Optimal Movement of Mobile Sensors for Barrier Coverage of a Planar Region
(Extended Abstract)

B. Bhattacharya[*][¶] M. Burmester[†] Y. Hu[*] E. Kranakis[‡][¶] Q. Shi[*]
A. Wiese[§]

March 16, 2008

Abstract

Intrusion detection, area coverage and border surveillance are important applications of wireless sensor networks today. They can be (and are being) used to monitor large unprotected areas so as to detect intruders as they cross a border or as they penetrate a protected area. We consider the problem of how to optimally move mobile sensors to the fence (perimeter) of a region delimited by a simple polygon in order to detect intruders from either entering its interior or exiting from it. We discuss several related issues and problems, propose two models, provide algorithms and analyze their optimal mobility behavior.

1 Introduction

Monitoring and surveillance are two of the main applications of wireless sensor networks today. Typically, one is interested in monitoring a given geographic region either for measuring and surveying purposes or for reporting various types of activities and events. Another important application concerns critical security and safety monitoring systems. One is interested in detecting intruders (or movements thereof) around *critical* infrastructure facilities and geographic delimiters (chemical plants, forests, etc). As a matter of fact, since the information security level of the monitoring system might change rapidly because of hostile attacks targeted at it, research efforts are currently underway to extend the scalability of wireless sensor networks so that they can be used to monitor international borders as well. For example, [11] reports the possibility of using wireless sensor networks for replacing traditional barriers (more than a kilometer long) at both the building and estate level. Also, "Project 28" concerns the construction of a virtual fence as a way to complement a physical fence that will include 370 miles of pedestrian fencing and 300 miles of vehicle barrier (see [8] which reports delays in its deployment along the U.S.-Mexico border).

[*]School of Computing Science, Simon Fraser University, Vancouver, BC, Canada.
[†]Department of Computer Science, Florida State University, Talahassee, Florida, USA.
[‡]School of Computer Science, Carleton University, Ottawa, Ontario, Canada.
[§]Institut für Mathematik, Technische Universität Berlin, Berlin, Germany.
[‖]Research supported in part by NSERC and MITACS.

To begin, we say that a point is *covered* by a sensor if it is within its range. In this paper we will use the concept of *barrier coverage* as used in [11] and which differs from the more traditional concept of *full coverage.* In the latter case one is interested in covering the entire region by the deployment of sensors, while in the former all crossing paths through the region are covered by sensors. Thus, one is not interested in covering the entire deployment region but rather to detect potential intruders by guaranteeing that there is no path through this region that can be traversed undetected by an intruder as it traverses the border. Clearly, barrier coverage is an appropriate model of movement detection that is more efficient than full coverage since it requires less sensors for detecting intruders (this is the case, for example, when the width of the deployment region is three times the range of the sensors).

In [3] the authors consider the problem of how individual sensors can determine barrier coverage *locally.* In particular, they prove that it is possible for individual sensors to locally determine the existence of barrier coverage, even when the region of deployment is arbitrarily curved. Although local barrier coverage does not always guarantee global barrier coverage, they show that for thin belt regions, local barrier coverage almost always provides global barrier coverage. They also consider the concept of *L-local barrier coverage* whereby if the bounding box that contains the entire trajectory of a crossing path has length at most L then this crossing path is guaranteed to be detected by at least one sensor.

Motivation, model and problem statement. Motivated from the works of [3] and [11], in this paper we go beyond by asking a different question not examined by any of these papers. More precisely, given that the mobile sensors have detected the existence of a crossing path (e.g., using any of the above algorithms) how do they reposition themselves *most efficiently* within a specified region so as to repair the existing *security hole* and thereby prevent intruders.

Further, we stipulate the existence of a geometric planar region (the critical region to be protected) delimited by a simple polygon and mobile sensors are lying in the interior of this polygon. We consider a set of mobile sensors (or robots) lying within a region that can move autonomously in the plane. Each sensor has knowledge of the region to be barrier-covered, of its geographic location and can move from its starting position p to a new position p' on the perimeter of this polygon. For each sensor, we look at the distance $d(p, p')$ between the starting and final positions of the sensors, respectively, and investigate how to move the sensors within this region so as to optimize either the *minimum sum* or the *minimum maximum* of the distances covered by the respective sensors. In the sequel we investigate the complexity of this problem for various types of regions and types of movement of the mobile sensors.

Related work. An interesting research article is by [1] which surveys the different kinds of holes that can form in geographically correlated problem areas of wireless sensor networks. The authors discuss relative strengths and short-comings of existing solutions for combating different kinds of holes such as coverage holes, routing holes, jamming holes, sink/black holes, worm holes, etc. [2] looks at critical density estimates for coverage and connectivity of thin strips (or annuli) of sensors. In addition, [5] and [6] design a distributed self deployment algorithm for coverage calculations in mobile sensor networks and consider various performance metrics, like coverage, uniformity, time and distance traveled till the algorithm converges. Related is also the research on art gallery theorems (see [14]) which is concerned with finding the minimal number of positions for guards or cameras so that every point in a gallery is observed by at least one guard or camera.

In addition to the research on barrier coverage already mentioned there is extensive literature on detection and tracking in sensor networks. [12] considers the problem of event tracking and sensor resource management in sensor networks and transforms the detection problem into finding and tracking the cell that contains the point in an arrangement of lines. [9] addresses the problem of tracking multiple targets using a network of communicating robots and stationary sensors by introducing a region-based approach for controlling robot deployment. [16] considers the problem of accurate mobile robot localization and mapping with uncertainty using visual landmarks. Finally, related to the problem of detecting a path through a region that can be traversed undetected by an intruder is the paper [15] which gives necessary and sufficient conditions for the existence of vertex disjoint simple curves homotopic to certain closed curves in a graph embedded on a compact surface.

Outline and results of the paper. Section 2 gives the formal model on a circle and defines the min-max (minimizing the maximum) and min-sum (minimizing the sum) problems for a set of sensors within a circle or a simple polygon. Section 3 looks at the simpler one dimensional case and derives simple optimal algorithms for the case the sensors either all lie on a line or on the perimeter of circle. Section 4 and Section 5 are the core of the paper and provide algorithms solving the min-sum and min-max problems, respectively. That is, in Section 4, an $O(n^{3.5}\log n)$-time algorithm for the min-max problem on a circle and an $O(mn^{3.5}\log n)$-time algorithm for the min-max problem on a simple polygon are proposed (m is the number of edges of the simple polygon). Our approximation algorithms for min-sum problems on a circle or a simple polygon are presented in Section 5. Finally, Section 6 gives the conclusion.

2 Preliminaries and Formal Model

First we describe the formal model on a circle and provide the basic definitions and preliminary concepts.

2.1 Optimization on a circle

The simpler scenario we envision concerns n mobile sensors which are located in the interior of a unit-radius circular region. A set of n sensors are located inside the disk. Further, assume that the sensors are location aware (i.e., they know their geometric coordinates) and also know the location of the center of the disk. We would like to move all the sensors from their initial positions to the perimeter of the circle so as to 1) form a regular n-gon, and 2) minimize the total/maximum distance covered.

The motivation for placing the sensors on the perimeter is because it provides the most efficient way to protect the disk from intruders. Observe that when all n sensors lie equidistant on the vertices of a regular n-gon, they each need to cover a circular arc of size $2\pi/n$ so as to be able to monitor the entire perimeter. Using elementary trigonometry, it follows easily that the transmission range of each sensor must be equal to $r = \sin(\pi/n)$.

More formally, for n given sensors in positions $A_1, A_2, \ldots, A_n$, respectively, which move to new positions $A'_1, A'_2, \ldots, A'_n$ at the corners of a regular n-gon the total distance covered is $\sum_{i=1}^{n} d(A_i, A'_i)$. Every sensor moves from its current position A_i to a new position A'_i. It is clear that the sum is minimized when each sensor moves to its new position in a straight line.

The reason for having the sensors at the corners of a regular n-gon is because this is evidently the optimal final arrangement that will enable them to detect intruders (i.e., by being equidistant on the perimeter). Thus, since the final position $A'_1 A'_2 \cdots A'_n$ of the sensors forms a regular n-gon it is clear that all possible solutions can be parametrized by using a single angle $0 \leq \theta \leq 2\pi$. However, a difficulty arises in view of the fact that we must also specify a permutation $\sigma : \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$ of the sensors such that the i-th sensor moves from position $A_{\sigma(i)}$ to the new position A'_i.

Let the n sensors have coordinates (a_i, b_i), for $i = 1, 2, \dots, n$. Let us parametrize the regular polygon with respect to the angle of rotation say θ. The n vertices of the regular n-gon that lie on the perimeter of the disk can be described by

$$(a_i(\theta), b_i, (\theta)) = \left(\cos\left(\theta + \frac{(i-1)2\pi}{n}\right), \sin\left(\theta + \frac{(i-1)2\pi}{n}\right)\right), \text{ (for } i = 1, 2, \dots, n), \tag{1}$$

respectively, where $(a_i(\theta), b_i(\theta))$ are the vertices of the regular n-gon when the angle of rotation is θ.

Minimizing the sum The optimization problem is $\min_\theta S_n(\theta)$, where the function $S_n(\theta)$ is defined by $S_n(\theta) := \sum_{i=1}^{n} \sqrt{(a_i - a_i(\theta))^2 + (b_i - b_i(\theta))^2}$, as a function of the angle θ. This of course assumes that the i-th sensor is assigned to position $(\cos(\theta + (i-1)2\pi/n), \sin(\theta + (i-1)2\pi/n))$ on the perimeter. In general, we have to determine the minimum over all possible permutations σ of the sensors. If for a given angle θ and permutation σ we define $S_n(\sigma, \theta) := \sum_{i=1}^{n} \sqrt{\left(a_{\sigma(i)} - a_i(\theta)\right)^2 + \left(b_{\sigma(i)} - b_i(\theta)\right)^2}$ then the more general optimization problem is $\min_{\sigma,\theta} S_n(\sigma, \theta)$.

Minimizing the maximum The previous problem was concerned with minimizing the sum of the distances of the robots to their final destinations. In view of the fact that the robots are moving simultaneously it makes sense to ask for minimizing the maximum of the distances of the robots to their final destinations $\max_{1\leq i\leq n} d(A_i, A'_i)$. The optimization problem is $\min_\theta M_n(\theta)$, where $M_n(\theta) := \max_{1\leq i\leq n} \sqrt{(a_i - a_i(\theta))^2 + (b_i - b_i(\theta))^2}$, as a function of the angle θ. This of course assumes that the i-th sensor is assigned to position $(\cos(\theta + (i-1)2\pi/n), \sin(\theta + (i-1)2\pi/n))$ on the perimeter. In general, we have to determine the minimum over all possible permutations σ. If for a given permutation σ we define the following maximum $M_n(\sigma, \theta) := \max_{1\leq i\leq n} \sqrt{\left(a_{\sigma(i)} - a_i(\theta)\right)^2 + \left(b_{\sigma(i)} - b_i(\theta)\right)^2}$ then the general optimization problem is $\min_{\sigma,\theta} M_n(\sigma, \theta)$.

2.2 Optimization on a simple polygon

We similarly define the problem of minimizing the sum and minimizing the maximum on a simple polygon as follows.[*] Let P be a simple polygon. (From now on, a polygon is always assumed to be simple.) We denote the boundary of P by ∂P. We assume that ∂P is oriented

[*]Although the approach proposed later (parametric search) will also work for arbitrary simple curves, we refrain from such a generalization so as to avoid unnecessary complications.

in the clockwise (also called positive) direction. For any two points $a, c \in \partial P$, we write $\hat{\pi}_P(a, c)$ to denote the set of all points $b \in \partial P$ such that when starting after a in positive direction along ∂P, b is reached before c. Let $p_0, p_1, \ldots, p_{m-1}$ denote the vertices on P ordered in the positive direction. The edges of ∂P are $e_0, e_1, \ldots, e_{m-1}$, where edge e_i has endpoints p_i and p_{i+l}, where $0 \leq i < m$ (i.e., the indices are computed modulo m; e.g., $p_0 = p_m$). We denote by $l(e_i)$ the length of edge $e_i, 0 \leq i < m$, and by $\hat{d}_P(a, b)$ the length of $\hat{\pi}_P(a, b)$ for any two points a and b on ∂P (called *polygonal distance* between a and b). Let $L(P) = \sum_{i=0}^{m-1} l(e_i)$.

We are given n mobile sensors which are located in the interior of P. Each sensor has the knowledge of its geometric coordinates and the simple polygon (i.e., the geometric coordinates of all vertices $p_i, 0 \leq i < n$ and the clockwise ordering of these vertices). The objective is to move all the sensors from their initial positions to ∂P such that 1) the polygonal distance between any two consecutive sensors on the polygon is $L(P)/n$, and 2) minimize the total/maximum distance covered. We postulate that if n given sensors are located at positions $A_1, A_2, \ldots, A_n$, and the destination positions are $A_1, A_2, \ldots, A_n$, respectively, then $\hat{d}_P(A'_i, A'_{i+1}) = L(P)/n, 0 \leq i < n$.

3 Mobile Sensors in One Dimension

In this section we look at the one dimensional problem and provide efficient algorithmic solutions. In particular, since optimization for the minimum maximum is similar (and simpler than the two dimensional analogue) we provide algorithms only for the minimum sum.

3.1 Sensors on a line segment

In this model we suppose that the sensors can move on a line segment. Further, instead of protecting a circular range the sensor can now protect an interval of a given size centered at the sensor. Consider the minimum sum optimization problem for the case of n sensors on a line. Without loss of generality assume the segment has length 1 and let the n sensors be at the initial locations $x_0 < x_1 < \cdots < x_{n-1}$, respectively. The destination locations are $\frac{i}{n-1}$, for $i = 0, 1, \ldots, n-1$.

Theorem 1 *The optimal arrangement is obtained by moving point x_i to position $\frac{i}{n-1}$, for $i = 0, 1, \ldots, n-1$, respectively.*

3.2 Sensors on the perimeter of a circle

In this model we suppose that the sensors can move on the perimeter of a circle. Further, instead of protecting a circular range the sensor can now protect an arc on the perimeter of a given size centered at the sensor. The same idea as for a line segment should work for the case of a unit circle when the sensors lie on the perimeter of the circle. The main difficulty here is that we no longer have a unique destination. Instead, we can parametrize all possible destinations of the n points by $\phi + \frac{2j\pi}{n}$, for $j = 0, 1, \ldots, n-1$, using a fixed angle $0 \leq \phi < \frac{2\pi}{n}$.

Theorem 2 *There is an algorithm that computes an optimal cost arrangement of the sensors.*

3.2.1 When the sensors' movement is in the interior of the circle

In this model we suppose that the sensors and their destination positions are located on the perimeter of a circle and the sensors can move to their destination positions along a straight line. Here we have the same result as the one in Lemma 14 (see Appendix). Its proof is based on the fact there is an optimal solution in which one sensor does not move at all.

Theorem 3 *There is an algorithm that computes an optimal cost arrangement of the sensors.*

4 Min-max problem in 2D

In this section we study the problem of minimizing the maximum (min-max problem) on a unit circle and a simple polygon, and provide efficient algorithmic solutions.

4.1 On a circle

Let $\lambda^*_{m,C}$ be the optimal value of the min-max problem on a circle $\mathcal{C}$, i.e., $\lambda^*_{m,C} = \min_{\sigma,\theta} M_n(\sigma,\theta)$. It is easy to see that $\lambda^*_{m,C}$ is no more than the diameter of the circle $\mathcal{C}$, i.e., $\lambda^*_{m,C} \leq 2$. In this section we propose a parametric-searching approach [13] to compute $\lambda^*_{m,C}$.

A non-negative value λ is *feasible* in the min-max problem if all the sensors can move from their initial positions to the perimeter of the circle such that the new positions form a regular n-gon and the maximum moving distance is no more than λ, otherwise λ is *infeasible*. Clearly, the min-max problem is to compute the minimum feasible value, which is equal to $\lambda^*_{m,C}$.

The remaining part of this section is organized as follows. We first show that a feasibility test of a given value $\lambda(0 \leq \lambda \leq 2)$ can be performed in time $O(n^{3.5})$. Then, a parametric-searching approach for the min-max problem is presented, which runs in $O(n^{3.5}\log n)$ time.

4.1.1 Algorithm to check the feasibility test of λ

For each $i, 1 \leq i \leq n$, we construct a circle of radius λ centered at position A_i, denoted by $\mathcal{C}_i$. If a circle $\mathcal{C}_i$ for some i is contained in $\mathcal{C}$, then λ is infeasible since sensor A_i cannot move to the perimeter of $\mathcal{C}$ within distance λ. We therefore assume that for each $i, 1 \leq i \leq n$, either circle $\mathcal{C}_i$ contains $\mathcal{C}$ or $\mathcal{C}_i$ intersects with $\mathcal{C}$.

Refer to Figure 3. For each $i, 1 \leq i \leq n$, we denote by Q_i the arc of $\mathcal{C}$ that lies in $\mathcal{C}_i$. Let $q_{i(1)}, q_{i(2)}$ be the angles of two endpoints of arc Q_i in clockwise order, $i = 1, \ldots, n$. We let $q_{i(1)} = 0$ and $q_{i(2)} = 2\pi$ if $\mathcal{C}_i$ contains $\mathcal{C}$.

The following property is important to our algorithm for the feasibility test of λ. Its proof is omitted here.

Lemma 4 *A given non-negative value λ is feasible if and only if there exists a regular n-gon on the perimeter of $\mathcal{C}$ such that one of its corner points is an endpoint of arc Q_i for some $i(1 \leq i \leq n)$.*

The algorithm (Algorithm **Check**) to check the feasibility of λ is described in the appendix. It is easy to see that the sorting in the first step can be done in $O(n\log n)$ and the computation of $S_j, j = 1, \ldots, 2n+1$ can be done in $O(n^2)$. In the third step, the process might try all $O(n)$ regular n-gons. For each regular n-gon, it takes $O(n^{2.5})$ time (see [7]). Therefore, we have the following lemma.

Lemma 5 *Whether a given positive value λ is feasible in the min-max problem can be determined in $O(n^{3.5})$ time.*

4.1.2 A parametric-searching approach

Our approach for the solution to the min-max problem is to run Algorithm *Check* parametrically, which has a single parameter λ, without specifying the value of $\lambda^*_{m,C}$ a priori. Note that for a fixed value of the parameter, the algorithm is executed in $O(n^{3.5})$ steps. Imagine that we start the algorithm without specifying a value of the parameter λ. The parameter is restricted to some interval which is known to contain the optimal value $\lambda^*_{m,C}$. (Initially, we may start with the interval $[0, 2]$.) As we go along, at each step of the algorithm we update and shrink the size of the interval, ensuring that it includes the optimal value $\lambda^*_{m,C}$. The final interval contains $\lambda^*_{m,C}$ and any value in it is feasible. Therefore, the minimum value of the final interval is the optimal value $\lambda^*_{m,C}$.

Theorem 6 *The min-max problem on a circle can be solved in $O(n^{3.5} \log n)$ time.*

Note that our algorithm can be easily extended to the model in which all sensors are arbitrarily located on the plane (not restricted to the interior of the circle $\mathcal{C}$).

4.2 On a simple polygon

The parametric-searching approach for a circle (described in section 4.1) should work for the case of a polygon where the destination positions of all sensors lie on the perimeter of the polygon. The main difficulty here is that to check the feasibility of a positive value λ, there might be $O(m)$ isolated polygonal chains of ∂P within the circle $\mathcal{C}_i$ (of radius λ centered at position A_i) for each sensor A_i. In other words, for a given positive value of λ each sensor will contribute $O(m)$ candidate sets of n destination positions on P instead of at most two candidate sets on a circle. Hence, whether a given positive value λ is feasible in the min-max problem on a simple polygon can be determined by solving $O(mn)$ matching problems of size n. Therefore, the feasibility test of the min-max problem on a simple polygon can be solved in $O(mn^{3.5})$ time.

Theorem 7 *The min-max problem on a simple polygon can be solved in $O(mn^{3.5} \log n)$ time where m is the size of the simple polygon.*

5 Approximation algorithms for the min-sum problem in 2D

In this section we discuss the problem of minimizing the sum (min-sum problem) on a circle and a simple polygon, and provide approximation solutions for them.

5.1 On a circle

Let $\lambda^*_{s,C}$ be the optimal value of the min-sum problem on a circle, i.e., $\lambda^*_{s,C} = \min_{\sigma,\theta} S_n(\sigma, \theta)$. We present two approximation algorithms for the min-sum problem. One algorithm (labeled as the *first approach*) has an approximation ratio $\pi + 1$ (section 5.2). The other one (labeled as the *second approach*) uses the first approach as a subroutine to obtain lower and upper bounds of $\lambda^*_{s,C}$ and has an approximation ratio $1+\epsilon$, where ϵ is an arbitrary constant (Section 5.3).

More notations are introduced as follows. Let $\hat{d}_C(x, y)$ denote the arc distance between two points x and y on the boundary if the cycle $\mathcal{C}$ and let $\hat{\pi}_C(x, y)$ denote the arc of length

$\hat{d}_C(x, y)$ between x and y. For a point x on $\mathcal{C}$, we denote by $\hat{Q}_x(r)$ the arc consisting of all points y on $\mathcal{C}$ such that $\hat{d}_C(x, y) \leq r$.

For each $i = 1, \ldots, n$, let ω_i be the smallest distance between A_i and a point on the cycle $\mathcal{C}$, and we denote by B_i the point on $\mathcal{C}$ such that the distance $d(A_i, B_i) = \omega_i$. We note that for each $i = 1, \ldots, n$, B_i is unique if A_i is not located at the center of $\mathcal{C}$. In the case when A_i is located at the center of $\mathcal{C}$, an arbitrary point on $\mathcal{C}$ is selected to be B_i. Let $\Omega = \sum_{i=1}^{n} \omega_i$. Obviously, we have the following lemma.

Lemma 8 $\Omega \leq \lambda^*_{s,C}$.

5.2 The first approach

The first approach (called Algorithm 1) consists of three steps.

Step 1 For each sensor $A_i, 1 \leq i \leq n$, compute B_i.
Step 2 Compute a destination regular n-gon for the set of n points $B_1, \ldots, B_n$, and find the optimal arrangement of the n points to the vertices of the n-gon, by using the algorithm for sensors on the perimeter of a circle described in Section 3.2. Let B'_i be the destination vertex of $B_i, 1 \leq i \leq n$.
Step 3 Move A_i to $B'_i, 1 \leq i \leq n$, and compute $S^1_n = \sum_{i=1}^{n} d(A_i, B'_i)$.

In section 3.2 we showed that step 2 of Algorithm 1 can be implemented in $O(n^2)$ time. Thus the above algorithm can be solved in $O(n^2)$ time.

5.2.1 Approximation bound of Algorithm 1

In this section, we show that S^1_n computed by the first approach is bounded by $(\pi + 1) \times \lambda^*_{s,C}$. Suppose that A'_i is the destination of sensor $A_i, i = 1, \ldots, n$, in an optimal solution. Clearly, $A'_1, \ldots, A'_n$ lie on $\mathcal{C}$ and form a regular n-gon. Obviously, $\sum_{i=1}^{n} \hat{d}_C(B_i, B'_i) \leq \sum_{i=1}^{n} \hat{d}_C(B_i, A'_i)$ since $\{B'_1, \ldots, B'_n\}$ is an optimal solution for the one dimensional min-sum problem with the input $\{B_1, \ldots, B_n\}$. The following lemma is easy to show.

Lemma 9 *For any two points x, y on $\mathcal{C}$, $\hat{d}_C(x, y) \leq \frac{\pi}{2} \times d(x, y)$.*

Theorem 10 *Algorithm 1 can be implemented in $O(n^2)$ time and its approximation ratio is no more than $\pi + 1$.*

5.3 The second approach

The following lemma is crucial for the second approach (Algorithm 2) described in the appendix.

Lemma 11 *In an optimal solution, there exists at least one sensor $A_i (1 \leq i \leq n)$ such that its destination A'_i on $\mathcal{C}$ is on the arc $\hat{Q_{B_i}}(\frac{\pi}{2} \times \frac{S^1_n}{n})$.*

Proof. It is clear that $S^1_n \geq \lambda^*_{s,C}$. Let A'_i be the destination of sensor A_i in an optimal solution, $i = 1, \ldots, n$. Then there is at least one sensor, say $A_k (1 \leq k \leq n)$, such that the distance $d(A_k, A'_k)$ is no more than $\frac{S^1_n}{n}$. According to Lemma 9, all points on $\mathcal{C}$ with the distance to A_k of no more than $\frac{S^1_n}{n}$ lie on the arc $\hat{Q_{B_k}}(\frac{\pi}{2} \times \frac{S^1_n}{n})$ (recall that B_k is the point on $\mathcal{C}$ closest to A_k), which completes the proof of Lemma 11. ■

5.3.1 Analysis of the second approach

First, it is evident that the running time of the second approach is determined by the time needed for solving $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1) \in O(\frac{n}{\epsilon})$ bipartite matching problems.

According to Lemma 11, there exists an optimal solution in which one of the corners of the corresponding regular n-gon is located at a point on the arc $\hat{Q_{B_k}}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ for some k, $1 \leq k \leq n$. In Step 2, the arc $\hat{Q_{B_k}}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ is partitioned into $\lceil \frac{1}{\epsilon'} \rceil$ pieces, and therefore, the length of each piece is no more than $\frac{\pi S_n^1 \epsilon'}{n}$ (note that the length of $\hat{Q_{B_k}}(\frac{\pi}{2} \times \frac{S_n^1}{n})$ is $\frac{\pi S_n^1}{n}$). Since all possible values of k are considered, the difference between S_n^2 (computed by the second approach) and $\lambda_{s,C}^*$ (the optimal cost) is no more than $n \times \frac{1}{2} \times \frac{\pi S_n^1 \epsilon'}{n} = \frac{\pi S_n^1 \epsilon'}{2} = \frac{S_n^1 \epsilon}{\pi+1} \leq \epsilon \lambda_{s,C}^*$, byTheorem 10). Therefore, we have the following theorem.

Theorem 12 *The approximation ratio of Algorithm 2 is no more than $1 + \epsilon$ for a given constant ϵ, and the running time of the second approach is $O(\frac{1}{\epsilon} n^4)$.*

5.4 On a simple polygon

Let $\lambda_{s,P}^*$ be the optimal value of the min-sum problem on a polygon P. In this subsection we present an approximation algorithm for the min-sum problem on P, which has an approximation ratio $1 + \epsilon$ (ϵ is an arbitrary constant).

Our algorithm for a simple polygon is very similar to the second approach for a circle. In the second approach for a circle, we use Algorithm 1 as a subroutine to obtain lower and upper bounds of $\lambda_{s,C}^*$. However, our approximation algorithm for a simple polygon will use the solution for the min-max problem on the polygon to obtain lower and upper bounds of $\lambda_{s,P}^*$. Let $\lambda_{m,P}^*$ be the optimal value of the min-max problem on P. It is easy to see that $\lambda_{m,P}^* \leq \lambda_{s,P}^* \leq n \times \lambda_{m,P}^*$. Our algorithm for a simple polygon P and details of proofs are described in the appendix.

Theorem 13 *The approximation ratio of the approach for a simple polygon is no more than $1 + \epsilon$ for a given constant ϵ, and the running time of the second approach is $O(\frac{1}{\epsilon} mn^5)$.*

6 Conclusion and Open Problems

In this paper we gave an algorithm for solving the min-max problem and a PTAS (Polynomial Time Approximation Scheme) for the min-sum problem in both one and two dimensions. Although it is unknown whether the min-sum problem is NP-hard, we conjecture that it can be solved in polynomial time. Evidence for this also comes from experimental results on finding the number of different counter-clockwise orderings of n sensors on the perimeter of a circle when we sweep a regular n-gon along the perimeter (see Appendix). In addition, several other variants of the problem on simple polygons and regions are of interest for further investigation, including k-barrier coverage, regions with holes, and various types of sensor placements and motions. Thus, in Subsection 2.2, in order to minimize the number of sensors used when scanning the perimeter one should take into account sections already scanned. For example, this is the case if the polygon is a narrow rectangle of height less than the range of a sensor; this in itself is an interesting optimization problem which is worth of further investigation. Also of interest is to refine the sensor motion model, the network model, and the communication model in order to enable effective intrusion detection and barrier coverage.

For example, the communication model becomes crucial when assuming the sensors either do not have knowledge of the region or do not know their coordinates.

References

[1] N. Ahmed, S.S. Kanhere, and S. Jha. The holes problem in wireless sensor networks: a survey. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(2):4–18, 2005.

[2] P. Balister, B. Bollobas, A. Sarkar, and S. Kumar. Reliable density estimates for coverage and connectivity in thin strips of finite length. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 75–86, 2007.

[3] A. Chen, S. Kumar, and T.H. Lai. Designing localized algorithms for barrier coverage. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 63–74, 2007.

[4] R. Cole. Slowing down sorting networks to obtain faster sorting algorithms. *Journal of the ACM (JACM)*, 34(1):200–208, 1987.

[5] N. Heo and PK Varshney. A distributed self spreading algorithm for mobile wireless sensor networks. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 3, 2003.

[6] N. Heo and PK Varshney. Energy-efficient deployment of Intelligent Mobile sensor networks. *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, 35(1):78–92, 2005.

[7] J.E. Hopcroft and R.M. Karp. An $n^{2.5}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[8] S. S. Hu. 'Virtual Fence' along border to be delayed. *Washington Post*, Thursday, February 28, 2008.

[9] B. Jung and G.S. Sukhatme. Tracking Targets Using Multiple Robots: The Effect of Environment Occlusion. *Autonomous Robots*, 13(3):191–205, 2002.

[10] H. W. Kuhn. The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[11] S. Kumar, T.H. Lai, and A. Arora. Barrier coverage with wireless sensors. *Wireless Networks*, 13(6):817–834, 2007.

[12] J. Liu, P. Cheung, F. Zhao, and L. Guibas. A dual-space approach to tracking and sensor management in wireless sensor networks. *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 131–139, 2002.

[13] N. Megiddo. Applying Parallel Computation Algorithms in the Design of Serial Algorithms. *Journal of the ACM (JACM)*, 30(4):852–865, 1983.

[14] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc. New York, NY, USA, 1987.

[15] A. Schrijver. Disjoint circuits of prescribed homotopies in a graph on a compact surface. *Journal of Combinatorial Theory Series B*, 51(1):127–159, 1991.

[16] S. Se, D. Lowe, and J. Little. Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks. *The International Journal of Robotics Research*, 21(8):735, 2002.

Appendix

Proof. (Theorem 1) It is clear that the n possible destinations are of the form $\frac{i}{n-1}$, for $i = 0, 1, \ldots, n-1$. For any point x let $d(x)$ be its destination. Recall that our goal is to determine the destination of each point x so that the sum

$$\sum_x |x - d(x)| \tag{2}$$

is minimized. We call the route from x to $d(x)$ the directed path on the line from x to $d(x)$ which is traced by the sensor located at x. The length of this path is equal to $|x - d(x)|$. For any two pints $x < y$ we say that their paths *inversely ovelap* if they are in opposite directions and they also overlap. As depicted in Figure 1, it is clear that for two points $x < y$ there

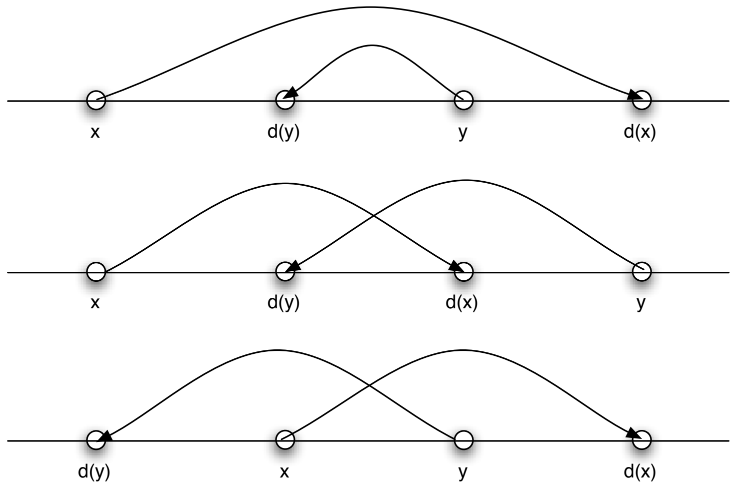

Figure 1: Three possible cases for overlapping paths between the two points $x < y$.

are three possible cases of overlapping paths. It is easy to show that in each case we can improve on the optimization sum (Equation 2) by merely switching the destinations of x and y, respectively. We call such a configuration between two points *inversion*.

Now consider the configuration resulting by moving point x_i to position $\frac{i}{n-1}$, for $i = 0, 1, \ldots, n-1$, respectively. Observe that for this arrangement there exist no two pairs $x < y$ of sensors which inversely overlap. Moreover, we claim that any two possible assignments that do not have pairs $x < y$ that inversely overlap must have identical sum (Equation 2). Indeed, the sum of the top configuration depicted in Figure 2 is $(d(y) - x) + (d(x) - y)$ and of the bottom it is $(d(x) - x) + (d(y) - y)$. Since $(d(y) - x) + (d(x) - y) = (d(x) - x) + (d(y) - y)$ the result of the claim follows. This completes the proof of the theorem. ■

Proof. (Theorem 2) First of all we prove the following lemma.

Lemma 14 *The minimal cost assignment of destinations for the given points, must be among the n assignments*

$$(x_0, x_1, \ldots, x_j, \ldots, x_{n-1}) \rightarrow \left(x_i, x_i + \frac{2\pi}{n}, \ldots, x_i + j\frac{2\pi}{n}, \ldots, x_i + (n-1)\frac{2\pi}{n}\right),$$

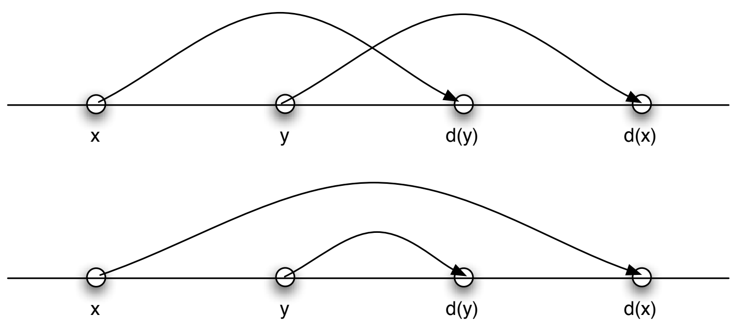

Figure 2: A non-overlapping transformation leaves the sum unchanged.

for $i = 0, 1, \ldots, n-1$.

Proof. (Lemma 14) Consider the minimal cost assignment based on the angle x. Suppose that in the corresponding n-gon none of the points $x_0, x_1, \ldots, x_j, \ldots, x_{n-1}$ is a vertex. Now rotate this n-gon and observe that either its clockwise or its counter clockwise rotation decrease the cost or remains the same. ■

Now we prove the main result. Suppose we are given n points $x_0, x_1, \ldots, x_{n-1}$ in this cyclical order along the perimeter of the circle. The main steps of the algorithm are the following.

1. For each point $x_i \in \{x_0, x_1, \ldots, x_{n-1}\}$, map all points x_j to destinations $x_i + j\frac{2\pi}{n}$, for $j = 0, 1, \ldots, n-1$ (this implies that x_i is mapped to itself).
2. Select the point $x_i \in \{x_0, x_1, \ldots, x_{n-1}\}$ that optimizes the sum

$$\sum_{j=0}^{n-1} \left| x_j - x_i - j\frac{2\pi}{n} \right| \tag{3}$$

It is easy to show that in this mapping each point x is mapped to a destination $d(x)$ such that $|x - d(x)| \leq \pi$. Now just like the case of a line (imagine Figure 1 on a circle) we can show that the optimal assignment cannot have paths that inversely overlap. Moreover, according to Figure 2 (imagine it on a circle) the assignments will be optimal each for the given starting position x_i. ■

Proof. (Theorem 3) Assume to the contrary that all sensors move in an optimal solution. For each $i, 1 \leq i \leq n$, let α_i be the angle of rotation of the sensor A_i to its new position A_i': positive angle denotes clockwise rotation, while negative angle denotes counter-clockwise rotation. For sensors which are moved to the opposite side of the circle we assume that $\alpha = -\pi$. From the law of cosine we derive that $d(A_i, A_i') = \sqrt{2 - 2\cos\alpha_i}$. Let $f(\alpha) = \sqrt{2 - 2\cos\alpha}$. Trivially, $f(\alpha)$ is concave over $[0, \pi)$ and $[-\pi, 0]$. We define $\alpha' := \max_{\alpha_i \leq 0} \alpha_i$ and $\alpha'' := \min_{\alpha_i > 0} |\pi - \alpha_i|$. Note that $\alpha' < 0$ since all $\alpha_i \neq 0$ and $\alpha'' > 0$ since all $\alpha_i < \pi$. We define the function $g : [a, b] \to \mathbb{R}$:

$$g(\beta) := \sum_{i=1}^{n} f(\alpha_i + \beta).$$

The value of $g(\beta)$ represents the total cost of the solution which we get by taking the original solution α_i and changing the positions of the sensors on the perimeter by the angel β. Note that g is concave for all $\beta \in (a,b)$. We want to show that there is a $\beta \neq 0$ such that $g(\beta) < g(0)$. If $g'(0) \neq 0$ then there is clearly a β with $g(\beta) < g(0)$. This is a contradiction. If $g'(0) = 0$ then $\beta = 0$ is a local maximum since $g''(0) < 0$. So there is a β with $g(\beta) < g(0)$. This is a contradiction.

Since Lemma 14 is still valid in the model using the straight-line distance metric, the same idea as for the arc distance metric can be applied here. ■

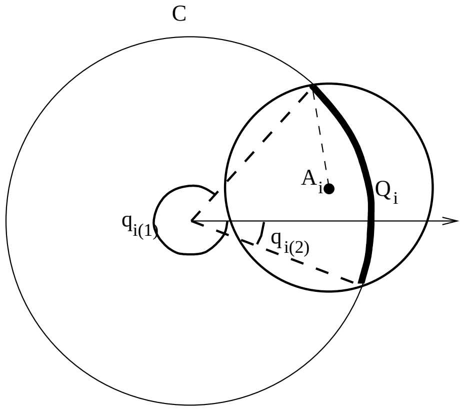

Figure 3: Notations $Q_i, q_{i(1)}$ and $q_{i(2)}$.

Proof. (Theorem 6) The whole approach for the min-max problem is described as follows. The sorting step (the first step) of Algorithm *Check* with unknown $\lambda^*_{m,C}$ can be performed by solving $O(\log^2 n)$ feasibility tests if a parallel sorting network that runs $O(\log n)$ steps with n processors is used [13]. Actually, it can be reduced to $O(\log n)$ feasibility tests if Cole's result is applied [4]. The second step does not need to solve feasibility tests since the order of endpoints of arcs $Q_i, 1 \leq i \leq n$, provides enough information to compute sets $S_j, 1 \leq j \leq 2n+1$.

We now show how to run the third step with unknown $\lambda^*_{m,C}$. We can see that no feasibility test is needed for the steps 3(b) and 3(c). However, to find the intervals where the angles $q'_j, (q'_j + \frac{2\pi}{n}) \bmod 2\pi, \ldots, (q'_j + (n-1)\frac{2\pi}{n}) \bmod 2\pi, j = 1, \ldots, 2n$ lie, we solve some feasibility tests since these angles and the endpoints of intervals are low-degree polynomials of unknown value $\lambda^*_{m,C}$.

It is inefficient to execute step 3(c) one by one. To speed up the computation, we can compute them in a parallel way. For each angle, we can locate the interval containing it in a binary-search fashion. Clearly, it can be done in $O(\log n)$ steps, and each step is a comparison between two low-degree polynomials that can be achieved by solving a constant number of feasibility tests.

Therefore, the computation of the step 3(a) for all possible $j \in [1, 2n]$ can be done in $O(\log n)$ parallel steps by using $O(n^2)$ processors (there are $2n^2$ queries in total), where each step is a comparison between two low-degree polynomials with unknown $\lambda^*_{m,C}$. By applying the idea of Cole [4], the computation of the step 3(a) for all possible $j \in [1, 2n]$ can be done by solving $O(\log n)$ feasibility tests. ■

Proof. (Theorem 10)

$$\begin{aligned}
S^1_n &= \sum_{i=1}^{n} d(A_i, B'_i) \\
&\leq \sum_{i=1}^{n} [d(A_i, B_i) + \hat{d}_C(B_i, B'_i)] \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \sum_{i=1}^{n} \hat{d}_C(B_i, A'_i) \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \frac{\pi}{2} \times \sum_{i=1}^{n} d(B_i, A'_i) \text{ (Lemma 9)} \\
&\leq \sum_{i=1}^{n} d(A_i, B_i) + \frac{\pi}{2} \times \sum_{i=1}^{n} [d(B_i, A_i) + d(A_i, A'_i)] \\
&\leq (\pi + 1) \times \lambda^*_{s,C} \text{ (Lemma 8).}
\end{aligned}$$

■

Algorithm Check

1. The first step is to sort the angles of endpoints of arcs $Q_i, 1 \leq i \leq n$. Let $q'_1, \ldots, q'_{2n}$ be the angles in increasing order. These angles partition the interval $[0, 2\pi]$ into at most $2n+1$ pairwise disjoint intervals, denoted by $I_1, \ldots, I_{2n+1}$.
2. For each interval $I_j, 1 \leq j \leq 2n+1$, we determine the set of sensors, denoted by S_j, that lie within distance λ to its corresponding arc on $\mathcal{C}$.
3. In the third step, we do the following for a regular n-gon with rotation $q'_j, i = 1, \ldots, 2n$.
   (a) It is easy to see that the angles of corner points of such regular n-gon are $q'_j, (q'_j + \frac{2\pi}{n}) \bmod 2\pi, \ldots, (q'_j + (n-1)\frac{2\pi}{n}) \bmod 2\pi$. We compute the intervals where these angles lie. Let $B_i, i = 1, 2, \ldots, n$ be the corner points.
   (b) Construct a bipartite graph between the set of corner points of the regular n-gon and the set of sensors. An edge is linked between corner point B_k and sensor A_i if $d(A_i, B_k) \leq \lambda$ $(1 \leq i, k \leq n)$. The bipartite graph can be obtained from the steps 2 and 3(a) .
   (c) Check if there exists a perfect matching. If it is so, terminate the process and return "Feasible".
4. Return "Infeasible" .

Algorithm 2

Step 1 Using Algorithm 1, compute S^1_n defined above.

Step 2 For each $i = 1, \ldots, n$, find the arc $\hat{Q_{B_i}}(\frac{\pi}{2} \times \frac{S^1_n}{n})$ and compute a set of points that partitions the arc into $\lceil \frac{1}{\epsilon'} \rceil$ pieces of equal length where $\epsilon' = \frac{2\epsilon}{\pi(\pi+1)}$.

Step 3 Clearly, there are $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1)$ points in total. For each point x, construct a regular n-gon P_x such that one of the corners of P_x is located at x, and find the optimal arrangement of the n sensors $(A_1, \ldots, A_n)$ to the vertices of the n-gon by solving a weighted bipartite matching problem. (The Hungarian method to solve the weighted matching problem in a complete bipartite graph of size n takes $O(n^3)$ time (see [10])).

Step 4 Among all $n \times (\lceil \frac{1}{\epsilon'} \rceil + 1)$ regular n-gons thus constructed, find the one with the minimum cost (denoted by S_n^2) and output the optimal arrangement of the n sensors to the vertices of the n-gon.

Experimental Results on the complexity of the min-sum problem

It is not known whether the min-sum problem can be solved optimally. Two related problems which could help clarify the issue are the following.

1. Given a counter-clockwise ordering of n sensors on the perimeter of $\mathcal{C}$, solve the min-sum problem.
2. Find the number of different counter-clockwise orderings of n sensors on the perimeter of $\mathcal{C}$ when we sweep a regular n-gon along the perimeter of $\mathcal{C}$.

Table 1 shows our experimental results to resolve the second problem described above. The

Table 1: Experimental result for the number of different orderings (over 20 test sets)

# of sensors (n)	# of regular n-gons (t)	average # of orderings
10	1,000	5.40
20	2,000	8.40
30	3,000	11.75
40	4,000	15.45
50	5,000	18.10
60	6,000	19.80
70	7,000	24.50
80	8,000	26.60
90	9,000	30.85
100	10,000	35.55

first column in the table represents the number n of sensors contained in $\mathcal{C}$ (the n sensors are randomly generated), the second column represents the number t of regular n-gons being tested (i.e., the arc distance between two consecutive regular n-gons is $\frac{2\pi}{t*n}$), and the third column represents the average number of different counter-clockwise orderings of the n sensors to the vertices of the t n-gons (for each value of n, 20 different sets of n sensors are generated). From this table, we can see that there are no more than n different counter-clockwise orderings in the experiment.

Min-sum algorithm on a simple polygon

Step 1 Using the approach for the min-max problem on P, compute $\lambda^*_{m,P}$ described above.

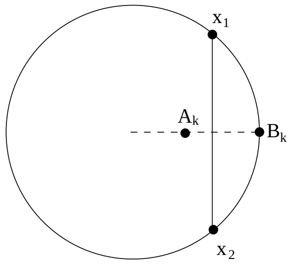

Figure 4: Lemma 11, $\hat{d}_C(x_1, B_k) = \hat{d}_C(x_2, B_k) = \frac{\pi}{2} \times \frac{S_n^1}{n} \Rightarrow d(x_1, x_2) \geq \frac{2S_n^1}{n}$ (Lemma 9).

Step 2 For each i, j where $1 \leq i \leq n$ and $0 \leq j < n$, find the sub-edge $e'_{i,j}$ of edge e_j that is within the circle of radius $\lambda^*_{m,P}$ centered at position A_i, and compute a set of points that partitions the the sub-edge into $\lceil \frac{n}{\epsilon} \rceil$ pieces of equal length.

Step 3 Clearly, there are $mn \times (\lceil \frac{n}{\epsilon} \rceil + 1) \in O(\frac{mn^2}{\epsilon})$ points in total. For each point x, construct a set of n positions on P such that one of them is located at x and the polygonal distance between any two consecutive positions is $L(P)/n$, and find the optimal arrangement of the n sensors $(A_1, \ldots, A_n)$ to the set of n positions by using the algorithm [10].

Step 4 Among all $O(\frac{mn^2}{\epsilon})$ candidate sets of n positions thus constructed, find the one with the minimum cost.

It is evident that the running time of the above approach is determined by the time needed for solving $O(\frac{mn^2}{\epsilon})$ weighted bipartite matching problems.

Proof. (Theorem 13) The reason why the approximation ratio of the above approach is bounded by $1 + \epsilon$, is as follows. Since $\lambda^*_{s,P} \leq n \times \lambda^*_{m,P}$, there is at least one sensor whose moving distance to its destination is no more than $\lambda^*_{m,P}$ in an optimal solution. Let A_i be one such sensor and its destination position lies on edge e_j in that optimal solution. In Step 2, the sub-edge $e'_{i,j}$ is partitioned into $\lceil \frac{n}{\epsilon} \rceil$ pieces, and therefore, the length of each piece is no more than $\frac{2\epsilon\lambda^*_{m,P}}{n}$. Since all possible values of i and j are considered, the difference between the value computed by the above approach and $\lambda^*_{s,P}$ (the optimal cost) is no more than

$$n \times \frac{1}{2} \times \frac{2\lambda^*_{m,P}}{n} = \epsilon\lambda^*_{m,P} \leq \epsilon\lambda^*_{s,P}.$$

■