Optimum Guard Covers and m-Watchmen Routes for Restricted Polygons*

Svante Carlsson[†] Bengt J. Nilsson[†‡] Simeon Ntafos[§]

Abstract

A watchman, in the terminology of art galleries, is a mobile guard. We consider several watchman and guard problems for different classes of polygons. We introduce the notion of vision spans along a path (route) which provide a natural connection between the Art Gallery problem, the m-watchmen problem and the watchman route problem. We prove that finding the minimum number of vision points, i.e., static guards, along a shortest watchman route is NP-hard. We provide a linear time algorithm to compute the best set of static guards in a histogram polygon. The m-watchmen problem, minimize total length of routes for m watchmen, is NP-hard for simple polygons. We give a $\Theta(n^3 + n^2m^2)$-time algorithm to compute the best set of m watchmen in a histogram.

1 Introduction

The problem of placing guards in an art gallery so that every point in the gallery is visible to at least one guard has been considered by several researchers. If the gallery is represented by a polygon, having n vertices, and the guards are points in the polygon, then visibility problems can be equivalently stated as problems of covering the gallery with starshaped polygons. Chvàtal [5] and Fisk [8] proved that $\lfloor n/3 \rfloor$ guards are always sufficient and sometimes necessary to guard a polygonal art gallery. Many other results on art gallery problems and general visibility problems can be found in [12].

Chin and Ntafos [3, 4] consider the problem of finding shortest watchman routes for different classes of polygons. The route is the path traced by a moving guard who in total sees the whole polygon. They prove that the problem is NP-hard for polygons with holes and for 3-dimensional simple polyhedra. They also give a $\Theta(n)$-time algorithm that solves the problem for simple rectilinear polygons. In addition they provide a solution to the slightly different question of finding a shortest watchman route given a "door", a point on the boundary that the route must pass through. They give an $O(n^4)$-time algorithm that solves this problem for simple polygons without holes. Recently Tan, Hirata, and Inagaki [14] improved this bound to $O(n^3)$. Both these bounds rely on the linear time triangulation algorithm by Chazelle [2].

The stationary art gallery problem and the watchman route problem can be thought of as special cases of the m-watchmen problem. Suppose that we have m mobile watchmen and we want to design routes for them so that each point in a polygon is visible to some point along a route. The objective is to minimize the total length of the m routes. Clearly, the total length of an optimum solution increases as the number of watchmen decreases and vice versa. At one

*The work of the second author was supported by the Deutsche Forschungsgemeinschaft under Grant No. Ot 64/5–4.

†Department of Computer Science, Lund University, Box 118, 221 00 Lund, Sweden.

‡Institut für Informatik, Universität Freiburg, Rheinstr. 10–12, D-7800 Freiburg, Fed. Rep. of Germany.

§Computer Science Program, University of Texas at Dallas, Richardson, TX 75083–0688, U.S.A.

limit, we have a situation where m is large enough so that the length of the optimum solution is zero, i.e., each watchman remains stationary, and we have the art gallery problem. At the other limit, if $m = 1$, we have the watchman route problem.

Another connection between static and mobile guard problems is obtained by considering a single mobile robot equipped with a vision system. We can think of the robot as performing two types of activity, motion and visual data acquisition. To minimize the distance traveled so that it sees all of the polygon, the robot traces a shortest watchman route. However, the robot's vision system does not need to be active throughout the route. Also, it is likely that the vision system performs better if the robot is not moving. We refer to points on the route at which the robot engages in vision activity as *vision points.* A continuous portion of the route where the robot engages in vision activity is referred to as a *vision span.* Then we have the art gallery problem, minimize the number of vision points, with the restriction that the solution must lie along a path, e.g., the shortest watchman route.

In this paper we show that finding the best set of vision points along a path is NP-hard for simple polygons. Also, the m-watchmen problem for simple polygons is NP-hard. In view of these results, we consider the same problems for restricted classes of polygons. Nilsson and Wood [11] treat the class of spiral polygons. They give a linear time algorithm for the minimum stationary guard problem and a $\Theta(n^2)$ time algorithm for the multiple watchmen routes problem. We consider the same problem for histograms where a (vertical) *histogram* is a rectilinear polygon with a horizontal edge, the base, equal in length to the sum of the lengths of all the other horizontal edges. An interesting property of histograms is that the m-watchmen problem and the m-vision span problem, along a shortest watchman route, have a common optimum solution. Histograms are the most complex of the restricted polygons that have been studied in the literature to have this property.

In the next section we show that finding the best set of vision points on a path is NP-hard for simple polygons. In Section 3 we show that a minimum guard cover for histogram polygons can be found in linear time. In Section 4 we develop the multiple watchmen routes idea on histograms and prove that an optimum set of watchmen lies on the base of the histogram. This enables us to solve the m-watchmen problem with dynamic programming in $\Theta(n^3+n^2m^2)$ time.

2 Minimum Guard Covers on Paths

In this section we study the problem of positioning stationary guards in simple polygons. We look at guards that are restricted to lie on a given curve, the *watchman route*. These guarding points are called the *vision points* of the route, and we prove that, in general, it is NP-hard to find the minimum number of vision points on a shortest watchman route.

Definition 2.1 *A vision point is a point along a given path in which the watchman will engage in visual data acquisition activity.*

We take the *watchman route problem* to be the problem where a starting point on the boundary is specified. We refer to the watchman route problem when no starting point is specified as the *floating watchman route problem*. We have the following result.

Theorem 1 *Finding the minimum number of vision points along a shortest floating watchman route in a simple polygon is NP-hard.*

Proof: The proof is by reduction from 3-SAT. Let a 3-SAT instance have k clauses and n variables. We modify the proof of Lee and Lin [9] and Aggarwal [1] that finding a minimum

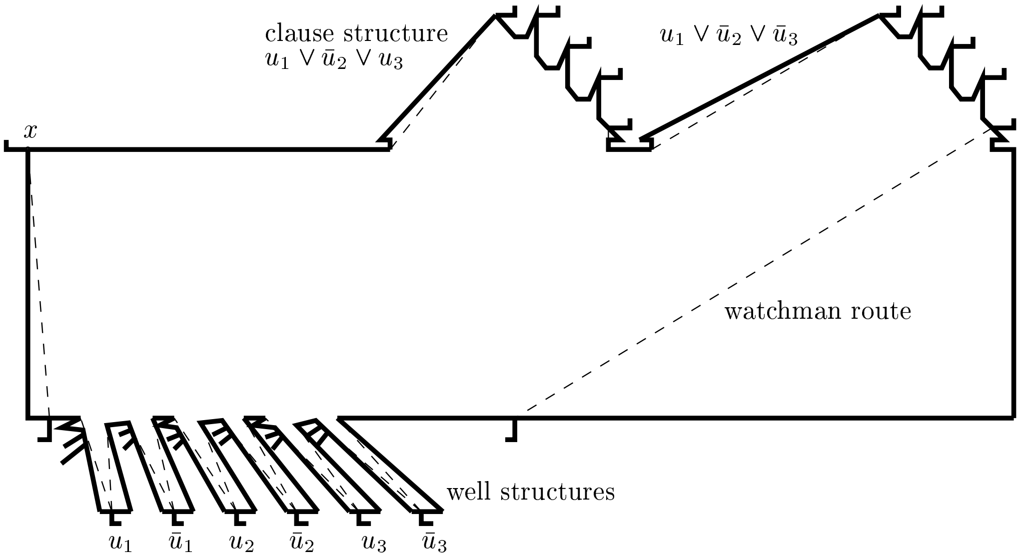


Figure 1: Five caves are added to each clause structure, one cave to each well structure, and three caves more, at x, to the left of the well sequence, and to the right of it. The corresponding 3-SAT formula for this example is $(u_1 \vee \bar{u}_2 \vee u_3) \wedge (u_1 \vee \bar{u}_2 \vee \bar{u}_3)$.

guard cover for simple polygons is NP-hard. We refer to [9] or [12] for the complete construction of the proof and show here only the modifications that have to be made.

The proof of Lee and Lin specifies $6k + 2n + 1$ distinguished points such that a minimum vertex guard cover uses $3k+n+1$ of them as guard locations and this is a minimum vertex guard cover if and only if the corresponding 3-SAT formula is satisfiable. We add a number of "caves" to the polygon in such a way that the shortest watchman route will be forced to visit all the distinguished points in the construction of Lee and Lin. The caves have two bends so that any watchman route must enter each cave, one vision point is needed in it and this vision point can not see too much outside the cave. We use one cave for each literal structure plus one cave on either side of a clause structure to force the shortest watchman route to visit the distinguished points in the clause structures giving $5k$ caves. We put one cave at the bottom of each well structure, $2n$ in total, one cave on either side of the well sequence, to tie the route down, and one cave at the upper left corner, to force the route to visit the point x. This gives a total of $5k+2n+3$ caves, each of which needs exactly one vision point. The shortest watchman route visits all the distinguished points so the polygon can be seen with $(3k + n + 1) + (5k + 2n + 3) = 8k + 3n + 4$ vision points if and only if the formula is satisfiable; see Figure 1 for an example.

Note that the vision points are not restricted to lie on vertices, either of the polygon or of the route, but this does not pose any problem, since the spikes of the well structures can only be seen from the distinguished points which are also isolated points of the route. □

Another question to look at is the following. Is there a relation between the minimum number of vision points and the minimum number of guards for a polygon? The answer is that, in general, there exists no such relation and examples can be easily constructed. Instead, we turn the question around and tighten the requirement: For what types of polygons is it true that there is always a shortest watchman route with the property that an optimum set of vision points along the route is also an optimum solution to the minimum guard cover problem for the polygon? We answer this question for some classes of polygons.

For convex and star polygons the answer to the question is trivial for the floating watchman route problem. The best route is a single point in the kernel. We have a single vision point which is also a solution to the minimum guard cover problem. However, for star polygons, the correspondence does not hold for the fixed watchman route problem. A counterexample is constructed by taking a star polygon where the watchman route does not intersect the kernel. In this case we need at least two vision points on the route while one guard is enough for the minimum guard cover problem. A similar point was made by Toussaint, who investigated diagonal guards in star polygons, as reported in [12].

Now consider monotone polygons. Here the correspondence between vision points and solutions to the minimum guard cover problem does not hold even for floating shortest watchman routes. For example, consider a monotone polygon in which the shortest route is a straight line connecting two limit lines. We can make the notches be such that two of them can be seen from a single point in the polygon but not from a watchman route. This also holds for rectilinear monotone polygons; see Figure 2.

For spiral polygons it can be proven that the first and last links of the floating shortest watchman route connect to the boundary chain consisting of reflex vertices (We disregard the special case when the shortest route consists of only one link). Hence, the end points of the route must be vision points and also every second vertex of the route, otherwise there is some edge on the chain of reflex vertices which is not seen by a vision point.

In the following we will look closer at a special class of polygons for which the vision point problem and the minimum guard cover problem are equivalent in the sense that the minimum guard cover lies on one, of many possible, shortest watchman routes.

3 Minimum Guard Covers of Histograms

The problem of finding minimum guard covers, or equivalently minimum covers of starshaped polygons, for simple polygons has been proven to be NP-hard by Lee and Lin [9] and Aggarwal [1]. The problem is still open for rectilinear polygons but is NP-hard when minimum covers with convex rectilinear polygons (rectangles) in rectilinear polygons are considered; see Culberson and Reckhow [6].

These results indicate that the problem of finding the best guard cover is NP-hard even for rectilinear polygons. Therefore one may ask the following question: For what classes of polygons are there efficient solutions to the optimum guard covering problem? For example, for starshaped polygons there is a linear time algorithm to find the kernel of the polygon; see [10]. For spiral polygons an algorithm exists which runs in linear time; see [11].

In this section we show that for histogram polygons there is a linear time algorithm to find the best guard set. We begin the section by defining this class of polygons.

Definition 3.1 *A histogram is a simple rectilinear polygon that is monotone with respect to the horizontal axis, and such that one of the monotone chains forming its boundary is a single straight line segment.*

We will henceforth assume that the lower monotone chain is a straight line segment, see Figure 3, and call this segment the *base* of the histogram. Edelsbrunner *et al* [7] use an equivalent definition and define a vertical histogram as an orthogonal polygon with a horizontal edge, the base, equal in length to the sum of the lengths of the other horizontal edges. The upper monotone chain will henceforth be refered to as the *top chain*.

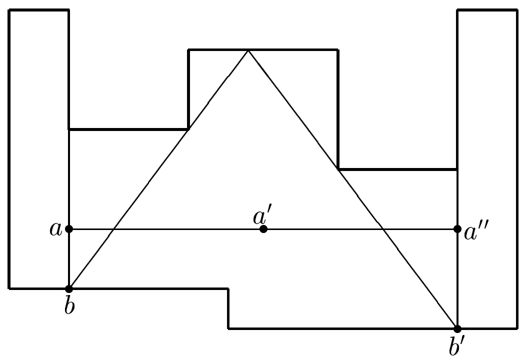


Figure 2: Optimum cover, the guards b and b', is not on any shortest watchman route contrary to the vision points a, a', and a''.

Note that the top chain consists of a sequence of pairs of staircase patterns, one up staircase followed by a down staircase. We call the part of the polygon under such a pair a *pyramid*. Sack [13] defines a pyramid to be horizontally convex vertical histogram. Horizontal convexity is the same as monotonicity along a vertical line. As shown in Figure 3 two consecutive pyramids have an overlapping rectangular region under the horizontal edge that starts the up staircase of one pyramid and ends the down staircase of the other.

The two staircases of a pyramid are connected by the *top edge*; see Figure 3. We will also use the term *west edge* to denote a vertical edge that has the interior to the right of the edge. An *east edge* is defined similarly as a vertical edge having the interior to the left. Note that a pyramid can be viewed as a starshaped polygon with the kernel under the top edge.

Consider the visibility polygon from an arbitrary point s lying on the base. The boundary of this visibility polygon consists of segments that are also segments of the histogram and segments that only intersect a discrete and bounded number of points (usually two) on the top chain. Consider the lines collinear with these last segments. These lines intersect the point s, at least one vertex on the top chain of the histogram, and some other point on the top chain in the order as we move through the interior of the histogram from s; see Figure 3. We call these lines *supporting lines*. The intersection of the supporting line and the histogram consists of a number of connected components, in most cases, however, only one. We are only interested in the component containing s and whenever we refer to the supporting line we will mean this component. Each supporting line is associated to the pyramid containing the segment end point of the supporting line that is not s. It is worth noticeing that not all pyramids need to have a supporting line from a point associated to it.

A supporting line divides a pyramid into one section which is visible from s and another section where there may exist points that are not seen from s, since the supporting line passes through a vertex.

First, we prove some simple properties of histograms. The following lemma provides the main basis for our algorith.

LEMMA 3.1 *Let s and t be two points on the base of a histogram, s being to the right of t, then s sees at least as much of the part of the polygon to the right of s as t does.*

PROOF: Take a point x interior to the polygon and to the right of s, such that x is seen by t. We must prove that s also sees x. Project x orthogonally onto the base giving point x'. the triangle having corners at t, x, and x' contains s and is convex. Hence, s sees x. □

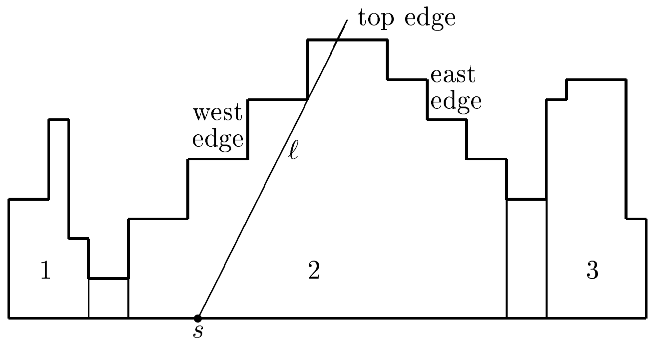


Figure 3: Histogram with three pyramids. ℓ is the supporting line of point s with respect to pyramid 2.

LEMMA 3.2 *For any histogram, a shortest floating watchman route and a minimum guard cover is positioned on the base of the histogram between the leftmost east edge and the rightmost west edge.*

PROOF: To see that a shortest floating watchman route lies on the base between the two edges, note that in order for the watchman to see both the leftmost and the rightmost pyramid it must intersect the kernels of these pyramids, i.e., cross the extensions of the leftmost east edge and the rightmost west edge. Also, since the two extended edges are parallel, all horizontal segments between the two extended edges have the same length, thus we can pick the segment along the base to be our shortest route.

To see that a minimum guard cover lies on the base, take any guard cover and project the guards orthogonally onto the base. This can be done since the polygon is monotone. Take a point x visible from the original guard g. Let g' be the projection of the guard g. Project the point x onto the base giving point x'. The four points g', g, x, and x' define a convex area, actually a trapezoidal area, and hence, x is seen by g'.

Any guard to the left of the leftmost east edge or to the right of the rightmost west edge can be moved, without affecting visibility, onto the intersection points of the extensions of the two edges and the base since they are still in the kernels of the "outermost" pyramids. □

Since an optimal set of guards lies on a shortest floating watchman route, the guarding points are also vision points on the route.

We state an algorithm to find a minimum guard cover of a histogram. The algorithm uses a greedy approach, walking from left to right. The first guard is placed on the base under the leftmost east edge. The algorithm then tries to place the next guard as far to the right of the previous one as possible.

We define the function $\mathit{partner}(s, \triangle)$ of a point s on the base and a pyramid $\triangle$ to be the furthest point from s, on the base, where s and the partner collectively see all of $\triangle$. If s already sees all of $\triangle$ we let the partner be undefined.

The following lemma shows how to compute the partner of a given point with respect to some pyramid.

LEMMA 3.3 *Let s be a guarding point not under the top edge of a pyramid $\triangle$. One of the following holds:*

1. *The supporting line of $\mathit{partner}(s, \triangle)$ w.r.t. $\triangle$ meets the supporting line of s on the top edge of $\triangle$ or*

2. *the point partner*$(s, \triangle)$ *lies under the top edge of* $\triangle$.

Proof: Assume w.l.o.g. that s is to the left of the top edge of $\triangle$.

We need to prove that if 2. does not hold then 1. holds. So, assume that *partner*$(s, \triangle)$ does not lie under the top edge of $\triangle$.

Suppose at most one of the supporting lines intersects the top edge, then one of s and *partner*$(s, \triangle)$ does not see any part of the top edge. Thus, the other guarding point, in order to see all of the top edge, must lie below the top edge. This is a contradiction to the assumption that neither s nor *partner*$(s, \triangle)$ lies below the top edge.

Suppose the supporting lines both intersect the top edge but at different points, and the two lines intersect outside the polygon, then the segment of the top edge between the intersection points is visible from both s and the partner so we can move the partner and increase the distance to s. We have a contradiction to the assumption that the partner is the furthest point from s retaining visibility.[1]

Suppose finally that the supporting lines intersect the top edge and each other but inside the histogram, then the segment of the top edge between the intersection points is not seen from any of the points. Again we have a contradiction to the visibility assumption. □

Given a point s on the base and a pyramid $\triangle$ the supporting line can be computed easily. First determine whether s lies to the left or right of the top edge of $\triangle$. Assume it lies to the left of $\triangle$, then follow the left staircase down, maintaining the interior line through s and the vertex while remembering the line having minimum slope.

Once the supporting line is given the point *partner*$(s, \triangle)$ can be computed by taking the intersection point of the supporting line of s and the top edge of $\triangle$, then by following the right staircase down, maintaining the interior line through the intersection point and the vertex of the staircase. The line having minimum absolute slope is the supporting line of the partner, and hence, the partner is the intersection of this supporting line with the base. If the supporting line of s does not intersect the top edge of $\triangle$, the partner is the point on the base under the leftmost east edge of $\triangle$.

If s lies to the right of $\triangle$, a similar scheme can be used to compute the supporting line and the partner.

We define *rlp*(s), the *limit point to the right of* s to be the closest partner to the right of s. More formally

$$rlp(s) = \min_{\triangle_i \succeq s} (partner(s, \triangle_i)), \tag{1}$$

where the value of a point is its distance from the leftmost edge of the histogram; $\triangle_i$, $1 \leq i \leq k$, are the pyramids of the histogram; and $\succeq$ means "not to the left of." Similarly we define the limit point to the left of s, denoted *llp*(s).

Next we show that the limit point to the right of a given point s is always the furthest point on the base which retains visibility between the two points.

Lemma 3.4 *Let s be a point on the base of a histogram. There does not exist any point to the right of rlp(s), on the base, which together with s sees all of the top monotone chain between the two points.*

Proof: By the definition of limit point we know that *rlp*(s) is the partner of s with respect to some pyramid $\triangle$. Thus, for any point v to the right of *rlp*(s) the supporting lines of s and v, with respect to $\triangle$, will intersect the boundary of the polygon at distinct points and the portion of the boundary between the intersection points is neither visible from s nor from v. □

[1] By retaining visibility we mean that all of the histogram between a point and its limit point is visible from the two points.

The right limit point of s can be computed by scanning the pyramids having their top edges to the right of s. This is done from left to right. For each pyramid the partner to s is computed and the leftmost partner found so far is remembered. The process stops when the top edge of the next pyramid to be scanned is to the right of the leftmost partner found, since all the partner points of the pyramids to the right of this point have to be to the right of the leftmost partner.

This gives us the following algorithm for the minimum cover.

Algorithm	*Histogram-Minimum-Cover*
Input:	A histogram polygon P
Output:	A minimum guard cover for P
Step 1	Divide P into its pyramids, $\triangle_1, \triangle_2, \ldots, \triangle_k$
Step 2	Place the guard g_1 at the intersection of the base and the extension of the leftmost east edge of P
Step 3	**while** not all pyramids are guarded /* g_i was the last guard to be placed */ **do**
Step 3.1	$g_{i+1} := rlp(g_i)$
Step 3.2	$i := i + 1$
	endwhile
End	*Histogram-Minimum-Cover*

Suppose we have two sets of guards that each cover a histogram. Index the guards in the two sets from left to right as $G = \{g_1, g_2, \ldots\}$ and $F = \{f_1, f_2, \ldots\}$. Define the $<$ relation between two guards as

$$a < b \text{ if and only if } \begin{cases} \text{guard } a \text{ is to the left of guard } b \\ a \text{ and } b \text{ coincide and } a \text{ has smaller index} \end{cases}$$

THEOREM 2 *The algorithm Histogram-Minimum-Cover finds an optimum guard cover for a histogram polygon in linear time.*

PROOF: The subdivision into pyramids takes linear time since the histogram is monotone. Each pyramid is then scanned only once when the partner of the previously placed guard is computed. Thus, the algorithm above uses $\Theta(n)$ time.

To see that there does not exist a better guard cover, assume the contradiction. Suppose there is a guard cover having fewer guards. Let G be the solution obtained by our algorithm and F the solution, having less guards, that most closely matches G going left to right. Find the first position where the solutions differ, i.e., $g_i = f_i$ but $g_{i+1} \neq f_{i+1}$. If $f_{i+1} < g_{i+1}$, then f_{i+1} can be moved to the right limit point of f_i, $rlp(f_i) = rlp(g_i) = g_{i+1}$, without affecting visibility. This contradicts our assumption that F matched G the best. On the other hand, $g_{i+1} < f_{i+1}$ is not possible because that implies that F is not a guard cover since $g_{i+1} = rlp(g_i)$, and by Lemma 3.4 this is the rightmost point which together with g_i sees all of the top monotone chain between the two guards. □

An example of an optimum cover for the histogram in Figure 3 is given in Figure 4.

Suppose now that we have two histograms, one with the upper chain as base and the other with the lower chain as base. We place them so that the bases coincide. What we have is a monotone rectilinear polygon in which the shortest watchman route is a straight line. It is no longer true that a minimum guard cover will also be a minimum set of vision points on a shortest watchman route; see Figure 2 and the discussion of the preceding section. However, if we restrict the guards to lie on the route, then the same approach we used for histograms will give an optimum set of vision points.

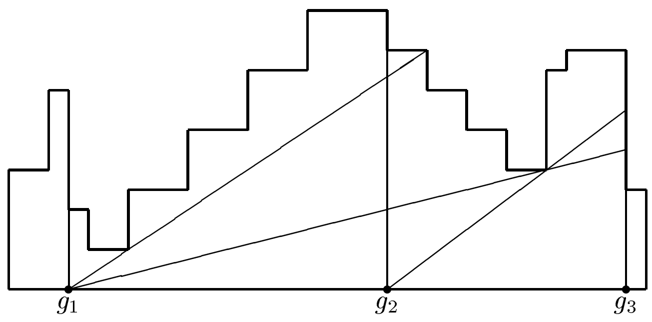


Figure 4: A simple example of the optimum cover produced by the algorithm.

The only difference is that we have to consider both monotone chains that make up the boundary, i.e., pyramids are on both sides of the base and the next guard point has to be computed with respect to both chains. Thus, a minimum set of vision points on a given straight line which is also the shortest watchman route for a monotone rectilinear polygon can be found in $\Theta(n)$ time.

4 Multiple Watchmen Routes

The problem of determining visibility properties using moving guards, from now on denoted watchmen, has received significantly less attention than the equivalent problems with static guards. However there are some results in this, harder, setting. If a watchman is allowed to "patrol" a line segment in the polygon it can be shown that $\lfloor n/4 \rfloor$ watchmen are necessary and sufficient to guard the polygon; see [12].

In this section we show that the problem of computing the m shortest watchmen routes is NP-hard. For histogram polygons we are able to show that the problem can be computed in polynomial time, and give a $\Theta(n^3 + n^2m^2)$ time algorithm based on dynamic programming.

The algorithm computes $\Theta(n^2m)$ candidate routes and chooses the m routes that give the shortest total solution. It should be noted that, although histograms are polygons with a very simple structure, the watchmen routes problem becomes very hard even for this "simple" class of polygons.

Definition 4.1 *The m-watchmen routes are a set of m floating routes which together sees a polygon. We define the length of the m-watchmen routes as the sum of the lengths of the routes. The m-watchmen routes problem is the problem of finding the m-watchmen routes with shortest total length.*

We prove that the m-watchmen routes problem is NP-hard.

Theorem 3 *The m-watchmen routes problem in simple polygons is NP-hard.*

Proof: It follows from the intractability of finding the minimum guard cover for a simple polygon; see [1, 9]. Suppose there exists an algorithm to compute the best m watchmen routes in polynomial time, then by running it for $m = 1, 2, \ldots, \lfloor n/3 \rfloor$ and, for each m, check whether the resulting routes have total length 0, we have a polynomial time algorithm for the minimum guard cover problem. Thus, we have proved the NP-hardness of the m-watchmen routes problem. $\square$

4.1 Multiple Watchmen Routes in Histograms

In this section we will look at how to compute the best set of routes, for any m moving watchmen in a histogram. It is interesting to note that this problem, although solvable in polynomial time, is much harder than finding the best set of static guards. To see this intuitively, consider the static guard problem of the previous section. Each guard is allowed to move only along the base, which gives each guard one degree of freedom. We will show that the watchmen routes also lie on the base but each watchman's route still has two degrees of freedom since both of the end points can move.

We sketch the algorithm we are going to apply. Instead of concerning ourselves with the watchmen routes, which lie on the base, we will treat the problem of computing the gaps between each pair of consecutive watchmen, i.e., compute the $m-1$ longest gaps such that the visibility requirements are not violated. Our first task is to define a bounded set of possible gaps between watchmen that provably always contains the solution. We then apply dynamic programming to choose the $m-1$ gaps with longest total length. The main problem lies in proving that the specified set of gaps always contains the solution.

We first prove that the optimum set of watchmen routes always lies on the base.

LEMMA 4.1 *There is an optimum solution in which all of the m floating routes are disjoint spans along the single watchman route on the base of the histogram.*

PROOF: Take any solution and project it orthogonally onto the base. Let point x be visible from some original watchman w. We need to show that the projected watchman w' also sees x. Since w sees x there is a point s_w on w for which the segment between x and s_w is interior to the polygon. By Lemma 3.2 the projected point s'_w also sees x and since w' is the projected watchman, s'_w lies on w'.

The two end points of the single watchman route see the two extremal pyramids, so any watchman of the solution which lies outside the route can be moved to one of the end points without affecting visibility. Also, this can never increase the total length of the solution. □

We will henceforth refer to the single shortest watchman route on the base as the $\mathcal{W}$-route. Furthermore, we denote the left end point of the $\mathcal{W}$-route as p and the right end point as q.

The next thing to show is that finding the gaps is equivalent to finding the watchmen routes.

LEMMA 4.2 *Finding the $m-1$ gaps with the largest total length, such that visibility requirements are fulfilled, is equivalent to the m-watchmen routes problem.*

PROOF: Take an optimum solution to the watchmen routes problem. The leftmost watchman has its left end point at p, the left end point of the $\mathcal{W}$-route, since if the point is to the left of p, it can be moved to the right without violating the visibility requirement. If the point is to the right of p, it does violate visibility. This follows since otherwise the $\mathcal{W}$-route would not be a shortest watchman route. By symmetry, the right end point of the rightmost watchman has its right end point at the point q, right end point of the $\mathcal{W}$-route.

Now, since the total length of the m watchmen and the $m-1$ gaps between the watchmen is fixed, i.e., equals the length of the $\mathcal{W}$-route, the lemma follows. □

In order to make the problem tractable we wish to discretize it and find a bounded number of points on the base which can be end points of the watchmen routes. To find these points we begin by making the following definition.

DEFINITION 4.2 *The concavity chains of the $\mathcal{W}$-route and a pyramid $\triangle$ are two reflex chains, called the left and right concavity chains. The left concavity chain is the shortest path between*

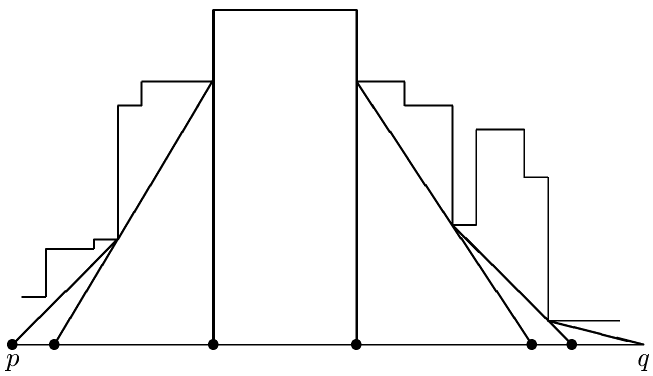


Figure 5: Pyramid with extended concavity chains and primary event points.

p, the left end point of the $\mathcal{W}$-route and the left end point of the top edge of $\triangle$. The right concavity chain is defined similarly as the shortest path between q and the right end point of the top edge of $\triangle$; see Figure 5.

The left concavity chains are easily computed by a scan of the top chain from left to right given the point p. The point p and the first vertex on the top chain to the right of p are pushed in this order onto a stack. For each vertex, until the top edge of the pyramid is found, a check is made that the current vertex and the two top elements on the stack do not form a right turn. If that is the case, the stack is popped and the check is performed again until the points form a left turn. The current vertex is then pushed onto the stack and the next vertex of the top chain becomes the new current vertex.

The right concavity chains can be computed in a similar way by a scan in the reverse direction.

DEFINITION 4.3 *The primary event points of a pyramid $\triangle$ are the intersection points of the $\mathcal{W}$-route and the extended links of the concavity chain that also intersect the top edge of $\triangle$; see Figure 5. The set of primary event points are the primary event points of all the pyramids excluding the first and the last pyramids plus the two points p and q.*

DEFINITION 4.4 *The secondary event points are all the points s on the $\mathcal{W}$-route such that $partner(s, \triangle_i) = partner(s, \triangle_j)$, for any two pyramids $\triangle_i$ and $\triangle_j$; see Figure 6.*

The primary event points can be computed by following the concavity chains, extending the links, and computing the intersection with the $\mathcal{W}$-route if the top edge is intersected. Since each vertex of a pyramid can allow at most one primary event point, there are at most a linear number of these points.

The secondary event points can be computed by looking at each pair of concavity chains. Consider a vertex on a concavity chain. The two primary event points, which have supporting lines passing through the vertex, specify the end points of an interval on the base. Not all vertices intersect the supporting lines of two event points but in this case we take the interval to be empty. In this way every vertex on a concavity chain has an interval associated to it. Now, for each pair of vertices lying on left concavity chains for pyramids $\triangle_i$ and $\triangle_j$, with $i \neq j$, compute the intersection of the associated intervals. If the intersection is empty, we are done. If the intersection is non-empty, let I be the intersection interval. Compute the partners of the

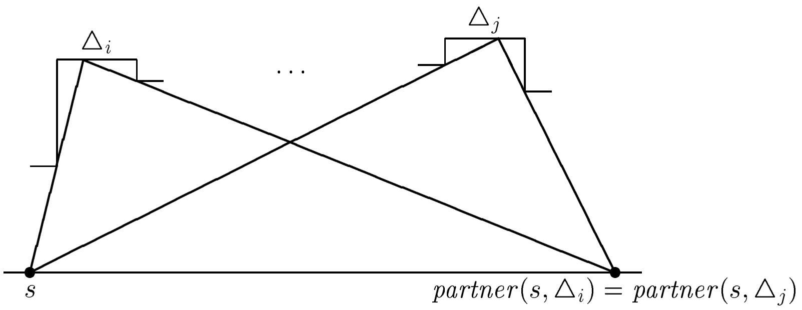


Figure 6: Secondary event points define local maxima of the distance function between guarding point and partner.

end points of I for both pyramids $\triangle_i$ and $\triangle_j$. These four points define two partner intervals to I, one for $\triangle_i$ and one for $\triangle_j$, denoted J_i and J_j. If one of J_i and J_j is contained in the other, then there is a secondary event point in I with a partner in J_i and J_j. If J_i and J_j overlap or are disjoint, then there is no secondary event point in I (corresponding to this pair of vertices). To see this note that the partner of a point s in I depends linearly on s; see Lemma 4.3. This means that if $s = a + (b-a)t$, then $rlp(s) = a_i + (b_i - a_i)t$, where the points a and b are the end points of I and the points a_i and b_i are the end points of J_i. Thus, the secondary event point can be computed by solving the equation[2]

$$a_i + (b_i - a_i)t = a_j + (b_j - a_j)t. \tag{2}$$

We can proceed similarly with the right concavity chains.

For the secondary event points, note that a pair of vertices from two different pyramids can allow a secondary event point. Hence, $O(n^2)$ is an upper bound on the number of secondary event points. We can, in fact, construct an example of a polygon which has $n = 8k + 8$ vertices and at least $k(k-1)/2$ secondary event points, giving $\Omega(n^2)$ secondary event points. Hence, the bound is asymptotically tight. The construction is depicted in Figure 7. The polygon consists of $k+2$ pyramids where the k middle ones are used to generate secondary event points. The left concavity chain of each pyramid consists of 4 segments and 5 vertices. The segments adjacent to the middle vertex of a concavity chain are extended to intersect the base at the end points of an interval I. We denote the left end point of I with l and the right one with r. Furthermore we let $a_i = partner(l, \triangle_{i+1})$ and $b_i = partner(r, \triangle_{i+1})$, with $J_i = [a_i, b_i]$. The pyramids are then drawn in such a way that $a_{i+1} = a_i + \alpha$ and $b_{i+1} = b_i - \alpha/2^i$, where α is a small value, with $\alpha < (b_1 - a_1)/k$.

Using this construction and Equation 2 it is then a simple matter of computing the secondary event points and verifying that there is one secondary event point per pair of pyramids, i.e., $k(k-1)/2$ secondary event points in total.

In the following we denote the primary and secondary event points simply as event points when we do not care of what type they are.

Each event point e has corresponding limit points $rlp(e)$ and $llp(e)$, as defined in Section 3, which define maximum visibility gaps from the event point. This means that two watchmen spanning the $\mathcal{W}$-route, except the part between the event point and one of the limit points, see the whole polygon and the limit point is also the furthest such point from the event point; see Lemma 3.4. The limit points can be computed using the same technique as in the algorithm *Histogram-Minimum-Cover* presented in Section 3.

[2]With a slight abuse of notation, since we are actually only concerned with the x-coordinates of these points.

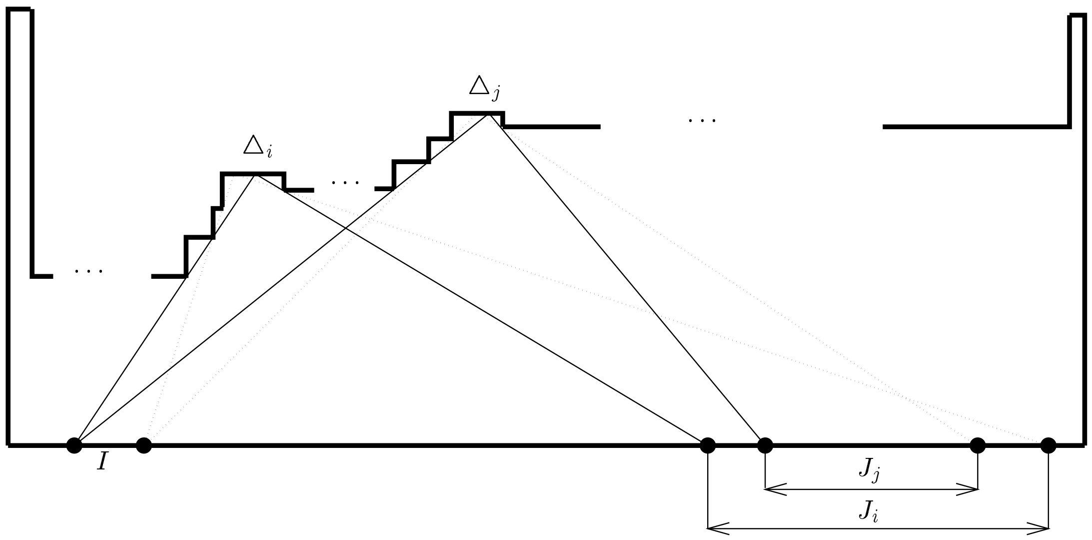


Figure 7: Example of histogram where each pair of pyramids generates a secondary event point.

The next lemma shows that these gaps are better than any other set of gaps. The idea of the proof is simply that any gap with end points that are not event points can be slided slightly to the right or left and since there is a linear relation between the point and its limit point the size of the gap increases in one of the directions.

LEMMA 4.3 *If* $[x, rlp(x)]$ *is a gap on the base of a histogram, such that* x *does not coincide with an event point or a limit point, then there exists a gap* $[y, y']$ *with* y *either slightly to the left of* x *or to the right of* x*, such that the gap* $[y, y']$ *is at least as long as the gap* $[x, rlp(x)]$.

PROOF: Let $x' = rlp(x)$. By the definition of right limit point we know that x' is the partner of x with respect to some pyramid $\triangle$. Hence, by Lemma 3.3, the supporting lines of x and x' intersect on the top edge of $\triangle$, and pass through the vertices $v_\triangle$ and $v'_\triangle$ of $\triangle$. Let a be a point such that it is either an event point with $a' = rlp(a)$ or a left limit point with $a = llp(a')$ and a being the closest such point to the left of x. Similarly let b be the closest such point to the right of x.

Take a point z in the interval $[a, b]$. We need to ensure that z and $rlp(z)$ have supporting lines with respect to $\triangle$ which intersect on the top edge of $\triangle$ and also pass through the vertices $v_\triangle$ and $v'_\triangle$ of $\triangle$.

Suppose this is not true. Either the supporting lines of z and $rlp(z)$ intersect at the top of another pyramid or the lines pass through different vertices than $v_\triangle$ and $v'_\triangle$ of $\triangle$. Hence, we have two cases to take care of:

1. **The case when the supporting lines of z passes through an other vertex than $v_\triangle$, say $u_\triangle$.**

   Take the upper of the two vertices (w.l.o.g. assume this to be $v_\triangle$) and its adjacent vertex w lying in between $v_\triangle$ and $u_\triangle$ on the concavity chain ($w = u_\triangle$ is possible). Extending the link $[v_\triangle, w]$ gives a primary event point in the interval between x and z which is a contradiction to the assumption that a and b were the closest such points to x. Similarly you can prove this for $rlp(z)$.

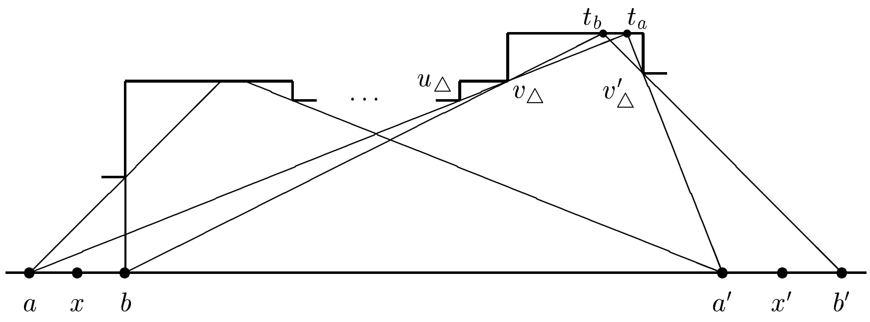


Figure 8: Gap $[b, b']$ is better than any gap starting between event points a and b. The points a', b', and x' are limit points to points a, b, and x, respectively.

2. **The case when the supporting lines of z and $rlp(z)$ intersect on the top edge of another pyramid $\triangle_+$.**

   If the supporting lines of z and $rlp(z)$ also intersect on the top edge of $\triangle$, then we immediately have a contradiction since then z is a secondary event point. Hence there is a point s between x and z where the intersection point of the supporting lines switches from pyramid $\triangle$ to pyramid $\triangle_+$. We can assume that there is no point between x and z for which the corresponding supporting lines intersect on the top edge of a third pyramid. If the supporting lines of s and $rlp(s)$ intersect on the top edges of both $\triangle$ and $\triangle_+$, we have again a contradiction since s is then a secondary event point. Hence we can assume that the supporting lines of s and $rlp(s)$ intersect on the top edge of only one pyramid, w.l.o.g. assume this to be $\triangle$, whereas all points to one side of s have supporting lines which together with the supporting lines of their right limit points intersect on the top edge of $\triangle_+$. We must prove that s or $rlp(s)$ is a primary event point, so assume the contradiction. Since neither s nor $rlp(s)$ is an event point there exists a neighbourhood around s for which the supporting lines of all points and their right limit points intersect on the top edge of $\triangle$, but this is not true for s so either s or $rlp(s)$ is a primary event point.

This proves our first claim, that the supporting lines of z and $rlp(z)$, with respect to $\triangle$, intersect on the top edge of $\triangle$ and also pass through the vertices $v_\triangle$ and $v'_\triangle$.

Event points mark the points where supporting lines move from one vertex to another; see Figure 8, where supporting lines to the left of a intersect vertex $u_\triangle$ and supporting lines to the right of a intersect vertex $v_\triangle$.

Let t_s be the point of intersection of the top edge of $\triangle$ and the supporting line of point s on the base. Now, since the triangles with corners at $a, x, v_\triangle$ and $t_a, t_x, v_\triangle$, as well as triangles with corners at $a', x', v'_\triangle$ and $t_a, t_x, v'_\triangle$, are similar, we have

$$\frac{|[a,x]|}{|[t_a,t_x]|} = \frac{|[a,v_\triangle]|}{|[t_a,v_\triangle]|} \qquad \text{and} \qquad \frac{|[a',x']|}{|[t_a,t_x]|} = \frac{|[a',v'_\triangle]|}{|[t_a,v'_\triangle]|}$$

and, hence,

$$\frac{|[a,x]|}{|[a',x']|} = \frac{|[a,v_\triangle]|}{|[a',v'_\triangle]|} \cdot \frac{|[t_a,v'_\triangle]|}{|[t_a,v_\triangle]|} = c, \text{ a constant.}$$

This gives us that the length of a gap $[y, rlp(y)]$, with $y \in [a, b]$ is

$$||[y, rlp(y)]|| = ||[x+\delta, rlp(x+\delta)]|| = ||[x, rlp(x)]|| + (c-1)\delta. \tag{3}$$

When $c \leq 1$ we can chose $\delta < 0$, i.e., y is to the left of x. Similarly, when $c > 1$ we chose $\delta > 0$ giving y to the right of x. In both cases $[y, rlp(y)]$ is at least as long as $[x, rlp(x)]$. □

The constant c obtained in the proof we call the *sliding constant of gap* $g = [x, rlp(x)]$ or in short notation c_g.

One might think that the gaps defined by the event points and the limit points, in this way, would be a sufficient set to define the best m watchmen routes. However, this is not the case. The reason for this is that two gaps in the set might overlap, and by placing a static guard somewhere on the overlap the resulting gap (containing the static guard) could be larger than the combined original gaps.

We prove that it is always preferable to place the static guard at an end point of such an overlap, and thus for each gap it is only necessary to compute new gaps defined by $m-2$ static guards to the left or right of any initially given gap.

We define the *right propagation points* to be a sequence of points $p_0, p_1, \ldots, p_{m-1}$ such that p_0 is an event point and $p_i = rlp(p_{i-1})$. Similarly we define the *left propagation points* propagating to the left of p_0. Such a sequence of points define gaps with static guards at the end points. For each event point we need to consider only $m-1$ propagation points since we have at most m watchmen in total.

Each event point corresponds to two sequences of propagation points, one propagating to the left and one propagating to the right. This gives us $2(m-1) \cdot O(n^2) = O(n^2 m)$ gaps in total. We let the set of gaps defined in this way be $\mathcal{G}$ and prove that the $m-1$ gaps giving an optimal solution are in this set, $\mathcal{G}$.

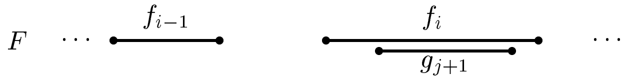
Lemma 4.4 *The set $\mathcal{G}$ contains $m-1$ non-overlapping gaps that have total length no smaller than an optimum set of gaps.*

Proof: We make a proof by comparing a solution contained in $\mathcal{G}$ with an allegedly optimal solution with at least one gap not in $\mathcal{G}$.

Let $F = (f_1, f_2, \ldots, f_{m-1})$ be an optimal solution where $f_1, f_2, \ldots, f_{m-1}$ are gaps ordered from left to right. We will show that there is a solution no smaller than the solution F having all its gaps in $\mathcal{G}$. Let f_i be the leftmost gap of F not in $\mathcal{G}$. Let g_j and g_{j+1} be the gaps in $\mathcal{G}$ that have their left end points closest to the left end point of f_i. We assume that the left end point of g_j is to the left of the left end point of f_i and thet the left end point of g_{j+1} is to the right of the left end point of f_i.

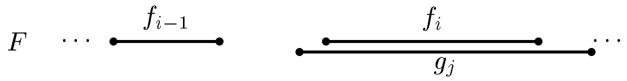
The following cases arise.

f_i contains g_{j+1}.



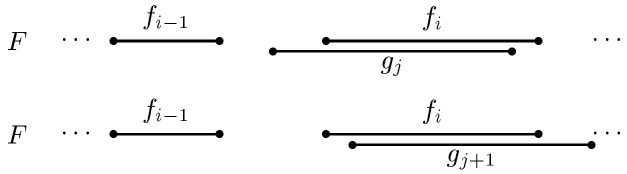
By Lemma 3.4, the gap g_{j+1} is the largest gap retaining visibility. Gap f_i therefore violates the visibility requirement.

f_i is contained in g_j.



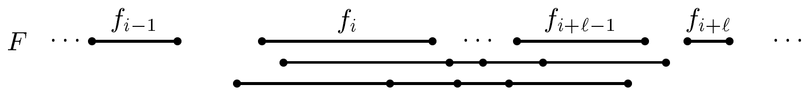
The solution $(f_1, \ldots, f_{i-1}, g_j, f_{i+1}, \ldots, f_{m-1})$ improves on F. Should g_j and f_{i+1} overlap a static guard can be placed anywhere in the overlap.

f_i and g_j overlap. This case divides into two subcases. The first subcase has two variants as depicted in the figure below.



If f_i does not have an end point in common with neither f_{i-1} nor f_{i+1}, then by Equation 3 in Lemma 4.3, there is a gap f_i' with $|f_i'| = |f_i| + (c_{f_i} - 1)\delta \geq |f_i|$.

If $c_{f_i} = 1$ the size of f_i' has not increased but we can then choose $f_i' = g_{j+1}$ or join f_i' to f_{i+1} with a static guard between them, whichever occurs first. In the latter case we have transformed the solution to the second subcase where f_i and f_{i+1} have a static guard between them. The general version is depicted in the following figure.



It is clear that f_i and f_{i-1} do not have a static guard between them since $f_{i-1} \in \mathcal{G}$ implies that the end points of f_i are propagation points of some event points. Hence, assume that the set $\{f_i, f_{i+1}, \ldots, f_{i+\ell-1}\}$ is a maximal set of consecutive gaps with only static guarding points between them. By repeated application of Equation 3 in Lemma 4.3 we can choose $\{f_i', f_{i+1}', \ldots, f_{i+\ell-1}'\}$ to be a sequence of gaps such that $|f_i'| + |f_{i+1}'| + \cdots + |f_{i+\ell-1}'| = |f_i| + |f_{i+1}| + \cdots + |f_{i+\ell-1}| + (c_{f_i} c_{f_{i+1}} \cdots c_{f_{i+\ell-1}} - 1)\delta \geq |f_i| + |f_{i+1}| + \cdots + |f_{i+\ell-1}|$ and increase the total size of the solution.

If $c_{f_i} c_{f_{i+1}} \cdots c_{f_{i+\ell-1}} = 1$, then again we can choose $f_i' = g_{j+1}$ or join $f_{i+\ell-1}$ to $f_{i+\ell}$ with a static guard in between, whichever occurs first. Repeated application of the argument proves the lemma in the case when f_i and g_j overlap.

f_i and g_j are disjoint.
This case can never occur since the union of the gaps in $\mathcal{G}$ span the complete $\mathcal{W}$-route. Therefore some gap in $\mathcal{G}$ will intersect f_i.

□

We get the following algorithm

Algorithm *m-Watchmen-Routes*
Input: A histogram polygon P and a positive integer m
Output: The m shortest watchmen routes in P
Step 1 Divide P into its pyramids, $\triangle_1, \triangle_2, \ldots, \triangle_k$
Step 2 Compute the event points for pyramids $\triangle_1$ to $\triangle_k$
Step 3 **for** each event point **do**
Step 3.1 Compute the $m-1$ gaps propagating to the right from the event point
endfor
Step 4 Redo step 3 for the gaps propagating to the left
Step 5 Sort the gap end points on x-coordinate
Step 6 Apply dynamic programming to choose among the set of gaps
End *m-Watchmen-Routes*

Let s be the right end point of some gap. If t is the left end point of the gap we call t the *mate* of s.

In Steps 3 and 4 we take care of the possibility of a sequence of propagation points. These points are computed using the same technique as in Section 3. The dynamic programming scheme will scan the gap end points from left to right, which is why these points are sorted in Step 5. In Step 6 we assume a matrix M of size $m \times O(n^2 m)$. Item $M[i,j]$ contains the best solution of i watchmen from the left initial point p, the left end point of the $\mathcal{W}$-route, to the j^{th} gap end point. The best solution to item $M[i,j]$ is achieved by taking the best of two items:

1. item $M[i-1,k]$, where $k < j$ and k is the index of the mate of the gap end point having index j. This corresponds to placing a static guard at the gap end point with index j or
2. item $M[i,j-1]$ + {the length between the gap end points having indices $j-1$ and j}. This corresponds to a watchman walking the distance between the $j-1^{\text{st}}$ and the j^{th} gap end points.

LEMMA 4.5 *The algorithm runs in* $\Theta(n^3 + n^2m^2)$ *time and uses* $\Theta(n^2m^2)$ *storage.*

PROOF: Steps 1 and 2 take $O(n^2)$ time by the discussion following Definitions 4.3 and 4.4. Steps 3 and 4 can take up to $O(n^3)$ time since one scan of the polygon is performed to compute each sequence of propagation points and there are $O(n^2)$ of these sequences. Sorting of the $O(n^2m)$ gap end points takes $O(n^2 m \log n)$ time in Step 5. By the previous discussion Step 6 takes $O(n^2m^2)$ time since the matrix has at most n^2m^2 entries and each entry can be computed in constant time. This gives a time complexity of $O(n^2(n + m^2 + m \log n)) = O(n^2(n + m^2))$.

The storage bound follows from the fact that the matrix has $m \times O(n^2m)$ entries.

Again, the example, having $\Omega(n^2)$ secondary event points, mentioned in the beginning of this section, uses $\Omega(n^3 + n^2m^2)$ time and $\Omega(n^2m^2)$ storage which shows the tightness of the complexity bounds for our algorithm. □

Lemmas 4.4 and 4.5 give the following theorem.

THEOREM 4 *The m-Watchmen-Routes algorithm computes an optimum set of* m *watchmen routes in* $\Theta(n^3 + n^2m^2)$ *time using* $\Theta(n^2m^2)$ *storage.*

5 Conclusions

We have presented a number of results in this paper. The first is that computing the restriction of a minimal guard set on a watchman route is NP-hard. We have also presented a linear time algorithm to compute the minimal guard set in a histogram polygon, which is also a minimal set of vision points on the watchman route along the base of the histogram.

In the last section we have shown an algorithm to compute the best set of m watchmen in histogram polygons. Our algorithm runs in $\Theta(n^3 + n^2m^2)$ time and uses $\Theta(n^2m^2)$ storage.

Other problems related to the problem of finding the best watchmen for any value of watchmen (which is NP-hard in the general case) is that of finding best watchmen for a fixed number of watchmen, e.g., does there exist a polynomial time algorithm to compute the best 2-watchmen routes for simple polygons? Even for the single watchman route case the problem has not been completely answered, although, as stated in the introduction, a $\Theta(n)$ time algorithm exists for simple rectilinear polygons.

References

[1] A. Aggarwal. *The Art Gallery Theorem: Its Variations, Applications and Algorithmic Aspects.* PhD thesis, Johns Hopkins University, 1984.

[2] B. Chazelle. Triangulating a Simple Polygon in Linear Time. In *Proc. 31st Symposium on Foundations of Computer Science*, pages 220–230, 1990.

[3] W. Chin, S. Ntafos. Optimum Watchman Routes. *Information Processing Letters*, 28:39–44, 1988.

[4] W. Chin, S. Ntafos. Shortest Watchman Routes in Simple Polygons. *Discrete and Computational Geometry*, 6(1):9–31, 1991.

[5] V. Chvàtal. A Combinatorial Theorem in Plane Geometry. *Journal of Combinatorial Theory B*, 13(6):395–398, 1975.

[6] J.C. Culberson, R.A. Reckhow. Covering Polygons is Hard. In *Proc. 29th Symposium on Foundations of Computer Science*, pages 601–611, 1988.

[7] H. Edelsbrunner, J. O'Rourke, E. Welzl. Stationing Guards in Rectilinear Art Galleries. *Comput. Vision, Graphics, Image Processing*, 27:167–176, 1984.

[8] S. Fisk. A Short Proof of Chvàtal's Watchman Theorem. *Journal of Combinatorial Theory B*, 24:374, 1978.

[9] D.T. Lee, A.K. Lin. Computational Complexity of Art Gallery Problems. *IEEE Transactions on Information Theory*, IT-32:276–282, 1986.

[10] D.T. Lee, F.P. Preparata. An Optimal Algorithm for Finding the Kernel of a Polygon. *Journal of the ACM*, 26:415–421, 1979.

[11] B.J. Nilsson, D. Wood. Watchmen Routes in Spiral Polygons. Technical Report LU-CS-TR:90-55, Dept. of Computer Science, Lund University, 1990. An extended abstract of a preliminary version was presented at the *2nd Canadian Conference on Computational Geometry*, pages 269–272.

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

[13] J.-R. Sack. *Rectilinear Computational Geometry.* PhD thesis, School of Computer Science, McGill University, 1984.

[14] X.-H. Tan, T. Hirata, Y. Inagaki. An Incremental Algorithm for Constructing Shortest Watchman Routes. In *Proc. ISA '91 Algorithms*, pages 163–175. Springer Verlag, Lecture Notes in Computer Science 557, 1991.