Placement and Orientation of Rotating Directional Sensors

Giordano Fusco and Himanshu Gupta
Stony Brook University.
Email: {fusco,hgupta}@cs.sunysb.edu

***Abstract*—In this paper, we address several problems that arise in the context of rotating directional sensors. Rotating directional sensors (RDS) have a "directional" coverage region that "rotates" at a certain speed. For RDS with fixed given locations, we address three problems with the objective to minimize different functions of the dark time (i.e., uncovered time) of the given points in the area. In addition, we also consider the problem of placement and orientation of the minimum number of given RDS, so as to reduce the dark time of all given points to zero. Finally, we address the barrier coverage problems wherein we wish to place and/or orient the RDS to ensure "detection" of maximum number of intruders who are attempting to cross the monitored area. We prove the addressed problems to be NP-hard; some of the them are showed to be even NP-hard to approximate. We provide approximation algorithms which are easy to decentralize.**

I. Introduction

A *directional sensor* is a sensor with "directional" coverage (e.g., a radar or a camera) which can sense only in the direction of its *orientation*. A *rotating directional sensor* can change the orientation of its sensor at a certain rotational speed. The rotational speed of the orientation is given, but the initial orientation can be set arbitrarily. Recently, there has been a lot of interest [1], [2], [3], [4] in coverage problems for directional sensors with fixed (but choosable) orientation, but coverage problems for rotating directional sensors have not been addressed yet. However, the choice of initial orientation in RDS gives rise to many interesting problems, addressed in our article.

In the first set of problems addressed (Section III), we are given a set of points and a set of RDS with fixed locations in a 2D region, and the problems are to choose an initial orientation for each RDS so as to optimize some function of the "dark" time of the given points. A point is considered to be in dark at a given time if it is not covered by any sensor at that time. We consider three problems corresponding to three different objective functions, viz., minimize the average of the dark times, minimize the maximum of the dark times, and minimize the maximum of the "contiguous" dark times. We show that all these problems are NP-hard, while the latter two problems are even NP-hard to approximate. We design a constant-factor approximation algorithm for the first problem, and "double-approximation" algorithms for the other problems. Then, in the next Section IV, we consider the problem of placement (locations) and initial-orientation of a minimum number of RDS, such that the dark time of all the given points is zero. We design two different approximation algorithms for this problem; one of our approximation algorithms is based on the recent work by Agarwal et al. [2] on a related problem. Finally, in Section V, we consider the barrier coverage problems wherein the objective is to ensure detection of all (or as many as possible) intruders trying to cross the monitored area. In this context, we consider both the problems, viz., (i) of determining the initial-orientation of already placed RDS, and (ii) of determining the placement and initial-orientation of a minimum number of RDS. We show these problems to be NP-hard, and we provide constant-factor approximation algorithms for them. For the special case of intruders moving at infinite speed, we give a polynomial-time algorithm.

We start with a discussion of related works in the next section.

II. Related Work

To the best of our knowledge, ours is the first work on placement and/or orientation of rotating directional sensors. Below, we discuss works on related problems.

Works on coverage using cameras or static directional sensors are the most closely related to our paper. In our previous work [1], we addressed the problem of selection and orientation of static directional sensors. In this article, we extend some of the techniques from [1]. For cameras which have static directional-cones as sensing regions, Ai and Abouzeid [3] proposed a greedy heuristic (without any performance guarantee) to orient the given cameras to maximize coverage. Hörster and Lienhart [4] addressed a number of camera-coverage problems and designed heuristics or exponential-time algorithms for them. The main focus of the above works is to orient already placed sensors. In a recent paper [2], Agarwal et al. gave an approximation algorithm to the problem of placing a minimum number of sensors; their work can be applied to static directional sensors. In particular, we adapt their technique to our problem of placement and orientation of RDS.

Coverage problems with static omni-directional sensors have been extensively studied [5], [6], [7], [8], [9], [10]. However, their results do not extend directly to the RDS coverage problems. The well-known Art Gallery problem is also related to the sensor coverage problem. The Art Gallery

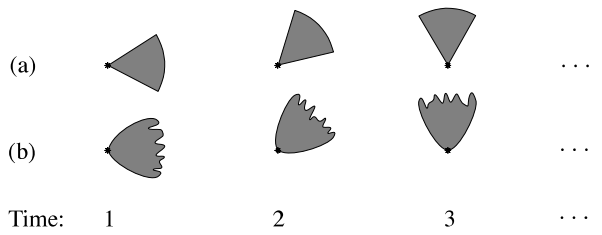


Fig. 1. Examples of *rotating directional sensors* with (a) cone-shaped sensing regions, and (b) more general sensing regions. The initial orientation (time slot 1) is towards right. The RDS rotate anti-clockwise.

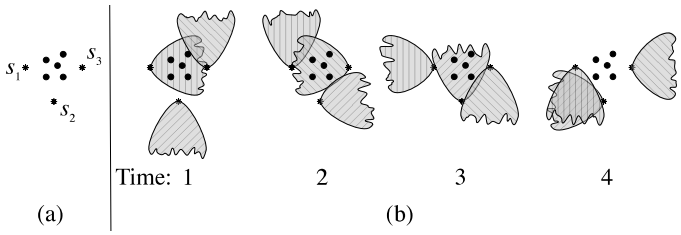


Fig. 2. Example of the Orient Placed RDS problem. Figure (a) shows the 3 RDS at s_1, s_2, and s_3 and 5 points. Figure (b) shows coverage of the points at each time slot corresponding to the initial-orientations shown/chosen in the first time slot. Here, each RDS has four different orientations and associated sensing regions.

problem [11], [12], [13], [14] is to place a minimum number of guards in a polygon so that each point in the polygon is visible from at least one of the guards. Guards can be looked upon as sensors with infinite range. Results for the Art Gallery do not extend directly to our problem of placing and orienting RDS, since the coverage region of an RDS changes over time.

Barrier coverage problems were first intruded in the field of robotics [15]. Since then, a lot of work [16], [17], [18], [19], [20], [21], [22] has been done. The recent work [23] is the only study we are aware of on barrier coverage with directional sensors. All of these works assume static coverage regions, and thus do not extend directly to our setting.

III. Orientation of Placed Sensors

In this section, we consider the problem of determining the initial orientation of already placed RDS in order to minimize some function of the dark time of the given points. We start by defining our model of rotational directional sensors.

A. Rotating Directional Sensors

In this subsection, we give a formal definition of *rotating directional sensors* (RDS) and *dark time*. Informally, an RDS senses a phenomena only in a certain direction, and this direction changes over time. Typical examples of RDS are radars or rotating cameras.

Definition 1. (Directional sensor.) A *directional sensor* is a sensing device with multiple 2D *sensing regions* associated with it, with only one of them active at a time. The active sensing region is determined by the chosen "orientation" of the directional sensor. □

For most of our results, we do not make any assumption on the shape of RDS sensing regions, except that each sensing region is closed, connected, and without holes. See Figure 1 for examples of sensing regions.

In this article, we assume the time to be divided in *time slots.*

Definition 2. (Rotating directional sensors (RDS)) A *rotating directional sensor* is a directional sensor whose orientations are ordered in sequence. Starting from a chosen *initial-orientation* in the first time slot, the RDS iterates through its orientations over time (time slots). See Figure 1. □

For simplicity, we assume that each RDS has the same number of orientations. Thus, the state of the system repeats after every P time slots, where P is the total number of orientations of each RDS.

Definition 3. (Dark time) A given point is considered *dark* in a given time slot if it is not covered by any active sensing region of an RDS in that time slot. The *dark time* of a point is the total number of time slots (among the period of P time slots) the point is dark. The *longest dark time* is defined as the longest *contiguous* number of time slots that a point is dark. □

B. Problem Definition and Summary of Results

We now formally define three problems related to minimization of dark time, when the given RDS are already placed. Then, we will illustrate the problems using an example, and give a brief summary of results.

Problem 1. (Orient Placed RDS) Given a set of RDS with their locations and a set of target points, determine the initial-orientation of each RDS so as to:

- *min-avg dark*: minimize the average (or the sum) of the dark time of all the given points;
- *min-max dark*: minimize the maximum of the dark time of all points;
- *min-longest dark*: minimize the longest dark time of all points.

For simplicity, in the above problems, we have assumed a finite number of given points. However, our techniques easily generalize to minimizing the dark time of a given area as observed in Subsection III-E. We assume that each RDS has only a finite number of orientations.

Example 1. Consider 3 RDS and 5 points as shown in Figure 2(a). Let each sensing region be as in Figure 1(b), and each RDS have 4 different orientations. If we set the initial-orientations of each RDS as in Figure 2(b)'s time slot 1, then all points have a dark time equal to 1 (because they are not covered only in time slot 4). These initial orientations actually give an optimal solution to each of the three problems: min-avg dark, min-max dark, and min-longest dark. □

Summary of results:

- min-avg dark, min-max dark and min-longest dark problems are NP-hard (see Theorem 1);
- min-max dark and min-longest dark problems are NP-hard to approximate (see Theorem 2);
- We give a greedy algorithm (GA) for these problems (see Algorithm 1);
- GA delivers a constant-factor approximation for the min-avg dark problem (see Theorem 3);
- GA gives a "double-approximation" for min-max dark and min-longest dark problems (see Theorems 4 and 5);
- We provide a decentralized version of GA (see Subsection III-E).

C. NP-Hardness Results

We start by defining a decision problem called 0-dark and prove it to be NP-complete. The NP-hardness of our problems follows from the NP-completeness of 0-dark decision problem.

0-dark: Given a set of RDS with fixed locations and a set of points, is it possible to set the sensors' initial-orientation such that each point has zero dark time?

Lemma 1 (NP-hardness of 0-dark)**.** *The 0-dark problem is NP-complete.*

Proof: We reduce 3SAT to the 0-dark problem with each RDS having only two orientations. Given an instance of planar-3SAT:

- Create a point for each clause. We refer to these as *clause-points*.
- Create one RDS for each variable. Each RDS has two sensing-regions (and orientations). The first sensing region covers all the clause-points wherein the variable appears as a positive literal, while the second sensing-region covers all the clause-points where the variable appears as a negated literal.
- Create an additional RDS, C_1 with two orientations. The first sensing-region of C_1 covers all the clause-points, while the second sensing-region of does not cover any point.

It is easy to see that the above is a correct reduction of the 3SAT problem to the 0-dark problem. ■

Note that the above reduction creates arbitrary complex sensing regions. However, with some effort, we can reduce planar 3SAT to the 0-dark problem, wherein the sensing regions are only cone-shaped. Since the 0-dark problem is a common sub-problem of our three problems, the following theorem follows.

Theorem 1 (NP-hardness)**.** *The min-avg dark, min-max dark, and min-longest dark problems are NP-hard.*

We now prove that both min-max dark and min-longest dark problems are NP-hard to approximate.

Algorithm 1: Greedy Algorithm (GA)

```
while there are points with dark time > 0 and
      there are sensors not yet oriented
    Select a sensor (that has not been selected yet) and an
    initial orientation that reduces the total dark time of all
    points the most;
```

Theorem 2 (NP-hardness of approximation)**.** *The min-max dark and min-longest dark problems are NP-hard to approximate.*

Proof: An approximation algorithm for the min-max dark (or the min-longest dark) problem can be easily used to solve the 0-dark instance constructed in Lemma 1, since in that instance each clause-point can only have a dark time of 0 or 1. ■

D. Greedy Approximation Algorithm

We now present the Greedy Algorithm which yields an "approximate" solution for each of the three addressed problems.

Greedy Algorithm (GA) The Greedy Algorithm (GA) works in iterations. In each iteration, it considers all the RDS for which the initial-orientation has not been determined yet. For each of these RDS, GA considers all possible initial-orientations, and picks the (RDS, initial orientation) pair that reduces the total dark time of all points by the maximum amount. The algorithm terminates when the initial-orientation of all the RDS has been determined, or the total dark time of all the points is zero. See Algorithm 1.

Example 2. Consider the instance described in Example 1 and the corresponding Figure 2(a). We are given 5 points, and 3 RDS, s_1, s_2, and s_3, each having 4 orientations. Before GA starts, no RDS has been selected/oriented, and all points have a dark time of 4. In the first iteration of GA, any choice of (initial-orientation, RDS) pair will reduce the dark time of all points by 1 each. Let s_1 be picked with its shown initial-orientation in time slot 1. Now, each point has a total dark time of 3. In the next iteration, s_3 is selected with the initial orientation set upwards. Finally, s_2 is selected and its initial-orientation is set downwards. The above gives a GA solution as shown in Figure 2(b). □

Performance Guarantee. We now analyze the performance of the GA algorithm, starting with its approximation factor for the min-avg dark problem. In particular, we analyze GA in terms of the total "coverage-time" of all points, which is the sum of the number of time slots each point is covered by an RDS's active region.

Theorem 3 (Performance of GA for min-avg dark)**.** *GA runs in $O(nt^2 \log nt)$ time, where n is the number of RDS and t is the total number of given points. Moreover, GA solution has a total coverage-time of at least 43% of the optimal coverage-time possible.*

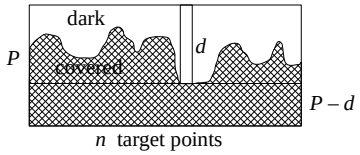


Fig. 3. An example of an optimal distribution of the coverage benefit, where P is the period and d is the min-max dark time. The coverage benefit of the optimal is at least $n(P-d)$. Note that dark and covered areas are drawn in a compact way, but they are not necessarily in this way.

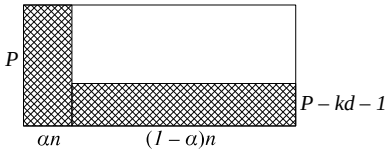


Fig. 4. The extreme case of the coverage benefit distribution.

We omit the proof of the above theorem, since it is very similar to the proof of Theorem 2 of [1]. Below, we analyze the performance of GA for the remaining two problems. Since these problems have been shown to be NP-hard to approximate, we resolve to a "double-approximation" factor.

Theorem 4 (Performance of GA for min-max dark)**.** *Let d be the minimum max dark-time. GA yields a solution such that at least αn points have a dark time of at most kd each, where n is the number of points, $\alpha = (1+(k-0.43)d-0.57P)/(kd+1)$, and P is the total number of orientations (length of the period).*

Proof: For this problem, coverage is the dual of dark time. In particular

$$\text{min-max dark} = P - \text{max-min coverage}.$$

The idea is to think in terms of coverage benefit. Since any optimal algorithm gives min-max dark = d, then max-min coverage = $P-d$, and hence the benefit of the optimal is at least $n(P-d)$. See Figure 3. Now, if a point has more dark time than kd, its dark time is at least $kd+1$. Or, in other words, it is covered for at most $P-kd-1$ time slots. In an extreme situation, the most unbalanced case is when αn points are covered for all t slots, and all other points are covered for $P-kd-1$ slots (any other configuration would give more "good" points). This gives the smallest number points with at least $P-kd$ coverage (i.e. at most kd dark time). See Figure 4. Carrying on the computation we get that

$$\alpha n P + (1-\alpha)n(P-kd-1) = 0.43n(P-d).$$

Finally, solving for α, we have

$$\alpha = \frac{1+(k-0.43)d-0.57P}{kd+1}.$$

■

We left α and k as parameters, so we can have several different trade-offs. For example, if $k=2$, and $P=8$, $\alpha = (1.57d-3.56)/(2d+1)$, or if $k=1$ and $P=4$, $\alpha = (0.57d-0.28)/(d+1)$.

This idea can also be used for the min-longest dark time.

Theorem 5 (Performance guarantee for min-longest dank)**.** *Let d be the minimum longest dark-time. GA yields a solution such that at least βn points have a dark-time of at most kd each, where n is the number of points, $\beta = (kd+0.57+0.43P/d-P)/(kd+1)$, and P is the length of a period.*

Proof: The proof is similar to the one for the previous theorem. However, in this case, we only know that d is the *longest* dark interval, but there might be many other dark intervals of length $d-1$. Then, the coverage benefit for the point with the longest dark interval is at least $(P-d)/d$ (in the really unlucky case of dark intervals of length $d-1$ alternated with 1 time slot of coverage). So we can only claim that the optimal has a coverage benefit of at least $n(P-d)/d$. Carrying on the computation, we obtain the value of β. ■

E. Extensions

Minimizing Total Dark Time of an Area. GA can also be used (with the same performance guarantees) to minimize the dark time over a given area, rather than a given set of points. Essentially, coverage of an area requires dividing the given area into "subregions" as in our previous work [9]; a subregion is defined as a set of points in the plane that are covered by the same set of sensing regions. The number of such subregions can be shown to be polynomial in the total number of cone-shape sensing regions in the system.

Distributed GA. In a distributed environment, conceptually, each yet-uncovered point reduces its dark time by selecting and fixing the initial orientation of a sensor that covers it and has the highest benefit at that stage. The above process continues until all points have the minimum dark time. To facilitate the above, a point (or a subregion) is "owned" by the highest-ID sensor that can cover it using one of its sensing regions. Thus, each RDS reduces the dark time of the points it owns, through selection and orientation of its near-by sensors.

IV. Placement and Orientation of Sensors

We now address the problem of placing (in a given polygon) and orienting the minimum number of RDS so as to reduce the dark time each point to 0. In this section, we define the problem in terms of coverage of a polygon rather than a set of points.

Problem 2. (Place and orient RDS) Place the minimum number of RDS in a polygon, possibly with holes, and specify their initial-orientation, so to reduce the dark time of all polygon's points to zero.

Summary of results:

- Problem 2 is NP-hard (see Theorem 6);
- We develop two different approximation algorithms (see Theorems 7 and 8).

Theorem 6. *Problem 2 is NP-hard.*

Proof: Note that Problem 2 is a more general problem that the well-known NP-hard Art Gallery Problem [11]. In fact, if sensing-regions are disks of infinite radii, then an RDS is equivalent to a guard of the Art Gallery Problem. ■

Adapting Technique of [2]. We will now show how to adapt the recent method of [2] by Agarwal et al. to Problem 2. The problem considered in [2] is to place minimum number of omni-directional sensors to cover at least $(1-\varepsilon)$ fraction of the given polygon's area. Their method works for sensing regions such as cones, disk, etc. Below, we give a brief description of their technique.

Description of [2]'s technique. The method in [2] works in two stages. In the first stage, a set of landmark points is obtained. These points have the property that if all of them are covered by a set of placed sensors, then $(1-\varepsilon)$ of the polygon's area is covered by this set of sensors. The second stage consists of actually placing a number of sensors to cover the polygon. This second stage is done by a simple greedy approach or a more efficient Monte Carlo algorithm.

Landmark positions, in the first stage, are determined using ε-nets [24]. Given a set of points and a collection of sets (of points), an ε-net is a subset of the points such that every "large-enough" set contains at least one of the points in the subset. A standard way to obtain ε-nets is by random sampling [24], but the random sampling approach is effective only when the given sets have a constant "VC-dimension." Intuitively, the VC-dimension measures the regularity of the set system. The formal definition is as follows.

Definition 4. (VC dimension) A set X is *shattered* by $\mathcal{C}$ if for each $Y \subseteq X$, there exists a set $S \in \mathcal{C}$ such that $X \cap S = Y$. The *VC dimension* is the cardinality of the largest set that can be shattered by $\mathcal{C}$. □

First Approximation Algorithm. In order to be able to apply [2]'s method to Problem 2, we need to assume/show that the sensing-regions of RDS have a constant VC dimension. Now, it is known [25] that sets of simple polygons have a VC-dimension of at most 23, while cone-shaped regions can be shown to have a VC-dimension of at most 8. If an RDS has P orientations, then the VC dimension is at most $23P$ (or $8P$ in the case of RDS with coverage regions with the shape of cones).

Finally, [2]'s method is applied to static sensing-regions. To apply their technique to Problem 2, we look at Problem 2 in the 3D space as follows.

3D interpretation. The 2D region and the time dimension can be visualized together as a 3D space. In this 3D space, each initial-orientation of an RDS yields a 3D sensing-region, and each point in the original 2D space becomes a vertical line. Finally, the original input polygon becomes a 3D object whose base is the given polygon. Now, the original Problem 2 of placing and orienting RDS so as to reduce the dark time of each point in the given polygon to zero, translates to the problem of covering the entire 3D object (corresponding to the polygon) using the 3D objects corresponding to the initial-orientations of RDS, under the constraint that for each RDS, only one 3D object can be used. □

The method of [2] can now be applied directly to the above 3D problem, yielding a way to cover at least $(1-\varepsilon)$ of the volume of the given space. Thus, at most ε of the volume is left uncovered. Hence, the average dark time is εP. This proves the following theorem.

Theorem 7. *The method of [2] when applied to Problem 2 yields a solution that, for any parameter ε, guarantees an average dark-time of εP where P is the length of the period, using at most $O(M \log M)$ RDS, where M is the optimal number of RDS required.*

Second Approximation Algorithm. Another way of applying the method of [2] is as follows. We start with first finding the landmark points, such that covering them with RDS will cover at least $(1-\varepsilon)$ of the area of the given polygon. Then, we translate the problem to the 3D space as before. Now, in the 3D view, each landmark point translates to a vertical line. Then, we can find a placement of the 3D objects corresponding to RDS so to cover all these landmarks. This guarantees that for at least $(1-\varepsilon)$ of the area of the polygon the dark time is 0. This proves the following theorem.

Theorem 8. *The method of [2] when applied to Problem 2, as shown above, yields a solution that, for any parameter ε, guarantees 0-dark time for at least $(1-\varepsilon)$ of the area of the given polygon using at most $O(M \log M)$ RDS, where M is the optimal number of RDS required.*

Running time. The running time of both the above methods is related to the running time of the algorithm in [2]. Let v be the number of vertices of the input polygon, and ς be the number of holes in the input polygon. Let h be the optimum number of RDS needed to reduce the dark-time to 0. Let us define $\lambda = (hP/\varepsilon)(1+\log(1+\varsigma))$, where P is the length of the period. Then the running time is $O(vm^2(1+\varsigma)\log(mv))$, where $m = O(\lambda \log \lambda \log(1/\varepsilon))$.

V. Barrier Coverage

We now turn our attention to the problem of barrier coverage. In some real applications, it might not be necessary to cover all points, as long as the sensors are able to detect all possible intruders that are trying to cross the monitored area. We restrict our attention to intruders moving along rectilinear parallel paths and, without loss of generality, we assume that they move vertically upwards. Note that our notion of barrier coverage corresponds to weak barrier coverage in the literature (see for example [16]). We start by defining the problem and then we will present our results.

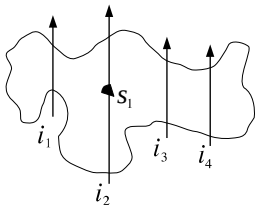


Fig. 5. An example of a polygon and 4 intruders crossing it. Intruder i_2 can be detected by the sensor s_1, if its initial orientation is set properly.

A. Problem Definition and Summary of Results

In this subsection, we first give a formal definition of intruders, then we define the barrier coverage problem, and finally we show the outline the results in this section.

Definition 5. (Intruder) An *intruder* is a moving point that we want to detect with the RDS. An intruder is *detected* by a sensor if it is within one of its active sensing region in any time instant. We assume that intruders move vertically, upwards, and at given constant speed[1]. □

Note that intruders can start crossing the monitored area in different positions or at different times. Our algorithms allow intruders to start at any continuous location, but the NP-harness proof considers a discrete number of intruders.

As we did for the problems in the previous sections, we consider both the problem of orientation of already-placed sensors, and the problem of sensor's placement and orientation.

Problem 3. (Barrier coverage) We consider two variants:

- *Orient Placed RDS*: Given a simple polygon and the locations of a set of RDS, determine their initial orientation so to detect the maximum quantity of intruders.
- *Place and Orient RDS*: Given a simple polygon, place the minimum number of RDS and specify their initial orientation, so to detect all intruders.

Example 3. Figure 5 shows an example of a polygon and 4 intruders that are crossing it. Intruder i_2 can be detected by the sensor s_1, if its initial orientation is set properly. □

Summary of results:

- The problem of orientation of placed RDS is NP-hard (see Theorem 9);
- We provide a constant-factor approximation for the problem of orientation of placed RDS (see Theorem 10);
- We provide a constant-factor approximation for the problem of RDS placement and orientation (see Theorem 11);
- We provide a polynomial time algorithm for for the problem of orientation of placed RDS for the special case of intruders moving at infinite speed (see Theorem 12).

[1]The speed of the intruders is not necessarily related to the rotation speed of the sensors.

B. NP-Hardness

We now prove that the problem of orientation of placed RDS is NP-hard in its discrete version by reducing SAT to it.

Theorem 9. *The problem of orientation of placed RDS is NP-hard.*

Proof: The proof consists of reducing SAT to the problem of detecting all possible intruders. Two intruders are considered different, if they start at different positions or in different time slots. The reduction is as follows:

- Let n be the number of variables, and m be the number of clauses.
- Let the monitored area be a rectangle of width m units and height $n+1$ units. Each intruder takes 1 time slot to cross 1 unit of the height, and he needs $n+1$ time slots to cross it completely. We can think as if the rectangle is divided in bands and intruders take 1 time slot to cross each band.
- Let the period be of $2n+2$ time slots.
- Clauses are represented by intruders. For each clause i, we create a set of intruders, starting at unit i of the basis of the rectangle. Each of these sets is composed by $n+1$ intruders, which start one for each time slot for the first $n+1$ time slots. During the remaining $n+1$ time slots, there are no intruders.
- Variables are represented by RDS. Each sensor has $2n+2$ sensing regions corresponding to $2n+2$ orientations. RDS have a sensing region, that we call "positive", in one time slot, and a sensing region, that we call "negative", $n+1$ slots apart. In all other $2n$ time slots they do not cover any significant part of the rectangle. Both positive and negative sensing regions cover completely the lowest band of the rectangle. In addition, they both have some vertical strips that span the full height of the rectangle. For the positive region these vertical trips are located in correspondence of entrance points of intruders corresponding to clauses in which the variable appears positive. Similarly, for the negative region, the vertical strips are in correspondence of intruders corresponding to clauses in which it appears negated.

In order to detect the highest number of intruders, no two RDS should overlap their base band. This means that we can easily detect all intruders that start in n time slots. So the only real difficulty is to cover the intruders that are starting in the remaining time slot. Since all RDS span the height of the rectangle for some entry points, potentially it is possible to detect all intruders in the remaining time slot. However if an algorithm finds a solution for that, it would also give a solution for SAT. ■

C. Orientation of Placed Sensors

We now give a method for setting the RDS's initial orientations that guarantees to detect at least 43% of the intruders that any optimal algorithm can detect. We start by describing a geometrical interpretation for the set of possible intruders.

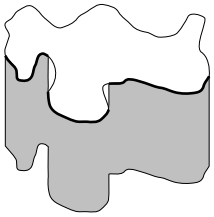
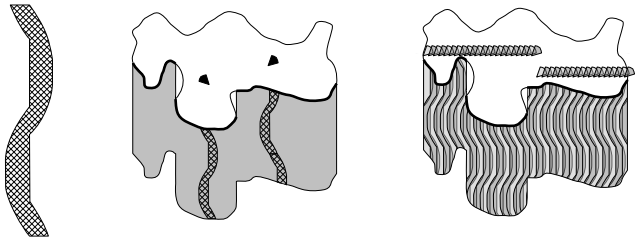


Fig. 6. The intruders polygon below the input polygon. The lowest set of points on the boundary of the input polygon are highlighted.



(a) (b) (c)

Fig. 7. (a) A possible shape for a *covering stripe*, which is the subregion that an RDS covers on the intruders polygon. The precise shape depends on the size of the cone, the speed of rotation, and the speed of the intruders. The picture drawn here spans the length of a period only, but in reality this figure is infinitely repeated above and below.
(b) An example of covering stripe corresponding to particular RDS locations and initial orientations.
(c) A solution constructed by replicating covering stripes at regular intervals.

Intruders polygon. A "stream" of intruders can be modeled as a polygon, which we call *intruders polygon*. The intruders polygon, denoted with $\mathcal{I}$, is constructed as follows and it is placed immediately below the input polygon $\mathcal{P}$. Consider the "lowest" set of points on the boundary of $\mathcal{P}$, i.e. those points that do not have any other point of $\mathcal{P}$ below them (note that these points are not always connected). These points constitute the upper boundary of $\mathcal{I}$ (after connecting them if necessary). Let h be the distance traveled by the intruders during a period. The lower boundary of $\mathcal{I}$ is obtained by shifting down its upper boundary by h. The region enclosed inside the upper boundary and the lower boundary of $\mathcal{I}$ constitutes the intruders polygon. See Figure 6 for an example. With this interpretation, a set of intruders that start crossing $\mathcal{P}$ at different times is equivalent to a set of intruders that start at the same time at different vertical positions of the intruders polygon.

43%-approximation algorithm. With the geometrical interpretation above, the problem of orientation of placed RDS becomes the problem of covering the largest area of the intruders polygon. Depending on the location of an RDS, for each of its starting orientation we can easily determine which intruders it can detect, and hence which subregion of the intruders polygon it can "cover". So, the problem of orientation of placed RDS is equivalent to coverage with directional sensors, and it can be solved with a greedy algorithm similar to GA (see Algorithm 1). At each iteration, this greedy algorithm selects the (RDS, initial orientation) pair that "covers" the largest yet uncovered region of the intruders polygon.

Theorem 10. *The above described algorithm gives a 43%-approximation to the problem of orientation of placed RDS and it runs in polynomial time.*

The proof is omitted, since it is very similar to the proof of Theorem 2 of [1].

Distributed algorithm. The above algorithm can be distributed in a similar way as GA (see Subsection III-E). The area of intruder polygon is partitioned, and each partition is assigned to the highest-ID sensor close to it. Each RDS is responsible to select and orient its near-by sensors to increase the coverage of its owned sub-regions. This process continues until the intruder polygon has been covered completely or until there are no more sensors to select and orient.

D. Placement and Orientation of Sensors

The idea of looking at the intruders polygon can also be used to construct a constant-factor approximation algorithm for the problem of RDS placement and orientation[2]. For this particular problem, our solution requires the assumption that RDS sensing regions have the shape of a cone. We start by giving a geometrical interpretation to the set of intruders that an RDS can detect.

Covering stripe. Once we know the size of the cone, the speed of rotation, and the speed of the intruders, we can efficiently construct the shape of the subregion that an RDS covers on the intruders polygon. We call this shape the *covering stripe*. See Figure 7(a) for an example of a covering stripe. Different starting orientations correspond to different vertical shifts of a covering stripe. Covering stripes are infinitely repeated vertically, but when they are placed over the intruders polygon, they get restricted to the height of the intruders polygon. See Figure 7(b) for an example of covering stripes corresponding to particular RDS locations and initial orientations. The problem of RDS placement and orientation reduces to the problem of placement of covering stripes so to cover completely the intruders polygon.

O(1)-approximation algorithm. A solution can be constructed by replicating covering stripes at regular intervals, so to cover completely the intruders polygon (see Figure 7(c) for an example). Once a covering stripe is placed, it is easy to reconstruct an RDS position and initial orientation that can generate it[3].

Theorem 11. *The above described algorithm gives a $O(1)$-approximation to the problem of RDS placement and orientation and it runs in polynomial time.*

Proof: In the solution given by the algorithm above, each point in the intruders polygon is covered by a constant number

[2]Note that we do not know whether this problem is NP-hard.

[3]Note that there can be multiple (RDS position, initial orientation) pairs that can generate the same covering stripe, but we just need reconstruct one of them.

of covering stripes. This implies that this method is a constant-factor approximation to the problem of RDS placement and orientation. ■

E. Infinite-Speed Intruders

Even if in general the problem of orientation of already-placed RDS is NP-hard, for the special case of intruders traveling at infinite speed, it can be solved with a polynomial time algorithm, if we assume that RDS sensing regions have the shape of a cone. This might seem an abstract formulation, but it could have applications in cases in which the speed of the intruders is much higher than the RDS speed of rotation.

The idea behind the following algorithm is that, from the point of view of an infinite-speed intruder, RDS are just 1D segments, corresponding to the projection of an RDS sensing region over a line perpendicular to the intruder's trajectory.

We start by considering a very simple case and we will extend it later. RDS are formed by 2 cones, each of which is half disk. The axis of all RDS is vertical. This implies that, when a sensing region is projected on the basis of the rectangle, it covers a segment of length equal to its radius, and it is either on right or on the left of its center. All RDS are synchronized, but they can have different initial orientations. Since RDS have only two discretized orientations, the period is $P = 2$.

The idea is to use dynamic programming, and determine the RDS initial orientation from left to right, using the fact that an RDS's best orientation is influenced only by a polynomial number of other sensor's orientations before itself.

Algorithm for infinite-speed intruders. Sort the RDS by the rightmost point they can cover. Consider one by one the RDS from left to right. Build partial solutions by adding one RDS at a time, and for each of its two starting orientations store the best starting orientations for all the RDS on its left. Suppose we already considered $i-1$ RDS, and we are looking for the best orientations to associate to the initial orientation of RDS i. Let c_i be the center of RDS i and r be its radius. Look at the RDS whose rightmost point falls inside the interval $(c_i - r, c_i + r)$, and let j be the leftmost of them. Let c_j be its center. Consider all the RDS whose center is between c_j and c_i. Among all the possible initial orientations of these RDS, consider only the 9×2 "significant"[4] ones (as given by the lemma below). These 9×2 significant orientations can be built by scanning from left to right the RDS between i and j. Combine each of these with the 2 solutions of RDS j, and the two orientations of RDS i. Store the 2 best combinations for RDS i, and proceed to RDS $i+1$. □

We are now going to prove that the above algorithm runs in polynomial time. We start by defining the concept that a segment is completely covered by a set of segments, and then we will prove a lemma used in the main theorem.

[4]An orientation is significant if the projected 1D segment is not completely covered by other projected 1D segments. Note that we should look at both time slots, and this gives the factor of 2.

Definition 6. (Completely-cover) A set of segments S *completely covers* a segment t if there is no point of t that is not covered by any of the segments in S. □

Lemma 2. *Consider a larger 1D segment L of size $|L|$, and a set S of smaller 1D segments s_i, each of size $|s|$ (i.e. $|s| < |L|$) that are all completely contained in L. There can be at most $3\lceil |L|/|s| \rceil$ 1D segments of S, that are not completely covered by any other segments of S.*

Proof: Consider two segment s_i and s_j that are at distance less than $|s|$ each other, with s_i on the left of s_j. In between s_i and s_j there can be at most two other segments that are not completely covered by any other segment. Namely, there can be one segments slightly shifted to the right of s_i and one slightly shifted to the left of s_j. In L there can be at most $\lceil |L|/|s| \rceil$ segments that do not touch each other, and in between each pair, there can be at most other 2, for the argument above. So this gives a total of at most $3\lceil |L|/|s| \rceil$. ■

Theorem 12. *The algorithm above, solves the problem of orientation of placed RDS for the special case of intruders traveling at infinite speed.*

Proof: The purpose of this proof is to show that the above dynamic programming algorithm runs polynomial time and uses polynomial space, so we will not go into the details of a tight analysis.

The largest distance between c_j and c_i is $3r$. According to the above lemma there can be at most $3\lceil 3r/r \rceil = 9$ 1D segments that are not completely covered by other segments. Potentially, there can be $n-2$ other RDS between c_j and c_i, but only 9 in each time slot are not completely covered by other ones in the 1D projection. We need to consider all possible ways of selecting 9×2 RDS, for a total of $\binom{n-2}{9\times 2} = O(n^{9\times 2})$ possibilities. These $O(n^{18})$ configurations can be enumerated by sweeping from left to right. Since we need to do this for every RDS, the total cost is $O(n^{19})$.

Note that we only need quadratic storage, because for each RDS, and each of its two orientations, we need to store the best initial orientations of all the previous RDS. ■

Extension to RDS with more orientations. The algorithm and its analysis extend easily to RDS with more discrete orientations. The lemma still applies, but this time, we need to use the smallest projection of an orientation. In general, the number of significant orientations is cP, where c is a constant that depends on the ratio between the smallest and the largest projection. This gives a $O(n^{cP+1})$ algorithm.

VI. Conclusions and Open Problems

We considered some new problems that arise from the use of rotating directional sensors. In particular we studied the problem of orientation of already-placed sensors to minimize the dark time (i.e. uncovered time) of a given set of points. We considered the problem of placement and orientation of the minimum number of sensors in a polygon to reduce to 0 the dark time of every point in the polygon. We also analyzed the

problems of orientation of already-placed sensors and sensor's placement and orientation to guarantee a barrier coverage.

We suggest the following open problems:

- Provide a PTAS for the min-avg dark problem.
- Study the complexity of the problem of placement and orientation of RDS for weak barrier.
- Extend the barrier coverage results to strong barrier coverage (in which intruders can move along any type of paths).

ACKNOWLEDGMENTS

We wish to thank Prof. Joseph Mitchell for his useful comments. The work was supported in part by NSF Grants IIS-0713186, CNS-0721701, and CNS-0721665.

REFERENCES

[1] G. Fusco and H. Gupta, "Selection and orientation of directional sensors for coverage maximization," in *Proceedings of 6th Annual IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2009.

[2] P. K. Agarwal, E. Ezra, and S. K. Ganjugunte, "Efficient sensor placement for surveillance problems," in *DCOSS 2009*, ser. LNCS, vol. 5516, 2009, pp. 301–314.

[3] J. Ai and A. A. Abouzeid, "Coverage by directional sensors in randomly deployed wireless sensor networks," *Journal of Combinatorial Optimization*, vol. 11, pp. 21–41, 2006.

[4] E. Hörster and R. Lienhart, "On the optimal placement of multiple visual sensors," in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks (VSSN)*, 2006, pp. 111–120.

[5] G. Fusco and H. Gupta, "ε-net approach to sensor k-coverage," in *4th International Conference on Wireless Algorithms, Systems and Applications (WASA '09)*, ser. LNCS, vol. 5682, 2009, pp. 104–114.

[6] K. Charkrabarty, S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transaction on Computers*, 2002.

[7] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor ad-hoc networks," in *Proceedings of the International Conveference of Communication (ICC)*, 2001.

[8] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Peas: A robust energy conserving protocol for long-lived sensor networks," in *Proceedings of the International Conference on Distributed Computing Systems*, 2003.

[9] H. Gupta, Z. Zhou, S. Das, and Q. Gu, "Connected sensor cover: Self-organization of sensor networks for efficient query execution," *ACM/IEEE Transactions on Networking (TON)*, vol. 14, no. 1, 2006.

[10] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of the IEEE INFOCOM*, 2003.

[11] J. O'Rourke, *Art Gallery Theorems and Algorithms*, ser. International Series of Monographs on Computer Science. New York, NY: Oxford University Press, 1987, vol. 3.

[12] T. C. Shermer, "Recent results in art galleries," in *Proceedings of the IEEE*, vol. 80, 1992, pp. 1384–1399.

[13] J. Urrutia, *Art Gallery and Illumination Problems,* In Handbook of Computational Geometry. J.R. Sack and J. Urrutia eds., 2000, ch. 22, pp. 973–1027.

[14] S. K. Ghosh, *Visibility Algorithms in the Plane*. Cambridge University Press, 2007.

[15] D. Gage, "Command control for many-robot systems," in *Proceedings of the 19th Annual AUVS Technical Symposium (AUVS '92)*, 1992.

[16] S. Kumar, T. Lai, and A. Arora, "Barrier coverage with wireless sensors," in *Proceedings of the 11th International Conference on Mobile Computing and Networking (MobiCom '05)*, 2005, pp. 284–298.

[17] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar, "Reliable density estimates for coverage and connectivity in thin strips of finite length," in *In proceedings of the 13th annual ACM international conference on Mobile computing and networking (MobiCom '07)*, 2007.

[18] A. Chen, S. Kumar, and T. H. Lai, "Designing localized algorithms for barrier coverage," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking (MobiCom '07)*, 2007.

[19] B. Liu, O. Dousse, J. Wang, and A. Saipulla, "Strong barrier coverage of wireless sensor networks," in *Proceedings of the 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, 2008.

[20] A. Chen, T. H. Lai, and D. Xuan, "Measuring and guaranteeing quality of barrier-coverage in wireless sensor networks," in *Proceedings of The ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, 2008.

[21] A. Saipulla, C. Westphal, B. Liu, and J. Wang, "Barrier coverage of line-based deployed wireless sensor networks," in *Proceedings of the IEEE INFOCOM*, 2009.

[22] G. Yang and D. Qiao, "Barrier information coverage with wireless sensors," in *Proceedings of the IEEE INFOCOM*, 2009.

[23] L. Zhang, J. Tang, and W. Zhang, "Strong barrier coverage with directional sensors," in *Proceedings of IEEE Global Telecommunications Conference (Globecom)*, 2009.

[24] D. Haussler and E. Welzl, "Epsilon-nets and simplex range queries," *Discrete & Computational Geometry*, vol. 2, pp. 127–151, 1987.

[25] P. Valtr, "Guarding galleries where no point sees a small area," *Israel Journal of Mathematics*, vol. 104, pp. 1–16, 1998.