The 3-dimensional Searchlight Scheduling Problem

Giovanni Viglietta*   Maurizio Monge†

Abstract

The problem of searching for a mobile intruder in a polygonal region by a set of stationary guards, each carrying an orientable laser, is known in the literature as the Searchlight Scheduling Problem. A long-standing conjecture concerns the NP-hardness of deciding if a given polygon is searchable by a given set of guards.

In this paper we introduce the more general problem of detecting an intruder in a 3-dimensional polyhedral region by a set of searchplanes within a given time, and we prove its NP-hardness.

1 Introduction

The 2-dimensional Searchlight Scheduling Problem (SSP) was first studied in [5] as a variation on the well-known Art Gallery Problem. Consider a set of point-guards, called *searchlights*, statically positioned in a polygonal environment (which may be thought of as a floor plan, with edges as walls). Each guard can only see along a ray, but may continuously vary the direction of visibility over time (as if each guard is carrying a laser beam). An intruder is also hiding in the polygon, and is allowed to continuously move with unbounded speed. SSP asks for a schedule for the angular motion of the searchlights, such that the intruder is necessarily detected by some searchlight in finite time, no matter what path he takes. The intruder is detected by a searchlight if it lies on its ray of vision, while the polygon's exterior blocks both vision and movement. A trivial necessary condition for a problem instance to be solvable is that each point in the polygon must be potentially visible by at least one guard, and that every guard which is not visible by any other guard must be located on the polygon's boundary. Therefore, all valid problem instances are supposed to satisfy this condition.

In [5] a heuristic is presented to search a simple polygon, which is guaranteed to solve all valid problem instances in which there is at least one searchlight located on the boundary for every connected component of their visibility graph. In particular, if all guards lie on the boundary, they have a successful searching schedule. Simple necessary and sufficient conditions for a polygon with one or two searchlights to have a successful schedule are also given in [5].

A complete searchlight scheduling algorithm for polygons with holes is described in [3], based on exact cell decomposition. The same paper also mentions the problem of minimizing searching time, in a scenario where searchlights have bounded angular speeds, as an interesting and yet unexplored variation of SSP.

The original SSP was further extended by considering guards carrying $k \geqslant 1$ searchlights, called k-guards. The ∞-guard is then defined as the traditional guard from the Art Gallery Problem. In [6] some upper bounds on $s_k(P)$ are given, that is the minimum number of k-guards required to search a polygon P, while it is well-known from [2] that computing $s_\infty(P)$ is NP-hard.

The problem of determining the exact complexity of SSP was not directly addressed in [5], but it has become a rather important issue over time. Even though it is still unknown whether SSP is NP-hard, the algorithm given in [3] can be practical for problem instances of useful size.

In this paper we model what we believe to be the first 3-dimensional version of the Searchlight Scheduling Problem. Specifically, we consider searchlights modeled as *segment-beams* statically placed in a polyhedral region, emitting *searchplanes* which can be rotated about such segments at bounded angular speeds. Our main contribution is the proof of the NP-hardness of computing the minimum time required to search a given polyhedral region by a given set of searchlights.

2 Definitions

For the purposes of this paper, a *polyhedral surface* will be a connected orientable 2-manifold without boundary, union of a finite number of plane polygons (possibly with holes) in $\mathbb{R}^3$, such that any two coplanar polygons are either disjoint or their intersection is a common vertex. Such a surface disconnects $\mathbb{R}^3$ in exactly one bounded component (*interior*) and one unbounded component (*exterior*). The union of a polyhedral surface and its interior is called *polyhedron*. A polyhedron is *orthogonal* iff each one of its edges is parallel to some axis.

The formulation of the 3-dimensional Searchlight Scheduling Problem is now straightforwardly adapted from [5], provided that a new definition of *searchlight* is given. In this context, a searchlight in a polyhedron

*Department of Computer Science, University of Pisa, `viglietta@di.unipi.it`

†Scuola Normale Superiore, Pisa, `maurizio.monge@sns.it`

P will be any non-degenerate line segment contained in P. A *schedule* for a searchlight ℓ is a continuous function $f_\ell : [0, T] \to \mathbb{S}^1$, expressing the orientation of ℓ's *searchplane* over time. So, a point $x \in P$ is visible by a searchlight ℓ at time $t \in [0, T]$ iff x is visible by some point in ℓ, and x lies on the half-plane whose end-line contains ℓ and whose orientation is expressed by $f_\ell(t)$. In other words, a searchlight is a line segment ℓ that is able to emit a half-plane of light in any desired direction, and rotate it continuously about the axis defined by ℓ itself. A searching schedule is said to *clear* P iff, by time T, any intruder has been caught by some searchlight.

Definition 1 *3-dimensional Searchlight Scheduling Problem (3SSP): Given a polyhedron P and a set of searchlights L, determine if a searching schedule exists for L which clears P in finite time.*

Since we're also concerned with time evaluation, let's assume that the angular speed of every searchlight ℓ is bounded by some constant K_ℓ, i.e. ℓ's schedules are Lipschitz continuous functions with Lipschitz constant K_ℓ.

Definition 2 *3-dimensional Timed Searchlight Scheduling Problem (3TSSP): Given a polyhedron P, a set of searchlights L and a time limit T, determine if a searching schedule exists for L which clears P within time T.*

3 Preliminaries

In this section we state some elementary properties of 3SSP, which also provide a rationale for the choice of our model. Specifically, our choice to employ line-guards as opposed to point-guards is supported by strong geometric facts concerning visibility.

It is well-known (see for example [4]) that the reflex vertices of a polygon can collectively guard its whole interior, and this can be proved by inductively splitting the polygon along the bisectors of reflex angles. On the other hand, this doesn't generalize to point-guards in polyhedra, since Seidel's polyhedra (see [4]) can't be guarded by vertex-guards at all, and actually they can't even be guarded by a linear number of point-guards.

However, the splitting argument can be successfully extended to polyhedra by cutting along reflex edges, also called *notches*. In [1] a partitioning algorithm into convex parts is described which is based upon such splitting technique. It follows that a polyhedron can be guarded by its notches, i.e. each point in a polyhedron is visible by at least one point on some notch.

So far, edge-guards in polyhedra acted like the natural 3-dimensional counterparts of vertex-guards in polygons, while point-guards seemed way too weak to guard polyhedra. To further pursue this analogy, let's compare the power of 2-dimensional searchlights and 3-dimensional searchplanes. Consider the partition we just described for polygons, and notice that every part is a convex polygon such that at least one of its vertices was originally a reflex vertex, while the union of all the cuts is a planar embedding of a forest graph. Hence, if searchlights are placed at reflex vertices, they are not only able to see the whole polygon, but they also have a successful searching schedule. Align them with the angle bisectors of their respective vertices, and move *some* of them one by one in the correct order (determined by the structure of the forest graph). Every time a searchlight is moved, it sweeps all the area it can see, and then returns back in place, along the angle bisector. By a straightforward induction, it can be shown that such a schedule clears the whole polygon.

Attempting to extend this reasoning to polyhedra gives rise to several difficulties. As shown in [1], cutting along a notch may split other notches, may fail to disconnect the polyhedron, may generate new polyhedra with higher genuses, and the cut itself is a polygon which may have holes. However, we are confident that a careful analysis of the cut structure will lead to a proof of the following:

Conjecture 1 *The instance (P, N) of 3SSP, where N is the set of notches of a polyhedron P, has a successful searching schedule.*

Another desirable feature of our searchlights is that their movement has only one degree of freedom, which makes for a simple description and analysis of their schedules. As a consequence, any instance (P, L) of SSP can be trivially transformed into an instance of 3SSP, by extruding P to form a prism and converting each searchlight in L into a maximal straight line parallel to the prism's sides.

Observation 1 *SSP $\leqslant_{\mathbf{P}}$ 3SSP.*

Notice that not all reasonable models of searchlights immediately enable this reduction: for instance, consider searchplanes rotating about a point with two degrees of freedom.

A hardly avoidable downside of our model is that a searchlight may fail to "disconnect" a polyhedron regardless of its genus, even when placed on its boundary and aimed at its interior. Hence, most of the sufficient conditions for searchability obtained in [5] don't generalize to 3SSP, and in particular the One Way Sweeping Strategy doesn't work for arbitrary searchlights. Explicit counterexamples are readily constructed: Figure 1 depicts two unsearchable instances, whose searchlights lie on the boundary and can collectively see everything. However, it is easily proved that no counterexample exists for instances with only one searchlight.

Observation 2 *The instance $(P, \{\ell\})$ of 3SSP has a successful searching schedule iff ℓ can see all P, and lies on P's boundary.*

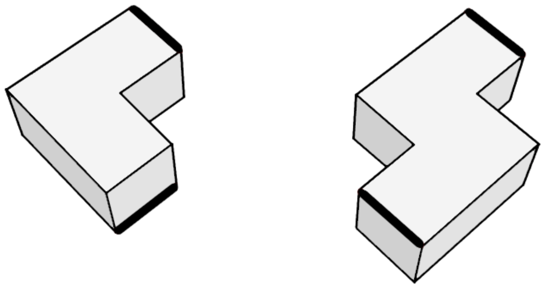

Figure 1: Two unsearchable instances of 3SSP. Thick lines mark searchlights.

4 NP-hardness of 3TSSP

Next we show a polynomial time reduction from 3SAT to 3TSSP. Our construction will transform a 3CNF formula φ into an orthogonal polyhedron P and a set of searchlights L lying on P's edges, each with maximum angular speed of $90°/\sec$, such that P can be cleared by L within 3 seconds iff φ is satisfiable.

A *link* is a thin uncapped parallelepiped, with a short searchlight in the middle. Clearing a link requires 1 second, and it's possible iff both ends remain capped by some external searchplanes. A *room* is a cube with a searchlight lying on some edge. Rooms and links can be attached together to form a *chain*, in such a way that the searchlight in each room has links on both its adjacent faces, as depicted in Figure 2.

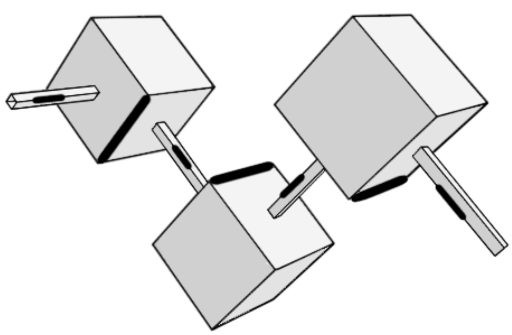

Figure 2: Odd-chain made of rooms and links.

If a schedule is going to clear a chain in 3 seconds, it has to move all the searchlights in every room during second 2, while during seconds 1 and 3 they have to remain on their sides, in order to cap the surrounding links. As a consequence, the links in a chain must be cleared at second 1 and second 3, alternately. A chain is called *even-chain* or *odd-chain*, depending on the number of its rooms. Chains will connect *variable devices* with the corresponding *clause devices*.

To represent a boolean variable x, we use a pair of rooms C_x and C'_x, each with a *fake link* on its left side, as Figure 3 suggests. The far end of each fake link is capped, as they're required for synchronization purposes only. For each occurrence of x in φ, we attach an even-chain to the right side of C_x, and an even-chain (resp. odd-chain) to the right side of C'_x if the occurrence of x is positive (resp. negative). Both chains will be connected to the same *valve* of the proper clause device, as described below. We'll say that variable x is assigned the value *true* iff both its searchlights rotate in the same direction (clockwise or counterclockwise) during second 2. Notice that the truth value of a variable device is well-defined in every schedule that clears it in 3 seconds.

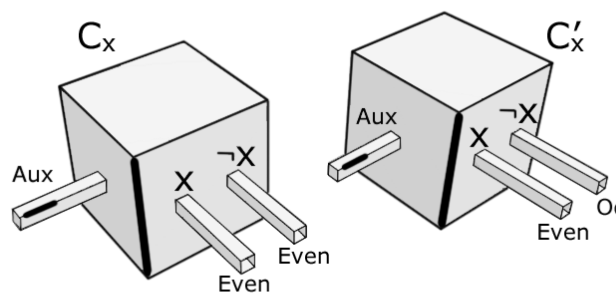

Figure 3: Boolean variable with 2 occurrences.

For each clause in φ, we construct a *clause device*, consisting of a *3-input OR gate* and 3 *valves* attached to it. Each valve corresponds to an occurrence of a variable, and is then attached to the proper chain pair coming from that variable.

An *OR gate* is a cube with 3 small holes and 2 searchlights, such that the searchlights can simultaneously close at most 2 holes (see Figure 4). Its clearing time is 1 second, provided that recontamination through holes is avoided, via external searchplanes.

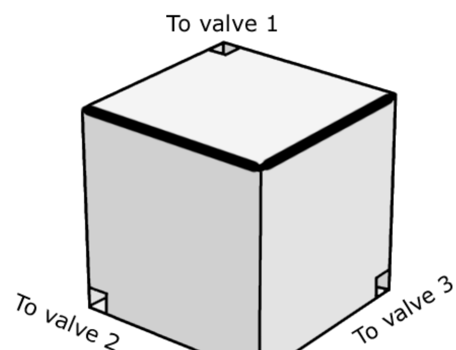

Figure 4: 3-input OR gate.

Each hole in an OR gate directly connects to a small *valve*, shown in Figure 5. Searchlight A is able to cap the chain pair and close the hole leading to the OR gate but, in order to do both, it has to spend 1 second switching position. Searchlight B always clears the valve no matter what A does, provided that the hole remains closed. The auxiliary fake links force B to move during second 2 only.

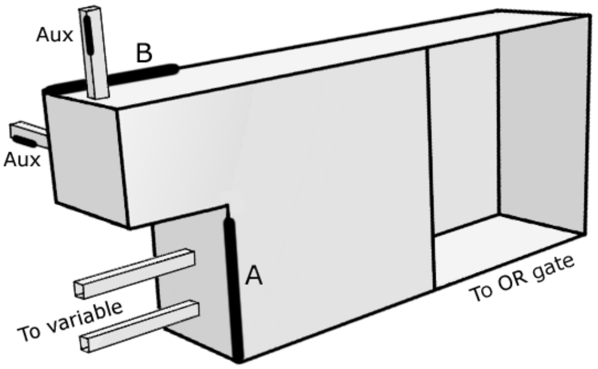

Figure 5: Valve.

This completes the construction. Notice that building a chain with the correct parity and direction is never an issue, because 3 rooms and 3 links can be added to extend a chain without changing the direction of its end links. Moreover, if the chains are thin enough, the number of their bends can be bounded by a constant, while keeping them pairwise disjoint and with rational vertices. Hence the size of the whole construction is indeed linear in the size of φ.

Suppose now that φ is satisfiable, and let's show that our construction can be cleared in 3 seconds. For each variable x, the searchlight in C_x starts on its right, while the searchlight in C'_x starts on its left (resp. right) if x is false (resp. true) in the chosen satisfying assignment. Consider now an occurrence of x, and its corresponding valve. If the occurrence is satisfied, both links attached to the valve can be cleared at second 1. So searchlight A caps the links during second 1, waits for B to move during second 2 (thus protecting the chains from the contaminated fake link still uncapped by B), and finally moves to close the valve during second 3. On the other hand, if the occurrence of x is not satisfied, A keeps capping the links and never moves (since they have to be cleared during seconds 1 and 3, respectively), while B moves during second 2. Thus, by assumption, in the end of second 3, each OR gate will have at least one hole closed by the corresponding valve. The searchlights in the OR gate keep the remaining 2 holes closed throughout seconds 1 and 2, and finally the proper searchlight moves to clear the OR gate during second 3. If a valve's hole is initially closed by the OR gate, then the valve itself is successfully cleared by searchlight B during second 2, hence the OR gate can't be contaminated by the valve during second 3. If a valve is initially open, then a small portion of it (the "nook" with the fake links) is cleared by B at second 2, while the rest is cleared by A at second 3. Notice also that no recontamination can ever occur between an OR gate and some auxiliary fake link in one of its valves, because searchlight A always separates the two regions.

Conversely, let's assume by contradiction that φ is not satisfiable, and some searching schedule clears our construction within 3 seconds. As already noticed, every variable device must be cleared at second 2, while each link must be cleared at second 1 or 3. By assumption, in at least one clause device every valve is attached to a link that is cleared during second 1 and to a link that is cleared during second 3. While links are being cleared, searchlight A has to cap them, and it's too slow to approach the valve's hole and come back in place during second 2. Therefore, the region around the hole must be cleared by searchlight B, which is allowed to move only during second 2, because of the fake links it has to cap. To avoid recontamination while B sweeps the hole, some searchlight in the OR gate must keep it closed for $\varepsilon > 0$ seconds. But since there are only 2 such searchlights for 3 holes, one searchlight has to close 2 holes in strictly less than 1 second, which is impossible.

This ends the proof of our main result:

Theorem 1 *3TSSP is NP-hard, even when restricted to orthogonal polyhedra with searchlights lying on edges.*

5 Conclusions

We have modeled a 3-dimensional version of the Searchlight Scheduling Problem, pointing out its basic features and proving the NP-hardness of computing the optimal searching time of a given instance. Besides studying new solving heuristics, further efforts may be devoted to proving the hardness of approximation of the optimal searching time, as well as the NP-hardness of 3SSP with no time constraints.

References

[1] B. Chazelle. Convex partitions of polyhedra: a lower bound and worst-case optimal algorithm. *SIAM Journal on Computing*, 13(3):488–507, 1984.

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

[3] K. J. Obermeyer, A. Ganguli, and F. Bullo. A complete algorithm for searchlight scheduling. *International Journal of Computational Geometry & Applications*, October 2008. Submitted.

[4] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, 1987.

[5] K. Sugihara, I. Suzuki, and M. Yamashita. The searchlight scheduling problem. *SIAM Journal on Computing*, 19(6):1024–1040, 1990.

[6] M. Yamashita, I. Suzuki, and T. Kameda. Searching a polygonal region by a group of stationary k-searchers. *Information Processing Letters*, 92(1):1–8, 2004.