2021

Designing Efficient Algorithms for Sensor Placement

Gabriel Loos
Georgia Southern University

DESIGNING EFFICIENT ALGORITHMS FOR SENSOR PLACEMENT

An Honors Thesis submitted in partial fulfillment of the requirements for Honors in Mathematical Science.

by

GABRIEL LOOS

Under the mentorship of Dr. Hua Wang

ABSTRACT

Sensor placement has many applications and uses that can be seen everywhere you go. These include, but not limited to, monitoring the structural health of buildings and bridges and navigating Unmanned Aerial Vehicles(UAV).

We study ways that leads to efficient algorithms that will place as few as possible sensors to cover an entire area. We will tackle the problem from both 2-dimensional and 3-dimensional points of view. Two famous related problems are discussed: the art gallery problem and the terrain guarding problem. From the top view an area presents a 2-D image which will enable us to partition polygonal shapes and use graph theoretical results in coloring. We explore this approach in details and discuss potential generalizations. We will also look at the area from a side view and use methods from the terrain guarding problem to determine where any more sensors should be placed. We provide a simple greedy algorithm for this.

Lastly, we briefly discuss the combination of the above techniques and potential further generalizations to suit specific problems where the limitation of sensors (such as range and angle) are taken into consideration.

Thesis Mentor: ______________________________

Dr. Hua Wang

Honors Director: ______________________________

Dr. Steven Engel

May 2021
Mathematical Science
University Honors Program
Georgia Southern University

ACKNOWLEDGMENTS

I wish to acknowledge my family, friends and outstanding professors at Georgia Southern. I would also like to give a special thanks to Dr. Hua Wang for everything he has done for me.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

In this chapter we present some background information on sensor placement as well as the mathematical tools that we will use.

1.1 SENSOR PLACEMENT

Sensor placement has many applications and uses that can be seen everywhere you go. Almost all public buildings use some some sort of sensors. Motion sensors are used to sense when someone enters a room and turn on the lights, there are sensors that tell air conditioning units when to turn on, and security cameras to survey a given area. Sensors are also used in the structural health monitoring of buildings and bridges. Sensors are used to measure the temperature of the structure, how much stress or strain the structure is undergoing, and how much the structure is leaning.

For an efficient deployment of sensors one has to take into account how much area is needed to be covered, the restrictions on the sensors, and then decide where to place them in order to use the sensors most efficiently. Knowing how much area needs to be covered is extremely important as placing sensors that do not cover the entire area is obviously not preferred. It is also inefficient to place two sensors that cover an area that can be covered by just placing one sensor. Sometimes the restrictions of the sensors need to also be taken into consideration.

1.2 BASIC GRAPH THEORY

When studying sensor placement some concepts from graph theory will be utilized. Of course we can not talk about graph theory without first talking about what a *graph* is, hence our first definition [8]:

Definition 1. A graph G consists of two sets:

- V, whose elements are referred to as the vertices of G (the singular of vertices is vertex); and
- E, whose elements are unordered pairs from V (i.e., $E \subseteq \{\{v_1, v_2\}|v_1, v_2 \in V\}$). The elements of E are referred to as the edges of G.

Now that we know what a graph is we can define some useful concepts that will help in our study. One of the tools from graph theory that will be particularly helpful to us is *graph coloring*.

Definition 2. A graph coloring is when every vertex is assigned a color. an example can be seen in Figure 1.1.

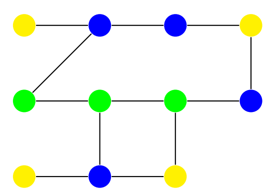


Figure 1.1: Example of a graph coloring.

More specifically, what we will be using in our algorithm is called a *proper coloring* of a graph.

Definition 3. A proper coloring is a special graph coloring when no vertex is connected to another vertex of the same color. A (proper) k-coloring is when k colors are used to color the graph. We can see an example of this in Figure 1.2.

The majority of the structures that we will be dealing with lead to geometric structures that are polygons or polygonal shapes. One of the most common way of partitioning a polygonal shape is *triangulation*.

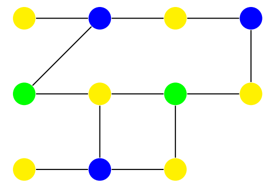


Figure 1.2: Example of a (proper) 3-coloring.

Definition 4. Triangulation is when you connect the vertices of a polygon in order to make triangles. An example of this can be seen in Figure 1.3

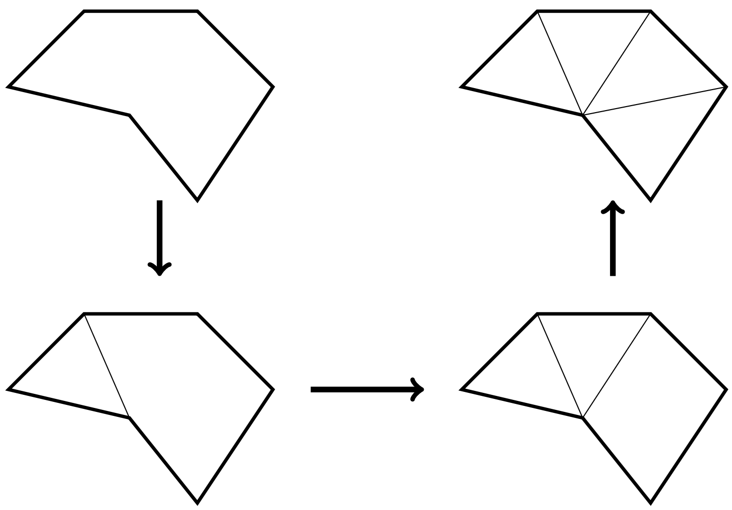


Figure 1.3: Example of the triangulation of a polygonal shape.

In graph theory there are special types of graphs that are frequently used in research. We first define the *paths* as follows.

Definition 5. A path is a sequence of vertices that are connected by edges of which no vertices are repeated. An example of this can be seen in Figure 1.4

In our study most paths will be x*-monotone*, as defined below. note that such characteristics refer to the geometric nature of the structures and differ from the traditional graph theory concepts.

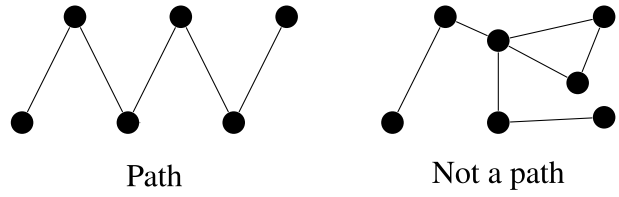


Figure 1.4: Examples of path and non-path.

Definition 6. A path is x-monotone if you can move a straight vertical line across the path and it never intersects the path more then once. We can see an example in Figure 1.5

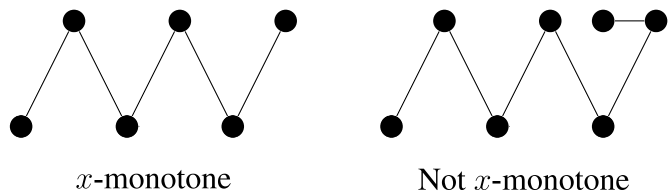


Figure 1.5: An example of what is and is not x-monotone.

1.3 THE ART GALLERY PROBLEM

The art gallery problem was first proposed in 1973 by Victor Klee [4]. The problem asked that if you had an art gallery consisting of n vertices, how many guards would be needed to see the entirety of the art gallery? Since then there have been many renditions of the art gallery problem including: limiting the guards to just the vertices, allowing the guards to move along a set route, guarding an orthogonal gallery, guarding a gallery with holes, etc.

In this thesis we will be presenting an approach to solve the problem for an orthogonal art gallery with holes, which we will talk about in the next a few chapters. Before it would be beneficial to mention the original problem and its solution. The original problem again asked, how many guards are needed to guard a n-vertex polygon.

The number of guards needed to guard a n-vertex polygon is $\frac{n}{3}$. There are two ways to show this, the first is a combinatorics proof done by Václav Chvátal in 1975 [6]. The second and easier way uses graph theory and was done by Steve Fisk in 1978 [6]. We will be using Fisk's proof as it is easier to understand and to apply.

To find how many guards are needed to guard a n-vertex polygon we will first triangulate the polygon. We know that we can triangulate any polygon using $n-3$ diagonals to make $n-2$ triangles. Once we have triangulated the polygon we are left with a graph that we can 3-color. If we look at Figure 1.6 we can see that a triangle is not only able to be 3-colored but can only be 3-colored (i.e. two colors are not enough). Once we have 3-colored one triangle we know that at least one other triangle shares an edge with it. Knowing this may not be easy to see at first so we will show it.

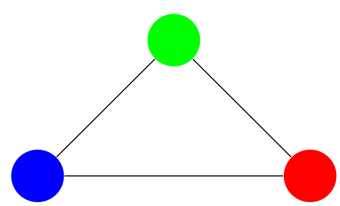


Figure 1.6: A (properly) 3-colored triangle.

We know from the definition of a polygon that you can trace around the edge of a polygon without tracing over the same spot more then once. We can see an example of this in Figure 1.7.

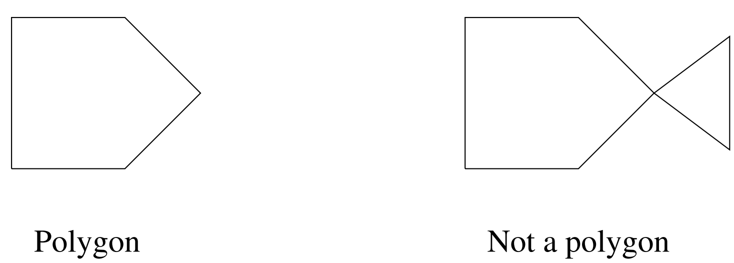


Figure 1.7: Example of what is and is not a polygon.

We also know that once we have triangulated the polygon if we trace around the the outermost edges of the graph it will have the same result as tracing around the outermost edges of the polygon. This is shown in Figure 1.8.

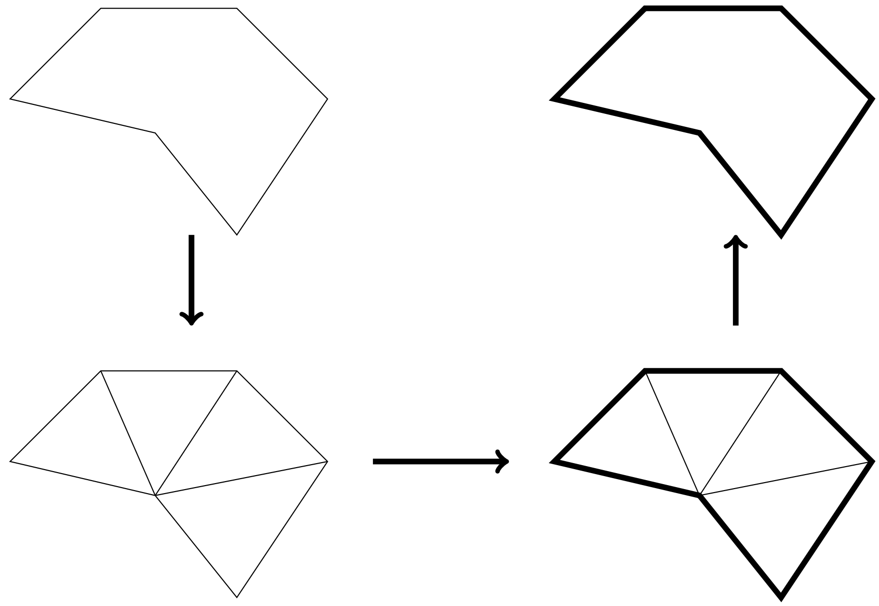


Figure 1.8: Example where the outermost edges of a triangulated polygon being the same as the original polygon.

Now we can show that once a triangle has been triangulated in the resulting graph every triangle will share at least one side with another triangle. To do so assume that after triangulating a polygon there is some triangle that does not share a side with another triangle. Then if we trace around the outermost edges of the graph, when we get to that triangle we will start at the point that connects it to the rest of the graph, then we will go around the triangle and end up at the same point that connects it to the triangle, after which we will have to continue going around the rest of the graph. A good visualization of this is in Figure 1.7 where we can see what the graph might look like. This is again not a polygon because we traced over the same point twice, a contradiction to the assumption that it was a polygon. So now we have shown that every triangle must share at least one edge with another triangle.

Next we can start by coloring one triangle. After we have colored this triangle we know we can color the triangle connected to it. This is because The triangle we have not colored shares to vertices with the triangle we have colored so those two vertices are already colored. This leaves only one other color left for the remaining vertex. This can be seen in Figure 1.9. We can repeat this process until we have colored the whole graph and we will not use any more then $\frac{n}{3}$ guards.

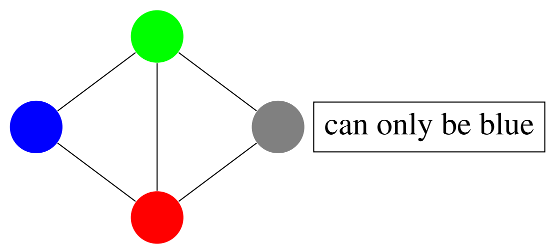


Figure 1.9: Visual of two triangles being 3-colored.

The only potential scenario of the above 3-coloring of triangles failing is when the triangles “cycle around”. This can be seen in Figure 1.10. We can also see that the only way that could happen is if the all connect at some center point. Because we are triangulating a polygon there will be no point in the polygon for the triangles to meet at. This is due to the fact that we are only connecting the vertices of the polygon to make triangles.

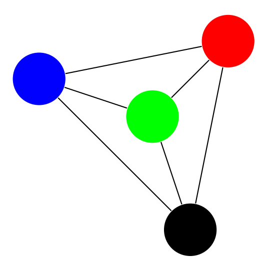


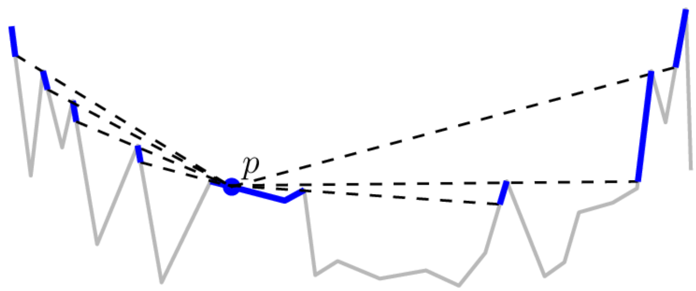
Figure 1.10: Visual of coloring a graph of cycling triangles.

In Chapter 2 we provide the detailed proof of a classic result of this nature, the notations are from [9] and the proof is from [5].

1.4 The terrain guarding problem

The terrain guarding problem, which has been extensively studied since 1995 [2], deals with a path of length n that is x-monotone. The problem asks where to place the least amount of guards so that the guards can see all other vertices on the terrain.

We say a guard can see another vertex if you can draw a straight line from the guard to the vertex without intersecting the terrain. We also assume that a guard can see the vertex that he is on. There is also a continuous version of this problem where the guards need to see every single point on the terrain. In Figure 1.11 we can see an example of where a guard can see.



(a) The visibility region $\mathcal{V}(p)$ of $p \in T$ has $\mathrm{O}(n)$ subterrains.

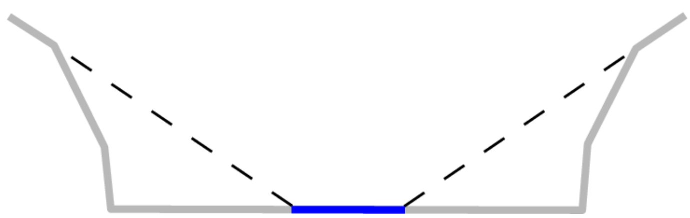


Figure 1.11: An example of where a point p can see.

For the continuous problem we will say that a guard can see an edge if he can see the two vertices that are connected by the edge. In Figure 1.12 we can see that v_1 can see v_2 and v_3 so it is able to see the edge $v_2 - v_3$, but v_1 can see v_5 but not v_4 so v_1 cannot see the edge $v_4 - v_5$.

In applications, we will first take vertical crossing sections of the area we will be placing guards around. The crossing sections will result in a 2-d image with the ground of the area making up our terrain that we want to guard. We will then apply the terrain guarding problem to place guards that will be able to see the whole terrain.

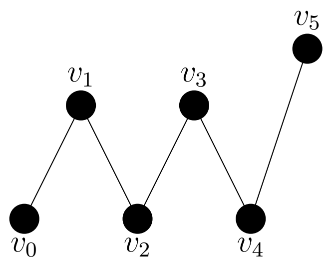


Figure 1.12: Example for what edges a vertex can see.

1.5 OTHER RELATED INFORMATION

The most relevant research towards efficient solution to our problem, is the study of partitioning of geometric shapes, coloring of the partitioned graphs, and various terrain guarding problems. In Table 1.1 below we list some of the algorithmic works we have found.

Table 1.1: Summary of related work

References	Findings
[6]	Lefthandside method of solving terrain guarding problem; Triangulation method of solving art gallery problem; $\frac{n}{3}$ guards are always sufficient for art gallery problem.
[2]	Annotated, Discrete, and Continuous terrain guarding problems can all be solved in $n^{(O(\sqrt{k}))}$ time where n is the number of vertices and $k \leq n$.
[4]	$\frac{n}{3}$ guards is always sufficient to cover a n-vertex polygon; Only one guard is needed for convex polygons, star shaped polygons, 4-sided polygons, and 5-sided polygons; Two guards needed for 6-sided polygons.
[10]	For *monotone* polygons (if one can move a straight line through the polygon without touching more than 2 sides.); Triangulation of a polygon with time complexity $O(n\log(n))$ for simple and $O(n)$ for complex structures.
[7]	A method is for triangulating a monotone polygon.
[3]	$O(n^{(3/14)})$ time complexity for three coloring a 3-colorable graph.

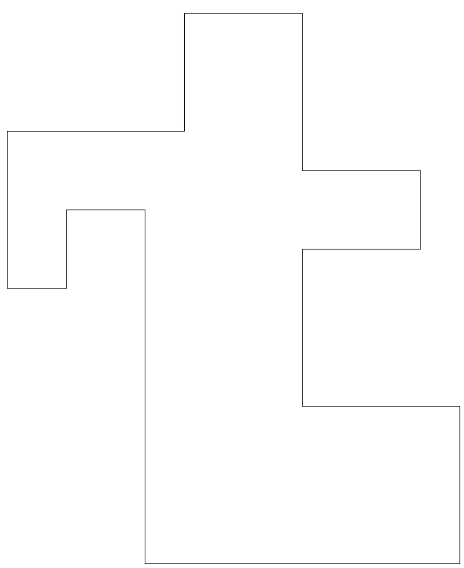
CHAPTER 2

ORTHOGONAL GALLERIES

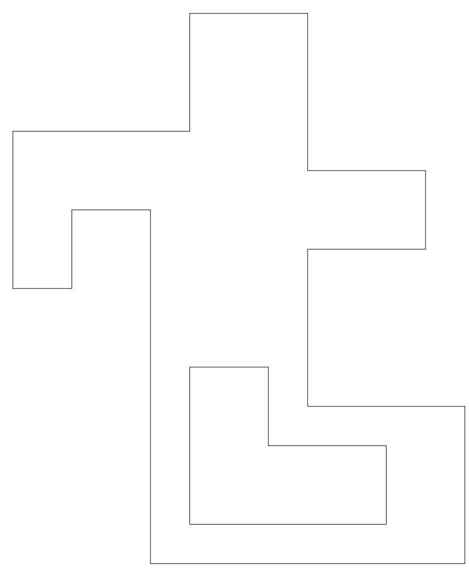
A common kind of geometric structures are bounded by mutually perpendicular sides, called *orthogonal* shapes. The question of guarding orthogonal art galleries will be studied in details in the upcoming chapters. Here, in preparation for the study of orthogonal gallery with "holes", we extensively discuss the orthogonal shapes and *quadrilaterizations* of such shapes.

Definition 7. An orthogonal gallery is a gallery where all of the edges make either 90 degree or 270 degree angles as seen in Figure 2.1a.

An orthogonal gallery with holes will satisfy the same conditions for both the edges of the polygon and the edges of the holes as seen in Figure 2.1b.



(a)



(b)

Figure 2.1: (a) An orthogonal gallery; (b) An orthogonal gallery with one hole.

2.1 ORTHOGONAL GALLERIES AND QUADRILATERALIZATIONS

First let us consider an orthogonal gallery without holes. It is shown in [5] that $\frac{n}{4}$ guards is always sufficient. This proof is done by showing that if you have a orthogonal polygon P and you can quadrilateralize every orthogonal polygon smaller then it then P can be quadrilateralized. In this thesis we will be following the same format as the proof in [9]. The argument we provide here follow the same reasoning as [5]. We supply it with much more details and sometimes slightly different explanation.

We will start with examining the quadrilateralization of our art gallery.

Definition 8. The quadrilateralization of a polygon is made by connecting the vertices of the polygon that result in making convex quadrilaterals. An example can be seen in Figure 2.2

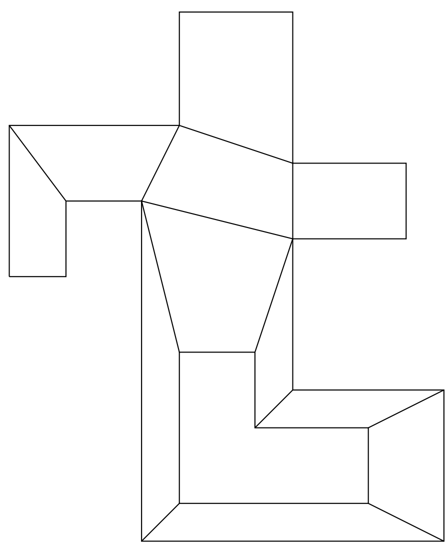
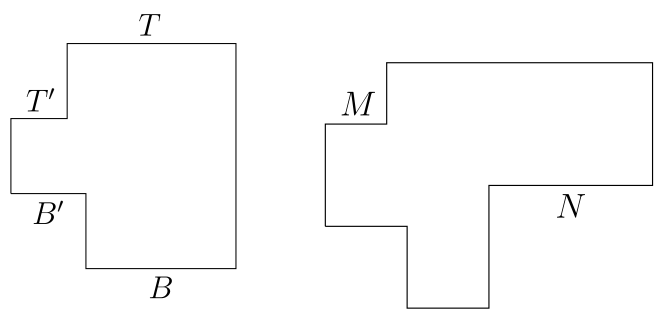


Figure 2.2: The quadrilateralization of an orthogonal gallery with one hole.

Before we get into the proof we should define a couple more terms. In an orthogonal polygon, a *top edge* is a horizontal edge where the interior of the polygon is below the edge. A *bottom edge* is a horizontal edge where the interior of the polygon is above the edge. *Left and right edges* are defined in the same way.

A top edge T and bottom edge B can see each other if there exists a point on t on T that can see a point b on B. T and B are *neighbors* if they can see each other and there is no bottom edge higher then B that can see T and no top edge lower than T that can see B. A *tab* is a pair of neighboring edges that are connected by a vertical edge. We can see some of these definitions illustrated in Figure 2.3a. We can see that T and B can see each other but they are not neighbors as T' is lower then T and can see B. We can also see that T' and B' can are neighbors and since they are connected by a vertical edge they form a tab. In Figure 2.3b we can see M and N are neighboring edges that do not form a tab.



(a) (b)

Figure 2.3: Orthogonal shape, neighboring edges, and tabs.

2.2 Some geometric observations

To prove that every orthogonal polygon can be quadrilateralized we will first show that if a orthogonal polygon has one of three structures, we can split it into two smaller orthogonal polygons that can be quadrilateralized, and then put the polygons back together to make a quadrilateralization of the original polygon. We call the splitting of the polygon a *reduction*. We will later show that every polygon contains one of the three structures.

Before we show that every polygon containing on of the three structures can quadrilateralized we first must introduce some geometric facts. The first deals with an orthogonal polygon who's edges are *in general position*. This means that the none of the horizontal edges of the polygon have the same vertical coordinate and none of the vertical edges have the same horizontal coordinate.

Lemma 2.2.1 ([5]). *An orthogonal polygon P that is not in general position has the same quadrilateralization as any "nearby" P' that is in general position.*

Proof. Consider a sequence of orthogonal regions P', that are all in general position, with the same number of edges as P. We can assume, without loss of generality, that this sequence converges to P. In other words, repeatedly making small changes to P' can result in the same structure as P. Then near the end of the convergence, P' will be extremely close to having the same structure as P, with the only difference being that it is in general position. Then because there are only finitely many ways to quadrilateralize this region P' must have the same quadrilateralization as P. □

Let $a = (a_x, a_y)$ and $b = (b_x, b_y)$ be two points on the x, y coordinate plane. We define a new point $a\#b = (a_x, b_y)$. Consequently, $\square(a, b)$ is the box made by the points $a, b, a\#b, b\#a$ [5]. Sometimes we also use the same notation $L\#T = (a, b)$ for a vertical line segment with x-coordinate a, and a horizontal line segment with y-coordinate b.

Lemma 2.2.2 ([5]). *Let T and B be neighboring top and bottom edges of an orthogonal polygon P. Then there is a left edge L left of both T and B, whose top endpoint is at least as high as T and whose bottom endpoint is at least as low as B, and a right edge R with analogous properties, such that $\square(L\#B, R\#T)$ lies completely inside of P.*

Proof. First we will show that if T and B are neighbors then there is a left edge, L, left of both T and B, and a right edge, R, right of both T and B, that "extends" both at least as high as T and at least as low as B. We will do so by contradiction.

So assume that T and B are neighbors and there is no left and right edge with such properties. Without loss of generality, we may assume that there is no left edge with such properties, then pick some left edge whose bottom end point can see B and T. If there is no such edge then pick some left edge whose top end point can see B and T. Note that at least one of the top and bottom end point of the edge adjacent to T or B can see both of them (under the assumption that no left edge is both high and low enough).

We may now assume, without loss of generality, that there is some left edge, L, whose bottom end point can see B and T (Figure 2.4). Then we also know that there must be some horizontal edge connected to L and that is a top edge, T' that is lower then T. We can see that the bottom endpoint of L is the same as one of the endpoints of T'. This means that one of the of the endpoint of T' can see B, contradicting the assumption that T and B are neighbors.

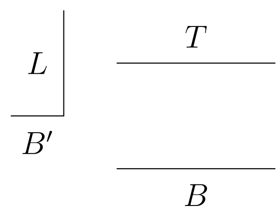


Figure 2.4: The neighboring edges T, B and the left edge.

Therefore, if T and B are neighbors there is is a left edge, L, left of both T and B, and a right edge, R, right of both T and B, that are both at least as high as T and at least as low as B.

Next we will show that the rectangle $\square(L\#B, R\#T)$ is completely inside of P. Consider some top edge T and some bottom edge B that are neighbors. Let L be an edge left of both T and B whose top endpoint is at least as high as T and whose bottom endpoint is at least as low as B and R be a edge right of both T and B with the same properties.

For contradiction assume that $\square(L\#B, R\#T)$ reaches outside of P. Then there would have to be some edge sequence that intersects $\square(L\#B, R\#T)$ as seen in Figure 2.5.

Consequently there would then be a bottom edge B' higher then B that could see all of T or a top edge lower than T that could see all of B. Either way this leads to a contradiction to T and B being neighbors. $\square$

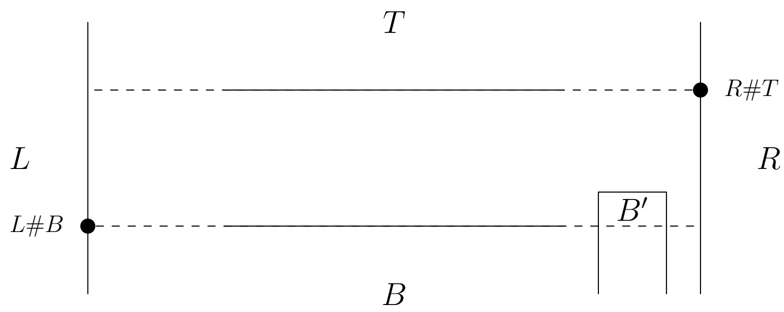


Figure 2.5: A $\square(L\#B, R\#T)$ that is not within P.

We may now assert the following for a tab, that it mst be "chopped off" in our "reduction" process.

Lemma 2.2.3. *If ab and cd are the horizontal edges of a tab, then any quadrilaterization must include the quadrilateral $abcd$.*

Proof. Without loss of generality, assume that a is left of b and c is left of d. Hence ac is the connecting edge of the tab. For contradiction assume that there is some quadrilateralization that does not include $abdc$. Because of Lemma 2.2.2, a can only be connected to another point, e, that is below cd.

Note that both c and d must be in the same quadrilateral as a and e. If not, either one of them is "cut off" from the rest of the shape, or they make a non-convex quadrilateral with other vertices. We would then have to connect d to e to make the quadrilateral $aedc$. This can be seen in Figure 2.6

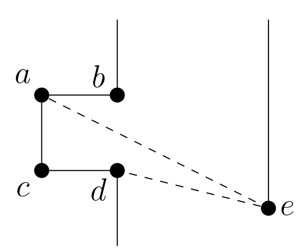


Figure 2.6: The tab ab, cd and a failed quadrilateralization.

It is now obvious that $acde$ is not convex (as cd is horizontal and de has negative slope), a contradiction. This means that the only way we can quadrilateralize tab ab, cd is by connecting bd to make the quadrilateral $abdc$. $\square$

CHAPTER 3

“REDUCTION” OF ORTHOGONAL POLYGONS

Our main idea is to convert the polygons into “smaller” ones that can be quadrilateralized. With the basic observations from the previous chapter, we can now get into how we can “reduce” orthogonal polygons.

A polygon P_1 is *smaller* then polygon P_2 if P_1 has either fewer holes then P_2 or P_1 the same amount of holes as P_2 and fewer vertices then P_2.

By reducing the polygon P to P' and showing that P' is both smaller then P and can be quadrilateralized we will show that P can also be quadrilateralized. We will discuss several different cases in the rest of this chapter.

3.1 Neighboring edges that do not form a tab

The first structure that allows us to reduce an orthogonal polygon is if the polygon contains a pair of neighboring edges that do not form a tab.

Lemma 3.1.1. *If P has a pair of neighboring edges that do not form a tab, then P is reducible.*

Proof. Let P be an orthogonal polygon with the following properties, all visualized in Figure 3.1a. Let ab be a top edge, and cd be the neighboring bottom edge. Assume ab cd is not a tab. Without loss of generality, further assume that a is left of b and c is left of d. We can then define $b' = d\#b$ and $c' = a\#c$.

By Lemma 2.2.2 we can see that $\square(a, d) = \square(b', c')$ lies completely inside of P. Assuming that P has no holes, we can now split P into P_1 and P_2.

We will split P using $\square(a, d)$ so that P_1 will include the tab $ab'dc$ and P_2 will include the tab $bac'd$, as seen in Figure 3.1b and Figure 3.1c. Both P_1 and P_2 are smaller then P because they both have the same amount of holes but fewer vertices.

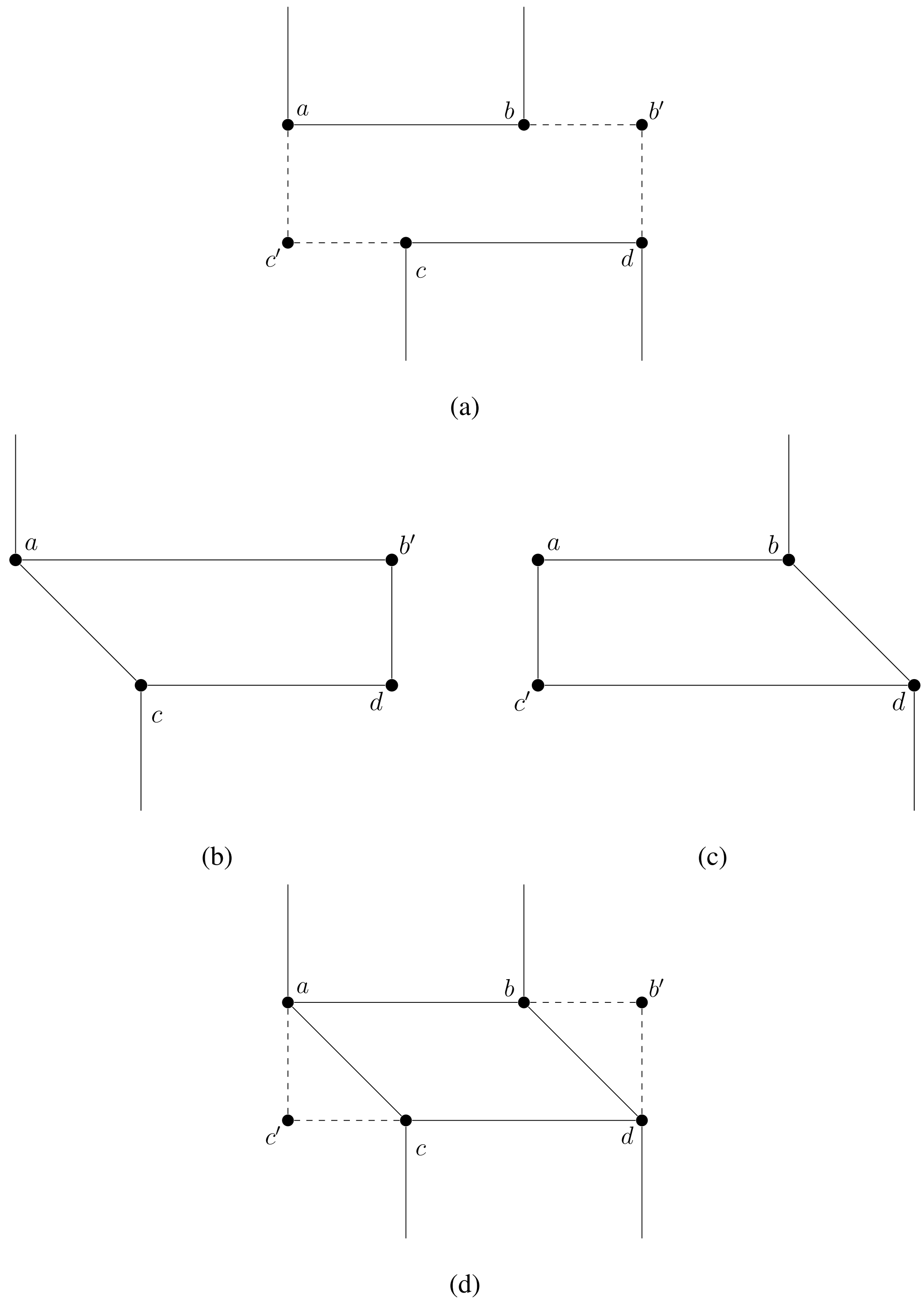


Figure 3.1: An illustration of quadrilateralizing reducible polygons.

With the assumption that every orthogonal polygon smaller than P can be quadrilateralized, both P_1 and P_2 can be quadrilateralized. By Lemma 2.2.3 we can see that the quadrilaterization of P_1 will include the quadrilateral $ab'dc$ and the quadrilaterization of P_2 will include the quadrilateral $bac'd$. We can then put quadrilateralizations of P_1 and P_2 back together to make the quadrilaterization of P.

This is illustrated by joining quadrilateral $ab'dc$ from P_1 and the quadrilateral $bac'd$ from P_2 to make the quadrilateral $abdc$, as seen in Figure 3.1d. $\square$

3.2 Good tabs

The second structure that will allow us to reduce an orthogonal polygon is described according to different types of tabs. A tab is an *up tab* if its bottom edge extends farther then its top edge and a *down tab* if its top edge extends farther then its bottom. Through Lemma 2.2.2 we know that there are two bounding vertical edges. One of the vertical edges will be the edge connecting the neighboring edges, the other is called the *facing edge*. For an up tab the top point of facing edge is called a *step point* and the connecting edge is called the *step edge*, seen in Figure 3.2. A down tab's step point is the bottom point of the facing edge and the edge connected is the step edge.

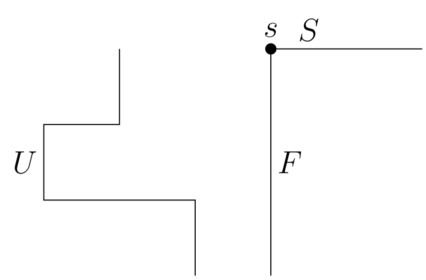


Figure 3.2: An up tab, U, and its facing edge, F, step point, s, and step edge, S.

Let ab be the top edge of an up tab, cd be the bottom edge, and s be the step point. An up tab is a *bad up tab* if the step edge is a bottom edge and $\square(b, s)$ is empty. Naturally a up tab is a *good up tab* if it is not bad (i.e. if the step edge is a top edge or if $\square(b, s)$ is not empty).

Down tabs are defined to be good or bad in the same manner. Let ab be the bottom edge of a down tab, cd be the top edge, and s be the step point. A down tab is bad if the step edge is a top edge and $\square(b, s)$ is empty. Naturally a down tab is good if it is not bad, (i.e. if the step edge is a bottom edge or if $\square(b, s)$ is not empty).

Lemma 3.2.1. *If P has a good tab, then P is reducible.*

Proof. Suppose that P contains a good up tab, as visualized in Figure 3.3, such that ab is the top edge, cd is the bottom edge, e is connected to b, and s is the step point.

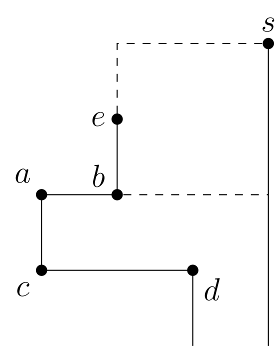


Figure 3.3: A good tab.

Because it is a good tab either the step edge is a top edge or if $\square(b, s)$ is not empty. If the step edge is a top edge let xy be that edge, if $\square(b, s)$ is not empty let xy be the lowest edge that intersects $\square(b, s)$. In both cases let x be to the left of y. This leads us to different cases, if x is to the left of b, in which case x must be higher then e, and if x is to the right of b, in which x can be either higher or lower then e. In all three scenarios y may be the same as s. All three scenarios can be seen in Figure 3.4.

Case (1) x is to the left of b, seen in Figure 3.5a. First we will replace xy and the chain $ebacd$ with two tabs. The first tab will be the chain made from y, $b\#y$, $b\#c$, d and the second

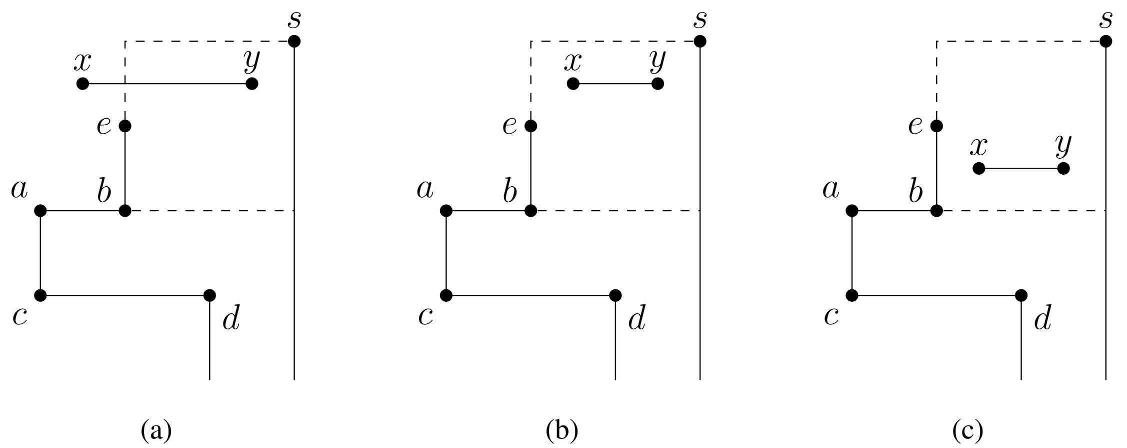


Figure 3.4: The good tab and the step edge.

will be the chain made from $x, y, y\#b, b, e$. Assume that P has no holes, then when we replace xy and the chain $ebacd$ with the two tabs we will get two new polygons P_1 and P_2. Without loss of generality, let P_1 contain $y, b\#y, b\#c, d$ and P_2 contain $x, y, y\#b, b, e$, seen in Figure 3.5c and Figure 3.5b. Because both P_1 and P_2 will have the same amount of holes as P and less vertices they are both smaller then P. Because every orthogonal polygon smaller P can be quadrilateralized, both P_1 and P_2 can be quadrilateralized. By Lemma 2.2.3, the quadrilaterization of P_1 will include the quadrilateral $y, b\#y, b\#c, d$ and the quadrilaterization of P_2 will include the quadrilateral $y, y\#b, b, e$. We can the put the quadrilateralizations of P_1 and P_2 back together to make the the quadrilaterization of P and we will see that the quadrilateral $y, b\#y, b\#c, d$ and the quadrilateral $x, y, y\#b, b, e$ will sequence $yebacdy$, seen in Figure 3.5d. and the only way this can be quadrilateralized is by making the quadrilaterals $yebd$ and $abdc$, seen in Figure 3.5e.

Case (2) x is to the right of b, seen in Figure 3.6a and Figure 3.7a. The same reduction will be able to be applied to both these scenarios. We will start by making the same replacements as in Case (1), replace xy and the chain $ebacd$ with two tabs. The first tab

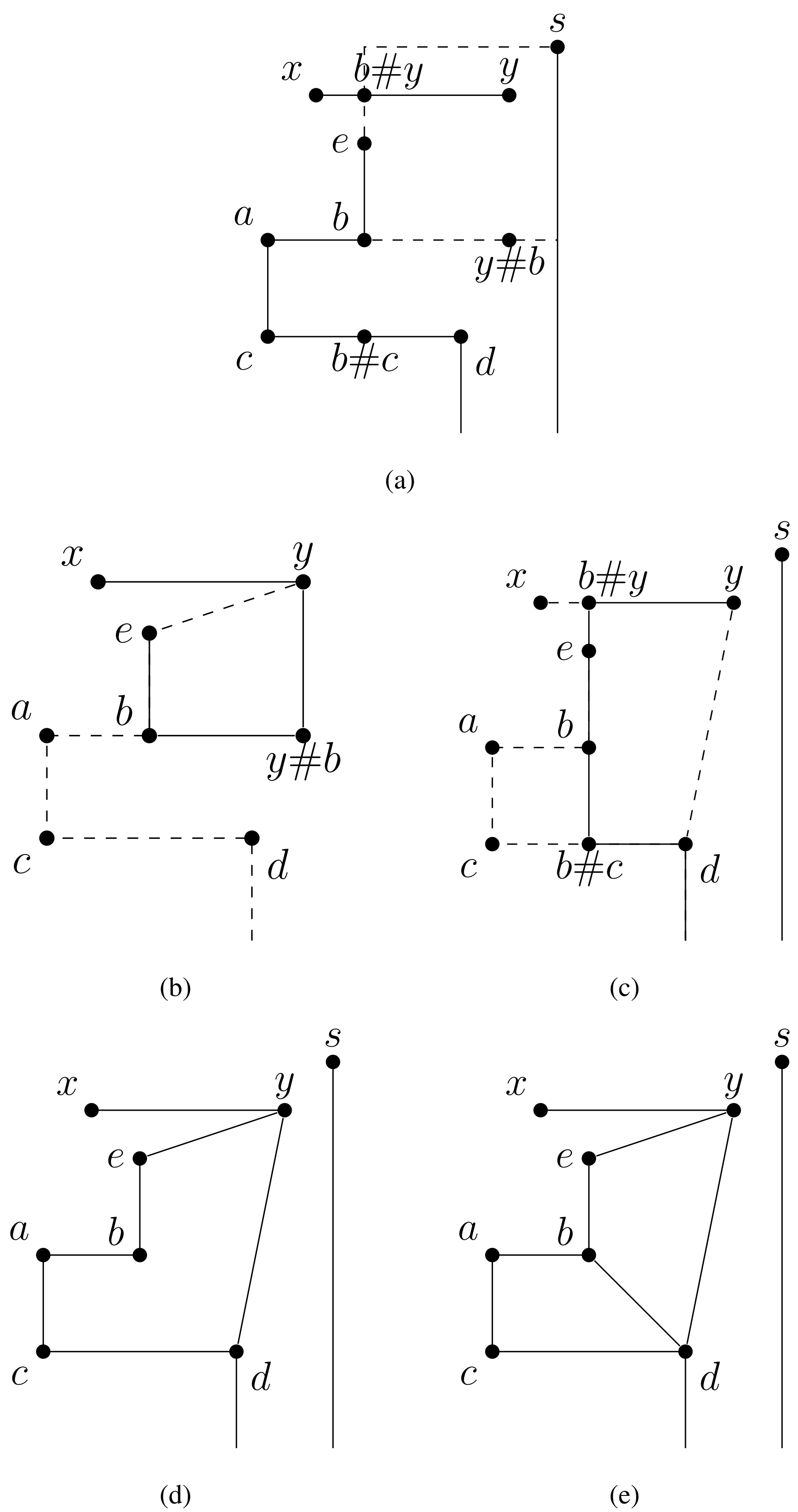


Figure 3.5: Illustration of Case (1).

will be the chain made from $y, b\#y, b\#c, d$ and the second will be the chain made from $x, y, y\#b, b, e$. Assume that P has no holes, then when we replace xy and the chain $ebacd$ with the two tabs we will get two new polygons P_1 and P_2. Without loss of generality, let P_1 contain $y, b\#y, b\#c, d$ and P_2 contain $x, y, y\#b, b, e$, seen in Figure 3.6c and Figure 3.6b. Because both P_1 and P_2 will have the same amount of holes as P and less vertices they are both smaller then P. Because every orthogonal polygon smaller P can be quadrilateralized, both P_1 and P_2 can be quadrilateralized. By Lemma 2.2.3, the quadrilaterization of P_1 will include the quadrilateral $y, b\#y, b\#c, d$.

Unlike in Case (1) $(y, y\#b)$ and eb are not neighbors in P_2 and do not form a tab. We can still show that either ey or bx is a part of the quadrilaterization. For contradiction assume $y\#b$ is a part of more then one quadrilateral. That would mean that $y\#b$ either connects to a vertex to the left of eb or above xy. This is because xy is the lowest edge that intersects $\square(b, s)$ so $\square(b, y)$ is empty. If $y\#b$ connects to a vertex left of eb, b can not be a part of a quadrilateral, a contradiction. If $y\#b$ connects to a vertex above xy, y can not be a part of a quadrilateral, a contradiction. Therefor either the quadrilateral $e, y, y\#b, b$ is a part of the quadrilaterization or the quadrilateral $b, x, y, y\#b$ is. In the case that x is between e and b we can see that P_2 can only contain the quadrilateral $b, x, y, y\#b$ and not the quadrilateral $e, y, y\#b, b$. This can be seen in Figure 3.7b.

We can then put the quadrilateralizations of P_1 and P_2 together to make the quadrilaterization of P. We can see that the quadrilaterization of P will include the quadrilaterals $yebd$ and $abed$ when we combine the quadrilaterals $e, y, y\#b, b$ and $y, b\#y, b\#c, d$, seen in Figure 3.6d. Or the quadrilaterization of P will include the quadrilaterals $yxdb$ and $abed$ when we combine the quadrilaterals $b, x, y, y\#b$ and $y, b\#y, b\#c, d$, seen in Figure 3.6e and Figure 3.7d.

$\square$

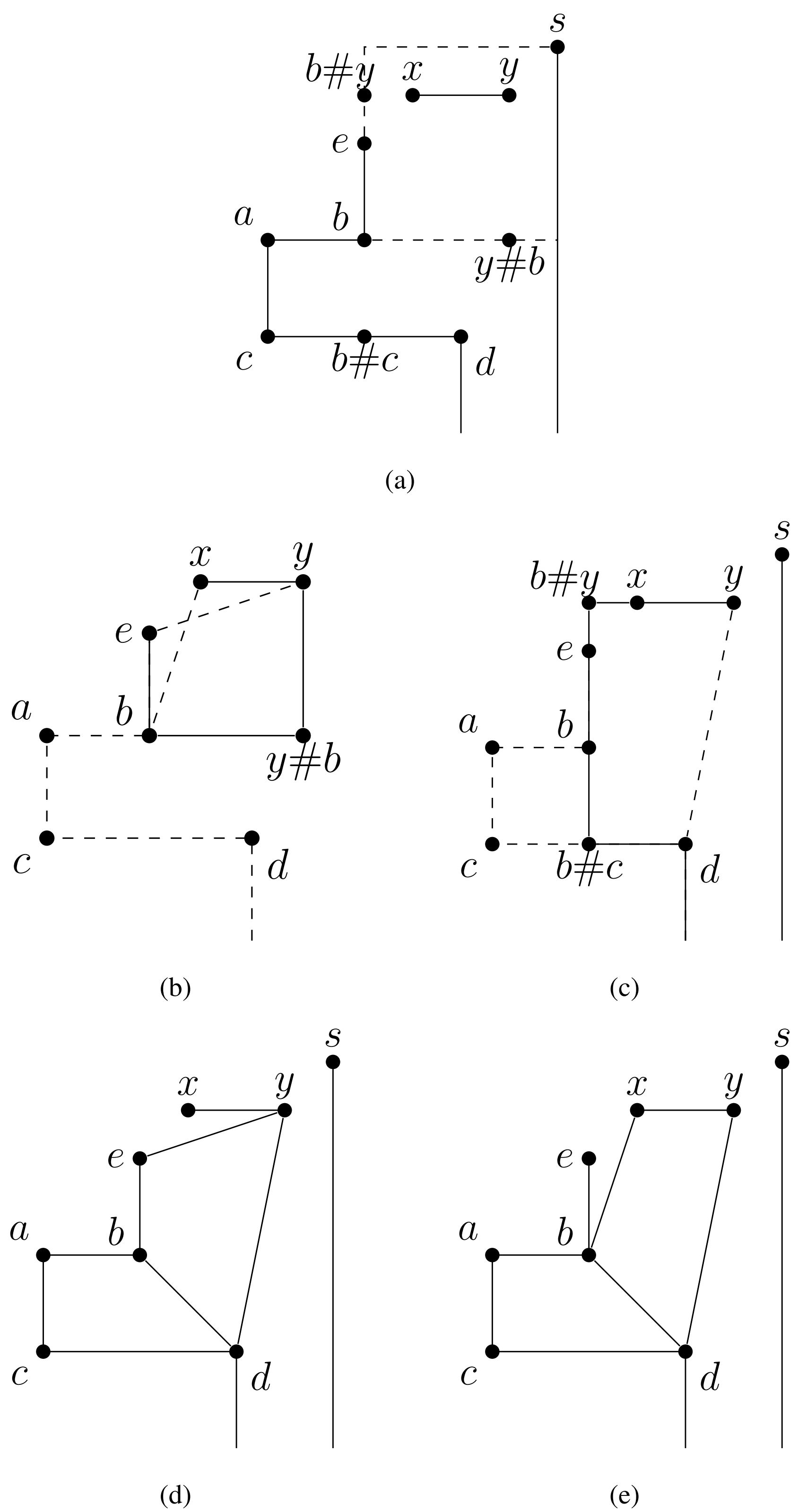
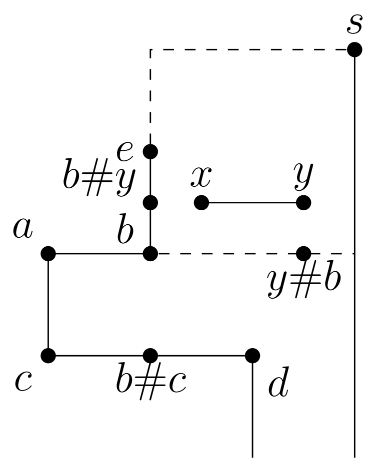
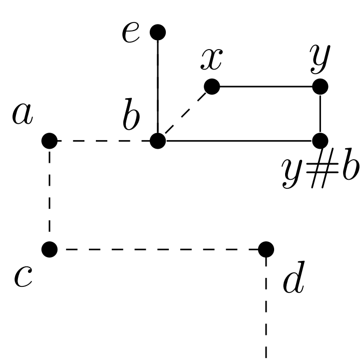
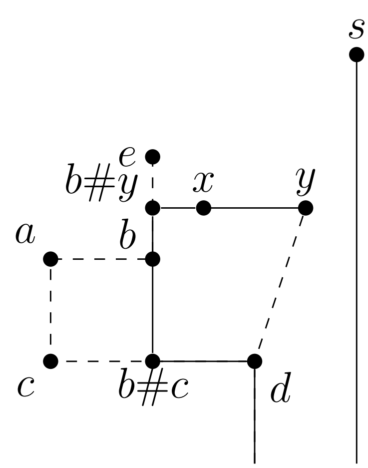


Figure 3.6: Illustration of Case (2).



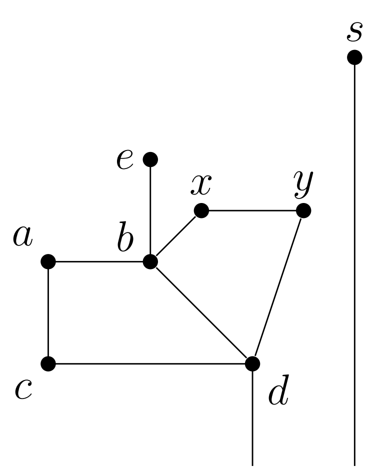
(a)





(b)

(c)



(d) Illustration of Case (2).

Figure 3.7: Illustration of Case (2).

3.3 Tab pairs

The final reduction is based around the exsitence of a *tab pair* in P. Let U be an up tab and D be a down tab. U and D make a tab pair if the step edge of U is the bottom edge of D and the step edge of D is the top edge of U.

Lemma 3.3.1. *If P contains a tab pair, then P is reducible.*

Proof. Let the chain $bacd$ form an up tab and the chain $fgih$ form a down tab, such that they both form a tab pair, as seen in Figure 3.8a.

Next replace the tab $bacd$ with the tab $f, a\#f, c, d$ and the tab $fghi$ with the tab $b, g\#b, i, h$. Assume that P has no holes, then this will create two new polygons, P_1 and P_2. Without loss of generality, let P_1 contain $f, a\#f, c, d$ and P_2 contain $b, g\#b, i, h$, seen in Figure 3.8b and Figure 3.8c. Both P_1 and P_2 have the same amount of holes as P and less vertices so are both smaller than P. Because every orthogonal polygon smaller P can be quadrilateralized, P_1 and P_2 can be quadrilateralized.

By Lemma 2.2.3, the quadrilaterization of P_1 contains the quadrilateral $f, a\#f, c, d$ and the quadrilaterization of P_2 contains the quadrilateral $b, g\#b, i, h$. We can then put the quadrilateralizations of P_1 and P_2 together to make the quadrilaterization of P. We can also see that the quadrilaterals $f, a\#f, c, d$ and $b, g\#b, i, h$ will make the chain $dcabhigfd$ in P, which can be quadrilateralized as $abed$, $bdfh$, and $hifg$, seen in Figure 3.8d. $\square$

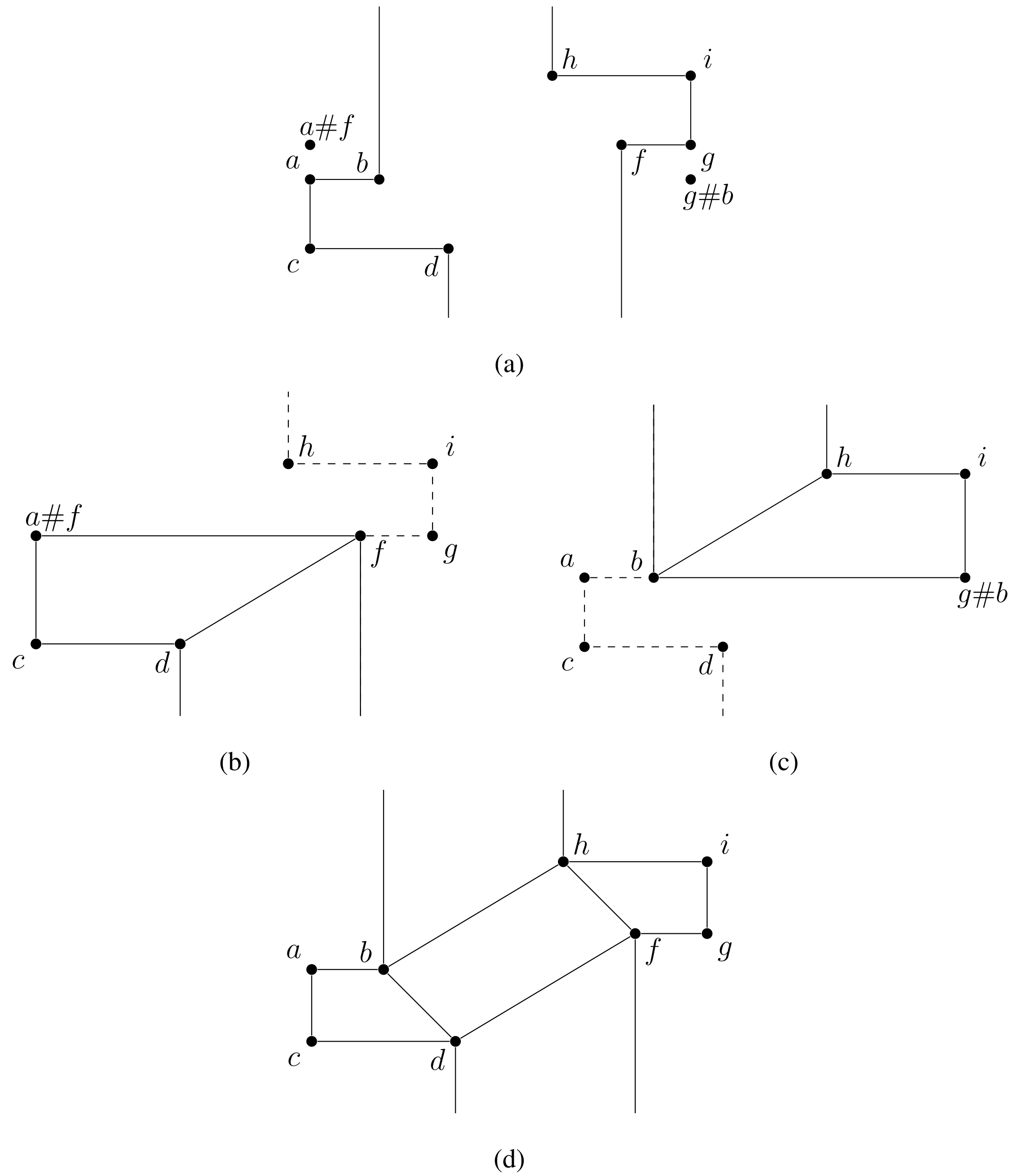


Figure 3.8: Reducing polygon with tab pairs.

CHAPTER 4

QUADRILATERALIZING THE ORTHOGONAL GALLERIES

We have shown, in the previous chapter, that if an orthogonal polygon contains one of three structures that it can be reduced. We now need to show that every orthogonal polygon contains at least one of these three structures.

This is done by contradiction, so for the following arguments we assume that P is irreducible.

4.1 LEMMAS

Let E be a top edge, define $n(E)$ to be the highest bottom edge that E can see. Similarly if E is a bottom edge, then $n(E)$ is the lowest top edge E can see. Notice that $n^{i+2}(E)$ must fall somewhere between $n^i(E)$ and $n^{i+1}(E)$ as $n^{i+2}(E)$ is supposed to be the closest "seeable" edge to $n^i(E)$. Also notice that if $n(n(E)) = n^2(E) = E$ then E and $n(E)$ are neighbors. Because P has a finite number of edges the sequence $E, n(E), n^2(E), \ldots, n^k(E), n^{k+1}(E)$ must also be finite ending when $n^k(E)$ and $n^{k+1}(E)$ are neighbors.

Because P is irreducible, by Lemma 3.1.1 there can not be any neighboring edges that are not a tab, so $n^k(E)$ and $n^{k+1}(E)$ must form a tab, call this $tab(E)$. By Lemma 3.2.1, P can not contain a good tab or it would be reducible so $tab(E)$ must be a bad tab.

Lemma 4.1.1. *Let E be a horizontal edge and $tab(E)$ as defined above. Let F the facing edge of $tab(E)$. Then*

- *if E is a top edge, E falls horizontally between F and the top edge of $tab(E)$; and*
- *E is a bottom edge, between F and the bottom edge of $tab(E)$.*

Proof. Consider $n^{k-1}(E)$, for contradiction assume that it extends horizontally past $n^{k+1}(E)$ or past F as seen in Figure 4.1.

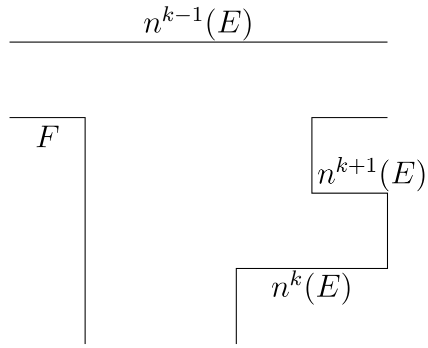


Figure 4.1: The edge $n^{k-1}(E)$ that extends past $n^{k+1}(E)$ or past F.

If $n^{k-1}(E)$ extends past $n^{k+1}(E)$ then it would see some bottom edge higher then $n^k(E)$ a contradiction.

Similarly, if $n^{k-1}(E)$ extends past F then it would see some bottom edge higher then $n^k(E)$ a contradiction.

We can repeat the above argument to claim the same about E. The analogous argument works for the case of bottom edge E. $\square$

Lemma 4.1.2. *Suppose P is irreducible and that E is a bottom edge such that $tab(E)$ is a down tab not containing E. Then there is a bottom edge $h(E)$ that is not part of a down tab.*

Proof. Let the top edge of $tab(E)$ be T and the bottom edge B, let the facing edge be F, the step point s and the step edge S, all as shown in Figure 4.2.

Because $tab(E)$ is a bad down tab we know that S is a top edge and that $\square(s, c)$ is empty. Because $\square(s, c)$ is empty we know that E must be below S.

We also know that E can not see S so let e be a point on E, then there is some point y that is blocking e from seeing S. This means there must be an edge going through y. The edge can not be a bottom edge as e can see y. It can also not be a top edge as then E would be able to see and therefor would see a lower edge than T, a contradiction.

Therefor the edge going through y must be a vertical edge. Let $h(E)$ be the horizontal

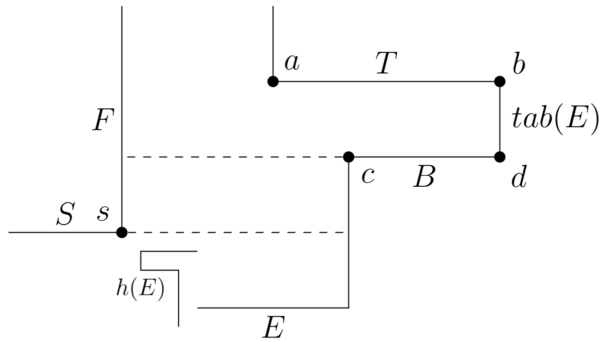


Figure 4.2: Illustration of Lemma 4.1.2.

edge connected to the vertical edge. $h(E)$ can not be a top edge or else E would be able to see it and therefor would see a lower edge. So $h(E)$ must be a bottom edge. Also, $h(E)$ can not be the bottom of a down tab because then e would be able to see the top edge of the tab and therefor would see a lower edge. □

Lemma 4.1.3. *If P is irreducible, then P has an infinite number of edges.*

Proof. Assume that P does have a finite number of edges. Then we know that the sequence $G, n(G), n^2(G), \ldots$ must lead to some tab U. Without loss of generality, assume that U is the highest up tab.

The following descriptions can be seen in Figure 4.3. Let ab be the top edge of U, S be the step edge of U which we know if a bottom edge because U is a bad up tab. Let s be the step point of U and e be the top of the vertical edge connected with b. Let E be the horizontal edge connected to e.

We can then have two possibilities to consider.

Case (1) S is not a part of $tab(S)$. We know that $tab(S)$ is above S and S is above U, so $tab(S)$ is above U. Because U is the highest up tab $tab(S)$ must be a down tab. If S is not a part of $tab(S)$ then we can apply Lemma 4.1.2 to S and get $h(S)$. This is because P is irreducible, S is a down tab, and $tab(S)$ is a down tab not containing S. We can then see that $tab(h(S))$ is above U because $tab(h(S))$ is above $h(S)$ and

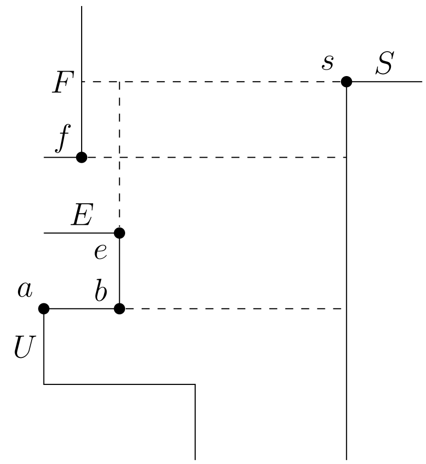


Figure 4.3: Illustration of Lemma 4.1.3.

$h(S)$ is above S. We can then apply Lemma 4.1.2 to get $h^2(S)$. We can then continue this process for an infinite amount of times showing that P is be infinite.

Case (2) S is a part of $tab(S)$. In this case $tab(S)$ must still be a down tab because it is higher then U. Let F be the facing edge of $tab(S)$ and f the step point. Because U and $tab(S)$ can not be a tab pair by Lemma 3.3.1 $f \neq b$. We also know that because $tab(S)$ is a bad tab that $\square(s, f)$ is empty and that because U is a bad tab $\square(b, s)$ is empty. Because $\square(b, s)$ is empty we can see that E must be a bottom edge. Because E is higher than U, $tab(E)$ must be a down tab.

We will now show that E can not be a part of $tab(E)$. If E was a part of $tab(E)$ then the top edge of the tab would have to extend farther then E. We can see that the top edge of $tab(E)$ must be horizontally between b and s. Because $\square(b, s)$ must be empty then it must be above bottom edge of $tab(S)$. But because of Lemma 2.2.2 the rectangle determined by $tab(S)$ must be empty so the top edge of $tab(E)$ can not go higher then bottom edge of $tab(S)$, a contradiction. Thus E can not be a part of $tab(E)$. This means that E is a bottom tab and $tab(E)$ is a down tab so we can now apply Lemma 4.1.2 to E to get $h(E)$. We now have a bottom edge $h(E)$ that is not a part of the down tab $tab(h(E))$ so we can apply Lemma 4.1.2 again. We can then continue this process for an infinite amount of times showing that P is be infinite.

□

4.2 MAIN THEOREM

From everything we have shown we can now state the following theorem.

Theorem 4.1 ([5]). *Every orthogonal polygon has a convex quadrilaterization.*

Proof. By Lemma 2.2.1 we know that we can consider a polygon whose vertices are in general position. We will then proceed by induction. The base case is a rectangle which can obviously be quadrilateralized as it is a quadrilateral. Assume that every orthogonal polygon smaller then P can be quadrilateralized. We then apply Lemma 4.1.3 to show that all orthogonal polygons contains at least one of these three structures, a pair of neighboring edges that do not form a tab, a good tab, or a tab pair. We next use Lemma 3.1.1, Lemma 3.2.1, and Lemma 3.3.1 to show that we can reduce P to a smaller P' that we can quadrilateralize. We can then use the quadrilaterization of P' to make a quadrilaterization of P. □

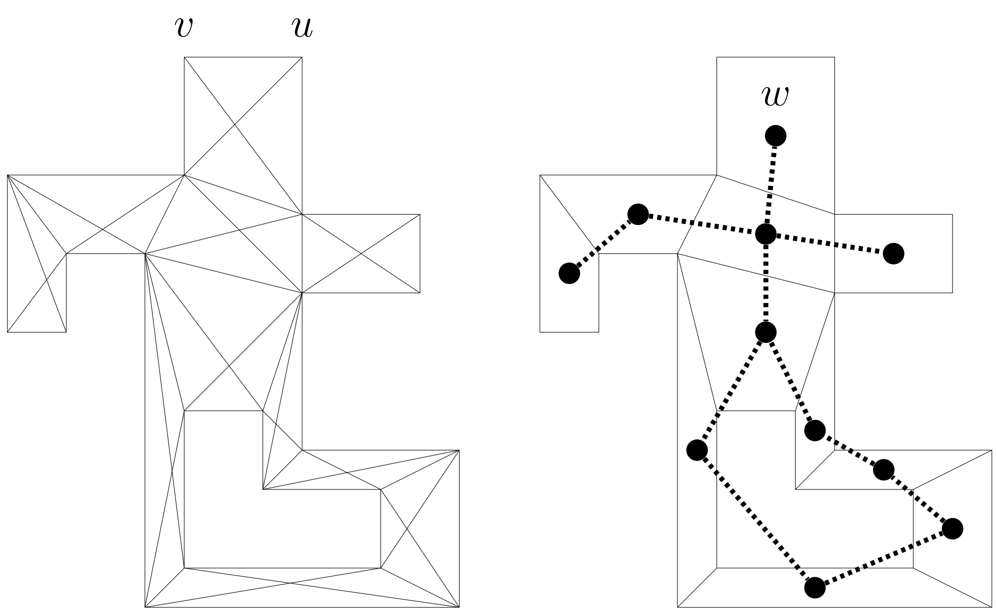
CHAPTER 5

GUARDING AN ORTHOGONAL GALLERY WITH "HOLES"

It is well known that through 4-coloring the quadrilateralization of an orthogonal one can guard the corresponding art gallery with at most $\frac{n}{4}$ guards. In this chapter, we will discuss this idea through the result of [11], where orthogonal galleries with holes are considered.

5.1 Preparation

Let us first look at an orthogonal gallery with one hole. Let Q be the quadrilateralization of the orthogonal gallery. After we quadrilateralize the orthogonal gallery we can add edges connecting the corner vertices of the quadrilaterals to make the quadrilateralization graph G_Q as seen in Figure 5.1a.



(a) (b)

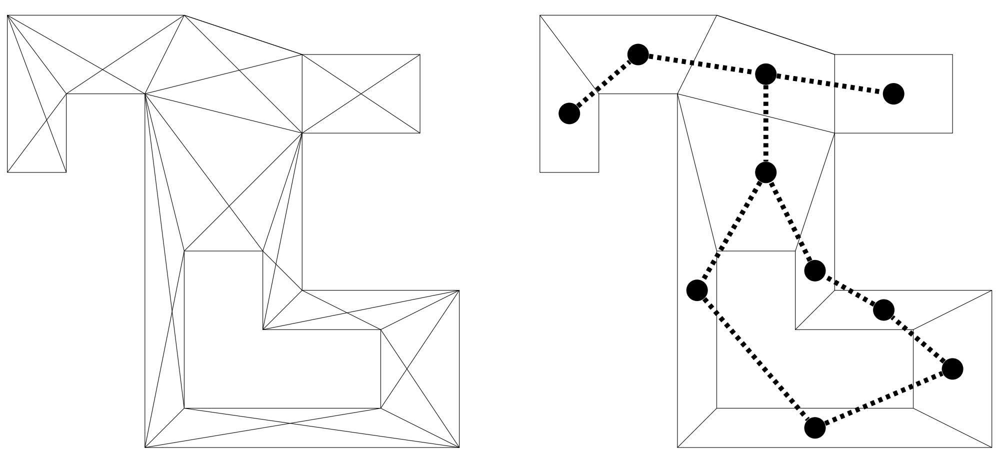
Figure 5.1: (a) Example of the quadrilateralization graph, G_Q, of an orthogonal gallery with one hole. (b) Example of G_D.

If we can show that we are able to 4-color G_Q then we can use that coloring to place guards that will sufficiently guard the orthogonal gallery.

We consider the dual graph G_D of G_Q. In G_D every vertex corresponds to a quadrilateral in G_Q. In G_D two vertices are connected by an edge if their corresponding quadrilaterals share an edge. An example of G_D can be seen in Figure 5.1b.

We can see that G_D is made up of a single cycle with multiple branches attached. At the end of each branch is a vertex of degree one in G_D. The corresponding quadrilateral in G_Q has two vertices of degree 3. This is because two of the quadrilaterals vertices will only be a part of that quadrilateral (and none else). Because of this those two vertices will only be connected to each other and the other two vertices in the quadrilateral. So the degree of each vertex will be 3.

We can observe the above discussion in Figure 5.1b that w has a degree of one. In Figure 5.1a we can see that the corresponding quadrilateral to w has two vertices, v and u of degree 3. We can remove these vertices and call the resulting graph G_Q^1 and its dual graph G_D^1, as seen in Figure 5.2.

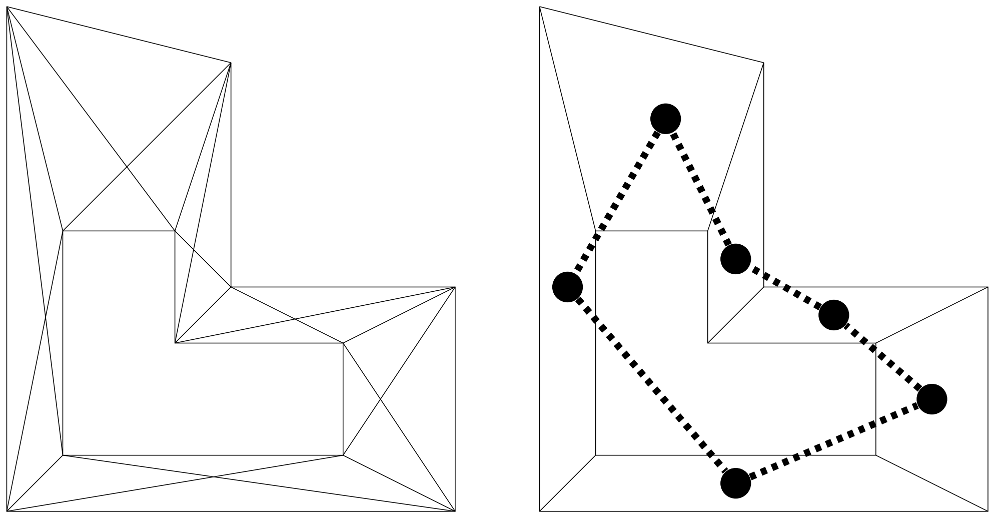


(a) (b)

Figure 5.2: (a) Example of G_Q^1. (b) Example of G_D^1.

If we can 4-color G_Q^1 then we can also 4-color G_Q. We can repeat this process of removing vertices with a degree of one from our last dual graph until we are left with

G_D^k and G_Q^k, as seen in Figure 5.3. G_D^k will just be a single cycle and G_Q^k will be the corresponding quadrilaterals. If we can 4-color G_Q^k then we can also 4-color G_Q.



(a) (b)

Figure 5.3: (a) Example of G_Q^k. (b) Example of G_D^k.

5.2 THE STUDY OF G_Q^k

In G_Q^k we can observe an interior boundary and an exterior boundary. We call a quadrilateral *balanced* if it has two vertices on the interior boundary and two vertices on the exterior boundary. A quadrilateral is called *skewed* otherwise.

Here we quote an observation from [11], that "the next step is to observe that"

Observation 5.1. *Each skewed quadrilateral has one vertex of degree 3 in graph G_Q^K.*

Proof. This is because each skewed quadrilateral has three vertices on the exterior boundary and one vertex on the interior boundary, or one vertex on the exterior boundary and three vertices on the interior boundary. Without loss of generality, suppose the skewed quadrilateral has three vertices on the exterior boundary and one vertex on the interior boundary, as seen in Figure 5.4, where v_1, v_2, v_7, and v_6 make up the skewed quadrilateral.

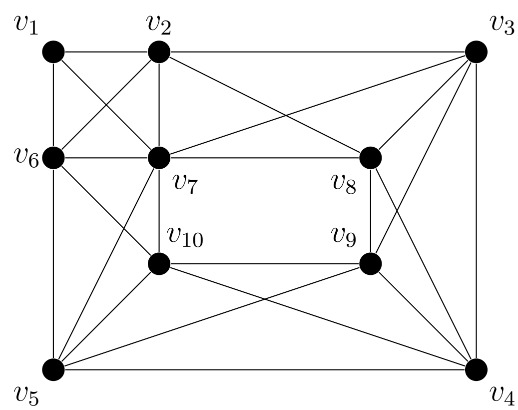


Figure 5.4: A graph with one skewed triangle.

We can see that v_6, v_1, and v_2 are the vertices on the exterior boundary. We can also see that because v_1 is in the middle of these vertices that it is not a part of any other quadrilateral. Because of this it only has edges connecting it to the other vertices that are a part of the same quadrilateral. □

5.3 Coloring algorithm

We can remove all vertices from G_Q^k that have a degree of three resulting in a graph G_Q^*. If we are able to 4-color G_Q^* then we are also able to 4-color G_Q^k. G_Q^* will consist of an even number of balanced quadrilaterals and some number of triangles. For each triangle we will call it an *e-triangle* if it has two vertices on the exterior boundary and an *i-triangle* if it has two vertices on the interior boundary.

We can now present an algorithm that will 4-color G_Q^*. Doing so will require 4 cases.

- The number of i-triangles is even and the number of e-triangles is even. From [1] we know that the cycle in the dual graph of any quadrilateralization of an orthogonal polygon with one hole has an even number (at least four) of balanced quadrilaterals. By that we can gather that G_Q^* has $m = 2l$ vertices, $l \geq 4$. We can now label the vertices on the exterior boundary $v_1, v_2, \ldots, v_{2k}$ for $k \geq 2$ and the interior vertices

$v_{2k+1}, v_{2k+2}, \ldots, v_m$, both being labeled in a clockwise manner. We can then define a coloring as follows:

$$c(v_i) = \begin{cases} 1 & \text{if } (i \leq 2k) \text{ and } (i \equiv 1 \pmod 2), \\ 2 & \text{if } (i \leq 2k) \text{ and } (i \equiv 0 \pmod 2), \\ 3 & \text{if } (2k < i \leq m) \text{ and} (i \equiv 1 \pmod 2), \\ 4 & \text{if } (2k < i \leq m) \text{ and } (i \equiv 0 \pmod 2), \end{cases}$$

– Before moving on, we will explain, in detail, why this coloring works. Similar discussions for other cases will be skipped.

  Note that what we have is essentially two *even cycles*. A cycle is a sequence of connected vertices that starts and ends with the same vertex and other then the first vertex no vertex is repeated. In Figure 5.4 we can see an example of a cycle in $v_1v_2v_3v_4v_5v_5v_6v_1$. We can also see that it is an even cycle because it has an even number of vertices in it. Similarly a odd cycle has an odd number of vertices. To properly color an even cycle we only need two colors. We can see this by considering any even cycle. Then label the vertices of the cycle in a clockwose manner $u_1u_2u_3 \ldots u_{2n-1}u_{2n}$ where n is any integer. We can then define a coloring to be as follows:

$$c(u_i) = \begin{cases} 1 & \text{if } (i \equiv 1 \pmod 2), \\ 2 & \text{if } (i \equiv 0 \pmod 2), \end{cases}$$

  We can also see in Figure 5.5, where instead of 1 and 2 we color the vertices red and blue, that this will hold. It is because when we define the coloring this way the vertices will alternate colors so a vertex colored 1 will only be connected to two other vertices and both will be colored 2.

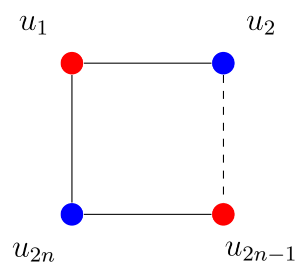


Figure 5.5: Coloring an even cycle.

We can now look back at our first case to see that the exterior boundary is an even cycle. This means that we can properly color the exterior boundary using only two colors. We can do the same with the interior boundary. We can then see that for the interior boundary if we pick two different colors from what we colored the exterior boundary with that the result will be a 4-coloring of G_Q^*. This is because both cycles will be properly colored and any edges connecting two vertices from the two cycles can not be connecting two vertices of the same color because the exterior and interior boundaries were colored using different colors.

- The number of i-triangles is odd, and the number of e-triangles is even. Using the same fact as before that the cycle in the dual graph of any quadrilateralization of an orthogonal polygon with one hole has an even number (at least four) of balanced quadrilaterals. we can gather that G_Q^* has $m = 2l + 1$ vertices, $l \geq 4$. We can now label the vertices on the exterior boundary $v_1, v_2, \ldots, v_{2k}$ for $k \geq 2$ and the interior vertices $v_{2k+1}, v_{2k+2}, \ldots, v_m$, both being labeled in a clockwise manner. W.L.O.G. assume that $v_m v_{2k+1} v_1$ is an i-triangle. We can then split v_m into v'_m and v''_m. Let $N(v'_m) = N(v_m)/\{v_{2k+1}\}$ and $N(v''_m) = \{v_1, v_{2k+1}\}$. We can now define a coloring as follows:

$$c(v_i) = \begin{cases} 1 & \text{if } (i \leq 2k) \text{ and } (i \equiv 1 \pmod 2), \\ 2 & \text{if } (i \leq 2k) \text{ and } (i \equiv 0 \pmod 2), \\ 3 & \text{if } ((2k < i \leq m) \text{ and } (i \equiv 1 \pmod 2)) \text{ or } (v_i = v'_m), \\ 4 & \text{if } ((2k < i \leq m) \text{ and } (i \equiv 0 \pmod 2)) \text{ or } (v_i = v''_m), \end{cases}$$

- The number of i-triangles is even, and the number of e-triangles is odd. This case is done the same way case 2 except that we use the endpoint of the external edge of an e-triangle.

- The number of i-triangles is odd, and the number of e-triangles is odd. This means that there is either an i-triangle or an e-triangle that shares an edge with a balanced quadrilateral. Without loss of generality, assume it is an i-triangle that shares an edge with a balanced quadrilateral. We can now label the vertices on the exterior boundary $v_1, v_2, \ldots, v_{2k+1}$ for $k \geq 2$ and the interior vertices $v_{2k+2}, v_{2k+3}, \ldots, v_m$, both being labeled in a clockwise manner. It is important to point out that m is even. We can then split v_1 into v'_1 and v''_1. Let $N(v'_1) = N(v_1)/\{v_2, v_{2k+3}\}$ and $N(v''_1) = \{v_2, v_{2k+2}, v_{2k+3}\}$. We can now define a coloring as follows:

$$c(v_i) = \begin{cases} 1 & \text{if } ((2 \leq i \leq 2k) \text{ and } (i \equiv 1 \pmod 2)) \text{ or } (v_1 = v_{2k+2}), \\ 2 & \text{if } ((2 \leq i \leq 2k) \text{ and } (i \equiv 0 \pmod 2)) \text{ or } (v_i = v'_1), \\ 3 & \text{if } (2k+3 \leq i \leq m) \text{ and } (i \equiv 1 \pmod 2), \\ 4 & \text{if } ((2k+3 \leq i \leq m) \text{ and } (i \equiv 0 \pmod 2)) \text{ or } (v_i = v''_1), \end{cases}$$

CHAPTER 6

A RECURSIVE APPROACH FOR TERRAIN GUARDING PROBLEMS

As mentioned before, another important aspect of our study of the sensor placement is the well known terrain guarding problem. In this chapter we describe an easy-to-use approach to provide a fast algorithm that generates a close to optimal solution. This is done through a *greedy algorithm*, discussed in detail in the next section.

6.1 GREEDY ALGORITHM

Definition 9. Greedy algorithm picks the option that will get you closest to your goal at the time.

Greedy algorithms can be found in many applications in everyday life. This is because it is a very easy and natural algorithm to use.

Example 6.1. *A good example of greedy algorithm is U.S. coins. U.S. coins are designed to work with greedy algorithm. This can be seen easily observed with some examples:*

- *If we want to get 67¢using greedy algorithm we will first start with one quarter. We will then have 25¢. Then to get to 67¢we will use another quarter as that will get us the closest. We now have 50¢so we will use a dime which will get us to 60¢. Then we an use a nickel which will get us to 65¢. And finally we can use two pennies which will get use to 67¢. We can also see that there is no way we could use less coins to make 67¢. Therefor by using greedy algorithm we have found the most efficient way to make 67¢.*
- *If the coins have different values, however, greedy algorithm will not necessarily achieve the best result (i.e. fewest coins). Suppose, for instance, we have coin values of 1¢, 7¢, and 10¢. To make 15¢we would have used a 10¢coin and five 1¢coins. On*

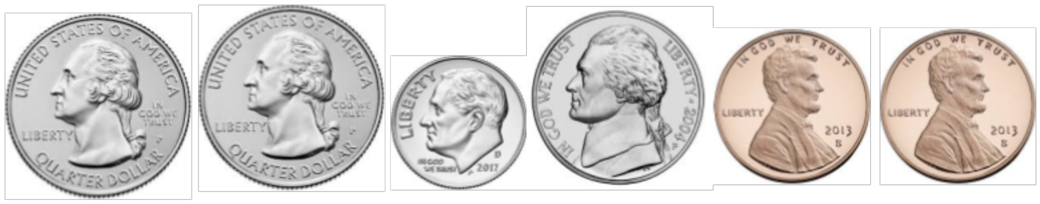


Figure 6.1: Example of the most efficient way to make 67¢.

the other hand, using two 7¢coins and 1¢coin is obviously a better solution (using fewer coins).

- *Furthermore, if wee use coins that have a value of 2¢, 3¢, 5¢, 7¢, and 11¢. If we wanted to make 23¢using greedy algorithm we would first pick 11¢. Then we would pick 11¢again. We would then have 22¢and there would be no way to get to 23¢. On the other hand one could have used 11¢, 5¢, 5¢, 2¢. Hence, greedy algorithm has failed to even produce a feasible solution in this scenario.*

6.2 OUR APPROACH FOR THE TERRAIN GUARDING PROBLEM

Here we will be considering the continuous version of the terrain guarding problem as it is more applicable. And to solve it we will be using the greedy algorithm.

Our plan is to first place a guard on the vertex that is able to see the most edges. We will then look at the remaining edges that cannot be seen and place a guard on the vertex that is able to see the most of the unseen (yet) edges. We will continue this process until all of the edges are seen.

We also know that all of the vertices will be seen as to see an edge you must see both the vertices that the edge is attached to.

Lastly, this process must terminate as there are finitely many edges and the number of unseen edges strictly decrease in each step.

Thus using greedy algorithm (Algorithm 1) we will be able easily and quickly place guards that will be able to see the whole terrain.

Algorithm 1 Finding the Optimal Solution to the Terrain Guarding Problem

procedure INITIALIZE

  $V_0 = \{v_1, v_2, \dots, v_n\} \leftarrow$ *The vertices of the terrain*;

  $E_0 = \{v_1v_2, v_2v_3, \dots, v_{n-1}v_n\} \leftarrow$ *The edges of the terrain*;

  $j = 0$;

  while $E_j \neq \emptyset$ **do**

    $v_i \leftarrow$ *The vertex that can see the most edges in* E_j;

    $W(v_i) = \{v_{a1}v_{a2}, \dots, v_{a(x-1)}v_{ax}\} \leftarrow$ *All the edges* v_i *can see in* E_j;

    $E_{j+1} = E_j / W(v_i)$;

    $j = j + 1$;

    $x(j) = v_i$;

  $x \leftarrow$ *The vertices that can see the entirety of the terrain*

Example 6.2. *As an example, we start with the structure in Figure 6.2a.*

Then

$$V_0 = \{v_0, v_1, v_2, v_3, v_4, v_5\}$$

and

$$E_0 = \{v_0v_1, v_1v_2, v_2v_3, v_3v_4, v_4v_5\}.$$

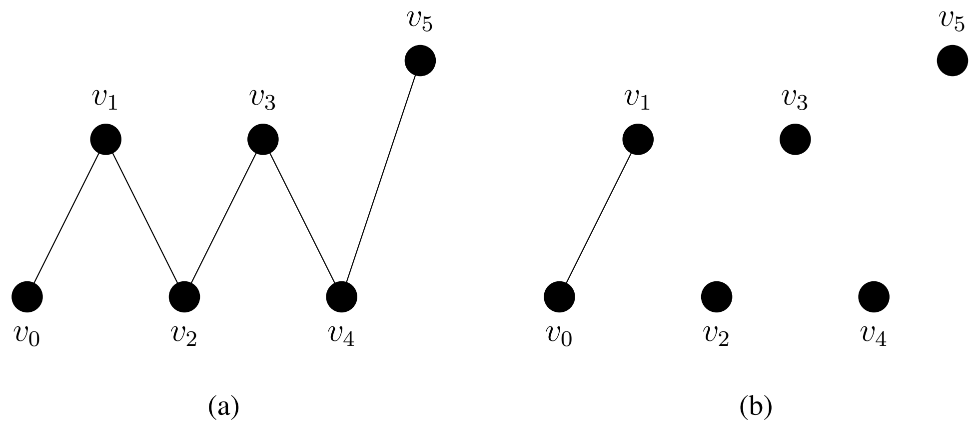


Figure 6.2: (a) Graph made of the sets V_0 and E_0. (b) Graph made of the sets V_0 and E_1.

We now consider E_0 and we find the vertex that is able to see the most edges. In E_0 that vertex is v_3. We can now let

$$W(v_3) = \{v_1v_2, v_2v_3, v_3v_4, v_4v_5\}$$

as those are the edges that v_3 can see. We will now let

$$E_1 = E_0/W(v_3) = \{v_0v_1, v_1v_2, v_2v_3, v_3v_4, v_4v_5\}/\{v_1v_2, v_2v_3, v_3v_4, v_4v_5\} = \{v_0v_1\},$$

as seen in Figure 6.2b, and $x(1) = v_3$ so that $x = \{v_3\}$.

Because $E_1 \neq \emptyset$ we will repeat this process again with E_1. We now find the vertex that can see the most edges in E_1. We can see that there are two vertices able to see all the edges in E_1 so we can just pick one. We will pick v_1.

Following the same process we let

$$W(v_1) = \{v_0v_1\}$$

and

$$E_2 = E_1/W(v_1) = \{v_0v_1\}/\{v_0v_1\} = \emptyset.$$

We then let $x(2) = v_1$ so now $x = \{v_3, v_1\}$. We can also see that $E_2 = \emptyset$ so we are done and we can guard the entire terrain by placing guards at the vertices in x.

REFERENCES

[1] A. Aggarwal, The Art Gallery Theorem: Its Variations, Applications, and Algorithmic Aspects, Ph.D. Thesis, The Johns Hopkins University 1984.

[2] Pradeesha Ashok, Fedor V. Fomin, Sudeshna Kolay, Saket Saurabh, Meirav Zehavi, Exact Algorithms for Terrain Guarding, ACM Transactions on Algorithms Volume 14 Issue 2 Article No. 25 June, 2018

[3] Pawan Kumar Aurora, Coloring 3-colorable Graphs

[4] Nicole Chesnokov, The Art Gallery Problem: An Overview and Extension to Chromatic Coloring and Mobile Guards

[5] J. Kahn, M.Klawe, D. Kleitman, Traditional Galleries Require Fewer Watchmen, SIAM journal on algebraic and discrete methods, vol. 4, No. 2, June 1983

[6] Erik Krohn, Survey of Terrain Guarding and Art Gallery Problems, Comprehensive Exam Paper. November, 2007.

[7] Jyh-Ming Lieng, Polygon Partitioning

[8] Joe Morris, Combinatorics, an upper-level introductory course in enumeration, graph theory, and design theory, Version 1.1 of June 2017

[9] Joseph O'rourke, Art Gallery Theorems and Algorithms, Oxford University Press, 1987

[10] Daniel Vlasic, Polygon Triangulation, Lecture notes: `https://people.csail.mit.edu/indyk/6.838-old/handouts/lec4.pdf`

[11] Pawel Zylinski, Orthogonal art galleries with holes: a coloring proof of Aggarwal's Theorem, March 7, 2006