Computational Geometry

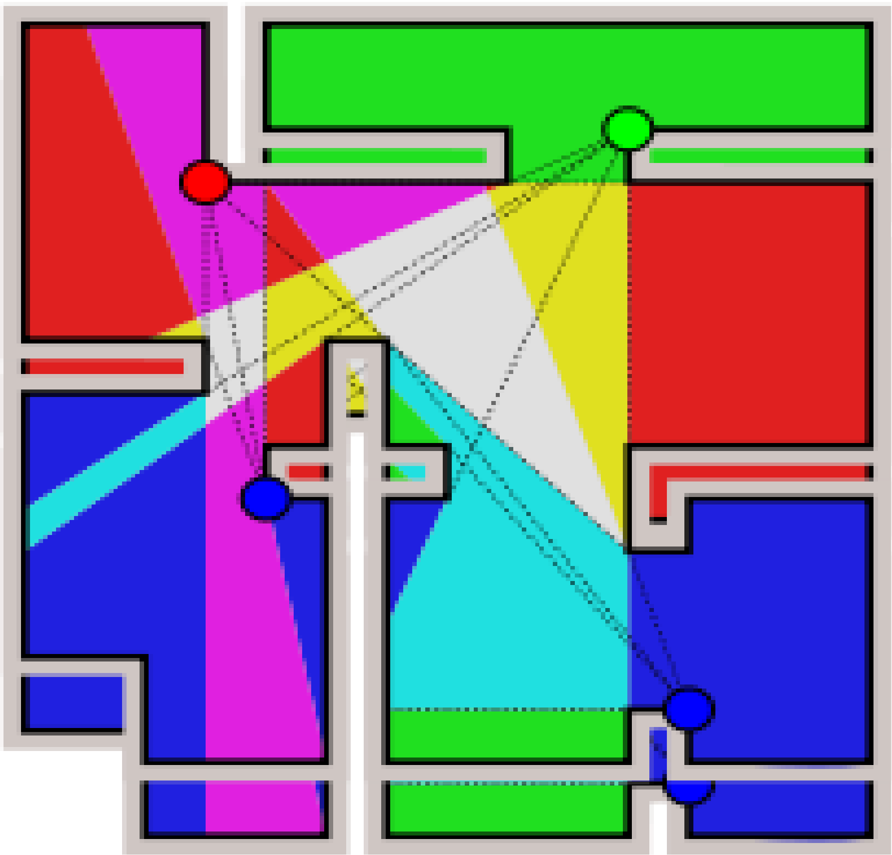
Final Report

Spring ‘19

VLASSIS PANAGIOTIS

1115201400022

“Art Gallery Theorems and Visibility Graphs”



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικό και Καποδιστριακό
Πανεπιστήμιο Αθηνών

ΤΜΗΜΑ
ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Table of Contents

Introduction

This assignment was developed in the context of course Computational Geometry, Dit@UoA. The project is divided in two research points:

- Art Gallery Theorems
- Visibility Graphs

1. Art Gallery Theorems

1.1 General

<u>Definitions:</u>

*<u>Art gallery theorem</u>: A typical "<u>art gallery theorem</u>" provides combinatorial bounds on the number of **guards** needed to visually cover a polygonal region P (the art gallery) defined by n vertices.*

<u>Guard</u>: A point, a line segment, or a line – a source of visibility of illumination.

<u>Interior visibility</u>: A guard $x \in P$ can see a point $y \in P$ if the segment xy is nowhere exterior to P: $xy \subseteq P$.

<u>Exterior visibility</u>: A guard x can see a point y outside P if the segment xy is nowhere interior to P; xy may intersect ∂P , the boundary of P.

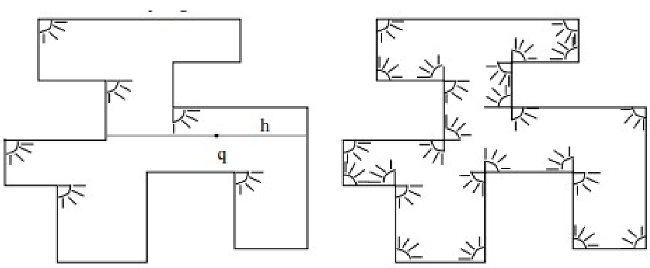


Figure 1.1 Illuminating a orthogonal polygon with orthogonal floodlights

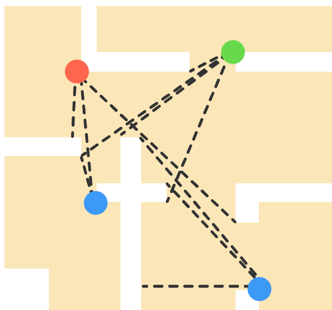


Figure 1.2 Point guards to guard a museum

1.2 Related Problems

The most general combinatorial results obtained to date are summarized in Figure 1.3. For some of the problems, a analysis is following next.

PROBLEM NAME	POLYGONS	INT/EXT	GUARD	NUMBER
Art gallery theorem	simple	interior	vertex	$\lfloor n/3 \rfloor$
Fortress problem	simple	exterior	point	$\lceil n/3 \rceil$
Prison yard problem	simple	int & ext	vertex	$\lceil n/2 \rceil$
Prison yard problem	orthogonal	int & ext	vertex	$[\lceil 5n/16 \rceil, \lfloor 5n/12 \rfloor + 1]$
Orthogonal polygons	simple orthogonal	interior	vertex	$\lfloor n/4 \rfloor$
Orthogonal with holes	orthogonal with h holes	interior	vertex	$[\lfloor 2n/7 \rfloor, \lfloor (17n-8)/52 \rfloor]$
Orthogonal with holes	orthogonal with h holes	interior	vertex	$[\lfloor (n+h)/4 \rfloor, \lfloor (n+2h)/4 \rfloor]$
Polygons with holes	polygons with h holes	interior	point	$\lfloor (n+h)/3 \rfloor$

Figure 1.3 Table with problems related to art gallery theorems

1.2.1 Art gallery theorem

Definition: Any simple polygonal museum with n walls can be guarded by $\lfloor n/3 \rfloor$ at most guards.

S.Fisk (1978) Short Proof

- First, the polygon is triangulated (without adding extra vertices). It is known that vertices of the resulting triangulation graph may be 3-colored.
- Clearly, under a 3-coloring, every triangle must have all three colors.
- The vertices with any color form a valid guard set, because every triangle of the polygon is guarded by its vertex with same color.
- Since the three colors partition the n vertices of the polygon, the color with the fewest vertices defines a valid guard set at most $\lfloor n/3 \rfloor$ guards.

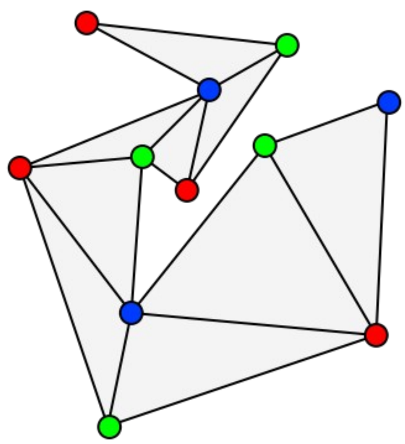


Figure 1.4 A 3-coloring of the vertices of a triangulated polygon. The blue vertices form a set of three guards, as few as is guaranteed by the art gallery theorem. However, this set is not optimal: the same polygon can be guarded by only two guards.

S.Fisk (1978) Constructive Proof

- Algorithm (or sequence of steps) that tells us exactly where to place the guards. To show that bound in the theorem is tight, consider the museum with 15 walls in the shape of a comb.

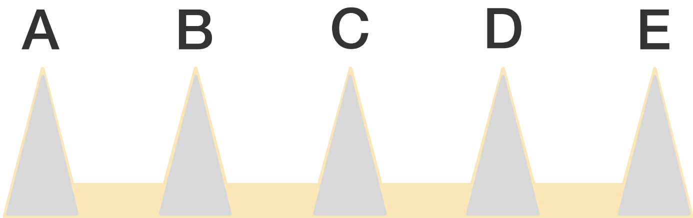


Figure 1.5 Museum in shape of a comb

- Then the guard for each point (A,B,C,D,E) must be stationed within the shaded triangle with related vertex (A,B,C,D,E).
- Since these triagles do not overlap, at least 5 guards are needed. But by the Art Gallery Theorem, $\left\lfloor \frac{15}{3} \right\rfloor = 5$ guards are also sufficient, which we can be observe by placing the guards at the lower left corner of each shaded triangle.
- In general, the comb museum layout gives an example of a museum with 3n walls that requires exactly $\left\lfloor \frac{3n}{3} \right\rfloor = n$ guards, which shows that the bound in the theorem is best possible.

1.2.2 Orthogonal Prison yard problem

In the prison yard problem, we are required to guard simaltaneously the interior and the exterior of a polygon. For arbitrary polygons this problem is closed. For orthogonal polygons, the problem reamins open. An upper bound was found by Hoffman & Krieger (1996), who showed that $\left\lfloor \frac{5n}{12} \right\rfloor + 2$ (resp. $\left\lfloor \frac{n+4}{3} \right\rfloor$) vertex guards (resp. point) guards are always sufficient to guard the interior and the exterior of an orthogonal polygon polygon with n vertices and holes.

This bound was obtained via the following graph-coloring theorem.via the following graph-coloring theorem. Every plane, bipartite, 2-connected graph has an even triangulation (all nodes have even degree) and therefore the resulting graph is 3-colorable.

They also conjecture:

Conjecture (HK93): There is a constant c such that any orthogonal prison yard can be guarded with $\left\lfloor \frac{5n}{16} \right\rfloor + c$ vertex guards.

This bound was subsequently improved to $\left\lfloor \frac{5n}{12} \right\rfloor + 1$ by Michael,Pinciu (2012).

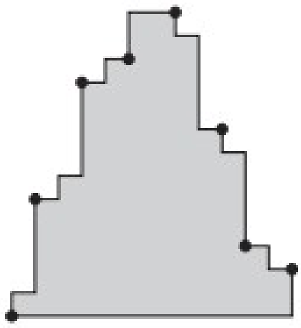


Figure 1.6

A pyramid polygon with $n = 24$ vertices whose interior and exterior are covered by 8 guards. Repeating the pattern establishes a lower bound of $5n/16 + c$ on the orthogonal prison yard problem [HK93].

1.2.3 Orthogonal polygons without holes

Theorem (Kahn,Klawe,Kleitman 1980): $\left\lfloor \frac{n}{4} \right\rfloor$ guards are sometimes necessary and always sufficient to cover the interior of an orthogonal polygon of n vertices.

Proof:

- Necessity is established by the orthogonal version of Chvatal’s comb example: one guard is needed for each tong in Figure 1.7.
- For sufficiency, construct a graph G from a quadrilateralization of P by adding both diagonals to each quadrilateral, as illustrated in Figure 1.8. Although it is not immediately obvious, G is planar, and therefore 4-colorable. We can establish 4-colorability without invoking the Four Color Theorem as Follows.
  - Let Q be the dual of the quadrilateralization of P: each node of Q corresponds to a quadrilateral, and two nodes are connected by an arc if their quadrilaterals share a side. Then Q must be a tree, for if it contained a cycle, this would imply that P has a hole.
  - Now proceed by induction. Remove any leaf quadrilateral q, leaving the tree Q’. Since q has degree 1, it may be removed by cutting along a single diagonal d of quadrilateralization. Four-color Q’ by the induction hypothesis, and reattach q to Q’. Two of q’s vertices are assigned different colors at the reattachment points, the endpoint of d, and the other two vertices of q can be assigned the remaining colors.

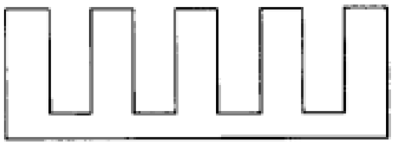


Figure 1.7 Orthogonal polygon establishes [n/4] necessity.

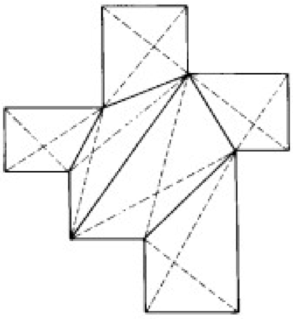


Figure 1.8 A 4-colorable graph derived from a quadrilateralization by adding all quadrilateral diagonals.

- Since the quadrilaterals cover P and are convex placing guards at the vertices assigned the least frequently used color will cover the interior of P. As this color must be used no more than *[n/4]* times, the theorem is established. □

Note that the quadrilaterals clipped in this proof are “orthogonal ears”; thus every orthogonal polygon has at least two such ears.

1.2.4 Orthogonal polygons with h holes

- Shermer's Conjecture I (1982): Any orthogonal polygon with n vertices and h holes can always be guarded by $\left\lfloor \frac{n+h}{4} \right\rfloor$ vertex guards.

  Lemma 1 : For any quadrilateralization Q of an orthogonal polygon with one hole, the G_Q^* graph is 4-colorable, although sometimes we have to split a vertex into two in order to get a legal 4-coloring.

  Proof of Shermer's Conjecture I:

- Let us first consider the dual graph of the quadrilateralization of the polygon from Fig. 1.9. It consists of four cycles connected by single edges, as illustrated in Fig. 1.10. Cutting along three diagonals of the quadrilateralization results in four one-hole polygons with quadrilateralizations Q_i and their quadrilateralization graphs G_{Q_i} , i=1,2,3,4, respectively. These polygons are usually not orthogonal, but it is easy to see that for each i=1,2,3,4, there exists a one-hole orthogonal polygon P^i with a quadrilateralization graph G_Q^i isomorphic to graph G_{Q_i} .Thus by Lemma 1, all graphs G_{Q_i} , i=1,2,3,4, are 4-colorable, after splitting at most four vertices in total. Now, we can reunite the polygons, thus getting a 4-coloring of graph G_Q with at most four splits (we can recolor graphs G_{Q_i} , i=1,2,3,4, if necessary). The least frequently used color will be used not more than $\left\lfloor \frac{n+h}{4} \right\rfloor$ times.

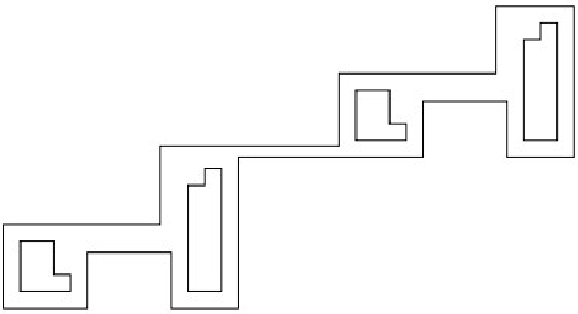


Figure 1.9 An orthogonal polygon with holes that requires $\lfloor \frac{n+h}{4} \rfloor$ vertex guards; here $n = 44$, $h = 4$ and the polygon requires 12 vertex guards. When not limited to vertices, 11 guards are sufficient.

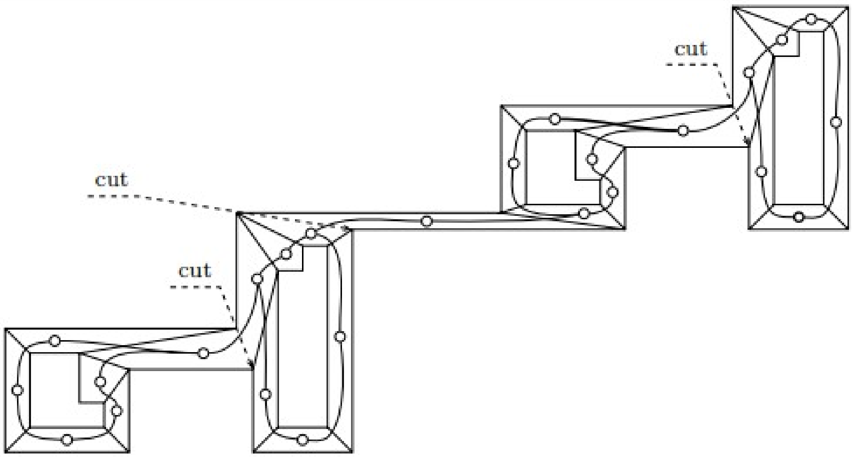


Figure 1.10 The dual graph of a quadrilateralization of the polygon from *Figure 1.9*

- The same argument can be applied to any orthogonal polygon with h holes, but only if the dual graph consists of h cycles connected by trees: cut the polygon along h − 1 diagonals of the quadrilateralization to get h “orthogonal” polygons with one hole, 4-color all graphs G_{Q_i} , i=1,2,…,h, and reunite the polygons. Placing guards at the vertices assigned to the least frequently used color (at most $\left\lfloor \frac{n+h}{4} \right\rfloor$ times) will cover the interior of P. This is the case when no two cycles in the dual graph have a vertex in common.
- Now, let us consider the dual graph of the quadrilateralization of the polygon in Fig. 1.11. It has two cycles with only one vertex v in common. We can also apply Lemma 1 to this case: we have to split the graph and duplicate vertex v, thus getting quadrilateralization graphs G_{Q_1} and G_{Q_2} , respectively. We can obtain a 4-coloring of the graph G_Q from 4-colorings of graphs G_{Q_1} and G_{Q_2} , and the least frequently used color must be used at most $\left\lfloor \frac{n+2}{4} \right\rfloor$ times.
- Furthermore, we can extend this observation to any number of holes only if any two cycles of the dual graph have only one vertex in common, and any such vertex is a cut-vertex. More precisely, suppose that the dual graph of the quadrilateralization of an orthogonal polygon consists of h cycles, any two of them having only one vertex in common, and any such vertex is a cut-vertex — let $v_1,...,v_{h-1}$ be these cut-vertices. We have to split the dual graph and duplicate vertices $v_1,...,v_{h-1}$, thus getting h cycles and their corresponding graphs G_{Q_1} ,…, $G_{Q_{h-1}}$, each of them isomorphic to a quadrilateralization graph of a one-hole orthogonal polygon, and each of them 4- colorable after splitting at most one vertex in each graph (by Lemma 1).

- Now, we can obtain a 4-coloring of graph G_Q by reuniting and possibly recoloring 4-colorings of graphs G_Q i = 1,...,h (compare Fig 1.11). As we have split at most h vertices in total, the least frequently used color is used at most $\left\lfloor \frac{n+h}{4} \right\rfloor$ times.

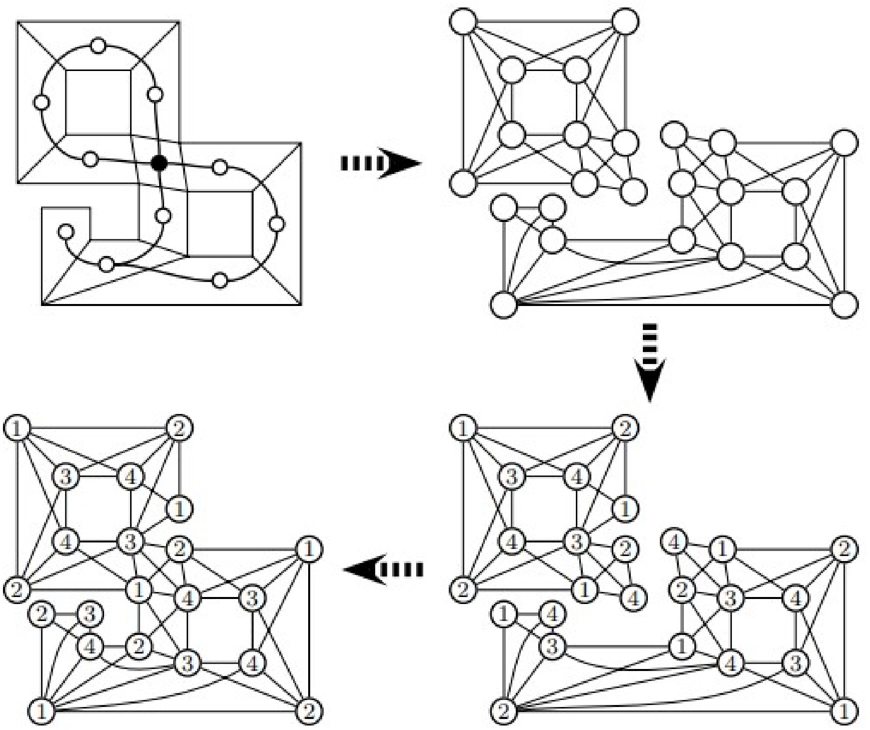


Figure 1.11 Two cycles of the dual graph have only one vertex in common.

Hence by combining all the above observations, the thesis of Conjecture I follows.

1.2.5 Polygons with h holes

J.Sidarto discovered the one-hole polygon shown in Figure 1.12.a. It has n=8 vertices, h=1 hole, and requires 3 guards. Note that 3> $\left\lfloor \frac{8}{3} \right\rfloor$. Shermer discovered the polygons in Figures 1.12.b and 1.12.c, which also have 8 vertices and requires 3 guards.

These one-hole examples can be extended to establish $\left\lfloor \frac{n+1}{3} \right\rfloor$ necessity for one hole: Figures 1.13a and 1.13.b show two examples for n=11, due, respectively, to Shermer and Delcher.

Finally, the examples can be extended to more than one hole: Figure 1.14 shows Shermer's method of stiching together copies of the basic one-hole example. The polygon shown has n=24 vertices, h=3 holes, and requires 9 guards. This example establishes $\left\lfloor \frac{n+h}{3} \right\rfloor$ necessity for h holes.

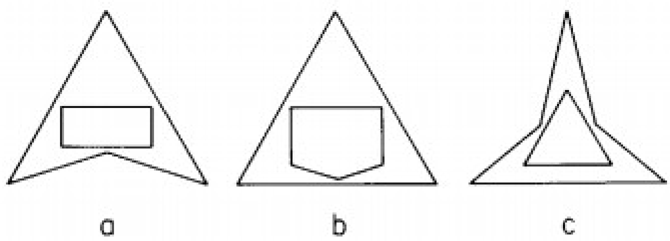


Figure 1.12 One hole polygons of 8 vertices that require 3 guards

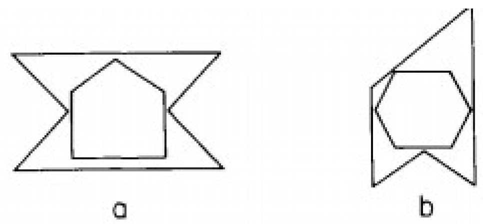


Figure 1.13 One-hole polygon of 11 vertices that require 4 guards

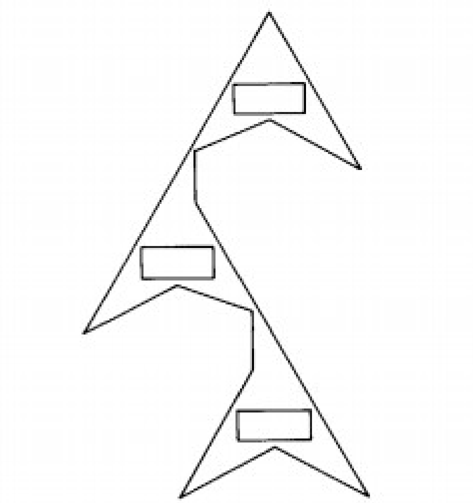


Figure 1.14 A polygon of 24 vertices with 3 holes that requires 9 guards

The following theorem summarizes the implications of these examples:

- Shermer's Conjecture II (1982): Any polygon with n vertices and h holes can be guarded by $\left\lfloor \frac{n+h}{3} \right\rfloor$ point guards.

2. Visibility graphs

2.1 General

Whereas art gallery theorems seek to encapsulate an environment's visibility into one function of n, the study of visibility graphs endeavors to uncover the more finegrained structure of visibility. The original impetus for their investigation came from pattern recognition, and its connection to shape continues to be one of its primary sources of motivation.

- *Visibility graph*: A graph with a node for each object, and arcs between objects that can see one another.
- *Vertex visibility graph*: The objects are the vertices of a simple polygon.
- *Endpoint visibility graph*: The objects are the endpoints of line segments in the plane.
- *Segment visibility graph*: The objects are whole line segments in the plane, either open or closed.
- *Object visibility*: Two objects A and B are visible to one another if there are points x $x \in A$ and $y \in B$ such that x sees y.
- *Point visibility*: Two points x and y can see one another if the segment xy is not "obstructed", where the meaning of "obstruction" depends on the problem.
- *Hamiltonian*: A graph is Hamiltonian if there is a simple cycle that includes every node.

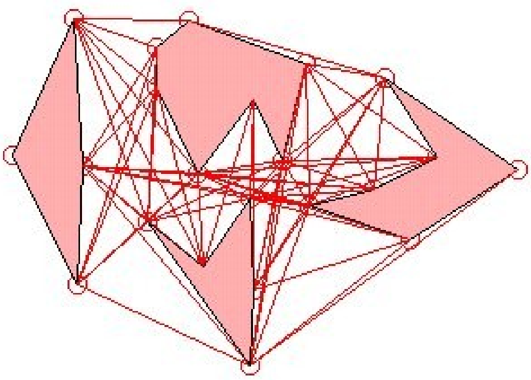


Figure 2.1 Example of a visibility graph

2.2 Obstructions to visibility

For polygon vertices, x sees y if xy is nowhere exterior to polygon, just as in art gallery visibility, this implies that polygon edges are part of the visibility graph. For segment endpoints x sees y if the closed segment xy intersects the union of all the segments either in just the two endpoints, or in the entire closed segment. This disallows grazing contact with a segment, but includes the segments themselves in the graph.

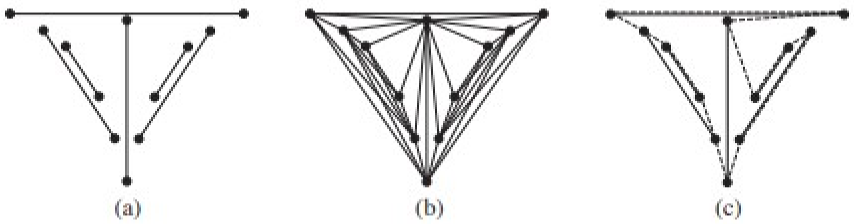


Figure 2.2
(a) A set of 6 pairwise disjoint line segments (b) Their endpoint visibility graph G.
(c) A Hamiltionian cycle in G.

2.3 Goals

Four goals can be discerned in research on visibility graphs:

1. *Characterization*: asks for a precise delimiting of the class of graphs realizable by a certain class of geometric objects.
2. *Recognition*: asks for an algorithm to recognize when a graph is visibility graph
3. *Reconstruction*: asks for an algorithm that will take a visibility graph as input, and output a geometric realization.
4. *Counting*: concerned with the number of visibility graphs under various restrictions.

2.4 Types of visibility graphs

- Point Visibility graphs
- Vertex Visibility graphs
- Endpoint Visibility graphs
- Segment Visibility graphs
- Invisibility graphs
- Obstacle Visibility graphs

2.4.1 Point Visibility graphs

Given a set P of n points in the plane, visibility between $x, y \in P$ may be blocked by a third point in P. The recognition of point visibility graph is NP-hard. However, for planar graphs, there is a complete characterization, and an O(n)-time recognition algorithm. Phender constructed point visibility graphs of clique number 6 and arbitary high chromatic number.

2.4.2 Vertex Visibility graphs

A complete characterization of vertex visibility graphs of polygons has remained elusive, but progress has been made by:

1. Restricting the class of polygons: polynomial-time recognition and reconstruction algorithms for orthogonal staircase polygons has been obtained. (Fig. 2.3)
2. Restricting the class of graphs: every 3-connected vertex visibility graph has a 3-clique ordering, i.e., an ordering of the vertices so that each vertex is part of a triangle composed of preceding vertices.
3. Adding information: assuming knowledge of the boundary Hamiltionian circuit, four necessary conditions have been established by Ghosh and others, and conjectured to be sufficient.

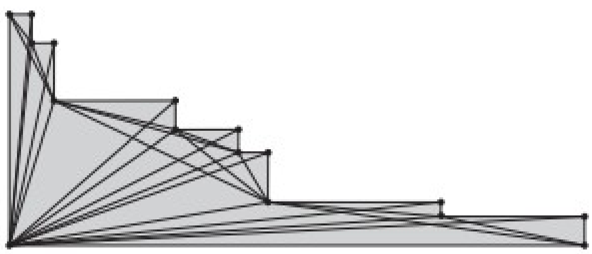


Figure 2.3
A staircase polygon and its vertex visibility graph

2.4.3 Endpoint Visibility graphs

A set of n pairwise disjoint line segments forms a noncrossing perfect matching on the 2n endpoints in the plane. For segment endpoint visibility graphs, there have been three foci:

1. There is always a Hamiltionian polygon for pairwise disjoint line segments, not all lying on a line.
2. In the quest for generating a random noncrossing perfect matching, Aichholzer conjecture that any two such matchings are connected by sequence of noncrossing perfect matchings in which consecutive matching are compatible (the union of the two marchings is also noncrossing). Every matching of 4n vertices is known to have a compatible matching.
3. Size questions: there must be at least $5n - 4$ edges and at least $6n - 6$ when no segment is a "chord" splitting the convex hull; the smallest clique cover has size Ω(n^2 / log^2(n)).

2.4.4 Segment Visibility graphs

Whole segment visibility graphs have been investigated most thoroughly under the restriction that the segments are all (say) vertical and visibility is horizontal. Such segments are often called bars. The visibility is usually required to be ε-visibility. Endpoints on the same horizontal line often play an important role here, as does the distinction between closed segments and intervals (which may or may not include their endpoints). There are several characterizations:

1. G is representable by segments, with no two endpoints on the same horizontal line, iff there is a planar embedding of G such that, for every interior k-face F , the induced subgraph of F has exactly $2k - 3$ edges.
2. G is representable by segments, with endpoints on the same horizontal permitted, iff there is a planar embedding of G with all cutpoints on the exterior face.
3. Every 3-connected planar graph is representable by intervals.

2.4.5 Obstacle Visibility graphs

An interesting variant of visibility graphs has drawn considerable attention. Given a graph G, an obstacle representation of G is a mapping of its nodes to the plane such that edge (x, y) is in G if and only if the segment xy does not intersect any “obstacle.” An obstacle is any connected subset of R^2 .The obstacle number of G is the minimum number of obstacles in an obstacle representation of G. At least one obstacle is needed to represent any graph other than the complete graph.

When the obstacles are points and G is the empty graph on n vertices, this quantity is known as the blocking number b(n).

2.4.6 Invisibility graphs

For a set X ⊆ R^d , its invisibility graph L(X) has a vertex for each point in X, and an edge between two vertices u and v if the segment uv is not completely contained in X. The chromatic number χ(X) and clique number ω(X) of L(X) have been studied, primarily in the context of the covering number, the fewest convex sets whose union is X. It is clear that ω(X) ≤ χ(X), and it was conjectured in [MV99] that for planar sets X, there is no upper bound on χ as a function of ω.

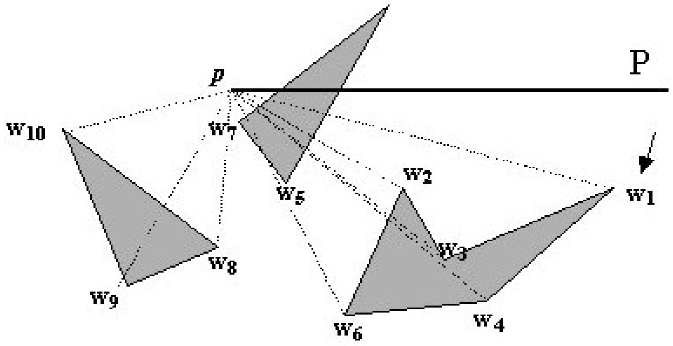
2.5 Open problems

1. Given a visibility graph G and a Hamiltonian circuit C, construct in polynomial time a simple polygon such that its vertex visibility graph is G, with C corresponding to the polygon’s boundary.
2. Given a visibility graph G of a simple polygon P, find the Hamiltionian cycle that corresponds to the boundary of P.
3. Develop an algorithm to recognize whether a polygon vertex visibility graph is planar. Necessary and sufficient conditions are known.

2.6 An algorithm for Computing Visibility Graphs

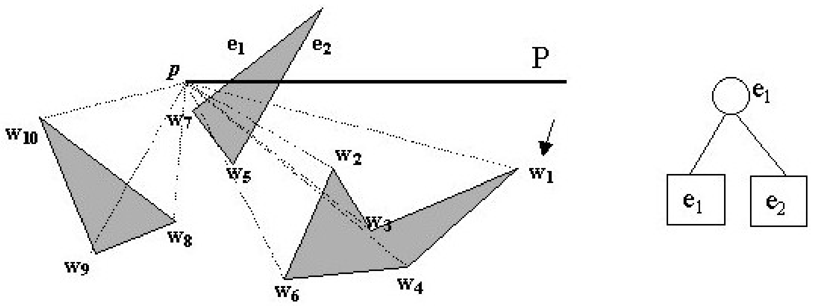
Our naïve algorithm checks for intersection for all pairs of vertices against all obstacles edges. However, there will be economy if we check for intersection in a certain order. In this algorithm, we check if other vertices are visible to a vertex by their cyclic order around that vertex. The best way to illustrate this algorithm will be to give an example of the execution for a certain vertex.

- **Step 1**: For a certain point p, sort all obstacle vertices according to he clockwise angle that the half-line from p to each obstacle vertex makes with the positive x-axis.



Let the sorted list be w_1, …, w_n. This is shown in the diagram above. This step takes O(nlogn).

- **Step 2**: Let P be the half-line parallel to the positive x-axis starting at p. Find the obstacle edges, e_i, that are properly intersected by P, and store them in a balanced search tree T (shown right below) in the order in which they are intersected by P. This step takes O(n).



- **<u>Step 3</u>**: for all vertices w_i where i = 1 to n

  if VISIBLE (w_i) **then Add** w_i to the list of visibility edges
  Insert into T the obstacle edges incident to w_i that lie on the clockwise side of the half-line from p to w_i

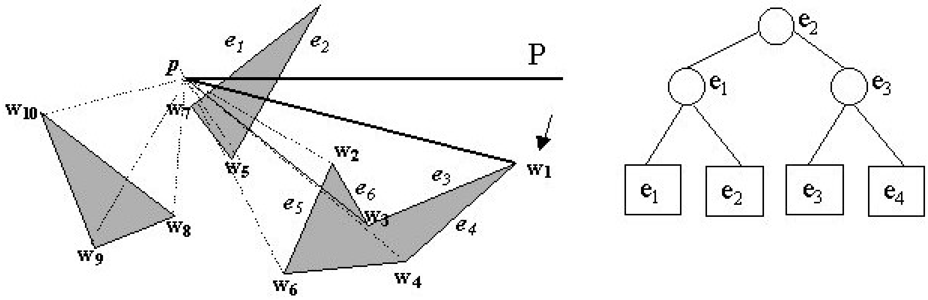
  Delete from T the obstacle edges incident to w_i on the counterclockwise side of the half-line from p to w_i

VISIBLE(wi)

1. **if** *pwi* intersects the interior of the obstacle of which w_i is a vertex **then return false**
2. **else if** i = 1 or w_{i-1} is not on the segment pw_i
3. **then** Search in T for the edge e in the leftmost left (the edge p intersects first)
4. **if** e exists and *pwi* intersects e *then return false*
5. **else return true**
6. **else if** w_{i-1} is not visible
7. **then return false**
8. **else** Search in T for an edge e that intersects $w_{i-1}w$
9. **if** e exists *then return false*
10. **else return true**

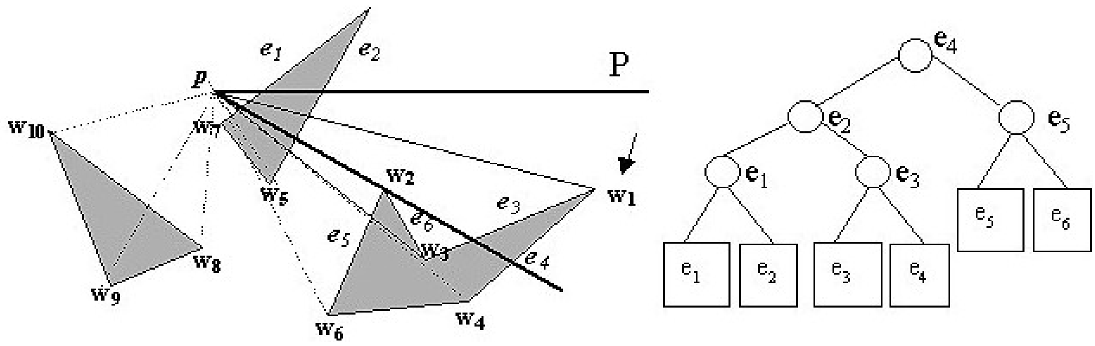
(the above pseudocode is taken directly from *M. deBerg et al, Computational Geometry: algorithms and applications, 1997.*)

The consequences of this step on w_1 is that lines 2-4 of VISIBLE gets executed because i = 1 and e_i in T intersects pw_1. w_1 is not visible from p. This step takes $O(\log n)$ time because of search on the binary tree.



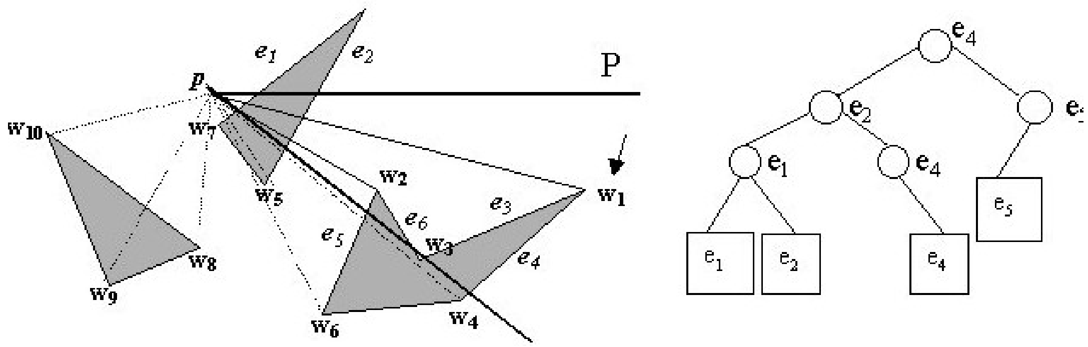
Moreover, edges e_3 and e_4 , which are incident to w_1 and lie on the clockwise side of the half-line from p to w_1, are added to T. Note that the edges now stored in T are the edges that will possibly intersect pw_2.

We now turn to w_2. The consequences of this step on w_2 is that lines 2-4 gets executed because w_1 is not on the segment pw_2, and that pw_2 intersects e_1. w_2 is not visible from p.



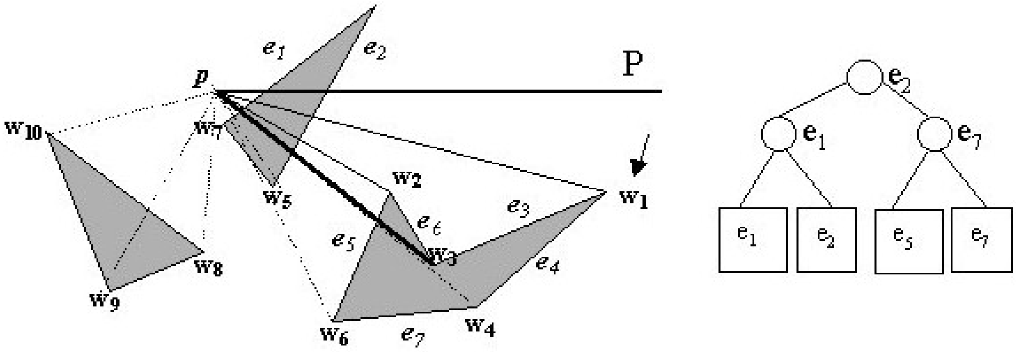
Edges e_5 and e_6, which are incident to w_2 and lie on the clockwise side of the half-line from p to w_2, are added to T. Note that the edges now stored in T are the edges that will possibly intersect pw_3.

At w_3, something interesting happens. e_6 and e_3, which are incident to w_3 on the counterclockwise side of the half-line from p to w_3, are deleted from T, the binary tree.



This is a crucial step. The deletion of e_6 and e_3 from the tree means that they will not be considered for intersection anymore in subsequent searches. The edges stored in T during the run of pw_{i-1} are the edges that will possibly interesect pw_i. This deletion, together with the fact that the possibly intersecting edges are stored in a binary search tree, gives us reduction in time complexity.

Similarly, at w_4, e_4 is deleted from T while e_7 is inserted into T.



This algorithm goes on until all the run for w_{10} has finished. This algorithm allows the storage of the obstacle edges that will possibly intersect the next pw_i pair. It therefore does not search for all obstacle edges unnecessarily. Moreover, these edges are stored in a binary search tree, so even if all obstacle edges are potential candidates, searching for one that does intersect will only cost $O(\log n)$. Lines 6-10 of VISIBLE deals with special cases when w_{i-1} is on the segment *pwi*, that is, p, w_{i-1} and w_i are collinear.

The most time-consuming step in the run for each vertex p is the sorting of obstacle vertices around p by angularity, which takes $O(n \log n)$. Since we repeat all this process for each of n vertex of the set of obstacles, the overall runtime is $O(n^2 \log n)$.

2.6 Visibility in Three Dimensions

2.6.1 Research Topics

Research on visibility in three dimensions (3D) has concentrated on three topics:

- Hidden Surface Removal
  - Key problem in computer graphics
  - The typical problem instance is a collection of (planar) polygons in space, from which the view from z= ∞ must be constructed
  - Classified algorithms as either image-space or object-space algorithms
    - Image-space: compute visible colors for image pixels
    - Object-space: exact computations on object polygons
- Polyhedral Terrains:
  - A polyhedral surface that intersects every vertical line in at most a single point
  - Special class of 3D surfaces, arising in a variety of applications, most notably geographic information systems
- Various 3D Visibility Graphs
  - Three primary motivations for studying 3D Visibility Graphs of Objects
    1. Computer graphics
    2. Computer vision
    3. Combinatorics

2.6.2 Hidden-surface removal

The complexity of output scene can be quadratic in the number of input vertices n. A worst-case optimal $\Theta(n^2)$ algorithm can be achieved by projecting the lines containing each polygon edge to a plane and constructing the resulting arrangement of lines [Dév86, McK87].

More recent work has focused on obtaining output-size sensitive algorithms, whose time complexity depends on the number of vertices k in the output scene (the complexity of the visibility map), which is often less than quadratic in n. In the table below, k is the complexity of the visibility map, the “wire-frame” projection of the scene.

ENVIRONMENT	COMPLEXITY	SOURCE
Isothetic rectangles	$O((n+k)\log n)$	[BO92]
Polyhedral terrain	$O((n+k)\log n \log\log n)$	[RS88]
Nonintersecting polyhedra	$O(n\sqrt{k}\log n)$	[SO92]
	$O(n^{1+\epsilon}\sqrt{k})$	[BHO+94]
	$O(n^{2/3+\epsilon}k^{2/3} + n^{1+\epsilon})$	[AM93]
Arbitrary intersecting spheres	$O(n^{2+\epsilon})$	[AS00]
Nonintersecting spheres	$O(k + n^{3/2}\log n)$	[SO92]
Restricted-intersecting spheres	$O((n+k)\log^2 n)$	[KOS92]

Figure 2.4 Hidden-surface algorithms complexities

2.6.3 Binary Space Partition Trees

Binary Space Partition (BSP) trees are a popular method of implementing the basic painter’s algorithm, which displays objects back-to-front to obtain proper occlusion of front-most surfaces. A BSP partitions R^d into empty, open convex sets by hyperplanes in a recursive fashion. A BSP for a set S of n line segments in R^2 is a partition such that all the open regions corresponding to leaf nodes of the tree are empty of points from S: all the segments in S lie along the boundaries of the regions.

In general, a BSP for S will “cut up” the segments in S, in the sense that a particular $s \in S$ will not lie in the boundary of a single leaf region. In the figure, partitions 1 and 2 both cut segments, but partition 3 does not.

```
DrawTree(BSPtree)
{
    if (eye is in front of root)
    {
        DrawTree(BSPtree->behind)
        DrawPoly(BSPtree->root)
        DrawTree(BSPtree->front)
    } else {
        DrawTree(BSPtree->front)
        DrawPoly(BSPtree->root)
        DrawTree(BSPtree->behind)
    }
}
```

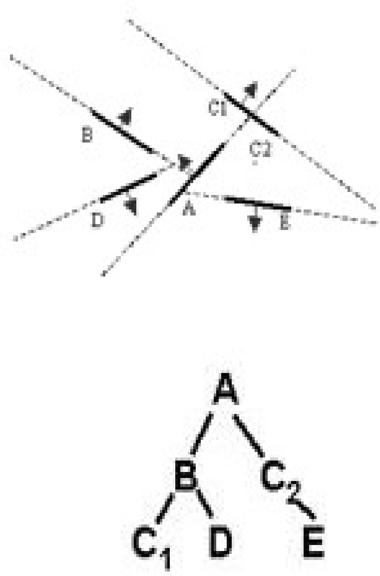


Figure 2.5 BSP draw recursively for segments

DIM	CLASS	BOUND	SOURCE
2	segments	$O(n \log n)$	[PY90]
2	isothetic	$\Theta(n)$	[PY92]
2	fat	$\Theta(n)$	[BGO97]
2	segments	$\Theta(n \log n / \log \log n)$	[Tót03a, Tót11]
3	polyhedra	$O(n^2)$	[PY90]
3	polyhedra	$\Omega(n^2)$	[Cha84]
3	isothetic	$\Theta(n^{3/2})$	[PY92]
3	fat orthog. rects.	$O(n \log^8 n)$	[Tót08]

Figure 2.6 BSP complexities

Bibliography