

5 Increasing Detection Performance of Surveillance Sensor Networks

Nelly Litvak^{1*} Muhammad Umer Altaf² Alina Barbu²
Sudhir Jain³ Denis Miretskiy¹ Leila Mohammadi⁴
Ertan Onur² Jos in 't panhuis⁵ Julius Harry Sumihar²
Michel Vellekoop¹ Sandra van Wijk⁵ Rob Bisseling^{6†}

Abstract

We study a surveillance wireless sensor network (SWSN) comprised of small and low-cost sensors deployed in a region in order to detect objects crossing the field of interest. In the present paper, we address two problems concerning the design and performance of an SWSN: optimal sensor placement and algorithms for object detection in the presence of false alarms. For both problems, we propose explicit decision rules and efficient algorithmic solutions. Further, we provide several numerical examples and present a simulation model that combines our placement and detection methods.

Keywords: sensor deployment, detection probability, overlap, hypothesis testing, Bayesian approach, hidden Markov models, Viterbi algorithm, simulations.

5.1 Introduction

An important class of wireless sensor networks (WSN) is the WSNs comprised of small and low-cost sensors with limited computational and communication power [1]. Sensors are deployed in a region, they wake up, organize themselves as a network,

¹University of Twente

²Delft University of Technology

³Aston University, Birmingham, UK

⁴Leiden University Medical Center

⁵Eindhoven University of Technology

⁶Utrecht University

*corresponding author, n.litvak@ewi.utwente.nl

†Thanks to other participants who helped during the week. We would like to thank Y. Boers, H. Driessen and P. Verveld from THALES Nederland B.V., Hengelo, for useful discussions during the week and for the help with preparation of the manuscript.

5 Increasing Detection Performance of Surveillance Sensor Networks

and start sensing the area. The objective of the sensors is sensing the environment and communicating the information to a data collection center. Many types of employment are envisaged for WSNs ranging from the monitoring of endangered animal populations to military surveillance or the surveillance of critical infrastructures [12], archaeological sites [2], perimeters, or country borders [10]. The tasks of a surveillance wireless sensor network (SWSN) is to detect objects crossing the field of interest. The sensors monitor the environment and send reports to a central control unit. The major requirement of a surveillance application is that the SWSN is to monitor the environment with a certain quality for a specific period of time. Important issues in designing an SWSN are the deployment decisions such as the sensing range of sensor nodes and density of the SWSN, and deployment strategy (random, regular, planned, et cetera.) to be applied [10].

Different types of sensors may have to be utilized in a WSN to address the problem at hand. For outdoor surveillance systems, radar, microwave, ultrasonic and/or infrared sensors are typical. To analyze the detection performance of the sensors or the surveillance systems, a common measure such as the single-sensor detection probability p may be utilized since it allows to abstract the different working principles of the sensors. The factors that affect p are the object-to-sensor distance, environmental characteristics, the size and the motion pattern of the object, et cetera. Moreover, He et al. [7, 8] showed that sensors produce a non-negligible amount of false alarms. The false alarms are defined as positive reports of a sensor when no object exists. Each sensor may produce a false alarm with a certain probability q . If data/decision fusion [5] is allowed, then the false alarm probability q negatively affects the detection performance of the network. The cost of false alarms varies depending on the application. For example, it is lower in a home surveillance system when compared to the cost of false alarms in a surveillance application of mission-critical infrastructure such as a nuclear reactor. Hence, the objective of an upstanding SWSN design is to maximize the detection probability of the system while minimizing or bounding the false alarm rate of the system. To this end, in the present paper, we study two problems concerning the design and performance of an SWSN: optimal sensor placement and algorithms for intruder detection in the presence of false alarms. Our main performance characteristics of the SWSN are the system's intruder detection probability and false alarm probability, for given input parameters p and q representing single-sensor probabilities. The problem of correctly communicating the reports of the sensors to the central control unit (with possibly additional failure probabilities) is beyond the scope of the present study. It has been studied elsewhere, among others in a previous study group Mathematics with Industry [9]. Therefore, we will assume perfect communication of the reports.

The sensor placement problem addressed in this work is formulated as follows: given a limited number of homogeneous sensors with an effective sensing range r and a field of interest modelled as a one- or two-dimensional area, determine the optimal location of the sensors that maximizes the detection performance of the SWSN. In Section 5.2.1, we study the trade-off involved in overlapping sensor

ranges. If the number of sensors is limited then, clearly, overlaps decrease the total sensing part of the area but increase the detection performance in the overlap of two or more sensor ranges. We give an explicit condition when overlap in sensor ranges leads to better detection performance of the system. Next, in Section 5.2.2 we propose an algorithm for efficient coverage of a 2-D area, based on a priori knowledge on the probability distribution of the intruder position. When the distribution of an object's location in the area is uniform, our algorithm performs closely to the optimal hexagonal placement.

Given a particular layout of the sensors, the probability of intruder detection and the false alarm probability of the network depend on the decision rule that prescribes in which situation an intrusion alarm has to be reported, based on observations from all deployed sensors. For instance, if we have two completely overlapping sensors and report an intrusion alarm only if both sensors signal an intruder, then the intruder detection probability of the SWSN is p^2 and the false alarm probability is q^2 . The problem is to determine a decision rule for reporting an intrusion alarm such that the detection performance of the network is maximized. In Section 5.3 we attempt to resolve this problem by statistical methods. Our main conclusion is that several observations of the same object are absolutely necessary to report an intrusion alarm with a reasonable confidence. However, multiple observations will result in a huge variety of observed patterns. Which patterns signal the intruder and which are caused by false alarms only? This question is tackled in Section 5.4 where we design a procedure for intruder detection, based on hidden Markov models and the Viterbi algorithm.

Finally, in Section 5.5 we present a simulation model that combines our placement and detection methods. Using this model, we characterize the detection performance in several configurations of a detection area.

Throughout the paper, we use the following notations:

- p – single-sensor detection probability, the probability that a sensor signals an object given that the object is present in the sensing range (assumed to be a circle, or sphere);
- q – the single-sensor false alarm probability, the probability that a sensor signals an intruder given that there is no intruder in the sensing range;
- r – sensing radius of a sensor;

Further, a random variable $X \in \{0, 1\}$ is an indicator of the event that an object is present in the sensing range of a sensor, and a random variable $Y \in \{0, 1\}$ is an indicator of the event that a sensor gives an alarm. We will also assume that the alarm events of individual sensors are mutually independent when conditioned on the absence or presence of the object.

5.2 Optimal coverage of the area

In this section we study the problem of optimal sensor placement, or sensor deployment, formulated as follows. Consider an area where a number of sensors are to be deployed, and assume that there is an object in the area. We define as $p_{\text{detection}}$ the probability that at least one sensor correctly detects the object. The goal is to find a sensor deployment maximizing $p_{\text{detection}}$. In order to compute $p_{\text{detection}}$, throughout the section we assume an a priori statistical knowledge on the object position.

One natural solution to this problem is to maximize the coverage of the observed area for a given number of sensors, or, equivalently, minimize the number of sensors employed while covering the complete area. If each sensor has a range with radius r , then we model the sensing area as a circle of radius r with a center at the sensor location. Thus, the question of minimizing the number of sensors while covering the complete area is equivalent to the so-called *covering problem* in two dimensions: cover a given area completely with the least amount of circles with a given fixed radius. This problem (and many others like the packing and kissing problems) is solved by using the hexagonal lattice, defined as the set of points $\lambda v + \mu w$, $\lambda, \mu \in \mathbb{Z}$, where $v = (1, 0)$ and $w = (1/2, \sqrt{3}/2)$ are the vectors spanning the lattice. To cover an area with circles of radius r , the vectors v, w must be scaled by a factor $r\sqrt{3}$. In the asymptotic limit, with a large area covered by sensors and with negligible boundary effects, the sum of the sensor ranges is 1.209 times the covered area, meaning that about 20.9% of the area is covered by two sensors and the remainder by one sensor. For further details, see [4]. An example of 7 sensors placed by using the hexagonal lattice and completely covering a rectangular area is given in Figure 5.1. An example of hexagonal placement of 105 sensors with non-covered gaps in between is given in Figure 5.3.

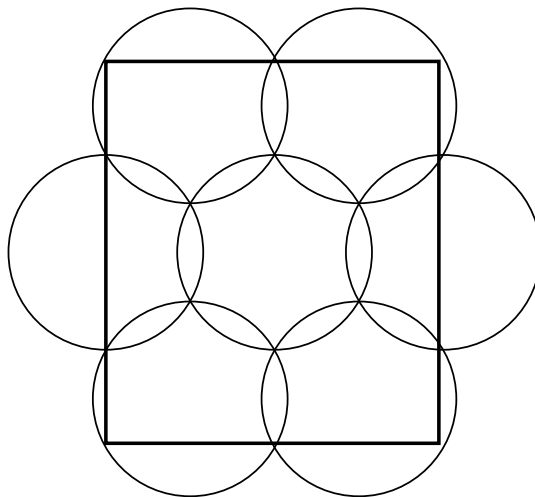


Figure 5.1: Rectangular area covered by seven sensors placed by using a hexagonal lattice.

Intuitively, a sensor placement with minimal overlapping or without overlapping must be optimal if the distribution of the object's position is uniform. Below in Section 5.2.1 we show that this is often the case also for non-uniform distributions, and in Section 5.2.2 we propose a procedure for close-to-optimal sensor placement.

5.2.1 Optimal allocation of two sensors

Does it make sense to let two sensors overlap? Having some overlap might be reasonable if we want a better detection in most vulnerable regions. However, if the number of sensors is limited then overlaps reduce the total coverage. In order to resolve this trade-off, we consider the following simple model. We restrict ourselves to a one-dimensional area, which constitutes an interval of length two, and a pair of sensors with $r = 1/2$. For each of the two sensors, the detection probability is p and the probability of a false alarm is q . The question is how to place these sensors so that the detection probability $p_{\text{detection}}$ is maximized.

Formally, let $S = [0, 2]$ be the area under surveillance. Denote by x_1 the leftmost point of the first sensor's coverage and by x_2 the leftmost point of the second sensor's coverage. Thus, the first sensor covers the segment $S_1 = [x_1, x_1 + 1]$ and the second one covers the segment $S_2 = [x_2, x_2 + 1]$, where $x_1 \in [0, 1]$ and $x_2 \in [x_1, 1]$, as shown in Figure 5.2.

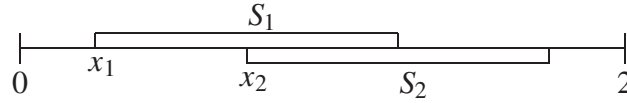


Figure 5.2: Partial overlapping of two sensors.

Now assume that the intruder location L has a distribution $P(L \leq x) = F(x)$, $x \in [0, 2]$. Then in the doubly covered segment $S_1 \cap S_2$ the detection probability by the two-sensor system is $p^2 + 2p(1 - p)$, and the object is in this segment with probability $F(x_1 + 1) - F(x_2)$. In the singly covered segment $(S_1 \cup S_2) \setminus (S_1 \cap S_2)$ detection probability is p , and the object is there with probability $F(x_2 + 1) - F(x_1 + 1) + F(x_2) - F(x_1)$. Finally, in the remaining uncovered part $S \setminus (S_1 \cup S_2)$ the detection probability is 0.

Rearranging the terms, we can formulate the problem of maximizing the detection probability $p_{\text{detection}}$ as follows:

$$\begin{aligned} \max_{x_1, x_2} \{p_{\text{detection}}(x_1, x_2)\} & \quad (5.1) \\ & = \max_{x_1, x_2} \{p(F(x_2 + 1) - F(x_1)) + p(1 - p)(F(x_1 + 1) - F(x_2))\}. \end{aligned}$$

In general, in order to find an optimal pair (x_1, x_2) we need exact knowledge of $F(x)$. However, as a direct consequence of (5.1), we can provide the following particular decision rule.

Lemma 5.2.1 (No-overlap principle). *It is optimal to allocate sensors without overlapping, if*

$$1 - p \leq \frac{f(x)}{f(x+1)} \leq \frac{1}{1 - p}$$

for every $x \in [0, 1]$, where $f(x) = \frac{dF(x)}{dx}$ is the probability density function of the object location.

Proof. By differentiating the expression to be maximized in (5.1), we show that it decreases in x_1 , if $f(x_1)/f(x_1 + 1) \geq 1 - p$, for every $x_1 \in [0, 1]$. In this case, 0 is the optimal value for x_1 . Similarly, this expression increases in x_2 if $f(x_2)/f(x_2 + 1) \leq 1/(1 - p)$ for $x_2 \in [0, 1]$, which sets 1 as the optimal value for x_2 . \square

The no-overlap principle indicates that it is optimal to maximize the coverage if the distribution of the intruder's position is sufficiently close to uniform. We illustrate the no-overlap principle by means of two examples, namely one example where the principle is applicable, and another where it is not.

Example 5.2.2. Assume that the intruder's entering position has uniform distribution, i.e., $f(x) = 1/2$, for every $x \in [0, 2]$. In this case our decision rule says that it is optimal to avoid any overlapping.

Example 5.2.3. Assume that the intruder's position has a linear density function, e.g., $f(x) = x/2$, for every $x \in [0, 2]$. The no-overlap rule cannot give us an unambiguous answer in this case. By solving (5.1), we obtain a more sophisticated joint sensor's allocation:

$$x_1 = \min \left\{ \frac{1 - p}{p}, 1 \right\} \text{ and } x_2 = 1.$$

5.2.2 Sensor deployment in a 2-D area

Let $N \in \mathbb{N}$, and let $\mathcal{X} \subseteq \{1, \dots, N\} \times \{1, \dots, N\}$ be a two-dimensional discrete grid. Further, for all $x \in \mathcal{X}$, let $f(x)$ be the probability that an object is at position x , provided that there is an object in the area. As before, r is an effective sensing range of a sensor, and p is the detection probability of one sensor. Our objective is to provide an algorithm which finds the 'optimal' deployment of sensors in \mathcal{X} , so that the probability to miss the object is decreased as much as possible. Note that the problem now is discretized by allowing only placements on some pre-specified points.

We say that a sensor is *deployed at position* $y \in \mathcal{X}$ if y is the center of the sensor's sensing range. Further, a tuple $\vec{y} = (y_1, \dots, y_n) \in \mathcal{X}^n$ ($n \in \mathbb{N} \cup \{0\}$) is called a *deployment of size* n , if n sensors are deployed at positions y_1, \dots, y_n . We use \emptyset for the empty deployment.

5.2 Optimal coverage of the area

Now, for $x \in \mathcal{X}$, $n \in \mathbb{N}$ and $\vec{y} = (y_1, \dots, y_n) \in \mathcal{X}^n$, define $g(\vec{y} \mid x)$ to be the probability that an intruder is *not* detected by any of the sensors deployed at positions y_1, \dots, y_n provided that the intruder's position is x . Further, denote by $p_{\text{missed}}(\vec{y})$ the probability that *none* of the sensors of the deployment \vec{y} detects the intruder. Then, given that there is an intruder in the area, we obtain:

$$p_{\text{missed}}(\vec{y}) = \sum_{x \in \mathcal{X}} f(x)g(\vec{y} \mid x), \quad \text{for } \vec{y} = (y_1, \dots, y_n) \in \mathcal{X}^n, \quad n \in \mathbb{N}. \quad (5.2)$$

One can compute $p_{\text{missed}}((y_1, \dots, y_m))$ for all $m \in \{1, \dots, n\}$ iteratively as follows. First, note that, naturally, $g(\emptyset \mid x) = 1$ for all $x \in \mathcal{X}$, and thus

$$p_{\text{missed}}(\emptyset) = \sum_{x \in \mathcal{X}} f(x)g(\emptyset \mid x) = \sum_{x \in \mathcal{X}} f(x) = 1.$$

Next, let $d : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be the Euclidean distance function. Take $m \in \{1, \dots, n\}$, $x \in \mathcal{X}$ and consider a deployment (y_1, \dots, y_m) of size m . Since the sensors are independent, we get

$$\begin{aligned} g((y_1, \dots, y_m) \mid x) &= g((y_m) \mid x)g((y_1, \dots, y_{m-1}) \mid x) \\ &= \begin{cases} g((y_1, \dots, y_{m-1}) \mid x) & \text{if } d(x, y_m) > r \\ (1 - p)g((y_1, \dots, y_{m-1}) \mid x) & \text{if } d(x, y_m) \leq r. \end{cases} \end{aligned} \quad (5.3)$$

Now, given the deployment (y_1, \dots, y_{m-1}) , the probability

$$p_{\text{missed}}((y_1, \dots, y_{m-1}, y_m))$$

can be computed using (5.2) and (5.3).

Using the described iterative approach, we can now address two (closely related) optimization problems: Minimum Size Deployment (MSD) and Minimum Probability Deployment (MPD).

- MSD: Given $\beta \in [0, 1]$, find a deployment \vec{y} of minimal size such that $p_{\text{missed}}(\vec{y}) \leq \beta$.
- MPD: Given $n \in \mathbb{N}$, find a deployment \vec{y} of size n such that $p_{\text{missed}}(\vec{y})$ is minimal.

We provide a heuristic algorithm described below, which can be used for both problems. The only difference is in the stopping criterion. In the main iterative step of the algorithm, a sensor is added to the deployment in such a way that the non-detection probability $p_{\text{missed}}(\cdot)$ is minimized (in case of a tie, the algorithm sticks to the candidate deployment that has been found first). This implies that the algorithm will find a ‘locally optimal’ solution, not necessarily the globally optimal one.

The heuristic algorithm

Input:

- MSD: $\beta \in [0, 1]$;
- MPD: $n \in \mathbb{N}$.

Initialization: $m := 0$.

Iterative Step:

$$\begin{aligned} y_{m+1} &:= \arg \min_{y \in \mathcal{X}} p_{\text{missed}}((y_1, \dots, y_m, y)) \\ &= \arg \min_{y \in \mathcal{X}} \sum_{x \in \mathcal{X}} f(x) g((y_1, \dots, y_m, y) | x), \end{aligned}$$

where $g((y_1, \dots, y_m, y) | x)$ is computed by (5.3) for all $x \in \mathcal{X}$;

$$m := m + 1.$$

Termination:

- MSD: $p_{\text{missed}}((y_1, \dots, y_m)) \leq \beta$, then STOP;
- MPD: $m = n$, then STOP.

Output: $\vec{y} := (y_1, \dots, y_m)$.

Note that there is a strong connection between the proposed algorithm and the no-overlap principle (see Lemma 5.2.1). Indeed, (3) says that the deployment of a new sensor at a position y reduces the non-detection probability $f(x)g(\vec{y}|x)$ by a factor $1 - p$ for all x such that $d(x, y) \leq r$. Since, ideally, we would like to reduce the highest values of $f(x)g(\vec{y}|x)$, the equivalent formulation of the iterative step is as follows:

$$y_{m+1} := \arg \max_{y \in \mathcal{X}} \sum_{x: d(x, y) \leq r} f(x) g((y_1, \dots, y_m) | x). \quad (5.4)$$

Now assume that we have deployed two sensors, and our algorithm allowed an overlap. Denote the sensing range of sensor $i = 1, 2$ by S_i . Then, since (5.4) holds for the deployment of sensor 2, it follows that

$$\sum_{x \in S_1 \cap S_2} (1 - p) f(x) + \sum_{x \in S_2 \setminus S_1} f(x) \geq \sum_{x \in S} f(x)$$

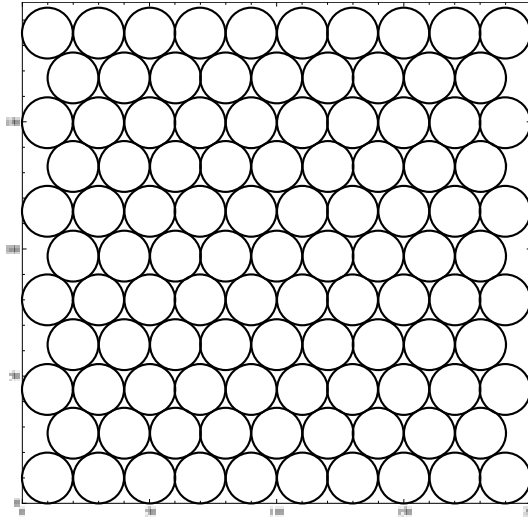


Figure 5.3: The hexagonal deployment of 105 sensors.

for any possible sensor range S in the area that does not overlap with S_1 . (Otherwise, S could have been chosen instead of S_2 .) Since the density in the first term of the left-hand side is taken with the factor $1 - p$ we see that for the inequality to hold, the values of $f(x)$ in $S_1 \cap S_2$ and/or in $S_2 \setminus S_1$ should be considerably larger than in their neighborhoods. This can be seen as the condition of the no-overlap principle, applied in two dimensions: overlap is possible only if there exist positions x such that the density $f(\cdot)$ varies considerably (by a factor of $1 - p$) within a sensor range of a sensor deployed in x .

In case two positions y would reduce the maximum non-detection probability by the same amount, we can break the tie arbitrarily, e.g. by using the first such position encountered, or by doing this randomly. The actual tie-breaking procedure does not matter too much on a global scale, because in the next iteration it is most likely that the other position will be chosen, except if the two positions are close (within a distance $2r$). Locally, there may occur significant effects of tie-breaking. We did not study this, but this topic warrants further investigation.

We have implemented the proposed algorithm in *Mathematica*. Below we present two examples of the deployment which is the output of our algorithm. Another example will be given in Section 5.5.

Example 5.2.4. Suppose $\mathcal{X} = \{1, \dots, 200\} \times \{1, \dots, 195\}$, $p = 0.9$ and $r = 10$. Moreover, suppose that f is the uniform distribution. We can construct a hexagonal deployment of 105 sensors in \mathcal{X} such that an intruder cannot be within the range of two different sensors (see Figure 5.3). It is easy to see that this deployment is optimal for the given number of sensors, and a simple calculation shows that the non-detection probability of this deployment is 0.255. Deploying the 105 sensors according to our algorithm leads to the deployment shown in Figure 5.4. The non-detection probability of this deployment is 0.267 which is close to the non-detection

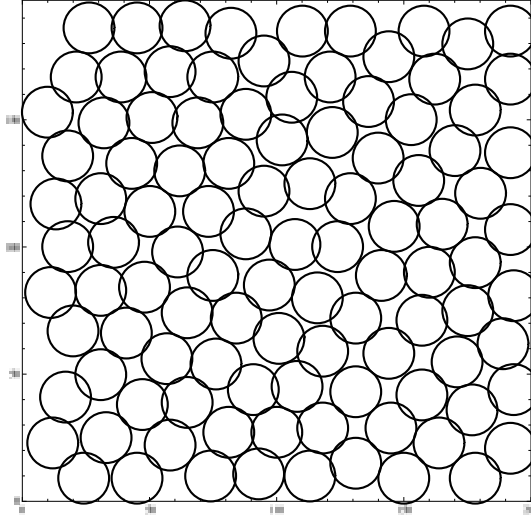


Figure 5.4: Deployment of 105 sensors according to the MPD algorithm: uniform distribution of the object location.

probability of the optimal hexagonal deployment.

Example 5.2.5. Suppose $\mathcal{X} = \{0, \dots, 100\} \times \{0, \dots, 100\}$, $p = 0.8$ and $r = 10$. Moreover, define $\sigma = 25$ and define r_x as the distance of x to the north-east axis (the line $y = x$) of \mathcal{X} for each $x \in \mathcal{X}$. Now suppose that $f(x) = ce^{-\frac{1}{2}(\frac{r_x}{\sigma})^2}$ for all $x \in \mathcal{X}$, where c is the normalization constant making f a probability distribution on \mathcal{X} . In other words, the signed distance between the intruder's position and the north-east axis of \mathcal{X} follows a discrete version of the normal distribution with mean 0 and standard deviation $\sigma = 25$. Here, the sign is positive for positions above the line, and negative for those below.

Having 200 sensors at our disposal, applying our algorithm leads to the deployment in Figure 5.5. As one would expect, the density of the sensor deployment increases when approaching the north-east axis. Moreover, a simple calculation shows that the non-detection probability of this deployment is 0.066.

We conclude that our heuristic algorithm can be used to find deployments which result in a good detection probability and are in line with the analytical results from Section 5.2.1. In particular, in the case of a uniform a priori probability distribution of the intruder position we found a nearly optimal solution.

5.3 Statistical methods for intruder detection

Optimal sensor deployment studied in the previous section is important for increasing the overall detection probability, that is, the number of true alarms produced by the system. However, since the false alarm probability q can be high in practice (e.g. q can be about 2%, which already has a considerable impact), sensor networks

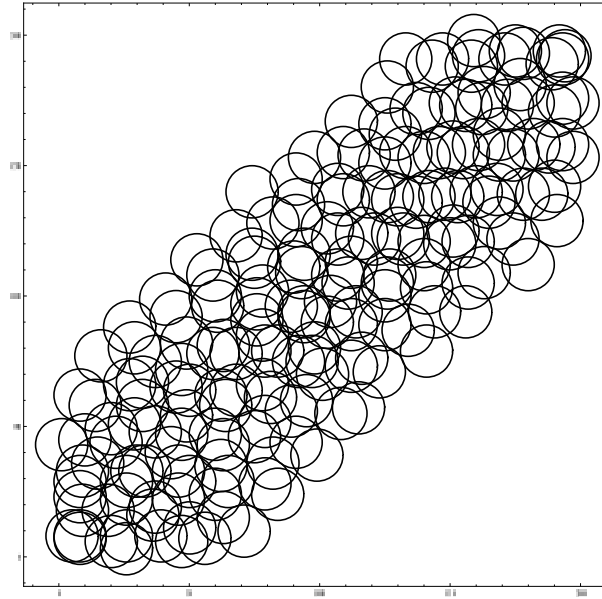


Figure 5.5: Deployment of 200 sensors according to the MPD algorithm: normal distribution of the signed distance between the object location and the north-east axis.

may even produce multiple false alarms at each moment in time. Still, the presence of an intruder increases the number of alarms, and after several observations one should be able to recognize an intrusion and report an alarm. To this end, we present in this section two statistical methods for intruder detection: one is based on classical hypothesis testing, and the other employs a Bayesian approach.

The hypothesis testing approach in Section 5.3.1 provides a decision making tool for reporting an intrusion alarm after a single observation of n identical sensors. In practice, false alarm reports are highly undesirable. Therefore, we bound the probability of a false report by choosing a high confidence level of the test. This sometimes leads to a poor performance of the test in a sense that with high probability, after one observation of n sensors, an object will stay undetected. In practice, however, this is not a big problem because there is usually enough time to produce several observations, not necessarily by the same sensor. Then the probability of the intruder's presence can be updated after each observation, for instance, using the Bayesian approach described in Section 5.3.2.

The Bayesian approach allows for great flexibility, because, along with the total number of alarms, it also takes into account the locations of the alarms. Therefore, in Section 5.3.2 we analyze a more general model than in Section 5.3.1. Specifically, we consider several non-overlapping parts of the coverage area, each deploying a number of completely overlapping sensors. Furthermore, we let the intrusion probabilities, as well as the detection and false alarm probabilities, depend on the sensor location. The motivation for this model is that although identical sensors will usually cover parts of the intrusion area with roughly equal sizes, the terrain

in which the sensors are placed may vary, e.g. in altitude, which can influence the local intrusion probabilities and the performance of the sensors.

5.3.1 Hypothesis testing for intruder detection

In the following, we consider the two extreme cases:

- Case I: n sensors, all at the same position; i.e. with identical sensing range;
- Case II: n non-overlapping sensors.

If there is an object in the area, then Case I is the case in which all sensors follow the same Bernoulli distribution with parameter p and Case II is the case that one of the sensors detects the object with probability p and each of the remaining sensors detect the object with probability q .

Assume that there can be at most one object in the area. Within the hypothesis testing formulation, we test the null-hypothesis that there is no intruder in the area against the alternative that the area is penetrated. If a critical number of alarms is observed then we reject the null-hypothesis and report an intrusion alarm. For $i = 1, \dots, n$ let $[Y_i = 1]$ be the event that sensor i detects an object and $[Y_i = 0]$ be the complementary event. Assuming that there is an intruder in the range of sensor i , we have $P(Y_i = 1) = p$.

Consider Case I: n sensors deployed at the same position with 100% overlap. Thus, our hypothesis testing formulation is as follows:

$$\text{Case I: } \begin{cases} H_0 : P(Y_i = 1) = q \text{ for all } i = 1, \dots, n, \\ H_1 : P(Y_i = 1) = p \text{ for all } i = 1, \dots, n. \end{cases}$$

In Case II, the sensors are not overlapping. Thus, the object can penetrate the range of at most one sensor. This leads to the following formalization:

$$\text{Case II: } \begin{cases} H_0 : P(Y_i = 1) = q \text{ for all } i = 1, \dots, n, \\ H_1 : P(Y_j = 1) = p \text{ for exactly one } j = 1, \dots, n; \\ \quad P(Y_i = 1) = q \text{ for } i = 1, \dots, n, i \neq j. \end{cases}$$

In both cases, as a statistic, we use the stochastic variable $T = Y_1 + \dots + Y_n$, the number of alarms produced by the system. We reject H_0 if and only if $T \geq c$, for some critical $c > 0$. Clearly, under H_0 , T has a Binomial(n, q) distribution. Denote the Binomial density function with parameters n and p at k by $B_{n,p}(k)$:

$$B_{n,p}(k) = \binom{n}{k} p^k (1-p)^{n-k}. \quad (5.5)$$

In our test, two types of errors can be made: false positives and false negatives (in statistical terms, type-one and type-two error, respectively). A *false positive* means

5.3 Statistical methods for intruder detection

| q | n | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|------|-----|---|---|---|---|---|---|---|----|
| 0.02 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0.04 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 |
| 0.06 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 |
| 0.08 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 0.10 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 |
| 0.12 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |
| 0.14 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |
| 0.16 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 4 |
| 0.18 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 4 | 4 |
| 0.20 | 2 | 2 | 3 | 3 | 3 | 4 | 4 | 4 | 4 |

Table 5.1: Critical number of alarms c for Cases I and II.

a false report, i.e., an intruder alarm is reported while there is no object in the area. In both Cases I and II, one has

$$p_{\text{false}} = P(\text{false positive}) = P_{H_0}(T \geq c) = \sum_{k=c}^n B_{n,q}(k).$$

We choose c in such a way that the above probability does not exceed an acceptable frequency of false alarm reports. A *false negative* means that an intruder is missed by the system, i.e., the intrusion alarm will not be reported while there was an object in the area. For Case I, we get

$$p_{\text{missed}}^I = P(\text{false negative}) = P_{H_1}(T < c) = \sum_{k=0}^{c-1} B_{n,p}(k),$$

and for Case II, we obtain

$$p_{\text{missed}}^{II} = P(\text{false negative}) = p \sum_{k=0}^{c-2} B_{n-1,q}(k) + (1-p) \sum_{k=0}^{c-1} B_{n-1,q}(k).$$

In this setting, the detection probability $p_{\text{detection}}$ of the system is equal to the power of the statistical test, i.e.,

$$p_{\text{detection}} = 1 - P(\text{false negative}).$$

We select some values for p and q and calculate corresponding values of c and $p_{\text{detection}}$ so that $p_{\text{false}} \leq 0.05$. In Tables 5.1 and 5.2 we present the values of c for Cases I and II. Table 5.3 gives the values of $p_{\text{detection}}$ for Case I, whereas Tables 5.4 and 5.5 give the values for Case II. In all the tables, the single-sensor detection probability is fixed at $p = 0.9$. The values of c used in Tables 5.3–5.5 are chosen according to the results of Tables 5.1 and 5.2.

As we see in Case I, $p_{\text{detection}}$ is very high. This is not surprising because in fact, in this case we have to distinguish between Binomial(n, p) and Binomial(n, q)

5 Increasing Detection Performance of Surveillance Sensor Networks

| q | n | 15 | 30 | 45 | 60 | 75 | 90 | 105 | 120 | 135 | 150 |
|------|-----|----|----|----|----|----|----|-----|-----|-----|-----|
| 0.02 | 1 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 6 | |
| 0.04 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 9 | 10 | |
| 0.06 | 3 | 4 | 6 | 7 | 8 | 9 | 11 | 12 | 13 | 14 | |
| 0.08 | 3 | 5 | 7 | 8 | 10 | 12 | 13 | 15 | 16 | 18 | |
| 0.10 | 4 | 6 | 8 | 10 | 12 | 14 | 16 | 18 | 19 | 21 | |
| 0.12 | 4 | 7 | 9 | 12 | 14 | 16 | 18 | 20 | 23 | 25 | |
| 0.14 | 4 | 7 | 10 | 13 | 16 | 18 | 21 | 23 | 26 | 28 | |
| 0.16 | 5 | 8 | 11 | 14 | 17 | 20 | 23 | 26 | 29 | 32 | |
| 0.18 | 5 | 9 | 12 | 16 | 19 | 22 | 26 | 29 | 32 | 35 | |
| 0.20 | 6 | 10 | 14 | 17 | 21 | 24 | 28 | 31 | 35 | 38 | |

Table 5.2: Critical number of alarms c for Cases I and II.

| q | n | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 0.02 | 0.9990 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.04 | 0.9990 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.06 | 0.9990 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.08 | 0.9990 | 0.9999 | 0.9995 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.10 | 0.9990 | 0.9963 | 0.9995 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.12 | 0.9990 | 0.9963 | 0.9995 | 0.9999 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.14 | 0.9720 | 0.9963 | 0.9995 | 0.9999 | 0.9998 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.16 | 0.9720 | 0.9963 | 0.9995 | 0.9987 | 0.9998 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| 0.18 | 0.9720 | 0.9963 | 0.9995 | 0.9987 | 0.9998 | 1.0000 | 0.9999 | 1.0000 | 1.0000 |
| 0.20 | 0.9720 | 0.9963 | 0.9914 | 0.9987 | 0.9998 | 0.9996 | 0.9999 | 1.0000 | 1.0000 |

Table 5.3: Values of $p_{\text{detection}}$ for Case I; $p = 0.9$.

| q | n | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|------|--------|--------|--------|--------|--------|--------|--------|--------|----|
| 0.02 | 0.9000 | 0.9001 | 0.9002 | 0.9004 | 0.9006 | 0.9008 | 0.9010 | 0.9013 | |
| 0.04 | 0.9002 | 0.9005 | 0.9009 | 0.9015 | 0.9022 | 0.9029 | 0.9038 | 0.2772 | |
| 0.06 | 0.9004 | 0.9010 | 0.9020 | 0.9032 | 0.2795 | 0.3170 | 0.3524 | 0.3857 | |
| 0.08 | 0.9006 | 0.9018 | 0.2554 | 0.3073 | 0.3551 | 0.3993 | 0.4402 | 0.4780 | |
| 0.10 | 0.9010 | 0.2440 | 0.3099 | 0.3694 | 0.4233 | 0.4721 | 0.1687 | 0.2035 | |
| 0.12 | 0.9014 | 0.2868 | 0.3609 | 0.4265 | 0.4846 | 0.1816 | 0.2242 | 0.2672 | |
| 0.14 | 0.2344 | 0.3278 | 0.4087 | 0.4788 | 0.1807 | 0.2310 | 0.2817 | 0.3318 | |
| 0.16 | 0.2650 | 0.3670 | 0.4534 | 0.1654 | 0.2232 | 0.2818 | 0.3396 | 0.1473 | |
| 0.18 | 0.2948 | 0.4044 | 0.4951 | 0.2006 | 0.2672 | 0.3332 | 0.1454 | 0.1907 | |
| 0.20 | 0.3240 | 0.4400 | 0.1629 | 0.2371 | 0.3119 | 0.1337 | 0.1838 | 0.2376 | |

Table 5.4: Values of $p_{\text{detection}}$ for Case II; $p = 0.9$.

| q | n | 15 | 30 | 45 | 60 | 75 | 90 | 105 | 120 | 135 | 150 |
|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|-----|
| 0.02 | 0.9031 | 0.4010 | 0.1989 | 0.3008 | 0.1680 | 0.2404 | 0.1421 | 0.1975 | 0.1208 | 0.1648 | |
| 0.04 | 0.3935 | 0.2944 | 0.2346 | 0.1922 | 0.1599 | 0.1344 | 0.1138 | 0.0968 | 0.1569 | 0.1337 | |
| 0.06 | 0.1841 | 0.2288 | 0.1114 | 0.1298 | 0.1429 | 0.1526 | 0.0876 | 0.0945 | 0.1004 | 0.1053 | |
| 0.08 | 0.2811 | 0.1813 | 0.1252 | 0.1759 | 0.1255 | 0.0909 | 0.1183 | 0.0875 | 0.1094 | 0.0822 | |
| 0.10 | 0.1434 | 0.1448 | 0.1339 | 0.1213 | 0.1091 | 0.0980 | 0.0879 | 0.0789 | 0.1135 | 0.1009 | |
| 0.12 | 0.2103 | 0.1157 | 0.1393 | 0.0836 | 0.0943 | 0.1020 | 0.1077 | 0.1117 | 0.0741 | 0.0772 | |
| 0.14 | 0.2836 | 0.1952 | 0.1424 | 0.1065 | 0.0810 | 0.1040 | 0.0798 | 0.0971 | 0.0754 | 0.0891 | |
| 0.16 | 0.1583 | 0.1567 | 0.1438 | 0.1297 | 0.1164 | 0.1044 | 0.0935 | 0.0839 | 0.0753 | 0.0677 | |
| 0.18 | 0.2143 | 0.1253 | 0.1440 | 0.0908 | 0.0987 | 0.1036 | 0.0691 | 0.0722 | 0.0743 | 0.0756 | |
| 0.20 | 0.1180 | 0.0995 | 0.0792 | 0.1084 | 0.0833 | 0.1020 | 0.0790 | 0.0925 | 0.0725 | 0.0828 | |

Table 5.5: Values of $p_{\text{detection}}$ for Case II; $p = 0.9$.

distributions. This can be done with a good precision because of a large difference between p and q . For instance, for 10 sensors, $p = 0.9$, $q = 0.02$, and $c = 5$, the probability $p_{\text{detection}}$ is 0.9999 while p_{false} is as small as 7.4×10^{-7} .

In Case II, $p_{\text{detection}}$ is low except for the cases when $c = 1$, that is, a detection signal of one sensor already triggers an intrusion alarm. The value $c > 1$ is obtained when the probability of just one alarm is reasonably high even if there is no intruder in the area. Effectively, $c > 1$ means that at least $c - 1$ false alarms are needed to detect the intruder. This is an undesirable result, which explains, in particular, the low power of the test. We conclude that in Case II one observation is simply not enough for efficient intruder detection, because in this case the observations with and without the intruder differ by at most one signal, which is difficult to reveal by classical hypothesis testing. One either has to make sensors overlap (as in Case I) or use several observations in a row. The latter can be done in several ways, for instance, one can use the Viterbi algorithm as in Section 5.4.

5.3.2 Bayesian approach for intruder detection

Consider Case II from the previous section, where $n \in \mathbb{N}$ different sensors are placed in such a way that the sensing ranges of different sensors do not overlap. Let $X \in \{0, 1\}$ denote the number of intruders present, with $P(X = 1) = \alpha$ an a priori probability of the intruder being present in the area. As before, let T be the stochastic variable denoting the total number of single-sensor alarms given at a particular time instant, so $T \in \{0, 1, \dots, n\}$. We have

$$\begin{aligned} P(X = 0 \mid T = k) \\ &= \frac{P(T = k \mid X = 0)P(X = 0)}{P(T = k \mid X = 0)P(X = 0) + P(T = k \mid X = 1)P(X = 1)}. \end{aligned}$$

Let F be the (unobservable) number of *false* alarms among the T . Then for all $k \geq 0$ we obtain

$$\begin{aligned} P(T = k \mid X = 1) &= P(T = k, F = k - 1 \mid X = 1) \\ &\quad + P(T = k, F = k \mid X = 1) \\ &= pB_{n-1,q}(k - 1) + (1 - p)B_{n-1,q}(k) \\ &= B_{n,q}(k) \left[\frac{kp}{nq} + \frac{(n-k)(1-p)}{n(1-q)} \right], \\ P(T = k \mid X = 0) &= B_{n,q}(k). \end{aligned}$$

Hence, the a posteriori probability of the presence of an object is

$$\begin{aligned}
 & P(X = 1 | T = k) \\
 &= 1 - \frac{P(T = k | X = 0)P(X = 0)}{P(T = k | X = 0)P(X = 0) + P(T = k | X = 1)P(X = 1)} \\
 &= 1 - \frac{1}{1 + \frac{P(T=k|X=1)P(X=1)}{P(T=k|X=0)P(X=0)}} \\
 &= 1 - \left(1 + \frac{\alpha}{1-\alpha} \left[\frac{kp}{nq} + \frac{(n-k)(1-p)}{n(1-q)} \right]\right)^{-1}. \tag{5.6}
 \end{aligned}$$

This formula can be generalized to the case combining Cases I and II from Section 5.3.1 as follows. Assume n non-overlapping ranges. Range $i = 1, \dots, n$ contains $m_i \in \mathbb{N}$ completely overlapping sensors. Let $T_i \in \{0, \dots, m_i\}$ be the number of alarms for range i and denote $\vec{T} = (T_1, \dots, T_n)$.

The stochastic variables $X_i \in \{0, 1\}$, $i = 1, \dots, n$, indicating the presence of an object in range i , have a priori probabilities $P(X_i = 1) = \alpha_i$ i.e., we allow certain parts of the area to have a higher a priori probability for intrusion than others. Also, we allow the detection and false alarm probabilities to depend on the sensor range; we use p_i and q_i to denote these respectively.

Since we assume that there can be at most one intruder at any given time instant, the vector $\vec{X} = (X_1, \dots, X_n)$ can attain values in the set $\{e_j : j = 0, \dots, n\}$ where e_j is the j th unit vector in \mathbb{R}^n and e_0 the zero vector in that space. We will use the notation $\mathcal{N} = \{0, 1, \dots, n\}$. We then calculate

$$\begin{aligned}
 & P(\vec{X} = e_j | \vec{T} = \vec{k}) \\
 &= \frac{P(\vec{T} = \vec{k} | \vec{X} = e_j)P(\vec{X} = e_j)}{P(\vec{T} = \vec{k} | \vec{X} = e_j)P(\vec{X} = e_j) + P(\vec{T} = \vec{k} | \vec{X} \neq e_j)P(\vec{X} \neq e_j)} \\
 &= \frac{P(\vec{T} = \vec{k} | \vec{X} = e_j)P(\vec{X} = e_j)}{P(\vec{T} = \vec{k} | \vec{X} = e_j)P(\vec{X} = e_j) + \sum_{s \in \mathcal{N} \setminus \{j\}} P(\vec{T} = \vec{k} | \vec{X} = e_s)P(\vec{X} = e_s)}.
 \end{aligned}$$

Further, we immediately have for $j > 0$ that

$$P(\vec{T} = \vec{k} | \vec{X} = e_j) = B_{m_j, p_j}(k_j) \prod_{i \in \mathcal{N} \setminus \{j\}} B_{m_i, q_i}(k_i). \tag{5.7}$$

If we define $m_0 = k_0 = 0$, this formula also holds for $j = 0$. Furthermore, if we

define $\alpha_0 = 1 - \sum_{i \in \mathcal{N} \setminus \{0\}} \alpha_i$, we can state

$$\begin{aligned}
 P(\vec{X} = e_j \mid \vec{T} = \vec{k}) &= \frac{P(\vec{T} = \vec{k} \mid \vec{X} = e_j) P(\vec{X} = e_j)}{P(\vec{T} = \vec{k} \mid \vec{X} = e_j) P(\vec{X} = e_j) + \sum_{s \in \mathcal{N} \setminus \{j\}} P(\vec{T} = \vec{k} \mid \vec{X} = e_s) P(\vec{X} = e_s)} \\
 &= \left(1 + \frac{\sum_{s \in \mathcal{N} \setminus \{j\}} P(\vec{T} = \vec{k} \mid \vec{X} = e_s) P(\vec{X} = e_s)}{P(\vec{T} = \vec{k} \mid \vec{X} = e_j) P(\vec{X} = e_j)} \right)^{-1} \\
 &= \left(1 + \frac{\sum_{s \in \mathcal{N} \setminus \{j\}} \alpha_s B_{m_s, p_s}(k_s) \prod_{i \in \mathcal{N} \setminus \{s\}} B_{m_i, q_i}(k_i)}{\alpha_j B_{m_j, p_j}(k_j) \prod_{v \in \mathcal{N} \setminus \{j\}} B_{m_v, q_v}(k_v)} \right)^{-1} \\
 &= \left(1 + \sum_{s \in \mathcal{N} \setminus \{j\}} \frac{\alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s} \left(\frac{p_j}{q_j}\right)^{-k_j} \left(\frac{1-p_j}{1-q_j}\right)^{-(m_j-k_j)}}{\alpha_j B_{m_j, p_j}(k_j) \prod_{v \in \mathcal{N} \setminus \{j\}} B_{m_v, q_v}(k_v)} \right)^{-1} \\
 &= \left(\sum_{s \in \mathcal{N}} \frac{\alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s} \left(\frac{p_j}{q_j}\right)^{-k_j} \left(\frac{1-p_j}{1-q_j}\right)^{-(m_j-k_j)}}{\alpha_j \left(\frac{p_j}{q_j}\right)^{k_j} \left(\frac{1-p_j}{1-q_j}\right)^{m_j-k_j}} \right)^{-1} \\
 &= \frac{\alpha_j \left(\frac{p_j}{q_j}\right)^{k_j} \left(\frac{1-p_j}{1-q_j}\right)^{m_j-k_j}}{\sum_{s \in \mathcal{N}} \alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s}}. \tag{5.8}
 \end{aligned}$$

For $j = 0$, we thus find

$$P(\vec{X} = e_0 \mid \vec{T} = \vec{k}) = \frac{1 - \sum_{i \in \mathcal{N} \setminus \{0\}} \alpha_i}{1 - \sum_{i \in \mathcal{N} \setminus \{0\}} \alpha_i + \sum_{s \in \mathcal{N} \setminus \{0\}} \alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s}},$$

so the conditional probability of an intruder given the observed area alarms vector \vec{T} equals

$$\begin{aligned}
 P(\vec{X} \neq e_0 \mid \vec{T} = \vec{k}) &= 1 - P(\vec{X} = e_0 \mid \vec{T} = \vec{k}) \\
 &= 1 - \left(1 + \frac{\sum_{s \in \mathcal{N} \setminus \{0\}} \alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s}}{1 - \sum_{i \in \mathcal{N} \setminus \{0\}} \alpha_i} \right)^{-1}.
 \end{aligned}$$

Notice that the Case II treated in Section 5.3.1 corresponds to

$$p_i = p, \quad q_i = q, \quad m_i = 1, \quad \alpha_i = \alpha/n,$$

for all $i \in \mathcal{N}$ and we then find back our earlier formula (5.6) for the conditional probability of an intrusion.

In the case where we use only one time instant to observe the alarms, it seems natural to conclude that an intruder is present whenever

$$A(\vec{k}) = P(\vec{X} \neq e_0 \mid \vec{T} = \vec{k}) = 1 - \left(1 + \frac{\sum_{s \in \mathcal{N} \setminus \{0\}} \alpha_s \left(\frac{p_s}{q_s}\right)^{k_s} \left(\frac{1-p_s}{1-q_s}\right)^{m_s-k_s}}{1 - \sum_{i \in \mathcal{N} \setminus \{0\}} \alpha_i} \right)^{-1} \tag{5.9}$$

5 Increasing Detection Performance of Surveillance Sensor Networks

satisfies $A(\vec{k}) \geq \gamma$ for some critical threshold γ , where e.g. we can choose $\gamma \in (0.5, 1)$, which means that we raise an alarm whenever the conditional probability of an intruder is sufficiently larger than the conditional probability that there is no intruder. The probability of a false intrusion alarm then becomes

$$\begin{aligned} p_{\text{false}} &= P((A(\vec{T}) \geq \gamma) \wedge (\vec{X} = e_0)) \\ &= \sum_{\vec{0} \leq \vec{k} \leq \vec{m}} \mathbb{1}_{\{A(\vec{k}) \geq \gamma\}} P(\vec{T} = \vec{k} \mid \vec{X} = e_0) P(\vec{X} = e_0), \end{aligned}$$

where we use the shorthand notation $\vec{0} \leq \vec{k} \leq \vec{m}$ for $\{\vec{k} : 0 \leq k_i \leq m_i, i \in \mathcal{N}\}$. The probability of a missed intrusion is

$$\begin{aligned} p_{\text{missed}} &= P((A(\vec{T}) < \gamma) \wedge (\vec{X} \neq e_0)) \\ &= \sum_{\vec{0} \leq \vec{k} \leq \vec{m}} \mathbb{1}_{\{A(\vec{k}) < \gamma\}} \sum_{j \in \mathcal{N} \setminus \{0\}} P(\vec{T} = \vec{k} \mid \vec{X} = e_j) P(\vec{X} = e_j). \end{aligned}$$

By substituting (5.7) and (5.9) in these expressions, we can now calculate explicitly what the probabilities of a false intrusion alarm or missed intrusion are (based on a single observation in time) for the given a priori probabilities in \vec{p} and \vec{q} and a given sensor configuration vector \vec{m} .

We note that the Bayesian approach can be also extended to a sequence of observations. For instance, the a posteriori probabilities obtained by using (5.8) after the first observation, can be substituted back into (5.8) instead of α_j 's to recompute the probabilities of the intruder's presence after the second observation, and so on.

5.4 Viterbi algorithm for intruder detection

In this section, we present a novel method of using sequential observations for intruder detection. We model the signals from the sensors as a so-called hidden Markov model. This is a stochastic process, based on a Markov chain to which noise is added. Using this representation we can distinguish between the signals that should have been given off by the sensors, i.e. the 'true' state of the system, and the signals that are actually given off, including the false alarms and missed detections.

Given a sequence of signals we determine the most likely sequence of true states, using the so-called Viterbi algorithm. In this way, we decide whether the signals indicate indeed an intruder, or are only false alarms. From this we derive a decision rule for when to report an intrusion alarm, thus reducing the number of false reports. All calculations needed to obtain this rule can be pre-computed.

We outline the proposed method for the case of one sensor. In particular, we explain the hidden Markov model, and illustrate how, based on a few signals from the sensor, we decide if an intrusion alarm should be given. We indicate how the method can be extended to networks of sensors. As the state space, and so the

number of calculations, increases exponentially with the number of sensors, we show how to truncate it in a clever way.

5.4.1 A one-sensor model

Consider the case of one sensor, where an object possibly passes by. Assuming a low speed of the object, the object is in the range of the sensor for multiple time steps. Let the stochastic process $\{X_t\}_{t \in \mathbb{N}}$ denote if an object is in the range of the sensor, where

$$X_t = \begin{cases} 1 & \text{if an object is in the range of the sensor at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

So X_t gives the ‘true’ state of the system at time t .

We assume that the process $\{X_t\}$ is a Markov chain, so the probability law for X_{t+1} only depends on X_t . Denote $p_{ij} = P(X_{t+1} = j \mid X_t = i)$. The speed of the object and its path through the range of the sensor are modelled in the transition probabilities. The number of consecutive ones in a Markov chain follows a geometric distribution, with $E(\# \text{ of steps in sensor range}) = 1/p_{10}$. We want the stationary distribution of $\{X_t\}$, say X_∞ , to be such that $P(X_\infty = 1) = 1 - P(X_\infty = 0) = \alpha$, the a priori probability that there is an object in the system. This gives the following transition probability matrix A :

$$A = \begin{pmatrix} 1 - \frac{\alpha}{1-\alpha}p_{10} & \frac{\alpha}{1-\alpha}p_{10} \\ p_{10} & 1 - p_{10} \end{pmatrix}.$$

We take the initial distribution for X_0 to be equal to the stationary distribution.

To the process $\{X_t\}$ we add noise, which consists of false alarms and missed detections. This gives the process of signals given off by the sensor, say $\{Y_t\}_{t \in \mathbb{N}}$. Let

$$Y_t = \begin{cases} 1 & \text{if the sensor gives an alarm at time } t, \\ 0 & \text{otherwise.} \end{cases}$$

So Y_t is the observed state at time t . The noise is such that Y_t only depends on X_t , in an independent and identically distributed (i.i.d.) way. A false alarm occurs when $[Y_t = 1]$ given $[X_t = 0]$, and this happens with probability q . A missed detection occurs if $[Y_t = 0]$ given $[X_t = 1]$, and this happens with probability $1 - p$.

We now have that the process $\{Y_t\}$ is a *hidden Markov model* [11]. We can interpret $\{Y_t\}$ as observing $\{X_t\}$ via a noisy channel. Only the process $\{Y_t\}$ is observed, while the states of the process $\{X_t\}$ are not known, i.e. hidden, which explains the name of this model. The process $\{X_t\}$ is often referred to as the underlying or hidden process. Whereas for a Markov chain it holds that the next state of the process depends only on the previous state, or a fixed number of previous states, for a hidden Markov model the transition probabilities depend on the entire history of the process.

5.4.2 The Viterbi algorithm for the one-sensor model

Given a sequence of observed states, say $O = \{O_1, O_2, \dots, O_L\}$, the question now rises, what is the most likely sequence of underlying ('true') states, $Q = \{Q_1, Q_2, \dots, Q_L\}$. There is an efficient algorithm for solving this problem, called the *Viterbi algorithm* [6]. This algorithm, based on dynamic programming, calculates

$$\max_Q P(Q | O).$$

Applying this algorithm we are able to correct false alarms and missed detections for a given sequence of observations. For example, a single one in between many zeros is likely to be a false alarm, while a zero in between many ones is probably a missed detection. If we, for instance, observe the sequence 000111011000 then it is not surprising that the most likely underlying state sequence is 000111111000, i.e., a missed detection is corrected. More important are the corrections of false alarms. The observed sequence 0001000 will most likely have an underlying sequence of all zeros, so a false alarm is corrected. In this way, we prevent reporting a false intrusion alarm. While for these two examples the most likely underlying states are straightforward to see, the algorithm also helps with cases like 00010100. Here, it is not immediately clear whether the ones are two false alarms, or the zero in between represents a missed detection.

Based on the results of this algorithm, we give a decision rule whether or not to report an intrusion alarm for a given sequence of observations. We illustrate this for two and for three consecutive observed states, but it can be done for every desired number of observations. We give an intrusion alarm if the most likely underlying state sequence contains at least one 1 in it, signifying that in the most likely scenario, an intrusion took place in at least one moment in time. We also calculate the probability that the underlying state sequence consists of only zeros, given the observation. One minus this quantity equals the probability that there was an intruder. The latter is equal to the probability p_{missed} that the intruder will pass undetected in case the sequence of all zeros happens to be most likely. All calculations can be done off-line, resulting in a list of observed states for which an intrusion alarm should be given.

For the values $p = 0.9$, $q = 0.02$, $\alpha = 0.01$ and $E(\# \text{ of steps in sensor range}) = 10$, the probabilities for all possible combinations of states are given in Table 5.6 for two and three consecutive observations. For two observations, we only give an intrusion alarm in case both observations are a 1. With probability 0.9441 this is indeed the underlying sequence, and the probability that there was no intruder is about 0.05. Giving no intrusion alarm when the observed sequence contained two or one zeros turns out to be correct with probabilities 0.9997 and 0.92, respectively. For three observations, there are four cases for which we give an intrusion alarm. To improve the probability of correct decisions further, one could make use of more consecutive observations.

| O | Q | alarm? | $P(Q O)$ | $P(Q = \vec{0} O)$ |
|-------|-------|--------|------------|----------------------|
| 0 0 | 0 0 | No | 0.9997 | 0.9997 |
| 0 1 | 0 0 | No | 0.9196 | 0.9196 |
| 1 0 | 0 0 | No | 0.9196 | 0.9196 |
| 1 1 | 1 1 | Yes | 0.9441 | 0.0512 |
| 0 0 0 | 0 0 0 | No | 0.9998 | 0.9998 |
| 0 0 1 | 0 0 0 | No | 0.9491 | 0.9491 |
| 0 1 0 | 0 0 0 | No | 0.9833 | 0.9833 |
| 0 1 1 | 0 1 1 | Yes | 0.4016 | 0.2177 |
| 1 0 0 | 0 0 0 | No | 0.9491 | 0.9491 |
| 1 0 1 | 1 1 1 | Yes | 0.6060 | 0.3577 |
| 1 1 0 | 1 1 0 | Yes | 0.4016 | 0.2177 |
| 1 1 1 | 1 1 1 | Yes | 0.9936 | 0.0013 |

Table 5.6: Hidden Markov Model for the case of one sensor. For each observed state O the most likely underlying state Q is given.

For this model we have assumed that $\{X_t\}$ is a Markov chain. The number of steps in the range of the sensor is geometrically distributed, which models a variable speed and direction of the object. We can improve this by letting $\{X_t\}$ be a Markov chain of order k , where the probability law of X_{t+1} depends on the last k states: X_{t-k+1}, \dots, X_t . This allows us to vary the distribution of the number of steps in the sensor range. For instance, in this way one can model a deterministic number of steps. The state space then increases to 2^k states, but the problem remains numerically tractable since the calculations for the decision rule need to be done only once.

5.4.3 A sensor-network model

We can extend this method to networks of several sensors. Consider for instance the following example with $n = 4$ non-overlapping sensors as given in Figure 5.6. Let $\vec{X}_t^T = (X_{1,t}, X_{2,t}, X_{3,t}, X_{4,t})$, where $X_{i,t} = 1$ if there is an intrusion in the range of sensor i at time t , and $X_{i,t} = 0$ otherwise, $i = 1, 2, 3, 4$; $t \geq 1$. Assume that there is at most one object in the area at any moment in time, so that the state space of $\{\vec{X}_t\}$ consists of $n + 1 = 5$ states: the all-zero state and the states where the object is in the range of one of the n sensors. We assume the process $\{\vec{X}_t\}$ again to be Markov. The path and the speed of the object are modelled in the transition probabilities. This can be based on historical data, or on other knowledge about the system. If the object can remain in the range of one sensor for several time steps, p_{ii} is positive. Here, we assume that the object always enters via sensor 1, and then continues its path through sensor 2, 3 or 4, or outside the range of any of these

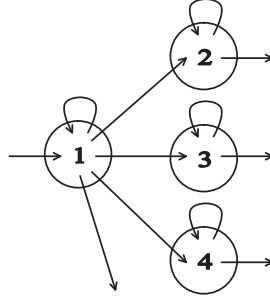


Figure 5.6: A network with four sensors. Indicated are possible transitions.

sensors. The transition probabilities and the corresponding states are given by

$$A = \begin{pmatrix} 1 - \alpha & \alpha & 0 & 0 & 0 \\ 1 - p_{1\bullet} & p_{11} & p_{12} & p_{13} & p_{14} \\ 1 - p_{22} & 0 & p_{22} & 0 & 0 \\ 1 - p_{33} & 0 & 0 & p_{33} & 0 \\ 1 - p_{44} & 0 & 0 & 0 & p_{44} \end{pmatrix}, \begin{matrix} (0, 0, 0, 0) \\ (1, 0, 0, 0) \\ (0, 1, 0, 0) \\ (0, 0, 1, 0) \\ (0, 0, 0, 1) \end{matrix}$$

with the necessary conditions on the p_{ij} imposed to let A be a stochastic matrix. Here, $p_{1\bullet} = \sum_{j=1}^4 p_{1j}$.

The probability law of observing \vec{Y}_t given \vec{X}_t follows a multinomial distribution. As before, there are four possibilities for the pair $(X_{i,t}, Y_{i,t})$, specifically, $P(Y_{i,t} = 1 | X_{i,t} = 0) = q$ and $P(Y_{i,t} = 0 | X_{i,t} = 1) = 1 - p$.

The state space of $\{\vec{Y}_t\}$ now consists of 2^n states: each sensor can give an alarm or not. As the size of the state space grows exponentially with n , already for a moderately large number of sensors n the problem becomes huge. Because of this, but moreover because many of these states are very unlikely to occur, we truncate the state space of $\{\vec{Y}_t\}$. For this, we calculate the number of false alarms, say c , that has a probability of occurring less than say 0.001:

$$P(\# \text{ false alarms} > c) < 0.001.$$

Now we allow only the vectors \vec{Y}_t in the state space of $\{\vec{Y}_t\}$ that are at Hamming distance $\leq c$ away from any of the states of $\{\vec{X}_t\}$, where the *Hamming distance* between two zero-one vectors is the number of indices in which they are different. In this way, we drastically reduce the state space of $\{\vec{Y}_t\}$, making the calculations more tractable.

We now again have a hidden Markov model, for which we can derive a decision rule when to give an intrusion alarm in the same way as for the case of one sensor. We can list all possible sequences of a number of observations of the process $\{\vec{Y}_t\}$. By the Viterbi algorithm, we calculate the most likely underlying state sequences of the process $\{\vec{X}_t\}$. If it contains at least one 1, for such a sequence an intrusion alarm should be given. By calculating the probability that the underlying states are only zeros, the probability of making an error is found.

The hidden Markov model method of the present section can be used in combination with the heuristic algorithm for placement of sensors presented in Section 2. One way of doing this is to use the Viterbi method to combine the results of multiple single-sensor readings into one result, giving improved values for p and q that can be used in the placement algorithm. This is done in some of the numerical experiments of the next section.

5.5 Numerical results

We verify and combine the proposed methods for sensor deployment and intruder detection using a simulation model of a network consisting of a number of individual sensors, which perform under uncertainty. The performance of each individual sensor is characterized by the probability of true detection p and the probability of false alarm q . As before, we use similar performance measures for characterizing the performance of the sensor network. Thus, our performance measures are the probability of true detection of the network $p_{\text{detection}}$ and the probability of a false intrusion alarm p_{false} .

The objective of a surveillance wireless sensor network (SWSN) design is to get a value $p_{\text{detection}}$ that is as high as possible and a value of p_{false} that is as small as possible. In this study, we explore numerically the possibility of affecting the values $p_{\text{detection}}$ and p_{false} of the sensors by arranging their locations as well as by exploiting multiple readings. In the numerical experiments, we estimate $p_{\text{detection}}$ and p_{false} for an SWSN. Numerically, these measures are defined as follows:

$$p_{\text{detection}} = \frac{N_{\text{detection}}}{N}, \quad (5.10)$$

$$p_{\text{false}} = \frac{N_{\text{false}}}{N}, \quad (5.11)$$

where $N_{\text{detection}}$ and N_{false} are the number of true and false detections respectively, while N is the total number of experiments, with or without the object in the area, respectively.

The experimental setup is as follows. The presence of an object in the SWSN is simulated N times, and the intrusion alarm is reported based on the readings of n individual sensors, according to the criteria of detection, e.g. as in Sections 5.3 and 5.4. Then $p_{\text{detection}}$ is computed by formula (5.10). In this study, N is set to 1000. To account for the variability of the simulation results, we have repeated all experiments 100 times. The estimate of $p_{\text{detection}}$ is represented by the average of the results as well as by the standard deviation. The results are also presented as a histogram, where the x -axis gives the values of the estimates obtained and the y -axis represents the relative frequency of occurrence of the estimates. The same experimental setup is used for computing the p_{false} of the SWSN by setting the object to reside outside of the SWSN coverage area for N consecutive times and

5 Increasing Detection Performance of Surveillance Sensor Networks

then using (5.11). In all the experiments presented here, the individual sensors are identical with $p = 0.9$ and $q = 0.02$. The other parameters are varied in the different examples to obtain the most demonstrative results.

To verify that our simulation gives correct estimates of $p_{\text{detection}}$ and p_{false} , we first perform an experiment using a simple sensor network of one sensor but with two consecutive readings. In this case, as suggested by the Viterbi algorithm from Section 5.4, the criterion of an intrusion alarm is that the sensor raises an alarm in two consecutive readings. Since the two readings are independent of each other, we have $p_{\text{detection}} = p^2 = 0.81$ and $p_{\text{false}} = q^2 = 0.0004$. The numerical results shown in Figure 5.7 demonstrate that the numerical method gives accurate estimates.

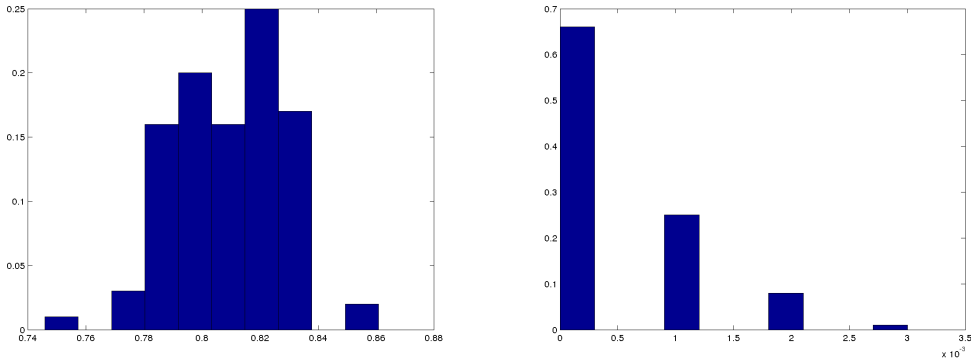


Figure 5.7: (Left) Estimate of $p_{\text{detection}}$ for one sensor with two consecutive readings. The mean is 0.8093, the standard deviation is 0.0181. (Right) Estimate of p_{false} : the mean of the estimate is 4.4×10^{-4} and the standard deviation is 6.9×10^{-4} .

In the example above, we have verified that our simulation program gives correct estimates of $p_{\text{detection}}$ and p_{false} . As a next step, in our simulation model we will combine the results on sensor deployment and intruder detection from the previous sections to detect a moving target. The area of interest is assumed to be the unit square, defined by $x \in [0, 1]$ and $y \in [0, 1]$, where (x, y) represents the location of a point. We describe the motion of an object using the white noise acceleration model described e.g. in [3, p. 263]:

$$x_o(t_{k+1}) = x_o(t_k) + v_x dt + \sqrt{dt} a_x \eta_x(t_k), \quad (5.12)$$

$$y_o(t_{k+1}) = y_o(t_k) + v_y dt + \sqrt{dt} a_y \eta_y(t_k), \quad (5.13)$$

where $(x_o(t_k), y_o(t_k))$ represents the object coordinate at time t_k , dt the time step, v_x and v_y the velocity in the x and y direction, respectively, a_x and a_y the acceleration terms, and η_x and η_y the noise terms, which are independent standard-normally distributed at each time step. The values of v_x , v_y , a_x and a_y are all set to 0.01 and dt is equal to 0.1. For illustration, we presented two realizations of the object's motion in Figure 5.8.

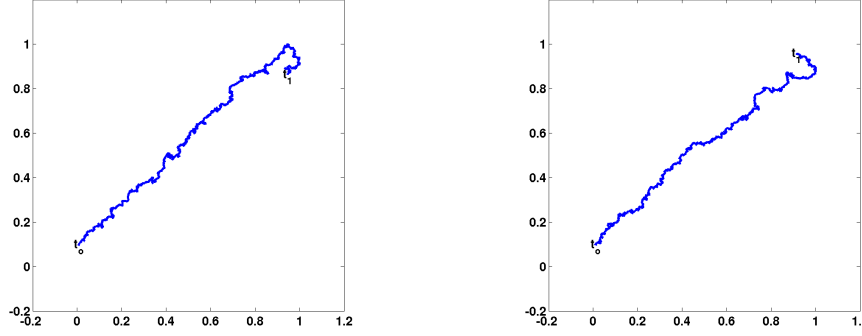


Figure 5.8: Some realizations of the object path used in the experiments; t_0 is the starting point and t_1 is the end point.

Now we would like to investigate the impact of sensor deployment. To this end, consider an SWSN consisting of eight individual sensors. Three different sensor arrangements are studied. In arrangements A and B, the locations of the sensors are determined randomly. In arrangement C, the sensors are located according to the MPD deployment algorithm from Section 5.2.2. Therefore, the sensors in arrangement C are located along a diagonal of the area of interest since these are the most likely locations of the object. The SWSN arrangements are depicted in Figure 5.9. The position of the object is depicted by an asterisk and the sensor that gives an intrusion alarm by a highlighted circle.

In this study, we have computed the $p_{\text{detection}}$ and p_{false} of the three sensor networks by exploiting the multiple readings by each sensor. Since the sensing ranges practically do not overlap, we are in the situation of Case II of Section 5.3.1. However, since each sensor raises an alarm based on the results of k readings according to the decision rule from Table 5.6, we have to adjust the probabilities p and q to the detection probability $p(k)$ and the false alarm probability $q(k)$ for $k = 1, 2, 3$. Simple calculations give:

$$\begin{aligned} p(1) &= p, & q(1) &= q; \\ p(2) &= p^2, & q(2) &= q^2; \\ p(3) &= p^3 + 3p^2(1 - p), & q(3) &= q^3 + 3q^2(1 - q). \end{aligned}$$

According to Table 5.1, the critical value for $q = 0.02$ is 1, that is the SWSN should give an intrusion alarm if the alarm is coming from at least one of the sensors. Since $q(2)$ and $q(3)$ are smaller than $q = 0.02$ the critical value remains the same if we use multiple readings from each sensor. Thus, if k readings of each sensor are used at each time point, for our three SWSN arrangements we have

$$p_{\text{detection}} = p_{\text{coverage}} \cdot p(k), \quad (5.14)$$

$$p_{\text{false}} = 1 - (1 - q(k))^8, \quad (5.15)$$

5 Increasing Detection Performance of Surveillance Sensor Networks

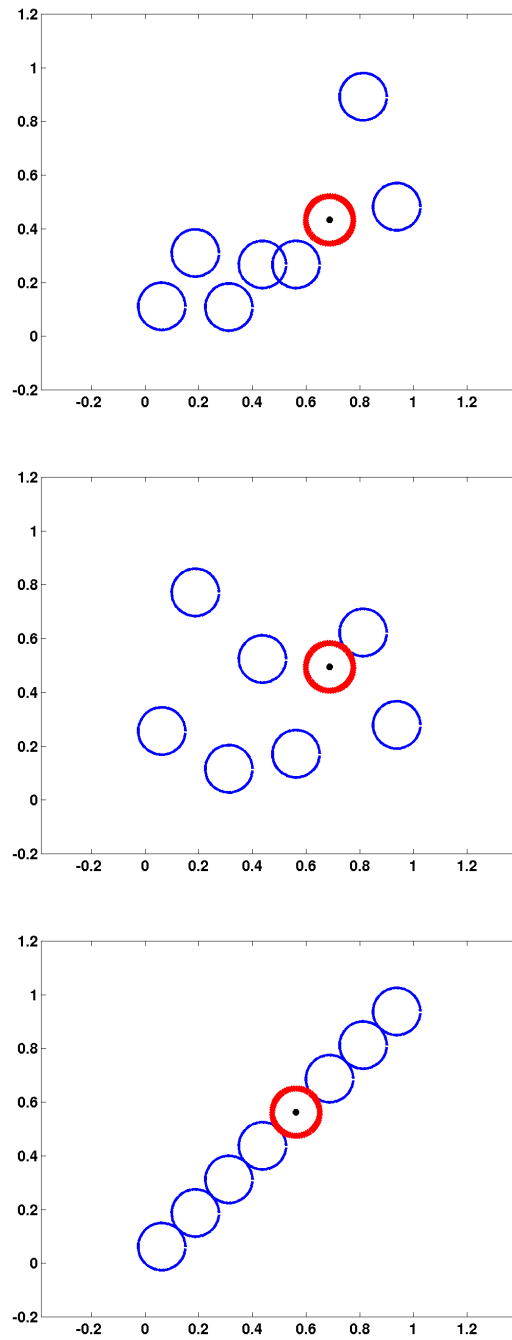


Figure 5.9: Example of sensor networks: (top) network A; (middle) network B; (bottom) network C.

where p_{coverage} is the probability that the object is within the network coverage, i.e., within one of the sensor ranges. Clearly, $p(k)$ in (5.14) and p_{false} in (5.15) are identical for the layouts A, B and C if the same number of readings is used. In this case, the performance of the SWSN is determined by how likely the object will pass through the network coverage, allowing the network to detect the existence of the object. This relative frequency is the estimate of the probability p_{coverage} that affects $p_{\text{detection}}$ in (5.14).

To estimate p_{coverage} , we simulate the object's motion into the area for each sensor network and compute the relative frequency of the object passing through the sensor network coverage. As in the previous experiments, the object is allowed to move inside the area of interest for 1000 time steps. Moreover, the experiments are repeated 100 times to account for the variability in the estimates. The results are presented in Figure 5.10. The estimates of p_{coverage} are 0.2884, 0.1420, and 0.6367 for sensor network A, B, and C, respectively. The conclusion is that the SWSN C is more likely to detect the object than the others.

Now, consider an SWSN of 50 sensors deployed by means of the MPD algorithm from Section 5.2.2 (see Figure 5.11). As before, the advancing of the object in the area is described by (5.12) and (5.13), where we choose $v_x = 0.02$, $v_y = 0.02$, $a_x = 0.001$, $a_y = 0.01$. Again, we report an intrusion alarm if a sensor signals an intruder in two consecutive readings, as suggested in Table 5.6 in case of two observations. In Figure 5.11, we show one time instant of a simulation run. An asterisk denotes the object position. The two overlapping highlighted circles depict the two sensors that give a correct intrusion alarm. The highlighted circle that does not contain the object, gives a false alarm.

For this network, the rate of false intrusion alarms is 0.0004. Furthermore, since the SWSN consists of an ample amount of sensors, our deployment strategy ensures that p_{coverage} (almost) equals one. The histogram for the detection probability at each time point is given in Figure 5.12. The high values of $p_{\text{detection}}$ are due to a considerable overlap of sensor ranges for the most likely positions of the object.

5.6 Conclusions

In this paper, we addressed two problems concerning design and performance of an SWSN: sensor placement and object detection. For the first problem, we suggest to use a hexagonal placement for optimal coverage. Further, we recommend to cover most vulnerable locations first, but avoid an overlap in sensor ranges unless the distribution of the object position is highly irregular. As a rule of thumb, one may call a distribution highly irregular if there exist pairs of points such that the distance between two points in such a pair is $\leq 2r$ while the value of the density differs by a factor $1 - p$.

For the detection problem, we state that several observations of the same object are absolutely necessary to report an alarm with reasonable certainty. A classical

5 Increasing Detection Performance of Surveillance Sensor Networks

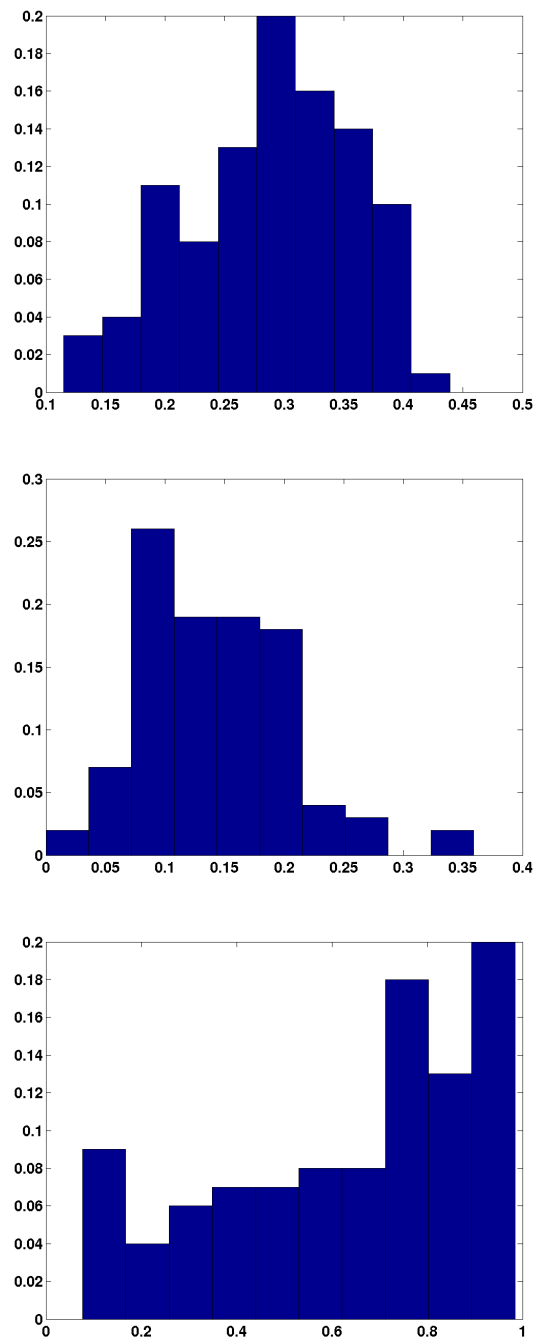


Figure 5.10: Estimate of p_{coverage} of SWSN. (Top) Network A. The mean of the estimate is 0.2884 and the standard deviation is 0.0698. (Middle) Network B. The mean of the estimate is 0.1420 and the standard deviation is 0.0631. (Bottom) Network C. The mean of the estimate is 0.6367 and the standard deviation is 0.2658.

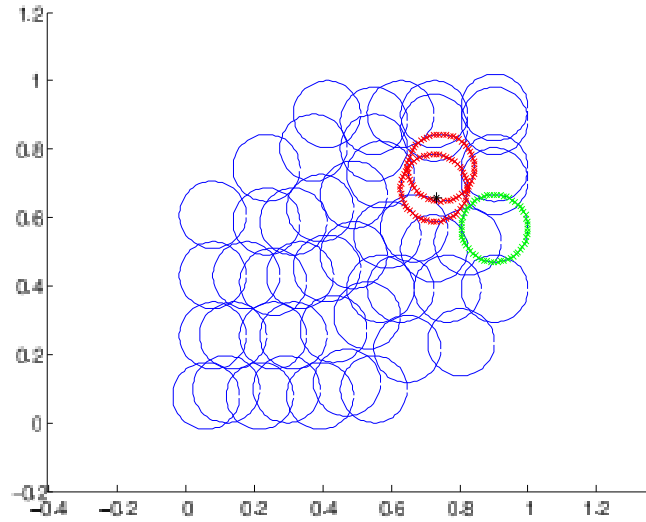


Figure 5.11: One time instant of a simulation run of the SWSN of 50 sensors containing a moving object (*). Highlighted circles: two correct intrusion alarms and one false alarm.

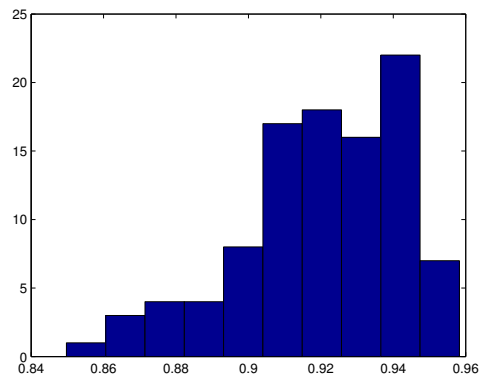


Figure 5.12: Estimate of $p_{\text{detection}}$ of the SWSN in Figure 5.11. The mean of the estimate is 0.9205 and the standard deviation is 0.0224.

hypothesis testing works well only if multiple sensors overlap in the same location. Otherwise, one must use information from consecutive readings of the SWSN. In the latter case, either a Bayesian approach or a hidden Markov model (HMM) approach can be used for object detection. To the best of our knowledge, the HMM approach involving the Viterbi algorithm to filter out the noise of non-detections and false alarms, has never been used in an SWSN before. The advantage of this approach is that it allows to pre-compute off-line all observation patterns that signal an intruder. Then the decision rule is very simple: report an intruder if one of the alarming patterns is observed. The HMM techniques in the SWSN context definitely deserve further study.

In this research, one could clearly see that the two problems under consideration are closely related. Although each of the proposed methods may be useful in its own right, it is essential to develop an integral approach to sensor deployment and intruder detection, in order to enhance the SWSN performance. In the last numerical example (see Section 5.5), we demonstrated that our techniques can be successfully combined, thus considerably increasing the efficiency of the network.

We would like to add that, potentially, our methods can be also used for tracking a target advancing through the area. For instance, by observing a simulation run of a moving object in the last numerical example, one could see that in spite of occasional false alarms, the correct intrusion alarms indicate a clear path that can be easily deciphered from multiple sensor readings.

Bibliography

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.
- [2] E. Ardizzone, M. La Cascia, G. L. Re, and M. Ortolani. An integrated architecture for surveillance and monitoring in an archaeological site. In *Proceedings of the third ACM international workshop on Video surveillance & sensor networks*, pages 79–86, Hilton, Singapore, 2005. ACM, New York, NY, USA.
- [3] Y. Bar-Shalom and X. R. Li. *Estimation and Tracking: Principles, Techniques and Software*. Artech House, 1993.
- [4] J. H. Conway and N. J. A. Sloane. *Sphere Packings, Lattices and Groups*, volume 290 of *Grundlehren der mathematischen Wissenschaften*. Springer-Verlag, New York, third edition, 1998.
- [5] H. Delic and P. Papantoni-Kazakos. Robust decentralized detection by asymptotically many sensors. *Signal Processing*, 33(2):223–233, 1993.

- [6] G.D. Forney Jr. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.
- [7] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh. Vigilnet: An integrated sensor network system for energy-efficient surveillance. *ACM Transactions on Sensor Networks*, 2(1):1–38, 2006.
- [8] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 270–283, New York, NY, USA, 2004. ACM.
- [9] P. Korteweg, M. Nuyens, R. Bisseling, T. Coenen, H. van den Esker, B. Frenk, R. de Haan, B. Heydenreich, R. van der Hofstad, J. in 't panhuis, L. Spanjers, and M. van Wieren. Math saves the forest: analysis and optimization of message delivery in wireless sensor networks. In Erik Fledderus, Remco van der Hofstad, Ellen Jochemsz, Jaap Molenaar, Tim Mussche, Mark Peletier, and Georg Prokert, editors, *Proceedings 55th European Study Group Mathematics with Industry Eindhoven 2006*, pages 117–140. Technische Universiteit Eindhoven, 2006.
- [10] E. Onur, C. Ersoy, H. Delic, and L. Akarun. Surveillance wireless sensor networks: Deployment quality analysis. *IEEE Network*, 21(6):48–53, November – December 2007.
- [11] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [12] R. Roman, C. Alcaraz, and J. Lopez. The role of wireless sensor networks in the area of critical information infrastructure protection. *Information Security Tech. Rep.*, 12(1):24–31, 2007.