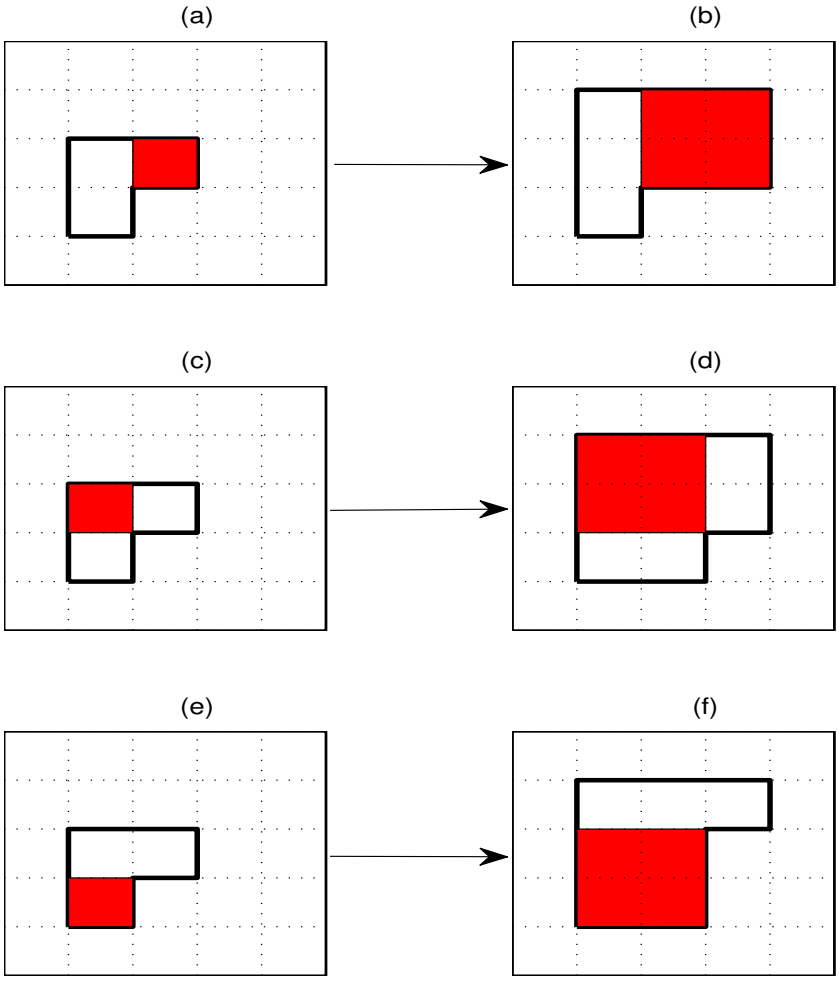

Figure 4.4: Inflate-Cut example 2

Figure 4.7 shows the relationship between the number of hole vertices and the number of light fixtures. The dotted line is obtained by linear regression. The average number of light fixtures is about $\lfloor \frac{1}{5.13} \rfloor$ of the total number of vertices on the outer boundary and the holes.

Table 4.2 shows the relation between the number of light fixtures and the number of holes. In the table, "Nhole" gives the number of holes. In this experiment, each hole has four vertices.

Figure 4.8 shows the relationship between the number of holes and the number

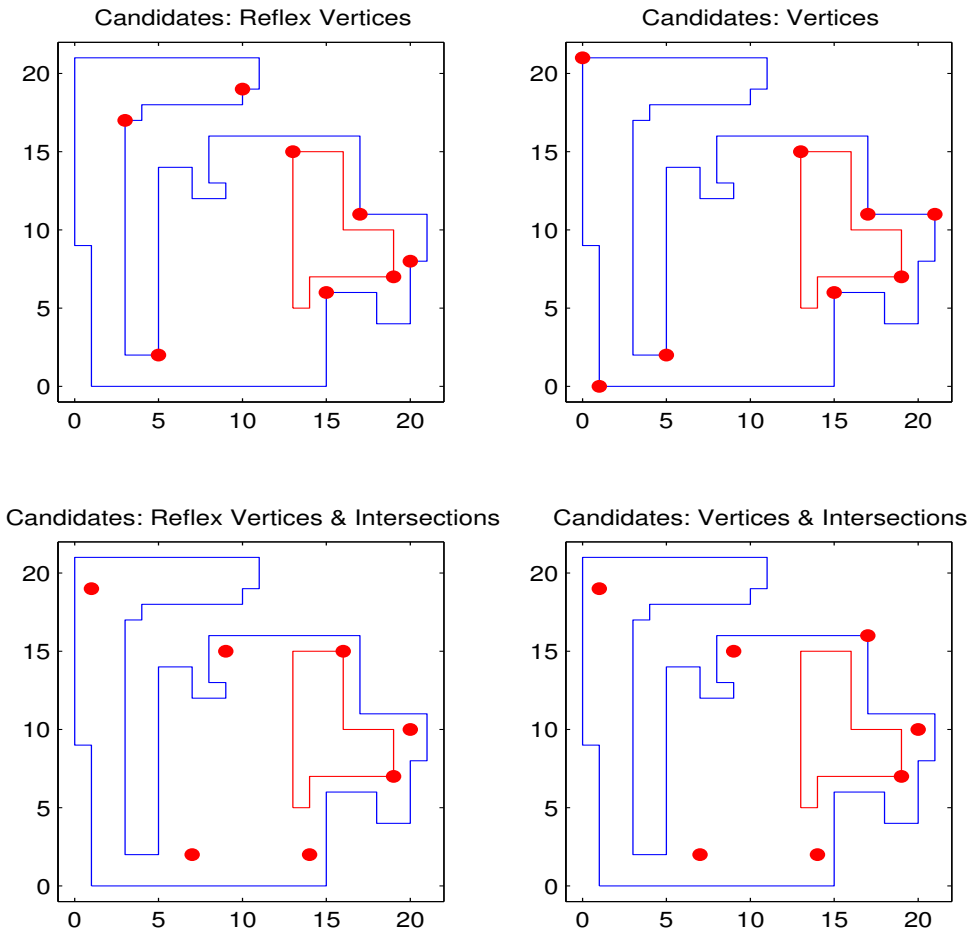

Figure 4.5: Orthogonal polygon with one hole

Table 4.1: Number of vertices vs. number of light fixtures

NVplg	NVhole	AVEgds	MAXgds	AVEtime(s)
30	4	6.64	8	0.0908
30	6	7.00	9	0.0964
30	8	7.64	9	0.0995
30	10	7.86	10	0.1195
30	12	8.38	10	0.1342
30	14	8.38	11	0.1454
30	16	8.94	11	0.1485
30	20	9.46	12	0.1716

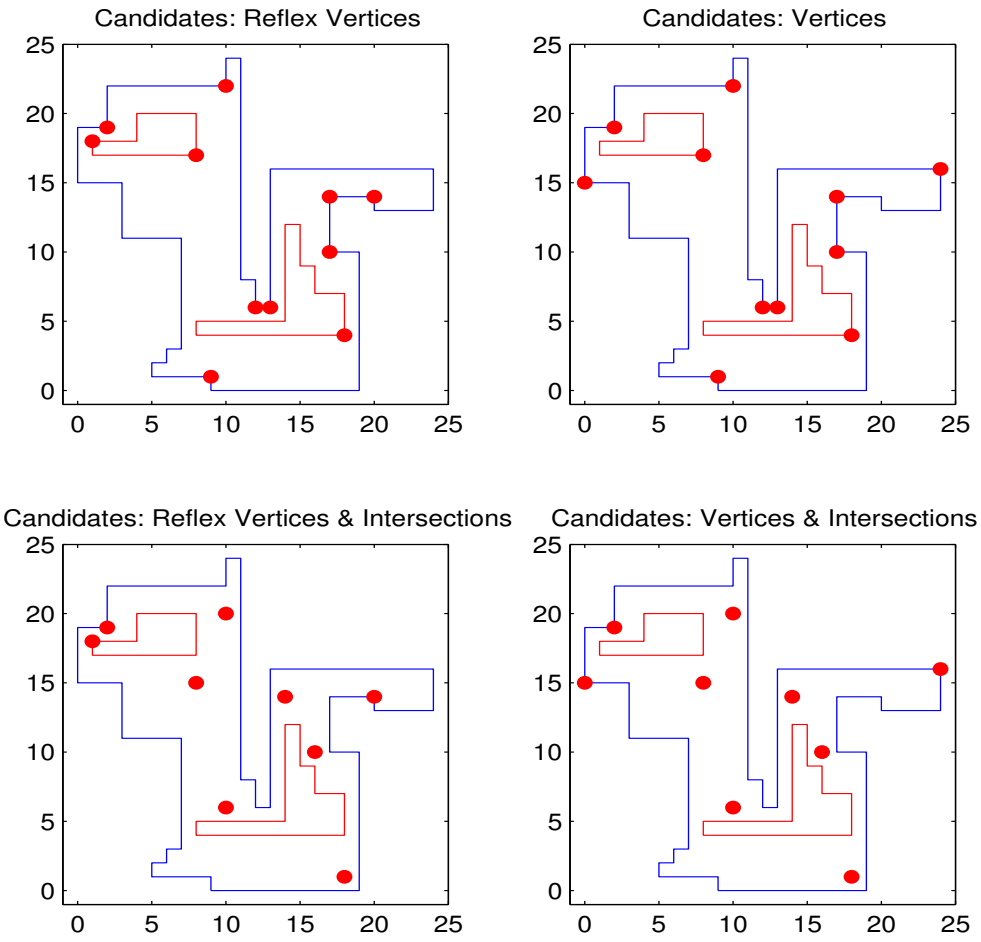

Figure 4.6: Orthogonal polygon with two holes

of light fixtures. When the number of vertices on the outer boundary is fixed, the ratio of the number of light fixtures to the total number of vertices decreases with the increasing number of holes. The ratio becomes approximately $\lfloor \frac{1}{6.5} \rfloor$ when there are 30 holes.

4.3.3 Evaluation Results for the OPH-L Problem

For the OPH-L problem, we also partition large blocks into smaller ones to make sure the effective illumination radius R is greater than every block's diagonal. The candidate light set is the combination of the reflex vertex set and the secondary-point set.

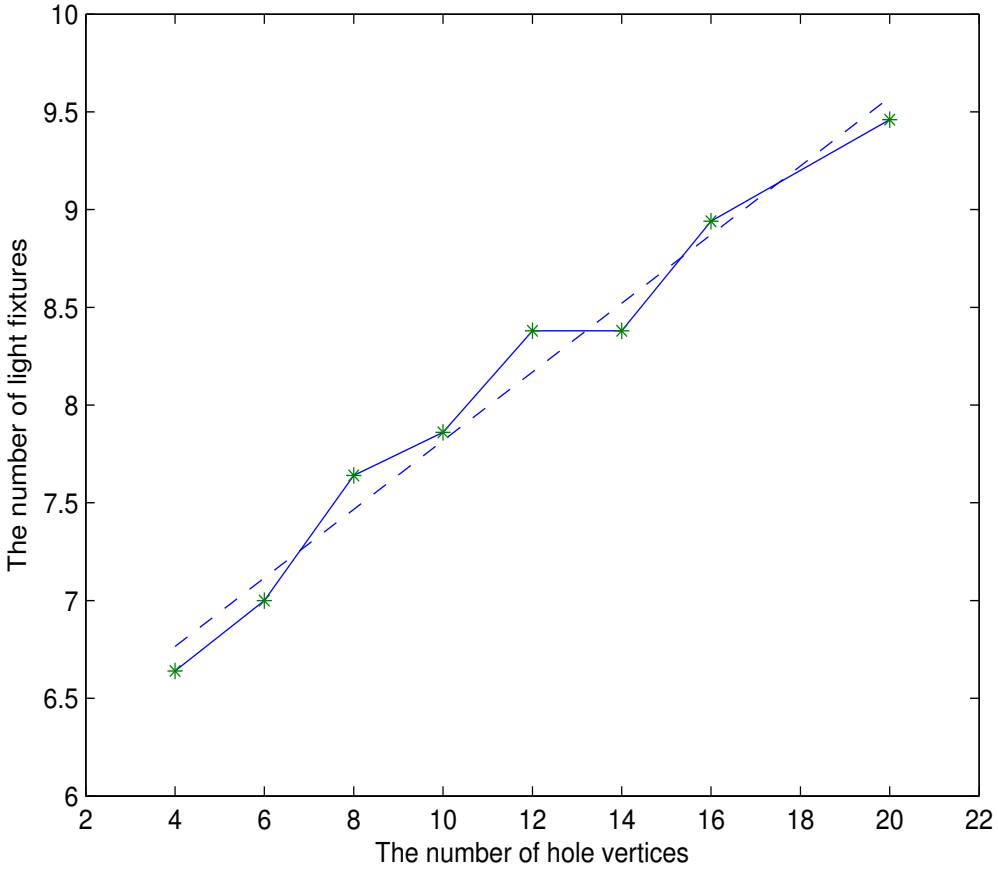

Figure 4.7: The number of hole vertices vs. the number of lights

Table 4.2: Number of holes vs. number of lights

NVplg	Nhole	AVEgds	MAXgds	AVEtime(s)
30	0	5.30	7	0.0474
30	1	6.64	8	0.0908
30	2	7.26	9	0.1273
30	3	8.18	10	0.1891
30	4	8.50	11	0.2543
30	5	9.16	12	0.2989
30	10	11.48	14	0.8808
30	15	13.08	16	1.5375
30	20	14.22	18	2.8523
30	30	17.24	23	6.1985

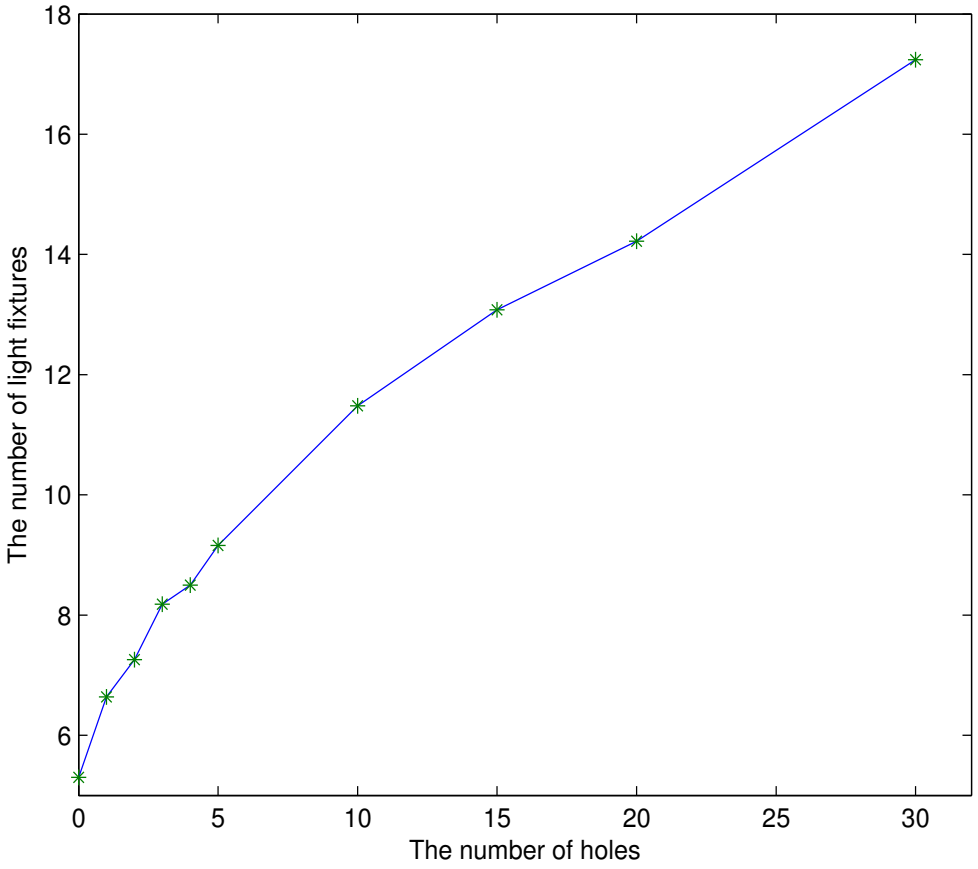

Figure 4.8: The number of holes vs. the number of lights

We generate 50 OPH instances for the cases that the outer boundary has 30, 50, 70, 90 vertices, respectively. Each instance has 3 holes. Figure 4.9 shows the relationship between the number of light fixtures and the effective illumination radius R. The number of light fixtures decreases when the effective illuminance radius R increases. The graphs decrease steeply at first. As they move to the right, they decrease very slowly.

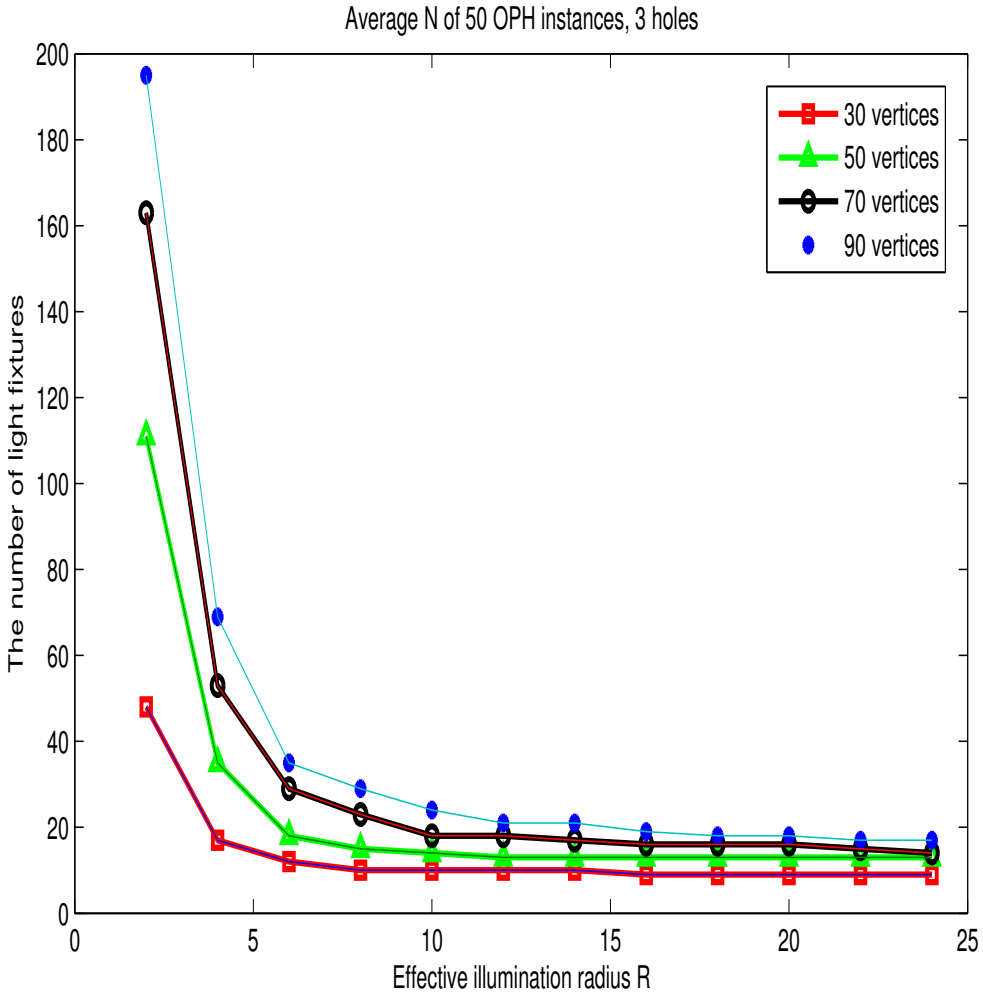

Figure 4.9: The number of light fixtures for OPHs vs. the effective illumination radius

Chapter 5

Solving CLDP with Binary Search and Clustering

5.1 Overview

The main objective of this thesis is to find a minimal set of light fixtures for an orthogonal polygon under illumination requirements. When the illumination requirement is given, we can obtain the effective illumination radius of the light fixtures. In Chapter 2 we used a simple method to calculate the lower bound on the number of necessary light fixtures, and in Chapter 3 and Chapter 4, we discussed how to determine the upper bound on the number of necessary light fixtures for a SOP and an OPH. This chapter proposes an algorithm to calculate the necessary light fixtures using binary search and a clustering algorithm.

5.2 Algorithm

We propose Algorithm 4 to find the necessary number of light fixtures. For the algorithm, we need to define the following notations and variables:

- n_{lb}: The lower bound on the number of necessary light fixtures.
- n_{ub}: The upper bound on the number of necessary light fixtures.
- n_s: The variable representing the number of light fixtures in the algorithm, which changes in each iteration.

Algorithm 4 Binary Search for Necessary Number.

Input:
The lower bound on the number of necessary light fixtures, n_{lb};
The upper bound on the number of necessary light fixtures, n_{ub};
The vertex list of the orthogonal polygon

Output:
The number of necessary light fixtures, N;
The locations of the N light fixtures

1: Initialize the lower bound n_{lb} and the upper bound n_{ub}
2: **while** $n_{lb} < n_{ub}$ **do**
3: Update the variable $n_s = \frac{n_{lb}+n_{ub}}{2}$
4: Detect whether n_s light fixtures are enough to light up the polygon by using Algorithm 5
5: **if** n_s light fixtures are enough to light up the polygon **then**
6: Update the upper bound $n_{ub} = n_s$
7: **else**
8: Update the lower bound $n_{lb} = n_s + 1$
9: **end if**
10: **end while**
11: **return** $N = n_{ub}$ and the locations of the centers obtained by Algorithm 5.

5.3 Determine Whether k Lights Are Enough

5.3.1 Clustering with Unobstructed Distance

Obviously, the light rays emanating from the light fixture could be blocked because of the presence of walls and some observation points may not receive light from the light fixture. As a requirement, the observation point should not belong to a cluster if it is not visible from the centre of the cluster.

In the second step of Algorithm 4, we use the clustering algorithm to detect whether n_s light fixtures are enough to light up the polygon and whether the centres of the clusters are good locations to place the lights. We divide the observation points into n_s clusters according to the unobstructed distance.

5.3.2 Hierarchical Clustering and Adjustment

When we need to determine whether k lights are enough to light up an orthogonal polygon, we can use a clustering algorithm to divide the observation points into k clusters based on the unobstructed distance. We use unobstructed distance to show

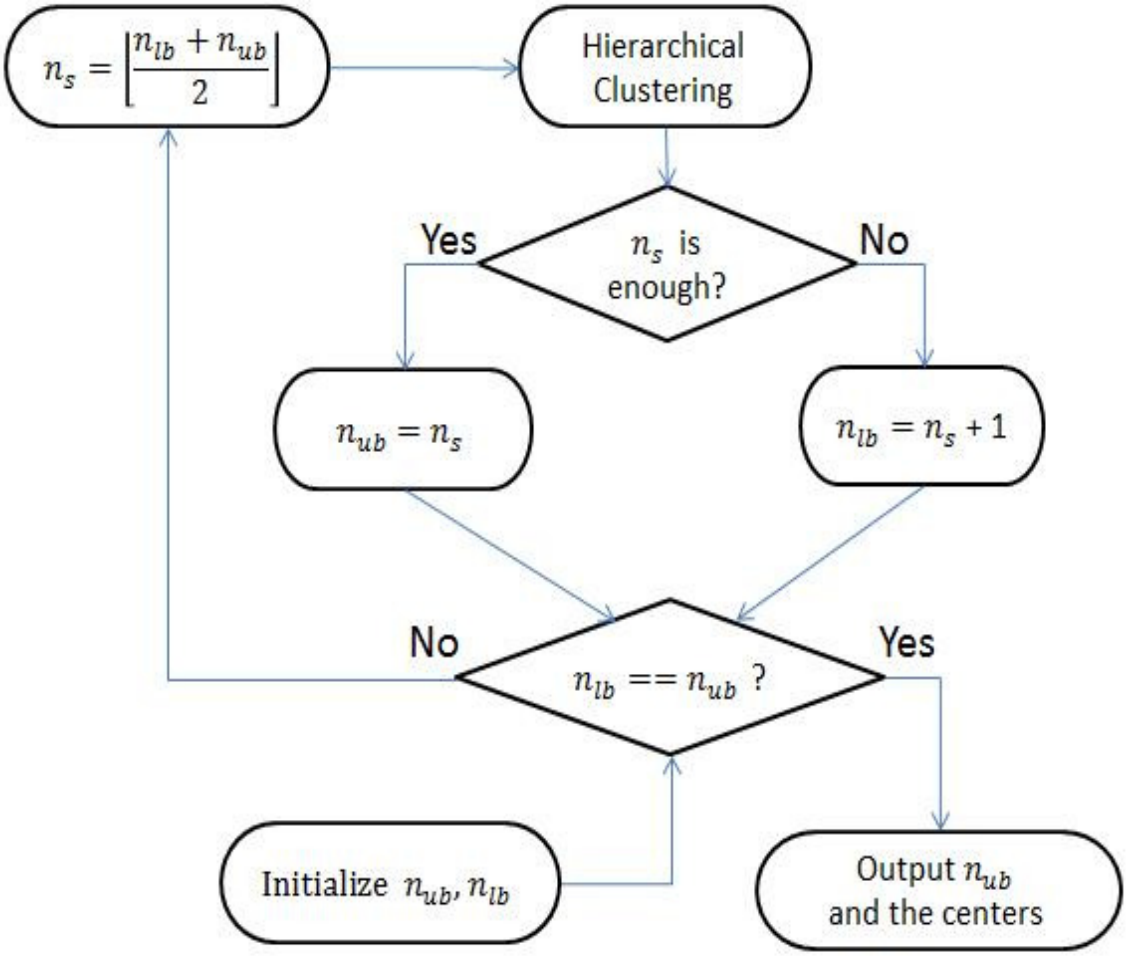

Figure 5.1: Search for the necessary number of lights

the visibility in our problem because an observation point cannot be assigned to a cluster if it cannot receive light from the centre. Since the performance of the clustering algorithms should be resilient to the initial selection of centres, we choose the Hierarchical Clustering algorithm [28, 3].

The Hierarchical Clustering algorithm, however, only divides the observation points into a number of clusters but doesn't provide the cluster centres. We need to calculate the location of the centres after the Hierarchical Clustering algorithm is executed. The k-means algorithm uses the average coordinates of each cluster as the cluster's centre. In our problem, averaging the coordinates may not give a point inside the polygon. We denote the sum of the unobstructed distances between one point p to the other points q in the same cluster as the cluster distance of p, which

can be formulated as the following equation:

$$d_{pc} = \sum_{q} d_{pq}.$$

For each cluster, we define the point of the discrete set of observation points having the minimum cluster distance as the cluster centre.

Proposition 6. *For the observation point p, if the unobstructed distance to cluster centre g is smaller than the unobstructed distance the other cluster centres, point p should be assigned to the cluster of g.*

According to the above proposition, we adjust the clusters after obtaining the hierarchical clusters and corresponding centres. After that, we calculate the new centres of the adjusted clusters.

Now we check whether each observation point is covered by at least one light or not. In other words, if the unobstructed distance between each observation point and its cluster centre is a finite value, then k lights are enough to light up the polygon; otherwise, k lights are not enough.

Based on the above analysis, we propose Algorithm 5 to judge whether n_s light fixtures are enough to light up the polygon or not.

5.4 Experimental Evaluation

In our experiments, we suppose that the effective illumination radius of each light fixture is 3, which means the illuminance at one observation point satisfies the requirement if only if the unobstructed distance between this point and the nearest light fixture is no greater than 3. For a given orthogonal polygon, we calculate the upper bound on the necessary light fixtures first. Figure 5.2 uses red solid points to represent the light fixtures, the green dotted line around each red solid point represents the effective illumination boundary of the light fixture. The figure shows that we need 8 light fixtures to light up the orthogonal polygon.

Figure 5.3 shows the Hierarchical Clustering results and the adjusted clustering results.

In the first run, $n_{lb} = 5$ and $n_{ub} = 8$, and we can get $n_s = \frac{n_{lb}+n_{ub}}{2} = 6$. We execute the judgment algorithm for $k = 6$ clusters. The blue solid squares in Figure 5.4 represent the observation points that don't satisfy the illumination requirement.

Algorithm 5 Determining whether n_s light fixtures are enough to light up the polygon

Input:
The assumed number of necessary light fixtures, n_s;
The vertex list of the orthogonal polygon
Output:
True or False for the judgement: n_s light fixtures are enough to light up the orthogonal polygon
1: Calculate the unobstructed distance between each pair of observation points;
2: Construct the clusters using the Hierarchical Clustering algorithm;
3: Calculate the centre of each cluster using Proposition 6;
4: Calculate the unobstructed distance between each observation point and each centre and reassign each point to the cluster whose centre is nearest to the point;
5: Calculate the new centres;
6: Calculate the unobstructed distance between each observation point and its centre;
7: **if** An infinite distance exists **then**
8: **return** False
9: **else**
10: **return** True
11: **end if**

In the second run, $n_{lb} = 6$ and $n_{ub} = 8$, and we can get $n_s = \frac{n_{lb}+n_{ub}}{2} = 7$. We execute the judgement algorithm for $k = 7$ clusters. Figure 5.5 shows that every observation point recieves enough light from its cluster centre. Therefore, the necessary number is 7 and the locations for the light fixtures are marked as the red solid points in the second subfigure of Figure 5.5.

5.4.1 Considerations on the Observation Points

In Algorithm 4 we deal with the observation points in the orthogonal polygon. Assume that the orthogonal polygon is formed by a great number of small individual squares, which are called pixels in this thesis. The observation points we used in the algorithm are the vertices of these pixels. If the size of a pixel is too large, then there may exist cases that even the four corners of the pixel are covered, the internal space of the pixel may not receive enough light.

Taking a look at the second subfigure of Figure 5.5, we can observe that some pixels may not be illuminated by the same light fixture. We summarized four kinds of worst special cases, which are shown in Figure 5.6. In the figure, the effective

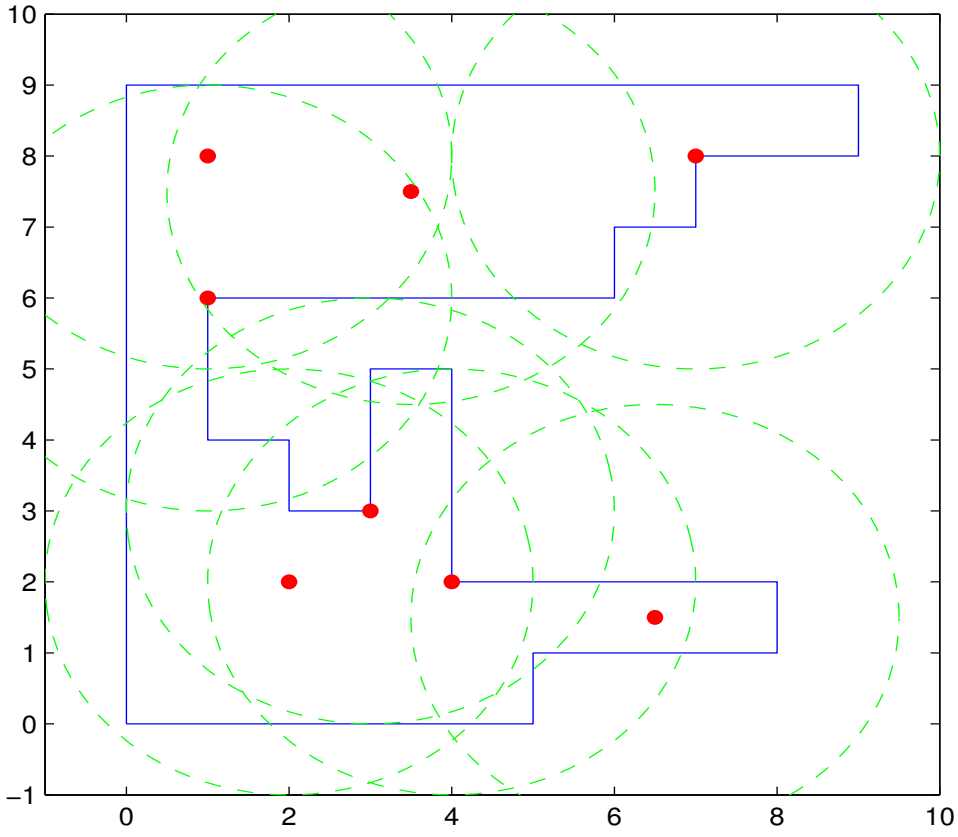

Figure 5.2: The upper bound results

illumination radius of each light fixture equals the side length of a pixel, and the four observation points are the vertices of the pixel, represented by the blue square frame. We use red solid points to represent the light fixtures. The black bold lines represent the possible locations for the light fixtures. The four observation points in Case (a) and (b) are illuminated by two different light fixtures, and they are illuminated by three and four different light fixtures, respectively, in Case (c) and (d). Although the four vertices are illuminated, how about the other points inside the pixel?

Proposition 7. *For a square pixel, whose side length is smaller than the effective illumination radiusof the light fixture, if the illuminance at its four vertices satisfies the illumination requirement, even though the vertices belong to different clusters, then the illuminance in the whole pixel satisfies the requirement.*

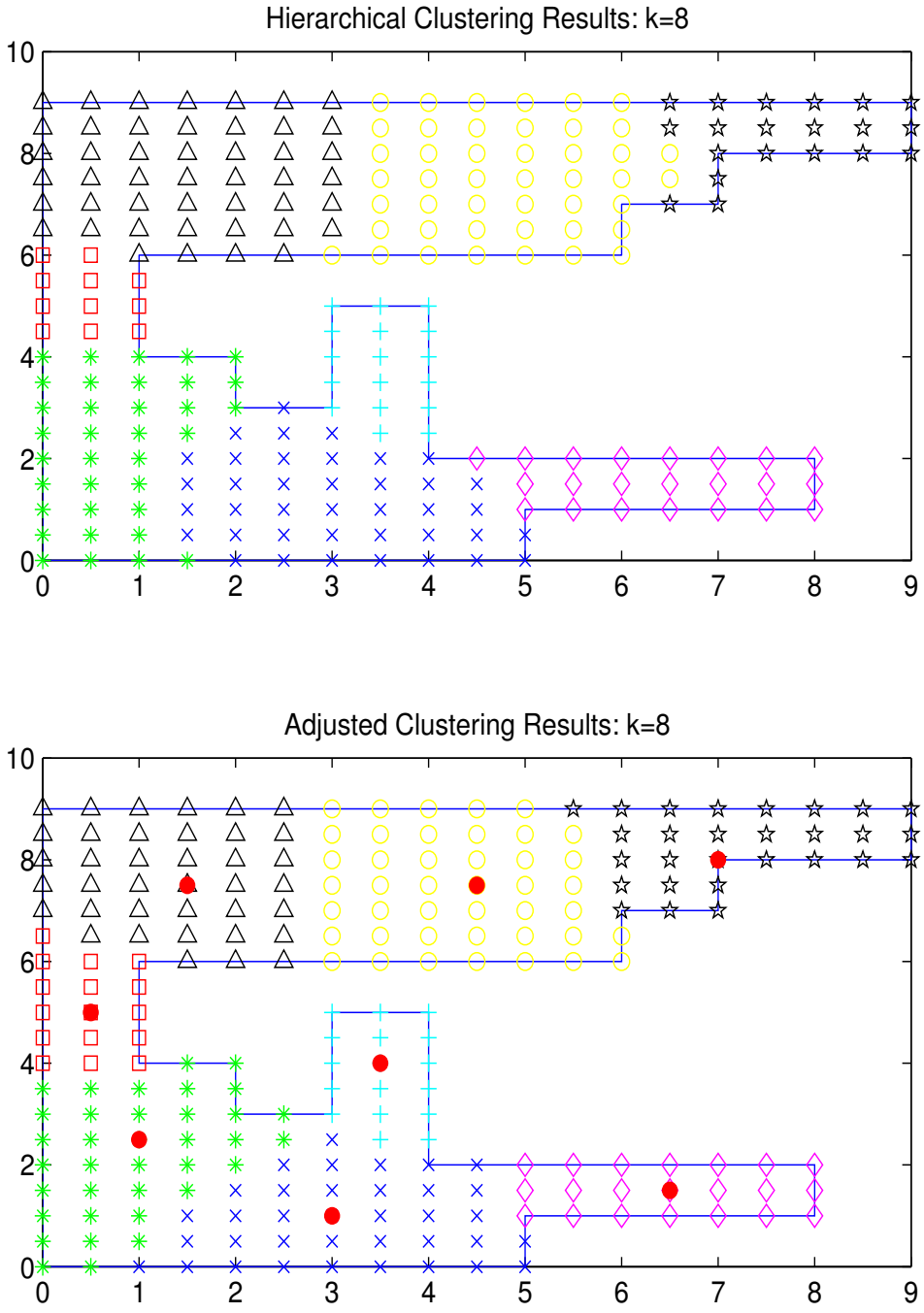

Figure 5.3: Clustering results, k=8

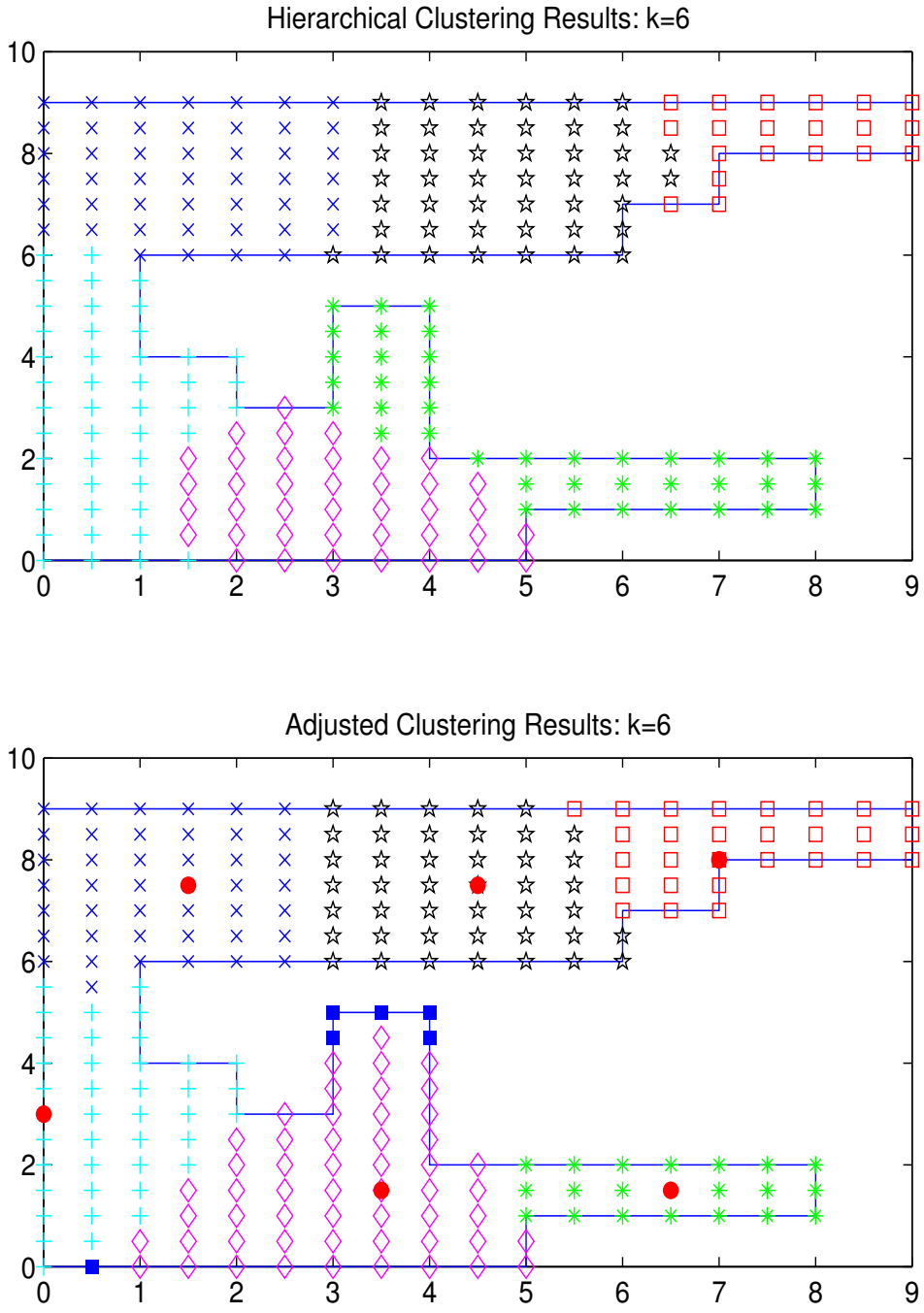

Figure 5.4: Clustering results, k=6

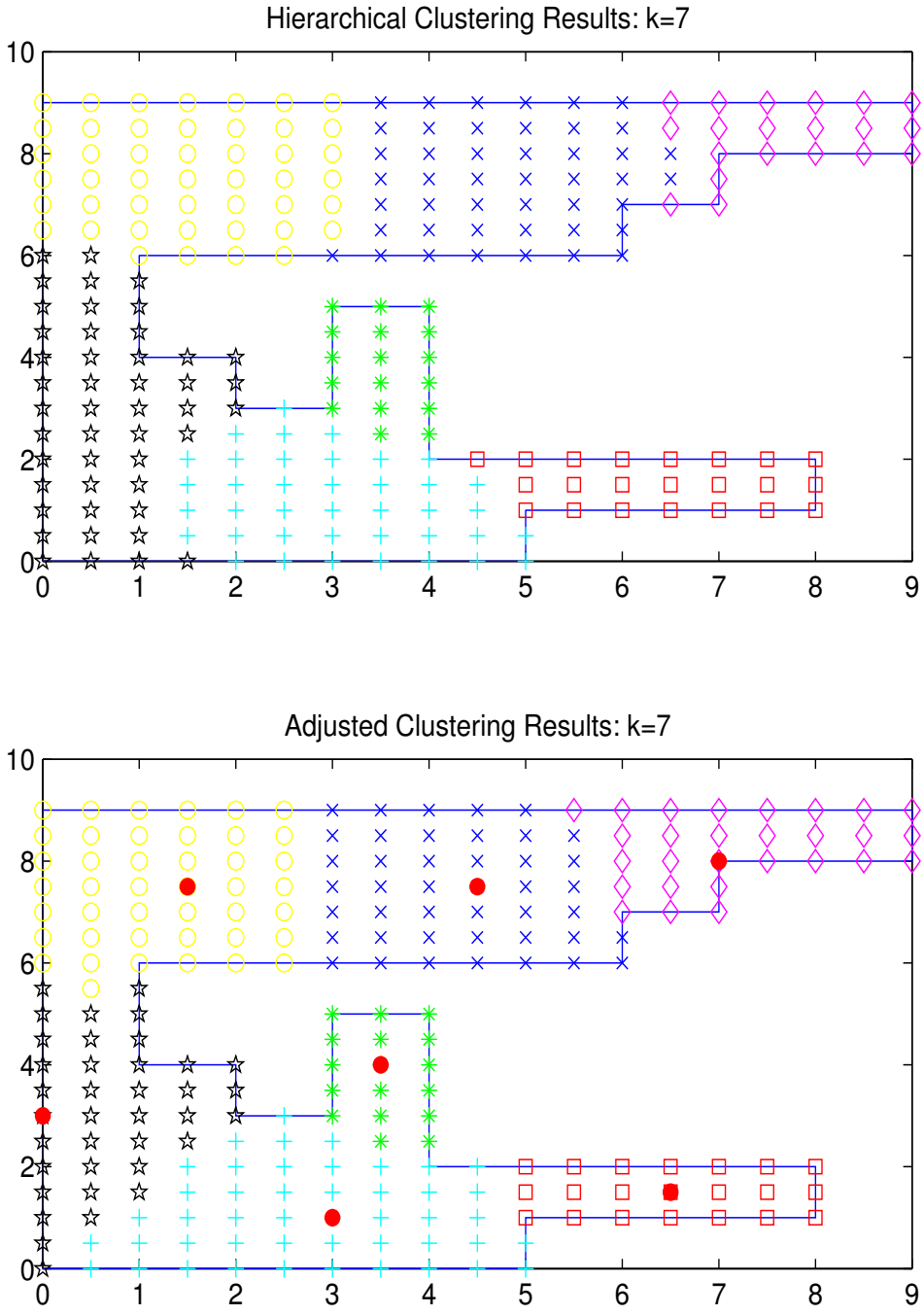

Figure 5.5: Clustering results, k=7

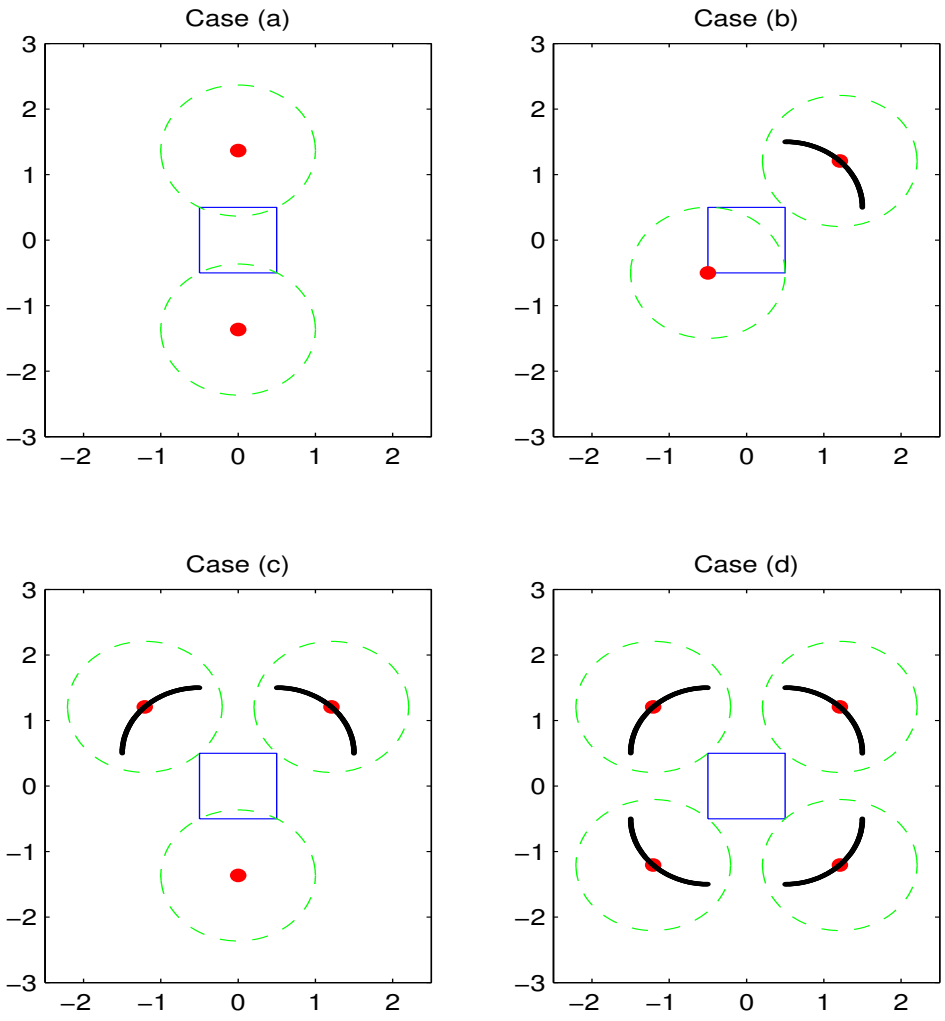

Figure 5.6: Special cases at observation points

Proof. At each the four vertices, the illuminance

$$\frac{I_0}{1+R^2} \geq \delta.$$

For an arbitrary point p inside the pixel, the illuminance

$$\sum_i \frac{I_0}{1+d_{pg_i}^2} \geq \frac{I_0}{1+R^2} \geq \delta,$$

where g_i represents the i-th light fixture.

Therefore, when the illumination intensities at all observation points satisfy the requirement, the whole polygon is sufficiently illuminated. □

5.4.2 Influence of the Effective Illumination Radius

In Figures 5.3, 5.4, 5.5, we set the effective illumination radius $R = 3$. If we relax the illumination requirement, the light fixtures will have a larger effective illumination radius. When we set $R = 4$, the results are shown in Figure 5.7. In this case, the whole polygon can be illuminated by 6 light fixtures.

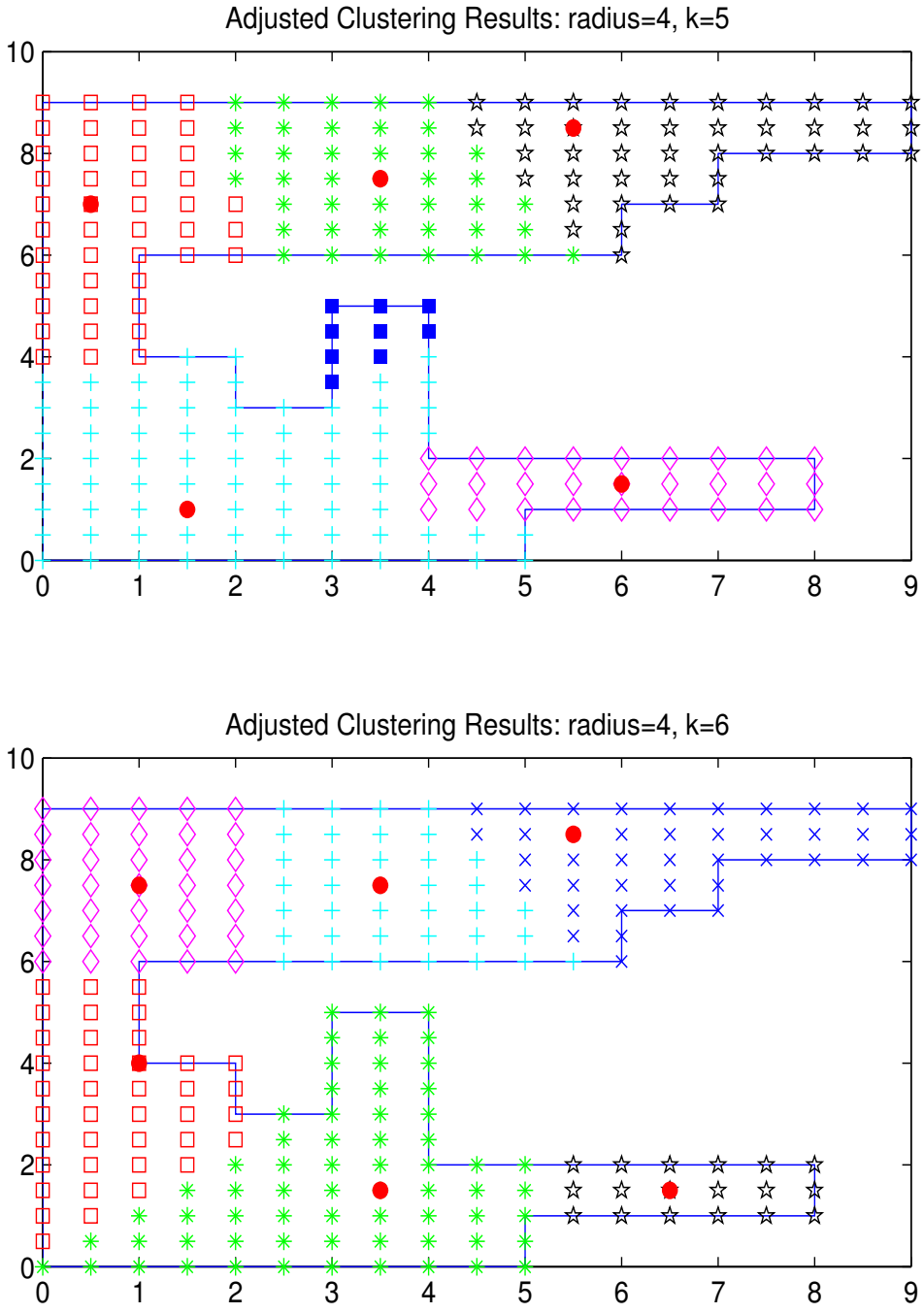

Figure 5.7: Clustering results, radius = 4

Chapter 6

Conclusions and Future Work

6.1 Summary

In this thesis, we formulated the Constrained Light Deployment Problem (CLDP), which is a new variation of the classical Art Gallery Problem. The objective of the CLDP problem is to minimize the number of necessary light fixtures for a given commercial building whose floor plan is an orthogonal polygon and the illuminance satisfies a given input requirement. The light fixtures are point light sources and the illuminance decreases with increase of distance. We proposed a heuristic algorithm based on polygon partitioning and an iterative set redundancy removal technique to solve the SOP-L problem in Chapter 3 and the OPH-L problem in Chapter 4, respectively. The solution of the SOP-L/OPH-L problem provides an upper bound for the CLDP problem. Finally, in Chapter 5, we solved the CLDP problem by a binary search strategy combined with a clustering algorithm to determine a minimal set of light fixtures that meets the requirement. The algorithm also provides the locations for those light fixtures, which are not necessarily the vertices of the orthogonal polygon.

6.2 Future Work

In this thesis, we have made several assumptions for the problem.

Firstly, we suppose the light fixtures in a commercial building are of the same specification. However, the problem can be extended by considering that the light fixtures with different power consumption are used. A light fixture consumes less

power will provide less illuminance. The new objective for future research is to minimize the total power consumption while the aggregated illuminance at any spot in the building is satisfied.

Another assumption is that the illuminance requirement in the whole building is the same. The avenue for future research is that in commercial buildings, different areas may have different illumination requirements. For example, the illumination requirement at a desk is higher than in an aisle. We can assign different weights to the observation points. This is a more challenging problem for future research.

Bibliography

[1] Louigi Addario-Berry, Omid Amini, Jean-Sébastien Sereni, and Stéphan Thomassé. Guarding art galleries: The extra cost for sculptures is linear. In *Algorithm Theory–SWAT 2008*, pages 41–52. Springer, 2008.

[2] Yoav Amit, Joseph SB Mitchell, and Eli Packer. Locating guards for visibility coverage of polygons. *International Journal of Computational Geometry & Applications*, 20(05):601–630, 2010.

[3] Kohei Arai and Ali Ridho Barakbah. Hierarchical k-means: an algorithm for centroids initialization for k-means. *Reports of the Faculty of Science and Engineering*, 36(1):25–31, 2007.

[4] Antonio Leslie Bajuelos Domínguez, Sergey Bereg, and Ana Mafalda Martins. Guarding orthogonal galleries with rectangular rooms. *The Computer Journal*, page bxt089, 2013.

[5] Antonio Leslie Bajuelos Domínguez, Gregorio Hernández Peñalver, Santiago Canales Cano, and Ana Mafalda Martins. Minimum vertex guard problem for orthogonal polygons: a genetic approach. 2008.

[6] Antonio Leslie Bajuelos Domínguez, Ana Mafalda Martins, Santiago Canales Cano, and Gregorio Hernández Peñalver. Metaheuristic approaches for the minimum vertex guard problem. In *Advanced Engineering Computing and Applications in Sciences, 2009. ADVCOMP'09. Third International Conference on*, pages 77–82. IEEE, 2009.

[7] Tobias Baumgartner, Sándor P Fekete, Alexander Kröller, and Christiane Schmidt. Exact solutions and bounds for general art gallery problems. *Science*, 407:14–1, 2011.

[8] Iliana Bjorling-Sachs and Diane L Souvaine. An efficient algorithm for guard placement in polygons with holes. *Discrete & Computational Geometry*, 13(1):77–109, 1995.

[9] Andrea Bottino and Aldo Laurentini. A nearly optimal algorithm for covering the interior of an art gallery. *Pattern Recognition*, 44(5):1048–1056, 2011.

[10] Vašek Chvátal. A combinatorial theorem in plane geometry. *Journal of Combinatorial Theory, Series B*, 18(1):39–41, 1975.

[11] Marcelo C Couto, Pedro J de Rezende, and Cid C de Souza. An exact algorithm for minimizing vertex guards on art galleries. *International Transactions in Operational Research*, 18(4):425–448, 2011.

[12] Marcelo C Couto, Cid C De Souza, and Pedro J De Rezende. Experimental evaluation of an exact algorithm for the orthogonal art gallery problem. In *Experimental Algorithms*, pages 101–113. Springer, 2008.

[13] Marcelo C Couto, Cid C de Souza, and Pedro Jussieu de Rezende. An exact and efficient algorithm for the orthogonal art gallery problem. In *Computer Graphics and Image Processing, 2007. SIBGRAPI 2007. XX Brazilian Symposium on*, pages 87–94. IEEE, 2007.

[14] Ajay Deshpande, Taejung Kim, Erik D Demaine, and Sanjay E Sarma. *A Pseudopolynomial Time O (logn)-Approximation Algorithm for Art Gallery Problems*. Springer, 2007.

[15] Herbert Edelsbrunner, Joseph O'Rourke, and Emmerich Welzl. Stationing guards in rectilinear art galleries. *Computer Vision, Graphics, and Image Processing*, 27(2):167–176, 1984.

[16] Schneider Electric. Leading techniques for energy savings in commercial office buildings. `http://www2.schneider-electric.com/documents/buildings/lleading_techniques_for_energy_savings_in_colleges_and_universities.pdf`, 2009.

[17] Steve Fisk. A short proof of chvátal's watchman theorem. *Journal of Combinatorial Theory, Series B*, 24(3):374, 1978.

[18] Deborah S Franzblau and Daniel J Kleitman. An algorithm for constructing regions with rectangles: Independence and minimum generating sets for collections of intervals. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pages 167–174. ACM, 1984.

[19] Subir Kumar Ghosh. Approximation algorithms for art gallery problems in polygons. *Discrete Applied Mathematics*, 158(6):718–722, 2010.

[20] Petr Hliněnỳ. An addition to art galleries with interior walls. *Discrete & Computational Geometry*, 25(2):311–314, 2001.

[21] Frank Hoffmann, Michael Kaufmann, and Klaus Kriegel. The art gallery theorem for polygons with holes. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*, pages 39–48. IEEE, 1991.

[22] Frank Hoffmann and Klaus Kriegel. A graph-coloring result and its consequences for polygon-guarding problems. *SIAM Journal on Discrete Mathematics*, 9(2):210–224, 1996.

[23] Joan Hutchinson and André Kündgen. Orthogonal art galleries with interior walls. *Discrete Applied Mathematics*, 154(11):1563–1569, 2006.

[24] Justin Iwerks and Joseph SB Mitchell. The art gallery theorem for simple polygons in terms of the number of reflex and convex vertices. *Information Processing Letters*, 2012.

[25] Dae-Sung Jang and Sun-Il Kwon. Fast approximation algorithms for art gallery problems in simple polygons. *arXiv preprint arXiv:1101.1346*, 2011.

[26] Jeff Kahn, M Klawe, and Daniel Kleitman. Traditional galleries require fewer watchmen. *SIAM Journal on Algebraic Discrete Methods*, 4(2):194–206, 1983.

[27] Matthew J Katz and Gabriel S Roisman. On guarding the vertices of rectilinear domains. *Computational Geometry*, 39(3):219–228, 2008.

[28] Leonard Kaufman and Peter J Rousseeuw. *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons, 2009.

[29] J Mark Keil. Minimally covering a horizontally convex orthogonal polygon. In *Proceedings of the 2nd Annual Symposium on Computational Geometry*, pages 43–51. ACM, 1986.

[30] James King. Fast vertex guarding for polygons with and without holes. *Computational Geometry*, 46(3):219–231, 2013.

[31] James King and David Kirkpatrick. Improved approximation for guarding simple galleries from the perimeter. *Discrete & Computational Geometry*, 46(2):252–269, 2011.

[32] Erik A Krohn and Bengt J Nilsson. Approximate guarding of monotone and rectilinear polygons. *Algorithmica*, pages 1–31, 2013.

[33] Aldo Laurentini. Guarding the walls of an art gallery. *The Visual Computer*, 15(6):265–278, 1999.

[34] D Lee and Arthurk Lin. Computational complexity of art gallery problems. *Information Theory, IEEE Transactions on*, 32(2):276–282, 1986.

[35] TS Michael and Val Pinciu. Guarding orthogonal polygons with h holes: A bound independent of h. 2013.

[36] Bengt J Nilsson. Approximate guarding of monotone and rectilinear polygons. In *Automata, Languages and Programming*, pages 1362–1373. Springer, 2005.

[37] Joseph O'rourke. *Art Gallery Theorems and Algorithms*, volume 1092. Oxford University Press Oxford, 1987.

[38] Dietmar Schuchardt and Hans-Dietrich Hecker. Two np-hard art-gallery problems for ortho-polygons. *Mathematical Logic Quarterly*, 41(2):261–267, 1995.

[39] Zoran Šunić. Normal art galleries: Wall in–all in. *Computational Geometry*, 46(1):7–16, 2013.

[40] Ana Paula Tomás. Generation of orthogonal polygons. `http://www.dcc.fc.up.pt/~apt/genpoly/`, 2013.

[41] Ana Paula Tomás and António Leslie Bajuelos. Generating random orthogonal polygons. In *Current Topics in Artificial Intelligence*, pages 364–373. Springer, 2004.

[42] Ana Paula Tomás and António Leslie Bajuelos. Quadratic-time linear-space algorithms for generating orthogonal polygons with a given number of vertices.

In *Computational Science and Its Applications–ICCSA 2004*, pages 117–126. Springer, 2004.

[43] Ana Paula Tomás, António Leslie Bajuelos, and Fábio Marques. Approximation algorithms to minimum vertex cover problems on polygons and terrains. In *Computational Science ICCS 2003*, pages 869–878. Springer, 2003.

[44] Davi C Tozoni, Pedro J de Rezende, and Cid C de Souza. A practical iterative algorithm for the art gallery problem using integer linear programming. 2013.

[45] Jorge Urrutia. Art gallery and illumination problems. *Handbook of Computational Geometry*, pages 973–1027, 2000.

[46] Wikipedia. Energy-saving lighting. `https://en.wikipedia.org/wiki/Category:Energy-saving_lighting`, 2015.

[47] Wikipedia. Illuminance. `https://en.wikipedia.org/wiki/Illuminance`, 2015.

[48] Wikipedia. Inverse square law. `https://en.wikipedia.org/wiki/Inverse-square_law`, 2015.

[49] Wikipedia. Set cover problem. `https://en.wikipedia.org/wiki/Set_cover_problem`, 2015.

[50] Paweł Żyliński. Orthogonal art galleries with holes: A coloring proof of aggarwal's theorem. *The Electronic Journal of Combinatorics*, 13(1):R20, 2006.