POWER-AWARE CONTROL STRATEGIES IN WIRELESS SENSOR NETWORKS

A Thesis
Presented to
The Academic Faculty

by

Hassan Jaleel

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2013

POWER-AWARE CONTROL STRATEGIES IN WIRELESS SENSOR NETWORKS

Approved by:

Dr. Magnus Egerstedt, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Douglas Blough
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Yorai Wardi
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Jorge Cortés
School of Mechanical and Aerospace Engineering
University of California, San Diego

Dr. Fumin Zhang
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: October 30, 2013

To my Parents

Their love, guidance, support, and prayers

made me who I am.

ACKNOWLEDGEMENTS

I cannot begin to express my gratitude to my advisor and mentor, Dr. Magnus Egerstedt, for his continuous guidance, profound belief in my abilities and patience that cannot be underestimated. The completion of this thesis would not have been possible without his unparalleled support, insightful suggestions and constructive criticism. I still remember his talk in an IEEE seminar five years ago which introduced me to the field of multi-agent systems, a field I had no prior exposure to. His highly energetic personality and passion for his work made the choice of a PhD advisor simple for me, and I am extremely thankful that he accepted me as a graduate student in his research group. Since then, every day working with him has been a joyful experience full of learning. Looking back after five years, I can confidently say that working for him was the best choice that I made as a graduate student.

I would like to express my deepest appreciation to Dr. Yorai Wardi, Dr. Fumin Zhang, Dr. Douglas Blough, and Dr. Jorge Cortés for agreeing to be on my PhD committee. Special thanks to Dr. Yorai Wardi who extended a great amount of assistance and taught me extremely useful concepts of optimal control theory. I had a lot of long meetings with him in which the topics of discussion varied form deep concepts of optimal control theory to the subtleties of various cuisines. Thanks to Dr. Fumin Zhang for teaching an interesting course on stochastic systems, which gave me useful ideas for my research. I am grateful to Dr. Doug Blough for his valuable insights on the field of topology control, which helped me focus my research efforts on relevant problems. I am also thankful to Jane Chisholm from the Center for the Enhancement of Teaching and Learning at Georgia Tech for an exceptional course on academic writing that helped me throughout my research. Moreover, I very

much appreciate the support of Higher Education Commission of Pakistan, Fulbright scholarship program, and National Science Foundation for providing the necessary funds for my education and research at Georgia Tech.

I had a great pleasure of working with some outstanding researchers at GRITS lab in the last five years. Particularly helpful to me during this time was Dr. Amir Rahmani who helped me a lot during the initial phase of my research. I would also like to extend my gratitude to my friend and colleague, Waseem Abbas, for his encouragement and support throughout the duration of this project. We took several courses together and had a lot of fun in exploring exciting new ideas particularly in the fields of control theory, algebraic graph theory, and networked control systems. His advice, suggestions, and insightful comments were instrumental in driving my research.

I am grateful to all of my friends at Georgia Tech especially Umer Tariq, Wasif Tanveer, Abdul Majid Naveed, Ubaid Ullah Fayyaz, Sajid Saleem, Bashir Akbar, Farhan Aziz, Ahmad Usman, and Saad Bin Nasir for making my time as a graduate student exciting and enjoyable. I will always remember the time that I spent in their company with all the late night dinners, long and sometimes heated discussions on a wide range of topics from which I learned a lot, and memorable road trips throughout United States. Without the company of such friends, five years of PhD would have been dull and exhausting.

Finally and most importantly, I am deeply indebted to my family because without their prayers, support and guidance, I would not have been at Georgia Tech in the first place. It was the confidence that my parents showed in my abilities, and the curiosity of knowledge that I inherited from them that motivated me to take up this challenge of doing PhD from one of the most prestigious universities of the world. All the accomplishments that I have made in my life up till now, I owe them to my family, and particularly to my parents.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

As the trends towards decentralization, miniaturization, and longevity of deployment continue in many domains, power management has become increasingly important. In sensing and communications networks, power management has long been a part of the design paradigm. However, an underlying assumption in most of the existing work is that the performance of the sensing devices remain the same throughout their lifetime, which is not always true. Moreover, when mobility is added to the mix, power management is not well understood. Thus, the research presented in this thesis is focused on developing power-aware control strategies for maximizing the lifetime of wireless sensor networks.

This research spans over two broad classes of wireless sensor networks, namely, static networks (comprising of agents with no mobility) and mobile networks (comprising of agents with mobility). For the case of static networks, the problem of the effects of power decay on the performance of an individual sensor and on the entire network is identified and is addressed for networks in which sensing devices are randomly deployed in a region of interest. In particular, networks comprising of agents whose sensing range model is a function of available power akin to those of RF- or radar-based sensors are examined and the performance of each sensor in these networks is related to its available power. Moreover, to compensate for the effects of decrease in available power on the global performance of the network, probabilistic scheduling controllers are developed that maintain a desired probability of event detection under two sensing models: Boolean and non-Boolean.

In addition to this problem of the effects of power decay on the performance of a sensor network, a novel probabilistic sleep-scheduling scheme is proposed in

which neighboring sensors are only allowed minimum coordination with each other for making intelligent switching decisions so that a desired level of partial coverage can be achieved and energy can be conserved. This scheme is based on the concept of a hard-core point process from stochastic geometry, in which neighboring points are allowed to interact with each other through some predefined interaction laws.

For the case of mobile networks, the goal of this research is to propose a solid framework for distributed power-aware mobility strategies that can achieve any desired global objective while minimizing total energy consumption. This goal is achieved by first exploring fundamental trade-offs among various modes of operations of mobile devices and then exploiting these trade-offs for minimizing energy consumption. The key idea is that different operating modes (e.g., low-power/high-power, on/off, or mobile/static) have different performance and power characteristics and these characteristics can be traded off by either optimally switching among these modes or by assigning optimal weights to each of them. For instance, in the case of mobile agents forming a relay network, the choice is typically between moving (higher power consumption) to improve the sensing profile, or remaining in the same location and transmitting data at higher power levels. Thus, the problem of minimizing energy consumption is formulated as an optimal control problem and for solving this problem efficiently, numerical techniques are employed. Through this framework, a whole class of power-aware controllers emerge for solving canonical problems in multi-agent systems like connectivity maintenance, rendezvous, and coverage control in a decentralized manner while minimizing power consumption.

CHAPTER I

INTRODUCTION

Wireless sensor networks are used in a wide range of applications such as detecting intruders in restricted areas; monitoring hazardous and potentially hostile environments like detecting fires in a forest or oil spillage in the ocean; neutralizing threats like land mines in war zones ; and performing search and rescue missions in the case of disasters and natural calamities. To achieve desired objectives, a mature body of work has emerged in the last decade on how to control both static and mobile sensor networks in a distributed manner (see e.g., [2], [3], [5], [6], [7], [8], [10], [11], [12], [13], [14], [16], [17], and [18]). However, one key limitation of such large-scale systems is that they are inherently power sensitive. To deploy a large collection of nodes in unknown territories as stand-alone units, payload issues (e.g., battery sizes) become a significant problem. Therefore, a critical problem, which is a subject of active research in the wireless sensor networks community, is power conservation.

In static sensor networks, power management has long been a part of the design methodology and redundancy in sensor deployment is utilized to maximize the lifetime of a network by developing intelligent switching schemes for turning the nodes on or off (see e.g., [19], [20], [21], [22], [23], [24], [25], and [26]). However, because of the fact that these networks are typically deployed for the purpose of monitoring critical areas for long periods of time, and comprise of a large number of low-cost, low-power devices with limited sensing, processing, and communication capabilities, there is an aspect of power-awareness that is neglected in the existing literature. In most of the existing sensor scheduling schemes, there is an inherent assumption that the performance of sensing devices remain constant throughout the lifetime of a network and this

assumption is not always true. Because of the low quality of the constituent devices and the harshness of the environments in which they are deployed, the batteries of these devices start to deteriorate as a result of which their available power decay with time. This decrease in available power has a direct impact on the performance of these devices, but the relationship between available power and performance depends on the type of the sensor that is used. For instance, if a system consists of vision-based sensors, power levels may be related to the maximally available frame rate; for RF- or radar-based sensors, the footprint area may be reduced as the power decreases; and in communication networks, latency issues may arise because of reduced power levels.

To support this point, in the summer of year 2002, a wireless sensor network comprising of approximately 50 nodes was deployed on an uninhabited island for habitat monitoring. This was one of the first networks of this size that ran unattended for a period of four months over which 1.1 million readings were received from the network. Using this data, the designers analyzed the performance of the network to deepen their understanding of the practical issues in the network design and later published their findings in [9]. The crux of their analysis was that the performance of the network was far below the level it was designed for. One important observation that they made after analyzing the data was that the batteries of these nodes were unable to maintain constant terminal voltage[1]. In fact, the terminal voltage decreased continuously not just at the end but throughout the operational lifetime of a node. Since the power that is delivered to a device is directly related to its terminal voltage, this implies that for each node the available power decreased with time. This decrease in available power must have a negative impact on the performance of sensing devices which violates the assumption of constant performance and results in a mismatch between the assumed system model and the actual system.

[1]The nodes were powered by two standard AA batteries

Thus, the first contribution of the research presented in this thesis is to minimize this mismatch between the actual system and its assumed model for a sensor network comprising of sensing devices whose sensing range is a function of available power. Note that this choice of sensors is made because of the fact that there is a large variety of sensors that belong to this class. To achieve this objective, we explicitly couple the global performance of a sensor network with available power. The relationship between available power and the performance of a network is used during the network design phase to ensure that a desired performance is maintained regardless of the adverse effects of power consumption. The details of this work were published in ([35], [36], [37], and [38]).

The second contribution of this research is related with a more traditional aspect of power-awareness, i.e., efficient utilization of available energy resources to maximize system lifetime. In this regard, a probabilistic power-efficient sensor scheduling scheme is proposed that is based on the concept of a hard-core point process form stochastic geometry to minimize communication among neighboring sensors in making switching decisions. Most of the existing schemes that are available in the literature are designed to maintain complete coverage throughout the lifetime of a network by ensuring that switching a particular sensor off does not deteriorate the coverage profile of a network. Maintaining complete coverage is important especially for time critical events that must be detected immediately. However, this complete coverage is typically achieved at the expense of considerable control and communication overheads, and these overheads make this objective over restrictive for applications that can tolerate partial coverage and some delay in the detection of an event. Thus, for certain applications like environmental monitoring for precision agriculture or detection of mobile targets, power consumption can be reduced by relaxing the desired performance criterion, which in this case is coverage. This tradeoff between power

consumption and desired performance criterion is exploited and a probabilistic switching scheme is proposed that can ensure a required level of partial coverage throughout the lifetime of a network while minimizing the overhead involved in making switching decisions. This work was published in [39].

The final contribution of this research is a framework for developing decentralized energy-efficient mobility controllers for mobile sensor networks. With the development of low-cost and fairly reliable mobile sensing devices like Packbot [59], Robomote [60], and Khepera [61], mobility has become an integral component of wireless sensor network. In recent years, a considerable amount of work has been done on utilizing mobility for improving the performance of these networks (see e.g., [65], [69], [70], [71], and [74]) without paying any serious attention to the added cost of mobility. However, energy-efficiency is more important from mobility vantage point than from a sensing vantage point because among the possible operations that can be performed by an agent, it is generally estimated that sensing and communication consume orders of magnitude more energy than processing while mobility is more expensive that all of them [60]. Moreover, the fact that these devices are typically powered through batteries that cannot be recharged in most of the cases makes power management one of the most critical issues in the design of mobile sensor networks.

Despite the fact that available energy is a major bottleneck in the design of mobile networks, very little work has been done on the design of energy-efficient mobility strategies, and energy management is not well understood from mobility vantage point. Thus, the framework that is presented in this thesis first explore fundamental trade-offs between different modes of operation of a mobile device (e.g., low-power/high-power or mobile/static). Then these trade-offs are exploited to minimize total energy-consumption because of both mobility and communication in a mobile sensor network. Although the problems of minimizing energy consumption due to either mobility or communication have been individually studied in the past,

co-optimization of both mobility and communication to minimize the total energy consumption started to receive attention just recently. Furthermore, when this problem is studied in the context of distributed systems, there is very little existing work. Thus, the purpose of this research is to provide a framework that will provide a solid foundation for thorough investigation of this problem in future. This work resulted in ([40], [41], and [43]).

1.1 Background

To develop power-aware control strategies for both static and mobile networks, we bring together ideas from the well-established fields of probability theory, stochastic geometry, and algebraic graph theory. Therefore, in the following sections, we will briefly introduce some fundamental concepts of these fields that are used in this work. However, before that, we provide a brief overview of some of the important sensor scheduling schemes that exist in the literature.

1.1.1 Sensor Switching Schemes

The need to minimize power consumption and enhance the lifetime of a sensor network has motivated a huge body of research in the wireless sensor networks community. A plethora of scheduling schemes have been proposed to minimize energy consumption and maintain given performance criteria that can be coverage, connectivity, packet delays, response time to an event, or a combination of any of these parameters. However, in this work, we are primarily concerned with coverage. For any level of required coverage, the objective is to propose a scheduling scheme that is scalable and distributed in a sense that each sensor must make its decisions based on the information of its neighbors only. The basic idea that is used in most of the existing work is as follows. If the number of deployed sensors is equal to the minimum number that is required to cover an area of interest, then to ensure coverage, all the sensors must be active all the time. This means that the only way to conserve power is to

design efficient hardware with low-power requirements. However, if there are redundant sensors, then at any time instant, only such a subset of sensors is required to be in the active state whose combined footprint [2] can cover the entire area of interest. This problem of efficiently selecting the minimum subset of sensors that can cover the entire area of interest at every decision time has inspired a lot of researchers and a long list of sensor scheduling schemes is available in the literature to address this problem.

One prevalent approach to efficiently select a subset of sensors is to use heuristics to solve set-cover problem (see e.g., [19] and [20]). The idea is to divide the sensors into disjoint sets such that each set can cover the entire area, and only one set of sensors can be active at any particular time. The lifetime of a network in this case is directly related to the number of sets. In another approach, sensors use some form of information of their neighbors, like their IDs, exact locations, or transmission power levels, to decide when they should switch to active state (see e.g., [27] and [28]). The schemes that follow one of the above approaches are normally deterministic and ensure complete coverage all the time.

In contrast to deterministic schemes, some probabilistic schemes also exist in the literature. In one such scheme [22], which also ensures complete coverage, each sensor turns on after a random wait time and probes its environment to check whether its footprint is completely covered by its neighbors or not. A sensor remains on if its sensing region is not completely covered, and it returns to the sleep state if the sensing region is completely covered. Instead of complete coverage, some schemes ensure partial coverage and are intended for applications like object tracking (see e.g., [23], [26], and [29]) and rare event detection (see e.g., [24]) that can tolerate some delay in event detection. Furthermore, in an effort to generalize system models, some work

[2]The combined footprint of a set of sensors is the union of the footprints of all the sensors of that set.

has been done to incorporate more elaborate models like non-symmetric footprints and shadowing effects because of transmission channel and study their effects on the coverage properties of a network (see e.g., [25]).

Next, we briefly present some of the concepts from stochastic geometry and algebraic graph theory that are important in the context of this work.

1.1.2 Stochastic Geometry

Definition 1.1.1. *[31] " Stochastic geometry is the study of random mechanism governing the positioning and configuration of random sets on the line, or in the plane, or indeed in any metric space."*

The history of stochastic geometry dates back to the early 19^{th} century when it was studied under the umbrella of probability theory. However, it was not until 1970s, when stochastic geometry started to develop as a separate field. The primary motivation for the study of random patterns of geometrical objects originated from problems in the fields of biology, physics, geology, and material research where there was a need to develop mathematical models for the measurements of spatial data. The fundamental object that is used in stochastic geometry is a point process, which is a model of randomly distributed points in some space.

Definition 1.1.2. *[30] A point process Φ on $\mathbb{R}^2$ is a random sequence of points,*

$$\Phi = \{x_1, x_2, \ldots\} \qquad x_i \in \mathbb{R}^2,$$

that is locally finite and simple.

Locally finite means that any bounded subset of $\mathbb{R}^2$ contains only a finite number of points of Φ and *simple* means that two points cannot overlap, i.e., $x_i \neq x_j$ if $i \neq j$. In Definition 1.1.2, a point process is defined as a random set of discrete points. An alternative is to define a point process as a random measure counting the number of points in some spatial region.

Definition 1.1.3. *[30] A point process* Φ *is a measurable mapping of a probability space* $(\Omega, \mathcal{A}, P)$ *into* $(\mathbb{N}, \mathcal{N})$*,*

where $(\Omega, \mathcal{A}, P)$ is a probability space, $\mathbb{N}$ is the family of all sequences φ of points in $\mathbb{R}^d$ that are locally finite and simple, and $\mathcal{N}$ is defined as the smallest σ-algebra making all the mappings $\varphi \to \varphi(B)$ measurable.

In this work, our field of interest is $\mathbb{R}^2$ because we are interested in sensors that are randomly deployed in a planar region. The location of a sensor that is randomly deployed in $\mathbb{R}^2$ can be viewed as a point in a point process. Thus, the theory of point processes provides a natural platform for modeling randomly deployed networks. A point process that is simplest to model is the one in which all the constituent points are uniformly distributed in a compact set $\mathcal{D} \subset \mathbb{R}^2$ and all these points are independent. If such a process comprises a finite number of points, say N, then it is called a binomial point process of N points. However, if the number of points increases (goes to ∞ in the limiting case) in such a manner that $N/\|\mathcal{D}\| \to \lambda$, where $\|\mathcal{D}\|$ is the area of domain $\mathcal{D}$ and λ is the expected number of points per unit area, then we have a Poisson point process.

Definition 1.1.4. *[31] A point process* Φ *is a Poisson point process if it satisfies the following two conditions:*

1. $P\left(\bigcup_i (\varphi(B_i) = n_i)\right) = \sum_i P(\varphi(B_i) = n_i)$*, where* $B_1, B_2, \ldots$ *are disjoint subsets of* $\mathbb{R}^2$ *and* $\varphi(B_i)$ *is the number of points of* Φ *in the set* B_i*.*

2. *For any set* $B \subset \mathbb{R}^2$*,* $\varphi(B)$ *is Poisson distributed.*

Moreover, if λ is constant throughout the area, then Poisson process is called homogeneous Poisson process. In addition to Poisson and binomial processes, in which all the points are independent, significant research efforts have been directed towards modeling generic random patterns of points that have mutual dependences. These

processes may contain clusters of points or enforce some restrictions on the distances among the constituent points. Point processes that exhibit inter-point interactions have great significance in the context of randomly deployed sensors since, depending on the applications, these processes can be used to model various switching patterns in a sensor network. However, Poisson and binomial processes are central to the theory of point processes because they assume no interactions among the points and are used as a reference against which the degree of interactions in a general point process is measured. In fact, a number of point process models can be generated from a Poisson process, and the analysis of these new processes is relatively simple because of the fact that the newly generated point processes can be analyzed in terms of the generating Poisson process. To produce new processes from a Poisson process Φ with intensity λ, three fundamental operations can be performed:

- Thinning: Deletion of points from Φ.
- Clustering: Replacement of every point of Φ with cluster of points.
- Superposition: Union of point processes.

However, in this work we are only concerned with the first of the three operations, i.e., thinning.

Definition 1.1.5. *A thinning operation deletes points from the process Φ according to some specified rule.*

If each point of Φ is deleted randomly with probability $1 - p$, in which p is the retention probability, and the deletion of each point is independent of locations and deletions of all the other points of Φ, then this operation is called p-thinning. The resulting point process Φ_p is also a Poisson point process with intensity $\lambda_p = p\lambda$. A generalization of p-thinning is $p(x)$-thinning, in which each point x of Φ is deleted with probability $1 - p(x)$ and is retained with probability $p(x)$, where $0 < p(x) < 1$

can be a deterministic function of x or a random function that is obtained from some random field. The process resulting form $p(x)$-thinning is again a Poisson process and its intensity measure is given by

$$\Lambda(B) = \int_B p(x)\Lambda(dx), \tag{1.1}$$

where Λ is the intensity measure of Φ.

Definition 1.1.6. *"The intensity measure of a point process Φ is defined as*

$$\Lambda(B) = \mathbb{E}(\Phi(B)) = \int \varphi(B)P(d\varphi) \quad \textit{for Borel } B, \tag{1.2}$$

so $\Lambda(B)$ is the expected number of points in B.

If the point process is stationary than $\Lambda(B) = \lambda\|B\|$, where $\|B\|$ is the size of the set B. In our case, $\|B\|$ will be the area of the set B.

A process that results from either p-thinning or $p(x)$-thinning of a Poisson process is still an independent process since there is no interaction between the points. Thus, these operations are called *independent* thinning. In contrast, there is another class of thinning operations called *dependent* thinning in which points of Φ are deleted or retained depending on their configuration. A process that is obtained as a result of dependent thinning is not Poisson because the constituent points are no longer independent. One example of such a process is a hard-core point process.

Definition 1.1.7. *[30] A hard-core point process is a point process in which the distance between any pair of points cannot be less than some specified value.*

A hard-core point process is important because it offers a basic framework for designing coordinated scheduling schemes in which there cannot be more than a specified number of active sensors in any given area.

Up till now, we have presented point processes from stochastic geometry that can be used for modeling random deployment of sensors. In addition to modeling

sensor deployment, we need to characterize the performance of a sensor network in terms of a given performance criteria. Since we are interested in sensor networks that are deployed for monitoring purposes, a natural performance criterion is the level of coverage a network is required to maintain throughout its lifetime. To estimate the performance of a sensor network in terms of its coverage level, we present coverage processes that are defined on top of point processes.

Definition 1.1.8. *Given a point process* $\Phi = \{x_1, x_2, \ldots\}$ *and a countable collection of sets* $S = \{S1, S2, \ldots\}$ *such that*

$$x_i + S_i = \{x_i + y \quad \textit{for all} \;\; y \in S_i\}.$$

Then a coverage process is defined as

$$\mathcal{C} \triangleq \{x_i + S_i : i = 1, 2, \ldots\}.$$

All the elements of the set S are independent and identically distributed random sets in $\mathbb{R}^d$. Moreover, the set S is independent of the point process Φ. Next, we define a Boolean model as

Definition 1.1.9. *Given a coverage process* $\mathcal{C}$ *as defined in Definition 1.1.8, the union of all the sets of* $\mathcal{C}$*, i.e.,*

$$\mathbb{S} = \bigcup_{i=1}^{\infty} (x_i + S_i) \tag{1.3}$$

comprises a Boolean model.

A Boolean model is also referred to as a germ-grain model in the literature, where the points $(x_1, x_2, \ldots)$ are called the germs and the sets $(S_1, S_2, \ldots)$ are called the grains. Thus, in our case, the location of a sensor, which is modeled as a point in a point process, corresponds to a germ, and its footprint corresponds to a grain, in the germ-grain model. Therefore, by using the germ-grain model, we can completely

characterize the coverage properties of a sensor network. Given a sensor network that is represented by the germ-grain model, we are interested in determining whether any given area is covered by the network or not. This is exactly what is represented by *vacancy* in stochastic geometry. Given a set $\mathcal{R} \subset \mathbb{R}^k$, the vacancy V in the set $\mathcal{R}$ is defined as the content of $\mathcal{R}$ that is not covered by the coverage process $\mathcal{C}$. Formally

$$V = V(\mathcal{R}) = \int_{\mathcal{R}} \bar{\mathbf{1}}(x)dx, \tag{1.4}$$

where $\bar{\mathbf{1}}(x)$ is defined as

$$\bar{\mathbf{1}}(x) = \begin{cases} 1 & \text{if for all i, } x \notin x_i + S_i \\ 0 & \text{otherwise} \end{cases}$$

1.1.3 Algebraic Graph Theory–Basics

In this section we introduce some basic tools from algebraic graph theory that is needed to formulate and analyze the problems in the later sections [33]. The basic object under investigation is a graph $\mathcal{G}(V, E)$ with a vertex set V and an edge set $E \subseteq V \times V$. If the edge set is unordered, i.e., $(v_i, v_j) \in E \Leftrightarrow (v_j, v_i) \in E$, we say that the graph is undirected. An undirected graph in which a path exists between any pair of vertices is called a connected graph, which is exactly what we need to establish convergence in the majority of problems in the distributed multi-agent systems.

If the edge set is ordered, the graph is directed (referred to as a digraph) and in this case, different types of connectivity related concepts will prove important. A digraph is called

- *weakly connected* if its disoriented graph (the graph obtained by removing the direction from the edges) is connected,
- *rooted out-branching* if it does not contain a directed cycle and has a vertex v_r (root) such that for every other vertex $v_i \in V$, there is directed path from v_r to v_i, and

- *balanced* if $deg^{in}(v) = deg^{out}(v)$, $\forall v \in V$, i.e., the number of edges going in to any node is the same as the number of edges going out.

Consider an undirected graph $\mathcal{G}(V, E)$ and associate an arbitrary orientation to its edges, $\sigma : E(\mathcal{G}) \to \{-1, 1\}$, such that $\sigma(i, j) = -\sigma(j, i)$. As a result of assigning orientation to each edge, a new digraph $\mathcal{G}_\sigma(V, E, \sigma)$ is generated, for which an incidence matrix, $\mathcal{I} = [e_{ij}]$, can be defined as

$$e_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is the head of the edge } e_j \\ -1 & \text{if } v_i \text{ is the tail of the edge } e_j \\ 0 & \text{otherwise.} \end{cases}$$

Using the incidence matrix, a new matrix called graph Laplacian is defined as $\mathcal{L} = \mathcal{I}\mathcal{I}^T$, where $\mathcal{L}$ is independent of the choice of orientation σ, and is always symmetric and positive semi-definite. Let $\lambda_1, \lambda_2, \ldots, \lambda_N$ be the (non-negative and real) eigenvalues of $\mathcal{L}$, indexed such that $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. Then the multiplicity of zero eigenvalues of $\mathcal{L}$ is equal to the number of connected components of the graph. For instance, if we have a connected graph, then the eigenvalues of $\mathcal{L}$ will be $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$, and the eigenvector corresponding to λ_1 is $\mathbf{1}$, where $\mathbf{1}$ denotes a vector with all the entries equal to 1.

In this work, we are interested in dynamic graphs whose edge sets change as the agents move in and out of the footprint of each other. In particular, we will study Δ-disk graphs where the vertex set $V = \{1, \ldots, N\}$ corresponds to the indices of the different agents, and $(v_i, v_j) \in E \Leftrightarrow \|x_i - x_j\| \leq \Delta$, for some given $\Delta > 0$. If the footprint radius is the same for all the agents, then these graphs are undirected. On the other hand, if the footprint radius varies among the agents, then the graph is directed, which means that $(v_i, v_j) \in E \Leftrightarrow \|x_i - x_j\| \leq \Delta_j$. This expression implies that information about agent i will be available to agent j if agent i is no further than Δ_j away from agent j, where $\Delta_j > 0$ is the footprint radius of agent j.

CHAPTER II

POWER-AWARE SCHEDULING OF WIRELESS SENSOR NETWORKS WITH DYNAMIC FOOTPRINTS

In this chapter we present power-aware scheduling schemes for sensor networks that consist of sensing devices with dynamic footprints to maintain a desired event detection probability. The footprints of the sensors comprising these networks are dynamic in nature because of the fact that variations in available power have a direct impact on the performance of sensing devices. Therefore, we select the area of a sensor footprint as a performance metric and use explicit relationship between footprint area and available power to quantify the effects of variation in available power on the performance of sensing devices. To compensate for this variation in sensor performance because of change in available power, we propose power-aware scheduling schemes in which sensors use their available power to determine their performance metric at each decision time and then update their control parameter accordingly such that the desired event detection is maintained while consuming minimum power. The impact of variations in available power on sensor performance was a missing link in the existing literature, and is addressed in this work for the first time. Initially this problem is simplified by assuming a particular power-decay model and a sensor footprint model. Later, these assumptions are removed and the derived results are extended for generalized system models.

The schemes that we proposed in this work are random in nature, which implies that we are interested in applications that can tolerate partial coverage of the area of interest, and also some delay in event detection since random schemes cannot ensure complete coverage (see e.g., [26], [10], and [23]). Therefore, if an application

requires complete coverage, then random schemes are not an option, and there is a plethora of coordinated schemes in the literature that can ensure complete coverage of the area of interest (see e.g., [19], [20], [27], and [28]). However, this complete coverage is achieved at the expense of additional cost in terms of power consumption and relatively expensive devices because sensors either have to communicate with other sensors or they require on-board GPS chips to get exact location information of themselves and their neighbors before they can make switching decisions. Moreover, there are applications which only require partial coverage. For instance, in the case of environment monitoring for precision agriculture where the purpose of the network is to monitor environmental factors like humidity and soil moisture level, the quantities of interest are slowing varying over space, so we do not need to cover each and every point of the area of interest. Similarly, if we are interested in detecting a moving target, then with high probability the target will move to an area that is covered by some sensors. Therefore, for such applications, even partial coverage is sufficient to fulfill the objective.

2.1 Exponential Power-decay and Circular Footprint

2.1.1 System Description

Consider a domain $\mathcal{D} \subset \mathbb{R}^2$ in which a large number of sensors are randomly deployed for monitoring purposes. In random deployment, sensors can either be dropped from a plane flying over the region of interest or any vehicle driving across it [1]. This deployment scheme is preferred for large networks because it allows to deploy a large number of nodes with minimum effort. However, random deployment does not provide any information about the particular locations of sensors which prohibits design and analysis of such networks. Thus, the first task is to model this sensor deployment in an efficient manner. By using the concepts from Stochastic Geometry presented in Chapter 1, this random deployment is modeled as a homogeneous Poisson point

process. Once it is established that the sensor deployment is modeled as a homogeneous Poisson point process with some intensity λ, then the number of sensors in any region of area A is determined using Poisson distribution.

$$P_n(A) = \frac{(\lambda A)^n e^{-\lambda A}}{n!}, \tag{2.1}$$

where $P_n(A)$ is the probability of having n sensors in some area A.

Next, the model of the sensing agents considered in this work is specified. Each sensing agent located at $x_i \in \mathbb{R}^2$

- is a stationary device whose sensing range is a function of available power,
- is battery powered where the available power decreases with time, and
- has a circular disk of radius Δ_i, centered at x_i, as its sensing region called the footprint of a sensor denoted as $\mathcal{B}_{\Delta_i}(x_i)$.

It is assumed that all the deployed sensors are initially identical, i.e., they have same battery power and same sensing, processing and communication capabilities. To conserve power, we let the sensors be on with probability $q \in [0, 1]$. Each sensor can switch its state from on to off or vice versa only at discrete time instances $k\Delta t$ (or simply at instance k), where Δt is the sample time. The activation (or lack thereof) of a sensor at instance k is maintained throughout the interval $[k, k + 1)$ of length Δt. A sensor can sense only when it is on, and for an event to be detected, it must be within the footprint of at least one on sensor. Moreover, when a sensor is on, it consumes power. Using the discrete time version of the battery dynamics in [44], we model the power levels of each sensor in the on state as

$$\eta(k + 1) = \eta(k) - \Delta t \gamma \eta(k),$$

where γ is the decay constant and $\eta(k)$ represents the battery power available for sensing at time instant k. Let $\sigma(k)$ be a switching signal for each sensor defined as

$$\sigma(k) = \begin{cases} 1 & \text{if a sensor is on at time instant } k \\ 0 & \text{if a sensor is off at time instant } k \end{cases}$$

Since a sensor is on with probability q, the expected value of $\sigma(k)$ is $E\{\sigma(k)\} = \hat{\sigma}(k) = q(k)$, and because power is consumed only when a sensor is on, the power decay model can be modified as

$$\eta(k+1) = \eta(k) - \Delta t \gamma \sigma(k) \eta(k), \tag{2.2}$$

and the expected power level of each sensor is

$$\hat{\eta}(k+1) = \left[\prod_{i=0}^{k} (1 - \Delta t \gamma q(i)) \right] \eta(0). \tag{2.3}$$

Moreover, for all $t \in [k, k+1)$, simply set $\eta(t) = \eta(k)$.[1]

2.1.2 Probability of Event Detection

Consider a non-persistent event that takes place at some point $x_e \in \mathcal{D}$ at some arbitrary time $t \in [k, k+1)$. A non-persistent event, which a sensor can detect only at the time of its occurrence, does not leave a mark in the environment. Hence, this event can be detected if it is within the footprint of at least one sensor in the on state at time k. Detection probability of a non-persistent event that can occur uniformly across the region of interest is equivalent to the expected coverage provided by the network. The probability of an event going undetected by a non-decaying sensor network (i.e., a network of sensors whose footprints and probabilities q do not change with time) deployed randomly with intensity λ is

$$P_u = e^{-\lambda A q}, \tag{2.4}$$

where A is the area of the sensor footprint with radius r and q is the probability of a sensor being on [26]. The proof of Equation (2.4) is based on the observation that

[1]Note that in this analysis, potential power consumption resulting from switching between the on and off states is not considered.

the probability of an event going undetected is equal to the sum of probabilities of the event being inside the range of $n \in \{0, \ldots, \infty\}$ sensors, all of which are off.

The next step is to investigate how this result changes in networks in which sensor footprints are reduced because of power consumption when the sensors are on. In [46], it was shown that if the sensor range model was based on the RF-power-density function for an isotropic antenna, then the footprint of the sensor was proportional to it available power, i.e.,

$$\Delta^2(t) \propto \eta(t), \tag{2.5}$$

where $\Delta(t)$ is the radius of the sensor footprint at time $t \in [k, k+1)$. Hence, the area of the footprint of a sensor at time t is

$$A(t) = \pi\Delta^2(t) = \alpha\eta(t), \tag{2.6}$$

where $\alpha = \zeta\pi$ is a constant with ζ being the constant of proportionality in Equation (2.5). If the power, η, in Equation (2.6) is substituted with the expected power, $\hat{\eta}$, from Equation (2.3), the expected footprint area of a sensor becomes

$$\hat{A}(k) = c\left[\prod_{i=0}^{k-1}(1 - \Delta t\gamma q(i))\right], \tag{2.7}$$

where $c = \alpha\eta(0)$ is a positive constant.

Lemma 2.1.1. *The probability of an event being detected by a decaying sensor network is given by*

$$P_d(k) = 1 - e^{-\lambda\hat{A}(k)q(k)}, \tag{2.8}$$

where $\hat{A}(k)$ is the expected footprint area of the sensors.

Proof. From Equation (2.4), it is known that an event at $x_e \in \mathcal{D}$ is detected in a non-decaying sensor network if atleast one sensor in the on state is present in $\mathcal{B}_\Delta(x_e)$, where r is the radius of the sensor footprint. For a decaying network, this reasoning cannot

be applied directly. Although all sensors are initially identical, there is no reason to believe that the battery power and the footprint areas are the same throughout the network at any time $k \neq 0$ because of individual sensor activations and resulting power decays.

From stochastic geometry (e.g., [31]), the probability of any given point $x \in \mathcal{D}$ not being covered by the set $\bigcup_i \mathcal{B}_{\Delta_i}(x_i)$ in the germ-grain model is

$$P(x \text{ not covered}) = e^{-\lambda \hat{A}}, \tag{2.9}$$

where $\mathcal{B}_{\Delta_i}(x_i)$ is the grain corresponding to the germ x_i with area A_i and $\hat{A}$ is the expected area of grains $\mathcal{B}_{\Delta_i}(x_i)$ over all i. The scenario under consideration slightly differs from that in [31], since in this system, a sensor is *on* with probability $q(k)$ at time instance k. Therefore, even if $x \in \mathcal{B}_{\Delta_i}(x_i)$ for any arbitrary i, it may still not be covered since that sensor can be *off*. As a result, the probability of an event being undetected, P_u, can be obtained by updating Equation (2.4) as

$$P_u(k) = e^{-\lambda \hat{A}(k) q(k)}. \tag{2.10}$$

Finally, to conclude the proof, substitute $\hat{A}(k)$ in Equation (2.10) with Equation (2.7) and use the relationship $P_d = 1 - P_u$. □

Equation (2.8) confirms that if the probability of sensors being on is constant then the chance of an event being detected, P_d, decreases over time as the footprint area decreases.

2.1.3 Duty Cycle Scheduling For Performance Maintenance

A key requirement in many practical applications of sensor networks is to maintain a minimum satisfactory performance, which in this case is probability of event detection. To maintain the desired probability of event detection, we propose a controller that adjusts $q(k)$, the probability of a sensor being *on* at time k, which is the main goal of

this work. In other words, the objective is to find $u(k) \in [0,1]$ such that the discrete time dynamical system

$$q(k+1) = u(k) \tag{2.11}$$

can be used as a controller for scheduling the duty cycle of sensors to maintain a desired probability of event detection.

Definition 2.1.1. *The desired network performance, P_{des}, is the minimum satisfactory probability of an event being detected.*

Theorem 2.1.2. *A feedback scheduling controller of the form*

$$u(k) = \min\left\{1, \frac{1}{1-\Delta t\gamma q(k)} q(k)\right\}, \tag{2.12}$$

will guarantee that the desired network performance is maintained for the lifetime of the network from the initial conditions

$$q(0) = \frac{\ln(\frac{1}{1-P_{des}})}{\lambda c}. \tag{2.13}$$

Proof. Using the result of Lemma 2.1.1, and prescribing that the probability of an event being detected is given exactly by P_{des} yields

$$\left[\prod_{i=0}^{k-1}(1-\Delta t\gamma q(i))\right] q(k) = \frac{\ln(\frac{1}{\beta})}{\lambda c}, \tag{2.14}$$

where $\beta = 1 - P_{des}$ is the probability of an event going undetected.

Replacing the value of k in the above equation with zero, gives the initial value of q which is

$$q(0) = \frac{\ln(\frac{1}{\beta})}{\lambda c}. \tag{2.15}$$

Rearranging the terms of Equation (2.14) results in a feedback controller for $q(k)$ as

$$q(k+1) = \frac{1}{1-\Delta t\gamma q(k)} q(k). \tag{2.16}$$

Since the input of the controller is a probability, it cannot have a value greater than one. Taking this fact into account, the proposed probabilistic scheduling controller takes the form

$$u(k) = \min\left\{1, \frac{1}{1-\Delta t\gamma q(k)} q(k)\right\}.$$

As long as $u(k)$ is less than one, $q(k)$ evolves according to Equation (2.16) and the desired performance is maintained. (Note that the lifetime of the sensor network is over when $u(k)$ reaches its maximum value, which is the topic of Lemma 2.1.4.) □

Theorem 2.1.3. *The maximum achievable event detection probability in a sensor network with given spatial distribution intensity λ is $1 - e^{(-\lambda c)}$.*

Proof. Consider Equation (2.15), which yields the initial probability of a sensor being in the on state. This probability should always be in $[0, 1]$. Since $\beta \in [0, 1]$, it is guaranteed that

$$0 \leq q(0) = \frac{\ln(\frac{1}{\beta})}{\lambda c} \leq 1,$$

for all given β, λ, and c. To ensure that $q(0) \leq 1$, the condition

$$\beta \geq e^{-\lambda c}$$

must be satisfied. Hence, $P_{des} \leq 1 - e^{-\lambda c}$. □

The next interesting question is to characterize the lifetime of a network, i.e., under the proposed feedback scheduling controller, how long a network can maintain the desired event detection probability.

Definition 2.1.2. *The lifetime of the sensor network is the time beyond which the desired network performance cannot be achieved by the proposed controller.*

By solving Equation (2.16) with initial condition (2.15) results in an expression for $q(k)$

$$q(k) = \min\left\{1, \frac{-1}{\gamma k\Delta t + \frac{\lambda c}{\ln(\beta)}}\right\}, \tag{2.17}$$

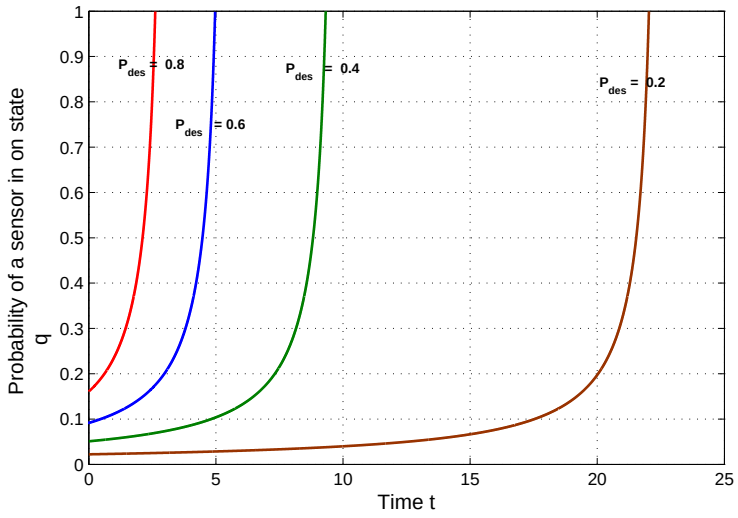


Figure 1: Evolution of the probability of a sensor being on for a given desired performance P_{des} when $\lambda = 10$, $c = 1$, and $r(0) = 2$. In each case, the lifetime of the network is achieved when $q = 1$.

which can indeed be used to explicitly compute the lifetime of the network.

Lemma 2.1.4. *The lifetime of the sensor network with desired event detection probability P_{des} is given by*

$$\frac{-1}{\gamma}\left(1 + \frac{\lambda c}{\ln(1 - P_{des})}\right).$$

Proof. At the end of the lifetime of a sensor network, all sensors should be on, i.e., $q(k_f) = 1$, where k_f denotes the final time instance. Suppose $q(k_f) \neq 1$, which suggests that one of the sensors can still be in the *o*ff state. This implies that turning this sensor on will increase the detection probability by an amount equal to the probability of an event being in its footprint. This increase in the detection probability will in turn increase the lifetime of the sensor network, which results in a contradiction since we have already assumed that the lifetime of the network has ended. Therefore, it follows that $q(k_f) = 1$ at the end of the lifetime of a network. Substituting 1 for $q(k_f)$ in Equation (2.17) yields $k_f \Delta t = \frac{-1}{\gamma}\left(1 + \frac{\lambda c}{\ln(1-P_{des})}\right)$. □

The variation of duty cycle of a sensor (the probability of sensors being on) over

time to maintain a constant event detection probability is depicted in Figure (1). For a constant event detection probability, P_{des}, the lifetime of the network is achieved when all sensors are turned on, as is shown in the proof of Lemma 2.1.4. Moreover, as P_{des} increases, the lifetime of the network decreases.

Corollary 2.1.5. *Given a desired lifetime of the sensor network, t_f, the maximum probability of event detection that can be maintained in time interval $[0, k_f]$ is $P_d = 1 - e^{\frac{-\lambda c}{1+\gamma k_f \Delta t}}$, where $t_f \in [k_f, k_f + 1)$.*

Proof. As in the proof of Lemma 2.1.4, at the end of the lifetime of a sensor network, all nodes are on to maintain the desired network performance, i.e., $q(k_f) = 1$. Therefore, substituting $q(k_f)$ in Equation (2.17) with 1 results in

$$\gamma k_f \Delta t + \frac{\lambda c}{\ln(\beta)} = -1.$$

Finally, solving the above equation for β and replacing it with $1 - P_d$ concludes the proof. □

2.1.4 Simulations

To confirm the validity of the proposed duty cycle scheduling strategy, we implement a Monte Carlo simulation of a sensor network that is deployed randomly. For this simulation, the area of interest is a 10 by 10 unit rectangular area with $A_{total} = 100$. Sensors are deployed in this area according to a spatial stationary Poisson point process with constant intensity per unit area of $\lambda = 10$, which implies that the expected number of sensors in the area of interest is $\lambda A_{total} = 1000$. The initial footprint of each sensor is set to be a closed ball of unit radius centered at the position of the sensor. Events are generated randomly at each time instant with uniform distribution throughout the area of interest. To increase the accuracy of the results, we averaged each value of P_d over 100 iterations of simulation.

To ensure that a decaying network maintains the desired performance throughout its lifetime, q is varied according to Equation (2.17) as is shown in Figure (1). The ef-

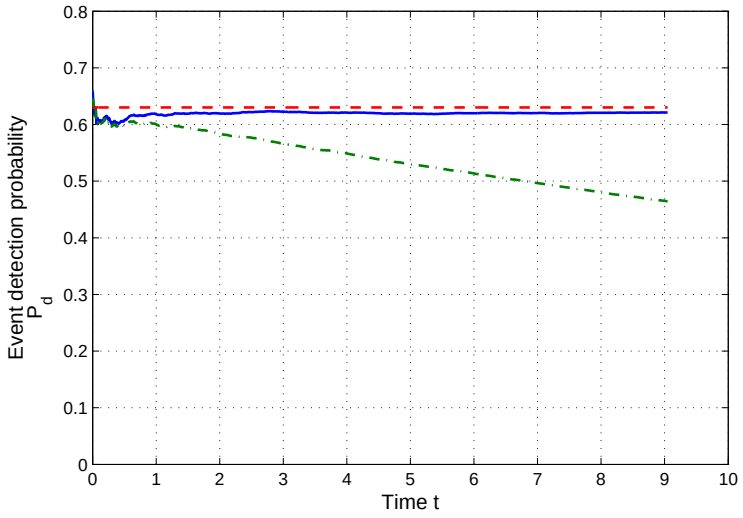


Figure 2: Event detection probability P_d vs time t for decaying networks with the scheduling scheme (solid line) and without the scheduling scheme (decaying dashed line). In this simulation, $P_{des} = 0.63$ (constant dashed line), $\lambda = 10$, $A(0) = 1$, $\gamma = 1$, and $c = 1$.

fects of varying q according to Equation (2.17) are depicted in Figure (2), in which the simulation results for a decaying network with and without the proposed scheduling scheme are presented. We set the desired network performance $P_{des} = 0.63$ (constant dashed line). First, we simulate the system without applying the proposed scheduling scheme and plot the probability of event detection (decaying dashed line). Then, we apply the proposed scheduling scheme and simulate the system again. The results for the later simulation (solid line) reveal that the probability of event detection is $P_d \approx 0.62$, which is very close to the desired performance, P_{des} indicating the validity of our scheme. Moreover, the improvement in performance is evident from comparing the plots for the two simulations (solid line vs decaying dashed line).

2.1.5 Detection Probability For Persistent Events

Until now, we have analyzed non-persistent events, i.e., events which a sensor can detect only when they occur. However, in practical scenarios, we regularly encounter events that persist for some time duration t_{ev}. It is clear that the probability of

detection of a persistent event is greater than for a non-persistent event, and this probability must increase with the increase in t_{ev}. However, this relationship between the probability of event detection and t_{ev} cannot be linear because of the shrinking footprints. In this section, we find the corresponding event detection probability for persistent events.

Let t_{ev} be the total time for which an event persists and P_{dp} be its detection probability. If $t_{ev} < \Delta t$ and $t_{ev} \subset [k, k+1)$, then $P_{dp} = P_d(k)$ because Δt is the sample time for which a sensor remains *on* or *off*. Now, consider a case in which an event persists for two time slots. Then

$$\begin{aligned} P_{dp} &= 1 - P \text{ (event is undetected in both slots k and k+1)} \\ &= 1 - P_u(k)P_u(k+1|k), \end{aligned}$$

where

$$P_u(k) = \sum_{n_1=0}^{\infty} (1-q(k))^{n_1} \frac{(\lambda \hat{A}(k))^{n_1} e^{-\lambda \hat{A}(k)}}{n_1!}.$$

In the above equation, the number of sensors that can detect the event during interval $[k, k+1)$ is n_1, which ranges from zero to infinity and cannot increase over subsequent time intervals since the footprints of all sensors decrease with time. Therefore, the number of sensors that can detect the event during interval $[k+1, k+2)$ is at maximum n_1. As a result,

$$P_u(k+1|k) = \sum_{n_2=0}^{n_1} (1-q(k+1))^{n_2} \frac{(\lambda \hat{A}(k+1))^{n_2} e^{-\lambda \hat{A}(k+1)}}{n_2!}.$$

Now, we can generalize the above equation for j time slots

$$P_{dp} = 1 - \prod_{i=0}^{j-1} P_u(k+i|k+(i-1)).$$

Thus,

$$P_{dp} = 1 - \prod_{i=0}^{j-1} \left[\sum_{n_i=0}^{n_{i-1}} (1-q(k+i))^{n_i} P_{n_i}(\hat{A}(k+i)) \right], \tag{2.18}$$

where, from Equation (2.1), $P_{n_i}(\hat{A}(k+i))$ is the probability of having n_i sensors in $\hat{A}(k+i)$.

2.1.6 Detection Probability For a Non-Boolean Sensing Model

All the results in the previous sections were derived for Boolean sensing models, i.e., an event is either detected with probability one, if it is within the footprint of a sensor that is on, or is undetected. However, the Boolean sensing model is a relatively simple model and it may not always be practically relevant. In this section we will analyze a more realistic sensing scheme in which the probability of event detection is a function of the distance from the sensor location.

Let l be the distance of an event from a sensor. Then, the probability of event detection increases as l decreases and vice versa, i.e., $P_d(k) \propto \alpha(l,k)$. Here, $\alpha(l,k)$ relates the probability of event detection to the distance of an event from a sensor, l, and the expected power level of a sensor at instance k, $\hat{\eta}(k)$. This relationship depends on the type of the sensing devices being used and can be described in various forms. In this work we define $\alpha(l,k)$ as

$$\alpha(l,k) = e^{-\frac{s}{\hat{\eta}(k)}l}. \tag{2.19}$$

According to this model, the probability of event detection decreases exponentially as a result of increase in l or decrease in $\hat{\eta}$. The third parameter in the model is s, which is a constant defining the rate of decay of the event detection probability. Even though, the model defined in Equation (2.19) is just one choice, it relates the parameters of interest in an appropriate manner and is close to the behavior of many sensors of interest.

We now design a scheduling scheme for non-Boolean sensing following the procedures from previous sections. One possible solution is to use the controller in (2.12), which was designed for Boolean sensing. This approach can be easily evaluated by including non-Boolean sensing schemes in the Monte Carlo simulation of Section 2.1.4. The results are illustrated in Figure (3), from which it is clear that the previous controller does not maintain a constant event detection probability under non-Boolean

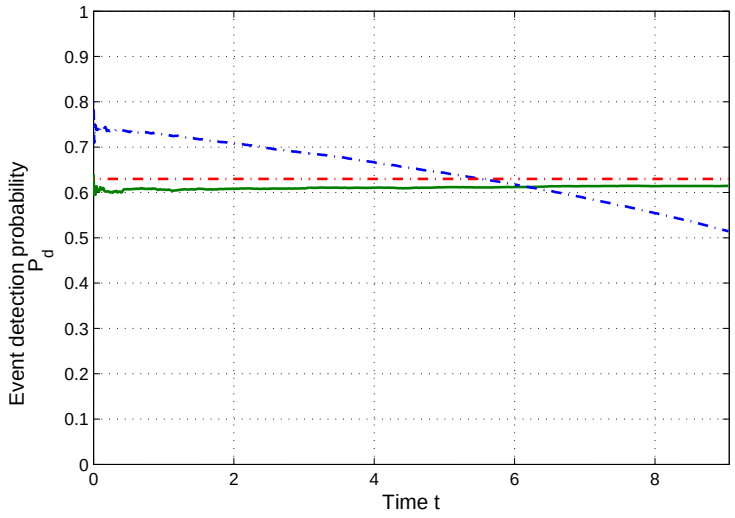


Figure 3: Comparison of Boolean sensing (solid line) and non-Boolean sensing (decaying dashed line) under the scheduling scheme of Equation (2.16). In this simulation, $P_{des} = 0.63$ (constant dashed line), $\lambda = 10$, $A(0) = 1$, $\gamma = 1$, and $s = 2$.

sensing. Using a similar concept as is proposed in [25], we develop a new scheme for maintaining the desired performance under non-Boolean sensing conditions:

Theorem 2.1.6. *For a non-Boolean sensing model with $\alpha(l,k)$ given in Equation (2.19), the probability of event detection for a non-persistent event is*

$$P_d(k) = 1 - e^{-\lambda \hat{s} \left[\prod_{i=0}^{k-1}(1-\Delta t \gamma q(i))\right]^2 q(k)} \tag{2.20}$$

where $\hat{s} = \frac{2\pi\eta(0)^2}{s^2}$.

Proof. Let x_e be the point where an event occurs. Consider a circular ring of area δA with centre at the x_e such that $\delta A = 2\pi l \delta l$ as depicted in Figure (4). Since the sensor locations have Poisson distribution, from Equation (2.1),

$$\text{Probality of } n \text{ sensors in area } \delta\text{A} = \frac{(\lambda \delta A)^n e^{-\lambda \delta A}}{n!}.$$

The probability that the event will be detected by the sensors in the ring of area δA is

$$\delta P_d(k) = 1 - \delta P_u(k),$$

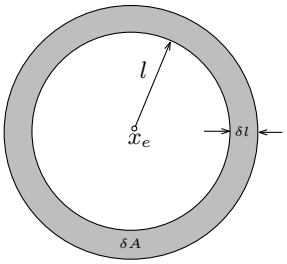


Figure 4: Illustration of non-Boolean sensing. x_e is the location of the event. δP_d is the probability of detection of this event by the sensors in area δA.

$$\begin{aligned} \delta P_u(k) &= \sum_{n=0}^{\infty} [1-\alpha(l,k)q(k)]^n \frac{(\lambda \delta A)^n e^{-\lambda \delta A}}{n!}, \\ &= e^{-(2\pi l \delta l)\lambda \alpha(l,k) q(k)}. \end{aligned}$$

To find the total probability of detection, we first divide the whole region into infinitesimal rings of area δA. Then, we find the corresponding $\delta P_u(k)$ for each ring, multiply all these probabilities, and in the limiting case when $\delta l \to 0$, we get the probability of an event not being detected. By following this strategy, we can directly compute detection probability.

$$\begin{aligned} P_u(k) &= \lim_{\delta l \to 0} \prod_{l=0}^{\infty} \delta P_u(k), \\ &= e^{-2\pi \lambda q(k) \int_{l=0}^{\infty} l \alpha(l,k) dl}. \end{aligned}$$

Replacing $\alpha(l,k)$ in the above expression with the right hand side of Equation (2.19), we get

$$P_d(k) = 1 - e^{-2\pi \lambda q(k) \int_{l=0}^{\infty} l e^{-\frac{s}{\hat{\eta}(k)} l} dl}. \tag{2.21}$$

Moreover, from Equation (2.3) we know that

$$\hat{\eta}(k) = \left[\prod_{i=0}^{k-1} (1 - \Delta t \gamma q(i)) \right] \eta(0).$$

Replacing $\hat{\eta}(k)$ in Equation (2.21) with the right hand side of the above equation

and using the identity

$$\int_0^\infty re^{-mr}dr = \frac{1}{m^2},$$

concludes the proof. □

By following the same technique as in Section 2.1.3, we can now design a controller that maintains the desired network performance, P_{des}, for non-Boolean sensing. In fact, from Equation (2.20), we know that

$$\left[\prod_{i=0}^{k-1}(1-\Delta t\gamma q(i))\right]^2 q(k) = \frac{\ln(\frac{1}{\beta})}{\lambda \hat{s}}. \tag{2.22}$$

From the above expression, we can find the initial value of q as

$$q(0) = \frac{\ln(\frac{1}{\beta})}{\lambda \hat{s}}.$$

Rearranging the terms in Equation (2.22) yields the dynamics of $q(k)$ as

$$q(k+1) = \frac{1}{(1-\Delta t\gamma q(k))^2} q(k). \tag{2.23}$$

Hence, we have a non-linear control law for duty cycle scheduling that maintains the desired performance measure throughout the lifetime. To verify the validity of this scheme, we again run a series of Monte Carlo simulations with the same parameters as in Section 2.1.4, and the result for non-Boolean sensing is demonstrated in Figure (5). From this plot, we can see that the proposed scheme maintains the desired performance throughout the lifetime of the network. The lifetime of the network in this case also depends on the parameter s, which in this example was set to $s = 2$.

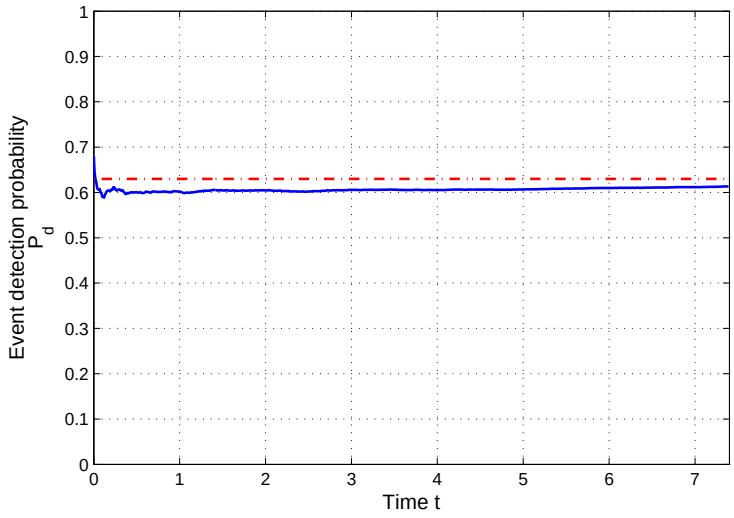


Figure 5: Event detection probability P_d vs time t for a decaying network and non-Boolean sensing (solid line) with given $P_{des} = 0.63$ (constant dashed line). The values of the parameters are $s = 2$, $\gamma = 1$, $\eta(0) = 1$.

2.2 Generalized System Model

In Section 2.1, we focused on the problem of the effects of decrease in available power on network performance. However, the analysis and design in Section 2.1 was limited to a scenario in which the available power was decreasing according to an exponential decay law, and the sensing footprint of each sensor was a circular disk centered at the location of the sensor. This footprint model is popular in wireless sensor networks community because it makes the analysis and design of scheduling schemes relatively simple. In this section we are extending the previous results for a generalized system model and propose power-aware controllers to maintain desired performance. *The key novelty is that we do not assume any particular power-decay model in the design and analysis of our proposed scheme. Moreover, we show that our results are valid not only for a circular sensing model, but for any model as long as the footprint is compact.*

In addition to the generalized footprint and power-decay models, the proposed scheme can handle increase in available power, which is important because of the two

reasons. First, power delivered by batteries is affected by atmospheric temperature implying that it may increase with the increase in temperature [55]. Secondly, with the advancements in energy harvesting technologies, it is becoming common that sensing nodes are equipped with small devices like solar panels to harvest energy ([66], [65], and [68]). In such scenarios, the available power may increase with time and if each node does not incorporate this increase in power, the system will consume more power than is required to maintain the desired performance. Our proposed scheme takes into account this increase in available power and adjusts the control parameter to reduce power consumption. Furthermore, the scheme we are proposing in this section is robust to node failures. The scheduling controller adjusts the control parameter to minimize the effects of node failures and maintain desired performance.

2.2.1 System Description

In this section, we consider the same setup as in Section 2.1 in which a large number of sensors were randomly deployed for monitoring a domain $\mathcal{D} \subset \mathbb{R}^2$. Again, this deployment is modeled as a stationary, homogeneous Poisson point process with intensity λ. However, there is a fundamental difference in the model of the sensing agents that are considered in this section. Each sensing agent located at $x_i \in \mathbb{R}^2$

- is a stationary device whose sensing range model is a function of available power,
- is battery powered where the available power may increase or decrease with time, and
- has a compact set as its sensing region called the footprint of a sensor, denoted by $\mathcal{F}_{x_i}$.

Notice that footprint of a sensor is no longer assumed to be circular. The only restriction that needs to be satisfied is that the footprint must be compact, which makes this system completely generalized in terms of footprint models. Next, we

assume that all the sensors initially have the same footprint, i.e., for all i, we have $\mathcal{F}_{x_i}(0) = \mathcal{F}$ and $A_i(0) = A$, where $\mathcal{F}_{x_i}(0)$ and $A_i(0)$ are the footprint and footprint area of the i^{th} sensor located at x_i at time 0 respectively.

To conserve power, sensors are switched between on and off states. To simplify the analysis, we assume that power is consumed only when a sensor is in the on state. However, the results derived in the later sections can easily be extended for a more generalized power model. We assume that sensors switch between on and off states at discrete time instants $k\Delta t$, where Δt is the length of an interval in which a sensor maintains its state.[2] At each switching time, a sensor decides to be in the on state with probability $q_i(k)$ and in the off state with probability $1 - q_i(k)$.

2.2.2 Power-aware Scheduling

Let $x_e \in \mathcal{D}$ be the location of a non-persistent event and we want to find the probability of detection for this event.[3]. To find the event detection probability, P_d, we start with the probability of an event not being detected, P_u. Event at x_e is not detected if either of the following two conditions is satisfied.

C1) x_e does not belong to the footprint of any sensor.

C2) All the sensors such that x_e belongs to their footprints are in the off state. Using results from [30] and [26], we can write P_u as

$$P_u = \sum_{n=0}^{\infty} \frac{(\lambda A)^n e^{-\lambda A}}{n!}(1-q)^n,$$

where A is the area of the footprint of a sensor, and q is the probability of a sensor being on, and both A and q are the same for all the sensors in the above expression. In this expression, $n = 0$ corresponds to Condition (C1), while the summation of terms having $n \neq 0$ corresponds to Condition (C2). Therefore, the event detection

[2]For brevity of notation, we will denote switching time by k instead of $k\Delta t$ in the rest of this work.

[3]A non-persistent event is one that does not leave a mark in the environment and can only be detected at the time of its occurrence

probability is

$$P_d = 1 - e^{-\lambda A q}. \tag{2.24}$$

To design power-aware controllers, we first need to establish a relationship between power that is available to a sensor and the desired performance criterion. Here, event detection probability is our desired performance criterion because for a randomly deployed network in which sensors randomly switch between on and off states, event detection probability provides a good measure of the expected coverage achieved by the network, which makes it a natural performance criterion. From Equation (2.24), we know that our performance criterion, i.e., detection probability, depends on the area of the footprint of a sensor, which implies that our desired relationship between available power and performance criterion is

$$A_i(k) = \mathcal{G}(\eta_i(k)), \tag{2.25}$$

where $A_i(k)$ is the footprint area of the i^{th} sensor at time k, $\eta_i(k)$ is the available power of the i^{th} sensor at time k, and $\mathcal{G}$ is a non-decreasing function of available power, which relates transmission power with area of the footprint. This function depends on the type of the sensing devices used and can be found from the device specifications provided by the manufacturer. For instance, in Section 2.1, we selected sensors with circular footprints and for such devices

$$A_i(k) = \alpha \eta_i(k),$$

where α is a proportionality constant. Equation (2.25) provides an explicit relationship between the performance of an individual sensor and its available power. However, we want to extend this relationship to the performance of the entire network, i.e., event detection probability.

To find the relationship between available power and the performance of the entire network when available power is varying with time, we start by modeling the

variations in available power of any sensor by a difference equation.

$$\eta_i(k+1) = \eta_i(k) + \Delta t f(k, \eta_i(k)),$$

where $f : \mathbb{N} \times \mathbb{R}_+ \to \mathbb{R}$ is a function of k and $\eta_i(k)$. In the above model, power variations are modeled by a function f which can be positive or negative, where $f \in \mathbb{R}_+$ implies power gain which may be due to temperature variations or energy harvesting where as $f \in \mathbb{R}_-$ implies power decay due to deterioration in battery performance. We define a switching function for each sensor

$$\sigma_i(k) = \begin{cases} 1 & \text{if sensor } i \text{ is on at time } k. \\ 0 & \text{if sensor } i \text{ is off at time } k. \end{cases}$$

Since we have assumed that power is only consumed when a sensor is *on*, we can modify the power model as

$$\eta_i(k+1) = \eta_i(k) + \sigma_i(k)\Delta t f(k, \eta_i(k)). \tag{2.26}$$

Next we propose a power-aware scheduling scheme to maintain desired event detection probability. In order to decide whether to be in the on or off state at some time instant k, each sensor follows Scheme 1.

Scheme 1. Sensor Scheduling Scheme

Given: *Desired event detection probability,* P_{des}.

Sensor deployment intensity, λ.

$\mathcal{G}(.) : \mathbb{R} \to \mathbb{R}$.

At Step 0:

- *Initializes its probability of being on*

$$q(0) = \frac{\ln\left(\frac{1}{1-P_{des}}\right)}{\lambda A(0)},$$

where $A(0) = \mathcal{G}(\eta(0))$.

- *Selects a number* m_i *such that* $m_i \sim$ *unif* $[0, 1]$.

- *Turns on if* $m_i < q(0)$.

At Step k:

- *Selects a number* m_i *such that* $m_i \sim$ *unif* $[0, 1]$.

  if $m_i < q_i(k)$

  - *Decides to be in the on state in interval* $[k, k+1]$.
  - *At time* $k+1$, *measures its current power level* $\eta_i(k+1)$ *and computes*

$$q_i(k+1) = \frac{A(0)}{A_i(k+1)} q(0),$$

  where $A_i(k+1) = \mathcal{G}(\eta_i(k+1))$.

  else

  - *Decides to be in the off state in interval* $[k, k+1]$.

$$q_i(k+1) = q_i(k).$$

Theorem 2.2.1. *If each sensor makes its switching decisions according to Scheme 1, i.e., updates its probability of being on according to the control law*

$$q_i(k) = \frac{A(0)}{A_i(k)} q(0), \tag{2.27}$$

with initial condition

$$q(0) = \frac{\ln\left(\frac{1}{1-P_{des}}\right)}{\lambda A(0)}, \tag{2.28}$$

where P_{des} is the desired event detection probability, then the network maintains P_{des}.

To prove this theorem, we first need to prove the following lemma.

Lemma 2.2.2. *The event detection probability of a sensor network consisting of sensing devices whose footprints are time varying and whose probability of being on are function of footprint area is*

$$P_d(k) = 1 - e^{-\lambda \mathbb{A}vg(\bar{A}(k)\bar{q}(k))}, \tag{2.29}$$

where $\bar{A}(k) = \{A_1(k), A_2(k), \ldots, A_M(k)\}$ and $\bar{q}(k) = \{q_1(k), q_2(k), \ldots, q_M(k)\}$. Moreover,

$$\mathbb{A}vg(\bar{A}(k)\bar{q}(k)) = \frac{1}{M} \sum_{i=1}^{M} A_i(k) q_i(k), \tag{2.30}$$

where $M \sim Poi(\lambda A_{dom})$ is the number of sensors in the network and A_{dom} is the area of $\mathcal{D}$.

Proof. To prove this lemma, we will use standard results from stochastic geometry [30]. At time $k = 0$, we assume that all the sensors are identical, i.e., all of them have the same available power $\eta_i(0) = \eta^0$, footprint area $A_i(0) = A^0$, and on probability $q_i(0) = q^0$. Then from Equation (2.24), the probability of an event not being detected for this network is

$$P_u(0) = e^{-\lambda A^0 q^0}.$$

At time $k = 1$, we have two classes of sensors. All the sensors that remained off during the time interval $[0, \Delta t]$, i.e., all i such that $\sigma_i(0) = 0$, belong to the first class and for these sensors $\eta_i(1) = \eta^0$, $A_i(1) = A^0$, and $q_i(1) = q^0$. The sensors that turned on during the interval $[0, \Delta t]$, i.e., all i such that $\sigma_i(0) = 1$, belong to the second class and for these sensors $\eta_i(1) = \eta^1$, $A_i(1) = A^1$, and $q_i(1) = q^1$, where η^1 is governed by the power variations model (2.26) and A^1 is computed from Equation (2.25). Therefore, at time $k = 1$, sensor deployment can be modeled as a superposition of two Poisson processes Φ^0 and Φ^1 with intensities $\delta^0\lambda$ and $\delta^1\lambda$, where

$$\delta^l(k) = \frac{\text{number of sensors with footprint area } A^l(k)}{\text{total number of sensors}},$$

$l \in \{1, 2\}$ and $\sum_l \delta^l = 1$. For this case, event is not detected if it is not detected by any sensor belonging to either of the two classes and this probability is the product of the probabilities

$$P_u(1) = \prod_{l=1}^{2} P_u^l = e^{-\delta^0\lambda A^0 q^0} e^{-\delta^1\lambda A^1 q^1}.$$

At time $k = 2$, we have three classes of sensors corresponding to the switching combinations $[(\sigma_i(0), \sigma_i(1)) : (0, 0)$, $(1, 1)$, and $\{(1, 0), (0, 1)\}]$, and using the same superposition argument as before, the probability of event not being detected is

$$P_u(2) = e^{-\lambda \sum_{l=1}^{3} (\delta^l A^l q^l)}.$$

Thus, generalizing this argument for any time instant k, the probability of an event being undetected is

$$P_u(k) = \prod_l P_u^l = e^{-\lambda \left(\sum_l \delta^l(k) A^l(k) q^l(k) \right)}. \tag{2.31}$$

It is important to note that $\sum_l \delta^l(k) A^l(k) q^l(k)$ is the weighted average, so the above expression can be written as

$$\sum_l \delta^l(k) A^l(k) q^l(k) = \frac{1}{M} \sum_{i=1}^{M} A_i(k) q_i(k).$$

Finally, updating Equation (2.31) according to above expression and using the fact that $P_d(k) = 1 - P_u(k)$ concludes the proof of Lemma 2.2.2. □

To prove Theorem 2.2.1 and to verify that Scheme 1 maintains P_{des}, replace $q_i(k)$ in Equation (2.30) with $q_i(k)$ from Equation (2.28) for $k = 0$ and Equation (2.27) for all $k > 0$, which yields

$$\mathbb{A}\text{vg}(\bar{A}(k)\bar{q}(k)) = A(0)q(0) = \frac{\ln\left(\frac{1}{1-P_{des}}\right)}{\lambda}.$$

Replacing $\mathbb{A}\text{vg}(\bar{A}(k)\bar{q}(k))$ in Equation (2.29) with the above expression shows that $P_d(k) = P_{des}$, which concludes the proof of Theorem 2.2.1.

Equation (2.27) is a power-aware controller that maintains desired event detection probability regardless of the variations in available power. *The time for which the proposed controller can maintain P_{des} is the lifetime time of the network.* It is important to note that unlike Section 2.1, this controller is independent of the law that dictates variations in available power, since the only quantity a sensor needs to compute the control parameter, $q_i(k)$, is its available power at time instant k, which can be computed.

The fact that each sensor requires its current power level only without any knowledge of the power-decay model in (2.27) makes this controller attractive for practical implementation even on the sensing devices that have low computational capabilities. One key observation about the proposed controller is that it depends only on the area of the footprint and the above scheme is valid as long as the footprint is compact [31]. There is absolutely no dependence on the shape of the footprint. *Therefore, unlike most of the existing literature on wireless sensor networks that deals with circular footprints only, the results of this work are valid for any sensor footprint provided it is compact.*

(a) Circular Footprint (b) Sector Footprint

Figure 6: Depicted are the two footprint models that are simulated in this work. In Figure 6(a), the footprint of a sensor is a circular region of radius r, where as in Figure 6(b), the footprint is union of four sectors.

2.2.3 Simulation

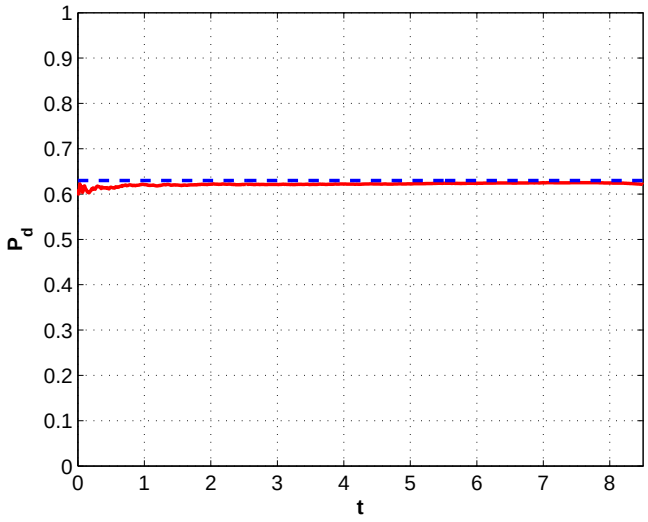
To verify the performance of Scheme 1, we performed Monte Carlo simulations in Matlab. We considered a rectangular domain of dimensions $[30 \times 30]$ in which sensors were deployed randomly according to a Poisson point process of intensity $\lambda = 10$. Each sensor had an initial footprint area $A(0) = 1$ and initial power level $\eta(0) = 1$. To compare the performance of this scheme with the results in Section 2.1, we first considered an exponential power decay law for each sensor given by

$$\eta_i(k) = \eta_i(k-1) - \gamma \Delta t \eta_i(k-1), \tag{2.32}$$

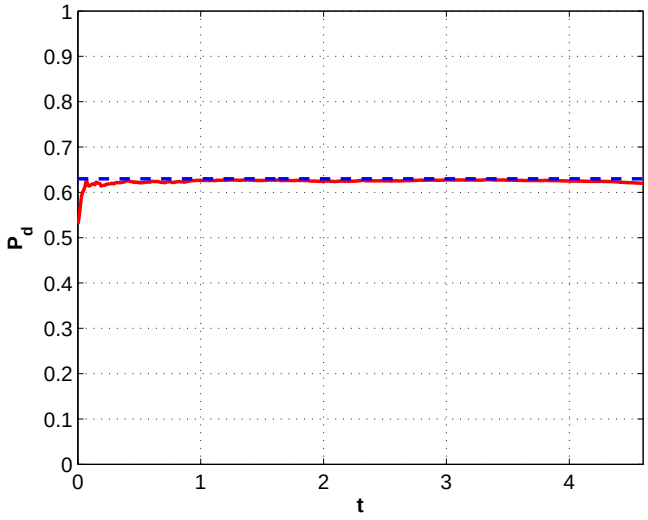
where γ is decay constant and for this simulation $\gamma = 1$. We also simulated a linear power decay law

$$\eta_i(k) = \eta_i(k-1) - \gamma \Delta t. \tag{2.33}$$

We want to point out again that in Section 2.1, each sensor had a complete knowledge of its power-decay law and it used this information to update its control parameter. However, in this scheme, no sensor has any information about the decay law, which makes this scheme more attractive for practical implementation. In addition to using different power decay models, we also simulated multiple footprint models to validate our claim that this scheme is valid for any footprint model as long as the footprint of each sensor is compact and the sensors are deployed randomly. We start with circular



(a) Circular footprint and exponential decay

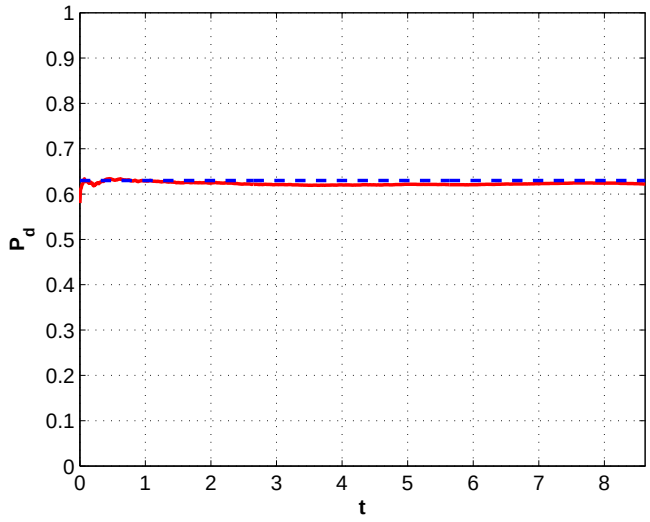


(b) Circular footprint and linear decay

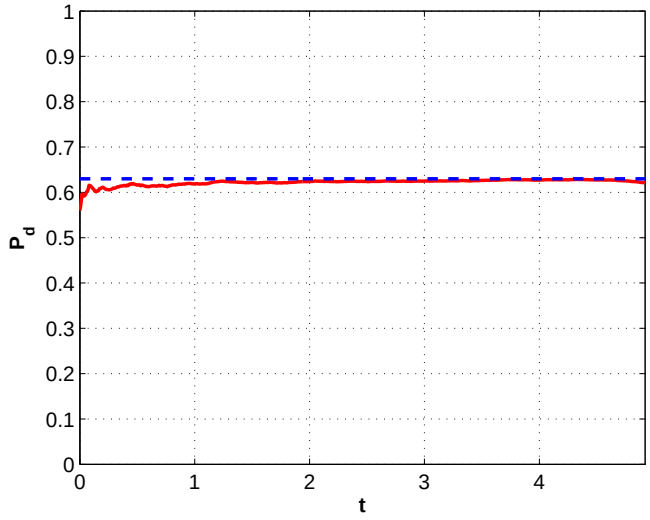
Figure 7: Event detection probability P_d vs time t for a sensor network with Scheme 1 (solid line) and given $P_{des} = 0.63$ (constant dashed line) for circular footprint. Here $\lambda = 10$, $A(0) = 1$, and $\gamma = 1$.

footprint model as shown in Figure 6(a) to compare it with our previous work. We also simulated a more practical footprint model that is depicted in Figure 6(b). In this model, each agent is equipped with four sensors that can sense a sector of radius r_s with central angle θ_s, where $s \in \{1, 2, 3, 4\}$. For each sensor

$$r_s^2(k) = \alpha_s \eta_s(k),$$



(a) Sector footprint and exponential decay



(b) Sector footprint and linear decay

Figure 8: Event detection probability P_d vs time t for a sensor network with Scheme 1 (solid line) and given $P_{des} = 0.63$ (constant dashed line) for sector footprint. Here $\lambda = 10$, $A(0) = 1$, and $\gamma = 1$.

where α_s is the constant of proportionality for the s^{th} sensor. The total power consumed is the sum of power consumed by all the sensors.

We ran the simulation 100 times with the settings described above and the event detection probability averaged over all the runs of simulations under the proposed scheme is demonstrated in Figures (7(a), 7(b), 8(a), and 8(b)) corresponding to the

two power decay laws and the footprint models. In each figure, the dotted line corresponds to the desired detection probability $P_{des} = 0.63$. The figures show that after the initial settling phase, the system maintains desired event detection probability for all the cases.

2.2.4 Decentralized Scheduling Scheme

In the previous section, we proposed a probabilistic power-aware scheduling scheme that can maintain desired event detection probability. One important aspect of this scheme was that each sensor was making its switching decision completely independent of all the other sensors. However, there was one global parameter that was used by each sensor in making its switching decision and that parameter was λ, i.e., expected number of sensors per unit area. In general, if λ is initially unknown, it can only be estimated in a centralized manner. However, if we have a sufficiently dense deployment of sensors such that $\bigcup_{i=1}^{M} \mathcal{F}_{x_i} = \mathcal{D}$, which is normally the case in the networks we are considering [1], then we can approximate λ in a decentralized manner. Thus, in this section, we will update Scheme 1 to remove the dependence on λ, which will make the updated scheme, Scheme 2, decentralized in true sense as no global knowledge will be used to make switching decisions.

In Scheme 2, λ_i is the estimate of λ generated by sensor i and $\mathcal{N}_i(0)$ is the set of sensors that belong to the footprint of sensor i at time 0. In this scheme, each sensor uses its own estimate of λ to compute $q_i(0)$. We know that the detection probability depends on

$$\mathbb{A}\text{vg}(\bar{A}(k)\bar{q}(k)) = \frac{1}{M}\sum_{i=1}^{M} A_i(k)q_i(k)$$

If we replace $q_i(k)$ in the above expression with Equation (2.27) for $k > 0$ and with

Scheme 2. Updated Sensor Scheduling Scheme

Given: *Desired event detection probability (P_{des})*

$\mathcal{G}(.) : \mathbb{R} \to \mathbb{R}$.

$n_{hops} \in \mathbb{N}$.

At Step 0:

- *Senses its footprint, $\mathcal{F}_{x_i}(0)$, having area $A_i(0)$ and computes λ_i.*

$$\lambda_i = \frac{Number\ of\ sensors\ in\ \mathcal{F}_{x_i}(0)}{A_i(0)}.$$

- **for** $l = 1 : n_{hops}$

  Shares its estimate λ_i with its neighbors

  and update its estimate

$$\lambda_i(l) = \lambda_i(l-1) + \Delta l \left[\sum_{j \in \mathcal{N}_i(0)} (\lambda_j(l-1) - \lambda_i(l-1)) \right] \tag{2.34}$$

  where $\mathcal{N}_i(0) = \{j\ s.t.\ x_j \in \mathcal{F}_{x_i}(0)\}$, and Δl is the step size.

  end

- *Initializes its probability of being on*

$$q_i(0) = \frac{ln\left(\frac{1}{1-P_{des}}\right)}{\lambda_i A(0)}, \tag{2.35}$$

  where $A_i(0) = \mathcal{G}(\eta_i(0))$.

- *Selects a number m_i such that $m_i \sim$ unif $[0, 1]$.*

- *Turns on if $m_i < q_i(0)$.*

At Step k:

- *Follow Scheme 1*

Equation (2.35) for $k = 0$, then

$$\mathbb{A}\text{vg}(\bar{A}(k)\bar{q}(k)) = \mathbb{A}\text{vg}(\bar{h}(.)),$$

where

$$h(\lambda_i) = \frac{\ln\left(\frac{1}{1-P_{des}}\right)}{\lambda_i}$$

is a convex function of λ_i, and $\bar{h}(.) = \{h(\lambda_1), h(\lambda_2), \ldots, h(\lambda_M)\}$ for all $i \in \{1, 2, \ldots, M\}$. If λ is initially known, i.e., $\lambda_i = \lambda$ for all i, then the detection probability depends on $h(\lambda) = h(\mathbb{A}\text{vg}(\bar{\lambda}))$, whereas if λ is unknown then the detection probability depends on $\mathbb{A}\text{vg}(\bar{h}(.))$. Here $\bar{\lambda} = \{\lambda_1, \lambda_2, \ldots, \lambda_M\}$. Since $h(.)$ is a convex function of λ_i, from Jensen's inequality

$$h(\mathbb{A}\text{vg}(\bar{\lambda})) \leq \mathbb{A}\text{vg}(\bar{h}(.)).$$

The effect of this inequality can be observed from Figure (9), which demonstrates event detection probability of the same network (comprising of sensors with circular footprint and exponential power decay) as simulated in Section 2.2.3 but using Scheme 2. As shown in Figure (9), in the beginning, event detection probability is higher than the desired value (dotted line), which seems good. However, to maintain a higher detection probability than P_{des}, more sensors have to remain on consuming more power than is required to achieve objective. Consequently, the detection probability falls below the desired level before the lifetime is over. This is the price we have to pay for making the system decentralized.

In order to quantify the performance loss because of the approximation of λ, we ran Monte Carlo simulations under the same setup as described in Section 2.2.3 100 times. As long as the detection probability remains greater than or equal to P_{des} our performance criterion is met. However, when it falls below P_{des}, we have performance loss, which is shown in Figure (4) . For each run of the simulation, Figure 10(a) shows the percentage average performance loss while Figure 10(b) shows the percentage maximum performance loss and the solid line in both the figures represent average of

the 100 values. These figures show that the maximum performance loss is on average 11% where as the average performance loss is 4.5 % on average.

If the performance loss shown in Figures (10(a) and 10(b)) are acceptable, then we can use Scheme 2 as described above in which there is no communication involved in making switching decisions even at the local level. However, to reduce performance decay, we can allow neighboring sensors to communicate with each other to improve their estimate λ_i by following the update law (2.34). This equation is the discrete time version of consensus equation, which will drive the estimates of all the sensors towards their initial average ([15] and [49]), i.e., for all i,

$$\lambda_i \to \mathbb{A}\text{vg}(\bar{\lambda}) = \frac{1}{M}\sum_{i=1}^{M}\lambda_i(0) \text{ as } n_{hops} \to \infty,$$

where n_{hops} is the number of iterations of the update law (2.34). Figure (11) shows the performance of the system under Scheme 2 with $n_{hops} = 100$, and it can be observed that the network maintains a constant performance that is very close to the desired value. One issue that we want to point out here is that $\mathbb{A}\text{vg}(\bar{\lambda})$ will always be a little off than actual λ. The reason is that according to [30], the true estimate of λ is

$$\lambda = \frac{\text{Total number of deployed sensors}}{\text{Total area of } \mathcal{D}}$$

However, in the case of randomly deployed sensor networks, the footprints of the sensors can overlap with each other and it is difficult to get an estimate of this overlap in practical systems. The consequences of these overlapping footprints are twofold. Firstly, with non-zero probability a sensor can be counted multiple times, which makes the numerator in the above relation higher than the actual value. Secondly, because of neglecting the overlap of the footprints, the total area of the domain $\mathcal{D}$, i.e., the denominator in the above relations, becomes larger than its actual value. These consequences of overlapping footprints results in $\mathbb{A}\text{vg}(\bar{\lambda})$ a little off than λ. However, this effect will not cause sever problem since the size of footprint is typically extremely

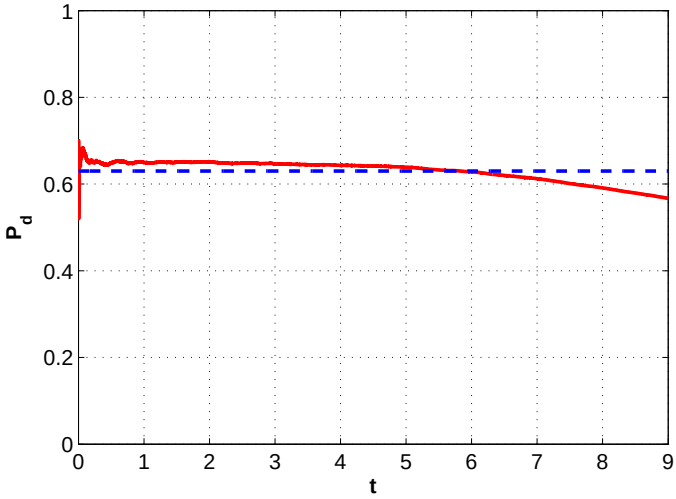
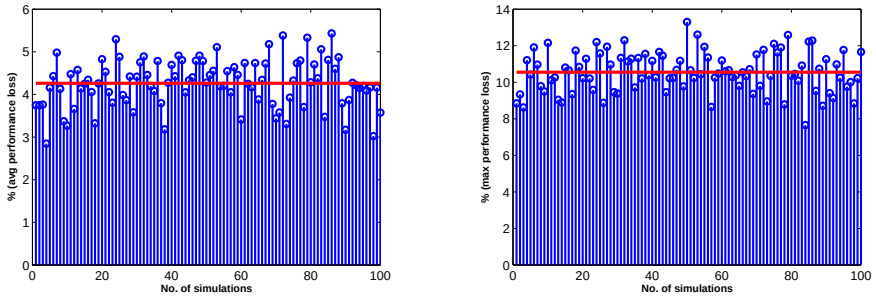


Figure 9: Event detection probability P_d vs time t (solid line) for a sensor network under Scheme 2 with given $P_{des} = 0.63$ (constant dashed line) . The sensors have circular footprint and exponential power decay. Here $n_{hops} = 0$, $\lambda = 10$, $A(0) = 1$, and $\gamma = 1$.

small as compared to the total area of the domain $\mathcal{D}$. For instance, in this simulation $\mathbb{A}\text{vg}(\bar{\lambda}) = 10.8$ instead of the actual value $\lambda = 10$.



(a) % Average Performance Loss (b) % Maximum Performance Loss

Figure 10: Percentage performance loss under Scheme 2 for 100 iterations of simulations

2.2.5 Robust and Decentralized Scheduling Scheme

Until now, we have assumed that all the sensors in a network remain operational throughout the lifetime of the network. However, we are dealing with networks comprising of cheap and low quality devices that are dropped in the region of interest

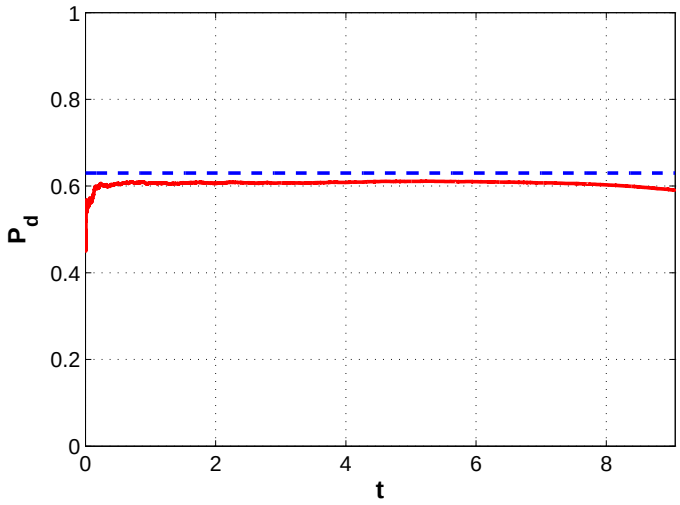


Figure 11: Event detection probability P_d vs time t for a sensor network (solid line) with given $P_{des} = 0.63$ (constant dashed line) and unknown intensity under Scheme 2. In this simulation sensors have circular footprint and exponential model. Here $\lambda = 10$, $A(0) = 1$, $n_{hops} = 100$, and $\gamma = 1$.

randomly. Therefore, it is possible that some of the sensors stop working unexpectedly before their operational lifetime is over. This will obviously have a negative impact on the performance of the network since we have not incorporated failures of the devices in the design of our scheduling scheme. There is also a possibility that at some time, we want to deploy more sensors to improve the performance of the network and reduce the workload of existing ones. Again, this is a scenario that our proposed schemes cannot handle. Therefore, we propose Scheme 3 to detect the failure of existing devices or addition of new devices and adjust the control parameter in a manner so that the desired performance criterion is maintained while reducing power consumption.

In this scheme, each sensor initializes its estimate of deployment intensity, λ_i, as in Scheme 2 and based on this estimate computes its initial probability of being on, i.e., $q_i(0)$. Then, at each step it follows the same control law to update its control parameter, $q_i(k)$ as in Equation (2.27). However, after every m steps, each sensor updates its estimate λ_i and uses this updated estimate to compute its probability

Scheme 3. Robust Sensor Scheduling Scheme

Given: *Desired event detection probability (P_{des})*

$\mathcal{G} : \mathbb{R} \to \mathbb{R}$.

At Step 0:

- *Follow Scheme 2.*

At Step k:

if mod *(k,m) = 0*

- *Senses its footprint $\mathcal{F}_{x_i}(k)$ having area $A_i(k)$ and computes λ_i.*

$$\lambda_i = \frac{\textit{Number of sensors in footprint } \mathcal{F}_{x_i}(k)}{A_i(k)}.$$

- *Updates its probability of being on*

$$q_i(k) = \frac{ln\left(\frac{1}{1-P_{des}}\right)}{\lambda_i A_i(k)} \tag{2.36}$$

- *Set $q_i(0) = q_i(k)$ and $\eta_i(0) = \eta_i(k)$.*

- *Selects a number m_i such that $m_i \sim$ unif $[0, 1]$.*

- *Turns on if $m_i < q_i(k)$.*

else

- *Follow Scheme 1*

of being *on*. This parameter m can either by given a priori based on the quality of devices or can be selected randomly by each device itself. By updating λ_i, each sensor takes into account the failure or addition of nodes in its sensing region and uses this information to adjust its probability of being on. In the case of node failures, λ_i will decrease which will force the sensor to increase $q_i(k)$ to maintain desired detection

probability. Similarly, in the case of the addition of new nodes, λ_i will increase which will allow the sensor to decrease $q_i(k)$ and reduce energy consumption while maintaining desired detection probability. Thus, Scheme 3 is robust to node failures and tries to maintain desired detection probability as long as a minimum number of sensors is available.

The performance of a network simulated under the same settings as in Section 2.2.3 with circular footprint and exponential power decay except for $\gamma = 0.4$, is shown in Figure (12). The detection probability for this network is shown in Figure (12) as decaying solid line and it is apparent that the network cannot maintain P_{des}. To improve network performance, we apply Scheme 3 with $m = 1$ and $n_{hops} = 0$, where the value of the parameter m dictates how often each sensor in the network updates its estimate of intensity, λ_i. The value of m equal to one implies that sensors update their estimate at each decision time while $n_{hops} = 0$ means that sensors do not communicate with their neighbors to improve their estimate. By applying Scheme 3, the performance of the network clearly improves as shown in the dotted decaying plot. However, the network is still unable to maintain P_{des} throughout its lifetime because of two reasons. Firstly, as discussed for Scheme 2, when sensors estimate λ, there is always going to be a performance loss. Secondly, in this case, sensor are failing constantly and after some time it becomes simply impossible for the network to maintain P_{des} because of insufficient sensor deployment.

2.3 Conclusions

In this chapter, we presented scheduling schemes for the duty cycle of dynamic sensor networks comprising of sensors whose footprints shrink with decrease in available power. In particular, we examined networks in which sensors were deployed randomly according to a stationary spatial Poisson point process. Initially, we simplified the problem by assuming an exponential power decay law and a circular footprint

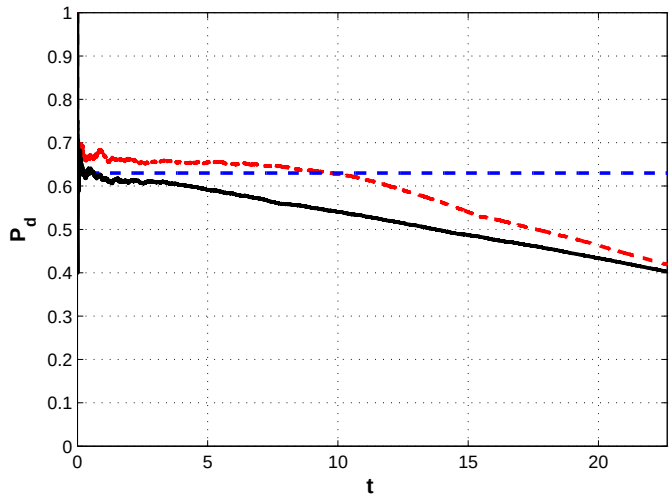


Figure 12: Event detection probability P_d vs time t for a sensor network with failing nodes and no compensation (solid decaying line), sensor network with failing nodes and Scheme 3 (dotted decaying line) with given $P_{des} = 0.63$ (constant dashed line). Here $\lambda = 10$, $A(0) = 1$, $\alpha = 1$, $\gamma = 0.4$, $m = 1$, and $n_{hops} = 0$

model. To establish the relationship between the desired performance criterion and the lifetime of a network, we analyzed both persistent and non-persistent events and proposed scheduling schemes for both of these scenarios to maintain desired network performance. Moreover, we examined two sensing models, Boolean and non-Boolean, to incorporate various physical sensing characteristics. Then we presented a similar power-aware controller but for a generalized system, which was independent of the of the power variation law as well as the shape of the sensor footprint. We proposed a decentralized scheme in which sensors estimated the deployment intensity based on their own observation of the environment and the observation of their immediate neighbors. We also proposed adaptive and robust versions of the proposed scheme, which can estimate the deployment intensity and can update the control law in the case of the failure of deployed nodes or the addition of extra nodes. The results were validated by Monte Carlo simulations of the proposed controllers, through which we showed that the proposed schemes maintained the desired performance throughout the lifetime of the network.

CHAPTER III

SLEEP SCHEDULING OF WIRELESS SENSOR NETWORKS USING HARD-CORE POINT PROCESSES

In this chapter, we deal with an other aspect of power-awareness that is related with the efficient utilization of available energy resources through intelligent scheduling schemes. This aspect of power-awareness is popular in wireless sensor networks community and a plethora of scheduling schemes have been proposed in the literature. A common technique to conserve energy is to add redundancy by increasing the intensity of sensor deployment such that only a subset of sensors is sufficient to maintain the required coverage level [27]. Thus, it is not necessary for all the sensors to be on in unison and we can switch sensors off in order to conserve energy. However, it is important to schedule the sensor switching intelligently as critical events can be missed if all the responsible sensors are off. There is a long list of such scheduling schemes that are available in the literature, e.g., ([19, 21, 27, 26, 36, 20, 22, 24], and [80]), just to name a few.

Most of the scheduling schemes that have been proposed can be divided into two broad classes: random switching schemes and coordinated switching schemes (e.g., [27] and [24]). It is important to point out that comparing random and coordinated switching schemes is rather difficult. In random switching, each sensor decides to switch its state randomly according to some probabilistic rule regardless of the states of the other sensors. The clocks of the sensors does not need to be synchronized and there is no communication cost involved in scheduling, which makes these schemes efficient in terms of power consumption. However, random switching schemes cannot guarantee complete coverage unless all the sensors are on. In contrast, in coordinated

switching schemes, sensors either communicate with their neighbors or acquire exact information about the state and location of their neighbors in order to decide whether to switch on or off. These schemes incur additional communication costs because sensors have to communicate with each other to make switching decisions. However, coordinated scheduling permits the designers to generate various sensor switching patterns depending on the applications. Moreover, these schemes can generally ensure complete coverage of the area of interest (e.g., [26] and [10]). In conclusion, both schemes have their merits and demerits and the decision has to be made based on the application.

In this work we propose a novel sleep scheduling scheme that is a compromise between coordinated and random switching. The proposed scheme is coordinated since sensors communicate with their neighbors for making switching decisions. In order to introduce coordination among the neighboring sensors, we use the concept of hard-core point processes from stochastic geometry, which are inhibition processes that maintain a certain minimum distance, d, between the constituent points [30]. However, information that is communicated between sensors for coordination consists of randomly generated numbers, which introduces randomness in the switching decisions. By communicating random numbers between the neighboring sensors, we can only achieve partial coverage. On the other hand, this coordination improves the coverage as compared to random switching with little communication overhead. We analyze the proposed scheme and derive an expression for the event detection probability for one particular case. We also perform extensive Monte Carlo simulations and use numerical techniques to accurately model the coverage process that is generated from the hard-core process with controllable inhibition distance. We use the proposed model to design our network for any desired detection probability and show through Monte Carlo simulations that through our proposed model, we can achieve the desired probability with average error of less that 1%.

In the literature, several coordinated scheduling schemes are available (see. e.g., [73] and the references therein). In most of the existing schemes, a sensor decides whether to turn on or off based on the exact location of itself and that of its neighbors and this information is made available through a GPS. These schemes make sure that turning a particular sensor off does not deteriorate the coverage, which is important especially for time critical events that must be detected immediately. However, the exact information about the location of a sensor and its neighbors, which is used to make decisions, is not always available. Even if this location information is available, it is a strain on the battery of a system as GPS devices consume significant power. In another approach, all the sensors in the network are assigned node IDs. To make switching decisions, each sensor communicates its ID and other information like its transmission power range and number of neighbors to its immediate neighbors. Using this information, a sensor decides whether its contribution is essential for ensuring coverage or not. However, assigning nodes IDs raises the issue of scalability of network in these schemes. Moreover, the information exchange between neighboring sensors often make these schemes very complicated and inefficient in terms of energy consumption which causes serious issues during practical implementation. In contrast, our proposed scheme does not require any such information related to the location of sensors, and it allows any level of partial coverage by varying the control parameter d.

3.1 System Description

Consider a domain $\mathcal{D} \subset \mathbb{R}^2$ in which a large number of sensors are randomly deployed for monitoring purposes. We assume that all the sensors are identical, i.e., they have the same initial power levels and same sensing capabilities. The footprint of each sensor is a ball of radius r_s, $B(x, r_s)$, where x is the location of the sensor, and a sensor can detect anything within its footprint, but nothing outside it. The

communication range of each sensor is adjustable and is controlled by varying the transmitted power level. The size of the domain, $\|\mathcal{D}\|$, is very large as compared to the footprint of an individual sensor in order to avoid any boundary effects. Each sensor can switch between on and off states only at discrete time instances $k\Delta t$, where Δt is the sampling interval. A sensor can sense only when it is in the on state. Since sensors are randomly deployed and we are assuming that the number of sensors is large, from Chapter 2 this deployment can be modeled as a homogeneous Poisson point process with intensity λ. Once it is established that the sensor deployment forms a stationary Poisson point process with some intensity λ, then the number of sensors in any region of area A can be determined using Poisson distribution.

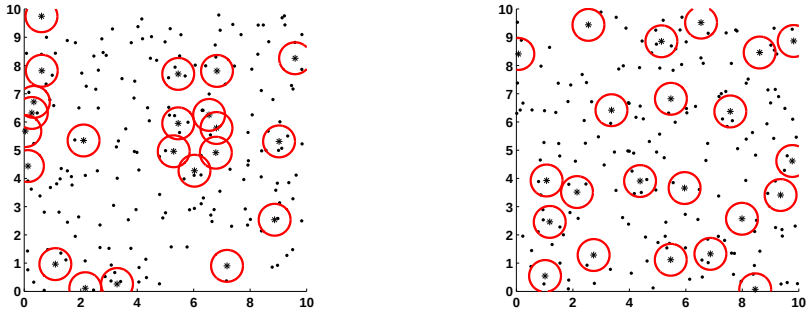
$$P_n(A) = \frac{(\lambda A)^n e^{-\lambda A}}{n!}. \tag{3.1}$$

Because sensor networks that we are studying are often deployed for monitoring purposes, in this work we select coverage as our performance criterion. To be more specific, we require a sensor network to maintain a desired level of partial coverage at any particular time. Indeed, this relaxation of partial coverage is practical especially in the case of persistent events or non-stationary events because probability of detection of these event keeps on increasing with time. In any case, the relaxation allows us to design simple scheduling schemes with small communication cost and simplifies the analysis of the system.

3.2 Problem Motivation

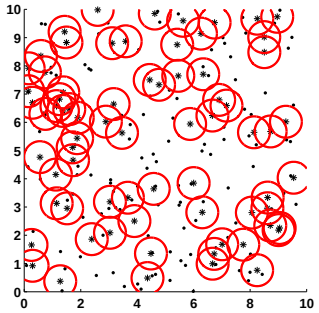
For maintaining partial coverage in an area of interest, a simple approach is random switching in which each sensor randomly decides to be in the on state with some given probability q_r and in the off state with probability $1 - q_r$. For this scheme, the probability that a non-persistent event is detected is [26]

$$P_d = 1 - e^{-\lambda A q_r}, \tag{3.2}$$

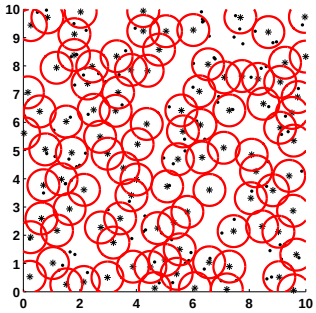


(a) Poisson process $q_r = 0.125$

(b) Hard-core process $q_h = 0.125$



(c) Poisson process $q_r = 0.4323$



(d) Hard-core process $q_h = 0.4323$

Figure 13: Comparison between possible realizations of a sensor network modeled as a Poisson process with random switching (13(a) and 13(c))and a hard-core process with coordinated switching (13(b) and 13(d)).

where A is the area of the sensing footprint of a sensor and q_r is the probability of a sensor to be in the on state. Here, we consider a non-persistent event as an event that does not leave a mark in the environment and must be detected when it occurs. Thus, Equation (3.2) is effectively the expected level of coverage that is provided by the network. In Chapter 2, we proposed a random switching scheme using Equation (3.2), in which given any desired coverage level, P_{des}, we first computed the probability of a sensor to be on, which was

$$q_r = \frac{\ln(\frac{1}{1-P_{des}})}{\lambda A}. \tag{3.3}$$

Then each sensor used this value of q_r to decide whether to turn on or not and in this way the network maintained the desired coverage. One characteristic of this scheme was that each sensor made its switching decision randomly and completely

independent of all the other sensors, which kept the analysis simple because of the Poisson point process. *However, this lack of communication between sensors in decision making typically results in more sensors to be on than are necessary to maintain the desired coverage.* Figures 13(a) and 13(c) depict possible realizations of a sensor network simulated in MATLAB under random switching for two values of q_r, in which the circles represent the footprints of the sensors that are on. It can be observed that the sensors that are on have formed clusters in certain areas leaving other areas completely uncovered.

The clustering of the sensors in the on state can be avoided by inhibiting any two sensors that are closer than a certain minimum distance to be on simultaneously. We call this minimum allowed distance between the sensors in the on state as the *inhibition distance*. Using this inhibition distance, we define the *d-neighborhood* of a sensor as

Definition 3.2.1. *The d-neighborhood of a sensor located at $x_i \in \mathbb{R}^2$, is defined as*

$$\mathcal{N}_d(x_i) = \{j : x_j \in B(x_i, d) \text{ for all } x_j \in \Phi\},$$

where Φ is a set that contains the locations of all the sensors.

Given a sensor located at x_i is on, we do not want any other sensor in its *d-neighborhood* to be on at the same time, and the point process in which this inhibition distance is enforced is a hard-core point process.

Definition 3.2.2. *[30] A hard-core point process is a point process in which the constituent points cannot lie together closer than a minimum specified distance.*

Figures 13(b) and 13(d) demonstrate realizations of the same network as in Figures 13(a) and 13(c) but under the scheme using a hard-core process with inhibition distances $2r_s$ and r_s. It is apparent that the level of coverage achieved by the same number of sensors has improved in this case with the inhibition distance imposed.

3.3 Probabilistic Sleep Scheduling Scheme

In this section, we present a novel switching scheme that we developed as part of this work, in which sensors communicate with all the sensors in their d-neighborhood to decide whether they should turn on or not. We derive analytical expression for event detection probability P_d for the case when $d = 2r_s$, and for all other cases we develop a model for P_d using numerical techniques. For this scheme, we show that event detection probability is higher than the random switching scheme that was discussed in Section 3.2. The basic concept that we use is that of a hard-core point process.

In order to decide whether to be in the on or off state at some time instance k, sensor i located at x_i has to perform the following steps:

Scheme 4. Proposed scheduling scheme

1. *Generate a number m_i such that $m_i \sim$ unif $[0, 1]$.*
2. *Transmit m_i to all the sensors in $\mathcal{N}_d(x_i)$.*
3. *Turn on if $m_i < m_j$ for all $j \in N_d(x_i)$.*

In the literature on point processes ([30], [31], and [77]), m_i is called a mark of a point i. Under this scheme, two sensors cannot be on simultaneously if the distance between them is less than d, which is the inhibition distance.

Lemma 3.3.1. *Under Scheme 4, the probability of a sensor being on at some time instance k is*

$$q_h = \frac{1 - e^{-\lambda A_d}}{\lambda A_d}, \tag{3.4}$$

where A_d is the area of a ball with radius d.

Proof. This lemma follows directly from the results in [30], so we only give a sketch of the proof. Since the switching decisions are made independent of time, k will have no effect. Now, according to the proposed scheme, a sensor at x_i with a mark m_i remains

on if no sensor in $B(x_i, d)$ has a mark lower than m_i. Therefore, the probability $q_h(m)$ that a sensor at location x with mark m remains *on* is

$$q_h(m) = \sum_{n=0}^{\infty} \frac{(\lambda A_d)^n e^{-\lambda A_d}}{n!} (p_h)^n,$$

where p_h is the probability that a sensor located in $B(x, d)$ has a mark greater than m. As marks are uniformly distributed in $[0, 1]$, so

$$p_h = \int_m^1 dt = 1 - m,$$

which yields

$$q_h(m) = e^{-\lambda A_d m}. \tag{3.5}$$

Because m is uniformly distributed in $[0, 1]$, integrating $q_h(m)$ from 0 to 1 produces the desired result. □

From [30], we know that the intensity of a hard-core process that is generated from a Poisson process with intensity λ is

$$\lambda_h = q_h \lambda, \tag{3.6}$$

where q_h is given by Equation (3.4). Figure (14) demonstrates the relationship between the probability of a sensor to be in the on state and the inhibition distance. It is evident from the figure that increasing the inhibition distance decreases the probability of a sensor to be on which obviously decreases the coverage. Therefore, we can use the inhibition distance as a control parameter to achieve any level of desired coverage. However, to use d as a control parameter, we need to completely characterize the probability of detecting a non-persistent event P_d, which is directly related to coverage, in terms of the inhibition distance. In the next section we will first present a derivation of the expression for P_d under the proposed scheme for a special case when $d = 2r_s$. After this special case, we will present a general expression, relating P_d and d, that we developed through Monte Carlo simulations.

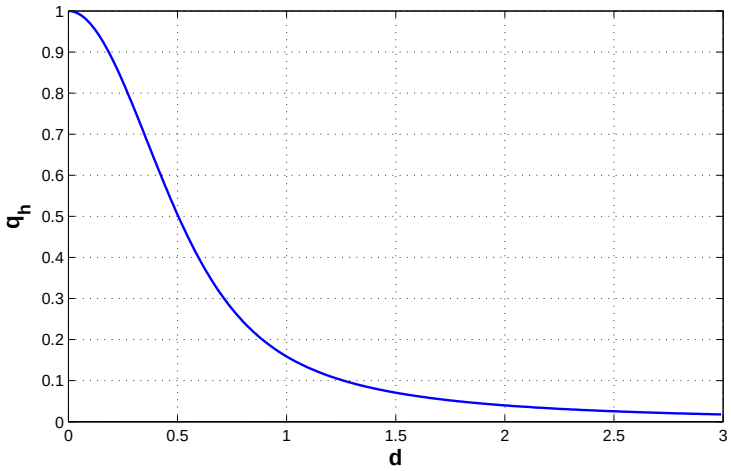


Figure 14: Variation in the probability of a sensor being on, q, with variation in the inhibition distance, d. Here radius of the footprint of a sensor is $r = 1$.

3.4 Event Detection Probability: $d = 2r_s$

The scheme that we presented in Section 3.3 is simple to implement, since each sensor only transmits a random number to all its neighbors and makes its switching decision based on the random numbers it receives from them. However, the primary challenge in this work is the analysis of this scheme because to achieve any desired level of partial coverage, we need an explicit relationship between detection probability, P_d, and inhibition distance d. In the existing literature on spatial point processes and sensor networks, no such relationship exists. The reason that makes this process extremely hard to analyze is that the points in the point process formed by the sensors that are in the on state are no longer independent. Moreover, by using the inhibition distance as our control parameter, we allow the sensing footprints to overlap. The coverage process that results from these overlapping disks no longer corresponds to a standard hard-core process and is therefore not being studied in detail. Next, we present the analysis of this scheme in which we derived relationship between P_d and d for a particular case of $d = 2r_s$. This value of d ensures that there is no overlap between the footprints of sensors, so no redundancy.

Theorem 3.4.1. *Under Scheme 4 when $d = 2r_s$, the probability of an event, located at some point $x_e \in \mathcal{D}$, being detected is*

$$P_d = -\frac{1}{4}e^{-4\lambda A} - \sum_{n=1}^{\infty}\left(-\frac{1}{3}\right)^n e^{-\lambda A}\sum_{k=0}^{n-1}\frac{(-3\lambda A)^k}{k!}, \tag{3.7}$$

where A is the area of the footprint of a sensor.

Proof. For the proof of this theorem we assumed that an event occurred at an arbitrary location x_e and we want to find its detection probability. Let us define a random variable Y_i such that

$$Y_i = \begin{cases} 0 & \text{if sensor at } x_i \text{ is off} \\ 1 & \text{if sensor at } x_i \text{ is on} \end{cases} \tag{3.8}$$

Let m_i be the mark associated with the sensor at x_i. We know that the event will be detected if there is at least one sensor in the on state in $B(x_e, r_s)$. Since, $d = 2r_s$, it means that no two sensors having a mutual distance less than $2r_s$ can be on simultaneously. This implies that at most one sensor can be on in $B(x_e, r_s)$, and this will be the sensor having the lowest mark.

$$P_d = \sum_{n=1}^{\infty}\frac{(\lambda A)^n e^{-\lambda A}}{n!}P(\text{sensor with lowest mark is on}). \tag{3.9}$$

Since there are n sensors in the $B(x_e, r_s)$, we define another random variable X such that

$$X = i \text{ if } \; m_i = \min\{m_1, m_2, \ldots, m_n\},$$

i.e., X is equal to the index of the sensor with the lowest mark in $B(x_e, r_s)$. Since the mark of any sensor in $B(x_e, r_s)$ can be the smallest, so

$$P_d = \sum_{n=1}^{\infty}\frac{(\lambda A)^n e^{-\lambda A}}{n!}\sum_{i=1}^{n}P(Y_i = 1|X = i)P(X = i).$$

Now, the random variable X will be equal to i if the marks of all the other $n-1$ sensors in $B(x_e, r_s)$ are greater than m_i. Thus,

$$P(Y_i = 1|X = i)P(X = i) = \int_0^1 q_h(m_i)(1 - m_i)^{n-1}dm_i,$$

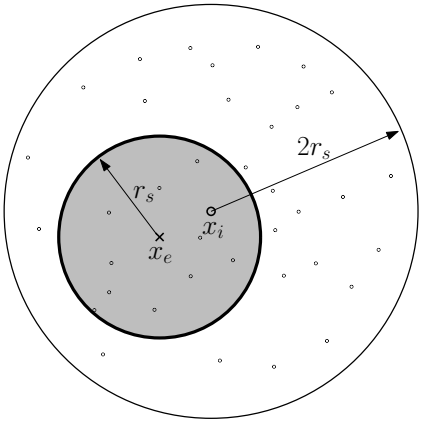


Figure 15: x_e is the location of the event and x_i is the location of the sensor with lowest mark in the shaded region.

where $q_h(m_i)$ is given by Equation (3.5). Figure (15) illustrates that for a sensor at x_i to be on there should be no sensor with a mark lower than m_i in $B(x_i, 2r_s)$. We already know that the sensor at x_i has the lowest mark in $B(x_e, r_s)$, which is the shaded region in Figure (15). Therefore, we only need to confirm that there does not exist any sensor with a mark lower than m_i in the unshaded region, whose area is $\pi(2r_s^2) - \pi r_s^2 = 3A$. Therefore, from Equation (3.5) the probability that a region of area $3A$ has no mark less than m_i is $e^{-3\lambda A m_i}$, which implies

$$P_d = \sum_{n=1}^{\infty} \frac{(\lambda A)^n e^{-\lambda A}}{n!} \sum_{i=1}^{n} \int_0^1 e^{-3\lambda A m_i} (1 - m_i)^{n-1} dm_i.$$

Integrating m_i from 0 to 1 yields the same result for all i, indicating that the same quantity is being added n times.

$$P_d = \sum_{n=1}^{\infty} \frac{(\lambda A)^n e^{-\lambda A}}{n!} n \int_0^1 e^{-3\lambda A t} (1 - t)^{n-1} dt.$$

Solving this integral and performing some simple algebraic manipulations yields the desired result.

□

In the next section, we show through Monte Carlo simulations how different coverage levels can be achieved by varying the inhibition distance.

3.4.1 Simulation

To validate the results proved in Section 3.3, we performed Monte-Carlo simulations of the sensor network. For the simulations, we considered a rectangular domain of dimensions [10 x 10] having an area $A_{dom} = 100$. In this domain, sensors were scattered with an intensity of 2 sensor per unit area, i.e., $\lambda = 2$, which implies that the expected number of sensors in the total area is $\Lambda = A_{dom}\lambda$. The footprint area of each sensor was $A = 1$, so that it was very small as compared to the total area. Events were generated randomly at each time instance throughout the domain and we considered total of 20,000 events in a single iteration of the simulation. To increase the accuracy of the results even further, the values of P_d and P_d were averaged over 100 iterations of the simulation. To ascertain the effects of the inhibition distance on the coverage, we considered four different values of d, i.e., $d \in \{r_s/2, r_s, 3r_s, 2r_s\}$. For each of these values of d, we computed the intensity of the corresponding hard-core point process $\lambda_h = q_h\lambda$, where q_h is given by Equation (3.4).

We started by selecting $d = r_s/2$, applied the coordinated switching scheme proposed in Section 3.3, and measured the detection probability, P_d. Next, we used the value of q_r corresponding to $d = r_s/2$ and measured the detection probability P_d under random switching. The same process was repeated for the remaining values of d. Figures (16(a) and 16(b)) show the results of the simulations. The simulations demonstrate that as d increases, the detection probability decreases exactly as we expected. Moreover, for all the cases, detection probability under the proposed scheme is better than the random scheme which validate our scheme. Finally, we consider the special case of $d = 2r_s$ and compare the detection probability under proposed scheme with the value computed analytically. The simulation yielded $P_d = 0.2527$,

very close 0.2498, which is the value computed analytically, indicating the validity of our analysis

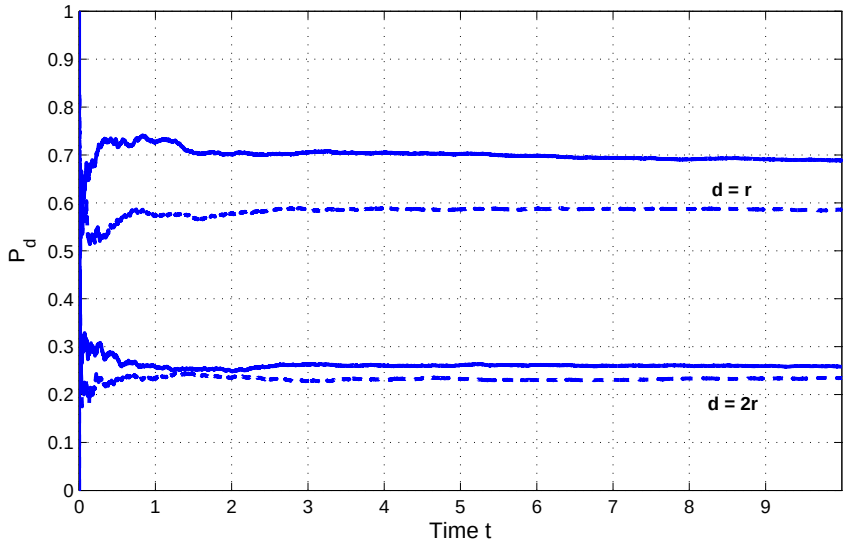
3.5 Event Detection Probability: Generalized Case

The detection probability derived in Equation (3.7) is valid only when $d = 2r_s$, i.e., when sensor footprints are not allowed to overlap. However, this is a restrictive case because for a given deployment intensity, λ, and sensing radius, r_s, we have no control on the achievable coverage. Moreover, it can be observed from the simulation results presented later in this section that we cannot ensure more than 25% coverage for this case. Therefore, it is imperative to find a general relationship between P_d and d to design for any desired coverage level. To derive this relationship, we have to rely on numerical techniques because by using d as a control variable, we are allowing the sensing footprints to overlap with each other. The coverage process that results from these overlapping disks is complicated and the theoretical analysis of this process presents enormous difficulties. This makes numerical techniques an attractive approach for this problem.

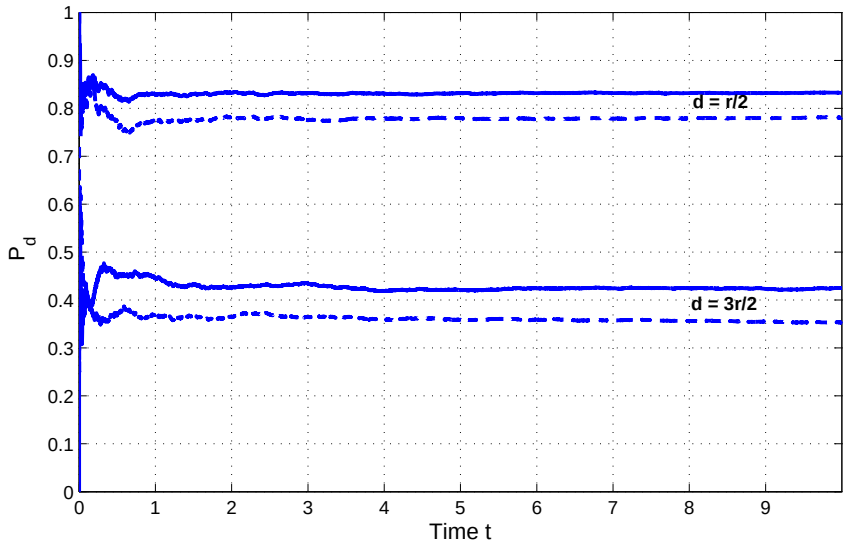
3.5.0.1 Simulation Setup

In these simulations we consider a rectangular deployment region of dimensions $[30 \times 30]$, in which sensors with footprint area A are randomly deployed with intensity λ. The parameters that we vary over different simulations are λ, A, and d and for each set of these values we simulate the network to find the corresponding detection probability. We simulate the network for all the values of λ, A, and d belonging to the sets $\bar{\lambda}$, $\bar{A}$, and Ω respectively, where

$$\begin{aligned} \bar{\lambda} &= \{1, 2, 3, 4, \ldots, 14, 15\}, \\ \bar{A} &= \{0.3, 0.5, 0.7, 1, 1.3, 1.5, 1.7, 1.9, 3, 4, 5\}, \\ \Omega &= \{0.1r, 0.2r, 0.3r, \ldots, 1.9r, 2r\}. \end{aligned}$$



(a) Inhibition distances $d = r$ and $d = 2r$



(b) Inhibition distances $d = r/2$ and $d = 3r/2$

Figure 16: Comparison of event detection probability between the proposed scheme (solid plot) and the random switching scheme (dashed plot) for different inhibition distances. For this simulation $\lambda = 2$ and $A = 1$.

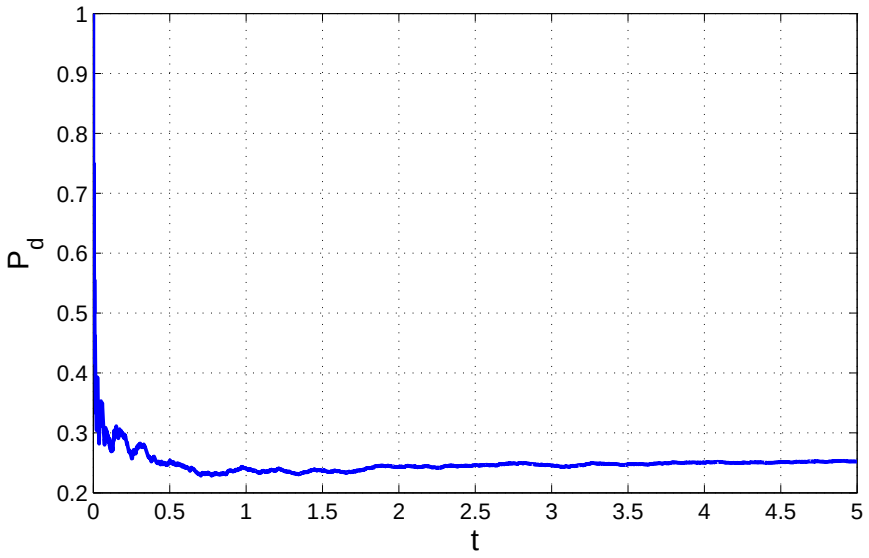
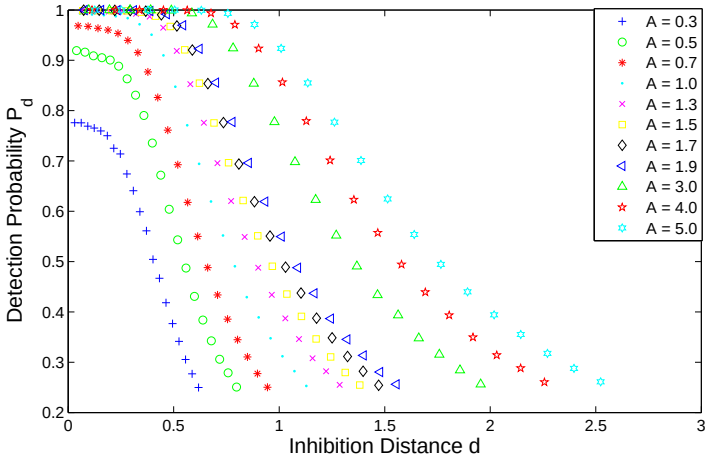


Figure 17: Event detection probability under the proposed scheme for the case $d = 2r$. $P_d = 0.2527$ which is close to the analytical value 0.2498.

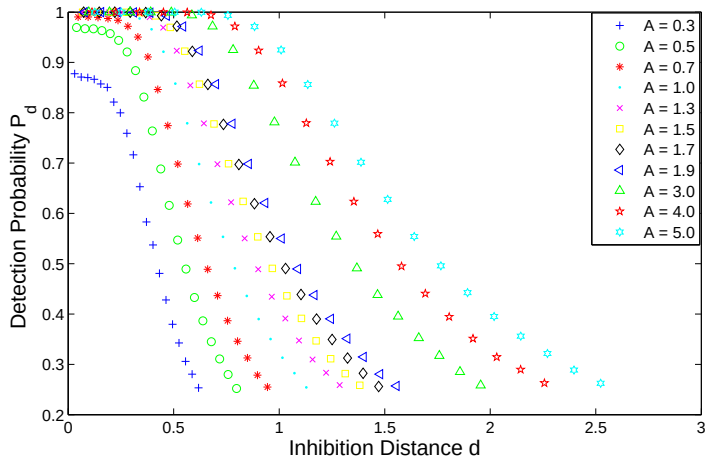
For each $(\lambda, A) \in (\bar{\lambda} \times \bar{A})$, we simulate the network 100 times for all $d \in \Omega$, and for each value of d, 10,000 events are randomly generated in the region of interest. For each case, P_d is computed using the relative frequency definition of probability, i.e.,

$$P_d = \frac{\text{Number of events detected}}{\text{Total number of events}}$$

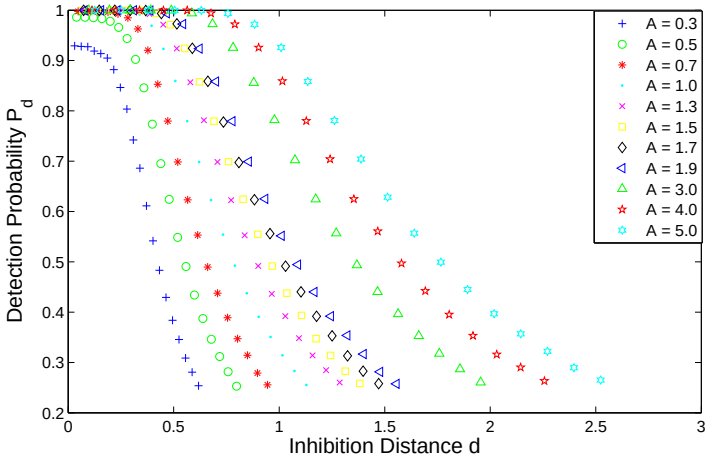
The results of this simulation are presented in Figure (18), in which each sub-figure from parts (a) to (h) shows plots of P_d vs d for all the values of $A \in \bar{A}$. Moreover, each of these sub-plots correspond to one value of deployment intensity λ. Thus, Figure (18) presents all the data that we require to model this process.



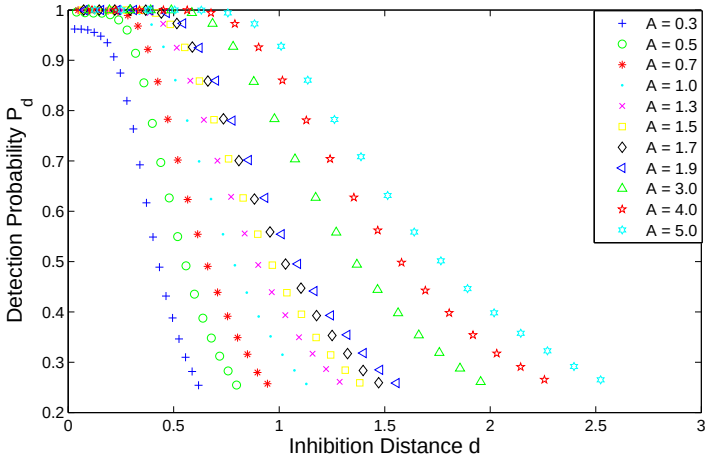
(a) Deployment Intensity $\lambda = 5$



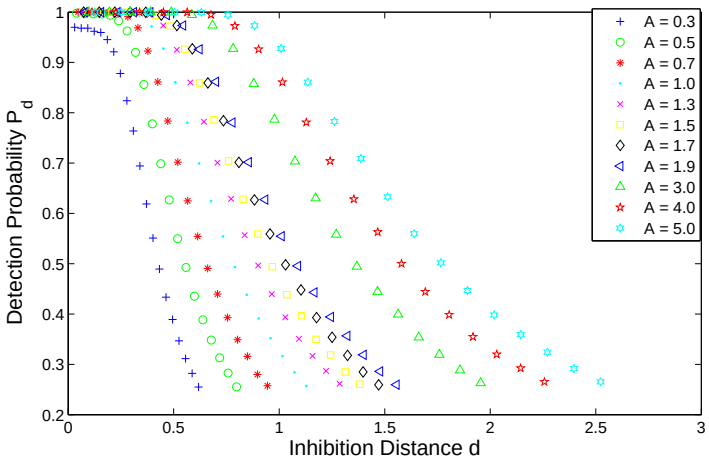
(b) Deployment Intensity $\lambda = 7$



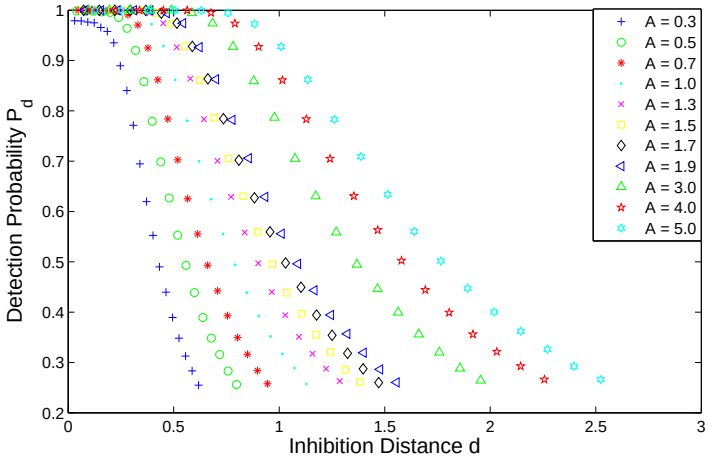
(c) Deployment Intensity $\lambda = 9$



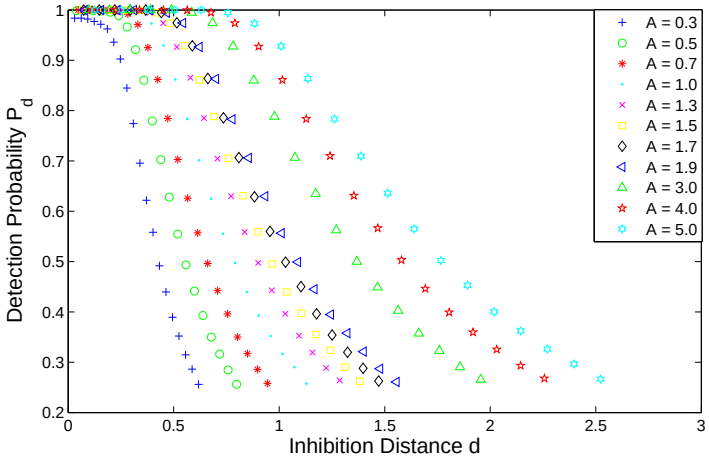
(d) Deployment Intensity $\lambda = 11$



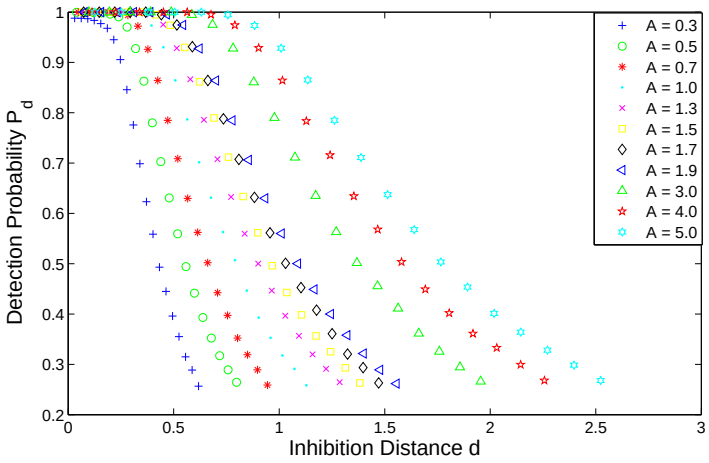
(e) Deployment Intensity $\lambda = 12$



(f) Deployment Intensity $\lambda = 13$



(g) Deployment Intensity $\lambda = 14$



(h) Deployment Intensity $\lambda = 15$

Figure 18: Plots of simulated detection probability P_d vs inhibition distance d for different values of footprint areas A and deployment intensities λ

After simulating the network for all the different combinations of the design parameters (λ, A, d), we develop a mathematical model to relate detection probability with inhibition distance using the curve fitting tool of MATLAB. Based on the careful examination of data and study of renewal process theory, we select the following equation for modeling our data.

$$P_d = 1 - c_1 \exp\left(-c_2 \exp(c_3 d)\right) - c_4. \tag{3.10}$$

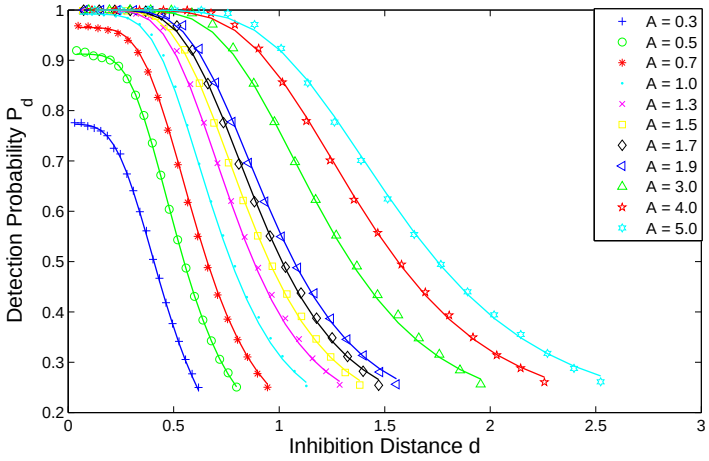
In the above equation, (c_1, c_2, c_3, and c_4) are the coefficients that uniquely correspond to a P_d vs d curve. The results of using the above expression to model our data are presented in Figure (19), and from the figure we can see that the results are remarkable since the curves generated by Equation (3.10) accurately fit the simulated data. This accuracy of the curve fitting clearly indicates that Equation (3.10) is a good choice for modeling this data. Thus, we have a closed form expression relating detection probability P_d and our control parameter d, using which we can find the value of d that can maintain any desired detection probability, P_{des}.

Theorem 3.5.1. *Given any desired level of event detection probability, P_{des}, the inhibition distance that can guarantee to maintain P_{des} is*

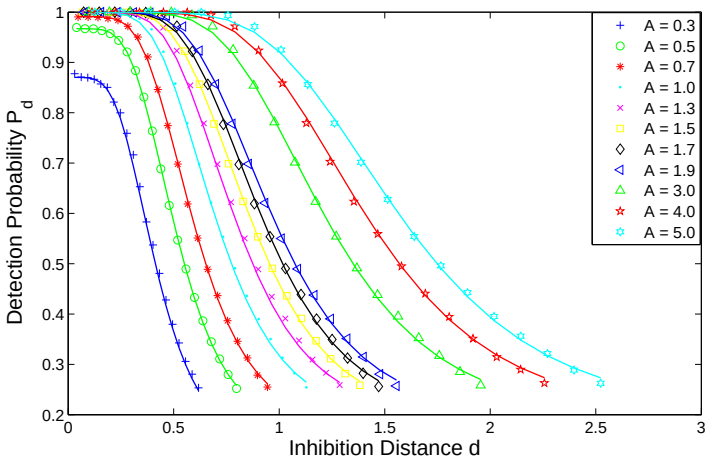
$$d = \frac{1}{c_3} \ln \left[\frac{1}{c_2} \ln \left[\frac{c_1}{(1 - c_4) - P_{des}} \right] \right]. \tag{3.11}$$

Proof. From the simulation results presented in Figures (18 and 19), we have shown that Equation (3.10) is a good choice for modeling the detection probability of this coverage process. Therefore, Equation (3.10) is a closed form expression relating our desired objective P_{des} and our design parameter d. Using this equation and performing simple algebraic manipulations yields Equation (3.11), which concludes the proof. □

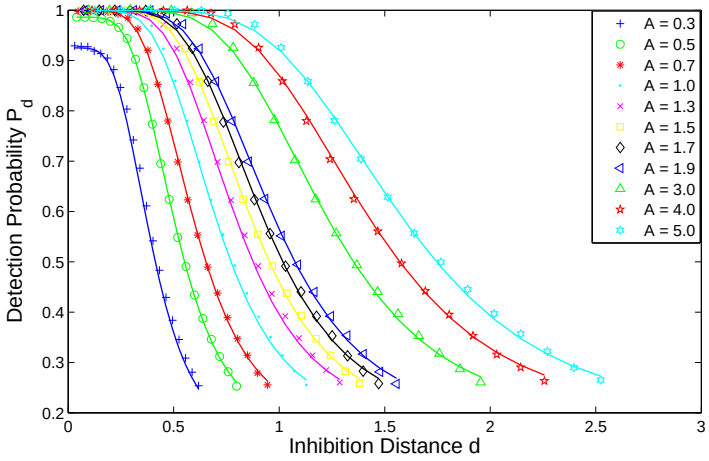
Thus, we can explicitly find out the value of d through Equation (3.11) that can guarantee any given P_{des}.



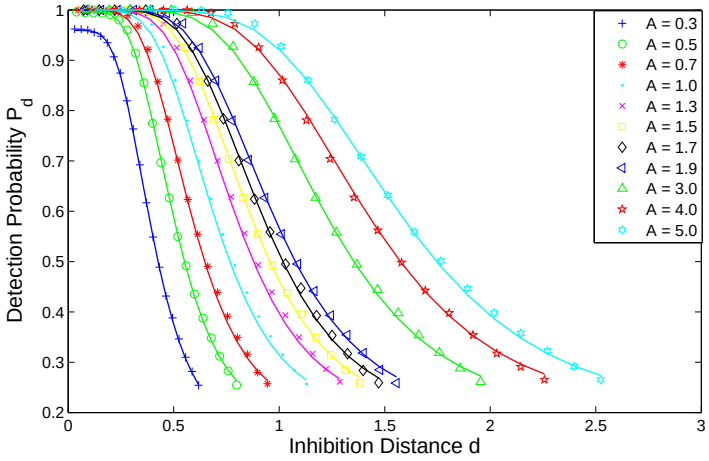
(a) Deployment Intensity $\lambda = 5$



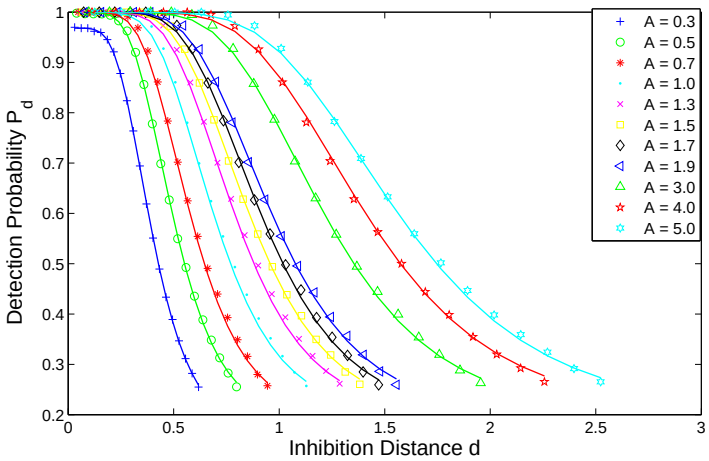
(b) Deployment Intensity $\lambda = 7$



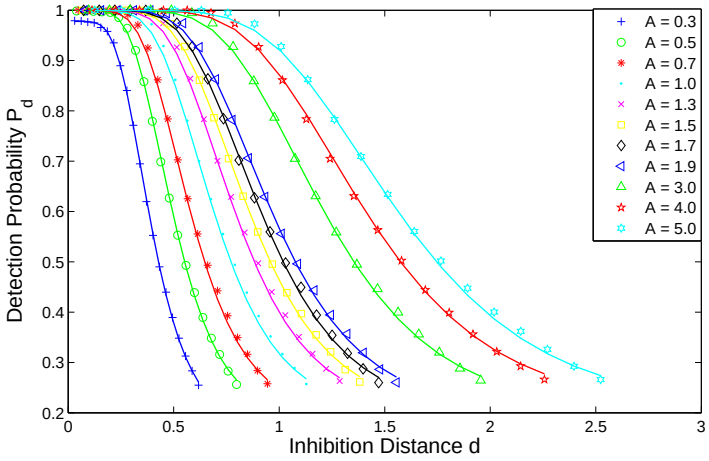
(c) Deployment Intensity $\lambda = 9$



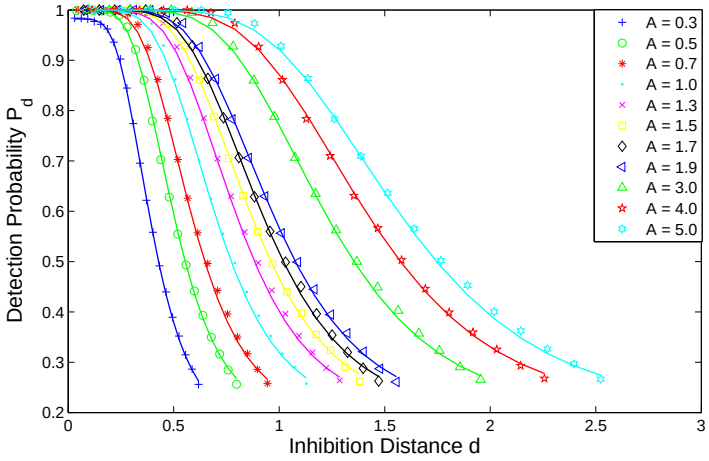
(d) Deployment Intensity $\lambda = 11$



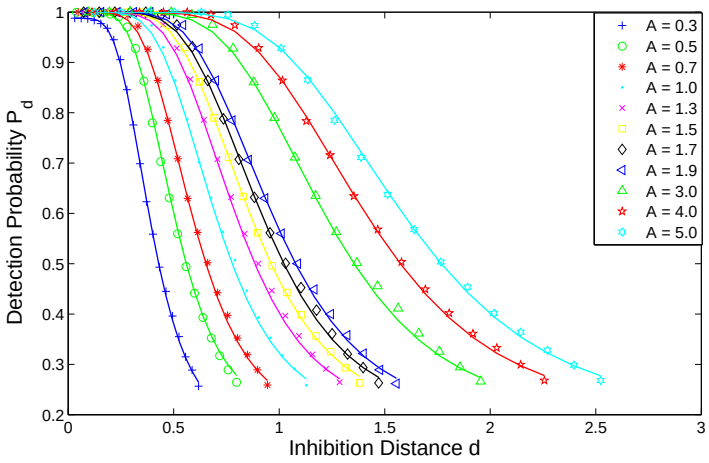
(e) Deployment Intensity $\lambda = 12$



(f) Deployment Intensity $\lambda = 13$



(g) Deployment Intensity $\lambda = 14$



(h) Deployment Intensity $\lambda = 15$

Figure 19: Curve fitting on the simulated data of detection probability P_d vs inhibition distance d according to Equation (3.10).

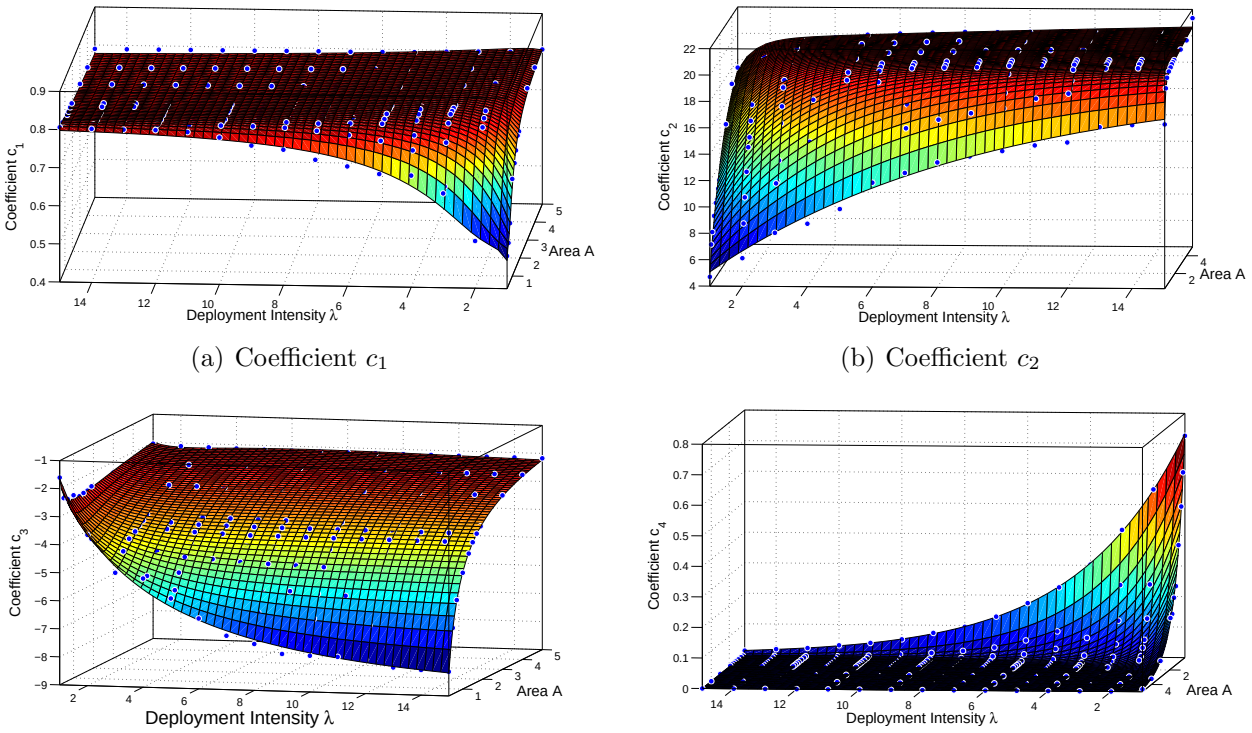
However, this relationship between P_d and d is in terms of coefficients (c_1, c_2, c_3, c_4) and for any deployment intensity λ and footprint area A, there is unique set of these coefficients that relate d with P_d. For the simulated scenarios, the MATLAB curve fitting tool generated the values of these coefficients that minimized the error between the data observed through simulations and the curves obtained from Equation (3.10). The values of these coefficients are plotted in Figure (20). To use these coefficients

in the design phase, we consider their values as our data points and again fit curves on these points. After carefully analyzing the data and trying various types of curves and adjusting their parameters, the curves that best fit the data points with minimum mean square errors are as follows

$$
\begin{aligned}
c_1 &= \frac{\ln(\lambda^{0.1808}A^{0.1853})\exp\left(\lambda^{-2.015}A^{-2.259}-0.301\right)-0.1133}{(\lambda A)^{0.7147}}+0.747 \\
c_2 &= 18.81\exp\left(0.006558\lambda^{0.6213}A^{0.4462}\right)-16\exp\left(-0.387\lambda A\right)+0.4462 \\
c_3 &= \frac{\ln(\lambda^{-1.603}A^{-1.621})\exp(A^{0.5179}\lambda^{0.6171}-0.3833)-2.506}{\lambda^{-0.1546}A^{0.3966}}-0.1546 \\
c_4 &= 0.9788\exp(-0.9655\lambda A) \qquad (3.12)
\end{aligned}
$$

The curve fitting that resulted from the above equations is shown in Figure (20) and it can be observed that the quality of fitting is very good. Although these equations look complicated, for a deployed network with given λ and footprint area A, it is a simple task to compute the four coefficients from the above expressions. Once the coefficients are computed, there values are substituted in Equation (3.11) to compute the design parameter d. *Therefore, we have completely characterized this process in terms of the parameters λ, A, and P_{des}. For any given set of these values, we can accurately find P_d under a hard-core point process, which is a major contribution of this work.*

After modeling the process, we verify its accuracy through Monte Carlo simulations. The simulation setup is the same in which the region of interest is a rectangular region of dimension $[30\times30]$ where sensors are randomly deployed. For this simulation we sweep the values of λ from 5 to 20 with an increment of two where as the values of



(a) Coefficient c_1

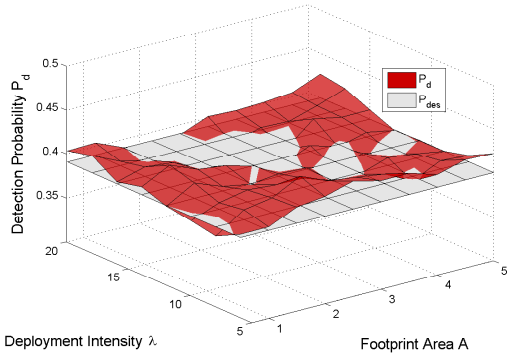
(b) Coefficient c_2

(c) Coefficient c_3

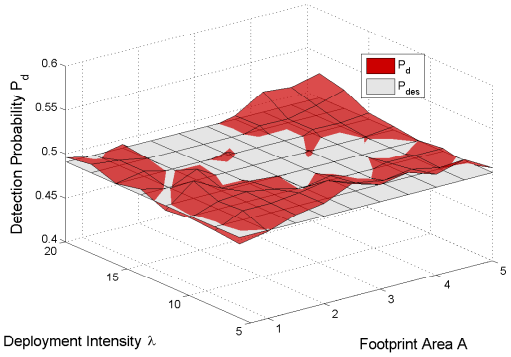
(d) Coefficient c_4

Figure 20: Curve fitting on the simulated data for coefficients c_1, c_2, c_3, and c_4

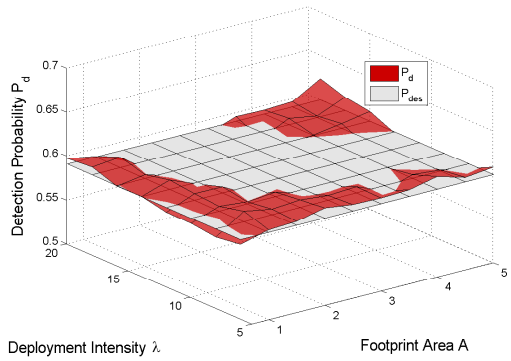
A are swept from 0.7 to 4 with an increment of 0.5. Moreover, for each combination of the values of λ and A, we simulate the network to maintain detection probability P_{des} from 0.4 to 0.9 with increments of 0.1. For each case, network is simulated for ten iterations and in each iteration 1000 events are randomly generated. The detection probability for each run is computed using the relative frequency definition of the probability and the total detection probability P_d for each case is the average value of ten iterations. The results of this simulation are presented in Figure (21). Each sub-figure corresponds to a single value of P_{des} that is simulated for all the possible combinations of λ and A. The two surfaces that are plotted are P_d (dark surface) and P_{des} (light surface). The important thing to observe is that in all the cases, the maximum error between P_{des} and P_d is less then 3% and the average percentage error was then 1% which is phenomenal and proves the fact that the expressions we



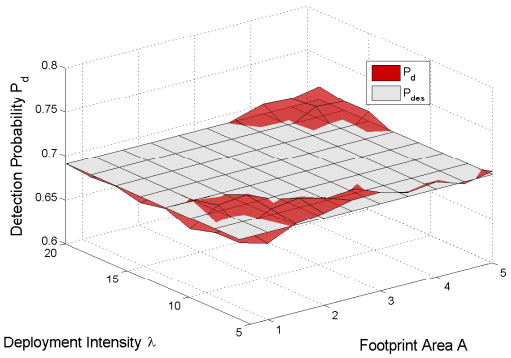
(a) $P_{des} = 0.4$, Mean error = 0.9606%, Maximum error = 2.94%'



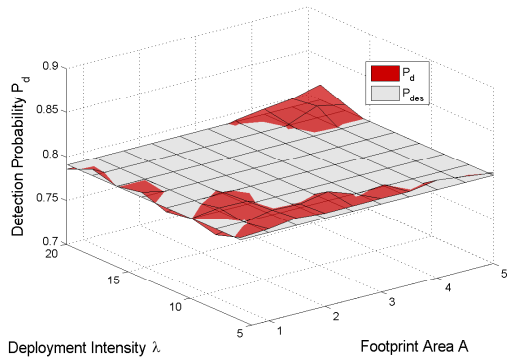
(b) $P_{des} = 0.5$, Mean error = 0.8564%, Maximum error = 2.4%'



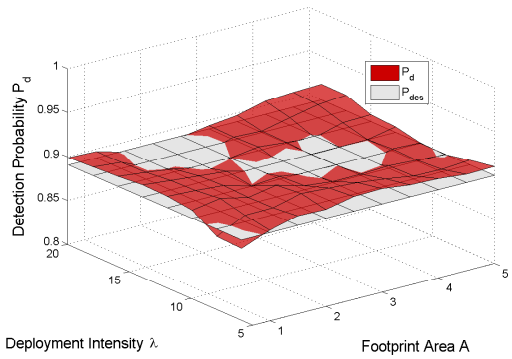
(c) $P_{des} = 0.6$, Mean error = 0.7752%, Maximum error = 2.38%'



(d) $P_{des} = 0.7$, Mean error = 0.7942%, Maximum error = 2.11%'



(e) $P_{des} = 0.8$, Mean error = 0.6956%, Maximum error = 1.87%'



(f) $P_{des} = 0.9$, Mean error = 0.6805%, Maximum error = 2.13%'

Figure 21: Comparison between desired performance P_{des} and the actual performance P_d obtained after designing d from the proposed model for parameters $A = \{0.5, 1, 1.5, \ldots, , 4.5, 5\}$ and $\lambda = \{5, 7, 9, \ldots, 18, 20\}$

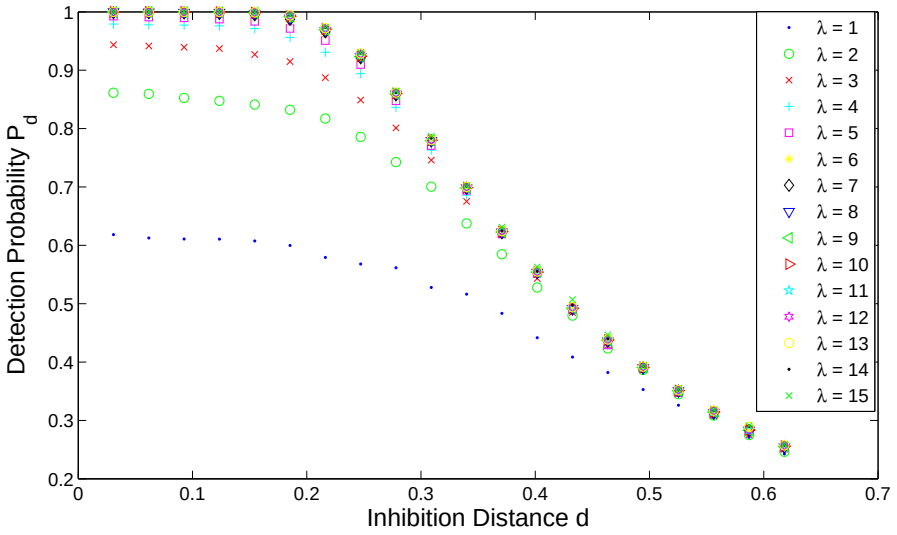
derived to model this process are accurate.

Although the modeling of the hard-core process we presented up till now is accurate as shown from the simulations, a potential problem is that the quality of modeling can deteriorate as we move away from the domain of the parameters that were used in the simulation. For instance, in this simulation, our design parameters were λ, A and d. As far as d is concerned, selecting it from $0.1r_s$ to $2r_s$, covers all the potentially useful values. For deployment intensity, we selected λ from 5 to 15, which is also sufficient, since from Figure 22(a) it can be observed that the curves for $\lambda > 10$ are very much the same. This leaves only A as the parameter that can cause problems. It turns out that we do not even need to simulate all the different values of sensing radii. In our implementation of this scheme, the inhibition distance was basically a multiple of the sensing radius, i.e., $d = \xi r_s$, where $\xi \in (0.1, 2)$ controls the overlap of the sensing regions that is required to maintain P_{des}. Thus, we refer to ξ as the *overlap coefficient*. In Figure 22(b), we plotted the relationship between P_d and ξ for a constant value of λ, from which we can see that except for small values of sensing area, i.e., $A < 1$, the detection probability is independent of the sensing area. This implies that for densely deployed networks consisting of sensors with reasonable sensing area, the only thing that matters to maintain a given P_{des} is the parameter ξ, from which we can simply compute the inhibition distance by multiplying it with the sensing radius, i.e., $d = \xi r_s$.

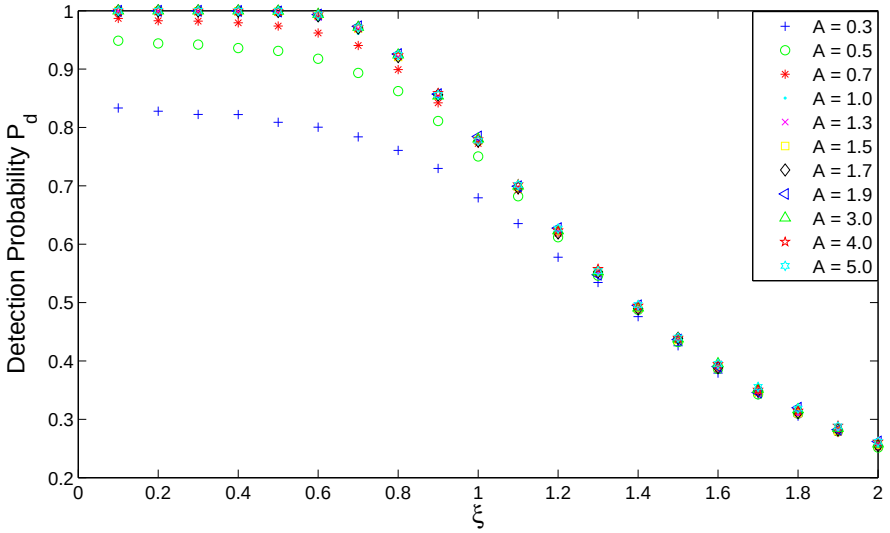
To verify the claim that P_d depends on the overlap coefficient only, we performed further simulations for the values of footprint areas that were not used in modeling the system. We selected A from 5.5 to 8 with increments of 0.5. For each of these simulations, we selected $A_{ref} = 5$ and using A_{ref} and λ we computed the inhibition distance, d_{ref}, from Equation (3.11) and the corresponding overlap coefficient

$$\xi = \frac{d_{ref}}{r_{ref}},$$

where r_{ref} was the sensing radius correspond to area A_{ref}. Finally, we computed the



(a) Detection Probability vs Area



(b) Detection Probability vs ξ

Figure 22: Part(a): Depicted are the plots of P_d vs d for all the values of $\lambda = \{1, 2, \ldots, 15\}$ and fixed $A = 5$. Part(b): Depicted are the plots of P_d vs the overlap coefficient ξ for different values of A.

desired inhibition distance $d = \xi r_s$, where r_s is the sensing radius corresponding to the simulated sensing area A. Figure (23) depicts the results of the simulation under the setup described above for sensing areas $A = \{5.5, 6, \ldots, 7.5, 8\}$ and from these plots we can observe that by using $A_{ref} = 5$ and finding the overlap coefficient ξ, the

average error between P_d and P_{des} is still less than 1.35% in all the cases which shows the validity of the proposed approach.

3.6 Comparison with Random Scheme

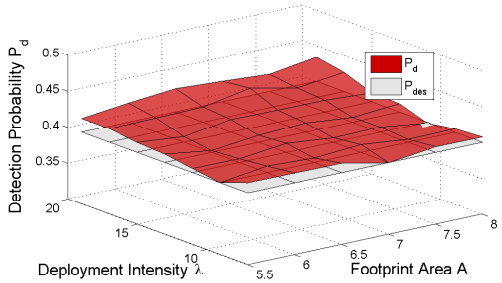
The purpose of studying the hard-core point process was to propose a switching scheme for efficiently utilizing the available energy resources and increasing the lifetime of the network by improving its coverage profile and reducing the number of redundant sensors that are in the on state because of random switching. After completely characterizing the coverage properties of this process, the next step is to compare the performance of the proposed scheme with random scheme and verify our claim that the proposed scheme is indeed energy efficient. In both the schemes, we started with a Poisson point processs with intensity λ. Under random scheme, each sensor switched on or off completely independent of all the other sensors. This process of independently deleting points from a point process is called independent thinning, and independent thinning of a Poisson process results in a Poisson process with intensity $q_r\lambda$, where q_r is the probability of a point not deleted and is given by

$$q_r = \frac{\ln\left(\frac{1}{1-P_{des}}\right)}{\lambda A}.$$

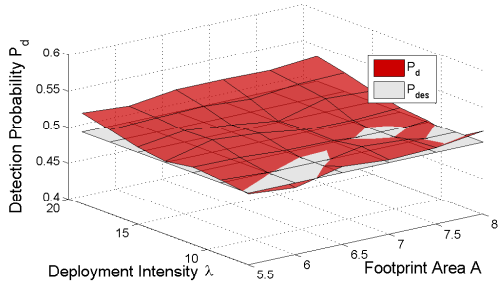
In the context of sensor networks, q_r is the probability of a sensor being on. In contrast, in a hard-core point process, points are not deleted independently from the original Poisson process. Instead, points interact with each other to make this decision, and the probability of a point not being detected q_h is

$$q_h = \frac{1 - \exp(-\lambda\pi d^2)}{\lambda\pi d^2},$$

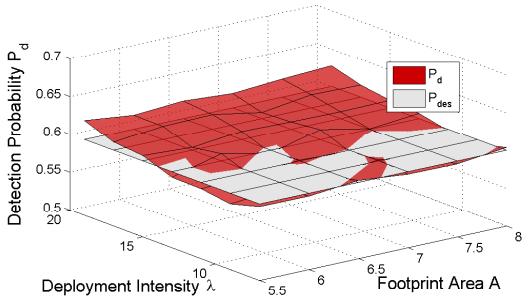
so the intensity of a hard-core point process is $q_h\lambda$.



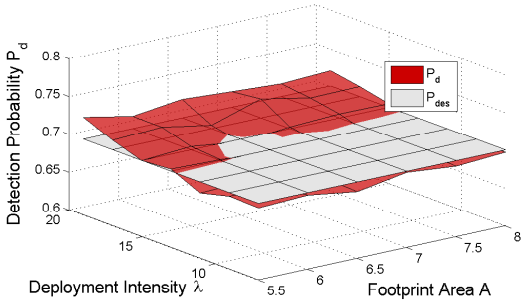
(a) $P_{des} = 0.4$, Mean error = 1.35%, Maximum Error = 3.3%'



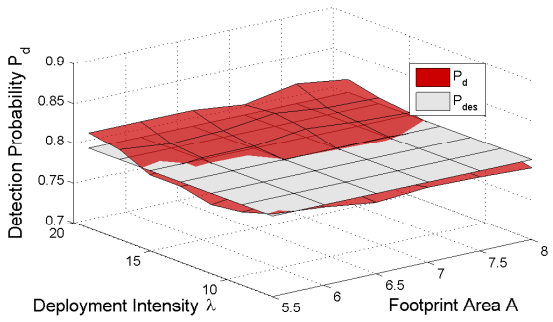
(b) $P_{des} = 0.5$, Mean error = 1.3%, Maximum Error = 2.4%'



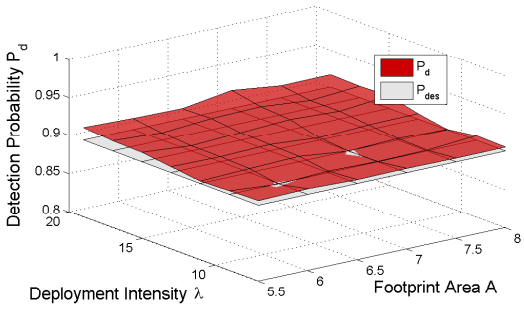
(c) $P_{des} = 0.6$, Mean error = 0.96%, Maximum Error = 2.72%'



(d) $P_{des} = 0.7$, Mean error = 0.92%, Maximum Error = 2.89%'



(e) $P_{des} = 0.8$, Mean error = 0.87%, Maximum Error = 2.46%'



(f) $P_{des} = 0.9$, Mean error = 0.5%, Maximum Error = 2.35%'

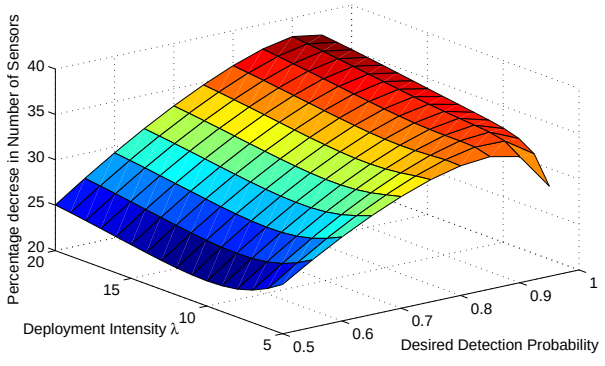
Figure 23: Comparison between desired performance P_{des} and the actual performance P_d obtained after designing d from the proposed model for parameters $A = \{5.5, 6.0, \ldots, 8\}$ and $\lambda = \{5, 7, 9, \ldots, 18, 20\}$

If N_{poi} is the total number of sensors in the Poisson process, then to maintain P_{des} the expected number of sensors in the on state will be $q_r N_{poi}$ and $q_h N_{poi}$ under the random schemes and the proposed scheme respectively. To compare these schemes, we computed the percentage decrease in the number of sensors in the on state under our proposed scheme using the expression

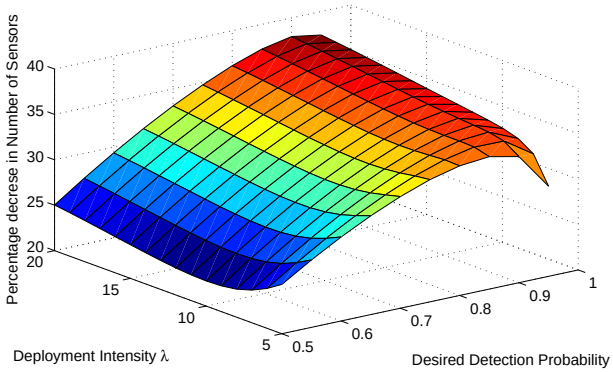
$$\% \text{ decrease in number of sensors} = \frac{q_r - q_h}{q_r} 100.$$

The results of this analysis are presented in Figure (24), from which it can be observed that the expected number of sensors in the on state under the proposed scheme is always less than the random scheme. Moreover, for all the cases, the percentage decrease in the expected number of sensors start from 20% and went upto 38%. Another important observation is that the percentage decrease in the number of sensors increases as the deployment intensity λ increases. This result is intuitive because with the increase in the number of deployed sensors, the redundancy in the number of sensors that are on will also increase, which will result in a better performance under the proposed scheme.

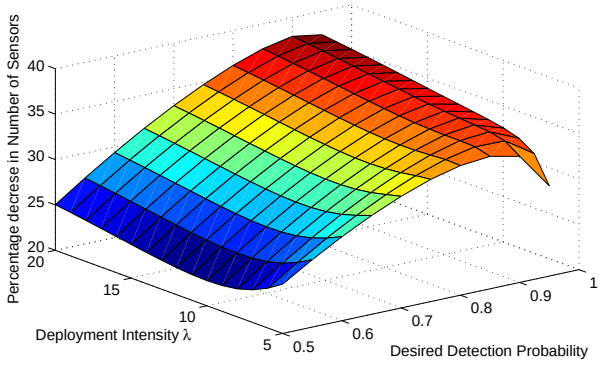
Although the expected number of sensors in the on state decreases by 20% to 38%, this improvement is achieved at the expense of sensors communicating with their neighbors, whereas no communication is involved in random switching. For the proposed scheme to be energy efficient, we have to show that even after the added cost of communication, the total energy consumption under the proposed scheme is less than the random scheme. For this analysis, we first select a sensing platform that is commonly used in the wireless sensor networks for deploying the types of networks we are considering in this work, i.e., networks comprising of large numbers of low cost, low power devices with limited sensing, processing and communication capabilities. This platform is the 3rd generation of Berkeley motes called MICA2 motes which are most commonly used in wireless sensor networks community. These motes have been used in a large number of practical systems for the purposes of surveillance,



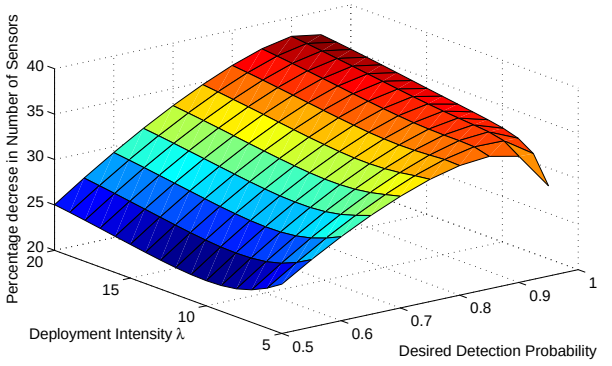
(a) Sensing Radius $r_s = 3$



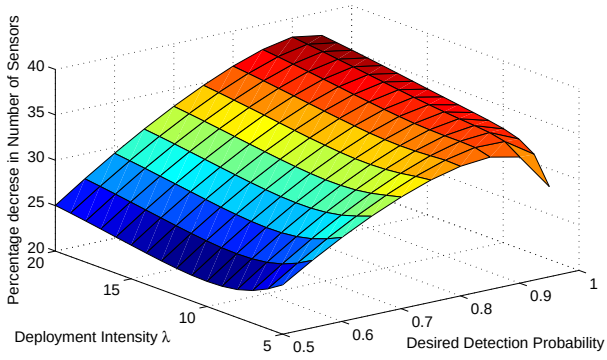
(b) Sensing Radius $r_s = 4$



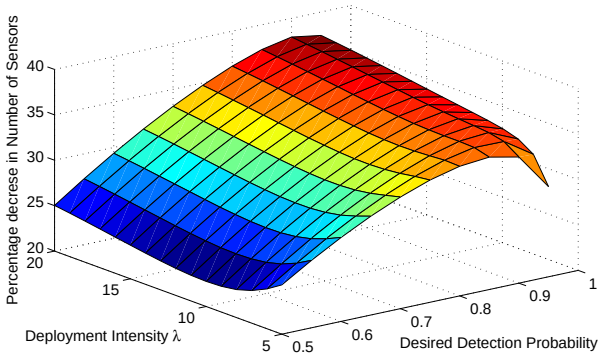
(c) Sensing Radius $r_s = 5$



(d) Sensing Radius $r_s = 6$



(e) Sensing Radius $r_s = 7$



(f) Sensing Radius $r_s = 8$

Figure 24: Comparison of expected number of sensors in the on state in random scheme and the proposed scheme for various values of deployment intensity, λ, and desired detection probability P_{des}. Each sub plot corresponds to different value of sensing range r_s.

environmental monitoring and precision agriculture (see e.g., [47], [48], and [9] and the references therein).

A MICA2 mote is powered through two AA batteries that typically have a current rating of 2200mAh. Each of these motes is equipped with a 433 MHz Chipcon radio that can transmit at different power settings from -10dBm to 20dBm, and can provide an effective data rate of 12.4Kbps [63]. Moreover, various sensor boards are available for these motes that are connected through a surface mount on the motes. The sensors that are available on various sensor boards for a MICA2 mote include photocells, thermistors, microphones, sounder and magnetic sensors [62]. In this analysis we considered magnetic sensor HMC1002 from Honeywell that detects magnetic field from the nearby objects and has an omni-directional field of view. These sensors are used to detect and track any metallic object like vehicles or weapons in their field of view.

For this platform, we need to model the energy consumption owing to both data communication and sensing, both of which are functions of distance. The model for sensing is

$$E_S(d) = \alpha r_s^2, \tag{3.13}$$

where α is a constant and r_s is the sensing range of the sensor. When the magnetic sensor HMC1002 is used to detect vehicles, its sensing range is between 4m - 8m and its power density is $388\mu W m^{-2}$ [34]. Therefore, total energy consumed by this sensor to monitor a circular area of radius r_s for an interval of length T seconds is

$$E_S(d) = 388 \times 10^{-6} r_s^2 T \text{ J}.$$

The energy consumption model for data transmission according to [76] is

$$E_T(d) = m(a + bd^2) \text{ J}, \tag{3.14}$$

where m is the number of bits transmitted, d is the distance over which data is transmitted and a and b are constants depending on environment. For the Chipcon

radio CC1000 that is available on MICA2 mote, the values for a and b are [76]

$$\begin{aligned} a &= 0.3 \times 10^{-7}\text{J/bit}, \\ b &= 2 \times 10^{-10}\text{Jm}^{-2}/\text{bit}. \end{aligned}$$

In the above models, we need to specify the exact number of bits that are transmitted by each node. In our proposed scheme, each sensor transmits a random number, which, in practical implementation, will require an infinite number of bits to encode and is thus impractical. Therefore, in our simulations each agent generates an eight bit number uniformly distributed between 0 and 255, which makes the proposed scheme implementable on an eight bit processor. However, the question is how is this going to effect the performance of the scheme. Fortunately, limiting the random number to eight bits has no noticeable effect on the performance of the proposed scheme. The only purpose of the random number is to assign a unique identifier to every sensor in a circular disk of radius equal to the inhibition distance, d, such that only the sensor with the lowest identifier can turn on. In the case when eight bit random numbers are generated with uniform distribution, the probability that any two sensors generate exactly equal number in the circular disk of radius d is $\frac{1}{256}$, which is very small. For the rare event that two sensors generate exactly the same number, we updated the proposed scheme as presented in Scheme (5).

Scheme 5. Proposed scheduling scheme

1. *Generate a number m_i such that $m_i \sim$ unif $[0, 1]$.*
2. *Transmit m_i to all the sensors in $\mathcal{N}_d(x_i)$.*
3. *Turn on if $m_i \leq m_j$ for all $j \in N_d(x_i)$.*

In the updated scheme, if two sensors generate same number and that number is minimum in the circular disk of radius d, then both the sensors will turn on. However, this event will occur with probability $\frac{1}{256}$, so it will have no noticeable effect on energy

consumption. Thus, with Scheme 5, we can put $m = 8$ in the energy consumption model in Equation (3.14), which yields

$$E_T(d) = (2.4 \times 10^{-7} + 16 \times 10^{-10} d^2) \text{ J}$$

Since MICA2 motes are powered by two AA batteries, the total energy of the network comprising of N_{poi} sensors is

$$E_{\text{tot}} = N_{poi}(6.6 \times 3600)\text{J}. \tag{3.15}$$

Next we consider energy consumption in the case of random switching scheme and the proposed scheme separately. For random scheme, there is no communication involved for making switching decision, so the only cost is that of sensing. Therefore, at the k^{th} decision time, energy available will be

$$E_{\text{av}}(k) = E_{\text{av}}(k-1) - q_r N_{poi} E_S(d), \tag{3.16}$$

where $E_{\text{av}}(k)$ is the expected energy available in the network at the k^{th} decision time, and T is the length of the time interval for which a sensor remains on for sensing. For this simulation we select $T = 10$ minutes. Here, we want to point out that switching itself is an expensive operation and for a network with lifetime of the orders of months, switching every 10 minutes is a high switching rate. Rewriting the above expression in terms of total energy.

$$\begin{aligned} E_{\text{av}}(k) &= E_{\text{tot}} - k q_r N_{poi}(0.388 \times 10^{-3} r_s^2 T), \\ &= N_{poi}(6.6 \times 3600) - k q_r N_{poi}(0.388 \times 10^{-3} r_s^2 T). \end{aligned}$$

The expected lifetime of the network is the minimum value of k times T such that $E_{\text{av}}(k) \leq q_r N_{poi} E_S(d) T$. Using this fact in the above expression, the lifetime of the network is over when

$$E_{\text{tot}} - k q_r N_{poi}(0.388 \times 10^{-3} r_s^2 T) \leq q_r N_{poi}(0.388 \times 10^{-3} r_s^2 T).$$

By performing simple algebraic manipulations, the expected lifetime of the network is

$$k_r = \left\lfloor \frac{6.6 \times 3600}{q_r(0.388 \times 10^{-3} r_s^2 T)} \right\rfloor. \tag{3.17}$$

In comparison, in the proposed scheme, energy is consumed because of both sensing and communication. At each decision time, all the N_{poi} sensors initially transmit 8 bits to all their neighbors in a disk of radius equal to the inhibition distance d. After that, on average, $q_h N_{poi}$ sensors remain on for time interval of length T for sensing a circular disk of radius r_s. Therefore, under the proposed scheme, energy available at the k^{th} decision time is

$$E_{\text{av}}(k) = E_{\text{av}}(k-1) - N_{poi} E_T(d) - q_h N_{poi} E_S(d), \tag{3.18}$$

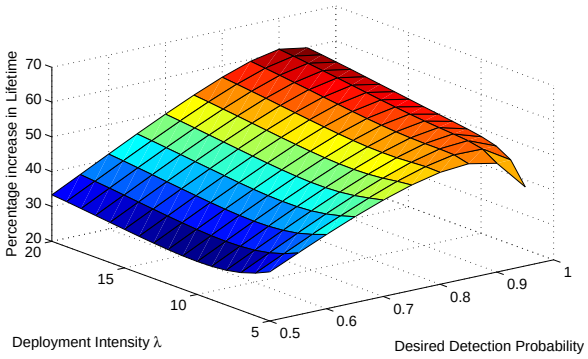
where $E_T(d)$ is the energy consumption due to data transmission and $E_S(d)$ is the energy consumption due to sensing. In terms of total energy,

$$E_{\text{av}}(k) = E_{\text{tot}} - kN_{poi}(2.4 \times 10^{-7} + 16 \times 10^{-10} d^2) - kq_h N_{poi}(0.388 \times 10^{-3} r^2 T).$$

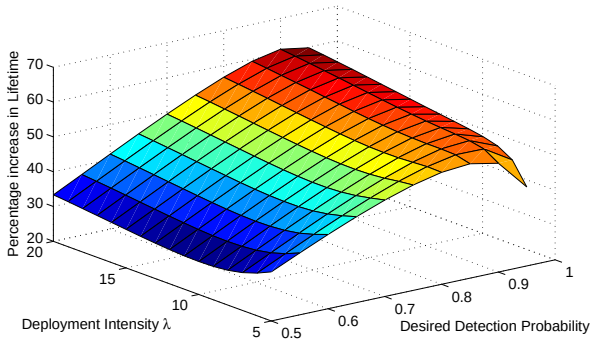
To find the expected lifetime under the proposed scheme, we use the same argument as for the random scheme and the expected lifetime is

$$k_h = \left\lfloor \frac{6.6 \times 3600}{(2.4 \times 10^{-7} + 16 \times 10^{-10} d^2) + (0.388 \times 10^{-3} q_h r^2 T)} \right\rfloor, \tag{3.19}$$

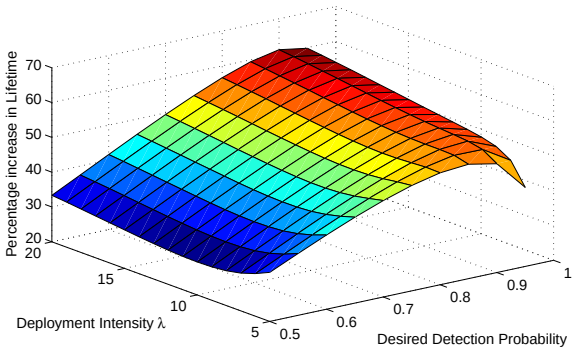
Based on the expressions for expected lifetime of the network under random and proposed scheme, we present the comparison of both schemes in Figure (25), which shows the percentage increase in the lifetime of the network under proposed scheme. It is clear from this figure that under all the different scenarios of sensing ranges, the lifetime of the network increased by 30% to 70% under the proposed scheme. Moreover, this improvement in the lifetime increased with the increase in the deployment intensity and increase in P_{des}. The reason is, higher P_{des} means larger number of sensors in the on state which implies higher level of clustering which resulted in higher gain under the proposed scheme. The same is true for higher deployment intensity.



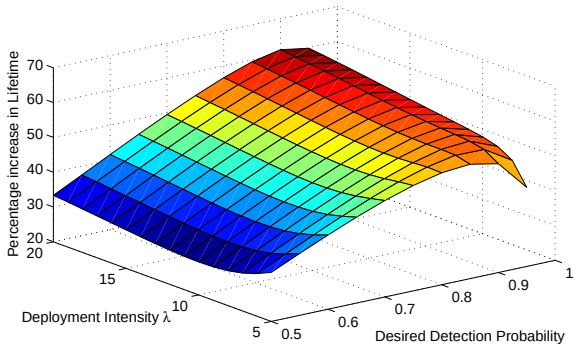
(a) Sensing Radius $r_s = 3$



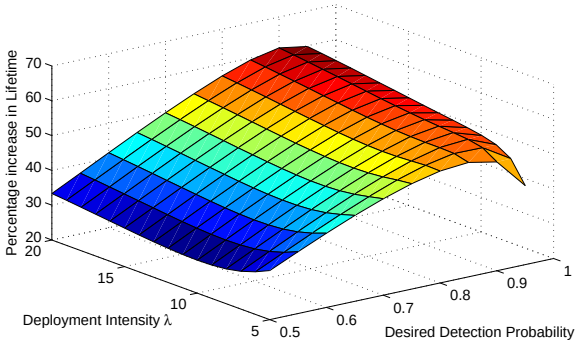
(b) Sensing Radius $r_s = 4$



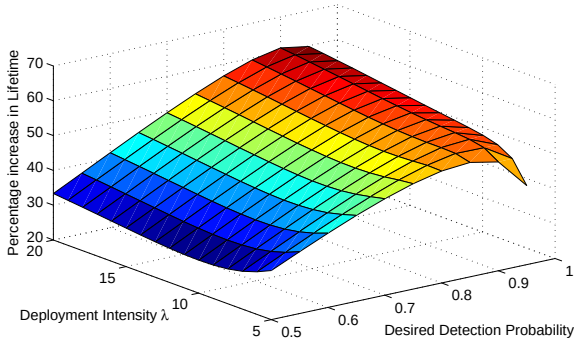
(c) Sensing Radius $r_s = 5$



(d) Sensing Radius $r_s = 6$



(e) Sensing Radius $r_s = 7$



(f) Sensing Radius $r_s = 8$

Figure 25: Comparison of expected lifetime of a network under random scheme and the proposed scheme for various values of deployment intensities and desired detection probabilities. Each sub plot corresponds to different value of sensing range r_s.

3.7 Design of Different Hard-core Point Processes

In Section 3.5, we presented a model relating the detection probability P_d with the inhibition distance d that we developed for a hard-core point process, and we showed through extensive simulations that the developed model can accurately design a hard-core process with desired properties. The process that we modeled is one of the three processes that were initially presented by Matérn [32] as hard-core point processes, all of which were obtained through dependent thinning of a Poisson process. In Matérn I process, he started with a Poisson point process and deleted any two points if the distance between them was less than the inhibition distance d. In Matérn II process, he initially assigned a mark $m \in \text{unif}[0, 1]$ to each point in the process. Then, a point was retained only if no other point in the circular disk of radius d centered at the point had a mark lower than the mark of that point. Therefore, in this model if the distance between two points was less than d, only one of them was deleted. Matérn II process is the model that we used for our sensor scheduling scheme. In Matérn III process, starting again from a Poisson process he retained a point if its distance from all the previously retained points was greater than d. All the three processes that were presented by Matérn fulfilled the basic requirement that their constituent points did not lie closer than d. However, the intensities of these processes are not the same and it is known that the $\lambda_I < \lambda_{II} < \lambda_{III}$, where λ_I, λ_{II}, and λ_{III} are the intensities for Matérn I, Matérn II, and Matérn III processes. These variations in intensities imply that the coverage properties of these processes will also be different.

Matérn I, Matérn II, and Matérn III processes have been individually studied and the first and second order properties of Matérn I and Matérn II processes have been derived. However, there are no results in the existing literature relating the intensities and coverage profile of these processes. These relationships can be useful because they will allow us to use our developed model for Matérn II process to generate any required type of hard-core process. It is also important to point out that the models proposed

by Matérn are not the only ways of generating a hard-core point process. In fact, in this work, we studied two different methods for generating this process. We are interested in these processes because of their potential usefulness in the context of mobile sensor networks and facility location problems. In order to design these two point processes with desired coverage properties, we derived relationships between the intensities of these processes and Matérn II process.

3.7.1 Simple Sequential Inhibition (SSI) Process

The first process that we studied in this work is a Simple Sequential Inhibition (SSI) process that is a hard-core point process on a bounded region but with finite number of points, and it does not require a Poisson process for its generation. In this model, the first point is generated randomly in the region of interest $\mathcal{D} \subset \mathbb{R}^2$. In each subsequent step, a random point is generated and this point becomes a part of the point process if its distance from all the previous points of the process is greater than d. The steps for generating an SSI process are presented in Algorithm 1.

Given the number of points N_{SSI} that we want to deploy in a region, Algorithm 1 outlines the steps that need to be followed to ensure that these points form a hard-core process. However, in this work, the problem that we are interested in is slightly different. In the problem that we are interested in, every point in the point process is the center of two disks, one with radius r and the other with radius d. Our first objective is to form a hard-core point process such that the disk of radius d associated with each point does not contain any other point of the process. Our second objective is that the coverage process

$$\mathcal{C} = \bigcup_{i=1}^{N_{SSI}} B(x_i, r),$$

where x_i is the lcoation of the i^{th} point and $B(x_i, r)$ is a ball of radius r centered at x_i that results from this hard-core process ensures a desired level of coverage in the domain $\mathcal{D}$ in which the points are deployed.

Algorithm 1. Simple Sequential Inhibition (SSI) Process

Given: N_{SSI}, d

Step 1:

- *Generate a random point in the region of interest, i.e.,* $x_1 \in \mathcal{D}$.

for $k = 2 : N_{SSI}$

$pos = 0$.

while *(*$pos == 0$*)*

- *Generate a random point* $r \in \mathcal{D}$.
- *Find* $dist(x_i, r)$ *for all* $i \in \{1, 2, \ldots, k-1\}$.

if $dist(x_i, r) < d$ *for all* $i \in \{1, 2, \ldots, k-1\}$

$x_k = r, \quad pos = 1$.

end

end

The first objective can be accomplished from Algorithm 1, but for the second objective, we need to know the exact inhibition distance d and the number of points that if deployed under SSI process with parameter d will ensure P_{des}. In the existing literature on point processes, no solutions exist for this problem. In this work, we solved this problem by developing an explicit relationship between the intensities of Matérn II process and SSI process. The rationale for this approach is that we have already developed a relationship between d and P_{des} for Matérn II process. Using this relationship we can find the inhibition distance d and the number of points N_h that will result in P_{des} in the case of Matérn II process. However, for the same number of points, the coverage of SSI process and Matérn II process is not guaranteed to be same [30]. Thus, we need to find the number of points N_{SSI} that can guarantee P_{des} for inhibition distance d.

To find the number of point N_{SSI}, we developed Algorithm 2, in which P_{des}, A_{dom}, and r are the desired detection probability, total area of the domain of interest and radius of the coverage disk of each point. In Step 1 of the algorithm, the inhibition distance d is computed based on the proposed expressions for Matérn II process. Using this inhibition distance, the intensity and the number of points N_h in the Matérn II process are computed. In Step 2, N_h points are initially deployed according to SSI process by following Algorithm 1 and for this process coverage is computed. If P_{des} is achieved, then the algorithm terminates. Otherwise, the number of deployed points is incremented by ζ. The algorithm terminates when P_d becomes equal to or greater than P_{des} and the corresponding number of points are stored as N_{SSI}.

In this work, we established a relationship between N_h and N_{SSI} empirically using extensive simulations and Algorithm 2. For these simulations, we considered a rectangular area of dimensions $[30 \times 30]$ as $\mathcal{D}$, and in this domain we deployed point process for the cases when $A = 5$ and for $P_{des} = \{0.40, 0.45, \ldots, 0.95\}$, where $A = \pi r^2$. These point processes were deployed according to Algorithm 2, and from the data obtained from these simulations, we found the relationship

$$N_{SSI} = \lceil 1.119 N_h - 0.1475 \rceil. \tag{3.20}$$

Thus, Equation (3.20) gives a relationship between the number of points of a Matérn II and an SSI process that can ensure P_{des}. To verify the accuracy of this relationship, we generated hard-core point processes from Algorithm 3 for $P_{des} = \{0.40, 0.45, \ldots, 0.95\}$ and $A = \{5.0, 6.0, \ldots, 20\}$. The results of this process are presented in Figure (26), and from the figure we can see that maximum error between P_{des} and P_d was 3.22% and the mean error was 0.9% which proved that our modeling of the process is quite accurate. In Figure (27) typical realizations of point processes that were generated through the proposed model are presented for the cases of P_{des} equal to 0.5 and 0.75 respectively.

Algorithm 2. Estimation of N_{SSI}

Given: *P_{des}, A_{dom}, and r*

Step 1:

- *Compute d from Equation (3.11).*
- *Compute $N_h = q_h N_{poi}$,*

  where $q_h = \frac{1-\exp(\lambda\pi d^2)}{\lambda\pi d^2}$ and $N_{poi} \sim Poi(\lambda A_{dom})$
- $N_{SSI} = N_h$

Step 2:

- *Deploy N_{SSI} points according to Algorithm 1 in the domain $\mathcal{D}$.*
- *Compute the detection probability P_d.*

While $P_d < P_{des}$

- $N_{SSI} = N_{SSI} + \zeta$.
- *Repeat Step 2.*

end

Algorithm 3. SSI Process for P_{des}

Given: P_{des}, r, A_{dom}

Step 1:

- *Compute d from Equation (3.11).*
- *Compute $N_h = q_h N_{poi}$,*

  where $q_h = \frac{1-\exp(\lambda\pi d^2)}{\lambda\pi d^2}$ and $N_{poi} \sim Poi(\lambda A_{dom})$
- *Compute N_{SSI} from Equation (3.20).*

Step 2:

- *Deploy N_{SSI} points according to Algorithm 1 in the domain $\mathcal{D}$.*

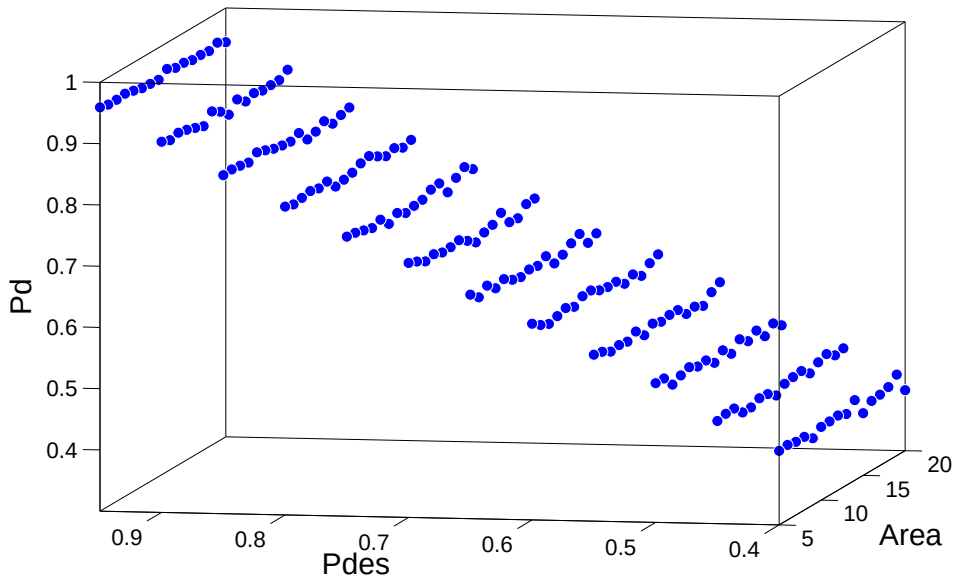
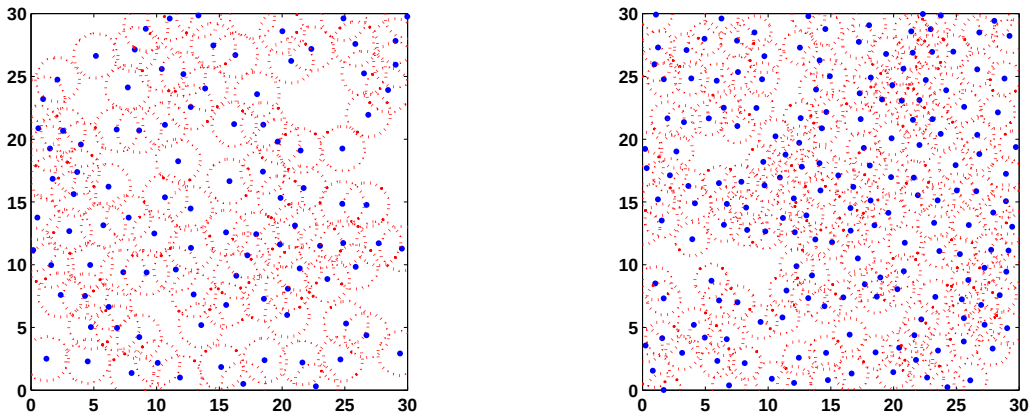


Figure 26: Detection Probability that is obtained after designing from our proposed model. Max Modeling Error 3.22%. Average Modeling Error 0.9%



(a) $P_{des} = 0.5$, $P_d = 0.5117$, $N_{SSI} = 103$ (b) $P_{des} = 0.75$, $P_d = 0.7591$, and $N_{SSI} = 182$

Figure 27: Deployment of a network under the proposed scheme with $P_{des} = 0.7$. In part(a) exact inhibition distance $d = 1.408$ is enforced while in part (b) $d + 0.5$ distance is enforced.

3.7.2 Mobility Based Hard-core Point Process

In this section, we design and analyze a hard-core point process that is motivated from the fact that the intensity of a hard-core process can be increased by introducing small vibrations in the points of the process [51]. These vibrations, typically introduced through a force-biased algorithm ([51], [54]) to increase the intensity of the process, correspond to the mobility of agents in the case of mobile sensor networks. Since the primary focus of this work is scheduling and coordination of wireless sensor networks, so we used this process to propose a distributed coverage control algorithm for mobile sensor networks.

Given d and N_m, Algorithm 4 deploys N_m points in a domain of interest $\mathcal{D}$ and ensure that the distance between any two points is at least d. To achieve this point process, in Step 1 of the algorithm, N_m points are uniformly distributed in $\mathcal{D}$. Then to enforce inhibition distance requirements, each point updates its location in Step 2. The controllers in Step 2 are both repulsive controllers that push points away from each other and from the boundary of the domain. In the first controller, a point located at x_i remains stationary if there are no other points in $B(x_i, d)$. However,

Algorithm 4. Mobility-based Hard-core Point Process

Given: N_m *and* d

Step 1:

- *Generate* N_m *random points in* $\mathcal{D} \subset \mathbb{R}^2$ *such that each point is uniformly distributed in* $\mathcal{D}$.

Step 2:

- *Each point updates its location according to the following controller*

$$\dot{x}_i = \begin{cases} \sum\limits_{j \in \mathcal{N}_d(x_i)} (x_i - x_j) & |\mathcal{N}_d(x_i)| \neq 0 \\ 0 & \textit{otherwise} \end{cases}$$

  where $\mathcal{N}_d(x_i)$ *is the* d*-neighborhood of the* i^{th} *agent located at* x_i.

- *Points avoid leaving the boundary of* $\mathcal{D}$ *by the following control law*

$$\dot{x}_i = \begin{cases} (x_i - x_{bd}) & \|x_i - x_{bd}\| < \delta \\ 0 & \textit{otherwise} \end{cases}$$

  where x_{bd} *is the point on the boundary of* $\mathcal{D}$ *closest to* x_i.

if the d-neighborhood of a point is not empty, then this controller forces the point to move in a direction opposite to the average of the locations of all other points in its d-neighborhood, and this repulsive controller remains active as long as d-neighborhood of the point is not empty. The other controller in Step 2 avoids the boundaries of the domain $\mathcal{D}$. In the scenario where an agent is near boundary and has other agents in its d-neighborhood, these two controllers can be combined to satisfy both the constraints. The combined controller will be

$$\dot{x}_i = \sum_{j \in \mathcal{N}_d(x_i)} (x_i - x_j) + (x_i - x_{bd}),$$

where x_{bd} is the point on the boundary closest to x_i. However, as in the case of SSI process, our objective is to deploy a point process $\Phi = \{x_1, x_2, \ldots, x_{N_m}\}$ such that

the corresponding coverage process

$$\mathcal{C} = \bigcup_{i=1}^{N_m} B(x_i, r),$$

can guarantee the desired coverage properties P_{des}. Again it is important to point out that the radii of the coverage disks and the inhibition disks, i.e., r and d are different. This implies that agents are not allowed to be closer than a distance d but there coverage disks are allowed to overlap since d can be less than r.

To ensure that any given P_{des} is maintained, we need the inhibition distance d and the exact number of points N_m that need to be deployed. To find these parameters, we followed the same procedure that we followed for SSI process in Algorithm 2. Algorithm 5 is used to relate the intensities of Matérn II process and the mobility based process. In this algorithm, given P_{des}, we first computed d and the number of points N_h that would ensure that Matérn II process will maintain P_{des}. In the equation $N_h = q_h N_{poi}$, N_{poi} is a Poisson distributed number with intensity λA_{dom}, where λ is the intensity of a Poisson process. For this model, any value of λ that is sufficiently large will serve the purpose because we showed in Figure 22(a) that for sufficiently large intensity, d becomes independent of λ. Therefore, for this simulation we selected $\lambda = 10$, but any higher value of λ would have served the purpose. We deployed N_h points according to Algorithm 4 and computed the detection probability. Then we kept on increasing the number of deployed points by $\zeta = 5$ until P_d was no longer less than P_{des}. We used Algorithm 5 to deploy a mobility based hard-core process in a rectangular region of dimension $[30 \times 30]$ for $P_{des} = \{0.40, 0.45, \ldots, 0.95\}$ and $A = \{5.0, 6.0, \ldots, 15\}$, where $A = \pi r^2$. For each simulation, the values of N_h and N_m are presented in Tables (28) and (29) respectively. We plotted N_h vs N_m for each case separately and used linear fitting on the data, and by fitting linear curves for all the cases, we found the following relationship between N_h and N_m

Algorithm 5. Estimation of N_m

Given: *P_{des}, A_{dom}, and r*

Step 1:

- *Compute d from Equation (3.11).*
- *Compute $N_h = q_h N_{poi}$,*

  where $q_h = \frac{1-\exp(\lambda\pi d^2)}{\lambda\pi d^2}$ and $N_{poi} \sim Poi(\lambda A_{dom})$
- $N_m = N_h$

Step 2:

- *Deploy N_m points according to Algorithm 4 in the domain $\mathcal{D}$.*
- *Compute the detection probability P_d.*

While $P_d < P_{des}$

- $N_m = N_m + \zeta$.
- *Repeat Step 2.*

end

$$N_m = \lceil cN_h + 5 \rceil, \tag{3.21}$$

where

$$c = 0.6944 P_{des} + 0.8048,$$

and N_m is the number of required agents.

Area	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10**	**11**	**12**	**13**	14	**15**
$P_{des} = 0.40$	72	61	52	45	41	37	33	30	29	26	24
$P_{des} = 0.45$	81	69	58	53	47	42	38	34	32	30	28
$P_{des} = 0.50$	93	78	68	58	51	47	43	40	35	34	31
$P_{des} = 0.55$	104	87	76	68	58	52	49	44	40	37	35
$P_{des} = 0.60$	116	96	84	72	64	59	54	49	44	41	40
$P_{des} = 0.65$	131	108	92	81	72	65	59	54	51	46	43
$P_{des} = 0.70$	145	122	104	92	80	73	66	61	56	52	49
$P_{des} = 0.75$	164	134	116	103	90	80	74	67	63	58	55
$P_{des} = 0.80$	183	154	131	115	103	93	85	77	71	68	63
$P_{des} = 0.85$	210	179	156	136	117	106	96	90	82	77	72
$P_{des} = 0.90$	251	216	180	165	144	127	118	107	100	92	86

Figure 28: Number of points required in the original hard-core process.

Area	**5.0**	**6.0**	**7.0**	**8.0**	**9.0**	**10**	**11**	**12**	**13**	14	**15**
$P_{des} = 0.40$	82	67	60	52	51	44	39	35	34	31	29
$P_{des} = 0.45$	97	80	71	63	57	49	48	44	37	40	38
$P_{des} = 0.50$	111	94	80	73	66	58	53	50	45	44	41
$P_{des} = 0.55$	127	112	94	83	73	66	60	58	52	50	45
$P_{des} = 0.60$	146	124	112	94	81	75	69	64	60	56	55
$P_{des} = 0.65$	166	143	124	108	97	85	80	74	67	62	61
$P_{des} = 0.70$	190	162	144	123	115	104	89	84	76	73	65
$P_{des} = 0.75$	222	184	161	141	127	114	106	98	89	86	77
$P_{des} = 0.80$	258	219	188	165	148	135	124	110	103	98	90
$P_{des} = 0.85$	306	255	223	188	170	166	144	132	128	115	113
$P_{des} = 0.90$	371	315	267	246	205	189	174	166	161	142	125

Figure 29: Number of points required in the mobility based hard-core process.

Algorithm 6. Distributed Coverage Control of Mobile Sensor Networks

Given: P_{des}, r

Step 1:

- *Compute d from Equation (3.11).*
- *Compute* $N_h = q_h N_{poi}$,

  where $q_h = \frac{1-\exp(\lambda\pi d^2)}{\lambda\pi d^2}$ *and* $N_{poi} \sim Poi(\lambda A_{dom})$
- *Compute* N_m *from Equation (3.21).*

Step 2:

- *Deploy* N_m *sensors according to Algorithm 1 in the domain* $\mathcal{D}$.

To check the validity of relationship (3.21), we used it in the design of a distributed coverage control algorithm for mobile sensor networks. The objective is to deploy mobile sensors in a distributed manner such that the desired coverage level P_{des} is maintained. The steps for this scheme are outlined in Algorithm 6. We simulated deployment of mobile sensor networks for the same values of P_{des} and footprint areas A as before and the results for these simulations are presented in Figure (30). We were able to maintain P_{des} quite accurately, since average error was 0.6553% which is negligible. However, it is important to point out that for this algorithm, maximum error is more important because once sensors reached their final locations, the network is static. The maximum error for all the simulations we performed was 2.67%, which is still very small and indicates the accuracy of our modeling.

The first step in Algorithm 4 requires randomly distributed points. For the case of mobile sensor networks this can be achieved in different ways. One possible approach is to drop the sensors randomly from some vehicle driving through the domain $\mathcal{D}$ and this approach will result in a uniformly distributed configuration. The other approach

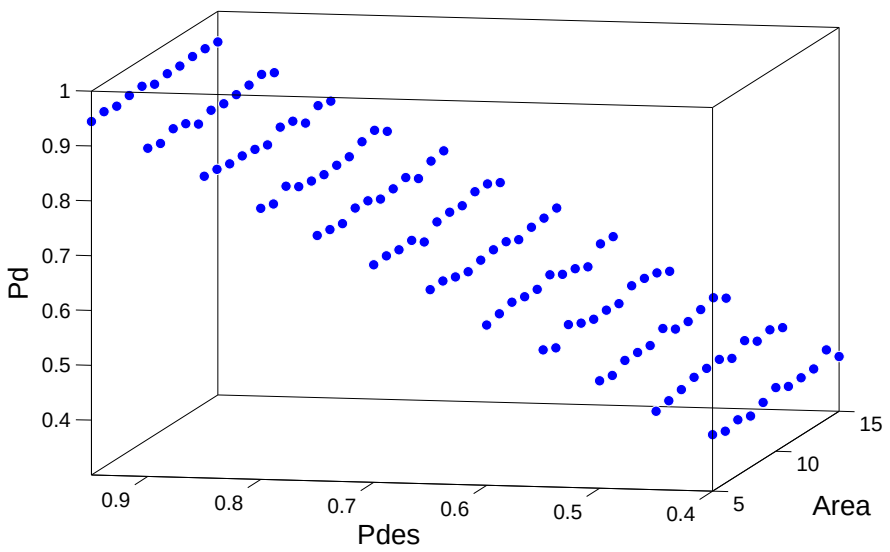


Figure 30: Detection Probability that is obtained after designing from our proposed model. Max Modeling Error 2.67%. Average Modeling Error 0.6553%

is that, all the sensors initially follow a random mobility model called *random direction model* presented in ([52] and [53]). In this model, each sensor randomly chooses its direction uniformly from $[0, 2\pi]$ and its velocity uniformly from $[v_{min}, v_{max}]$. Then it starts moving in the chosen direction at the chosen speed for a random time that is exponentially distributed. After that it selects a new direction and velocity and repeats the same process. It has been shown that this mobility model results in uniform spatial distribution of nodes. We can further improve the coverage properties of the network by increasing the separation between the sensors. To get this improvement in coverage, the underlying idea is that by ensuring separation d between the sensors, we have ensured P_{des}. Now for the same number of agents, if we increase the inter-agent separation from d to $d' = d + \epsilon$, for some $\epsilon > 0$, we can improve the level of coverage. However, this ϵ has to be bounded from above because of the constraints imposed by area of the domain and number of agents.

Lemma 3.7.1. *To maintain P_{des} in a rectangular domain,*

$$0 \leq \epsilon \leq \frac{1}{2d}\sqrt{\frac{A_{dom}}{4N_{P2}}}.$$

Proof. To maintain P_{des}, each agent has to maintain atleast distance d form its neigbors. Since $\epsilon < 0$ will result in $d' < d$ which violates the inhibition distance requirements, which proves the first inequality. For the second inequality, consider the case when domain has dimension [dim1 $\times$ dim2].For this case, the maximum number of balls of radius $\prime d$ that can fit in this region is

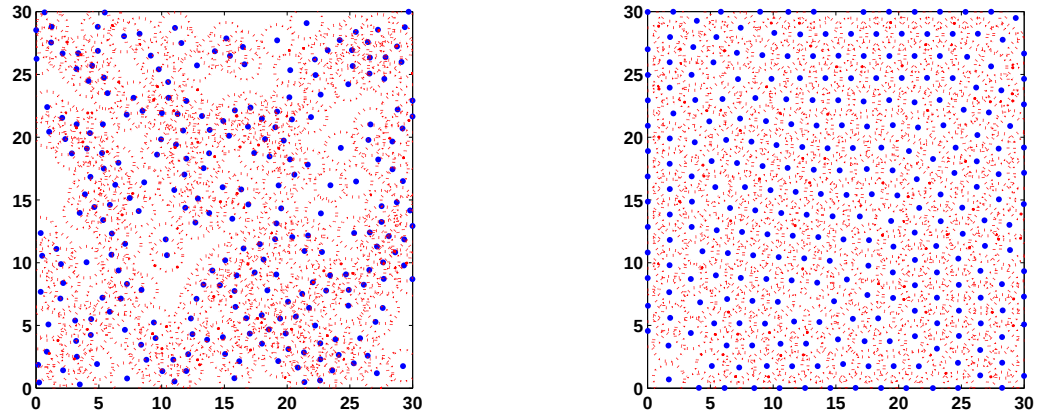
$$N = \frac{A_{dom}}{4d'^2}.$$

In this case, the number of balls are given which is N_{P2}. Thus, using simple algebraic manipulations we can find the maximum radius of each ball such that N_{P2} balls can fit in the domain, which concludes the proof. □

Figure (31) shows typical realization of a sensor network under the proposed algorithm for the case when $P_{des} = 0.8$, $A = 5$, and $dim = 30$. In Figure 31(a), there are $N_{P2} = 254$ sensors and $\epsilon = 0$, and for this case $P_d = 0.8011$ as compared to $P_{des} = 0.8$. To maximize the coverage that can be obtained from these sensors, Figure 31(b) corresponds to the case with $\epsilon = 0.7552$ which is the upper bound for ϵ in this case and we can see that $P_d = 0.9736$ which is almost complete coverage.

3.8 Conclusion

In this chapter we presented a novel sleep scheduling scheme for wireless sensor networks to conserve power while maintaining partial coverage. We introduced the concept of inhibition distance as the minimum distance allowed between the sensors in the on state and used it to control the number of redundant sensors in the on state over the entire domain of interest. To enforce the inhibition distance with little communication overhead, we used hard-core point processes from stochastic geometry



(a) $P_{des} = 0.8$, $P_d = 0.8011$, and $N_m = 254$ (b) $P_{des} = 0.8$, $P_d = 0.9736$, and $N_m = 254$

Figure 31: Deployment of a network under the proposed scheme with $P_{des} = 0.8$. In part(a) exact inhibition distance $d = 1.408$ is enforced while in part (b) distance enforced is $d + \epsilon$ where $\epsilon = 0.7552$.

and proposed a simple sleep scheduling scheme, which accomplished the task. Then, we considered one special case, in which the inhibition distance was twice the radius of the sensor footprint and derived an expression for the event detection probability in this particular case. For the generalized design of a system, we used Monte Carlo simulations to model Matérn II hard-core point process and using the data from these simulations we derived relationship between P_d and d. We showed through simulations that the proposed model accurately modeled a hard-core point process. Using this model, we designed energy efficient scheduling scheme and showed that the proposed scheme improved the lifetime of the network from 40% to 70%.

CHAPTER IV

POWER-AWARE MOBILITY STRATEGIES

4.1 Power-aware Rendezvous

In this section, we study the by-now classic rendezvous problem [13], i.e., the problem of having all the nodes meet in a common, a priori unspecified location using only relative position information. Our take on this problem is to use a sensor footprint model that depends on the current power levels and that shrinks as the power level decreases. Moreover, the rate at which the power level decreases is proportional to the input to the system. As a result, the more the agents move, the less power they have, and, subsequently, the smaller their sensor footprints become. The reason why a shrinking sensor footprint is important is that the agents can only sense the relative positions of other agents within their sensory range. This formulation of the power-aware mobility problem thus affords a natural formulation of the trade-offs between mobility and power consumption.

4.1.1 The Effect of Shrinking Footprints

To establish some of the implications that shrinking sensor footprints have on the performance of the coordination algorithms, consider a system in which each agent is a mobile sensing device that uses an omni-directional antenna for communication. We define the footprint of a sensor as the region in which a sensor can detect any event and can communicate with other sensors. The footprint of these sensors is a disk of radius Δ, and if we assume that Δ is fixed and same for all agents, we can represent the interaction topology with an undirected Δ-disk graph ([2] and [5]), in which an edge exists between two nodes if the distance between them is less than or equal to Δ. We will use the notation $\mathcal{N}_i$ to denote the index of all agents that are

inside the Δ-disk of agent i. Let $x_i \in \mathbb{R}^2$ be the location of each agent. Then we can solve the rendezvous problem using the consensus equation

$$\dot{x}_i = -\sum_{j \in N(i)} (x_i - x_j). \tag{4.1}$$

It is known from ([2], [10], [14], and [49]) that if the graph remains connected for all the time, then the dynamics in Equation (4.1) asymptotically drives all the agents to the initial centroid $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i(0)$.

Typically multi-agent systems comprise of agents that are battery powered and the available power level decreases with time. For RF- or radar-based sensors that are under investigation in this work, the area of the sensing region is directly related to the available power [46]. Since the available battery power decreases as a result of agents performing various tasks, the radius of the sensing disk also decreases, which results in sensors having shrinking footprints. What if we want to achieve rendezvous in such a system? Is it possible? How is the shrinking of the footprints going to effect the system? Is the consensus equation still helpful in this scenario? In spite of the importance of all these questions in many real life systems, they are still unanswered and in this work, we lay down a basic framework for the analysis of such systems.

For the sake of argument, assume (this assumption will be relaxed in later sections) that the radius of the footprint of a sensor decreases according to the following dynamics

$$\dot{\Delta}(t) = -\gamma\Delta(t), \tag{4.2}$$

with decay rate $\gamma > 0$. Since size of the footprint is directly related to power, the radius of the footprint of a sensor is assumed to decay exponentially as a result of power decay. Now, one can ask a question whether the linear consensus equation still solves the rendezvous problem or not? To explore this scenario, a situation involving two mobile sensors is demonstrated in Figure (32), in which we start with a connected graph, but the decaying power levels cause the footprints of both the agents to shrink,

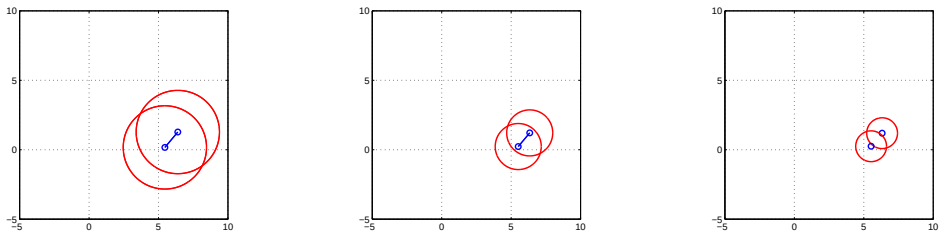


Figure 32: Consensus algorithm on a network of two nodes with shrinking footprints. The parameters used in this simulation are, $\gamma = 5$ and $\Delta(0) = 3$. The straight line between the nodes in the first two figures indicates that the nodes are connected. However, as a result of the shrinking footprints, the eventually gets disconnected in the last figure (no line between the two nodes).

and the distance between the two agents becomes greater than $\Delta(t)$. Consequently, the connection between the agents is lost and the fundamental condition for the convergence of consensus algorithm, i.e., connectivity, is violated. From this simple simulation, we can conclude that the controller in Equation (4.1) cannot guarantee to solve rendezvous problem.

Thus, in this work, we derive conditions that must be satisfied to achieve consensus in the case of shrinking footprints, and propose controllers to solve rendezvous problem for both undirected and directed graphs. We start with a simple system model in which power decay is only a function of power that is transmitted to maintain a required footprint, and has nothing to do with mobility. Later in this section, we generalize our model to include mobility as well.

4.1.2 The Weighted Consensus Equation

To solve the rendezvous problem for directed and undirected graphs, we know we can use the controller

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i} (x_i - x_j),$$

as long as the graph remains connected for all the time. Suppose we have N agents in $\mathbb{R}^2$, and let $x_i^T = (x_{i,1}, x_{i,2})$ be the location of agent i. Then we can produce a new

vector $c(x, j)$ as

$$c(x, j)^T = (x_{1,j}, x_{2,j}, \ldots, x_{N,j}).$$

If we decompose the problem along each dimension, the consensus controller in Equation (4.1) can be written as

$$\dot{c}(x, j) = -\mathcal{L}(\mathcal{G})c(x, j) \qquad j = \{1, 2\}, \tag{4.3}$$

where $\mathcal{L}(\mathcal{G})$ is the Laplacian matrix of the graph $\mathcal{G}$. However, for Δ-disk graphs (even without shrinking footprints), this condition on connectivity cannot be guaranteed, as is illustrated in [58]. One solution is to introduce edge weights $w(x_i, x_j)$ that turns the linear consensus equation into a non-linear equation

$$\dot{x}_i = -\sum_{j \in N(i)} w(x_i, x_j)(x_i - x_j), \tag{4.4}$$

where $w(x_i, x_j) : (\mathbb{R}^2, \mathbb{R}^2) \to \mathbb{R}$. The matrix form of the updated controller is

$$\dot{c}(x, j) = -\mathcal{L}_w c(x, j) \quad \forall j = \{1, 2\}, \tag{4.5}$$

where $\mathcal{L}_w = \mathcal{I}\mathcal{W}\mathcal{I}^T$ is the weighted Laplacian matrix. Moreover, $\mathcal{W}$ is a diagonal matrix whose entry w_{kk} is the weight of the k^{th} edge, and whose dimensions are $(M \times M)$, where M is the number of edges in the graph.

For proper selection of these weights, an edge tension energy along each edge $(v_i, v_j) \in E(\mathcal{G})$ was defined in [58] as

$$\begin{cases} \mathcal{E}_{ij}(x) > 0 \quad \text{if } (v_i, v_j) \in E(G) \text{ and } x_i \neq x_j; \\ \mathcal{E}_{ij}(x) = 0 \quad (v_i, v_j) \notin E(G). \end{cases}$$

Then the total energy of the system is

$$\mathcal{E}(x) = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{2} \mathcal{E}_{ij}(x). \tag{4.6}$$

In [58], it was proved that if the network topology is undirected, and is connected at $t = 0$, and if the edge tension energy is

$$\mathcal{E}_{ij}(x) = \frac{\|x_i - x_j\|^2}{\Delta - \|x_i - x_j\|}, \tag{4.7}$$

then the controller

$$\dot{x}_i = -\frac{\partial \mathcal{E}}{\partial x_i}^T = -\sum_{j \in N(i)} w_{i,j}(x_i, x_j)(x_i - x_j) \tag{4.8}$$

ensures that the graph remains connected for all the time and solves the rendezvous problem for Δ-disk proximity graphs for a time-invariant Δ.

4.1.3 Power-aware Rendezvous: Undirected Graph Topologies

The weighted consensus equation solves the rendezvous problem under a somewhat simplistic assumption. We have to assume that the radius of the sensor footprint, Δ, is the same for all the agents, and that the radius remains constant with time. In this section, we will relax this assumption by first incorporating a time varying Δ. However, this Δ will still be the same for all the agents. In the next section, we will further relax this assumption and consider directed graphs in which each agent can have different footprint radius.

Let us investigate undirected network topologies comprising of agents that use omni-directional RF or radar based antennas for sensing and communications and all the agents have the same footprint radius, Δ. For such systems, the size of the footprint $\Delta(t)$ decreases over time with the decrease in available power. We model this decrease as

$$\dot{\Delta} = f(\Delta), \tag{4.9}$$

where $f(\Delta)$ is a negative definite, Lipschitz continuous function, i.e.,

$$f(\Delta) < 0 \ \ \forall \ \ \Delta > 0 \qquad \text{and} \qquad f(0) = 0.$$

In Equation (4.9), we have assumed that power decay depends only on the power that is transmitted to maintain a footprint of radius Δ. This is an over-simplified power decay model because in any real life system, power decay must also be a function of mobility. However, being a first step in this direction, we have used a simplified model and a more detailed model is presented in the following sections.

Starting with the edge tension function that is presented in [58] (under the additional caveat that Δ is now a function of time), we get

$$\mathcal{E}_{ij}(x) = \frac{\|d_{ij}\|^2}{\Delta(t) - \|d_{ij}\|}, \tag{4.10}$$

where $d_{ij} = x_i - x_j$.

Following the procedure in [58], we have

$$\dot{\mathcal{E}}(t) = \frac{\partial \mathcal{E}}{\partial x}\dot{x} + N\frac{\partial \mathcal{E}}{\partial \Delta}\dot{\Delta}.$$

By replacing $\dot{x}_i$ in the above expression with the right hand side of Equation (4.8), we get

$$\dot{\mathcal{E}}(t) = -\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2 - N\left(\frac{\|d_{ij}\|^2}{(\Delta - \|d_{ij}\|)^2}\right) f(\Delta).$$

Now, we need to show that $\dot{\mathcal{E}}(t) < 0$, for all t. Since $f(\Delta)$ is negative definite, this means the second term in the above expression has to be greater than $\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2$ and it should be true for all possible $f(\Delta)$, which implies that the following should always hold.

$$|f(\Delta)| \leq \frac{2\Delta - \|d_{ij}\|}{(\Delta - \|d_{ij}\|)^2}.$$

The right side of the above expression is a function of distance between the agents and Δ while the left side is function of Δ only. Thus, we can not guarantee that the energy is always decreasing (which is the key to the convergence argument) for all negative definite $f(\Delta)$, which means that we need to modify our controller. .

Theorem 4.1.1. *Given an undirected Δ-disk graph $\mathcal{G}(V, E)$ that is connected at $t = 0$ in such a way that*

$$\|x_i(0) - x_j(0)\| < (\Delta(0) - \epsilon)$$

for some $\epsilon > 0$, and for all $(v_i, v_j) \in E(0)$, then under the control law

$$\dot{x}_i = -\left(\frac{\partial \mathcal{E}}{\partial x_i}^T + \frac{\partial \mathcal{E}}{\partial x_i}^T \frac{\partial \mathcal{E}}{\partial \Delta}\frac{\dot{\Delta}}{\left\|\frac{\partial \mathcal{E}}{\partial x_i}\right\|^2}\right), \tag{4.11}$$

the graph $\mathcal{G}(V, E)$ remains connected for all the time.

Proof. Consider the energy function in Equation (4.7). Then

$$\dot{\mathcal{E}} = \frac{\partial \mathcal{E}}{\partial x}\dot{x} + N\frac{\partial \mathcal{E}}{\partial \Delta}\dot{\Delta}.$$

Using $\dot{x}_i$ as defined in Equation (4.11), we obtain

$$\frac{\partial \mathcal{E}}{\partial x}\dot{x} = -\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2 - N\left(\frac{\partial \mathcal{E}}{\partial \Delta}\dot{\Delta}\right),$$

which results in

$$\dot{\mathcal{E}} = -\left\|\frac{\partial \mathcal{E}}{\partial x}\right\|^2 \leq 0 \qquad \forall t > 0. \tag{4.12}$$

The above equation indicates that the energy in the system is never increasing. Now, suppose on the contrary that there exists an edge (v_i, v_j) such that at some time $\hat{t}$ the corresponding length is $\Delta(\hat{t})$. We know that at $t = 0$ the edge length is less than $(\Delta(0) - \epsilon)$ and the total energy $\mathcal{E}(0)$ of the system defined in Equation (4.6) is finite. However, if at time $\hat{t}$, the edge length is equal to $\Delta(\hat{t})$, then the energy is $\mathcal{E}(\hat{t}) = \infty$, meaning that $\mathcal{E}(\hat{t}) > \mathcal{E}(0)$, which is a contradiction, and the lemma follows. □

Theorem 4.1.2. *Given an undirected Δ-disk graph that is connected at $t = 0$ with edge lengths less than $(\Delta(0) - \epsilon)$ for some $\epsilon > 0$. Under the controller in Equation (4.11), the system converges asymptotically to the initial centroid of the network.*

Proof. Rearranging the terms in Equation (4.11) yields

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i(t)} \left[1 + \frac{\partial \mathcal{E}}{\partial \Delta}\frac{\dot{\Delta}}{\left\|\frac{\partial \mathcal{E}}{\partial x_i}\right\|^2}\right] \frac{\partial \mathcal{E}_{ij}}{\partial x_i}^T,$$

where $\mathcal{N}_i(t)$ is the neighborhood set of agent i at time t. This expression can be formulated in the form of standard, weighted consensus equation

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i(t)} w_{ij}(x_i, x_j)\,(x_i - x_j)\,, \tag{4.13}$$

with

$$w_{ij}(t) = \left[1 + \frac{\partial \mathcal{E}}{\partial \Delta}\frac{\dot{\Delta}}{\left\|\frac{\partial \mathcal{E}}{\partial x_i}\right\|^2}\right] \frac{2\Delta - d_{ij}}{(\Delta - d_{ij})^2}. \tag{4.14}$$

We can also write this controller in terms of a weighted Laplacian.

$$c(x,j) = -\mathcal{L}_w c(x,j) \;\; \forall j = \{1,2\}. \tag{4.15}$$

We already know that this controller drives all the agents asymptotically to initial centroid as long as the network stays connected [58]. From Lemma 4.1.1, we know that connectivity is preserved and the proof follows. □

What this new controller allows us to do is to compensate for the effects of shrinking footprints at the control design phase by explicitly taking into account the footprint shrinkage model. In Figure (33), a MATLAB simulation of the controller in Equation (4.13) with weight function as in Equation (4.14) for an undirected graph is shown. However, in this section, we still assumed that the footprints of all the agents were the same. What remains to be done is to investigate what happens if they are no longer the same, which is the topic in the next section.

4.1.4 Power-aware Rendezvous: Directed Graph Topologies

In this section, we explore the case in which size of the footprint can vary among the agents, i.e., we no longer assume that $\Delta_i = \Delta_j$. However, for each agent, the footprint decay model is still the same, i.e.,

$$\dot{\Delta}_i = f(\Delta_i).$$

As a result, we have a directed graph because in this case, $\|d_{ij}\| \leq \Delta_j$ does not implies that $\|d_{ij}\| \leq \Delta_i$. Now, we need to show that our previously defined controller works for directed graph topologies as well.

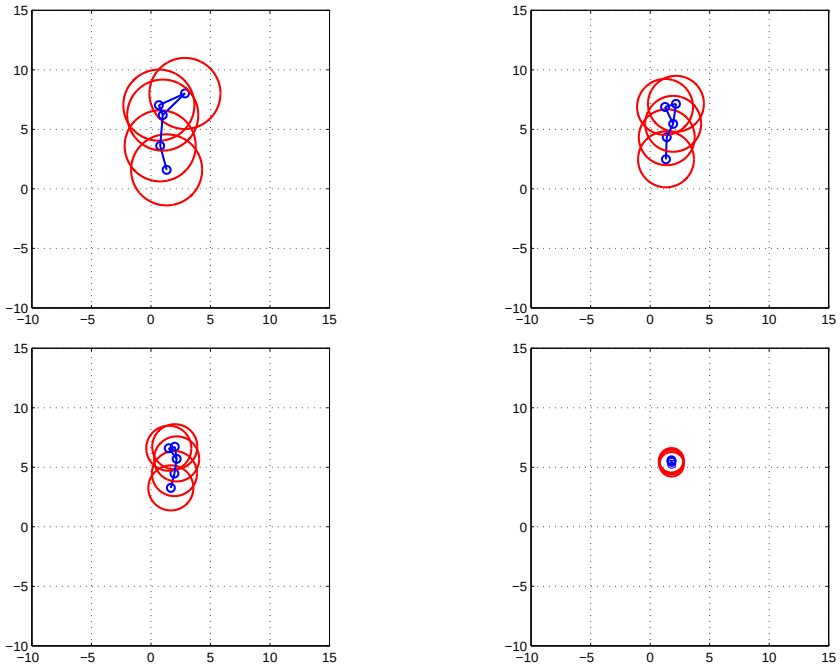


Figure 33: Demonstration of consensus algorithm of Equation (4.11) on an undirected network of 5 nodes with shrinking footprints, where the footprints shrink according to the decay law in Equation (4.2). In this simulation, $\gamma = 5$, $\Delta(0) = 3$, $t_1 = 0.0010$ sec, $t_2 = 0.0490$ sec, $t_3 = 0.0920$ sec, $t_4 = 0.2140$ sec.

Theorem 4.1.3. *Given a directed disk graph $\mathcal{G}(V, E)$ of N agents with $\bar{\Delta}(t)^T = [\Delta_1(t), \ldots, \Delta_N(t)]$. If the initial graph is balanced and weakly connected and the length of all the edges (v_i, v_j) at time $t = 0$ is less than $\min_i (\Delta_i(0) - \epsilon)$ for some $\epsilon > 0$, then the control law*

$$\dot{x}_i = -\sum_{j \in \mathcal{N}_i(0)} w_{ij}(x_i, x_j)\,(x_i - x_j)$$

with weight function w_{ij} given as

$$w_{ij}(t) = \left[1 + \frac{\partial \mathcal{E}}{\partial \Delta_i} \frac{\dot{\Delta}_i}{\left\| \frac{\partial \mathcal{E}}{\partial x_i} \right\|^2}\right] \frac{2\Delta_i - d_{ij}}{(\Delta_i - d_{ij})^2}.$$

makes the multi-agent system converge asymptotically to initial centroid, i.e., the rendezvous problem is solved

Proof. From the results proved in [5], we know that the agreement protocol $\dot{x} = -\mathcal{L}_w x$

over directed graph reaches average consensus if and only if the directed graph is weakly connected and balanced. We have already shown in the previous section that we can write our controller in terms of a weighted Laplacian (4.15), and also that this controller does not loose any edges. Moreover, we have restricted the neighborhood set to be equal to the initial neighborhood set $\mathcal{N}_i(0)$, which implies that if we start with a balanced and weakly connected graph, then these characteristic will be maintained and all the agents will drive asymptotically to initial centroid, which concludes the proof. □

This means that we have indeed solved the rendezvous problem for directed graphs with shrinking footprints, which is demonstrated in Figure (34).

If we thoroughly analyze the weight function in Equation (4.14) for undirected graphs, we find a potential problem. Consider two agents, i and j that are not neighbours at time $t = 0$. Suppose at time $t = t_1$, the separation between these agents is exactly $\Delta(t_1)$ and because of Δ-disk graph topology, these agents immediately become neighbours with the edge tension energy as is given in Equation (4.7). However, at this moment $\|d_{ij}\| = \Delta(t_1)$, which results in infinite edge tension energy. To avoid this situation, one simple solution is to introduce hysteresis by defining a function h_{ij} such that

$$h_{ij} = \begin{cases} 1, & \|d_{ij}\| \leq (\Delta - \epsilon); \\ 0 & otherwise. \end{cases}$$

Adding this aspect to Equation (4.8), we get the following controller

$$\dot{x}_i = -\sum_{j=1}^{N} h_{ij} w_{ij} \left(x_i - x_j\right). \tag{4.16}$$

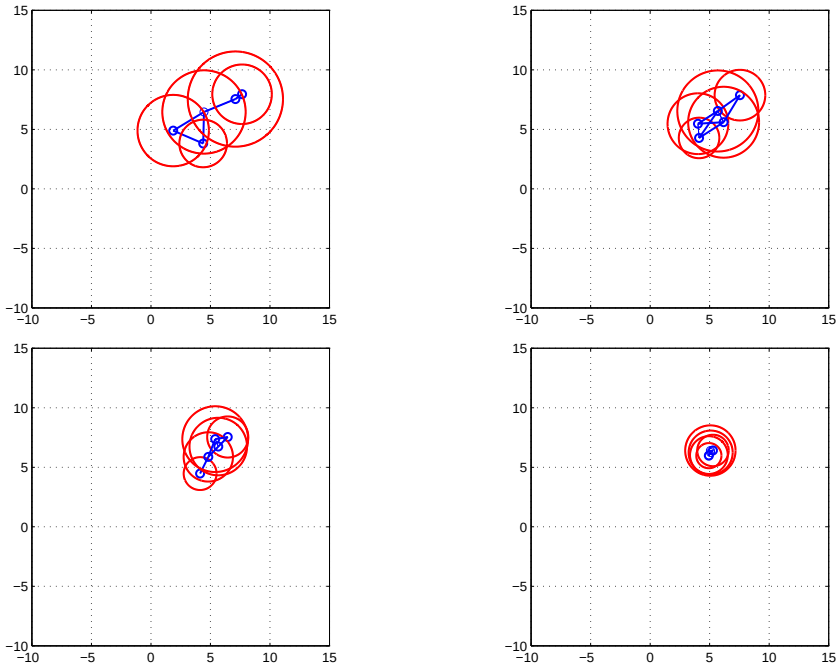


Figure 34: Demonstration of the consensus algorithm of Equation (4.11) on a directed network of 5 nodes with shrinking footprints where the footprints shrink according to the decay law in Equation (4.2). In this simulation, $\gamma = 5$, $\bar{\Delta}(0) = [2, 2.5, 3, 3.5, 4]$, $t_1 = 0.0010$ sec, $t_2 = 0.0323$ sec, $t_3 = 0.0741$ sec, $t_4 = 0.1275$ sec.

4.2 Power-aware Rendezvous for MoPS Agents

4.2.1 Power, Sensing, and Mobility Models

In the previous section, we assumed a model in which available power decreases as a function of the radius of the footprint. This model is simplistic since it does not assign any cost to mobility, which consumes a major portion of the power in most of the systems. In this section, we propose a detailed model that includes both mobility and power transmission and again investigate rendezvous problem under this model.

Consider a network of N planar, mobile agents, with positions $x_1, \ldots, x_N \in \mathbb{R}^2$. We assume that the dynamics of each of these agents is given by a single integrator, i.e,

$$\dot{x}_i = u_i, \;\; i = 1, \ldots, N. \tag{4.17}$$

Now, each of the agents has a corresponding non-negative power level $p_i \in \mathbb{R}_+, \;\; i =$

$1, \ldots, N$, and as the agents move around, this power level is depleted. In this work, we simply assume a direct proportional decay rate

$$\dot{p}_i = -c\|u_i\|, \tag{4.18}$$

where $c > 0$ is the power loss coefficient [44]. It should be noted that much more elaborate power loss models can be constructed (see e.g., [50], [56], and [55]), but for the purpose of the initial developments in this work, we stick with this first order model.

The way we tie the effect of the power levels to what the agents can do is by relating the power levels to the sensor footprints. In fact, the way the agents interact with each other is by measuring their displacements relative to neighboring agents, i.e., agents that are in their sensor ranges. In other words, if we let Δ_i be the sensor range associated with agent i and if $\|x_i - x_j\| \leq \Delta_i$, i.e., agent j is within the sensory range of agent i, then u_i is allowed to depend on the relative displacement $x_i - x_j$, which we assume is what the agents can in fact measure. Tallying up the contribution from all agents within sensory range of agent i, the control law in Equation (4.17) is assumed to be of the form

$$u_i = \sum_{j|\|x_i - x_j\| \leq \Delta_i} f(x_i - x_j)\sigma_{ij}, \tag{4.19}$$

for some interaction law $f : \mathbb{R}^2 \to \mathbb{R}^2$. Here $\sigma_{ij} \in \{0, 1\}$ is an indicator function that dictates whether or not agent j should be effecting the movement of agent i. (To paraphrase [5], "just because you're neighbors doesn't mean you're friends.") Note, by letting $\sigma_{ij} = 1$ and $f(x_i - x_j) = x_j - x_i$, we recover the standard consensus equation ([2], [10], and [14]).

The final part of the construction relates the power levels to the effective sensor footprint. This connection depends on what type of sensor is used and, for the purpose of the discussion in this work, we follow the model developed in [46], as mentioned in the previous sections. In this case, the sensor range model is based on the RF power

density function for an isotropic antenna with the sensor footprint being proportional to the available power of the sensor node. Since the footprint (disk) is quadratic in the sensor range (radius), we get that

$$\Delta_i^2 = \gamma_i p_i, \tag{4.20}$$

where $\gamma_i > 0,\ i = \{1, \ldots, N\}$, is a constant that depends on various factors such as the transmission medium and source. Putting all of these individual components together, we obtain the main object of study in this work, namely an agent model that we choose to call a **MoPS** (**Mo**bility, **P**ower, and **S**ensing) agent.

Definition 4.2.1 (MoPS Agent)**.** *A* MoPS *agent is a first-order* **Mo***bility,* **P***ower, and* **S***ensing agent, whose dynamics are given in Equation (4.17), whose power decay is given in Equation (4.18), whose sensory footprint is given by Equation (4.20), and whose control law satisfies the restrictions given in Equation (4.19).*

The key question under investigation is what effects the shrinking footprints have on the performance of the agent team. For instance, if two agents are to meet at a common location, they need to be "visible" to each other (or at least one of them to the other). Even though they may be within the sensing range of each other initially, by moving around, the power consumption may cause the agents to lose track of each other since the sensor ranges may become too small. This is also an issue in more elaborate cooperative control scenarios and what is needed is a systematic approach to designing coordinated controllers that take power consumption into account already at the design stage.

4.2.2 Power-aware Rendezvous: Network of Two Agents

As already stated, the rendezvous problem involves moving a collection of mobile agents to the same spatial location. Moreover, this task should be accomplished with only local information given in terms of the relative inter-agent displacements.

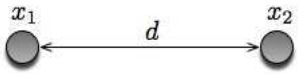


Figure 35: Rendezvous between two MoPS agents.

To start the discussion, we first study the rendezvous problem for a pair of MoPS agents, and in particular, we investigate the implications in terms of power consumption. We make two additional, simplifying assumptions about the two agents, depicted in Figure (35), namely (1) that they do not act stupidly, in the sense that they do indeed move towards each other, and (2) that they act symmetrically and have the same initial power levels and power decay rates. A consequence of the first assumption is that we can restrict the problem to a one-dimensional problem in which the agents are moving on the line between them. The second assumption implies that the two agents are executing the same anti-symmetric control strategy in that

$$\dot{x}_1 = f(x_1 - x_2) = -f(x_2 - x_1) = \dot{x}_2,$$

where f is the particular control strategy used.

If we assume, again, without loss of generality, that $x_1, x_2 \in \mathbb{R}$ and that $x_1 \leq x_2$, we can let u constitute the applied control action, in the sense that

$$\dot{x}_1 = -u, \ \dot{x}_2 = u,$$

where is u is some scalar. Under this formulation, with the assumption that the agents do not act stupidly, we immediately see that $u \leq 0$. As a consequence, we get that the distance between the agents, $d = x_2 - x_1$, has the dynamics

$$\dot{d} = 2u, \tag{4.21}$$

with solution

$$d(t) = d_0 + 2\int_0^t u(s)ds = d_0 - 2\int_0^t |u(s)|ds = d_0 - 2\mathcal{U}_t, \tag{4.22}$$

where d_0 is the initial distance between the two agents, and where $\mathcal{U}_t = \int_0^t |u(s)|ds$ is the total control energy used by an agent over the interval $[0, t]$.

Since we assume that the agents act symmetrically and have the same initial power levels, we can use $p(t)$ to denote this level, which satisfies

$$\dot{p} = cu. \tag{4.23}$$

In light of Definition 4.2.1, we can observe that for rendezvous to be successfully executed by these two agents, they need to be able to sense each other, i.e., we need to ensure that

$$d^2(t) \leq \gamma p(t) \tag{4.24}$$

throughout the duration of the movement. We let $e(t)$ denote the power gap, i.e.,

$$e(t) = \gamma p(t) - d^2(t), \tag{4.25}$$

with the interpretation that $e(t) \geq 0$ implies that the agents can sense each other while $e(t) < 0$ implies that they are not within the range of each other. One natural question to ask now is how much control energy can be injected into the system without rendering e negative, i.e., without causing the underlying interaction network to become disconnected.

Lemma 4.2.1. *The maximum energy that can be injected into a two MoPS system, whose initial separation and power satisfies $\gamma p_0 - d_0^2 > 0$, over a given time interval $[0, t]$, without rendering the underlying network disconnected is*

$$\mathcal{U}_t^\star = \frac{d_0 - c\gamma/4}{2} + \sqrt{\left(\frac{d_0 - c\gamma/4}{2}\right)^2 + e_0}, \tag{4.26}$$

where $e_0 = \gamma p_0 - d_0^2$.

Proof. The solution to Equation (4.23) is given by

$$p(t) = p_0 - c\mathcal{U}_t. \tag{4.27}$$

Replacing the expressions for $d(t)$ and $p(t)$ in Equation (4.25) with the explicit solutions for p and d yields,

$$e(t) = \gamma(p_0 - c\mathcal{U}_t) - (d_0 - 2\mathcal{U})^2.$$

To find the maximum energy that can be injected while maintaining connectivity, we need to put $e(t) = 0$ and solve the resulting quadratic equation to find

$$\mathcal{U}_t^\star = \frac{d_0 - c\gamma/4}{2} + \sqrt{\left(\frac{d_0 - c\gamma/4}{2}\right)^2 + e_0},$$

and the proof follows. □

Now, we need to relate this maximal energy injection to the achievement of rendezvous. In particular, we need to ensure that if the agents move as much as they possibly can without causing the network to get disconnected, they do in fact end up at the same location. The subsequent theorem establishes conditions on the initial power level that ensures that this is in fact achievable.

Theorem 4.2.2. *For a two MoPS system, rendezvous can be achieved if*

$$p_0 \geq \frac{6d_0^2 + \gamma c d_0}{8\gamma}, \tag{4.28}$$

Proof. At any time, the distance between the two agents is given by Equation (4.22). If rendezvous is achieved at time t, then we have $d(t) = 0$, which, in the light of Equation (4.22) implies that

$$d_0 = 2\mathcal{U}_t. \tag{4.29}$$

Replacing $\mathcal{U}_t$ by $\mathcal{U}_t^\star$ in the above expression yields the most restrictive conditions when rendezvous can in fact be achieved and straightforward algebraic manipulation generates the condition on p_0 that

$$p_0 = \frac{6d_0^2 + \gamma c d_0}{8\gamma}.$$

Since $\mathcal{U}_t^\star$ is the maximum energy that can be injected without loosing connectivity, which suggests that for any $U_t \leq \mathcal{U}_t^\star$, the inequality in (4.28) is satisfied. □

One consequence of Lemma 4.2.1 and Theorem 4.2.2 is that for a system of two MoPS agents in which the rendezvous problem is reduced to a one-dimensional problem, the type of the controller or the time needed to solve the problem, does not matter. The only thing that matters is the total energy, $\mathcal{U}_t$, supplied to the system, which depends completely on the initial conditions. For example, if we want to achieve rendezvous in T time units and the condition in Equation (4.28) is satisfied, then a constant u given by

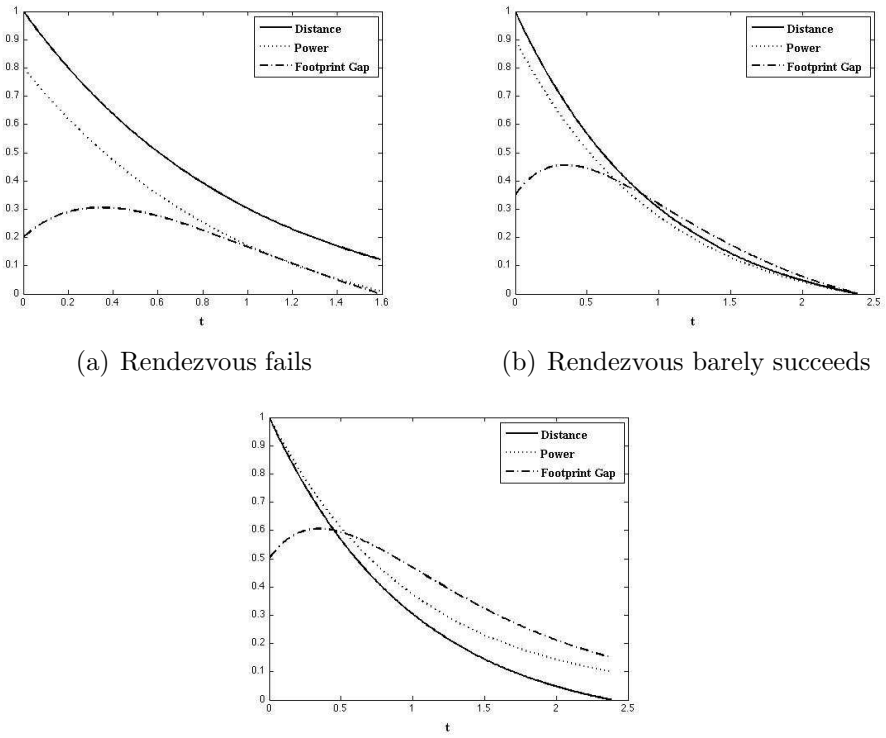
$$u(t) = -\frac{1}{T}\mathcal{U}_T^\star$$

will solve the problem. This controller is used in Figure (36), where three different initial power levels are used. In fact, we let

$$p_0 = \frac{6d_0^2 + \gamma c d_0}{8\gamma} + \epsilon.$$

In the left figure, $\epsilon < 0$, which results in loss of connectivity before rendezvous is achieved. In the middle figure $\epsilon = 0$ and rendezvous is achieved exactly at the time when footprint becomes zero, and in the right figure $\epsilon > 0$ and rendezvous is achieved with power left over.

From the above observations, one would be tempted to draw the conclusion that the condition in Equation (4.28) is not only sufficient but also necessary. It is in fact also necessary under the assumption that connectivity is maintained. But, the agents may, by pure luck or in some other open-loop fashion, still be able to achieve rendezvous despite not being able to "see" each other, which is why we formulate this condition as a sufficient but not necessary condition. The rather surprising fact that the actual control law does not matter in this case is of course not true in general. If there are more than two agents (as we will see in the next section) or if the dynamics is double integrator rather than single integrator, we are no longer this fortunate.



(a) Rendezvous fails

(b) Rendezvous barely succeeds

(c) Rendezvous succeeds

Figure 36: Depicted are the distance between agents (solid), the available power (dotted), and $e(t)$, i.e., the power gap (dashed-dotted) which corresponds to how close agents are to becoming disconnected. In the left figure, $e = 0$ before rendezvous is achieved. In the middle figure, rendezvous is achieved at the very moment when e becomes zero. In the right figure, rendezvous is achieved with $e > 0$.

4.2.3 Power-aware Rendezvous: Directed Cycle Topologies

In this section, we consider a more involved situation in which we have a network of N MoPS agents. We assume that the interaction topology, i.e., the underlying graph that dictates the information flow, is given by a directed cycle that remains static throughout the motion [57]. The number of agents, N, is greater than 2 and every agent $i \in \{1, \ldots, N\}$, with position x_i is connected to $(i+1)$-th agent at position x_{i+1} (modulo N), as shown in Figure (37). Moreover, we assume that all the agents have the same initial power levels.

Since the graph representing the system is balanced and has a rooted out branching, under the standard consensus algorithm, all the agents will meet at $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i(0)$,

i.e., centroid of their initial positions. However, for consensus algorithm to work in the presence of decaying power levels, the graph must remain connected, which, from Equation (4.24), implies that

$$\|x_i(t) - x_{i+1}(t)\|^2 = d_i^2(t) \leq \gamma_i p_i(t), \tag{4.30}$$

for all time t. Here $p_i(t)$ is the power level of agent i at time t and $d_i(t)$ is the distance between agents i and $i+1$.

Since we can no longer hope for a situation where the results do not depend on the particular control laws we use, we choose to work with the consensus equation over a directed cycle, i.e., the interaction law executed by the MoPS agents in this system is given by

$$u_i = k(x_{i+1} - x_i), \tag{4.31}$$

where $k > 0$ is a constant. Using the notation from Equation (4.19), $f(x_i - x_j) = k(x_j - x_i)$ and $\sigma_{ij} = 1 \Leftrightarrow j = i+1$ (modulo N) .

The overall system can be written as

$$\dot{x} = Ax,$$

where A is an $N \times N$ circulant matrix,

$$A = \begin{pmatrix} -k & k & 0 & \cdots & \cdots & 0 \\ 0 & -k & k & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & -k & k \\ k & 0 & \cdots & \cdots & 0 & -k \end{pmatrix}$$

The following result characterizes a sufficient condition that ensures that rendezvous is achieved in the sense of all agents being within an ϵ distance of the initial

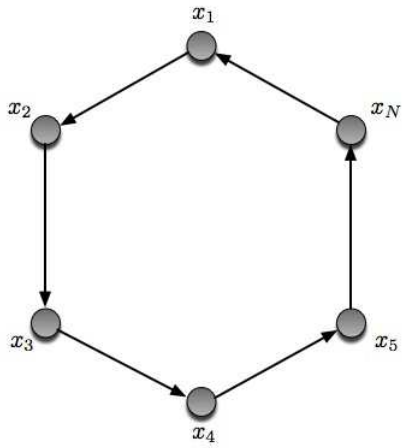


Figure 37: System Model for Directed Cycle Graph

centroid without rendering the network disconnected. This result is slightly more involved than the one in the previous section because of the fact that the network is more complex.

Theorem 4.2.3. *For a system consisting of N MoPS agents arranged in a directed cycle topology and executing the control law (4.31), suppose that:*

1. *the power loss coefficient c_i satisfies*

$$c_i \leq \frac{2\sqrt{2}\lambda e^{-\lambda T(\epsilon)}}{k\gamma_i} \max_{i\in\{1,\ldots,n\}} \|x_i(0) - \bar{x}\|, \tag{4.32}$$

   where

$$T(\epsilon) := \frac{1}{\lambda} \max_{i\in\{1,\ldots,n\}} \ln\left(\frac{\|x_i(0) - \bar{x}\|}{\epsilon}\right), \tag{4.33}$$

   and λ is the real component of the second largest eigenvalue of A, which is a function of k, and

2. *the initial power levels satisfy*

$$p_i(0) \geq \frac{2}{\gamma_i} \max_{i\in\{1,\ldots,n\}} \|x_i(0) - \bar{x}\|^2. \tag{4.34}$$

Then, the connectivity constraint (4.30) *holds at all times* $t \in [0, T(\epsilon)]$*, and all agents are within a distance* ϵ *of* $\bar{x}$ *at time* $t = T(\epsilon)$*.*

Proof. Recall that without the connectivity constraint, the control law (4.31) ensures exponential convergence of all the agents at the centroid ([14], and [49]). Thus, there exists a constant $\lambda > 0$ (the real component of the second largest eigenvalue of A), so that for every $i \in \{1, \ldots, N\}$,

$$\|x_i(t) - \bar{x}\| \leq \|x_i(0) - \bar{x}\| e^{-\lambda t} \tag{4.35}$$

Therefore, given an $\epsilon > 0$, after the time $T(\epsilon)$ satisfying Equation (4.33), all the agents are within a distance ϵ of the centroid of the initial locations.
It now remains to show that under the conditions of this proposition, the connectivity constraint (4.30) holds at all times $t \in [0, T(\epsilon)]$. Now,

$$\begin{aligned} d_i^2 &= \|x_i - x_{i+1}\|^2 = \|x_i - \bar{x} - (x_{i+1} - \bar{x})\|^2, \\ &\leq \|x_i - \bar{x}\|^2 + \|x_{i+1} - \bar{x}\|^2, \end{aligned}$$

Using (4.35)

$$d_i^2 \leq 2 \max_i \|x_i(0) - \bar{x}\|^2 e^{-\lambda t}. \tag{4.36}$$

Assuming Equation (4.32) holds then from Equation (4.36),

$$d_i^2 \leq \frac{16\lambda^2 e^{-4\lambda t}}{\gamma_i^2 c_i^2 k^2} \max_i \|x_i(0) - \bar{x}\|^4.$$

From the power decay model (4.18), the control law (4.31) and the above inequality we get

$$\dot{p}_i \geq -\frac{4\lambda e^{-2\lambda t}}{\gamma_i} \max_i \|x_i(0) - \bar{x}\|^2,$$

After integrating the above inequality and using the condition (4.34)

$$p_i(t) \geq \frac{2e^{-2\lambda t}}{\gamma_i} \max_i \|x_i(0) - \bar{x}\|^2,$$

Comparing this with Equation (4.36) results in

$$p_i(t) \geq \frac{d_i^2}{\gamma_i}.$$

which proves that the connectivity constraint holds at all times $t \in [0, T(\epsilon)]$, thus proving the theorem.

□

A couple of observations can be made from Theorem 4.2.3. Firstly, to satisfy the condition on power loss coefficient (4.32), each agent needs to estimate the maximum initial distance of the centroid $\bar{x}$ from an agent. Secondly, keeping all the other parameters fixed, as ϵ becomes smaller, $T(\epsilon)$ becomes larger, and therefore the condition (4.32) implies that c_i needs to be smaller. This is intuitive because with a longer time to rendezvous, each agent is expected to spend more power, and therefore, the power-loss coefficient must be smaller.

4.3 Distributed Framework for Energy-efficient Mobility Controllers

Energy-efficient mobile sensor networks have been investigated extensively in the past few years, where one of the main issues is how to use mobility to reduce the energy required for sensing and communications; see [81] and references therein. Most of these papers either ignore the cost of energy needed for mobility of the sensors, or assume unlimited sources of such energy. However, lately the question of balancing the energy costs of mobility and communication has begun to attract attention [83, 82, 76].

The development of inexpensive mobile, wireless sensing devices in the past few years (e.g., [60, 61]) has suggested the eventual massive deployment of mobile sensor networks in communication and control applications [76]. In many such applications the devices (agents) are tasked with transmitting data from one or more source objects to a remote station (controller), and to this end they have to arrange themselves in a network configuration. However, the agents often are powered by on-board,

limited-energy sources such as batteries, which cannot be replenished during the application's lifetime. Therefore, the network has to be configured in a way that balances, optimally, the energy required for communication and mobility.

The problem of optimizing mobility/communication energy tradeoffs has been formulated only recently; please see [83, 82, 76] for surveys. Reference [82] considers a robot tasked with transmitting a given number of bits while in motion on a predetermined trajectory with variable degrees of channel fading. Using a realistic, detailed, probabilistic model of the channel's fading, that paper determines the robot's speed, transmission rate, and stopping times that minimize the total energy required for mobility and communication. Reference [76] considers the task of distributing wireless mobile agents so as to provide transmission of sensor data from one or more objects to a remote station, and doing it in a way that minimizes the total required energy. That paper uses graph-theoretic techniques to compute an optimal strategy comprised of the sequential scheduling of mobility followed by transmission. The problem considered in this paper is stated in similar terms in an abstract context, but it is defined in the dynamic setting of optimal control, where the agents carry out their communication tasks while in motion. Furthermore, this paper is fundamentally different from [76] in several other ways, as will be made clear in the sequel.

Consider a scenario in which a supervisory controller instructs a collection of wireless mobile agents to form a tandem, point-to-point connection for transporting sensory data from a given object to a remote station (controller). Sensing and communication must commence immediately and be maintained for a given amount of time. Meanwhile the agents are arranging themselves dynamically in a network configuration where each one of them acts as a relay between a single downstream node and a single upstream node, and they determine their trajectories in a way that minimizes the energy spent on both communication and motion. The power required for transmission on a link is related to the link's length, and the motion energy is

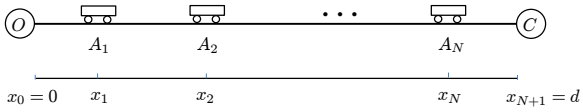


Figure 38: Tandem network

related to the distance traveled. We define the problem in the setting of optimal control, and devise a highly-efficient algorithm for its solution, that lends itself to a natural distributed implementation.

The above scenario is fairly generic, and although the power and energy models that we use are similar to those in [76], our primary concern is not with specific application problems. Instead, our objective is to propose a general-purpose computational framework for a general problem formulation and highlight its salient points via analysis and simulation. We start our investigation for this problem for a simple case of a single stationary object, a stationary remote station, and a one-dimensional movement of the agents. Later on we will generalize our results for a complete class of systems.

4.3.1 Problem Formulation

Consider the network shown in Figure (38), consisting of N mobile agents, $A_1, \ldots, A_N$, moving between an object and a remote controller station, denoted by O and C, respectively. Let $x_k(t)$, $k = 1, \ldots, N$, denote the relative position of A_k with respect to the object, and let d denote the relative position of the controller with respect to the object. Since we only consider motion in the line adjoining O to C, we have that $x_k(t) \in R$ and $d \in R$ as well. To simplify the notation we define $x_0 = 0$ and $x_{N+1} = d$, and we note that these are the positions of O and C; assuming that both the object and the controller station are stationary, x_0 and x_{N+1} are constants and not functions of time. We define the vector notation $x(t) := (x_1(t), \ldots, x_N(t))^\top \in R^N$ to denote the position of the agents, and assume that $x(0)$ is given and fixed. Furthermore, we define $u(t) = (u_1(t), \ldots, u_N(t))^\top \in R^N$ to be the vector of velocities of the agents,

namely

$$\dot{x} \;=\; u, \tag{4.37}$$

where the notational dependence on t is suppressed. The problem that we consider is to determine the control $u(t)$ and related state trajectory $x(t)$ (via (4.37)) for a given time-interval $t \in [0, t_f]$, in a way that minimizes a weighted sum of the agents' transmission energy and communication energy, subject to amplitude constraints on the controls.

The power required for moving an agent is proportional to its speed [76], and hence the associated performance-functional term is $J_{mobility} := \sum_{k=1}^{N} \int_0^{t_f} |u_k(t)|dt$. For the transmission energy cost, let $\psi(z) : R^+ \to R^+$ be a non-decreasing, continuously-differentiable function representing the transmission power of each agent over a link of length z. Commonly $\psi(z) = a+bz^2$ for given constants $a \geq 0$ and $b > 0$ [76], but we consider a more general function ψ. Note that the transmission down the line is from A_n to A_{n+1}, $n = 0, \ldots, N$, and hence the total transmission energy can be represented by the cost functional $J_{trans} := \sum_{k=1}^{N+1} \int_0^{t_f} \big(\psi(x_n(t) - x_{n-1}(t)\big)dt$. The performance function that we consider is a weighted sum of $J_{mobility}$ and J_{trans}, namely, for a given $C > 0$,

$$\begin{aligned} J = \sum_{k=1}^{N+1} \int_0^{t_f} \psi\big(x_k(t) - x_{k-1}(t)\big)dt \\ +C \sum_{k=1}^{N} \int_0^{t_f} |u_k(t)|dt. \end{aligned} \tag{4.38}$$

The constraints that we consider are $|u_k(t)| \leq 1$ for every $k = 1, \ldots, N$ and for all $t \in [0, t_f]$. The problem that we solve is to minimize J subject to these constraints.

Let us denote by $p(t) = (p_1(t), \ldots, p_N(t))^\top \in R^N$ the costate (adjoint) variable. Then by Equation (4.37), the costate equation is

$$\dot{p}_k \;=\; \frac{d\psi}{dx}(x_{k+1} - x_k) - \frac{d\psi}{dx}(x_k - x_{k-1}), \tag{4.39}$$

$k = 1, \ldots, N$, with the boundary condition $p_k(t_f) = 0$. The Hamiltonian has the following form,

$$H(x, u, p) \;=\; \sum_{k=1}^{N} \big(p_k u_k + C|u_k|\big) + \sum_{k=1}^{N+1} \psi_k(x_k - x_{k-1}), \tag{4.40}$$

and this particular form is especially suitable for the algorithm that we used for solving this problem.

The algorithm that we used, presented in [43], is a descent technique whose direction is computed by minimizing the Hamiltonian at each time t. Obviously this is often impossible, and hence impractical in the general setting of optimal control, but the special structure of our problem makes it possible and even simple, and hence yields effective descent directions. The step size of this algorithm is determined via the Armijo procedure [85, 84] which, though having linear asymptotic convergence, often has the practical advantage of rapid progress at the initial phases of the algorithm's runs. This point, demonstrated via simulations, is argued in [43] to suggest the eventual use of the algorithm in real-time tuning of the agents' trajectories.

We next describe the algorithm that we used for the general optimal control problem described above.

For every $\lambda \geq 0$, define the function $\tilde{J}_{u,w}(\lambda) : R^+ \to R$ as

$$\tilde{J}_{u,w}(\lambda) \;=\; J\big((u + \lambda(w - u)\big). \tag{4.41}$$

Given a control u, denote by $T(u)$ the next iteration point that the algorithm computes from u, and thus, $u_{i+1} = T(u_i)$. The formal computation of $T(u)$, for a given control u, is as follows.

Algorithm 1. *Parameters:* $\alpha \in (0, 1)$, $\beta \in (0, 1)$.

Step 1: Compute the state trajectory $x(t)$ *and costate trajectory* $p(t)$*, associated with the control* u*.*

Step 2: Compute a control $w := \big(w(t)\big)\big|_{t\in[0,t_f]}$ *such that* $w(t) \in \arg\min\big\{H(x(t), \cdot, p(t))\big\} \forall\, t \in [0, t_f]$.

Step 3: Compute

$$k'(u) := \min\left\{k' = 0, 1, \ldots, : \tilde{J}_{u,w}(\beta^{k'}) - \tilde{J}_{u,w}(0) \leq \alpha\beta^{k'}\theta(u)\right\}. \quad (4.42)$$

Step 4: Set the step size to be $\lambda(u) := \beta^{k'(u)}$*, and set* $T(u) = u + \lambda(u)(w - u)$.

4.3.2 Simulation Examples

In this section we present results of the application of Algorithm 1 to the problem described in Section II. We start with the simplest problem where $N = 2$ in order to highlight some properties of the algorithm, and then we present a more complicated case where $N = 6$.

The case of two agents The problem in question is to minimize J as defined in (4.38) subject to the dynamics in (4.37) and the constraints $|u_i(t)| \leq 1$. The distance of the object from the controller is $d = 10$, and the final time is $t_f = 20$. The transmission power over a link of length z is $\psi(z) = z^2$ and hence (see Equation (4.38)) $\psi(x_k - x_{k-1}) = (x_k - x_{k-1})^2$, and the constant C in (4.38) is $C = 7$. The initial condition for the state equation (4.37) is $x(0) = (1, 9)^\top$ for every control u. Algorithm 1 was used with $\alpha = \beta = 0.5$.

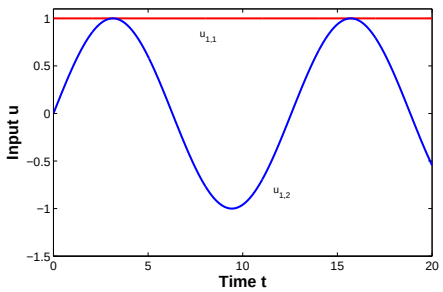
This problem can be solved analytically due to the particular form of the function $\psi(z)$, but we use the algorithm in order to examine its performance. The algorithm was run for 200 iterations computing, recursively, controls u_i, $u = 1, \ldots, 200$. Note that each control is two dimensional, $u_i = (u_{i,1}, u_{i,2})^\top$, and we chose, arbitrarily, the initial control to be $u_{1,1}(t) = 1.0$ and $u_{1,2}(t) = sin(t/2)$. We used a uniform grid overlaying the time-interval $[0, t_f]$ with $\Delta t = 0.01$, for the various computations in Equations (4.37)-(4.40), and for the differential equations we used the forward Euler

method. The minimizer of the Hamiltonian in (4.40) is known to be

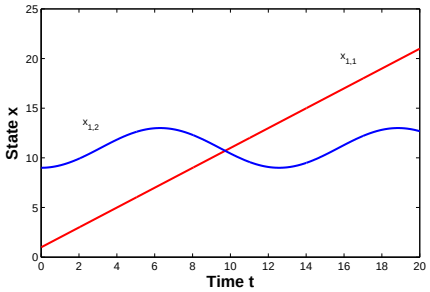
$$w_k(t) = \begin{cases} -1, & p_k(t) > C \\ 0, & -C < p_k(t) < C \\ 1, & p_k(t) > -C, \end{cases} \tag{4.43}$$

The results of this simulation are shown in Figure 39. Part (a) depicts the graphs of the two components of the initial control, $u_{1,1}(t)$ and $u_{1,2}(t)$, while part (b) shows the corresponding state trajectories, $x_{1,1}(t)$ and $x_{1,2}(t)$. This control obviously is far off the optimum; the associated cost is $J(u_1) = 4,012$ and the value of the optimality function is $\theta(u_1) = -14,910$, where $\theta(u) \leq 0$ is the optimality function for every control u, and if $\theta(u) = 0$ then u satisfies the maximum principle [86]. After 200 iterations, the graphs of the controls $u_{200,1}(t)$ and $u_{200,2}(t)$ are shown in part (c) of the figure, while the graphs of their corresponding state trajectories, $x_{200,1}(t)$ and $x_{200,2}(t)$, are depicted in part (d). These graphs show quite clearly that the behavior of the control u_{200} is compliant with Equation (13) which indicates a bang-off-bang control. The associated cost is $J(u_{200}) = 724.5$, and the proximity of u_{200} to an optimum is evident from the optimality function, $\theta(u_{200}) = -0.03265$.

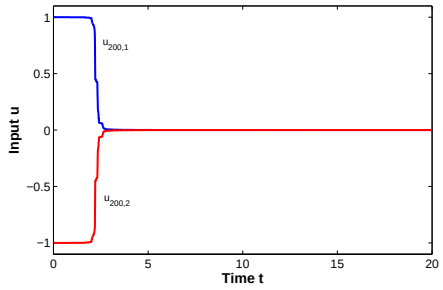
Throughout the course of 200 iterations the cost came down from $J(u_1) = 4,012$ to $J(u_{200}) = 724.5$, but this reduction is by no means linear in the number of iterations. In fact, the graph of the cost $J(u_i)$ as a function of i, shown in part (e) of the figure, shows a rapid decrease in a few iterations at the early part of the algorithm's run, followed by a relatively flat curve. 95% of the cost reduction was achieved by 5 iterations, while 98% was obtained by 8 iterations. The corresponding graph of $\theta(u_i)$ as a function of i is shown in part (f), and it exhibits a rapid ascent towards 0 in a handful of iterations. These graphs are consistent with cumulative experience with gradient-descent algorithms with Armijo step sizes, which tend (in many cases) to make most of their strides towards minimum-points in a handful of iterations. This factor, together with the global stability of such algorithms, has made them



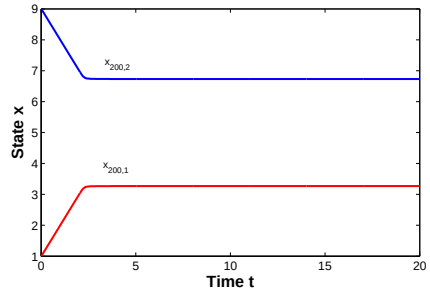
(a) Initial input, $u_{1,1}(t)$ and $u_{1,2}(t)$



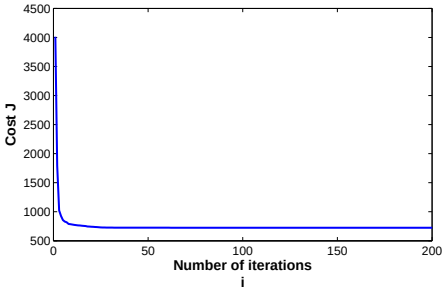
(b) Initial State, $x_{1,1}(t)$ and $x_{1,2}(t)$



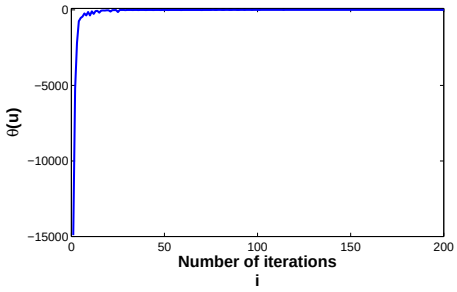
(c) Final Input, $u_{200,1}(t)$ and $u_{200,2}(t)$



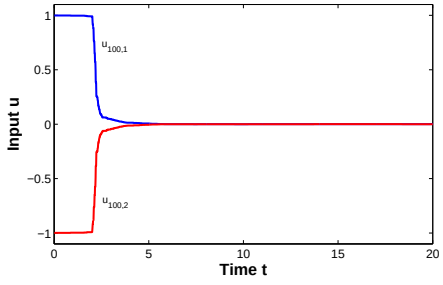
(d) Final State, $x_{200,1}(t)$ and $x_{200,2}(t)$



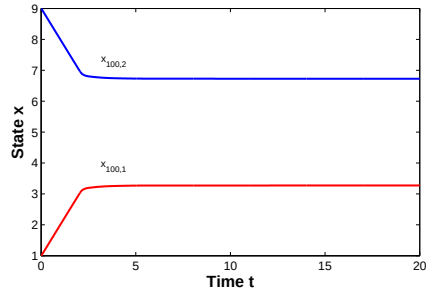
(e) $J(u_i)$ as a Function of i



(f) $\theta(u_i)$ as a Function of i



(g) $u_{100,1}(t)$ and $u_{100,2}(t)$



(h) $x_{100,1}(t)$ and $x_{100,2}(t)$

Figure 39: Results of Algorithm 1: Two agents, $\Delta t = 0.01$

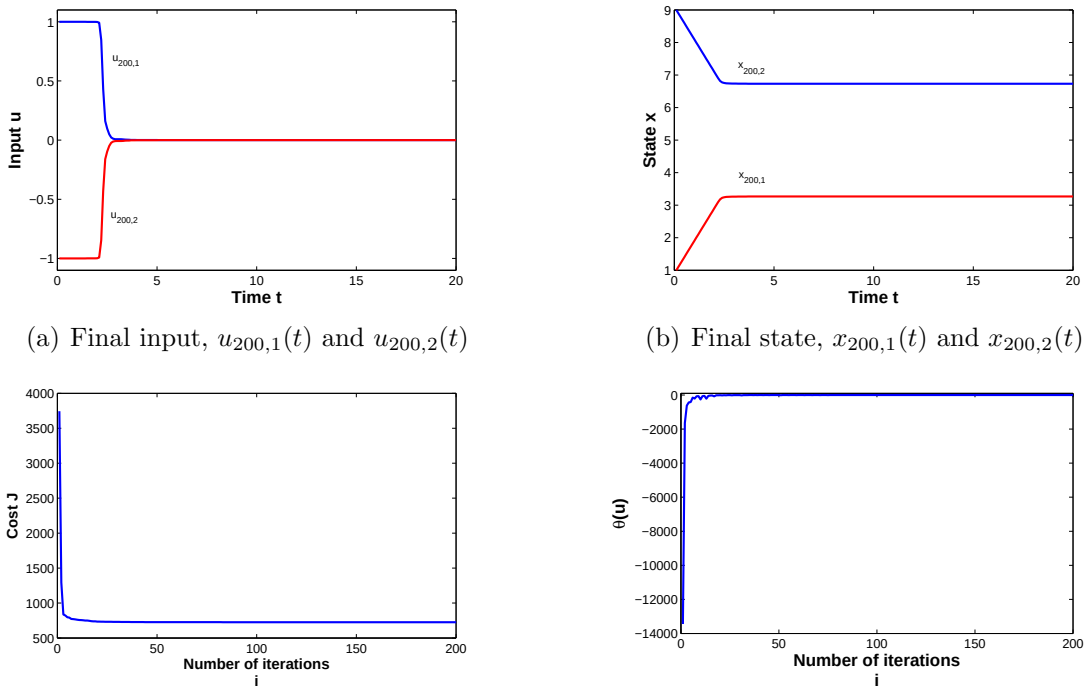
attractive despite the fact that their asymptotic convergence is slower than that of other techniques.

The algorithm's run of 200 iterations took 5.95 seconds of CPU time on a PC laptop with Intel Core i7-2630QM (2.00 Gigahertz) processor. However, Figure 2(e) indicates that fewer iterations would suffice while reducing the CPU times. For instance, a run of 100 iterations takes 2.99 seconds of CPU times, and would produce a final cost of $J(u_{100}) = 724.6$ as compared to $J(u_{200}) = 724.5$. Its final control and state trajectory, depicted in parts (g) and (h) of the figure, are barely distinguishable from those of u_{200} and x_{200}. As few iterations as 20 may suffice as can be seen in Figure 2(e).

To further reduce the execution time we increased the grid's time interval Δt from 0.01 to 0.1. Remember that the CPU time for 200 iterations with $\Delta t = 0.01$ was 5.95 seconds. However, the case with $\Delta t = 0.1$ took 0.75 seconds of CPU time, and the cost went down from $J(u_1) = 3,743.4$ to $J(u_{200}) = 726.2$ (as compared to 724.5 when $\Delta t = 0.01$), while the optimality function went up from $\theta(u_1) = -13,445$ to $\theta(u_{200}) = -0.0356$. The results are shown in Figure 3 where part (a) and part (b) depict u_{200} and x_{200}, while part (c) shows the graph of $J(u_i)$ vs. i. Again, we see that most of the decline in the cost is achieved in the first few iterations.

The case of six agents The system is similar to that of the last subsection except that the distance of the object from the controller is $d = 20$, and there are six agents and hence $u \in R^6$ and $x \in R^6$. The initial position of the agents is $x(0) = (1 \quad 2 \quad 7 \quad 9 \quad 12 \quad 19)^\top$. The simulation was done under the same setup as before, i.e., $t_f = 20$, $\Delta t = 0.01$, and $C = 7$. The algorithm was run for 200 iterations, starting from the initial input $u_{1,k}(t) = 1.0$ for every $k = 1, \ldots, 6$, and for all $t \in [0, t_f]$. The execution CPU time was 8.67 seconds.

The results are shown in Figure 4, whose parts (a) and (b) depict the final input and corresponding state trajectories, while parts (c) and (d) show the graphs of the cost and optimality function as functions of the iteration count. Part (c) of the figure



(a) Final input, $u_{200,1}(t)$ and $u_{200,2}(t)$

(b) Final state, $x_{200,1}(t)$ and $x_{200,2}(t)$

(c) $J(u_i)$ as a function of i

(d) $\theta(u_i)$ as a function of i

Figure 40: Results of Algorithm 1: Two agents, $\Delta t = 0.1$

exhibits a similar pattern of rapid descent of the cost from $J(u_1) = 7,969$ to $J(u_{200}) = 1,253.6$, and it took 7 and 14 iterations to achieve 95% and 98% of the total cost reduction, respectively. Correspondingly, the optimality function rises from $\theta(u_1) = -28,537$ to $\theta(u_{200}) = -3.506$. The proximity of u_{200} to the optimum, or at least a local minimum, was tested by various runs of 400 iterations starting from different initial inputs. The lowest value of $J(u_{400})$ we obtained was $1,252.3$ as compared to $J(u_{200}) = 1,253.6$, and the corresponding value of the optimality function was -0.805 as compared to $\theta(u_{200}) = -3.506$. Thus, we believe that the algorithm practically converged, and by parts (c) and (d) of the figure, quite rapidly. The rapid convergence of the algorithm is despite the fact that not all of the components of the input variables, $u_{200,k}(t)$, $k = 1, \ldots, 6$, resemble a bang-of-bang control. To explain this phenomenon, we point out that there is a great degree of insensitivity of J to certain variations in u. To see this point, recall that the chattering lemma implies that certain

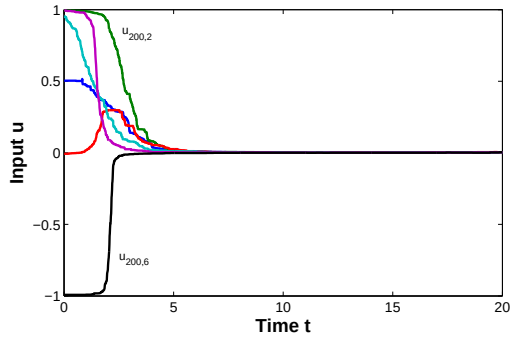
large L^1 variations in u yield small variations in the corresponding state trajectory x. Some such variations result in small variations in J. Figure 4(c) leads us to believe that our algorithm drives the controls toward such a region containing the optimal control, and that is why the computed cost-performance and state trajectories, but not the controls, are close to those of the optimal ones. *The effectiveness of the algorithm is not in approximating the optimal control $u^\star$ by a computed control u_k, but in approximating the optimal cost $J(u^\star)$ by the cost of a computed control, $J(u_k)$.*[1] The above-reported run achieved that in 8.25 seconds, and further reduction in the CPU time can be obtained by running fewer iterations: it takes CPU times of 0.8703, 1.669, and 4.115 seconds to execute 20 iterations with $J(u_{20}) = 1335$, 40 iterations with $J(u_{40}) = 1,284$, and 100 iterations with $J(u_{100}) = 1,257.5$ respectively, as compared to 200 iterations with $J(u_{200}) = 1,253.6$. Further reductions can be obtained by increasing Δt. In fact, we ran the algorithm with $\Delta t = 0.1$, and the resulting values were $J(u_{20}) = 1,335.5$ with CPU time of 0.178; $J(u_{100} = 1,259.4$ with CPU time of 0.5741; and $J(u_{200}) = 1,255$ with CPU time of 1.077 seconds. These numbers are quite close to those obtained with $\Delta t = 0.01$, and further point to the effectiveness of the algorithm.

4.3.3 Two-dimensional System

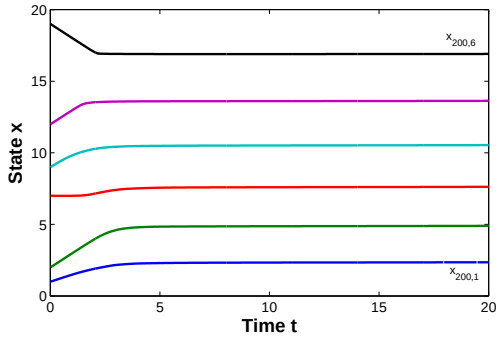
In this section, we extend the problem of forming a relay network between a transmitter an a receiver to two dimensions. In addition to that, we propose a framework that results in a whole class of energy-efficient distributed controllers that can be used for solving classic problems in multi-agent systems like rendezvous, formation-control, and coverage control in an energy efficient manner.

We start with the problem setup shown in Figure (42), in which N mobile agents relay information between base stations B_1 and B_2 and they have to maintain this

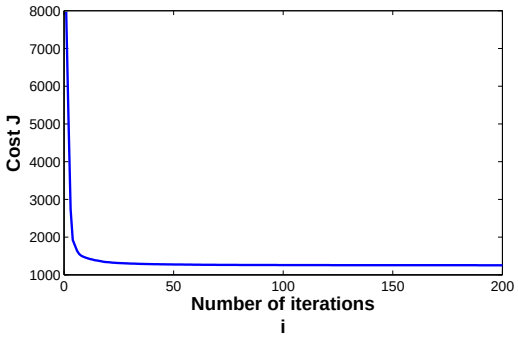
[1] Practically, what may matter most in various applications is convergence in the weak topology on the space of controls.



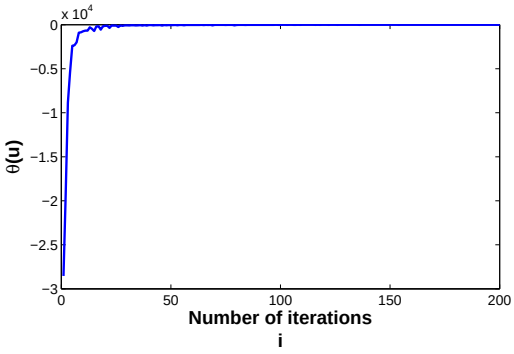
(a) Final input, $u_{200,k}(t), \quad k = 1, \ldots, 6$



(b) Final state, $x_{200,k}(t), \quad k = 1, \ldots, 6$



(c) $J(u_i)$ as a function of i



(d) $\theta(u_i)$ as a function of i

Figure 41: Results of Algorithm 1: Six agents, $\Delta t = 0.01$

network for time t_f. Let $z_k \in \mathbb{R}^2$, $k = 1, \ldots, N$ be the location of each agent where $z_k \triangleq [x_k \;\; y_k]^T$. For each agent, we assume single integrator dynamics

$$z_k = u_k. \tag{4.44}$$

Our objective as before is to minimize total energy consumption over time interval $[0, t_f]$, and to achieve this objective agents can move around to balance out their locations in order to reduce the distance over which they need to transmit data. We assume that agents use broadcast model for data transmission in which each agent broadcasts its data to all its neighbors. Therefore, to reduce transmission energy, each agent tries to minimize its distance from the neighbor that is farthest away. As a result, the cost function that we are trying to minimize is

$$J = \sum_{k=1}^{N} \int_0^{t_f} \left[C\|u_k(t)\| + \max_{j \in \mathcal{N}_k} \|z_k - z_j\|^2 \right] dt \tag{4.45}$$

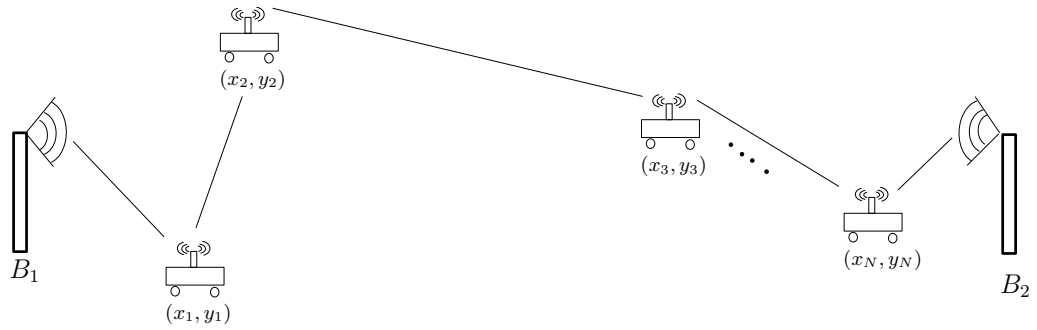


Figure 42: Tandem network in two dimensions

The Hamiltonian has the following form,

$$H(x, u, p) = \sum_{k=1}^{N} \left[C\|u_k(t)\| + p_k^T u_k + \max_{j \in \mathcal{N}_k} \|z_k - z_j\|^2 \right], \tag{4.46}$$

where p_k is the costate.

In one-dimensional case, the control u_k was restricted to $[-1, 1]$ which simplified the problem because the optimal solution was a bang-off-bang control. However, in two dimensions, we need to find optimal trajectories for each agent that minimize the cost. Moreover, we want to present a formulation for a complete class of problems, in which case, depending on the nature of the problem, there can be complex constraints on the state like connectivity, obstacle avoidance or formation maintenance. Solving optimal control problems with constraints is in general difficult and to propose a generalized solution for a class of problems is an extremely challenging task.

To simplify this task, we formulate this problem as an optimal control problem with no constraints, for which we have to define the state dynamics such that the constraints are handled completely by these dynamics. For instance, in the problem that we are trying to solve, we know that if there is no cost associated with mobility then the optimal locations for the agents lie on the straight line between B_1 and B_2 such that the agents are distributed equidistantly. From ([14] and [16]), the agents are guaranteed to reach optimal locations asymptotically if their dynamics are governed

by consensus equation. Therefore, we define the following dynamics for each agent.

$$\dot{z}_k = u'_k \sum_{j \in \mathcal{N}_k} (z_j - z_k), \tag{4.47}$$

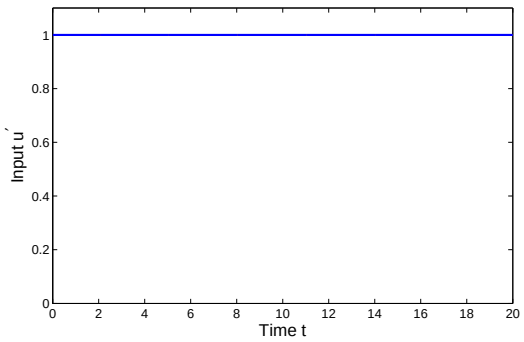
where $u'_k \in [0, 1]$ and $\mathcal{N}_k$ is the set of neighbors of agent k. Through above dynamics, we have incorporated all the constraints for this problem in the state dynamics and reduced our problem to finding optimal control u'_k that minimizes the cost (4.45). For a generalized formulation, we can use weighted consensus equation, through which, by intelligently selecting weights, we can solve interesting multi-agent coordination problems like connectivity maintenance, formation control, obstacle avoidance and coverage control in a distributed manner. Let us define the switching signal

$$\mathcal{S}_k(t) = C \left\| \sum_{j \in \mathcal{N}_k} (z_j(t) - z_k(t)) \right\|^2 + p_k^T(t) \sum_{j \in \mathcal{N}_k} (z_j(t) - z_k(t)) \,.$$

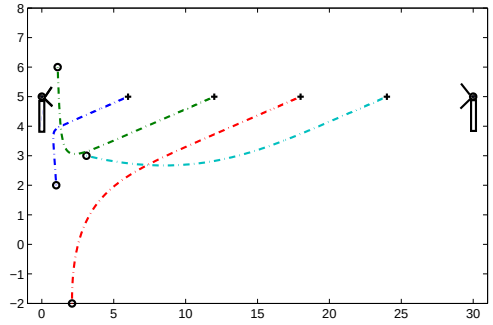
Then, the minimizer for the Hamiltonian (4.46) is

$$u'_k(t) = \begin{cases} 0, & \mathcal{S}_k(t) > 0 \\ 1, & \mathcal{S}_k(t) < 0, \end{cases} \tag{4.48}$$

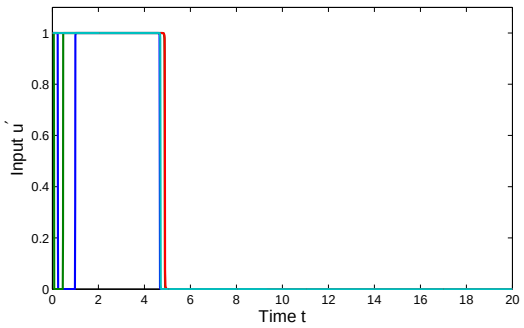
Thus, the control u'_k is simple a bang-off control that controls how much each agent is allowed to move under the consensus dynamics depending on the cost of mobility. Another major advantage of this problem formulation is that we can still use Algorithm (1) for solving the optimal control problem since our state dynamics and cost have the special structure that is required by this algorithm. To get a better insight and to check the validity of the proposed framework, we simulated the network in Figure (42) for $N = 4$ agents and $t_f = 20$, and the results are presented in Figure (43). Parts (a) and (b) of this figure correspond to final input and state of the agents after 200 iterations of the algorithm when mobility cost $C = 0$. In this case, we can see that the control $u'_{200}(t) = 1$ for all $t \in [0, t_f]$ and for all the agents. From the state trajectories of each agent, represented by dotted lines, we can see that the final location of the agents (represented by '$\times$' at the end of trajectories) lie on the straight



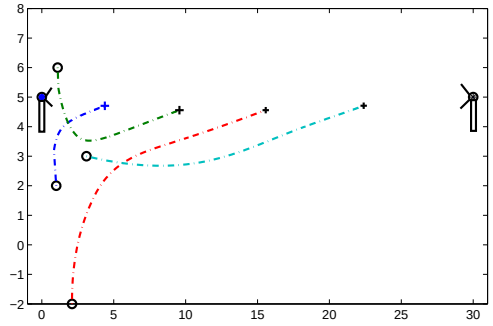
(a) Final input, $C = 0$, $u'_{200,k}(t)$, $k = 1, \ldots, 4$



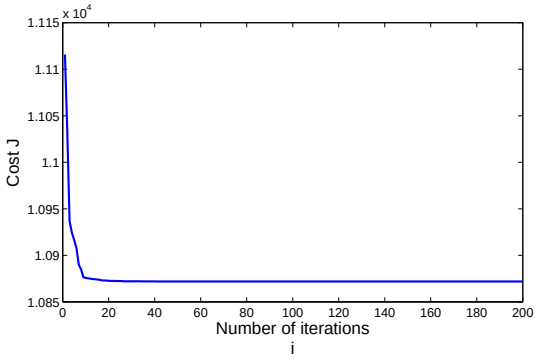
(b) Final state, $C = 0$, $x_{200,k}(t)$, $k = 1, \ldots, 4$



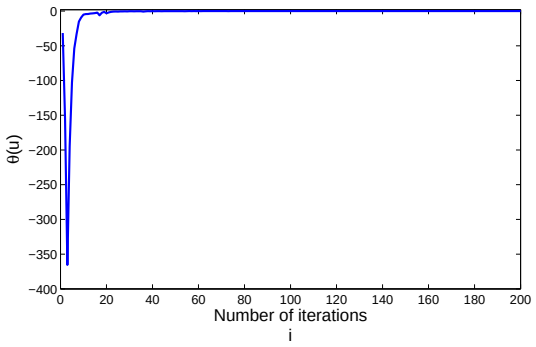
(c) Final input, $C = 50$, $u'_{200,k}(t)$, $k = 1, \ldots, 4$



(d) Final state, $C = 50$, $x_{200,k}(t)$, $k = 1, \ldots, 4$



(e) $J(u'_i)$ as a function of i for $C = 50$,



(f) $\theta(u'_i)$ as a function of i for $C = 50$,

Figure 43: Results of proposed framework for two dimensional system: Four agents, $\Delta t = 0.01$

line between the base stations and the agents are equidistant. Next we simulated the same system for $C = 50$, implying that mobility is 50 times more expensive then data transmission. In this case, from part (d) of the figure, we can see that the final location of the agents are still on the straight line between the base stations, but the agents are no longer equidistant. Moreover, the corresponding optimal control for

each agent, shown in part (c), is bang-off control, because after $t = 5$, input is zero for all the agents. The cost associated with this problem is shown in part (e), and we can see that the cost reduced from $J(u_1') = 11116.4$ to $J(u_{200}') = 10872$. The fact that this cost is minimum is proved by the plot of $\theta(u')$, which shows that after 200 iterations $\theta(u_{200}') = -0.0007501$ which is almost equal to zero.

4.3.4 Real-time Implementation

The fast convergence in the initial phase and the low execution time requirements of Algorithm 1 allows us to implement this algorithm in real time. In the real-time implementation of the system described in Section 4.3.1 with six agents, each agent solves the problem of optimizing its own cost, i.e., each agent minimizes the cost

$$J_{dist}^i = J_{trans}^i + J_{mobility}^i, \tag{4.49}$$

where

$$J_{trans}^i = \max_{j \in \mathcal{N}_i} \int_0^{t_f} \|x_i - x_j\|^2 dt,$$

and

$$J_{mobility}^i = C \int_0^{t_f} |u_i(t)| dt.$$

To minimize this cost, agent i located at x_i performs the following steps.

Algorithm 2. *Parameters: k, Δt, C, t_f, t_s.*

for $k_1 = 0 : \lfloor t_f/t_s \rfloor$

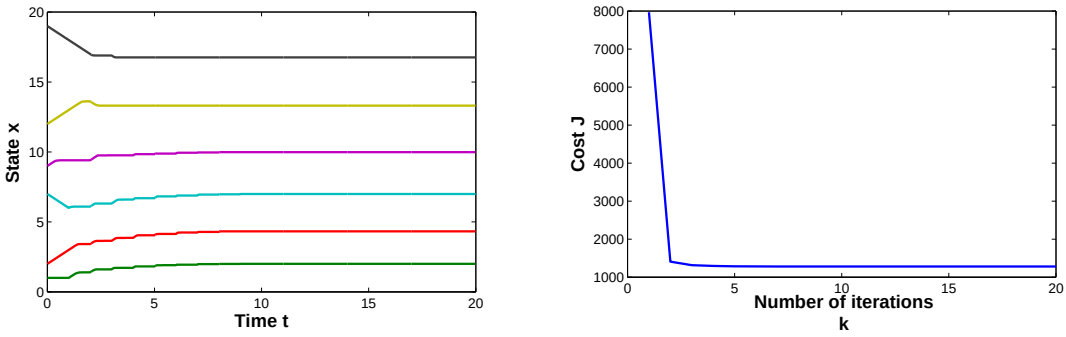
Step 1: Assumes its neigbors, i.e., j such that $j \in \mathcal{N}_i$, are stationary

Step 2: Applies Algorithm 1 to find optimal control u_i that minimizes J_{dist}^i over time interval $(k_1 t_s, t_f)$.

Step 3: Uses the control u_i to update its state according to system dynamics (4.37) for time interval $((k_1 - 1)t_s, k_1 t_s)$.

Step 4: $k_1 = k_1 + 1$.

end



(a) Final State (b) Cost

Figure 44: Results of Algorithm 2: Six agents, $\Delta t = 0.01$, $T = 20$, $k = 200$, and $T_1 = 1$ second

Thus, for the scenario when Algorithm 2 is applied to a tandem network of six agents (as in the previous example), each relay node effectively solves the optimal control problem for one node only between a transmitter and a receiver, and for $k = 200$ and $\Delta t = 0.01$ this problem is solved in 1.89 seconds. Since all the relay nodes will be solving this problem in parallel, the total computation time will still be 1.89 seconds. Each node will use this control for $t_s = 1$ and then solve the optimal control problem to find the cost to go.This means that for total time $t_f = 20$, each node will have to compute cost to go 20 times. The simulation results of the system with six agents under real-time scheme are presented in Figure 44, where Figures 44(a) and 44(b) show the final states of the agents after $k = 200$ iterations and the cost of the system at each iteration respectively. From Figure 44(b), $J(u_{200}) = 1279.3$, which is very close to the optimal cost for the centralized problem, which was $J(u_{200}) = 1253.6$. To improve the execution time, we reduced Δt from 0.01 to 0.1. As a result the execution time for solving the problem one time was reduced to 0.42 seconds from 1.89 seconds and the cost increased from 1279.3 to 1282.1. To further improve the execution time, we showed in the previous section that for the case of six agents, 98% of the cost was reduced in 14 iterations. Therefore, we simulated the system for $k = 20$ iterations which resulted in a final cost of $J(u_{20}) = 1286.6$, where as the

execution time now reduced to 0.04 seconds. Next, we reduced the time for which each agent uses the control it computed from $t_s = 1$ to $t_s = 0.1$. In this case, the cost is $J(u_{20}) = 1272.8$ and $J(u_{50}) = 1267.4$

Despite these promising simulation results, real-time algorithm still requires thorough investigation. To make any claims regarding the performance of this algorithm, we have to perform detailed convergence analysis of this algorithm and analytically compare its performance with the centralized algorithm. The contribution of this work is to propose a framework that has great potential for solving an interesting and all important problem of minimizing total energy consumption in distributed multi-agent systems. However, this framework still requires rigorous mathematical treatment before it can be used for practical implementation.

CHAPTER V

CONCLUSIONS

In this work, we presented power-aware scheduling schemes for wireless sensor networks, which were categorized into two broad categories, namely static networks and mobile networks. Static networks comprised of agents with no mobility whereas mobile networks comprised of agents with mobility. For both of these classes of sensor networks, power management was shown to be major design consideration since limitation of available power is a bottleneck in the design of these systems.

In the context of power management in randomly deployed wireless sensor networks, there are two different notions of power-awareness. The first notion of power-awareness that we presented in this work is the effect of environmental factors and ageing effect of power supplies on the performance of individual sensing devices and on the performance of the entire network. We supported our claim from references from the existing literature that this indeed is an important factor that can impede the capacity of the network to ensure desired performance level. However this aspect of power-awareness is completely missing from the literature on wireless sensor networks. Therefore, in Chapter 2, we considered this aspect of power-awareness, i.e., how does decrease in available power affects the performance of individual devices and of the entire network.

To address this problem, we used area of the sensor footprint as a metric for sensor performance and presented explicit relationship between expected footprint area of a sensor in the network and the desired event detection probability, which was our desired performance criterion. Using this relationship, we presented feedback scheduling controllers that allowed sensors to update their control parameter based on

their available power at the decision time. The strength of these controllers was that they were completely decentralised since sensors did not have to communicate with their neighbors to make switching decisions. Moreover, the controller only needed to compute the ratio of current power to initial power to update the control parameter which simplified the implementation of these controllers on cheap and low quality sensing devices. For these controller, we considered Boolean sensing model in which case the proposed scheme was shown to be applicable as long as the sensor footprint was compact. We also proposed a non-Boolean sensing model in which event detection probability was a function of the distance between an event and a sensor.

In Chapter 3, we considered the other notion of power-awareness which is related with the efficient utilization of available energy resources to maximize the lifetime of the network. This notion of power awareness is popular in sensor networks community and a huge body of work exists on energy-efficient sensor scheduling schemes. However, in this work, we proposed a novel sleep-scheduling scheme that was a mix of random and deterministic schemes and we showed through extensive simulations that the proposed scheme extended the lifetime of the network from 40% to 70% as compared to random switching scheme. To propose this scheme, we borrowed the concept of a hard-core point process form stochastic geometry, which is an inhibition process that does not allow the constituent points to lie closer than certain minimum distance. However, to enforce this inhibition distance in sensor networks, sensors had to communicate with their neighbors. The information that sensors communicated with their neighbors consisted of random numbers which made this scheme random. Nevertheless, the major challenge was the modeling of the coverage properties of the point process in the case of overlapping disks since no results were available in the existing literature. Thus, we started with the mathematical analysis of this process and derived an expression for event detection probability when the coverage process formed by sensor network was a pure hard-core process, i.e., distance between sensors

was twice their sensing radius.

Then to generalize our analysis so that we can design sensor scheduling scheme for any desired level of coverage, we developed a complete model of the coverage process of a hard-core point process in which the sensing disks were allowed to overlap. To derive this model, we performed extensive Monte Carlo simulations and used curve fitting toolbox from Matlab. The validity of the proposed model was shown through designing a sensor network for various desired performance levels and it was shown that the designed network maintained the performance with average error of less than 1%. Later on we compared the performance of a sensor network comprising of MICA2 motes installed with magnetic sensors for detecting the magnetic field of vehicles in their footprint. For these devices we showed that even after the added cost of communication, the proposed scheme extended the lifetime of the network by 40% to 70% as compared to random scheme.

We also studied different types of hard-core point process and developed relationships between the intensities of these new hard-core processes and the process that we modeled, which allowed us to design these additional types of hard-core processes using the model that we developed. Moreover, we used one of the hard-core processes, in which points were vibrated to increase the intensity, to propose a distributed coverage control algorithm that can guarantee partial coverage.

In Chapter 4 of this work, we dealt with mobile sensor networks. We started by looking at the classic rendezvous problem and proposed power-aware controllers that guaranteed to achieve global objective in the presence of shrinking footprints. The work corresponded to the first notion of power-awareness, i.e., to be aware of the power level that is available and adjust the control parameter according to compensate for any negative effects because of variations in available power.

In the second half Chapter 4, we studied the problem of minimizing total energy consumption in network of mobile agents because of both mobility and data

transmission. We formulated this problem in generalized optimal control setting and identified an efficient numerical technique for solving this optimal control problem. We started our analysis with a simplified problem in one dimension and applied the selected numerical algorithm for solving it, and the results of the simulation showed the efficacy of this scheme. The strength of this numerical algorithm was shown to be the fact that it reduced around 95% of the cost in less than 20 iterations which was extremely fast.

We extended our problem formulation for two-dimensional case. For problem formulation in two-dimensions, we proposed to use consensus equation to govern the state dynamics and introduced a switching signal whose purpose was simply to control how long agents were allowed to move under these dynamics depending on the cost associated with mobility. The advantage of this approach was that it reduced a complex optimal control with complicated constraints on the state to a simple bang-off control. Moreover, numerous variations of consensus equation exist in the literature to solve canonical problems like flocking, formation control, obstacle avoidance and coverage control. Therefore, the proposed framework led to a whole class of power-aware controllers for solving canonical problems in multi-agent systems. Finally, we also proposed framework for real-time implementation of these power-aware controllers and showed its validity by simulating a simple system with six agents. However, the real-time implementation of these controllers is still an open problem since there is a need to provide analytical proofs for the convergence of the algorithm under real-time implementation. Moreover, it will be instructive to compare the performance of real-time algorithm with the centralized case that we presented to check whether there are any performance compromises under real-time implementation.

Following is a list of publications resulted from this work.

Journal Publications

- H. Jaleel, A. Rahmani and M. Egerstedt, "Probabilistic lifetime maximization of sensor networks," in *IEEE Transactions on Automatic Control*, vol. 58, iss. 2, pp. 534–539, Feb. 2013.

- H. Jaleel and M. Egerstedt, "Power-aware scheduling of wireless sensor networks with dynamic footprints," *Submitted in IEEE Transaction on Control of Network Systems*.

Conference Publications

- H. Jaleel, A. Rahmani and M. Egerstedt, "Duty cycle scheduling in dynamic sensor networks for controlling event detection probabilities," in *Proc. of IEEE American Control Conference*, pp. 3233–3238, July 2011.

- H. Jaleel and M. Egerstedt, "Power-aware rendezvous with shrinking footprints," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2772–2777, Oct. 2011.

- H. Jaleel, S. Bopardikar, and M. Egerstedt, "Towards power-aware rendezvous," in *Proc. of IEEE Conference on Decision and Control*, pp. 6642–6647, Dec. 2011.

- H. Jaleel and M. Egerstedt, "Sleep scheduling of wireless sensor networks using hard-core point processes," in *Proc. of IEEE American Control Conference*, Washington D.C., pp. 788–793, June 2013.

- H. Jaleel and M. Egerstedt, "Distributed and adaptive power-aware scheduling of wireless sensor networks," in *IEEE Conference on Decision and Control,* Florence, Italy, Dec. 2013. To appear.

- H. Jaleel, Y. Wardi, M. Egerstedt, "Minimizing Mobility and Communication Energy in Robotic Networks: an Optimal Control Approach," *Submitted in American Control Conference*, Portland, OR, Sept. 2013

- W. Abbas, H. Jaleel, and M. Egerstedt, "Energy-Efficient data collection in heterogeneous wireless sensor and actor networks, *IEEE Conference on Decision and Control,* Florence, Italy, Dec. 2013. To appear.

VITA

Hassan Jaleel received his B.S. degree in Electrical Engineering from the University of Engineering and Technology in Lahore, Pakistan, in 2007, where he also served as a Lecturer from 2007 to 2009. He received his M.S. degree in Electrical and Computer Engineering from the Georgia Institute of Technology in Atlanta, Georgia, in 2010, where he is currently a doctoral student working on distributed coordination algorithms for wireless sensor networks. His research focuses on designing probabilistic sleep scheduling schemes for minimizing energy consumption in order to maximize the lifetime of wireless sensor networks while maintaining desired performance criterion. He is also interested in developing energy-efficient mobility strategies for achieving desired global objectives including connectivity maintenance , coverage control, and rendezvous, while minimizing total energy consumption. Jaleel has been a student member of IEEE since 2004 and a Fulbright scholar since 2009.

Bibliography

[1] I. F. Akyildiz, E. Cayirci, W. Su and Y. Sankarasubramaniam, "A survey on sensor networks," in *IEEE Commun. Mag.*, vol. 40, issue 8, Aug. 2002, pp.102-114.

[2] J. Cortés, S. Martinez, and F. Bullo, "Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions," in *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1289–1298, Aug. 2006.

[3] J. Cortés, S. Martinez, T. Karatas and F. Bullo, "Coverage control for mobile sensing networks," in *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, Apr. 2004.

[4] L. Wang and Y. Xiao, "A survey of energy-efficient scheduling mechanisms in sensor networks," in *Mobile Networks and Applications*, vol 11, issue 5, pp. 723–740, 2006.

[5] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*, Princeton University Press, Princeton, NJ, Sept. 2010.

[6] J. Lawton, R. Beard, and B. Young, "A decentralized approach to formation maneuvers," in *IEEE Transactions on Robotics and Automation*, vol. 19, no. 6, pp. 933–941, Dec. 2003.

[7] J. Desai, J. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots," in *Proc. of the IEEE International Conference on Robotics and Automation*, pp. 2864–2869, May 1998.

[8] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," in *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 947–951, Dec. 2001.

[9] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *First European Workshop on Wireless Sensor Networks*, Jan. 2004.

[10] W. Ren and R. Beard, "Consensus of information under dynamically changing interaction topologies", in *Proc. of American Control Conference*, vol. 6, pp. 4939–4944, June 2004.

[11] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," in *Proc. of the IEEE Conference on Decision and Control*, pp. 2968–2973. Dec. 2001.

[12] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," in *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, June 2003.

[13] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita, "Distributed memoryless point convergence algorithm for mobile robots with limited visibility," in *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 818–828, Oct. 1999.

[14] R. O. Saber and R. M. Murray, "Agreement problems in networks with directed graphs and switching toplogy," in *Proc. 42nd IEEE Conf. Decision Control*, vol. 4, Maui, HI, Dec. 2003, pp. 4126–4132.

[15] S. Kar and J. M. F. Moura, "Distributed average consensus in sensor networks with quantized inter-sensor communication", in *Proc. ICASSP*, Apr. 2008, pp. 2281 - 2284.

[16] R. O. Saber and R. M. Murray, "Flocking with Obstacle Avoidance: Cooperation with Limited Communication in Mobile Networks", in *Proc. of the IEEE Conference on Decision and Control*, vol. 2, pp. 2022–2028, Dec. 2003.

[17] P. Ögren, M. Egerstedt, andX. Hu, "A control Lyapunov function approach to multi-agent coordination," in *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, Oct. 2002, pp. 847–851.

[18] M. M. Zavlanos, H. G. Tanner, A. Jadbabaie and G. J. Pappas. "Hybrid control for connectivity preserving flocking," in *IEEE Transactions on Automatic Control*, vol. 54, no. 12, pp. 2869–2875, Dec. 2009.

[19] A. Alfieri, A. Bianco, P. Brandimarte and C.F. Chiasserini, "Maximizing system lifetime in wireless sensor networks," in *European Journal of Operational Research*, vol. 181, issue 1, pp. 390–402, Aug. 2007.

[20] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Proc. of IEEE International Conference in Communications*, vol. 2, pp. 472–476, June 2001.

[21] R. Subramanian and F. Fekri, "Sleep scheduling and lifetime maximization in sensor networks: fundamental limits and optimal solutions," in *Proc. of ACM International Symposium on Information Processing in Sensor Networks*, pp. 218–225, Apr. 2006.

[22] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang. "PEAS: A robust energy conserving protocol for long-lived sensor networks," in *Proc. of IEEE International Conference on Distributed Computing Systems*, pp. 28–37, May 2003.

[23] C. Gui and P. Mohapatra, "Power conservation and quality of surveillance in target tracking sensor networks," in *Proc. of ACM International Conference on Mobile Computing and Networking*, pp. 129–143, Sept. 2004.

[24] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic, "Towards optimal sleep scheduling in sensor networks for rare-event detection," in *Proc. of ACM International Symposium on Information Processing in Sensor Networks*, no. 4, Apr. 2005.

[25] Y. Tsai, "Sensing coverage for randomly distributed wireless sensor networks in shadowed environments," in *IEEE Transactions on Vehicular Technology*, vol. 57, no. 1, pp. 556–564, Jan. 2008.

[26] C. Hsin and M. Liu, "Network coverage using low duty-cycled sensors: random & coordinated sleep algorithms," in *Proc. of ACM International Symposium on Information Processing in Sensor Networks*, pp. 433–442, Apr. 2004.

[27] D. Tian and N. D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *Proceedings of ACM Workshop on Wireless*

Sensor Networks and Applications., pp. 32–41, 2002.

[28] V. Rodoplu and T. Meng, "Minimum energy mobile wireless networks," in *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1333–1344, Aug. 1999.

[29] S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang, "Design and analysis of sensing scheduling algorithms under partial coverage for object detection in sensor networks," in *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 3, pp. 334–350, Mar. 2007

[30] D. Stoyan, W. S. Kendall and J. Mecke, *Stochastic Geometry and its Applications*, Wiley (Chichester W. Sussex and New York), 1987.

[31] P. Hall, *Introduction to the Theory of Coverage Processes*, New York, Wiley, 1988.

[32] B. Matérn, *Spatial Variation*, Lecture Notes in Statistics, Springer-Verlag, New York, NY, 1986

[33] C. Godsil and G. Royle, *Algebraic Graph Theory*, Berlin, Germany, Springer, 2001.

[34] P. Dutta, M. Grimmer, A. Arora, S. Bibyk, and D. Culler, "Design of a wireless sensor network platform for detecting rare, random, and ephemeral events," in *Proceedings of The Fourth IPSN,* Apr. 2005.

[35] H. Jaleel, A. Rahmani and M. Egerstedt, "Duty cycle scheduling in dynamic sensor networks for controlling event detection probabilities," in *Proc. of IEEE American Control Conference*, pp. 3233–3238, July 2011.

[36]

[36] H. Jaleel, A. Rahmani and M. Egerstedt, "Probabilistic lifetime maximization of sensor networks," in *IEEE Transactions on Automatic Control*, vol. 58, iss. 2, pp. 534–539, Feb. 2013.

[37] H. Jaleel and M. Egerstedt, "Distributed and adaptive power-aware scheduling

of wireless sensor networks," in *IEEE Conference on Decision and Control,* Florence, Italy, Dec. 2013. To appear.

[38] H. Jaleel and M. Egerstedt, "Power-aware scheduling of wireless sensor networks with dynamic footprints," in *Submitted in IEEE Transaction on Control of Network Systems.*

[39] H. Jaleel and M. Egerstedt, "Sleep scheduling of wireless sensor networks using hard-core point processes," in *Proc. of IEEE American Control Conference*, Washington D.C., pp. 788–793, June 2013.

[40] H. Jaleel and M. Egerstedt, "Power-aware rendezvous with shrinking footprints," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2772–2777, Oct. 2011.

[41] H. Jaleel, S. Bopardikar, and M. Egerstedt, "Towards power-aware rendezvous," in *Proc. of IEEE Conference on Decision and Control*, pp. 6642–6647, Dec. 2011.

[42] W. Abbas, H. Jaleel, and M. Egerstedt, "Energy-Efficient data collection in heterogeneous wireless sensor and actor networks, *IEEE Conference on Decision and Control,* Florence, Italy, Dec. 2013. To appear.

[43] H. Jaleel, Y. Wardi, M. Egerstedt, "Minimizing Mobility and Communication Energy in Robotic Networks: an Optimal Control Approach," in *http://users.ece.gatech.edu/ magnus/Papers/MobComRobOpt13.pdf*, Technical Report Sept. 2013.

[44] F. Zhang and Z. Shi, "Optimal and adaptive battery discharge strategies for cyber-physical systems," in *Proc. of IEEE Conference on Decision and Control*, pp. 6232–6237, Dec. 2009.

[45] K. Chang, *RF and Microwave Wireless Systems*, New York, Wiley, 2000.

[46] P. Martin, R. Galvan-Guerra, and M. Egerstedt, "Power-aware sensor coverage: An optimal control approach," in *International Symposium on Mathematical Theory of Networks and Systems*, July 2010.

[47] D. M. Doolin and N. Sitar, " Wireless sensors for wildfire monitoring," in *Smart Structures and Materials. International Society for Optics and Photonics*, pp. 477-484, May 2005.

[48] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pp. 13–24, Nov. 2004.

[49] R. O. Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays", in *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, Sept. 2004.

[50] C.-F. Chiasserini and R. Rao, "Energy efficient battery management," in *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 7, pp. 1235–1245, July 2001.

[51] M. Bargiel and E. M. Tory, " Packing fraction and measures of disorder of ultra-dense irregular packings of equal spheres. I. Nearly ordered packing," in *Advanced Powder Technology*, vol. 4 no. 2, pp. 79–101, 1993.

[52] E. Royer, P. Melliar-Smith, and L. Moser, "An analysis of the optimum node density for ad hoc mobile networks," in *IEEE International Conference on Communications,* vol. 3, pp. 857-861, Jun. 2001.

[53] C. Bettstetter, " Smooth is better than sharp: A random mobility model for simulation of wireless networks," in *Proc. ACM Intern. Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems,* pp. 19–27, July 2001.

[54] J. Mościński, M. Bargiel, Z. A. Rycerz, and P. W. M. Jacobs, "The force-biased algorithm for the irregular close packing of equal hard spheres," in *Mol. Sim.* vol. 3, pp. 201–212, 1989.

[55] R. Rao, S. Vrudhula, and D. N. Rakhmatov, "Battery modeling for energy aware system design," in *Computer*, vol. 36, no. 12, pp. 77–87, Dec. 2003.

[56] D. Rakhmatov, S. Vrudhula, and D. A. Wallach, "A model for battery lifetime analysis for organizing applications on a pocket computer," in *IEEE Transactions on VLSI Systems,* vol. 11, no. 6, pp. 1019–1030, Dec. 2003.

[57] J. A. Marshall, M. E. Broucke and B. A. Francis, "Formations of vehicles in cyclic pursuit," in *IEEE Transactions on Automatic Control*, vol. 49, no. 11, pp. 1963–1974, Nov. 2004.

[58] M. Ji and M. Egerstedt, "Distributed coordination control of multiagent systems while preserving connectedness", in *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 693–703, Aug. 2007.

[59] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling with dynamic deadlines," in *IEEE Transactions on Mobile Computing,* vol. 6, no. 4, 2007.

[60] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: enabling mobility in sensor networks," in *IPSN,* 2005.

[61] http://www.k-team.com/robots/khepera/index.html

[62] http://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf

[63] http://www.ti.com/lit/ds/symlink/cc1000.pdf

[64] P. Santi, "Topology control in wireless ad hoc and sensor networks," in *ACM Computing Surveys (CSUR)* vol. 37 iss. 2 2005, pp. 164 - 194.

[65] A. Kansal, D.D. Jea, D. Estrin, and M.B. Srivastava, "Controllably mobile infrastructure for low energy embedded networks," in *IEEE Trans. Mobile Computing,* vol. 5, no. 8, pp. 958-973, Aug. 2006.

[66] X. Jiang, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks," in *Proc. IPSN*, Apr. 2005, pp.463 - 468.

[67] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, " Power management in energy harvesting sensor networks," in *ACM Tran. on Embedded Computing Systems*, vol. 6, no. 4, Sept. 2007, article no. 32.

[68] C. Alippi and C. Galperti, "An adaptive system for optimal solar energy harvesting in wireless sensor network nodes," in *IEEE Tran. on Circuits and Systems*, vol. 55, no. 6, pp. 1742 - 1750, July 2008.

[69] G. Xing, T. Wang, W. Jia, and M. Li, "Rendezvous design algorithms for wireless sensor networks with a mobile base station," *Proc. ACM MobiHoc*, pp. 231-240, 2008.

[70] D. Jea, A.A. Somasundara, and M.B. Srivastava, "Multiple controlled mobile elements (data mules) for data collection in sensor networks," in *Proc. IEEE DCOSS*, 2005.

[71] S. Jain, R. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks," *Mobile Networks and Applications*, vol. 11, pp. 327-339, 2006.

[72] D.K. Goldenberg, J. Lin, and A.S. Morse, "Towards Mobility as a Network Control Primitive," in *Proc. ACM MobiHoc,* pp. 163 - 174, 2004.

[73] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks," in *Proc. ACM MobiCom,* 2005.

[74] B. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley. "Mobility improves coverage of sensor networks." in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing,* 2005, pp. 300 - 308.

[75] Y. Yan and Y. Mostofi, "Co-Optimization of communication and motion planning of a robotic operation under resource constraints and in fading environments," in *IEEE Transactions on Wireless Communications*, vol. 12, iss. 4, Apr. 2013, pp. 1562 - 1572.

[76] F. El-Moukaddem, E. Torng, G. Xing, and S. Kulkarni, "Mobile relay configuration in data-intensive wireless sensor networks," in *IEEE Transactions on Mobile Computing,* vol. 12, no. 2, Feb. 2013, pp. 261 - 273.

[77] A. Karr, *Point processes and their statistical inference*, 2nd Ed. Marcel Dekker

Inc., 1991.

[78] B. Phelan, P. Terlecky, A. Bar-Noy, T. Brown, and D. Rawitz, " Should I stay or should I go? Maximizing lifetime with relays," in 8^{th} *IEEE DCOSS*, May 2012.

[79] N. Heo and P. K. Varshney, "Energy,efficient deployment of intelligent mobile sensor networks," in *IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans,* vol. 35, no. 1, Jan. 2005, pp. 78 - 92.

[80] Q. Dong, "Maximizing system lifetime in wireless sensor networks," in *Proc. IPSN*, 2005.

[81] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," in *Ad Hoc Networks,* vol. 6, issue 4, 2008, pp. 621-655

[82] Y. Yan and Y. Mostofi, "Co-Optimization of Communication and Motion Planning of a Robotic Operation under Resource Constraints and in Fading Environments," *IEEE Transactions on Wireless Communications,* Vol. 12, pp. 1562 - 1572, 2013.

[83] C. C. Ooi and C. Schindelhauer, "Minimal energy path planning for wireless robots," *Mobile Netw. Appl.,* Vol. 14, No. 3, pp. 309-321, 2009.

[84] L. Armijo, "Minimization of Functions Having Lipschitz Continuous First-Partial Derivatives," *Pacific Journal of Mathematics,* Vol. 16, pp. 1-3, 1966.

[85] E. Polak, *Optimization Algorithms and Consistent Approximations,* Springer-Verlag, New York, New York, 1997.

[86] A.E. Bryson and Y.C. Ho, *Applied Optimal Control*, Hemisphere Publ. Co., 1975.