Université des Sciences et Technologies de Lille – Lille 1

Département de formation doctorale en informatique **École doctorale SPI Lille**
UFR IEEA

Mobile wireless sensor network architecture: Applications to mobile sensor deployment

THÈSE

présentée et soutenue publiquement le 11/10/2013

pour l'obtention du

Doctorat de l'Université des Sciences et Technologies de Lille
(spécialité informatique)

par

Milan Erdelj

Composition du jury

Président :	Jean-Pierre Richard (Ecole Centrale de Lille, France)
Rapporteurs :	Gianni Di Caro (IDSIA, Switzerland) Thomas Noël (University of Strasbourg, France)
Examinateur :	Angela Schöllig (University of Toronto, Canada)
Directeur de thèse :	David Simplot-Ryl (Inria Lille – Nord Europe, France)
Co-Encadrant de thèse :	Tahiry Razafindralambo (Inria Lille – Nord Europe, France)

Inria Lille – Nord Europe

Numéro d'ordre: 41197

Acknowledgments

I would like to express my sincere gratitude to my Ph.D. advisors, David Simplot-Ryl and Tahiry Razafindralambo, for the continuous support of my Ph.D research, for their enthusiasm, motivation and immense knowledge that they shared with me. I am grateful for their inspiration, their good ideas and the advice which they offered me.

A very special thanks goes out to Nathalie Mitton, whose encouragement and motivation helped me during the three years of my research. I have always admired her dedication, hard work and insightful comments which helped me greatly.

Furthermore, I would like to thank the members of my thesis jury, professors Jean-Pierre Richard, Gianni di Caro, Thomas Noël and Angela Schöllig, for their opinion on this dissertation, interesting questions and helpful suggestions on how to further improve my work.

I am hugely grateful to my colleagues and friends Enrico Natalizio and Jovan Radak, who were there when I needed help, support, guidance and advice, both in professional and in personal life. Also, I would like to thank my colleagues, Ibrahim Amadou, Roudy Dagher, Thierry Delot, Rim Driss, Tony Ducrocq, Arnaud Fontaine, Nicolas Gouvy, Natale Guzzo, Khalil Hammami, Fadila Khaddar, Kalypso Magklara, Karen Miranda, Riccardo Petrolo, Roberto Quilez, Gabriele Sabatino, Loïc Schmidt, François Serman, Anne-Sophie Tonneau, Julien Vandaële and Dimitrios Zorbas, for the stimulating discussions, motivation and all the fun that we had together. Dealing with complicated administration procedures during my stay in France would not be that easy without the invaluable help of Anne Rejl.

Special thanks to my dear friends and flatmates, Benjamín Torres, David Fuertes, Kate Price and Noémie Pichon, for their encouragement, inspiration, motivation, help and support which they offer me every day.

In the end, I would like to thank my family for the encouragement and love they shared with me throughout my entire life, for believing in me and for supporting my decisions. I dedicate this thesis to them.

Abstract

The advances in mobile robotics allow us today to add the mobility concept into many different classes of Wireless Sensor Networks. The deployment of mobile sensors is possible and useful in many application scenarios, ranging from the environmental monitoring and public safety applications, to the industry, healthcare and military applications. Two topics are elaborated in this thesis: networked robot middleware design and a set of approaches for mobile robot deployment in the context of wireless sensor networks. The middleware proposed and described in this thesis allows the user to easily implement different types of deployment algorithms for mobile robots. In the second part of the thesis, three main problems are presented and analyzed. The first is the problem of improving the quality of service with the use of mobile robotic networks, which is solved with the use of deployment algorithm that improves the performance of a multimedia communication by using an intrusive methods to gather the necessary transmission quality evaluation metrics. Second problem is the coverage of the point of interest with mobile robots, where the fixed base station is placed inside the field of interest, while the available mobile robots cover the point of interest and relay the information about it towards the base station in a multi-hop manner. The third problem is the point of interest discovery and coverage with the use of mobile robots which follow concentric circular paths to explore and cover the field of interest, and, by adjusting the movement velocity, they satisfy the constraints on point of interest coverage and connectivity with the base station.

Résumé

Les progrès de la robotique mobile nous permettent aujourd'hui d'ajouter la notion de mobilité dans plusieurs classes de réseaux de capteurs sans fil. Le déploiement de capteurs mobiles est possible et utile dans de nombreuses applications, comme la surveillance de l'environnement, les applications dans l'industrie, dans la santé et le domaine militaire. Le terme robot mobile peut représenter n'importe quel type de robot avec la capacité de modifier sa position. Cette notion inclut une vaste gamme de robots industriels utilisés dans les lignes de production. Dans le contexte spécifique de cette thèse, l'attention se focalise uniquement sur les robots mobiles et plus particulièrement les véhicules autonomes dont les mouvements ne sont pas limités par leur taille physique. Ainsi, un robot ou un groupe de robots mobiles peuvent être utilisés pour explorer des environnements inconnus et effectuer une variété de fonctions. La mobilité du robot dans le contexte des réseaux de capteurs, nous permet de résoudre les problèmes qui ne pourraient pas être résolues dans un cas statique. Les robots mobiles permettent d'augmenter la robustesse du réseau en remplaçant des nœuds de capteurs et de s'adapter aux environnements inconnus ou dynamiques. Deux thèmes sont abordées dans cette thèse : la conception d'un intergiciels pour les réseaux de robots mobiles et un ensemble d'approches pour le déploiement de robots mobiles dans le cadre de réseaux de capteurs sans fil.

L'intergiciel proposé et décrit dans cette thèse permet à l'utilisateur de facilement mettre en œuvre différents types d'algorithmes de déploiement pour les robots mobiles. Il permet de déployer une application sur la station de base centrale qui permet à un utilisateur de rassembler toutes les informations captées par la flotte de robots. L'application de la station de base permet à un utilisateur d'envoyer des commandes à un groupe ou à un robot, introduisant ainsi la commande manuelle en option dans le réseau robotique. L'intergiciel présenté dans ce travail est dédié à être utilisé avec des robots mobiles Wifibot. Il permet réaliser plusieurs tâches. Tout d'abord, il interagit avec le microgiciel du robot pour piloter les moteurs des roues et recueille les informations concernant la sortie du capteur et de l'état de la batterie. Deuxièmement, il gère la communication avec d'autres robots et les stations de base du réseau. Troisièmement, il traite les informations sur l'environnement et les messages reçus des voisins dans le réseau. Enfin, il réagit et il s'adapte de manière rapide et fiable pour aux événements de l'environnement.

Dans la deuxième partie de la thèse, trois problèmes sont présentés et analysés : le problème de l'amélioration de la qualité de service avec l'utilisation des réseaux robotiques mobiles, la couverture du point d'intérêt avec des robots mobiles et la découverte de points d'intérêt et leur couverture avec l'utilisation des robots mobiles. Le premier problème est résolu avec l'utilisation de l'algorithme de déploiement qui améliore les performances de la transmission multimédia. Cet algorithme utilise une méthode intrusive pour réunir les métriques de qualité de service. Ensuite, l'attention est focalisé sur l'application des réseaux de capteurs sans fil est la surveillance de l'environnement. Au lieu de surveiller toute la région, couvrir seulement un ensemble de points d'intérêt spécifiques accroît les performances du réseau et réduit le coût de déploiement. Nous faison l'hypothèse que la station de base fixe est placé à l'intérieur du domaine d'intérêt, tandis que les robots mobiles disponibles couvrent le point d'intérêt et relayent l'information vers la station de base. L'approche pour résoudre le dernier problème est basée sur le mouvement continu et à vitesse variable de capteurs mobiles, qui suivent des trajectoires circulaires concentriques afin d'explorer et de couvrir le domaine d'intérêt. En se déplaçant constamment, les capteurs exécutent la tâche de découverte de l'environnement et, en ajustant la vitesse de déplacement, ils répondent aux contraintes de la couverture et la connectivité avec la station de base. L'algorithme installé sur tous les capteurs mobiles est distribué et introduit une nouvelle technique de calcul de la vitesse en fonction des informations disponibles à partir des capteurs dans le voisinage à un-saut. Ces algorithmes de déploiement de robots mobiles ont prouvé leur faisabilité à travers de nombreuses simulations ainsi que dans la mise en pratique en s'appuyant sur l'intergiciel proposé.

Dedicated to my parents.

Contents

1 Introduction **1**
1.1 Basic concepts in WSN 1
1.2 Mobile robots and WSN 3
1.3 Mobility induced issues 5
1.3.1 Communication 5
1.3.2 Infrastructure based problems 6
1.3.3 Robot robustness, heterogeneity and scalability 6
1.3.4 Robot and system design 7
1.3.5 Testing 7
1.4 Mobile robot deployment 8
1.5 Structure of the document 10

2 State of the art **11**
2.1 Multi-robot middleware 11
2.2 Deployment applications 14
2.2.1 Improving the QoS with mobile robots 14
2.2.2 Deployment algorithms for Points of Interest discovery and coverage 16
2.3 Conclusion 19

3 Mobile wireless sensor middleware **21**
3.1 General middleware architecture 22
3.1.1 Initial Wifibot configuration 22
3.1.2 Towards the use of networked robots 24
3.2 Implemented middleware for Wifibots 25
3.2.1 Motor control block 25
3.2.2 Localization block 27
3.2.3 Communication block 30
3.2.4 Deployment algorithm block 31
3.3 Deployment control and data acquisition 32
3.3.1 Sockets 33
3.3.2 Message format 34
3.3.3 Multi-hop gradient routing 37
3.4 Discussion 38

4 Deployment applications in mobile WSN **41**
4.1 Improving the QoS with mobile sensors 41
4.1.1 QoS improvement algorithm 42
4.1.2 Experimental results 45
4.1.3 Conclusion 51
4.2 Points of Interest coverage with mobile sensors 52
4.2.1 Preliminaries 53
4.2.2 Deployment algorithm for PoI coverage 55
4.2.3 Static PoI 60
4.2.4 Moving PoI 63
4.2.5 Multiple PoIs 65
4.2.6 Implementation using Wifibots 68
4.2.7 Wifibot energy model 72
4.2.8 Conclusion 75
4.3 PoI discovery and coverage with mobile sensors 76
4.3.1 General assumptions 77
4.3.2 Circular movement analysis 80
4.3.3 Approach evaluation 86
4.3.4 Conclusion 92

5 General conclusion **93**
5.1 Summary of contribution 93
5.1.1 Middleware for networked robots in WSN 93
5.1.2 Mobile sensor deployment applications 94
5.2 Perspectives 96
5.2.1 Short-term 96
5.2.2 Mid-term 96
5.2.3 Long-term 97

Bibliography **99**

List of Publications **109**

List of Figures

1.1 Autonomous deployment algorithm in general. 9

2.1 Middleware layer. 11

3.1 Wifibot mobile robot and the schema of the user control. . . . 23
3.2 An instance of a robot middleware architecture. 25
3.3 Detailed robot middleware architecture. 26
3.4 Motor control block. 26
3.5 Wifibot motor control board [94]. 27
3.6 Localization block. 28
3.7 Incremental localization technique for a differentially driven robot. 29
3.8 Communication block. 30
3.9 Deployment algorithm block. 31
3.10 Flow diagram for TCP and UDP socket communications. . . . 34
3.11 *Hello* and *Data* message format. 35
3.12 Multi-robot control application. 35
3.13 *Control* message format. 36
3.14 Gradient assignment to robots in the network (a) and information routing towards the base station (b). 37

4.1 Different positions of a static relay node (static robot). 43
4.2 Two scenarios with mobile robots. 43
4.3 Delay and jitter for each position of the robot. 46
4.4 Frame loss depending on a robot static position. 47
4.5 Peak signal to noise ratio and percentage of frames with MOS worse than referent for a sliding window of 25 frames. 47
4.6 The evolution of robot's position (in meters) during 40 iterations of the PAA. 48
4.7 End-to-end delay (a, b, c) and cumulative jitter (d, e, f) of a transmitted video during robot movement (in different phases). 49
4.8 Frame loss of a transmitted video during robot movement (in different phases). 49
4.9 Peak signal to noise ratio and percentage of frames with mean opinion score worse than in referent video for a sliding window of 25 frames, during robot movement (in different phases). . . 50

4.10 Trajectory and packet delays for one robot running the moving algorithm. Figure (a) depicts the position of mobile relay during 40 iterations of the PAA. 51
4.11 Trajectories and packet delays for two robots running the moving algorithm. Figure (a) depicts the position of mobile relays during 40 iterations of the PAA. 51
4.12 Example of RNG edge removal. 54
4.13 PoI coverage algorithm illustration. 55
4.14 Example of sensor movement. 56
4.15 PoI deployment algorithm. 57
4.16 Minimization of number of connectivity sensors. 59
4.17 Evolution of sensors' positions depending on time. In this simulation there are 20 sensors with a range of 10 on a square of 100×100. The PoI is located at $p(70, 100)$. 61
4.18 Coverage quality, deployment speed and energy consumption. 62
4.19 Evolution of sensors' position depending on time. In this simulation there is 20 sensors with a range of 10 on a square of 100×100. The PoI is first located at $p'(70, 0)$ and then at $p''(70, 70)$ after $200s$. 65
4.20 Number of covering sensors w.r.t time. Simulation parameters: $R = 10$, 20 sensors including the base station. The simulation lasts $3000s$. A new location of the PoI is chosen every $500s$. . 66
4.21 Sensors' positions with multiple PoIs depending on time. . . . 68
4.22 Number of covering sensors w.r.t time for R-PDA and B-PDA. 69
4.23 Coverage of $p(40, 60)$ with comm. range of $15m$ (6 robots). . . 70
4.24 Coverage of $p(70, 100)$ with comm. ranges of $20m$ and $15m$, respectively (9 robots). 71
4.25 Coverage of $p(50, 40)$ with changing the comm. range from $8m$ to $15m$ (5 robots). 71
4.26 Implementation of R-PDA and B-PDA for two PoI coverage, $p_1(25, 45)$ and $p_2(45, 25)$. 72
4.27 Wifibot deployment with an obstacle. 73
4.28 Speed and power data during the 5m movement. The speed of 140 ticks per 50ms corresponds to the speed of 0.5m/s. 74
4.29 Average power during the movement with different speeds (from 0.1 to $0.9m/s$). 75
4.30 Simulation and implementation results for total consumption with relation to the robot position. 76
4.31 Mobile sensors (S_i) follow the concentric circular paths and cover the PoIs (P_i). Node S represents the data sink. 79

4.32 (a) Mobile sensor with its communication (r_c) and sensing (r_s) ranges. The ratio between communication and sensing range is referred to as path density, D_L. (b) The distance between neighboring paths is set to $2r_s$ in order to minimize the overlap of sensing ranges. 80
4.33 (a) The angle between the barycenter and two sensors on neighboring paths in the moment when they enter (or exit) each other's communication ranges. (b) Communicating sensors' angle difference between n and $n-1$ path levels ($\Theta_{comm}(n, n-1)$), for different values of the path density $D_L = r_c/r_s$. 81
4.34 Inter-contact time for sensors on neighboring paths in the case of single sensor per path. 83
4.35 Inter-contact time for sensors on neighboring paths in the case of multiple sensors per path. 84
4.36 The difference between angular coordinates of sensor S_n and the PoI P_i in the moment when sensing begins. 86
4.37 (a) Distribution of PoIs with relation to the path covering it and (b) the distance between the sensors on the path during the movement. 87
4.38 Percent of PoIs discovered after a certain time for both approaches and different movement velocities. 88
4.39 PoI information delivery time and the percent of reported PoIs to the sink node for circular path approach. 89
4.40 Message loss related to T_{data} in circular path approach. 89
4.41 PoI update time with relation to different movement velocities. 90
4.42 PoI coverage with relation to different movement velocities. . . 91
4.43 Covering time with relation to different number of PoIs. 92

List of Tables

4.1 Summary of the simulation parameters in PoI coverage campaigns. 61
4.2 Simulation and experimentation results for energy model verification 75
4.3 Summary of the simulation parameters in PoI discovery campaigns 86
4.4 PoI distance from the data sink and its covering sensor path level 86

Chapter 1
Introduction

Contents

1.1 Basic concepts in WSN 1
1.2 Mobile robots and WSN 3
1.3 Mobility induced issues 5
1.4 Mobile robot deployment 8
1.5 Structure of the document 10

The advances in mobile robotics allow us today to add the mobility concept into many different classes of Wireless Sensor Networks (WSN). The deployment of mobile sensors is possible and useful in many application scenarios, ranging from the environmental monitoring (volcanic activity [93], forest fire detection [103], pollutants or gas plumes [78]) and public safety applications [39], to the industry (structure [44] and machinery health [85]), healthcare [46] and military applications [25, 51, 95]. In this thesis, the accent is put on the middleware architecture that allows the deployment of mobile sensors, as well as on the analysis of specific deployment applications.

1.1 Basic concepts in WSN

A wireless sensor is a device with limited resources (data storage, processing, energy and transmission means), that measures a physical quantity, processes it and transmits the information about it. The *communication* between wireless sensors represents the exchange of information between them through the use of wireless communication channel. A set of wireless sensors (that can be of the same type or heterogeneous) with the common goal constitutes a Wireless Sensor Network (WSN). The possibility for sensors in WSN to communicate with each other is referred to as network *connectivity*.

Due to the sensors' limited energy resources, wireless sensor networks are prone to sensor failures that affect the operation of the network. The time period measured from the start of the network operation until the energy exhaustion of a sensor in the network is referred to as *network lifetime*. In order to maximize the usability of the network, it is necessary to prolong the network lifetime by designing energy efficient sensor operation and/or *network maintenance* techniques (manual or automatic network servicing). The case of unexpected sensor failures can lead to network disconnections and failures in desired network operation. The ability to cope with sensor failures, and thus to prolong the lifetime and maintain the connectivity, is referred to as network *robustness*.

A set of wireless sensors that constitute a wireless sensor network usually transmits the sensed information towards the specialized device called *data sink*. In order to efficiently transmit the information towards it, the WSN needs to rely on a data acquisition infrastructure, that comprises a method of communication between individual sensors and a *routing protocol* that defines the information transmission from an individual sensor towards the data sink. If a sensor cannot communicate directly with the data sink, the information is transmitted in a *multi-hop* manner, with the use of intermediate sensors that act as relays for information transmission. The communication in the network can be represented with a *communication graph*, where each sensor is represented as a vertex and each communication link between sensors is represented with an edge in a resulting graph. In order to cope with the complexity of a resulting graph (network *scalability*), *graph reduction* methods can be used.

Notions of *deployment* and *deployment objective* are hard to define since they depend on the actual application of the WSN. Furthermore, the concept of *deployment quality* strongly depends on the deployment goals, sensor and environment characteristics. Indeed, different deployment solutions can be envisaged in the case of sensors with limited communication and movement capabilities, or the absence of knowledge regarding the deployment environment. In other words, the quality of the deployment is not comparable in the case of mobile sensors with total knowledge of the environment and the availability of the absolute localization techniques and in the case of absence of any localization technique followed by the completely unknown deployment environment. Bearing in mind that the majority of the applications focuses on a certain type of event (or a set of events) monitoring and data acquisition, the deployment can be referred to as the process of optimally placing a group of sensors (static and/or mobile) in an environment containing the events of interest. The amount of resources used during the deployment is referred to as *deployment cost*. In the context of environmental monitoring, deployment quality and cost notably depend on the environment, covered area, deployment speed, and energy consumption, etc.

1.2 Mobile robots and WSN

The term *mobile* robot could represent any type of robot with movable parts that is capable of changing its position in a certain way. That notion includes the wide range of industrial robots used in production lines. In the specific context of this thesis, the attention is focused only on mobile robots that represent autonomous vehicles whose movements are not limited by their physical size. Hence, an individual or a group of mobile robots as autonomous vehicles can be used to explore unknown environments and perform a variety of functions (often referred to as multi-robot systems). They are classified in three large groups depending on their operating environment:

1. *Ground vehicles (land-based robots) [14, 32].* This type of vehicle is designed to operate while maintaining constant contact with the ground. Sections 4.1 and 4.2 analyze the deployment algorithms for this type of robots.

2. *Aerial vehicles (flying drones).* The use of flying drones allows us to avoid the problems of physical obstacles encountered by the ground vehicles in the deployment field, simply by flying over them. This type of mobile robots is often used for the applications of area surveillance and target detection/tracking [10, 26, 56]. The use of flying drones is supposed and analyzed in Section 4.3.

3. *Surface and underwater vehicles (aquatic robots).* This group of robots introduces a specific set of challenges due to the deployment environment characteristics. Being deployed in the water, these robots face the problem of communication in the aquatic medium and the problem of localization, since the global localization techniques are not available if the robot is submerged under water [66, 101, 104].

A random deployment of static WSN that requires a number of sensors that is greater than optimal, which impacts the overall deployment cost. One of the solutions to this problem is the conjunction of a classic static WSN with a set of mobile nodes [50]. In this context, the role of mobile robots is twofold. First, the set of mobile robots serves as mobility provision agents. In this case, the goal is to physically displace already deployed static sensors in the deployment field and thus increase the deployment quality. However, it cannot be guaranteed that in every WSN application, such an approach would improve the quality of the deployment while minimizing the deployment costs. In a hostile environment, it is worth considering the trade off between the cost of introducing the mobility versus the additional set of static nodes in the network.

Another role of the mobile robots in the interaction with the WSN is automated sensor network servicing. Although not directly involved in sensing and acquisition tasks, a set of mobile robots can influence the deployment quality by replacing damaged or discharged sensors [75]. Furthermore, it can even behave as a mobile recharging station, thus prolonging the lifetime of the network.

Including the robot mobility in the WSN deployments allows us the following:

- the possibility to resolve problems that could appear in the network that are not solvable by static nodes,
- the increased the network robustness by automated sensor node replacements,
- the adaptability to unknown or dynamic environments.

There are three major types of mobility that are considered in the context of WSN/WSAN:

1. *Static (immobile) WSN.* In this first case, the sensors in the network do not possess any kind of locomotion or displacement capability. Furthermore, there are no entities that could interact with or displace sensors in the network.

2. *Assisted mobility.* This second type of mobility assumes a sensor network composed of static sensors that are unable to move autonomously. However, these sensors are usually mounted on different types of mobile agents (robots, vehicles, animals, people, etc.) that provide them with mobility. Although this kind of movement pattern is not controllable, it can be mathematically modeled.

3. *Controlled mobility.* Finally, the third type of sensors mobility assumes that the network is entirely or partly composed of mobile sensors (mobile robots) that can be manually or self-controlled. This type of mobility allows us to increase the deployment quality in a way that suits the best to the user or the application of the network. Controlled mobility has received much attention in recent years due to the ever expanding possibilities for different applications, notably area exploration and rescue missions [11, 62, 82].

In this thesis, the attention is focused on the controlled mobility of robots during the network deployment.

1.3 Mobility induced issues

1.3.1 Communication

The most important issue in the wireless network is the communication aspect that has its own special characteristics. Two different communication paradigms can be considered in this case: direct and indirect communication. The first way of communication is the explicit one, two sensors represented as sensor nodes in the network can communicate with each other through an established one or multi-hop wireless links in the case of short sensors' communication range.

The second way of communicating is the indirect one – the communication through the signs in the environment. This way of communication is bio-inspired, where instead of creating a direct wireless link, sensors change the deployment environment (by leaving signs, pheromones, etc.) in the way that will be understood by other sensors in the deployment. An example of indirect communication in the set of servicing robots is the communication through messages left at the serviced nodes that can be read by other robots that will pass by. It is worth noting that in this way of communication, there is no need for constant connectivity maintenance.

Another essential problem that arises in the implementation of the robotic wireless networks is due to the wireless channel properties (such as environment interferences, propagation and fading effects, overhearing, etc.) [16, 60]. The effect of a wireless channel in certain cases makes impossible for two physically close sensors to establish a wireless link. Likewise, in some cases the link can be established even if the distance between two sensors is way beyond the expected maximal communication range. Therefore, the robot deployment becomes a highly complex task if the communication medium properties are taken into account. Resolving these problems leads to unnecessary energy depletion in the WSN.

The issues caused by the wireless channel, combined with the mobility introduced in the sensor network, that make links between sensors change rapidly and unexpectedly, highly affect the availability of communication paths and the network topology. The robot deployment algorithms proposed and analyzed in this thesis, have the common goal to preserve the communication between the robots all along the deployment procedure and the network operation.

1.3.2 Infrastructure based problems

An integral part of a sensor network is the establishment of the data acquisition infrastructure. Due to the dynamic nature of the robot deployment, the data acquisition infrastructure must be auto-adaptable to wireless channel, interferences and environment conditions.

The cost of the auto-adaptable network infrastructure becomes an obstacle in remote and large construction sites where the robotic network is used for a structure and machinery health monitoring. In certain military applications that require fast and reliable response to environmental changes, the network infrastructure reaction time represents one of the major issues [6]. The approaches described in this thesis rely on the ad-hoc multi-hop communication among the mobile robots, that easily and efficiently resolves aforementioned infrastructure based problems.

1.3.3 Robot robustness, heterogeneity and scalability

Another major obstacle in the widespread robot deployments is the reliability of mobile robots in the presence of environmental disasters. Robot failures lead to the loss of gathered information and possible network disconnections. These problems could be overcome with the appropriate information routing techniques, however, they do not guarantee that the network will be able to overcome all the problems. In the practical implementations of robotic sensor networks, robots that are used can be heterogeneous, and therefore, prone to different sets of environmental hazards. Furthermore, robot robustness is usually examined on the scale of individual robot. In the practical implementations, failures that appear are usually linked to more than one robot that operates in the desired environment. These failures are sometimes environmentally provoked, meanwhile some other times they are induced by the interaction between the robots. In any case, they are not trivial to detect and overcome.

Standard WSN data acquisition techniques assume a dense sensor network which is still not the case in robotics. The greatest obstacle to achieve in a dense robotic network is the robotic unit price – robotic sensors cannot be considered as cheap sensing devices with limited storage, energy, and processing power (which is a general assumption in WSN). Due to the sparsity, individual node failures can lead to greater disasters in the networks that could be expected with reasoning inherited from WSN principles. As stated in this thesis, the development of distributed approaches for mobile robot deployment copes with the problems of robot failures and thus improves the network robustness.

1.3.4 Robot and system design

Robot network design generally aims at finding the balance between the simplicity of the individual robotic sensor units and the complexity of the final system that comprises networked robots, but the communication and control flow as well. A number of problems arise due to the lack of understanding of the final application goals and needs, along with the compromise between the highly specialized and generalized modular components used in the construction of the robotic sensors. Modular and reusable components generally reduce the effort and work needed to conceive and implement mobile robots, and in this manner reduce the development costs linked to new component testing. On the contrary, specialized components used in the construction of mobile robots provide the sensor network with the increased suitability and higher performance in the desired application.

The work on networked robotic middleware and deployment algorithms presented in this thesis, insists on the design simplicity. A simple middleware specifically dedicated for the platform that will be used on, achieves faster response and better overall performances. Similarly, a simple deployment algorithm does not introduce an important overhead and thus leaves the bandwidth and computational power for the purposes of data acquisition.

1.3.5 Testing

The final and fundamental component of any system integration is testing. Robotic networks dedicated for information acquisition applications in the context of WSN, suffer from the same problem that strikes any product in development – the compromise between thorough testing and the necessity to move a designed system to the market quickly. Full system testing is impossible to achieve, above all in the design of mobile robots dedicated for the aforementioned applications, since it is impossible to envisage all the possible situations and hazards that could appear in the real world.

First level of testing is the testing of the used components in the construction of the robotic platform in order to verify their functionality as stated in their specification. When the complete deployment system is integrated, the next level of testing focuses on the functionality of the system itself. This testing phase can take a long period of time in order to ensure the reliability and robustness of the single components integrated in the complex system. The last and the critical part of the reliable robotic network is the implementation of the internal self-monitoring techniques. These techniques allow the system (as well as the individual robots) to detect, recognize, and solve a set of potential problems that may arise in a real world implementation.

The middleware described in this thesis provides the user with the capability of gathering the real-time data by directly connecting and observing the output of each individual robot, which gives the user the information regarding the state of the network. Furthermore, the user can send individual commands to control robots during the testing phases, in order to predict and correct the robot behavior in practical implementations.

1.4 Mobile robot deployment

In general, the sensors are usually deployed in a random or deterministic manner. Former are hardly feasible in any other situation than the small network deployed in the known environment. The necessity of larger sensor networks in an unknown environment leaves us with the random deployment as the only choice. The random deployments in an unknown environment is usually done by scattering the sensors over an area of interest, such as volcano or forest, from an aircraft [57, 79, 102]. As expected, certain number of sensors deployed in such a manner will not be usable due to failures caused by the aircraft scattering. In order to guarantee the coverage and connectivity of such a deployed network, the number of sensors deployed must be larger than the minimal necessary number, which increases the overall costs of the network.

All the approaches to sensor deployment that include controlled mobility can be classified into two deployment schemes: centralized and distributed deployments. The centralized approach assumes the existence of a central entity that is not necessarily a part of the set of mobile sensors. The central entity in this type of deployment techniques collects all the necessary information about all the sensors in the network and the deployment environment itself. Moreover, it processes this information regarding the goal of the deployment and chooses the optimal positions for each sensor in the network. Finally, it directs each individual sensor towards its future destination. This type of approach can achieve excellent results in the static environment, since the optimization algorithms can be applied in order to achieve the optimal deployment. However, the necessity of the global network information acquisition imposes high computational overhead in energy, time, and storage space, that collides with the concept of WSN composed of cheap sensors with limited processing power. Furthermore, due to the centralized approach depending on central entity, the complete network is dependent on the errors and failures that can happen in the central entity, which makes the network highly vulnerable. Finally, the scalability of the network in this case represents another huge problem, since the central entity has to manage ever increasing amount of information in real time. The aforementioned problems, together with the dynamics in the most practical environments that increase the complexity of the computation and communication, make the centralized approach infeasible in most of practical applications.

On the other hand, the distributed approach easily copes with the problems of the dynamic and unpredictable environments, as well as the problem of scalability, by allowing each robot to calculate its own behavior and mobility pattern depending on the perceived local neighborhood and environment information. In this manner, the computation complexity is reduced to a limited set of locally perceivable neighboring sensors, whatever the size of the complete network is. The goal of the distributed deployment techniques is to combine all the movement decisions that are brought locally and combine them in order to approach the optimal solution achievable by the centralized approach. The drawback of the distributed approach is that the lack of complete knowledge makes impossible to achieve the optimality. However, bearing in mind the environment conditions in practice, followed by the absence of scalability and computational complexity issues, this thesis focuses only on the distributed approaches to multi-robot WSN deployments.

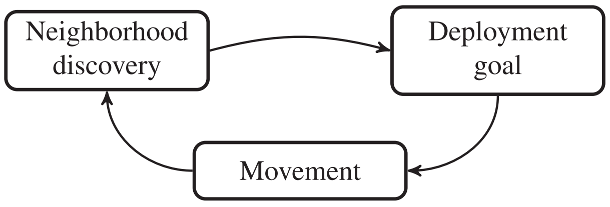


Figure 1.1: Autonomous deployment algorithm in general.

All the different deployment approaches that are examined in this chapter and that depend on the specific application of WSN, can be described with one generalized distributed deployment scheme, shown in Figure 1.1. This scheme is iterative and comprises three essential parts: neighborhood discovery, deployment goal calculation and movement towards the computed target point. In the neighborhood discovery part, the robot transmits its own information (position, energy level, etc.) and receives the local information from neighboring robots in the deployment field. This information is used in order to construct the communication graph based on a graph reduction technique. The second part of the scheme employs different probabilistic or geometrical techniques to choose the best potential displacement target point while applying the connectivity preservation constraints (if any). Finally, the third part executes the movement towards the selected point.

Deployment algorithms for the specific set of applications proposed in this thesis, are based on the same basic structure as in Figure 1.1. The simplicity of the deployment algorithm allows easy implementation on various robotic platforms. Although being simple, this basic scheme describes a significant percent of specific deployment algorithms used in various practical mobile network applications.

1.5 Structure of the document

Two topics are elaborated in this thesis: networked robot middleware design and a set of approaches for mobile robot deployment in the context of wireless sensor networks. The state of the art in the networked robotic middleware design, followed by the analysis of the approaches and techniques of mobile robot deployment are presented in Chapter 2.

Chapter 3 is dedicated to the first topic – the middleware architecture for a networked robot. First, the Wifibot mobile robots are introduced, together with their pre-installed middleware application that allows manual guidance and basic communication with the robot. Then, the proposed and implemented middleware is presented and explained in detail, following by the communication paradigm and user interaction and control over the robotic network. Finally, existing flaws and space for improvements are identified and possible solutions are discussed.

Second topic of the thesis, deployment algorithms for mobile robots and the applications of mobile wireless sensor networks are proposed in Chapter 4. In this chapter, three main problems are evoked and analyzed: improving the quality of service with the use of mobile robotic networks, coverage of the point of interest with mobile robots and point of interest discovery and coverage with the use of mobile flying sensors (drones). These mobile robot deployment algorithms proved their feasibility through numerous simulation campaigns as well as in the practical implementation relying on the proposed middleware.

Finally, conclusions regarding the work presented in previous chapters are drawn and future works that could improve the overall performance of the middleware architecture and deployment algorithms are proposed in Chapter 5.

Chapter 2
State of the art

Contents

2.1 Multi-robot middleware 11
2.2 Deployment applications 14
2.3 Conclusion 19

In this chapter, the state of the art in the field of robot middleware solutions is analyzed in Section 2.1. Section 2.2 provides the reader with the current set of applications and algorithms for mobile wireless sensor network deployment.

2.1 Multi-robot middleware

A robot middleware is defined as a class of software technologies designed to help manage the complexity and heterogeneity inherent in distributed systems [3]. As shown in Figure 2.1, it is a software layer that is positioned above the operating system and below the application layer. Its role is to simplify the software design by hiding the unnecessary low-level implementation details from the user, allow the code reusability and manage the heterogeneity of the hardware.

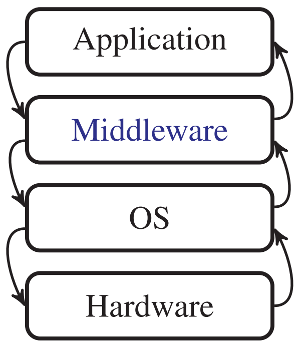


Figure 2.1: Middleware layer.

Parker proposes one of the first works that focus on an architecture for mobile multi-robot management in [68]. In her thesis, Parker describes a software architecture that facilitates the cooperation between heterogeneous mobile robots. The proposed middleware allows the fleet of robots to perform a specific task and to respond to unexpected environmental changes by relying on the set of preprogrammed behaviors. The architecture in this thesis focuses more on the communication aspects between the robots and the ease of basic algorithm implementation for wireless mobile sensor applications (Chapter 4).

Further development of robotic middleware continued with Miro framework [88]. It allows a distributed robot control and comprises 3 layers: the device layer that provides interface abstractions for the hardware components and that is platform dependent, service layer that provides services by using CORBA [20] interface definition language, and the framework layer that provides the user with modules for robot control, localization, path planning, etc. In the same year, in [29] a hierarchical framework is proposed by the authors. This work mainly focuses on programming aspects and the formal definition related to the proposed high level language. Compared to the work in this thesis, the proposed work only describes some basic functionality which does not include communication paradigms.

In [23], authors propose a control software for multiple robot architectures. This work mainly focuses on high level primitive to create and provide a software interface to ease the implementation of robot deployment algorithms. The proposed framework uses a Java virtual machine. The work in this thesis does not rely on any specific software and use the robot low level primitives to build the robot cooperation and facilitating the programming methods by giving an access interface to each block. In [41], authors propose a QoS-aware middleware that provides real-time support for robotic applications. The proposed operating environment and development framework architecture is too resource demanding for the use in the context or WSN. Middleware proposed in this thesis provides the user with a simplified architecture that facilitates the implementation of mobile sensor wireless applications. In [43] the authors proposed a layered cooperative middleware of mobile system. The proposed work is very interesting but it is not suitable for mobile wireless sensor networks platform. Indeed, the layered architecture imposes the sue of some specific tools such as the OSI model imposes the use of some specific protocol that may not be suitable for mobile wireless sensor networks. In [61], the authors describe a multi-agent based solution to control and coordinate teamworking mobile robots. The proposed work is very interesting and divides the architecture into three different blocks: physical, control and coordination. However, it does not describe in detail the communication modules which is one of the goals of this thesis.

Other approaches to robotic middleware design and robotic frameworks such as Pyro [8], Player [19], Claraty [65], Marie [22], Open RTM [67] and Orocos [77], can be found in survey papers of Mohamed et al. [59] and Sanfeliu et al. [73].

A programming environment that facilitated the robot application programming is proposed in [8]. Its simplicity and stability makes it useful for teaching purposes. [19] proposes a concept that is similar to the middleware in this thesis, where a client user application communicates with the server on the robot via TCP socket connection. However, the use of TCP connection is not suitable for applications that need to be reliable and auto-adaptable (as discussed in Section 3.3). In [65], authors propose the arhitecture that allows the implementation of different applications for mobile robots. The architecture is based on a client-server relationship between the functional (path planning, motion, navigation, etc.) and decision (system status, resource monitoring, environmental information, etc.) layers. The architecture proposed in [22] consists of core, component and application layers. The core layer is in charge of low-level functions of the robot hardware, such as input/output control and communication. Other two layer are in charge of application and robot component implementation. The robot deployment algorithm can be implemented in the application layer, while the communication between the robot components can be output on communication port and accessed externally. However, it is dedicated for development and robot software integration, white the use for multi-robot cooperation is put into second plane. Similarly, [67] proposes a component based robot software development platform. All the robot hardware part and software blocks are represented as components that can exchange data between each other. A complete robotic system is then constructed by combining and connection the components together, but it is not specifically dedicated to design of the network that comprises multiple robots. In [77], authors propose a toolkit consisting of components that are able to be run on real-time operating systems. It contains libraries that provide the user with control, movement and filtering components, that facilitate the robot software design and implementation. However, it does not focus on the interaction among the robots in the network, which is the focus of the work in this thesis.

One of the mostly used robotic middleware is Robotic Operating System (ROS) [72]. It is a component-based platform supporting a client-server scheme for control flow and a publisher-subscriber scheme for data flow. ROS is actually not an operating system but rather a middleware that consists of nodes, messages, topics and services. Nodes communicate among each other by publishing messages and subscribing to published messages. However, its size and complexity make it infeasible for mobile devices with constrained memory and computational power. The interested reader can refer to [28] for a more detailed and recent survey on robotic middleware.

The middleware implementation proposed in this thesis covers the basic aspects needed by the mobile robot deployment applications. Although there are some missing features that are present in other existing solutions, the middleware presented here is simple and portable, which makes it a good choice for small, energy and memory constrained robotic platforms used in swarm robotics.

2.2 Deployment applications

This section tackles the state of the art in the field of mobile robotic network dedicated to improving the video transmission quality with mobile robots and deploying a network of mobile robots in order to cover the points in known and unknown environment of interest.

2.2.1 Improving the QoS with mobile robots

The literature lacks an algorithm for networked robots deployment that aims to improve multimedia communications parameters for delivering QoS, such as throughput, delay or jitter. Several proposals regarding the improvement of the same parameters for other kind of wireless networks exist and few testbeds of multimedia networks have been deployed in the recent past. In the following two subsections, we will scan the literature in order to show recent works on the two cited topics: improvement of transport layer parameters and testbed of multimedia sensor networks.

Improvement of transport layer parameters. Transport layer in wired and/or wireless networks is responsible for the provision of end-to-end QoS between two clients [87]. Thus, the transport layer adaptively tunes its behavior in order to improve the performance of relevant QoS parameters. In the context of wireless networks, some interesting works for improving QoS parameters through transport layer protocols have been recently proposed for Wireless Mesh Networks (WMN) [37] and Mobile Backbone Networks (MBN) [81].

In [105], authors propose a new performance metric called multi-hop traffic-flow weight (MTW), which is based on the number of mesh clients, the number of gateways, traffic demand from mesh clients, locations of gateways, and possible interference among gateways. Authors also design a new gateway placement scheme around this new performance metric. In [52], authors propose a method to deploy a certain number of WMN gateways depending on the traffic demand and the interference model in the network, in order to maximize the throughput of the WMN. In [64], authors propose a new transport protocol based on a congestion control that allows nodes to adapt their transmitting rates in a distributed way and to keep track of dynamic multi-hop network characteristics in a responsive manner. The protocol achieves very high throughput with low delays. The extension of all these works to different kind of wireless networks would be possible, but, since they have been proposed for WMN, they do not consider the mobility of nodes and they can not be used to dynamically react to changes in the network or the traffic load.

In [80] authors determine the best positions for Mobile Backbone Nodes (MBNs) in order to a) maximize the minimum throughput of the network composed of nodes with restricted mobility, b) maximize the aggregated throughput of these nodes. They formulate and solve the two problems. Unfortunately the solution of the problem requires a central point of computation and centralized processes. The two heuristics introduced in their work, which are distributed and use only local information, do not achieve satisfying results in comparison to the optimal centralized solution. Therefore, even if the theoretical scheme is useful, the proposed algorithms are practically useless. In [71] authors present the optimization problem that aims to maximize the weighted throughput of a node that achieves the lowest weighted throughput among all nodes. Furthermore, they develop an algorithm for the optimal placement of the relay node. The work assumes to have one single mobile node to control and deploy, and this is more restrictive than the case tackled in this thesis with all mobile nodes.

Testbed of Multimedia Sensor Networks. Several testbeds have been proposed in order to test and evaluate the behavior of a multimedia sensor network, for a comprehensive survey on this topic, a reader should refer to [1]. Specifically, [24] extends the capabilities of Explorebots to comprise a mobile network experimentation testbed. Explorebots are expandable, vision- and sensor-equipped wireless robots built around MICA motes [40]. The testbed developed through Explorebots supports experimental analysis of protocols for mobile multi-hop networks. The authors focused specifically on three research issues: incorporation of localization, target localization, and hybrid routing protocols.

In [42], robots carry motes and single board computers through a fixed indoor field of sensor-equipped motes, all running the user's selected software. Authors of [42] describe three experiments using their mobile testbed. The first experiment demonstrates the network-level irregularity of real physical environment, the second experiment evaluated an acoustic ranging sensor network application, and the third aimed to the creation of a multi-hop network. Authors of [48] present the design and implementation of SensEye, a multi-tier network of heterogeneous wireless nodes and cameras, in order to show that a camera sensor network containing heterogeneous elements provides numerous benefits over traditional homogeneous sensor networks. They also propose an experimental evaluation of their prototype in a surveillance application that comprises three tasks: object detection, recognition and tracking.

However, none of the above proposed multimedia testbeds have been tested, specifically for multimedia transmission. The testbed we implemented focuses on perceived quality, throughput, delay and jitter in the communication.

2.2.2 Deployment algorithms for Points of Interest discovery and coverage

This section covers the works focused on three different operational phases of mobile wireless sensor networks:

- exploration of the sensor field and discovery of the Points of Interest (PoI),
- monitoring of and data gathering from the PoIs,
- connection with and data reporting to the base station.

This section analyzes only the most recent works referred to these three main issues.

The attention of this state of the art review is focused on the deployment of mobile sensors, but more interested readers can refer to [12, 76] for static random deployment strategies, to [2, 47, 70] for offline computation of sensor placement and to [54, 89, 99, 102] for complete surveys. The evaluation of the impact of number and placement of heterogeneous resources on performance in networks of different sizes and densities is presented in [100].

Regarding the deployment or placement of mobile sensors, there are mainly three ways of optimization that were previously described in [89]: coverage pattern such as in [91], grid quorum such as in [13] and virtual force based movement such as in [4]. In the virtual force based movement, sensors are repelled or attracted each other by using virtual forces like electromagnetic particles. The sensors move step by step. Virtual forces are computed based on a set or a subset of neighboring sensors and allow the computation of the sensor's next movement. The sensor can undergo attractive forces, for preferential coverage areas, repulsive forces for obstacle avoidance and forces exerted by another sensor. The deployment algorithm proposed in Section 4.2 uses this kind of approach.

The coverage requirement is the primary aim that describes how the sensors have to be deployed over the field. Even if some ways of moving are strongly related to the coverage requirements, it is important to notice that movement and coverage are independent. These two aspects must be decorrelated in order to have simple deployment algorithms. Coverage requirements can be divided into three categories: full coverage [49, 69, 84], barrier coverage [74, 83] and the Point of Interest (PoI) coverage problem [15, 53, 98] which is in the focus of this thesis.

In the following, works that tackle issues of network connectivity, PoI discovery and PoI coverage are analyzed.

Discovery. The exploration of a geographical area by a group of autonomous agents looking for PoI has been widely studied in the literature of robotics. In fact, many variations of this task have already been considered. For the sake of synthesis, only few works from the extensive literature in this area are mentioned, in order to highlight the discovery phase of the robot deployment approach presented in this thesis. The basic difference with respect to the existing works is that, in proposed case, mobile devices exploit their communications capabilities to coordinate in order to fully explore the area and proceed covering the assigned area with no need of further formation control. In literature, there exist works aiming at the same purpose by using a single mobile robot, as in [58], where the authors propose an energy-efficient technique to determine the trajectories to choose in order to minimize the total travelled distance. Of course, also more sophisticated works have been presented for the PoI discovery task. In these works, the number of mobile devices is higher and coordination issues arise. Usually, in the robotic applications, communication capabilities are seen as an unnecessary and unwanted supplement of the basic devices' facilities, as in [34] and [55], where robots do not have any communication facility to exploit, and in [36] where also the perception capabilities of the devices is restricted. Few works consider the possibility for the nodes to communicate and exchange useful information, as in [30], but even in these cases the communication is limited and relayed by a central unit. In the approach proposed in this thesis, the mobile devices use their communication capabilities during all the phases of the algorithm, both to coordinate during the initial formation control and to relay the information about the monitored events towards the base station. Furthermore, robotics works on formation control (please refer to [92] for a survey) assume continuous communications among the devices in order to support the formation control. However, this cannot be always guaranteed. On the contrary, in this work, the assumption is that devices communicate only at the beginning of the discovery phase, in order to coordinate, to avoid overlaps of their coverage areas and to distribute on the whole field.

Connectivity. Regarding the network connectivity, an interesting task is to evaluate the performance of both the upper layer parameters, through the measurement of the communication delay between each node and the base station, and the one-hop communication, such as the inter-contact time among neighboring nodes. A lot of efforts have been already dedicated to Delay Tolerant Networks (DTN) when mobility of nodes is taken into account. In [90], authors show that mobility increases connectivity in k-hop clustered networks, and significantly improves the network lifetime and the power-delay trade-off. As usual when authors aim at demonstrating some interesting properties achieved by letting nodes move, the mobility follows one of the classical schemes, specifically the random walk mobility model with non trivial velocity. Instead, in the approach analyzed in this thesis, we assume that nodes can control their mobility according to surrounding conditions. Even in [31], authors show the positive impact of mobility on the determination of delay-throughput trade off in mobile ad-hoc networks. Even in this case, nodes' mobility is restricted to movement around nodes' own home-points. In [63], the mobility is controlled and nodes move from the initial configuration towards a spatial distribution that increases the network's throughput capacity and decreases the mean packet delay. A similar approach is used in [21] to determine the optimal placement in terms of energy consumption by virtualizing nodes' movements. In these works, upon the determination of the optimal placement, nodes do not have to move further. In the case analyzed in this thesis, the coverage of the area requires nodes to keep moving for the whole network lifetime. An interested reader may refer to [33] for the survey on sensing coverage and network connectivity in WSN.

Coverage. The coverage and monitoring of a PoI, an area of interest or the whole sensor field is a subject covered from both the ad hoc and sensor and the robotics community by using different approaches and by focussing on different aspects. Works dedicated to ad-hoc and sensor networks consider devices such as sensors and actuators, whose computational and execution capabilities are limited, but have the possibility to communicate with each other wirelessly. On the other hand, the works in the field of robotics take smarter devices into account and assume that communications do not have a basic importance in achieving the coverage of the area. Since the topic has been extensively treated by both research communities in recent years, the focus is put on the efforts produced by the ad-hoc and sensor networks community, which are more relevant in the context of the work covered in this thesis. In [17], [89] and [102], authors survey coverage path planning algorithms for mobile robots, movement strategies for improving network coverage and general strategies and techniques for node placement, respectively. Details regarding these works are not provided, since the issue of coverage has been already discussed.

Discovery, connectivity and coverage combined. The swarm intelligence community is very active in coordination of mobile agents by mimicking natural systems. This methodology is useful for making nodes perform the exploration of the field while they maintain connectivity with each other [27]. Usually for these purposes, hybrid network architectures are considered, as in the cited paper that uses two kinds of mobile robots: wheeled and flying. In this thesis, a general assumption is the use of homogeneous devices which are not specialized in a specific task. In a less recent work, the problem of achieving full coverage of a field while preserving 2-connectivity[1] among the nodes is considered [2]. As it is clear, the final objective is the deployment of the nodes, therefore further movements of the nodes after the optimal placement is achieved, are not considered. At last, a work from the robotics community considers PoI discovery and coverage of the network [5]. The discovery phase is performed by mobile robots and driven by a network of radio beacons which assists the robot(s) also during the coverage. On the contrary, the scheme described in Section 4.3 does not consider any additional pre-deployed hardware for achieving the same objectives.

2.3 Conclusion

To conclude the state of the art both on the topic of robotic middleware architecture and the set of deployment applications, it is worth emphasizing that it is important and strongly recommended to work on both topics in order to ameliorate the overall efficiency of the system.

[1]a *k-connected* network is a network where each node is at least connected to k neighbors.

Chapter 3
Mobile wireless sensor middleware

Contents

3.1 General middleware architecture 22

3.2 Implemented middleware for Wifibots 25

3.3 Deployment control and data acquisition 32

3.4 Discussion 38

Multiple robots that coordinate or cooperate with other sensors, robots or human operator, allow the Wireless Sensor and Actuator Networks (WSAN) to perform tasks that are beyond the scope of a single robot unit. Indeed, the performance of networks composed of coordinated robots that cooperate in achieving their goal, exceeds the performance of one specialized robot that needs to achieve all the given tasks. The improved performance is due to the parallelism introduces in the network. Improved performance induces the improved efficiency, that reflects in the quality of fulfilled goal and the speed of its execution. One of the challenges in networked robotics is the robot intelligence, i.e., the ability recognizing the environment and specific situation, solve the unexpected problems and dynamically reconfigure the network in the case of sudden errors and robot failures.

In this chapter, a robot middleware architecture dedicated to the multi-robot deployment applications is presented. Due to the unpredictability of the deployment environment, followed by the uncertainties regarding the robot robustness, the robot middleware should allow the implementation of a deployment algorithm that is:

- *localized* – each robot relies only on the locally available information, which corresponds to practical implementations where global information is now always available,

- *distributed* – each robot performs its calculations concerning neighborhood discovery and path planning, which facilitates the overall amount of computations needed and resolves the problem of scalability,

- *efficient* – minimal number of robots is used to achieve the maximum goals as possible by using the minimal amount of resources (energy, time, etc), thus increasing the efficiency by reducing cost,
- *self-reconfigurable* – the network infrastructure is automatically recoverable in the case of robot failures, which improves the overall robustness of the network.

The middleware proposed and described in this chapter allows the user to easily implement different types of deployment algorithms. It includes the central base-station application that allows a user to gather all the information sensed by the fleet of robots. Furthermore, the base-station application allows a user to send commands to a group or an individual robot, thus introducing the optional manual control in the robotic network. The middleware presented in this work is dedicated to be used with Wifibot mobile robots [94].

3.1 General middleware architecture

The middleware embedded in networked robots achieves several tasks. Firstly, it interacts with the robot firmware in order to drive the wheel motors and collects the information regarding the sensor output and the battery state. Secondly, it manages the communication with other robots and the base-station in the network. Thirdly, it processes both sensed information about the environment and the messages received from the neighbors in the network. Finally, it reacts and adapts in a fast and reliable way to the events in the environment.

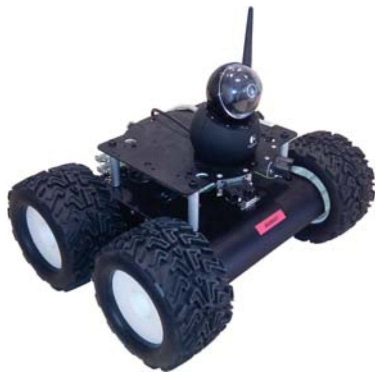
Although many robotic networks assume the existence and reliability of the global network infrastructure that can be used for the communication and information collection among the robots, this scenario cannot hold in the Wireless Sensor Network (WSN) applications. Therefore, the robotic network relies on ad-hoc network infrastructure that allows any-to-any communication paradigm among robots.

3.1.1 Initial Wifibot configuration

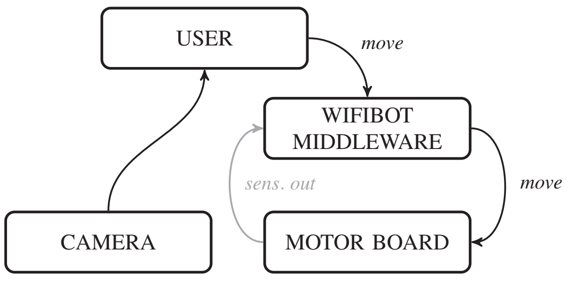
Middleware presented in this work is dedicated to the implementation on Wifibot mobile robots [94]. Wifibots are differentially driven, battery powered, mobile development platforms with integrated on-board computer (Figure 3.1(a)). The base system comprises four wheel drive motor board controllable via RS232 link, 2 infrared range sensors, camera, mini-pci WIFI card, Intel Atom D510 Duo Core processor, operating system installed on 4G compact flash memory and a WIFI access point used for the tele-operation. Further details about the robot construction can be found on manufacturer's website[1].

Figure 3.1(b) depicts the intended way of use where a user manually controls the robot movement by sending the movement commands via TCP link (further details can be found in Section 3.2.3). The manufacturer provides a simple user control application that simulates the use of joystick for piloting the robot. The robot behavior can be observed visually (during the testing phases) or by the visual feedback streamed towards the user. The movement control is implemented in the terms of simple communication protocol that establishes the connection and then periodically transmits the motor speed data towards the robot middleware. The provided middleware accepts, processes the commands and transfers them towards the motor drivers.

Since robot operator has the actual real-time video information of the robot's surroundings, the manufacturer did not provide the robot with a complex set of other types of sensors, and therefore, Wifibot mobile robots are additionally equipped with only two IR proximity sensors on the front side of the chassis. Their robust construction allows them to be used in wide range of applications including otherwise unreachable or harsh environments. The on-board computer serves us as the platform for the deployment algorithm implementation, while the motor driver provides us with the real-time information on power drawn from the battery.



(a)



(b)

Figure 3.1: Wifibot mobile robot and the schema of the user control.

This simple configuration allows the user to manually guide the robot in the specific cases among surveillance applications. Transferred video is used as a visual feedback during the robot control. The physical design of the robot is rather bulky and durable, with sturdy chassis and large wheels

[1]http://www.wifibot.com/

capable of coping with different kinds of surfaces and terrains. These robots are intended to be used in the open spaces (fields, forest, etc.) rather then in closed environments (interior of the building, parking lots, hallways, etc.).

However, the need for human interaction and control makes the use of original Wifibot configuration impractical in many cases. First, it is the human factor included into the robot control process. While being considered as the intelligence included into the system or the ultimate decision taker, the human control can be unreliable under critical circumstances. Second, the positive comment on the chassis sturdiness does not hold for the mounted computer on top of the robot (all together with the set of upgradeable components). Third, the connection used for the control of the robot is a TCP connection which needs to be reestablished in the case of the connection loss, thus loosing both the time and data. Finally, in the case of multiple robot deployment, each robot requires its own operator. It is interesting to note that all aforementioned cases of problems usually appear in practical applications, and are often neglected during the development phase of both the robotic software and hardware.

The possibility to improve the sensory capabilities by introducing new electronic components in the mounted computer board makes the available Wifibot robots a good choice for the development and implementation of multi-robot deployment solutions. This is the reason why Wifibots are chosen as the experimental platforms for the work presented in this thesis.

3.1.2 Towards the use of networked robots

The basic idea in the process of migration towards the concept of networked robots is the idea to create a generic robot middleware that could provide the user with the possibility of use the group of robots as a whole. The role of the user in this case would be to provide the network of robots with their common task or goal, and the group of robots should be able to self-organize in a certain way in order to achieve the desired task. Basically, the user of the robotic network should not be interested in the steps needed to accomplish the goal, notably regarding the specific tasks performed by each individual robot. This is achieved with the use of distributed approach to problem solving, robot deployment in the context of this thesis.

The concept of the generalized middleware architecture proposed in this thesis is shown in Figure 3.2. This architecture represents a hypothetical system design for a networked robot and comprises 4 main blocks: communication, localization, motor control and deployment algorithm block.

The communication block updates the deployment goal and links towards other robots in the network, which makes it important in the context of net-

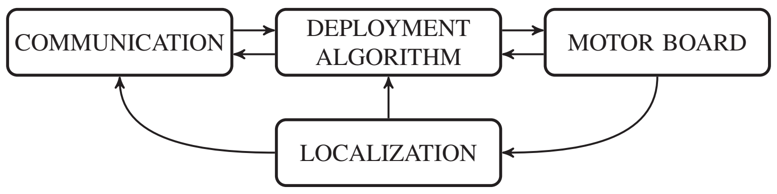


Figure 3.2: An instance of a robot middleware architecture.

worked robotics. Localization part takes care of the real-time position information that are used for further computations by other blocks. The motor control block represents the motor control board provided by the manufacturer, that includes the motor driver, a set of inner sensors (battery state and motor encoders) and external proximity sensors. The block that represents the deployment algorithm will get a lot of attention throughout the thesis, since it is the place where the actual localized decision making as the result of implemented deployment algorithm. Further details about each block can be found in the next section.

3.2 Implemented middleware for Wifibots

In this section, the implemented middleware for networked Wifibots is presented in detail. Each building block of the proposed architecture is separately described and the interaction between the blocks are explained and justified. Figure 3.3 presents the detailed schema of the proposed and implemented middleware architecture. The block significations and the links that interconnect them are explained in the following sections.

3.2.1 Motor control block

The motor control block is the crucial block in the complete robot middleware architecture, since it is in charge of controlling the built-in sensors and motor drivers. Main logical parts that constitute the motor control block are the motors, motor drivers and motor encoder, followed by the set of proximity sensors and battery state indicators (Figure 3.4).

When the movement decision has been brought in the deployment algorithm implementation block, or by the user command, it is processed by the motor driver and the motors are activated. As described in the Wifibot datasheet, each motor is equipped with an encoder that gives the information regarding the quantity of movement. This information is processed in the localization block (Section 3.2.2).

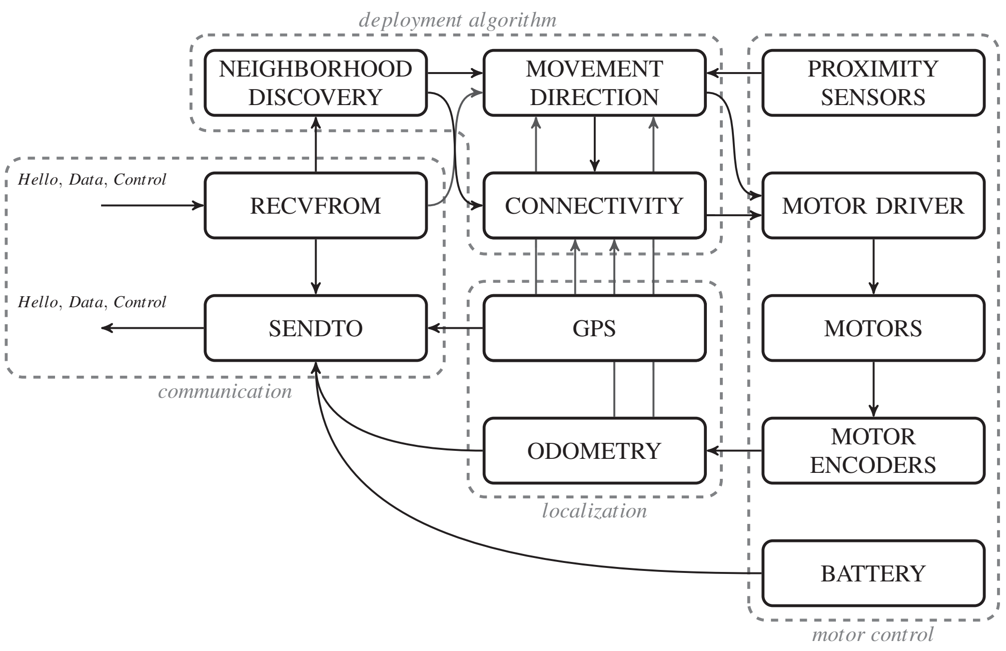


Figure 3.3: Detailed robot middleware architecture.

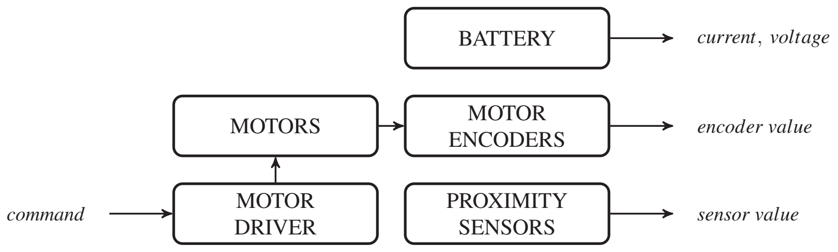


Figure 3.4: Motor control block.

In the case of Wifibots, two infrared range sensors that are mounted on the front part of the chassis represent the proximity sensors block. The output from these two sensors is used to prevent the collision with obstacles during the deployment (including other robots as well). The information obtained from proximity sensors overrides the movement commands given to the motor driver block, thus preventing any kind of collision, even the intended one.

The block representing the battery state information is present in the implementation of the middleware, however it is not used as a feedback that influences the deployment control. The battery block reports the electrical current and voltage to the user, and therefore can be used to evaluate the energy consumption of the robot during the deployment. The information regarding the battery state can be used in order to predict the time of death of a robot, and it can be shared among the networked robots in order to prevent the sudden robot deaths and the network disconnections. In current implementation where the neighborhood discovery is being done through the exchange of *Hello* messages (more details in Section 3.3.2), the death of a robot is declared if it does not send its *Hello* message during a predefined time threshold.

The details about the motor board mounted in the robot chassis are depicted in Figure 3.5, and further details about the motor driver implementation can be found on Wifibot website [94].

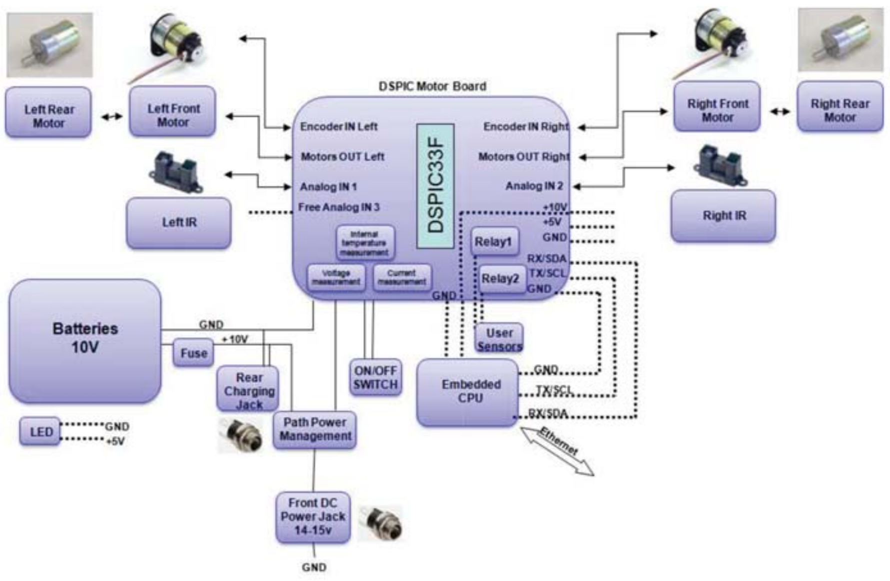


Figure 3.5: Wifibot motor control board [94].

As well as any block in the middleware architecture, the motor control block can be disabled. This feature is used in Section 4.1.1, where all the robot movements need to be disabled for testing purposes.

3.2.2 Localization block

The problem of robot localization represents one of the most important problems in the modern robotics. Indeed, all the applications that rely on the mobile entities require the, as precise as possible, information about the entity position. In the mobile robot deployment approaches that are usually geometrically based, the localization technique plays a crucial role. More information about the localization techniques in mobile robotics can be found in [9, 38].

In this middleware architecture intended to be implemented on Wifibots,

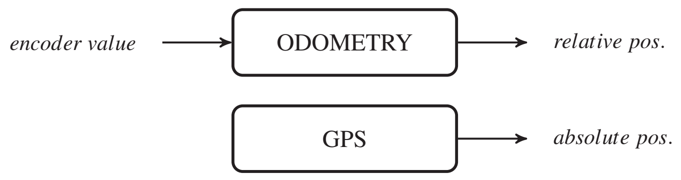


Figure 3.6: Localization block.

two localization techniques are used (Figure 3.6). The first one is the absolute localization technique that relies on the global positioning system (GPS) receiver connected to the on-board computer. The GPS localization technique relies on the signal received from a set of satellites orbiting around the Earth, and is widely used for different applications, notably for navigation in automotive industry. Without providing the details about the GPS localization technique, a general comment about the localization accuracy is that it is sufficient for the general use in automotive industry and that it can be sufficient for a variety of robot deployment applications. Furthermore, increased precision is achievable with the use of more advanced Differential GPS devices[2].

However, it is worth noting that all the GPS localization techniques need a signal from the satellites, which rules out this approach in confined and indoor environments. In general, this technique cannot be used in any environment that can somehow block or alter the signal received form the satellites (interior of the buildings, tunnels, underwater applications, streets surrounded by high and dense buildings, etc). In the context of implemented middleware for networked robots, the GPS localization technique is used only for outdoor robot deployments where the connected GPS device provide the robot with the ready-to-use localization information.

The second localization technique is the dead-reckoning localization based on the output from motor encoders. This kind of simple localization method is widely used since the position derived from the motor movements can be easily calculated in real-time. On the other hand, the biggest flaw of this kind of approach is the localization error that accumulates over time and the estimated location needs to be corrected in order to be usable. In the case of Wifibots, after the odometry parameter calibration, this method can be used for the indoor localization in the case where movement distances stay short. Another problem that arises with the use of dead-reckoning methods is the need for a relative coordinate system that needs to be shared among the set of networked robots. In order words, all the robots need the same relative coordinate system in order to be able to localize themselves properly

[2]http://en.wikipedia.org/wiki/Differential_GPS/

and achieve successful deployment. The details about the coordinate system used among the robots are provided in Section 3.3.

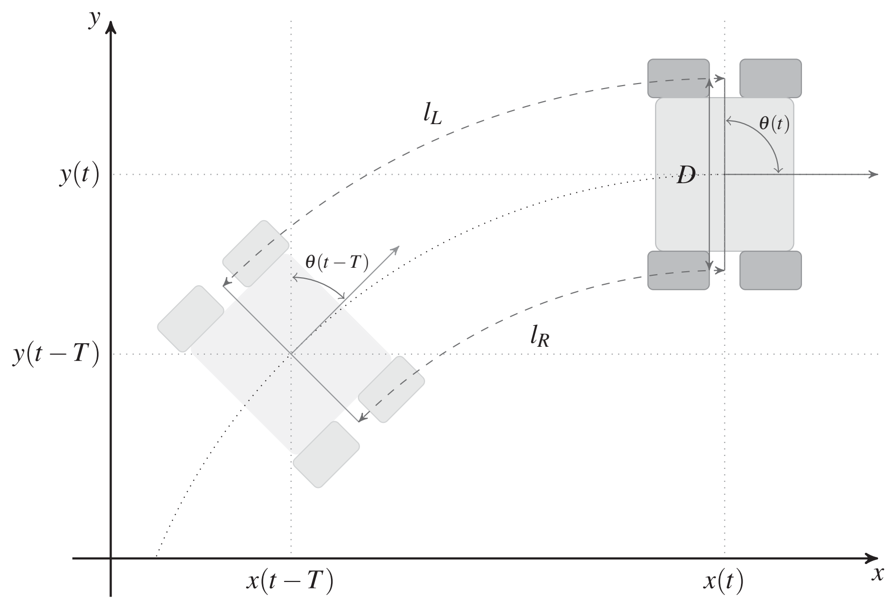


Figure 3.7: Incremental localization technique for a differentially driven robot.

Figure 3.7 shows a scheme of a simplified differentially driven mobile robot with 4 wheels, where l_R and l_L represent the travel distance from the last odometry calculation for right and left wheels respectively, Δl is the overall traveled distance, θ represents the robot relative orientation with relation to the adopted coordinate system, while $x(t)$ and $y(t)$ are the geographical coordinates of the robot in the time instant t and D represents the distance between the robots wheels. Robot position is calculated periodically and for the sake of notation, the calculation period is referred to as T. Robot coordinates and orientation calculated at time instant $t-T$ are referred to as $x(t-T)$, $y(t-T)$ and $\theta(t-T)$. The robot should keep track of the distance traveled by both wheels from the last moment when the location was calculated (l_R and l_L), and with this information, the current position can be obtained by applying equations 3.1.

$$\begin{aligned}
\Delta l &= (l_R + l_L)/2 \\
\theta(t) &= (l_R - l_L)/D + \theta(t-T) \\
x(t) &= \Delta l \cos\theta(t) + x(t-T) \\
y(t) &= \Delta l \sin\theta(t) + y(t-T)
\end{aligned} \tag{3.1}$$

The set of robots with the task of covering the Point of Interest in the indoor environment uses this kind of localization technique (Section 4.2). More generally, the proposed middleware architecture assumes any kind of localization technique that can be used in order to obtain the precise robot localization information (e.g., dead reckoning, global positioning, landmark based positioning, etc.).

3.2.3 Communication block

The communication block is in charge of information exchange with neighboring entities in the network – robots and base station. It is essentially composed of two parts: receiving and sending blocks (Figure 3.8).

All the messages destined to a robot are processed by the receiving block, *Hello* messages are transfered towards the deployment algorithm block (the part dedicated to neighborhood discovery, described in Section 3.2.4). *Control* messages destined to the robot that received the message processes the message and act accordingly to its contents by setting the parameters sent by the user. Robots in the network are used as relays, therefore *Data* and *Control* messages not destined to the current robot are transfered forward towards the sending block. More details regarding the messages used in the communication can be found in 3.3.2.

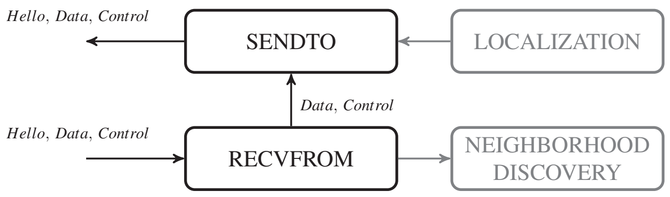


Figure 3.8: Communication block.

The role of the sending block is twofold. First, it periodically checks the port where the localization block outputs robot's current position, then it creates the *Hello* message with the updated location information and broadcasts it. Second role is the *Data* and *Control* retransmission towards their destination. In case there is a need for message transmission towards a destination other that actual robot, the sending block consults the neighborhood table constructed by the neighborhood discovery block and sends the message towards its destination if it is found in the list of neighboring robots. This allows the multi-hop *Control* message transfer from the base station towards all the robots in the network, as well as information acquisition by relaying the *Data* messages from the robots towards the base station.

It is worth noting here that all the messages that are transmitted between the blocks are also sent to a separate port on each robot, thus leaving space for other future upgrades and other block integration in the system. This way of communication between robots is also used for testing purposes, where a user can connect and obtain all the information needed from any of the robots in the network by knowing its address (Section 3.3).

3.2.4 Deployment algorithm block

In this part of the middleware, the deployment algorithm is being implemented. The blocks presented in Figure 3.9 cover all major types of deployments employed for networked robot applications. Taking a generalized deployment scheme into account (Figure 1.1, page 9), the first part of the deployment algorithm is the neighborhood discovery. Based on the information gathered upon the reception of *Hello* messages from neighboring robots, the neighborhood table is updated in the neighborhood discovery block. Different graph reduction methods can be employed in this step in order to cope with the network scalability issues. More information about the graph reduction method used in this thesis can be found in Section 4.2.1, page 53. Graph reduction methods are important because of the scalability issues in sensor networks, instead of taking account of all the neighboring robots in the vicinity, the communication with a reduced set of robots is faster, less energy demanding and provides the same amount of useful information.

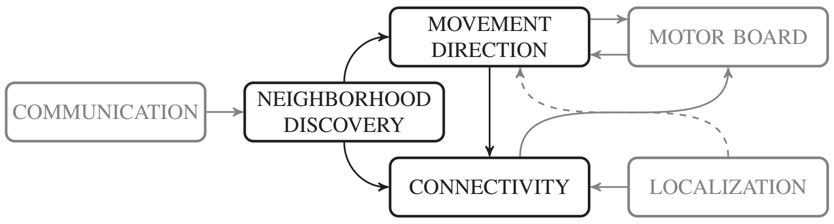


Figure 3.9: Deployment algorithm block.

Besides following the predefined task during the deployment in the sense of an area or a point of interest that has to be covered, in some approaches for robot deployment (e.g., virtual force based technique [4]), an individual robot calculates its movement direction based on the set of neighboring robots. In a majority of geometry-based deployment techniques, a robot needs the exact position of the neighboring robots from the neighbor table, as well as its own precise position. All these calculations are covered by the movement direction block. After the direction calculations, a robot has the defined direction and is ready to move.

Although the movement and direction have been calculated and the robot is ready to move, there are two cases when it will be stopped. The first case is based on the output from the proximity sensors. If the obstacle is detected, all the movements (even the deliberate movement commands sent by the user) will be stopped. The second case is related to the network connectivity preservation and is processed in the connectivity block. After the movement direction has been calculated, the maximal allowed movement distance is calculated as well in order to prevent the network disconnections. This kind of computation is based on the sensor network model and it is analyzed in details in Section 4.2.2, page 55. After the maximal movement distance that keeps the network connected has been calculated, this information is sent towards the motor control block and transferred towards the motor driver.

The separation between the movement direction and connectivity blocks is important since some deployment applications may require k-connectivity or may not require connectivity constraints at all. Furthermore, these requirements can be different all along the deployment procedure. The separation of the blocks allows this kind of flexibility in the implementation.

Depending on the set of commands and the deployment goal set by the user, the deployment algorithm block can serve different types of algorithms, from simple point-to-point linear movement to more complex deployment algorithm presented in Chapter 4.

3.3 Deployment control and data acquisition

An integral part of the sensor deployment in most applications is the establishment of the data acquisition infrastructure. The dynamic nature of the robot deployment changes this paradigm in a sense that the network infrastructure must be auto-adaptable to environment conditions. Examples for this are the disaster areas where it is not possible to elect a set of sensors/robots that will play the role of the communication backbone due to the possibility of sensor failures. The complete network should rather be equipped with the mechanisms of overcoming these types of unexpected environment behavior.

Setting up the network infrastructure may seem to be not a complex task as studied by the robotic communities, however, the problems of cost and time to set it up can arise. The cost of the auto-adaptable network infrastructure becomes an obstacle in remote and large construction sites where the robotic network is used for a structure and machinery health monitoring. In most applications that require fast, large scale and reliable response to environmental changes, the network infrastructure reaction time represents one of the major issues. This section provides the details about the communication in the network.

3.3.1 Sockets

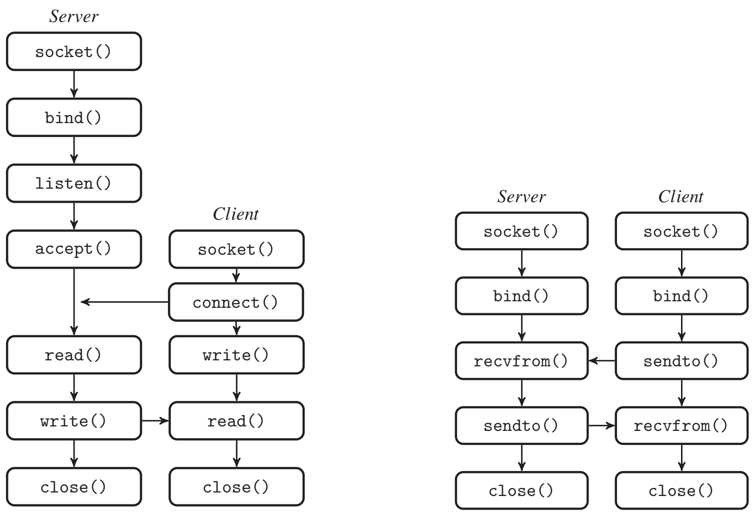
A connection between two software instances is called socket[3]. The communication by using sockets is usually being done between two computers in the network, but sockets can also be used on a single computer where processes communicate between each other. During the communication, both sides are capable of sending and receiving data, which is referred to as bidirectional socket communication. There are three categories of sockets available: connection-oriented (or stream), connectionless (or datagram) and raw sockets.

Stream sockets that implement connection-oriented semantics use Transmission Control Protocol (TCP) or Stream Control Transmission Protocol (SCTP). A stream communication requires the two communicating parties to establish a socket connection beforehand, in order to be able to guarantee the reception of the data transmitted through the connection in the same order in which it was sent. A server can create multiple TCP sockets with the same port number and IP address where each is treated with its own server-child process, however they are treated as different sockets by the operating system. The flow diagram for TCP sockets presented in Figure 3.10a, shows the steps in the stream communication between a server and a client. On the server side, these are creating a TCP socket, binding the socket to the listening port, preparing the socket to listen for incoming connections, accepting incoming connections (`accept()` can be called again at any time until it is closed), communication with the remote host and finally closing each socket that was opened. Steps during the communication on the client side are creating the socket, connecting to the server and communication with it, and finally terminating the connection by closing the socket.

Datagram sockets are also known as connectionless sockets, which use User Datagram Protocol (UDP). In the connectionless semantics, connections are implicit rather than explicit as with streams. Both parties simply send datagrams as needed and wait for the other party to respond. Since there is no message delivery guarantee, UDP packets may be lost in transmission, received out of order or received multiple times, but these problems should be treated in the client/server application. A UDP socket cannot be in an established state, since UDP is connectionless. Therefore, a UDP server does not create new child processes for every concurrently served client, but the same process handles incoming data packets from all remote clients sequentially through the same socket. The data flow for the UDP sockets is shown in Figure 3.10b.

The raw sockets are used for custom low-level protocol development by bypassing the standard protocols of transport layer. In this case the packet

[3]http://en.wikipedia.org/wiki/Network_socket



(a) TCP (b) UDP

Figure 3.10: Flow diagram for TCP and UDP socket communications.

headers are made accessible to the application.

Because of a simple protocol design, UDP has considerably less overhead than TCP. Maintaining a Stream socket in the context of mobile networks is usually hard and inefficient. Implementing datagram sockets can give some applications a performance boost and additional flexibility compared to using stream sockets, justifying their use in applications for networked robot deployment.

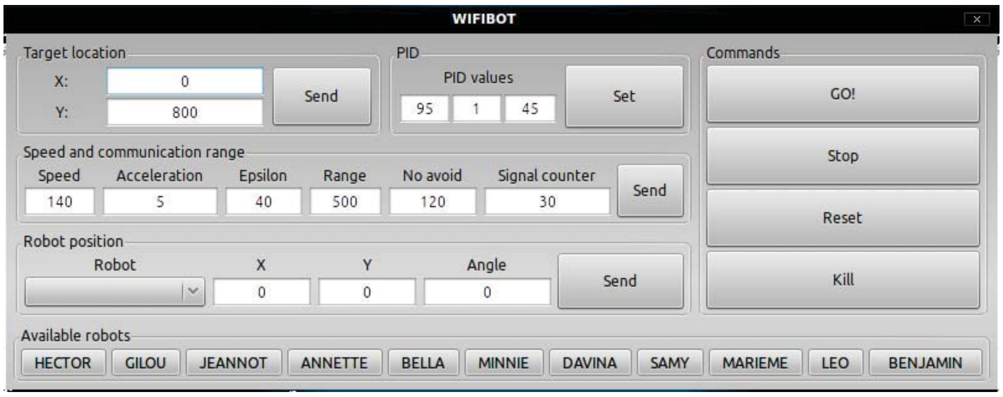
3.3.2 Message format

There are three types of messages that are employed in this network architecture: *Hello*, *Data* and *Control* messages (Figures 3.11 and 3.13). As it is mentioned in Section 3.2.1, page 25, robots discover their network neighborhood by exchanging *Hello* messages. These are short messages that are broadcast periodically with an arbitrarily chosen broadcast frequency depending on the dynamics of the network. Each message contains its identifier (0 for a *Hello* message), sender's unique identifier (robot's ID), set of geographical coordinates, the optional battery state and a checksum field. Depending on the indoor or outdoor application, coordinates contained in the message are relative and/or absolute coordinates, depending on the chosen localization technique. The battery state is not currently used in the implementation, although its use is foreseen. The battery state is intended to be used as an estimate of the robot's lifetime that can be used by other robots in order to prevent unexpected robot deaths and network disconnections. It is also used as the input of deployment algorithm block, where the low battery level prevents the robot movement. The checksum field in each message is a simple method of preventing unexpected erroneous messages due to the errors in transmission.

Hello :	`0`	`ID`	`pos. x`	`pos. y`	`battery level`	`checksum`
Data :	`1`	`ID`	`length`	`seq. num.`	`... data ...`	`checksum`

Figure 3.11: *Hello* and *Data* message format.

Data acquisition in the network is achieved by the use of *Data* messages. These messages originate in a robot and are always destined towards the base station. They contain the ID of the originating robot, a field that indicates the length of the data contained in the message, followed by the actual data that needs to be transmitted. The user's interaction with the network as well as setting the deployment parameters are achieved through the use of base station application (Figure 3.12) that sends *Control* messages in the network. These messages as well as *Data* messages are transmitted in the multi-hop manner, where a set of intermediate relay robots is used if two communicating parties cannot exchange messages directly.


WIFIBOT

Target location
X: 0
Y: 800
Send

PID
PID values: 95 1 45
Set

Commands
GO!
Stop
Reset
Kill

Speed and communication range

Speed	Acceleration	Epsilon	Range	No avoid	Signal counter
140	5	40	500	120	30

Send

Robot position

Robot	X	Y	Angle
	0	0	0

Send

Available robots
HECTOR GILOU JEANNOT ANNETTE BELLA MINNIE DAVINA SAMY MARIEME LEO BENJAMIN

Figure 3.12: Multi-robot control application.

There are different types of *Control* messages (Figure 3.13):

- Control 2 (setting the gradient) – the message is sent in the beginning of the data acquisition phase. The message contains the *gradient* field with a gradient number that is incremented at each retransmission (explained in Section 3.3.3). A robot that receives this kind of message, sets its gradient number based on the value received in this kind of control message.

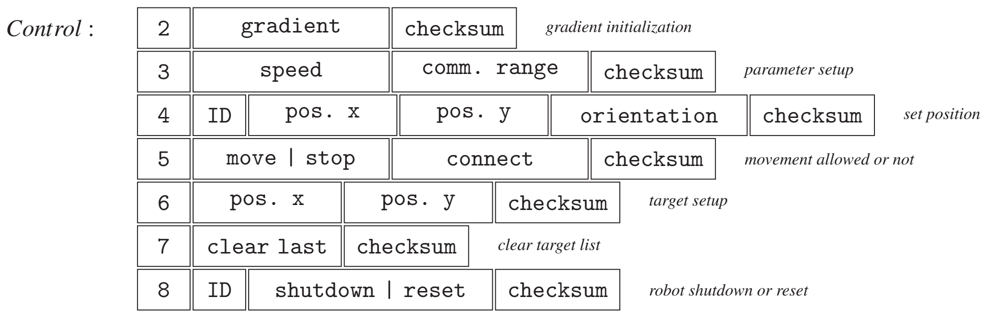

Control :
2 | gradient | checksum | *gradient initialization*
3 | speed | comm. range | checksum | *parameter setup*
4 | ID | pos. x | pos. y | orientation | checksum | *set position*
5 | move | stop | connect | checksum | *movement allowed or not*
6 | pos. x | pos. y | checksum | *target setup*
7 | clear last | checksum | *clear target list*
8 | ID | shutdown | reset | checksum | *robot shutdown or reset*

Figure 3.13: *Control* message format.

- Control 3 (robot parameter setup) – this message contains the maximal speed of the robot and the maximal allowed communication range (physical distance) between the communication robots. The communication range set in this way is used in the connectivity preservation part of the deployment algorithm.

- Control 4 (robot position setup) – in the case of indoor deployment when the odometry is used for the localization, this control message is used to set a robot's position and orientation in the relative coordinate system. This command is invoked for each robot in the network in order to ensure the proper setup of the network.

- Control 5 (movement control) – this is a message that is broadcast in the network in order to start or stop the deployment process. Initially all the movements are disabled and robots wait for this message to be able to start the movement. If the *movement* field is set to 1, the robots are allowed to move, and during the movement the network connectivity is maintained if the *connect* field is set to 1 (this is its default value). If the *movement* field is set to 0, all the robots stop their movement.

- Control 6 (target setup) – in order to set the deployment objective, a user should use this command and provide all the robots with the coordinates of the target (Point of Interest in Section 4.2). This message can be broadcast multiple times with different deployment targets, thus creating a list of targets. Robots therefore have a list of objectives and it is up to deployment algorithm to define the way these targets are going to be covered.

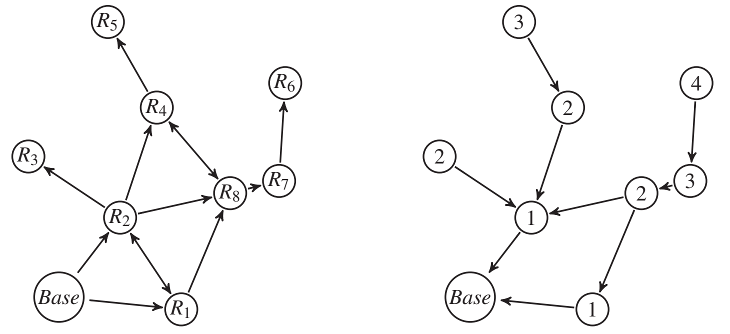
- Control 7 (clears target list) – following a queue of previous messages, a user can erase the target list by broadcasting this message in the network. Upon reception of the message, a robot clears the list of targets.

If the *clear last* field is set to 1 (by default it is set to 0) only the last added target is erased from the list.

- Control 8 (shutting down) – finally, in the end of deployment application or the experiment, all the robots need to be shut down, and this is the message that achieves this once broadcast in the network (with the *shutdown / reset* field set to 1 and *ID* field set to 0). In the case where only some of the nodes need to be shut down, the base station application provides the user with the possibility to set the desired robot's ID and to shut it down separately from other robots in the network. The base station application sends multiple messages if more than one robot is chosen as the destination. If the *shutdown / reset* field is set to 0, the robot middleware application is restarted upon the reception and the transmission of the message.

3.3.3 Multi-hop gradient routing

To conclude, the approach presented in this thesis relies on the use of connectionless communication protocol, that allows us to boost the robustness of the network. The communication among the robots in the network is done by using two types of messages: *Hello* (used for neighborhood discovery) and *Data* (used for data transmission). *Hello* messages are only considered by one-hop neighbors, while the *Data* messages represent the essence of the WSN (gathering information) and therefore must be transmitted in a multi-hop manner towards the base station. In this work, this is done by the use of gradient routing.



(a) Gradient construction with *Control* message

(b) *Data* message routing

Figure 3.14: Gradient assignment to robots in the network (a) and information routing towards the base station (b).

Prior to actual information routing in the network it is necessary to construct routing tables that contain the information about the routes towards the base station. The base station broadcasts the gradient control message in the network in the multi-hop manner (Figure 3.14a). During this process, on each message reception, a robot increments the gradient value in the message or discards the message if it has already been received. After all the robots have received the message from the base station, the process of establishing the robot gradient is completed. When a robot has some information to transmit towards the base station, it sends it to its neighboring robot with the lowest gradient (Figure 3.14b). For example, the robot $R4$ will send its information to robot $R2$. Due to the network dynamics, robots can lower or increase their gradient numbers in case of changes in the neighborhood.

3.4 Discussion

This section discusses some of the issues that a general networked robot architecture takes into account.

As already pointed out, one of the biggest issues in the domain of robotics in general is the problem of precise **localization**. Indeed, in order to guarantee the precision and accuracy of the network deployment, especially in the case of deployment algorithms based on geometry, it is necessary to provide the robots themselves and their neighboring set of robots with the accurate localization information. Two types of localization techniques presented in this thesis have their own flaws and conditions that need to be fulfilled in order to guarantee the network operation without errors. There are numerous solutions available that can solve the problem of localization both for indoor and outdoor applications. One of the possible solutions to localization problem can be the use of landmark based localization, notably for testing purposes where the environment is known prior to actual deployment. However, in this work, the focus is put on the issues of networked robotics in the terms of robot cooperation in order to achieve a given task. Therefore, the localization techniques are used as they are. It is worth noting that middleware architecture assumes the use of any kind of localization technique.

Practical implementations of a robotic network demand the use of different types of robots in the network. Depending on the specific application, it is sometimes necessary to combine different types of robots that can serve different purposes at the same time, all in the context of the same network. The issue of **heterogeneity** has not been thoroughly discussed from the point of view of the middleware architecture, however this problem stays in the domain of the robot interaction with the environment and not in the domain of cooperation in the network. In other words, even if the robotic network is heterogeneous in the sense of different types of robots employed in the deployment application, their general behavior in terms of inter-robot cooperability stays unchanged. In essence, this is the goal of proposing the middleware that can unify different types of robotic platforms into one robotic network with a common deployment goal. The simplicity of middleware implementation proposed in this thesis allows its **portability** towards other robotic platforms. The size of the compiled middleware allows its use even on miniaturized mobile robotic devices with small amount of available memory and processing power. In order to cope with the heterogeneity and portability issues, requirements are the compatibility of the communication interfaces and the possibility to create and understand the described message format.

Finally, as in all technical and informatics systems, the problem that gets a lot of attention is the question of system **security**. The problem of system security in the context of robotic networks applies to all aspects of the network - hardware, software and communication. Hardware security issues focus on the problem of robot construction, its sturdiness and ability to cope with physical challenges during the practical application in hostile deployment environments. From the networking point of view, much more perfidious are the issues arising from the security breaches in the communication part. Indeed, insecure connection can lead to unwanted confidential data leak. Furthermore, an attacker can even insert false commands in the network and thus completely change the behavior of the individual robots and take the global control of the network, making it useless or even counterproductive. Being one of the essential problems in the robotic network, the communication security issues will represent a significant part of the future works.

CHAPTER 4

Deployment applications in mobile WSN

Contents

4.1 Improving the QoS with mobile sensors 41

4.2 Points of Interest coverage with mobile sensors . . . 52

4.3 PoI discovery and coverage with mobile sensors . . . 76

In this chapter, three main problems are presented and analyzed: the problem of improving the quality of service with the use of mobile robotic networks (Section 4.1), coverage of the point of interest with mobile robots (Section 4.2) and point of interest discovery and coverage with the use of mobile robots (Section 4.3). These mobile robot deployment algorithms proved their feasibility through numerous simulation campaigns as well as in the practical implementation relying on the proposed middleware.

4.1 Improving the QoS with mobile sensors

One fundamental advance in the context of wireless multimedia sensor networks is the motion capabilities provided to the sensors (Mobile Multimedia Wireless Sensor Networks MMWSN). This controlled mobility is used to optimize the placement of sensors, to provide different topologies, to enhance the reliability and to increase the performance of the communication protocol stack. Dimensioning the multimedia sensor network should be done by anticipating needs and issues. Unfortunately, this is not always feasible. It is difficult to increase wireless link capacity or performance (delay, bandwidth, etc.) since sensors and wireless links have finite resources.

One of the challenges in the context of wireless multimedia sensor networks is to use controlled mobility to improve the performance of the network and to be able to provide QoS to the application layer. The controlled mobility can, therefore, help the network to face evolving conditions such as interferences or contentions.

In this section the algorithm to control the mobility of a sensor is proposed. The goal of the algorithm is to improve the performance of a multimedia communication by using an intrusive methods to gather the necessary transmission quality evaluation metrics. The approach is tested on mobile robots in order to show its feasibility.

4.1.1 QoS improvement algorithm

In order to examine the behavior and create the reference point for further video transmission evaluation, a simple set of test scenarios is created. These scenarios are created in order to see how the position of the node in a multi-hop transmission affects the quality of the transmitted video. The first one is the simple scenario with three nodes: source S and destination D (further in the text referred to as sender and receiver node, respectively) that are fixed during all the tests, and one relay node R that is placed in 5 different positions between source and destination (Figure 4.1). Note that the relay node is static during the video transmission, as the consequence of the motor control block being disabled for testing purposes. The goal of this test is to understand the influence, if any, that the different positions have on transport and application layer parameters. Data gathered from these test will serve as a reference for successive scenarios.

The next set of tests considers the behavior of mobile robots between the video transmission source S and destination D (Figure 4.2) and its impact on video transmission quality. These robots can move on the straight line between source and destination in order to ensure good quality to the transmitted video, by measuring the ping[1] delay and data loss. The goal in these experiments is to examine if it is possible to use a simple movement algorithm to achieve better video transmission quality and if it affects the parameters of the transmitted video. The assumption is that the mobile robots are going to be placed at the distance $d = 2(n_{node} - 1)$ from the source node (where n_{node} represents the robot's position in the network and the distance is given in meters). Further descriptions of the position adjustment algorithm follow.

It is worth noting that all the experiments are done indoors and therefore the maximal distance between the source and the destination node is limited by the free space withing the testing area (30 meters in this case).

4.1.1.1 Position Adjustment Algorithm (PAA)

The Position Adjustment Algorithm (PAA, Algorithm 1), that is run independently on each mobile relay node, is based on packet delay and data loss

[1] http://en.wikipedia.org/wiki/Ping_(networking_utility)

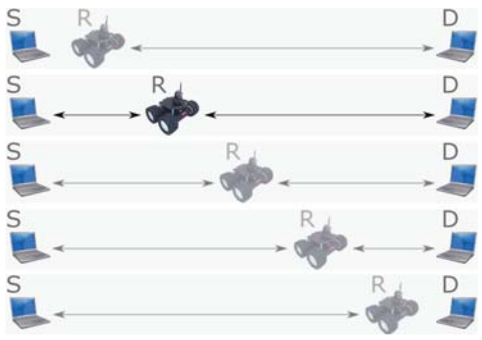


Figure 4.1: Different positions of a static relay node (static robot).

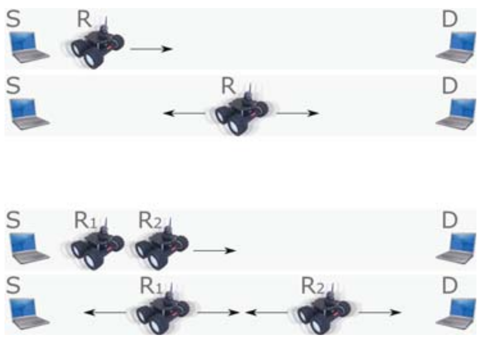


Figure 4.2: Two scenarios with mobile robots.

by using a burst of pings exchanged with one-hop neighbors. The algorithm comprises 3 consecutive phases: ping sending, movement direction calculation and actual movement, following the block division or the middleware architecture. In the first phase, the relay node pings one-hop neighbor in both directions α times, with the time interval of β between two pings. Since the relay node sends 2α pings in this manner, this phase finishes in $2\alpha\beta$. In the second phase of the PAA, the data regarding the packet delays and overall packet loss (gathered in the first phase and processed in the neighborhood discovery block of the middleware) is used for the calculation of the movement direction (in the movement direction block). Actual movement is done in the last phase of the PAA and the movement distance is represented by the value γ in the motor control block. Within the experiments, the values for α, β and γ are arbitrarily set to $\alpha = 40$, $\beta = 200ms$ and $\gamma = 2m$ considering the length of the experiment and the constraints of the indoor testing environment.

In the second phase, in order to determine the direction of the movement, the PAA uses the *average delay* experienced by the ping requests in the forward (d_{next}) and the backward (d_{prev}) directions and the maximum deviation (d_{dev}) from the higher average. It compares them so that if $(d_{prev} - d_{next}) > d_{dev}$ the movement will be towards the previous node, and similarly, if $(d_{next} - d_{prev}) > d_{dev}$ the movement will be towards the next node. The next criterion considered in the PAA is the percentage of *data loss* after the pings have been sent to both sides. The comparison in this case is straightforward, if $l_{prev} > l_{next}$ the movement is towards the previous node and if $l_{next} > l_{prev}$ the movement is towards the next one hop node (where l_{prev} and l_{next} represent percentage of data loss from the previous and next node, respectively). Note that in the PAA, data loss has greater priority than packet delay, so the delay based movement decision will be overridden by the

Algorithm 1 Position Adjustment Algorithm (PAA).

```
 1: Send $\alpha$ pings every $\beta$ to previous one-hop node
 2: Send $\alpha$ pings every $\beta$ to next one-hop node
 3: Calculate the average delay of the packets ($d_{prev}$ and $d_{next}$), its standard devi-
    ation ($d_{dev}$) and percentage of packet loss for both directions ($l_{prev}$ and $l_{next}$)

 4: if (($d_{prev} - d_{next}$) > $d_{dev}$) then movement will be backward
 5: if (($d_{next} - d_{prev}$) > $d_{dev}$) then movement will be forward
 6: if ($l_{prev} > l_{next}$) then movement will be backward
 7: if ($l_{next} > l_{prev}$) then movement will be forward
 8: if ($l_{next} = l_{prev}$) then cancel any movement decision
 9: if (($l_{prev} = 100\%$) and ($l_{next} = 100\%$)) then movement will be backward
10: If the movement decision is brought, move to a new position by moving $\gamma$ in the
    chosen direction
```

loss-one if the calculated direction differ. A special case is when data loss is the same on both sides. In this case the delay based movement decision is not to move, except if the loss on the both sides is 100% when the movement direction is towards the previous node. Recall the assumption that the robots are positioned close to the source node, and in the case of complete loss of data we want the robots to be able to restore network connectivity autonomously.

Since the algorithm is meant to be run constantly during the video transmission, and since the QoS data are most likely to change due to different reasons, it can be expected that robots running Algorithm 1 will oscillate around or eventually stop in the area with the best link quality. In the set of tests, robots run 40 movement decision periods of the PAA (since it is enough number of steps needed for the robot to reach and stay in the high QoS area).

4.1.1.2 Video evaluation

In order to evaluate the video samples that are constantly being sent from the source to the destination, EvalVid tool-set is used [45]. EvalVid allows the user to easily measure and evaluate not only the QoS parameters like loss, jitter and delay, but also standard video quality metrics like peak signal to noise ratio (PSNR) and subjective metrics like mean opinion score (MOS).

Before starting the transmission, the raw video is compressed with 30 frames per second and a GOP (Group Of Pictures) length of 30 frames. The video with a hinted track that describes how to packetize the frames for the RTP (Real-time Transport Protocol) is then created. During the video transmission, trace of every frame is kept both on sender and receiver, which is used later in the evaluation for received video reconstruction.

It is worth pointing out the meaning of two terms that are used in the experimental evaluation part: peak signal to noise ratio and mean opinion score. Peak signal to noise ratio is actually the derivative of the regular signal to noise ratio, which is why it correlates better with the subjective perceived quality of the video. For the quality evaluation, the PSNR is calculated for every frame in the original video, and compared with PSNR graphs from the videos on the receiver side. To express the subjective quality of the video, the mean opinion score (MOS) of the frames is used, both for the reference video and the videos for the receiver side. In experimental results, the percentage of the frames in 25-frame sliding window that have worse MOS than in the reference video are compared. More information concerning the video quality metrics can be found in [35].

4.1.2 Experimental results

The set experiments is divided in two parts: testing the video transmission quality by using different *static* positions and testing the video transmission quality over the autonomous *mobile* robots that run the PAA.

The first set of experiments, where the relay node is manually set on different locations between the sender and receiver nodes, tends to find the location where the transmitted video quality is the highest. After comparing the delay, jitter, overall frame loss, PSNR and MOS of the transmitted video, the experimental results show that the best location for a relay node is *in the middle* between the sender and receiver nodes (Section 4.1.2.1).

The experiments in the second scenario show that the transmitted video quality is improving by utilizing the PAA on the mobile relay nodes. Trajectories passed by mobile robots are analyzed, and the conclusion is that they tend to position themselves in the middle between their one-hop neighbors, as predicted in the first set of experiments (Section 4.1.2.2).

4.1.2.1 Relay node placed on different positions

In the first part of the experimentation, a two-hop video transmission is tested through 3 different scenarios that are presented in Figure 4.1. Due to the constraints of the indoor experimental environment, the sender and receiver nodes are placed 30m away from each other, and the robot (acting as a static relay node) is placed at 5 different positions between them (at 5m, 10m, 15m, 20m and 25m from the source). After the video transmission is finished, different parameters and quality indicators are analyzed, evaluation metrics that can be obtained from available data: end-to-end delay, cumulative jitter, frame loss, peak signal to noise ratio, mean opinion score (MOS) and percentage of frames with MOS worse than the reference value.

Figure 4.3 presents the end-to-end delay and cumulative jitter for all the different positions of the robot. When the robot is in the middle position (third scenario) between the source and the destination, the transmission suffers the highest delay. In fact, the middle position is the furthest position for both source and destination together, and this turns in a low data sending rate that produces the high delay values. Furthermore, the positions closer to the source than to the destination perform better, because the video transmission establishes an asymmetric data flow between source and destination, and positions of the relay node closer to the source allow it to increase the sending rate [7]. On the other hand, asymmetric positions between source and destination suffer higher jitters, in respect of the more stable intermediate position that guarantee fairness of treatment in both directions.

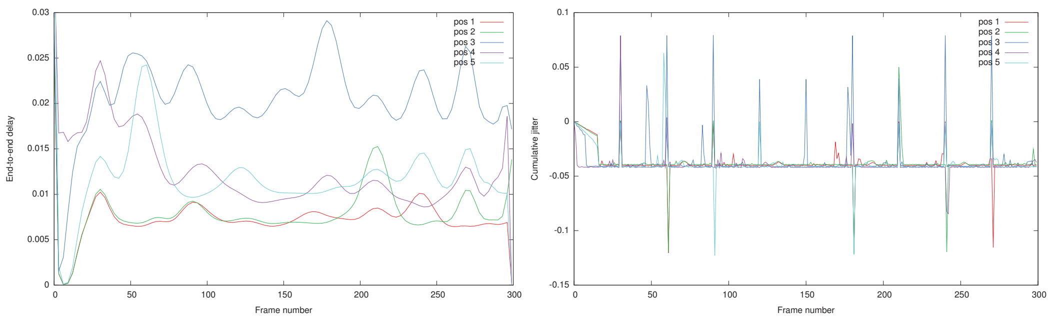


Figure 4.3: Delay and jitter for each position of the robot.

The same behavior is evident from Figure 4.4, where the loss of intra-frames (I), predicted frames (P) and the overall loss is shown. Although the rate is low and the delay values are the highest in the third (middle) position, the overall data loss has its lowest values exactly there. Clearly high data loss values for other robot positions in this experimentation setup is again related to different data rates as a cause of different physical distances from the neighbors. The data rate is higher, which gives shorter end-to-end delays, but as an effect of this, the link is prone to packet losses and therefore unreliable transmission.

Concerning the subjective video quality in Figure 4.5, all the positions show a low quality of PSNR, because of the physical and electromagnetic characteristics of the environment where the tests are executed. Only few frames for some of the tested scenario achieve a quality comparable to the reference. As for the overall data loss, the third (middle) position of the robots shows the best results. It is clear that in the PSNR graphs, PSNR corresponding to the third position is the closest in values to the reference

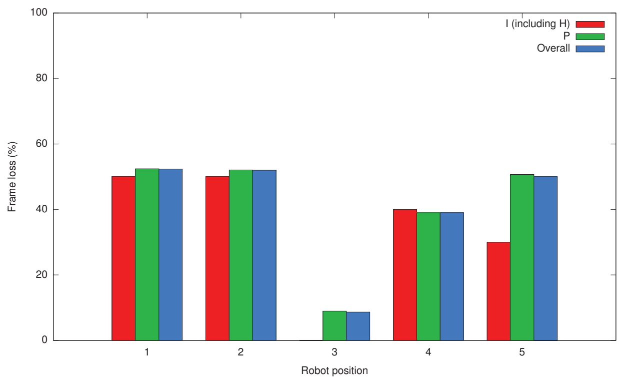


Figure 4.4: Frame loss depending on a robot static position.

ones. Obviously, the lowest percentage of frames with MOS worse than the reference occurs for the intermediate position.

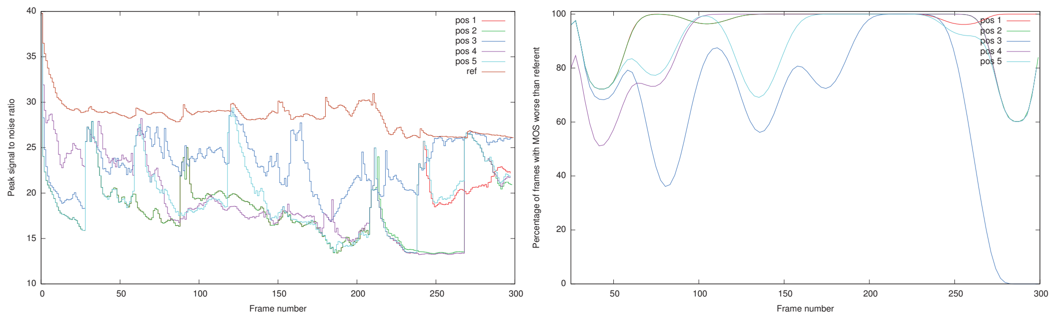


Figure 4.5: Peak signal to noise ratio and percentage of frames with MOS worse than referent for a sliding window of 25 frames.

Based on these results, the conclusion is to focus the attention on the positions around the middle point between one-hop neighbors. It is also expected that the autonomous movement based on packet delay and loss, as described in Algorithm 1, will place robots in equidistant positions between source and destination.

4.1.2.2 Video transmission during the movement of relay node

This section is dedicated to tests with autonomous mobile robots, when video transmission is forwarded through a multi-hop chain of relay nodes. The motivation for this test is to see if it is possible to use network layer parameters to evaluate the quality of communication.

The setup of the experiments is as follows: the video transmission source and destination are fixed on the same distance as in the previous set of experiments and the video sample is continuously transmitted over the link, while the transmission is done via one or more mobile nodes (Figure 4.2). First, only one mobile robot is used and it is placed at γ distance from the source node ($\gamma = 2m$ in this case), and its behavior is observed. Running the PAA to get the average delays of the ping to one hop neighbors (source and destination nodes in this case), the robot positions itself in order to equalize the delays/losses on both sides. Figure 4.6 depicts the evolution of the trajectories of the robot in six successive tests (robot's position in relation to particular movement decision time).

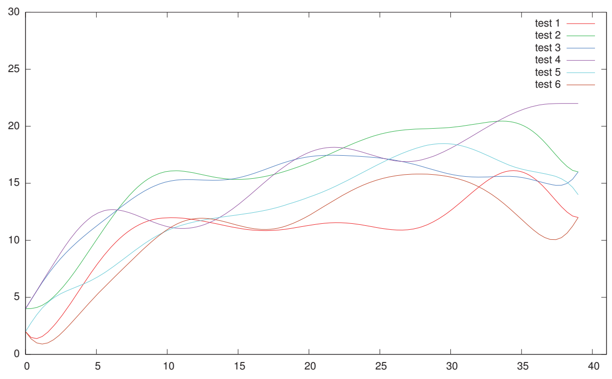
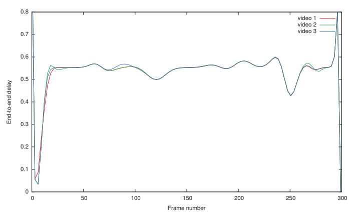


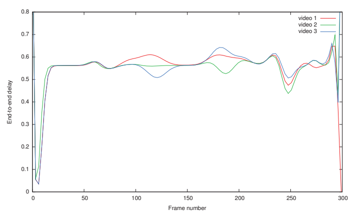
Figure 4.6: The evolution of robot's position (in meters) during 40 iterations of the PAA.

In order to evaluate the video transmission during the robot movement, the following evaluation metrics are used: end-to-end delay, cumulative jitter, overall loss, peak signal to noise ratio and mean opinion score. These metrics are extracted for three videos from three different robot positions – when the movement begins (when the robot is the closest to the sender), when the robot is in the middle between the sender and the receiver, and from in between these two positions. The EvalVid video evaluation tool is used in order to easily compare videos transmitted while robot is moving.

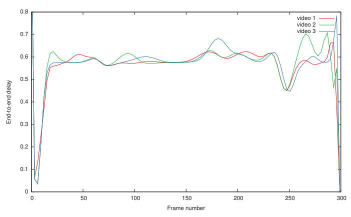
The end-to-end delay for the three positions is presented in Figures 4.7 (a) to (c). The first set of transmitted videos can be used as a reference point, since all the transmitted frames of the video file follow the same pattern depending on the length of the frames. It can be seen in these graphs that the delays of the video frames from different places do not stand out much from the video from the beginning. Furthermore, the delays greatly depend on video itself (similar trace shape). The influence of the actual robot position is clear in Figure 4.7(c), the delays are increased, as we expected after the test with different robot positions (Figure 4.3) – the third position from these examples corresponds to the middle position from the measurements with the static



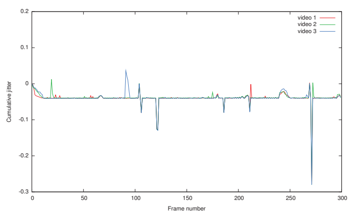
(a) start



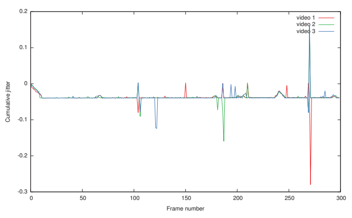
(b) middle



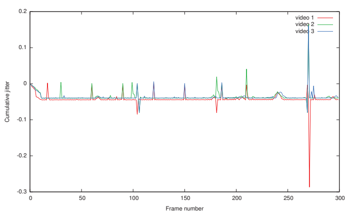
(c) end



(d) start



(e) middle

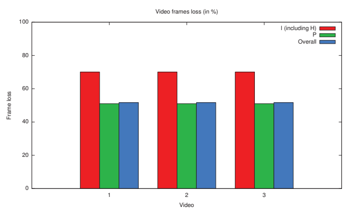


(f) end

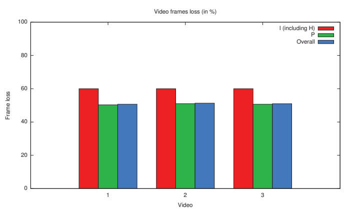
Figure 4.7: End-to-end delay (a, b, c) and cumulative jitter (d, e, f) of a transmitted video during robot movement (in different phases).

routing robot. The comparison of the cumulative jitter evolution is shown in Figure 4.7 (d) to (f).

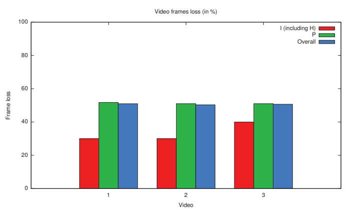
One of the most important properties concerning the video transmission is the percentage of lost data. As mentioned earlier, the loss of the different types of frames within the video file is observed. It is evident that the percentage of lost header frames (H) significantly drops, which produces an higher peak signal to noise ratio, although overall loss remains practically unchanged (Figure 4.8).



(a) start



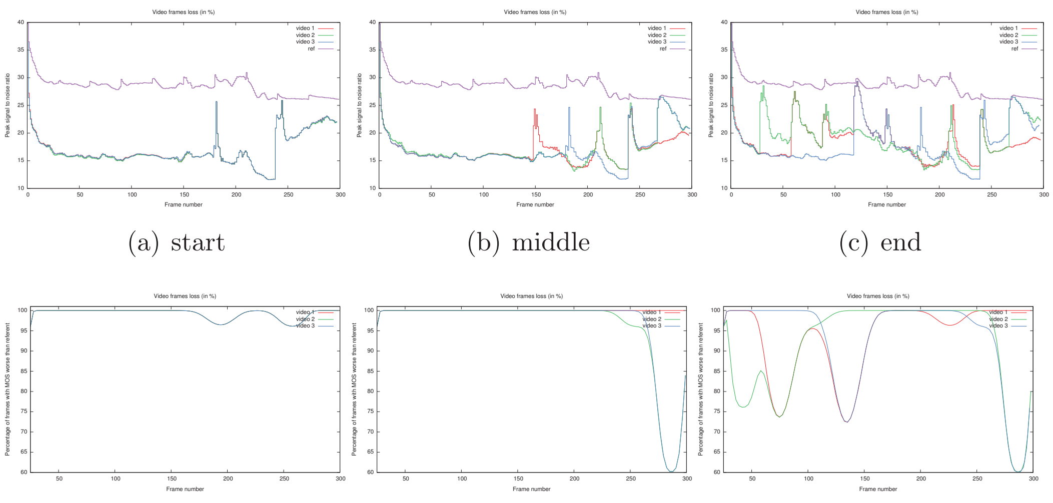
(b) middle



(c) end

Figure 4.8: Frame loss of a transmitted video during robot movement (in different phases).

To evaluate the subjective quality of the transmitted video, the results for the PSNR of the three transmitted videos in comparison with the reference video are shown in Figures 4.9(a) to (c). It can be seen that the PSNR in comparison to reference video is the best in the third sample. A better representation of the quality is the graph of MOS of the 25-frame sliding window that is worse than the reference one (depicted in Figures 4.9(d) to (f)). It is evident that in the graphs that represent the videos in the furthest position from the source, percentage of frames with the MOS worse than reference significantly drops, which is a proof of better subjective video quality.



(a) start (b) middle (c) end

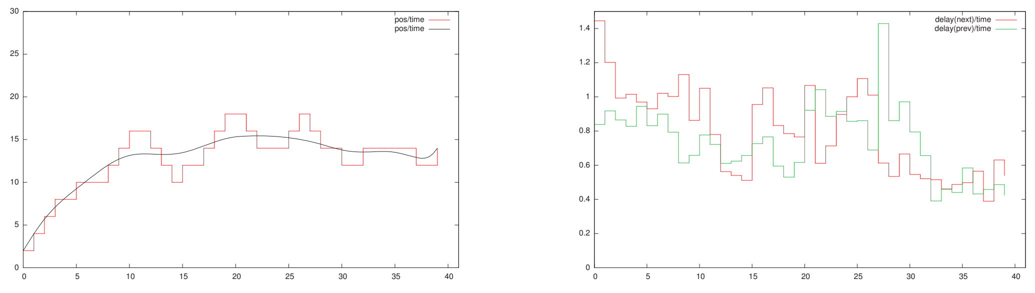
(d) start (e) middle (f) end

Figure 4.9: Peak signal to noise ratio and percentage of frames with mean opinion score worse than in referent video for a sliding window of 25 frames, during robot movement (in different phases).

4.1.2.3 Robot trajectories

After the video transmission being evaluated, the output of Algorithm 1 is presented in a sense of mobile robot positions evolving with time, which is actually the output of the neighborhood discovery and localization blocks of the middleware architecture. Two cases are presented, with one and two robots autonomously positioning themselves in between source and destination nodes. In Figures 4.10 and 4.11, the time units are actually the movement decision steps from Algorithm 1, where each phase lasts for 16s.

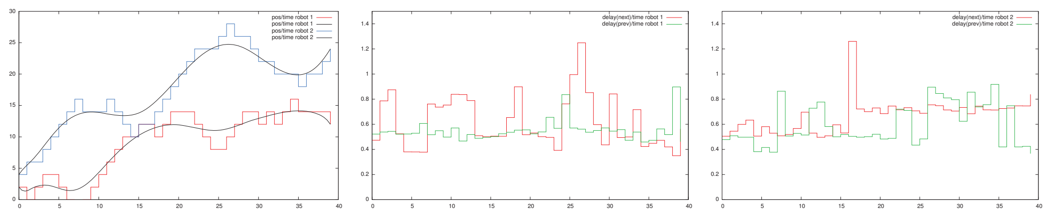
The evolution of the position of the single mobile robots between static source and destination nodes is depicted in Figure 4.10(a). The robot positions itself in the low data loss region after approximately 10 movement decisions, and it slightly oscillates around the middle point until the end of the test. This is an expected behavior since the lowest loss region of the video transmission link is exactly the middle region regarding the distance. Average ping delays to the source and destination nodes used in the movement algorithm are presented in Figure 4.10(b).



(a) robot position in relation with time (b) average ping delays

Figure 4.10: Trajectory and packet delays for one robot running the moving algorithm. Figure (a) depicts the position of mobile relay during 40 iterations of the PAA.

Two mobile robot scenario shows satisfactory results regarding the robot placement during the algorithm run (Figure 4.11). As it could be expected, robots choose approximately equidistant positions to preserve low data loss.



(a) positions of the robots in relation with time (b) average ping delays for first robot (c) average ping delays for second robot

Figure 4.11: Trajectories and packet delays for two robots running the moving algorithm. Figure (a) depicts the position of mobile relays during 40 iterations of the PAA.

4.1.3 Conclusion

The approach to control the mobility of wireless mobile robots to improve the performances of multimedia video transmission was presented in this section. The robot movement algorithm is based on local, but intrusive, measurements and uses the delays and percentage of packet losses of pings sent to previous and next node on the path between the source and destination. The effectiveness of the proposed movement strategy is evaluated on a real multimedia testbed based on a set of Wifibot mobile robots and the middleware described in Chapter 3.

Obtained results show that it is possible to use a simple algorithm for autonomous mobile robots in order to reduce the video transmission loss by changing the node positions in a multi-hop transmission link. Despite the use of ping delay and loss data as the input to the movement algorithm, it still proves to be feasible. However, these experimental scenarios imply the use of certain network metrics obtained by intrusive methods. Concerning the PAA execution time, the use of different network metrics is suggested.

4.2 Points of Interest coverage with mobile sensors

In this section, the attention is focused on the application of wireless sensor networks is the environment monitoring. Environmental and military applications are demanding in terms of efficient monitoring of events of interest. Efficient solutions should be provided in order to solve the problem of deployment of mobile sensors capable of capturing the real time video of possibly mobile events (Points of Interest in this section). In all these applications, sensors have to be deployed and placed on strategic monitoring locations. In many cases, monitoring the whole area might be unnecessary. Instead, monitoring only a set of specific PoIs increases the network performance and reduces the deployment cost. When sensors have motion capabilities, monitoring only a set of PoIs instead of the whole area also allows time dependent coverage.

The assumption is that a fixed base station is placed inside the field of interest. At the beginning of the deployment, the base station already possesses the information about PoI location. Its tasks are to:

- spread out the information about PoI locations among the sensors,
- collect the information reported from the sensors about the events happening at the PoI.

In the case where it is possible to have several simultaneous PoIs in the field and that the PoI can be mobile, the deployment algorithm has to adapt its behavior depending on evolving requirements. In order to dynamically adapt to the changing requirements, the deployment algorithm must guarantee the connectivity all throughout the deployment procedure. This enables the base station to track the position of the existing PoIs and/or to consider a new PoI even during the deployment procedure.

In the following, the set of preliminaries regarding the PoI coverage is provided in Section 4.2.1. PoI coverage algorithm is presented in Section 4.2.2, while the deployment properties for a static, moving and multiple PoIs are analyzed in Sections 4.2.3, 4.2.4 and 4.2.5, respectively. Details regarding the algorithm implementation using the middleware for Wifibots is presented in Section 4.2.6. Based on the obtained data during the algorithm implementation, the Wifibot energy model is provided in Section 4.2.7.

4.2.1 Preliminaries

Following definitions, assumptions and notations are used for the network model. The Unit Disk Graph (UDG) is used to formally describe the sensing and communication model of a sensor in the network [18].

Definition 1 *Let $G(V, E)$ be the graph representing the sensor network. V is the set of vertices each one representing a sensor. $E \subseteq V^2$ is the set of edges; $E = \{(u, v) \in V^2 \mid u \neq v \ \wedge \ d(u, v) \leq R\}$, where $d(u, v)$ is the euclidean distance between sensors u and v and R is the value that represents the maximal communication range between sensors. $G(V, E)$ is the model of the sensor network.*

Definition 2 *$N(u) = \{v \in E \mid d(u, v) \leq R\}$. $N(u)$ is the set of 1-hop neighbors of the sensor u.*

Assumption 1 *Euclidean coordinates of the sensor u and the PoI p are denoted by $u(x, y)$ and $p(x, y)$, respectively. This position information is obtained from the localization block of the robot middleware.*

Assumption 2 *At the beginning of the deployment sensors are randomly spread out around the base station at a maximum distance of $d < R/4$ from the base station. This condition ensures that the network is initially connected and that it remains connected during the deployment (detailed explanation can be found in Section 4.2.2.1, proof of Theorem 4).*

Assumption 3 *PoI locations are known and provided to sensors before the deployment with the use of* Control *messages for target setup (Section 3.3).*

This work uses the Relative Neighborhood Graph (RNG) [86] approach to graph reduction. Given an initial graph G, the RNG graph extracted from G is a graph with a reduced number of edges but the same number of vertices. Let the sensors be the vertices of the initial graph and that an edge between two vertices exists if the two sensors can communicate directly. The assumption is that the communication between two sensors is possible only if the distance between them is shorter than a given communication range. In order to build

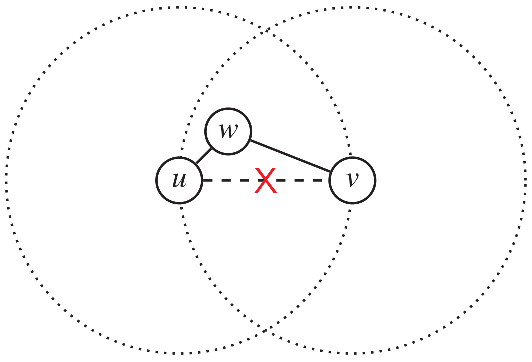


Figure 4.12: Example of RNG edge removal.

a RNG from an initial graph G, the edge that connects two sensors is removed if there exists another sensor that is at a lower distance from both sensors. Figure 4.12 shows an example of edge removal, where edge between sensors u and v is removed since there exists a sensor w that is closer to both u and v. In this way, the number of neighboring sensors taken into consideration is reduced.

The formal definition of the RNG graph is as follows:

Definition 3 *Let $RNG(G)$ be the relative neighborhood graph extracted from $G(V,E)$. $RNG(G) = (V, E^{rng})$, where $E^{rng} = \{(u,v) \in E \mid \nexists w \in (N(u) \cap N(v)) \wedge d(u,w) < d(u,v) \wedge d(v,w) < d(u,v)\}$.*

It is worth noting here that $RNG(G)$ can be different after each sensor movement.

Definition 4 *$N_{RNG}(u)$ is the set of u's RNG neighbors, $N_{RNG}(u) = \{v, w \in N(u) \wedge v \in N(w) \mid d(u,v) < d(v,w) \wedge d(u,w) < d(v,w)\}$. We denote by $|N_{RNG}(u)|$ the number of sensors in $N_{RNG}(u)$.*

Definition 5 *$N^{+}_{RNG}(u)$ (resp. $N^{-}_{RNG}(u)$) is the farthest sensor that is part of $N_{RNG}(u)$, the distance between u and $N^{+}_{RNG}(u)$ (resp. $N^{-}_{RNG}(u)$) is denoted by $d^{+}(u)$ (resp. $d^{-}(u)$).*

The RNG reduction has two main advantageous properties. First, the RNG reduction can be computed locally by each sensor, with the knowledge of its 2-hop neighborhood. Second, given that the initial graph is connected, the RNG reduction is also connected. These two properties are important for scalability and connectivity preservation. Indeed, to preserve the connectivity of the whole network, each sensor has to preserve the connectivity with its RNG neighbors. In the PoI coverage algorithm proposed in this section, these properties are used in order to preserve the network connectivity and to cope with the scalability issues, thus facilitating the movement computation.

4.2.2 Deployment algorithm for PoI coverage

At the beginning of the deployment, all the sensors are within both the communication range of the base station and the communication range of each other. Each sensor moves independently from the other sensors. All the sensors run the same algorithm, but their motion decisions are taken individually and the algorithm steps are not synchronized between the sensors. It is important to notice that the base station could compute an optimal placement for each sensor and provide them with this information, so that they would be able to move towards the optimal positions. However, by doing so, it is hard to ensure that the network would remain connected all throughout the deployment procedure since this approach is not robust to sensor failures. Therefore, the tracking of a moving PoI would result inaccurate because sensors may not have up-to-date information about position and placement.

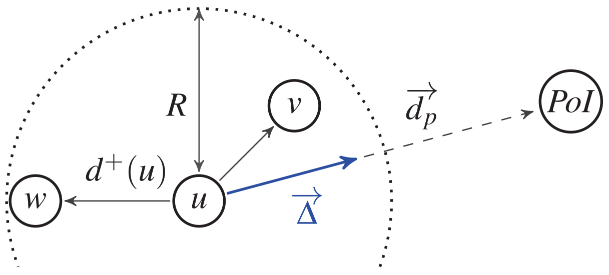
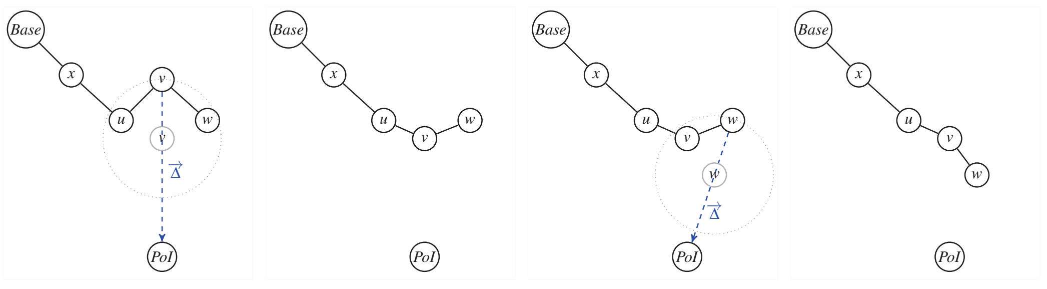


Figure 4.13: PoI coverage algorithm illustration.

In order to cover the PoI, sensors move towards one predefined point that could be the PoI itself or the barycenter of PoIs. These movements are constrained by the connectivity requirements and are implemented in deployment algorithm block (Section 3.2.4). While sensors are moving, they must maintain connectivity with their RNG neighbors of the dynamic graph. Indeed, even if a sensor does not cover the PoI, it must stop moving in order to maintain the connection with its RNG neighbors. It is worth noting that, when a sensor covers the PoI, it also stops its movement. The direction of a sensor is given by the following unit vector : $\overrightarrow{\Delta} = \overrightarrow{d_p}/||\overrightarrow{d_p}||$, where $\overrightarrow{d_p}$ is the vector connecting the current position of the sensor with the PoI (Figure 4.13). When a sensor has computed $\overrightarrow{\Delta}$, it will move in this direction. However, the distance travelled by the mobile sensor is constrained by maintaining connectivity with its RNG neighbors. Thus, the movement vector of a sensor is $\overrightarrow{m} = d \cdot \overrightarrow{\Delta}$, where d is the maximum distance that the sensor can travel while maintaining connectivity with its RNG neighbors.

Figure 4.14 shows an example of sensor movements. It is shown how



(a) Sensor v decides to move (b) Sensor v moves (c) Sensor w decides to move (d) Sensor w moves

Figure 4.14: Example of sensor movement.

sensors move toward the PoI and how connectivity is preserved by maintaining the connectivity with the RNG neighbors. It is worth noting that x is a neighbor, but not an RNG neighbor, of sensor v. The sensor v does not move at distance R from the sensor u due to the upper bound on the distance $d \leq (R - d^+(u))/2$ (detailed explanation about this constraint on d can be found in Section 4.2.2.1, in the proof of Theorem 1). Following conditions are set for the maximum distance d:

1. $d \leq (R - d^+(u))/2$,
2. if $d < \varepsilon_1$, then $d = 0$,
3. if $d < \varepsilon_2$, then $d = 0$.

where R is the communication range, $d^+(u)$ is the distance from sensor u to its farthest RNG neighbor, ε_1 and ε_2 are two threshold values that are constant for each sensor.
Condition (1) ensures that sensor u and $RNG^+(u)$ remain connected to each other, even in the case of movements in opposite directions, as it will be formally proven by mathematical demonstration in Section 4.2.2.1.
Condition (2) will be used to avoid an infinite sequence of sensor movements by introducing the minimal distance (ε_1) that a sensor can travel.
Condition (3) will be used to stop sensor movement when their distance to the PoI is below the threshold ε_2.

Algorithm 2 (PoI Deployment Algorithm, PDA) formally describes the deployment process that is implemented in deployment algorithm block of the robot middleware (Section 3.2.4). In an asynchronous environment, sensors can run PDA at any time. In the first part of the PDA, the sensor u computes its direction based on its own position and the coordinates of the PoI, thus the sensor can compute the movement direction $\vec{\Delta}$. In the second part, the sensor

Algorithm 2 Single PoI deployment algorithm (PDA) that runs on all the sensors.

```
Require: The PoI location $p(x,y)$.
Ensure: The coverage of the PoI $p(x,y)$.
1: repeat
2:     Check RNG neighbors' positions
3:     $\overrightarrow{\Delta} = \frac{\overrightarrow{d_p}}{||\overrightarrow{d_p}||}$
4:     $d = (R - d^+(u))/2$
5:     Sensor movement using the direction $\overrightarrow{\Delta}$ and distance $d$
6: until the PoI is reached
```

u calculates the distance to travel, and it performs the actual movement. The calculated distance must take the connectivity constraint into account by considering the worst case movement of the RNG neighbors of the sensor. Since the connectivity with the farthest RNG neighbor ($d^+(u)$) must be preserved, the moving distance should always satisfy Condition (1). Recall that all the sensors run the same algorithm. Therefore, if a sensor $v = RNG^+(u)$, then $d(u,v) \leq d^+(v)$. This inequality ensures that if sensor v is the farthest RNG neighbor of sensor u, and sensor u is not the farthest RNG neighbor of a sensor v, connectivity is still preserved between these two sensors. As stated in Section 2.2, the usage of virtual force based movement implies a step by step computation of sensor movements. At the end of the Part 2 in the algorithm, the sensor knows the distance it has to travel and it proceeds with the real movement. After the movement is done, the steps are repeated until the PoI is reached. The deployment algorithm is illustrated in Figure 4.15.

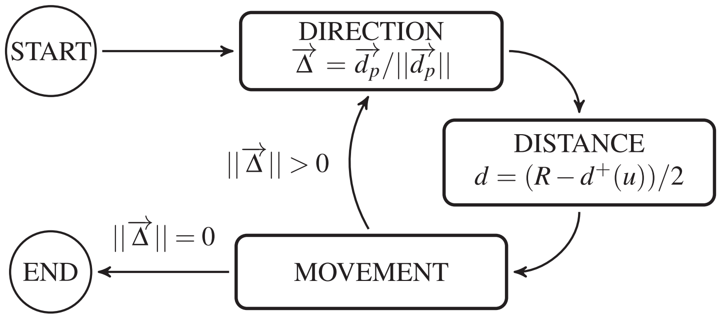

START
DIRECTION
$\overrightarrow{\Delta} = \overrightarrow{d_p}/||\overrightarrow{d_p}||$
DISTANCE
$d = (R - d^+(u))/2$
$||\overrightarrow{\Delta}|| > 0$
$||\overrightarrow{\Delta}|| = 0$
END
MOVEMENT

Figure 4.15: PoI deployment algorithm.

The two parts of the PDA are related to two important aspects of deploying a fleet of mobile sensors. The first part is related to the deployment scheme while the second part guarantees connectivity preservation and sensor movement. Since this algorithm is divided into two separate parts, it is easy to modify the parts independently from one another, and thus use different direction calculation techniques following the generic approach driven by the design of the middleware architecture.

4.2.2.1 Algorithm properties

Theorem 1 *Connectivity. If at time $t = T_1$ the graph is connected, $\forall t = T_2$, with $T_2 > T_1$ the resulting graph is connected.*

Proof 1 *Let u and v be two sensors and u and v are connected at time $t = T_1$. Let $u \in RNG(v)$, $v \in RNG(u)$ and $d(u,v) = d^+(u)$. Assume that two sensors run PDA at the same time and that they are moving in opposite directions. The maximum distance traveled by sensor v depends on $d(u,v)$. Since $d(u,v) \leq d^+(v)$ the maximum distance traveled by sensor v is $d_v = (R - d^+(v))/2 \leq (R - d^+(u))/2$. Therefore, the maximum distance between sensor u and v after their movements are $d(u,v) + (R - d^+(u))/2 + (R - d^+(v))/2 \leq R$. Thus, after their movements, sensors u and v are still connected. If the connection to the farthest RNG neighbor is maintained, then the connection to closer RNG neighbors is maintained as well. Therefore, if the connectivity with RNG neighbors is preserved, network connectivity is also preserved [86].*

Theorem 2 *Termination. There exists a time $t > T_3$ when all the sensors stop moving.*

Proof 2 *Assume the case where sensor deployment is composed of the base station $b(x_b, y_b)$, the PoI $p(x_p, y_p)$ and the mobile sensor $u(x_u, y_u)$. At the beginning of the deployment ($t = 0$), $d(u^{(0)}, b) < R$. After the first iteration $d(u^{(1)}, b) = d(u^{(0)}, b) + (R - d(u^{(0)}, b))/2$ after the i^{th} iteration,*

$$d(u^{(i)}, b) = \frac{1}{2^i}\left((2^i - 1)R + d(u^{(0)}, b)\right), \tag{4.1}$$

therefore:

$$\lim_{i \to \infty}\left(R - d(u^{(i)}, b)\right) = 0. \tag{4.2}$$

Hence, there exists a $t > T_3$ such that condition $R - d(u^{(i)}, b) < \varepsilon_1$ is satisfied, and sensor u stops moving. The same proof also holds for an arbitrary number of sensors. Moreover, if the number of available sensors is large enough to reach and cover the PoI while maintaining connectivity, then the value of T_3 can be further reduced by satisfying Condition (3), $d(u,p) < \varepsilon_2$.

Theorem 3 *Straight line deployment. Let $b(x_b, y_b)$ be the base station, $p(x_p, y_p)$ be the position of the PoI and let us assume that sensor $u(x_u, y_u)$ is not on the segment $[b, p]$. The distance h between sensor u and the segment $[b, p]$ is strictly decreasing.*

Proof 3 *At each step of the deployment algorithm, sensor u moves toward the PoI. Since the direction of the sensor movement is $\overrightarrow{up}$, where u is the sensor position and the traveled distance is $d \geq 0$, the distance between a sensor and the PoI is strictly decreasing. As a consequence, the distance between the sensor and the segment $[b, p]$ is also decreasing. It is worth noting that when the sensor $u \in [b, p]$, it remains on the segment during the movement and $h = 0$.*

Theorem 3 shows that the deployment is more likely to place sensors along the straight line between the base station and the point of interest.

Theorem 4 *Minimization of number of connectivity sensors. If the PoI $p(x_p, y_p)$ is at distance $d = \infty$, at the end of the deployment each sensor has two RNG neighbors at the most.*

Proof 4 *Without loss of generality, it can be considered that the PoI is at distance $d = \infty$ for two reasons. First, this assumption implies that sensors are moving in parallel to the segment $[b, p]$. Second, it ensures that no sensor can reach the PoI, therefore all sensors are connectivity sensors.*

If the deployment terminates, then the distance between the generic sensor u and one of its neighbors v is $d(u, v) > R - \varepsilon_1$. In order to better understand the proof, consider the configuration depicted in Figure 4.16.

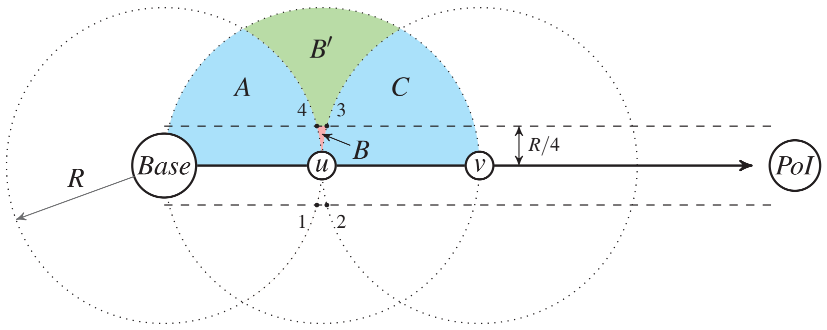


Figure 4.16: Minimization of number of connectivity sensors.

In this configuration, sensors u and v cannot move anymore since they are at distance $R - \varepsilon_1$ from b and u, respectively. It is also important to notice that due to Theorem 3, sensors stay at a distance of $R/4$ from the segment $[b, p]$ at

the most. Assume that the sensor u has more than two RNG neighbors when the deployment is complete. In this case, the sensor $w \in RNG(u)$ exists. In the configuration depicted in Figure 4.16, w must fall into one of the surfaces indicated by A, B, B' or C.
Case A or C: *If w falls into surface A (or C), $w \in RNG(u)$, but $b \notin RNG(u)$ (or $v \notin RNG(u)$). Therefore, $d(b, w) \leq R - \varepsilon_1$ (or $d(v, w) \leq R - \varepsilon_1$) and w can move. Which is contrary to the assumption that the deployment terminated.*
Case B': *sensor w cannot fall into surface B', since the assumption is that sensors are at the maximum distance $d = R/4$ from the base station at the beginning of the deployment, and Theorem 3 ensures that this distance is decreasing.*
Case B: *If the sensor w falls into surface B, $w \in RNG(u)$ and Theorem 3 is verified. However, if $w \in B$, $d(u, w) \leq R - \varepsilon_1$ and thus w can move, which is contrary to the assumption that the deployment is complete. This proof can be extended to any configuration since the maximum distance between u and the set of intersection points 1, 2, 3 and 4 is $\max d(u, i) = R\sqrt{2 - \sqrt{3}}$, for $i = \{a, b, c, d\}$. This is the case, when b and u are located on the bottom dashed line. Therefore, if $w \in B$, then $d(u, w) \leq R\sqrt{2 - \sqrt{3}} < R - \varepsilon_1$, $\forall \varepsilon_1 < R(1 - \sqrt{2 - \sqrt{3}})$.*

The theorems above show that this deployment algorithm preserves connectivity all along the deployment procedure. Furthermore, provided proofs show that the deployment will eventually terminate and that, at the end of the deployment, sensors used for connectivity are more likely to form a straight line and to be at distance $R - \varepsilon_1$ from each other. It is also shown that, at the end of the deployment, each sensor used for connectivity has two RNG neighbors at the most, which proves that the number of connectivity sensors is minimized.

4.2.3 Static PoI

In this section deployment simulations performed by using WSNet [97] simulator are provided. In the simulations, the communication range is set to be equal to the sensing range but this assumption can be easily modified without affecting the behavior of the deployment. Simulation parameters are given in Table 4.1.

Figure 4.17 shows an example of the deployment's evolution where the PoI is located at position $p(70, 100)$. After $180s$, the deployment is finished. In the simulation setup, the sensors move during five seconds and compute a new direction after their movements. This figure shows that the sensors

Table 4.1: Summary of the simulation parameters in PoI coverage campaigns.

Field size	$100m \times 100m$
Sensing range	$10m$
Communication range	$10m$
Decision period δ	$5s$
Maximal speed	$1m/s$

form a straight line between the base station and the PoI, which reduces the number of sensors used for connectivity preservation and therefore increases the number of sensors involved in coverage.

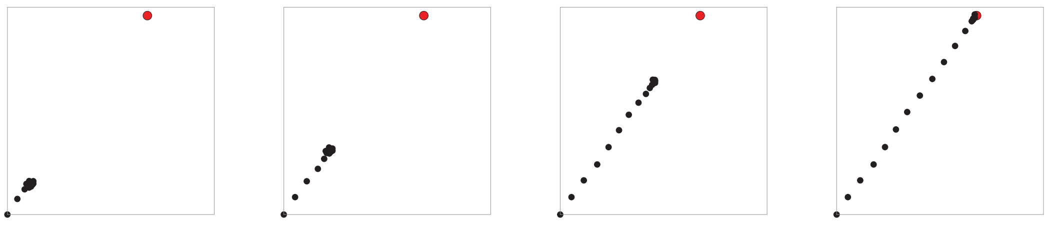
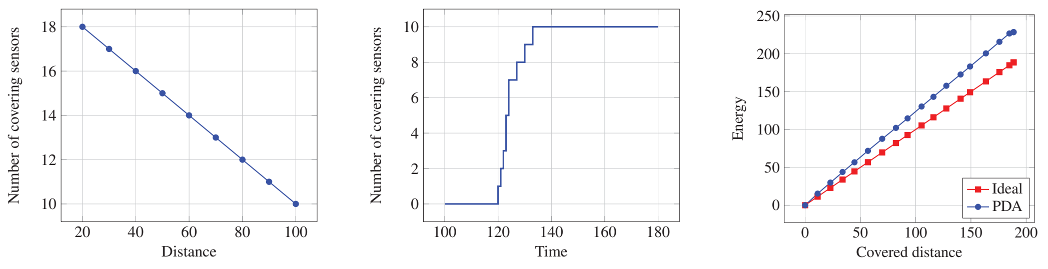

(a) 30s (b) 60s (c) 120s (d) 180s

Figure 4.17: Evolution of sensors' positions depending on time. In this simulation there are 20 sensors with a range of 10 on a square of 100×100. The PoI is located at $p(70, 100)$.

4.2.3.1 Coverage quality, speed and energy consumption

Figure 4.18(a) presents the number of covering sensors w.r.t. the distance between the PoI and the base station. In the simulation, the base station is considered as a sensor which is not mobile. That is, the network comprises 20 sensors including the base station. This figure shows that the number of sensors used for connectivity is minimized and that the number of covering sensors is maximized. For example, when the PoI is at distance 40, we need 3 sensors for connectivity at distances 10, 20, 30 and the base station at distance 0, which means that 4 sensors are needed for connectivity and 16 sensors can cover the PoI for a total of 20 sensors.

Figure 4.18(b) plots the number of covering sensors depending on time. In this simulation, PoI is at distance 100 and 20 mobile sensors are considered. A movement decision is taken every δ, where δ is arbitrarily set to $5s$. This figure shows that the first PoI is covered by at least one sensor after $120s$. Note here that we check the coverage every $1s$. This means that the first covering sensor has a mean speed of $0.75m/s$ ($90m$ covered distance after $120s$).



(a) Number of covering sensors w.r.t distance.

(b) Number of covering sensors w.r.t time.

(c) Energy consumed w.r.t distance.

Figure 4.18: Coverage quality, deployment speed and energy consumption.

Note that in an ideal case, the distance a sensor can cover at each step is R meters (the communication range) and each step lasts for $\delta = 5s$ (motion decision). This means that the maximum (in the best case) speed of a sensor is : $\nu = 10/5 = 2m/s$. Note here that since the goal is to ensure connectivity, a sensor must consider the worst case movement of its neighbors since the assumption is that the sensors are not synchronized. Therefore, the maximum speed is reduced to $\nu = 2/2 = 1m/s$ to take the connectivity constraint into account. These results are very close to the results obtained above. One way to increase deployment speed is to reduce the motion decision period or to increase the communication range.

In order to evaluate the deployment algorithm energy consumption, it is compared with the ideal deployment, the deployment where all the sensors are provided with their final positions and where they move towards them without any movement or connectivity constraint. For both cases, a simple energy model is considered, a model where the energy consumed by the sensor u is: $E(u) = d\alpha + \beta$, where d is the covered distance and α and β are constants (here $\alpha = 1$, $\beta = 1$). This simple energy model considers the distance covered by a sensor but also penalizes multiple small movements.

Figure 4.18(c) shows the energy consumption of each sensor for a deployment of 20 sensors and a PoI at $p(100, 100)$. This figure shows that the energy consumption is linear depending on the covered distance. Moreover, PDA consumes small amount of energy since (for example) for a covered distance of $105m$, 130 energy units are needed. It can be noticed that a sensor can cover $R/2 = 5m$ in every movement decision period since it has to maintain connectivity with its neighbors. Therefore, the sensor needs at least $105/5 = 21$ iterations to cover $105m$. The energy consumed by the sensor is at least $E(u) = 105 \times 1 + 1 \times 21 = 126$ which is very close to 130. In order to reduce energy consumption by removing periodical algorithm calculations and movements, each sensor can be given its final destination at the beginning of the deployment. However this deployment cannot guarantee connectivity during the deployment, is not robust against obstacles, and is not suitable for the coverage of moving PoI.

4.2.4 Moving PoI

In this section, a single moving PoI is considered. When a set of sensors is deployed over a given PoI, the sensing application may require the sensor to move to another location. This scenario is possible with the PDA algorithm since it maintains connectivity all along the deployment procedure.

There are three different strategies for covering a new PoI when the sensors are already deployed. In the first strategy (Algorithm 3), hereafter referred to as `STR1`, the sensors first move back to the base station before deploying toward the new location of the PoI. This first strategy provides a high coverage quality but increases the deployment duration and the amount of energy consumed.

Algorithm 3 First tracking strategy (STR1).

```
Require: The PoI location $p(x_p, y_p)$.
Ensure: The coverage of the PoI $p(x_p, y_p)$.
 1: Run the PDA to reach the base station $b(0, 0)$
 2: Run the PDA to cover the PoI $p(x_p, y_p)$
```

In the second strategy (Algorithm 4), hereafter referred to as `STR2`, the sensors try to move directly toward the location of the PoI without going back to the base station. This second strategy reduces the time needed to cover the new PoI but also reduces the coverage quality since an increasing number of sensor is needed to preserve connectivity.

Algorithm 4 Second tracking strategy (STR2).

```
Require: The PoI location $p(x_p, y_p)$.
Ensure: The coverage of the PoI $p(x_p, y_p)$.
 1: Run the PDA to cover the PoI $p(x_p, y_p)$
```

The third strategy (Algorithm 5) is a mix of `STR1` and `STR2` and is referred to as `STR3`. In this strategy, sensors move toward the segment $[b, p]$ and when the distance between the particular sensor and the segment is lower than $R/4$, the sensor moves toward the PoI. This strategy combines the advantages of `STR1` and `STR2`.

Algorithm 5 Third tracking strategy (STR3).

```
Require: The PoI location $p(x_p, y_p)$.
Ensure: The coverage of the PoI $p(x_p, y_p)$.
 1: Run PDA in order to reach the segment $[b, p]$
 2: Run the PDA to cover the PoI $p(x_p, y_p)$
```

4.2.4.1 Example of deployment for moving PoI

Figure 4.19 shows the example of deployment for different tracking strategies. Figures 4.19(a) to 4.19(e) show the deployment using `STR1`. This set of figures shows that after $450s$, the deployment reaches its end and that the first covering sensor reaches the PoI between $[350 - 450]s$. Figures 4.19(f) to 4.19(j) show the deployment using `STR2`. This set of figures also shows that the deployment terminates after 7 hours but that the PoI is reached after $300s$. The long termination time is mainly due to the fact that sensors can only make small movements since they are at a distance close to R of each other. Figures 4.19(k) to 4.19(o) show that the deployment using `STR3` terminates after $900s$ and that the PoI is first reached between $[350 - 900]s$, which is a consequence of this deployment strategy being a trade-off between `STR1` and `STR2`.

4.2.4.2 Coverage quality and deployment speed

The coverage quality and the deployment speed of each strategy are evaluated at this point. A set of simulations with duration of $3000s$ and including 20 sensors are run. The PoI in the simulations move at a random location every $500s$. Figure 4.20 plots the number of covering sensors depending on time, coverage quality and (re)deployment speed for these three strategies.

Figure 4.20 shows that each new PoI location is covered by at least one sensor for each strategy. It also shows that from the coverage quality point of view, `STR1` shows very good performances compared to other strategies. Actually, if the coverage of the last PoI is considered (between $[2500-3000]s$), `STR1` has more than 15 covering sensors, `STR2` has less than 5 covering sensors and `STR3` has around 7 covering sensors. More generally, when using `STR1` the coverage quality depends only on the distance between the base station and the PoI which is not the case for `STR2` and `STR3`. From the redeployment speed point of view, `STR2` shows very good performance. In the interval $[1000-1500]s$ the PoI is covered at most after $10s$ since the number of covering sensors is sampled every $10s$. For `STR1`, $200s$ are needed and for `STR3`, $30s$ are needed. At the time interval $[500 - 1000]$ the PoI is located at $p(93, 27)$ and between $[1000 - 1500]s$ it is at $p(75, 1)$.

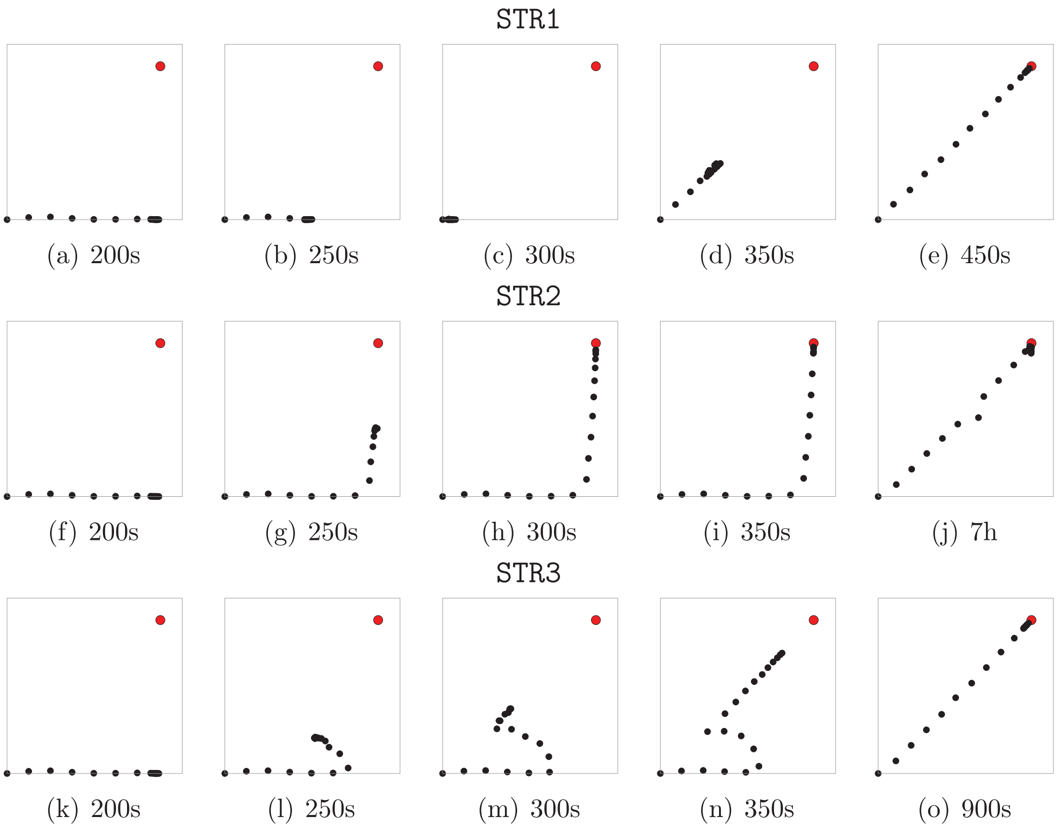


Figure 4.19: Evolution of sensors' position depending on time. In this simulation there is 20 sensors with a range of 10 on a square of 100×100. The PoI is first located at $p'(70, 0)$ and then at $p''(70, 70)$ after $200s$.

These results show that when the PoI is moving or when the sensor redeployment is needed, three proposed strategies have their advantages and drawbacks but they keep the properties described in Section 4.2.2.1 such as connectivity and termination. Note that the trade-off proposed with `STR3` can be optimized depending on the application requirements. Moreover, it could be of interest to use `STR1` or `STR2` depending on the distance between the old and the new location of the PoI or any other metric, such as angle.

The implementation of these algorithms is easily done in the movement direction block of the middleware architecture described in Chapter 3.

4.2.5 Multiple PoIs

Starting from the single PoI case (PDA, Algorithm 2), the initial algorithm is extended and two approaches for multiple PoI coverage are designed: Random PoI Deployment Algorithm (R-PDA) and Barycenter PoI Deployment Algorithm (B-PDA). The management of the multiple PoI case is done by

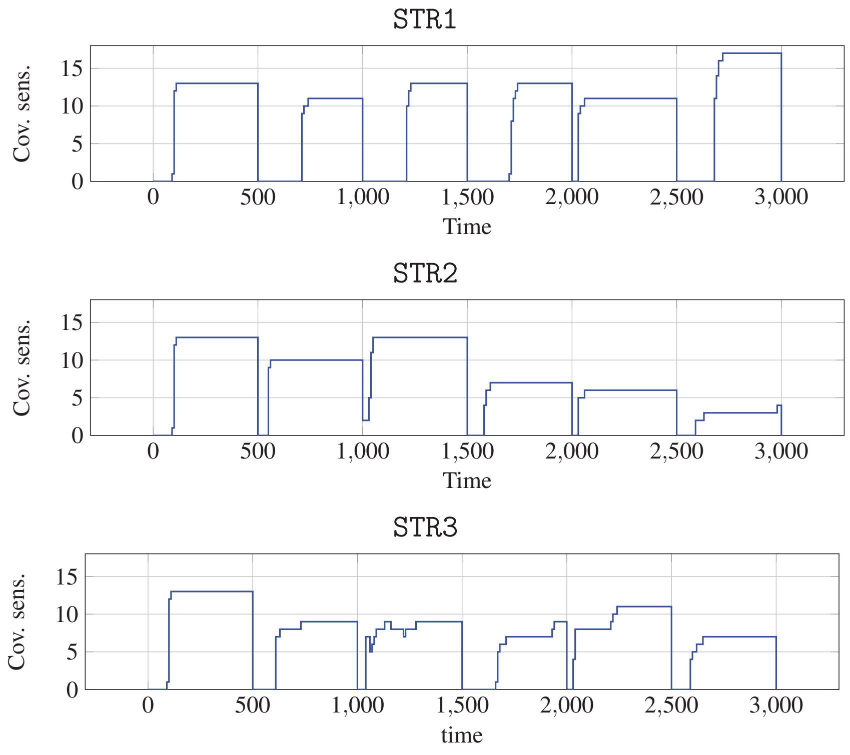


Figure 4.20: Number of covering sensors w.r.t time. Simulation parameters: $R = 10$, 20 sensors including the base station. The simulation lasts $3000s$. A new location of the PoI is chosen every $500s$.

downgrading the problem complexity to single PoI problem, followed by the utilization of PDA.

The first approach in multiple PoI coverage is the application of PDA to a multiple PoI coverage scenario, where each sensor in the deployment randomly chooses one of the PoIs and runs the PDA. Coverage of all the targets will be achieved if it is assumed that a large enough number of sensors is used for the deployment. The deployment process is named Random PoI Deployment Algorithm (R-PDA) and is formally described in Algorithm 6.

Algorithm 6 R-PDA, multiple PoI coverage algorithm where sensor u randomly chooses one of the PoIs and moves to cover it by using PDA.

```
Require: Positions of all the PoIs.
Ensure: Multiple PoI coverage.
 1: Randomly choose one of the PoIs, $p_{rand}(x, y)$
 2: Run the PDA to cover the chosen PoI
```

Second approach relies on covering the barycenter of all the PoIs and the base station location, before the application of R-PDA. In this approach, every sensor calculates the location of the barycenter ($B(x, y)$) for all the PoIs and the base station and runs the PDA to cover it. After covering the calculated barycenter, the sensors run the R-PDA in order to cover a given set of PoIs. This deployment process is formally described in Algorithm 7 and named Barycenter PoI Deployment Algorithm (B-PDA). Note that we can also consider the Steiner tree [96] in order to choose intermediate points instead of barycenter. The use of Steiner tree would minimize the number of relay nodes, thus improving the deployment, since only the optimal intermediate points would be chosen. However, the calculation of Steiner tree between the given set of targets represents a hard problem to solve, and therefore it is not feasible on the computationally weak mobile robots.

Algorithm 7 B-PDA, multiple PoI coverage algorithm that uses the barycenter of PoIs and base station as the intermediate point.

```
Require: Positions of all the PoIs.
Ensure: Multiple PoI coverage.
1: Calculate the barycenter ($B(x, y)$) for all the PoIs and the base station
2: Run the PDA to cover $B(x, y)$
3: Run the R-PDA to cover all the PoIs
```

Since both R-PDA and B-PDA directly use the PDA, the proofs of network connectivity and deployment termination are trivial and, therefore, will be omitted. The properties of PDA assure that the final deployment comprises straight line segments, that the number of connectivity sensors is minimized and that the number of covering sensors is maximized.

4.2.5.1 Example of deployment for multiple PoIs

Figure 4.21 shows the example of deployment for R-PDA and B-PDA respectively. In these simulations, two PoIs ($p_1(90, 50)$ and $p_2(50, 90)$) and 30 sensors are considered. For R-PDA, it is considered that the set of sensors is divided into two subsets and each subset is assigned to one PoI. Figure 4.21 shows that for R-PDA the deployment terminates after $180s$ and that the PoIs are considered independently. For B-PDA, the gravity center of the two PoIs and the base station is chosen as an intermediate point. In B-PDA (as in R-PDA), each sensor is also assigned to a given PoI by the base station. However, before effectively moving toward its PoI, the sensors need to reach the intermediate point. Figure 4.21 shows the two steps of the deployment for B-PDA.

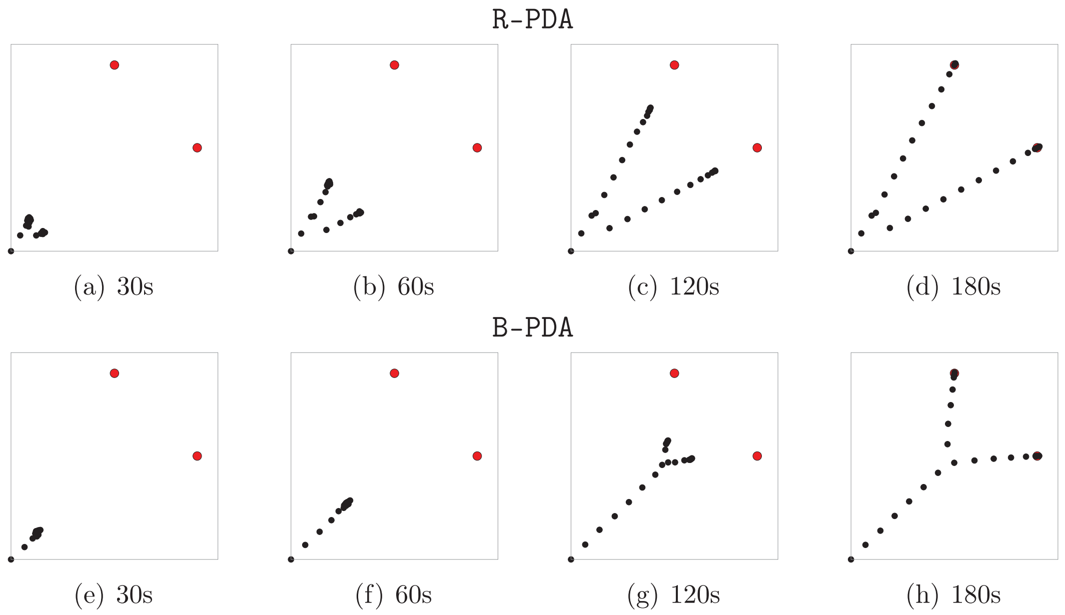


Figure 4.21: Sensors' positions with multiple PoIs depending on time.

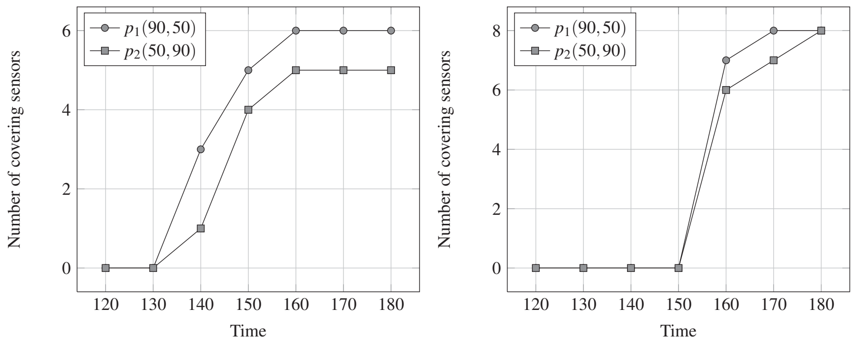
4.2.5.2 Coverage quality and deployment speed

Figure 4.22 plots the number of covering sensors depending on time for the two families of deployment strategies. This figure shows the trade-off performance between deployment speed and coverage quality. Indeed, R-PDA outperforms B-PDA regarding deployment speed since the two PoIs are covered by at least one sensor at $140s$ for R-PDA and this value is $160s$ for B-PDA. However, the coverage quality provided by B-PDA is better than the coverage provided by R-PDA since the maximum number of covering sensors is 6 for R-PDA and 8 for B-PDA. Note that for R-PDA, the number of covering sensors is not equal for the two PoIs since in the simulation setup 30 sensors are considered, including the base station. Therefore, 14 sensors are dedicated to one PoI and 15 sensors are dedicated to the other. This is not the case for B-PDA since a subset of sensors is used in common for connectivity.

Similarly to the implementation of STR1, STR2 and STR3, both R-PDA and B-PDA are easily implemented in the movement direction block of the middleware architecture from Chapter 3.

4.2.6 Implementation using Wifibots

The deployment algorithm for Wifibots is implemented by using the middleware architecture shown and described in Chapter 3. In order to cope with deployments in unknown environments, a simple obstacle avoidance technique



(a) R-PDA (b) B-PDA

Figure 4.22: Number of covering sensors w.r.t time for R-PDA and B-PDA.

is integrated into the initial PDA by taking the output of the outer sensors block into account. If a robot detects an obstacle during the deployment, it tries to cover the auxiliary PoI ($p_A(x_A, y_A)$) based on the gathered sensory information about the obstacle. After the auxiliary PoI is reached, it continues with initial PoI coverage steps. Note that in case of obstacle detection during the auxiliary PoI coverage, obstacle avoidance steps are run iteratively in the deployment algorithm block, until all the auxiliary PoIs are covered or the boundary of the communication range is reached. This deployment process is formally described in Algorithm 8 and referred to as Implemented PoI Deployment Algorithm (I-PDA).

Algorithm 8 Implemented PoI deployment algorithm (I-PDA) that runs on all the robots.

```
Require: The PoI location $p(x,y)$.
Ensure: The coverage of the PoI $p(x,y)$.
1: repeat
2:     Check RNG neighbors' positions
3:     $\overrightarrow{\Delta} = \frac{\overrightarrow{d_p}}{||\overrightarrow{d_p}||}$
4:     $d = (R - d^+(u))/2$
5:     Movement using the direction $\overrightarrow{\Delta}$ and distance $d$.
6:     if obstacle detected then
7:         Run I-PDA for auxiliary PoI $p_A(x_A, y_A)$
8: until the PoI is reached
```

4.2.6.1 Testing results

In first deployment example, the PoI coverage of $p(40, 60)$ is observed, while the communication range of all robots is set to $15m$. The PoI is covered after approximately $160s$ and deployment frames are shown in Figure 4.23. It can be seen in these examples how the deployment actually works – all the robots are moving to a known target with constraints regarding movement forward and constantly preserving connection with the base station. As the group of robots advances towards the target, after reaching the boundary of the communication range, the closest robot to the base station stops with its movement, thus creating a communication path back to the base station.

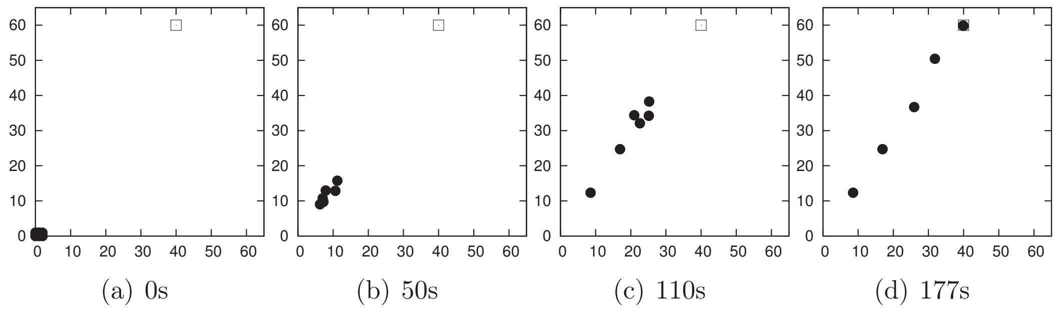


Figure 4.23: Coverage of $p(40, 60)$ with comm. range of $15m$ (6 robots).

Figure 4.24 presents deployment examples with $p(70, 100)$, where communication ranges are set to $20m$ and $15m$ respectively. In both examples the PoI is covered after approximately $260s$ which shows that the deployment time is not related to the number of robots or communication range, but only to robots' maximal speed (under the assumption that there are enough available robots to reach the PoI).

The situation where a group of robots cannot reach the target is shown in Figure 4.25. Five robots are trying to reach the target with their communication range set to $8m$, but after approximately $90s$ they all reach the end of their communication range and therefore complete deployments stops.

Figure 4.26 shows deployment results after the implementation of R-PDA and B-PDA (as presented in Section 4.2.5). In this experimentation 8 Wifibots are used, while the communication range is set to $15m$ and two PoIs are located at $p_1(25, 45)$ and $p_2(45, 25)$.

Figure 4.27 shows the example of deployment for a single PoI, but with an obstacle in the deployment field. In this experimental scenario, 3 Wifibots and a base station $b(0, 0)$ are used to cover the PoI $p(0, 11)$. The communication range is set to $4.5m$ while all other parameters stayed the same as in simulation campaigns. This example illustrates the behavior of the implemented obstacle

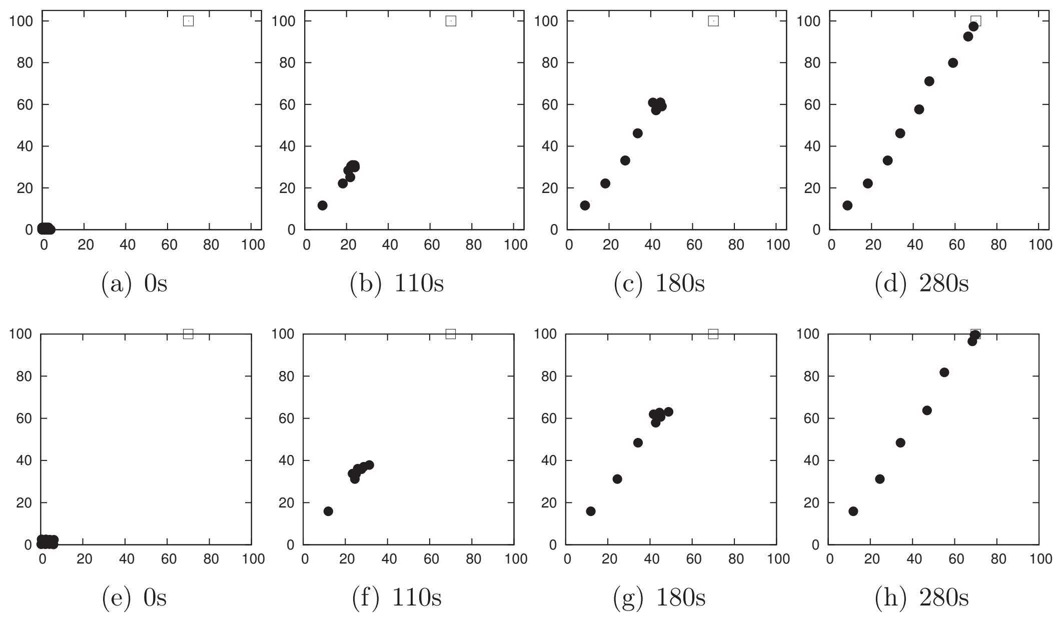


Figure 4.24: Coverage of $p(70, 100)$ with comm. ranges of $20m$ and $15m$, respectively (9 robots).

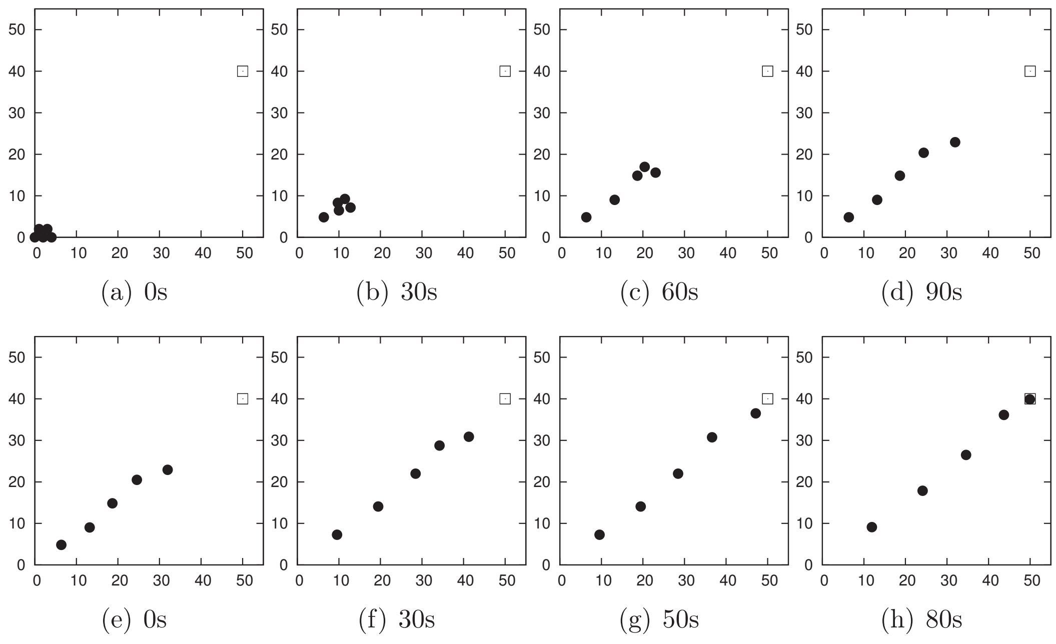


Figure 4.25: Coverage of $p(50, 40)$ with changing the comm. range from $8m$ to $15m$ (5 robots).

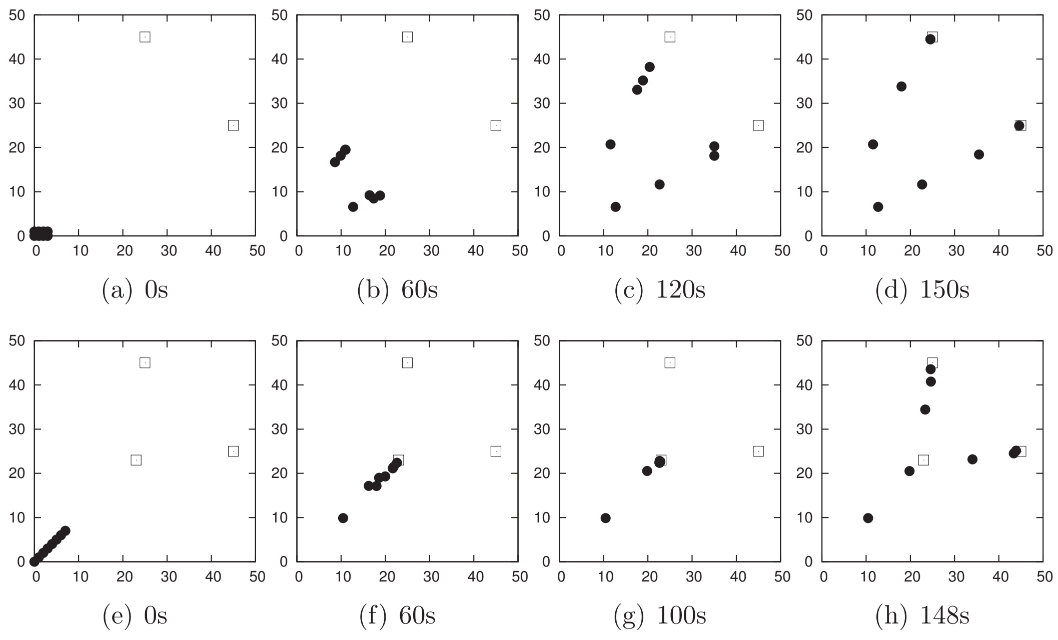


Figure 4.26: Implementation of R-PDA and B-PDA for two PoI coverage, $p_1(25, 45)$ and $p_2(45, 25)$.

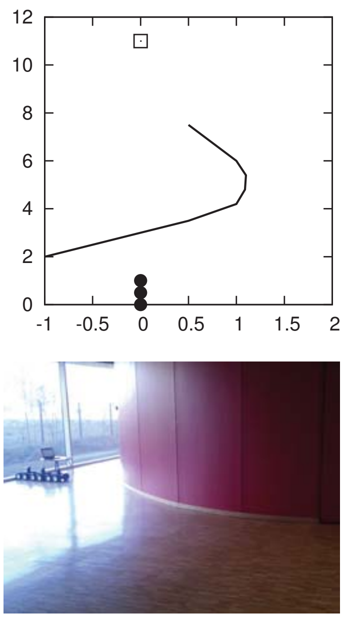
avoidance technique in an indoor environment. Figure 4.27 provides photos of the environment and the robots during the deployment.

4.2.7 Wifibot energy model

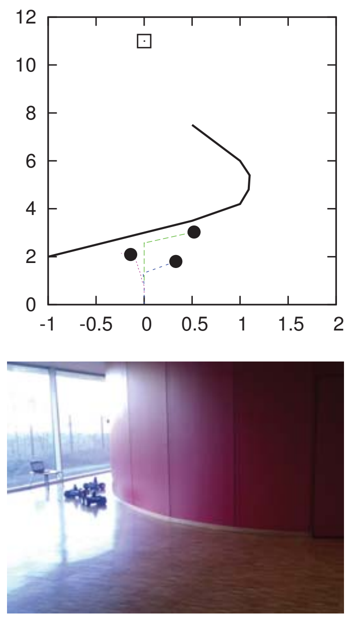
By measuring the power consumption during the robot movement, it is possible to create a Wifibot energy model that can be used in the simulation campaigns for precise energy consumption estimation. The energy consumption information during the robot movement is obtained from the robot middleware motor control block described in Chapter 3.

During the execution of the PDA algorithm, robot varies its movement speed from the zero value v_0 (the case there the robot does not move, *standby mode*) to the desired speed $v < v_{max}$ (*movement mode*, v_{max} represents the maximal obtainable robot speed, $0.95m/s$ for Wifibots). Figure 4.28 shows the evolution of desired and actual speed of Wifibot during the movement of $5m$. The speed in Figure 4.28 is represented in motor encoder ticks per $50ms$, where the value of 140 corresponds to the velocity of $0.5m/s$. Furthermore, the same figure shows the real-time consumed power during the same movement.

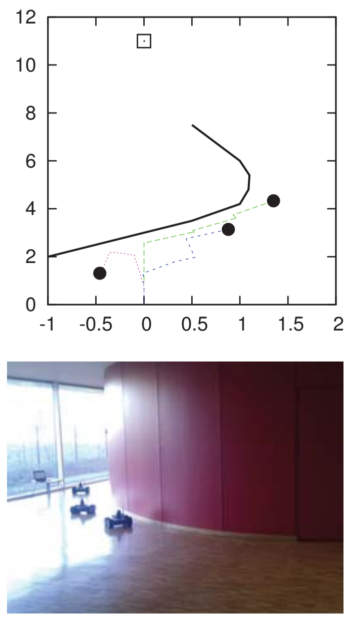
By running the PDA, a robot can stop with its movements either because it has reached the end of its communication range, or it has reached the PoI. In both cases, the movement velocity is $v = v_0 = 0$. In order to quantify



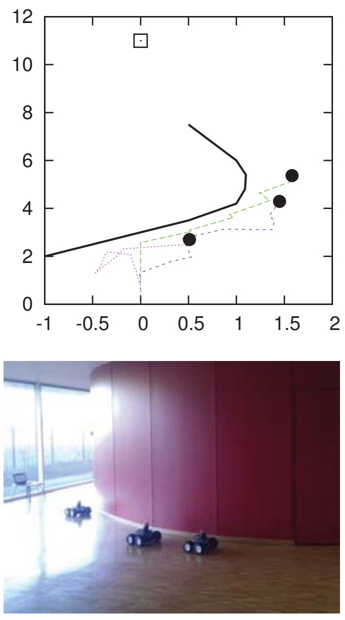
(d) 0s



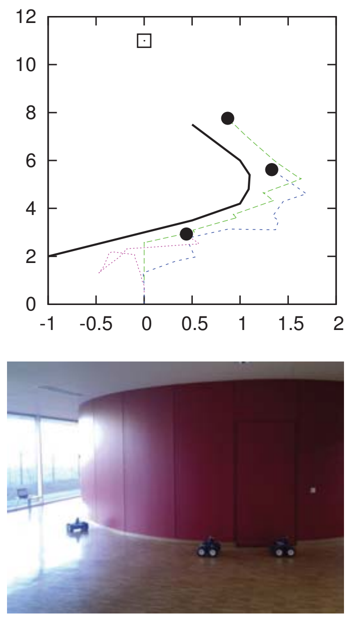
(e) 10s



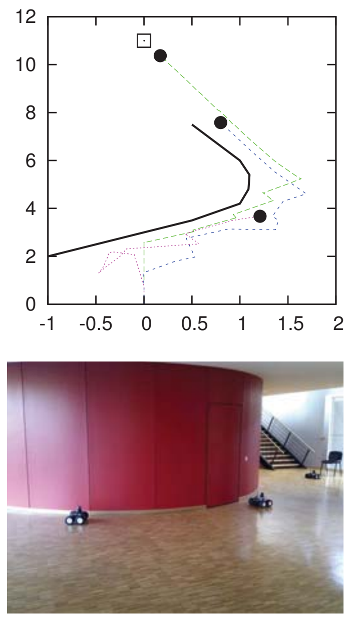
(f) 30s



(j) 40s



(k) 50s



(l) 60s

Figure 4.27: Wifibot deployment with an obstacle.

the robot energy consumption while not moving (standby mode), the average power during the period of $T = 600$s is measured. Obtained results show that

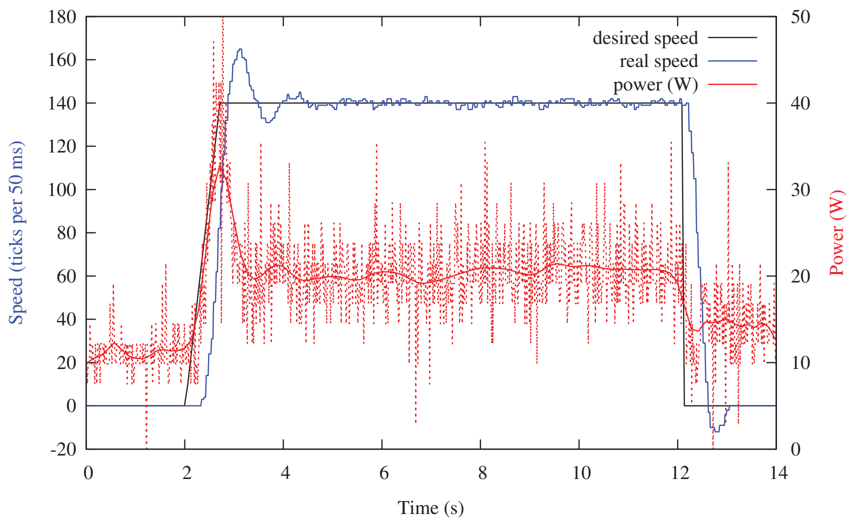


Figure 4.28: Speed and power data during the 5m movement. The speed of 140 ticks per 50ms corresponds to the speed of 0.5m/s.

the power in standby mode (P_0) is $P_0 = 8.568$ W. Likewise, in order to quantify the power drawn from the battery during the movement, the desired speed is varied from 0.1m/s to 0.9m/s with the step of 0.1m/s and the consumption power is measured during the period of $T = 120$s. The obtained results are presented in Figure 4.29. The power depending on the movement velocity can be expressed as $P_{mov}(v,t) = 11.55v(t) + 14.838$ W. The robot varies its velocity during the deployment, thus both the velocity $v(t)$ and the power $P_{mov}(v,t)$ are the functions of time.

Since the energy consumption E during the period T is calculated as the time integral of power, by combining the standby mode power P_0 and the movement mode power P_{mov}, the desired energy model can be expressed as:

$$E = \begin{cases} \int_0^T P_0 \, \mathrm{d}t = 8.568T \ [J] & \text{if } v = 0 \\ \int_0^T P_{mov}(v,t) \, \mathrm{d}t = 11.55 \int_0^T v(t) \, \mathrm{d}t + 14.838T \ [J] & \text{if } v > 0. \end{cases}$$

In order to verify the energy model, a set of simulation results is compared to the measured energy consumption on 5 Wifibots ($S_1..S_5$) used in the implementation. The communication range is arbitrarily set to $15m$, and the PoI is located at position $p(45,45)$. Since all the robots are located in the vicinity of the base station (located at the position $(0,0)$), the maximal distance that a robot must pass before covering the PoI is $d <= 63.7$m. The desired movement speed of robots both in simulations and the implementation is set

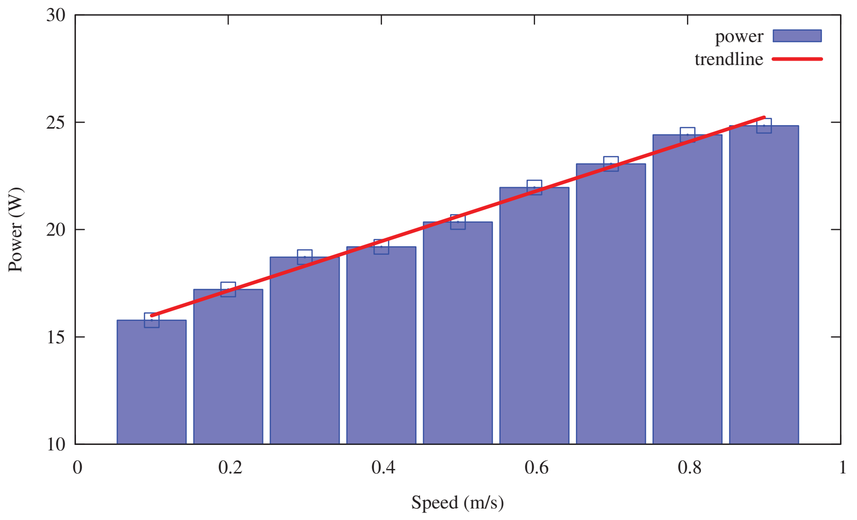


Figure 4.29: Average power during the movement with different speeds (from 0.1 to $0.9m/s$).

Table 4.2: Simulation and experimentation results for energy model verification

	Simulation		Implementation		Difference	
	Distance	Energy	Distance	Energy	Distance	Energy
S_1	14.46 m	1490.4 J	14.43 m	1605.5 J	-0.2 %	7.7 %
S_2	28.88 m	1800.9 J	28.95 m	1833.3 J	+0.3 %	1.8 %
S_3	43.33 m	2109.2 J	42.64 m	2041.3 J	-1.6 %	-3.2 %
S_4	57.76 m	2415.3 J	57.91 m	2410.6 J	+0.3 %	-0.2 %
S_5	62.97 m	2523.1 J	63.22 m	2526.5 J	+0.4 %	-0.1 %

to $0.5m/s$.

The simulated energy consumption and actual energy consumed during the implementation for each robot are presented in Table 4.2 and Figure 4.30. The results show that the energy consumption values are correlated, with a small difference in the energy of the robots closer to the base station and high accuracy of predicted energy consumption for the robots that are closer to the PoI. Obtained result prove that the proposed energy model accurately describes the energy consumption of Wifibot robots, especially for the robot as the distance that robots travel increases.

4.2.8 Conclusion

This section presented the algorithm for PoI coverage with mobile wireless sensors. Mobile robots have the objective to cover the PoI while maintain-

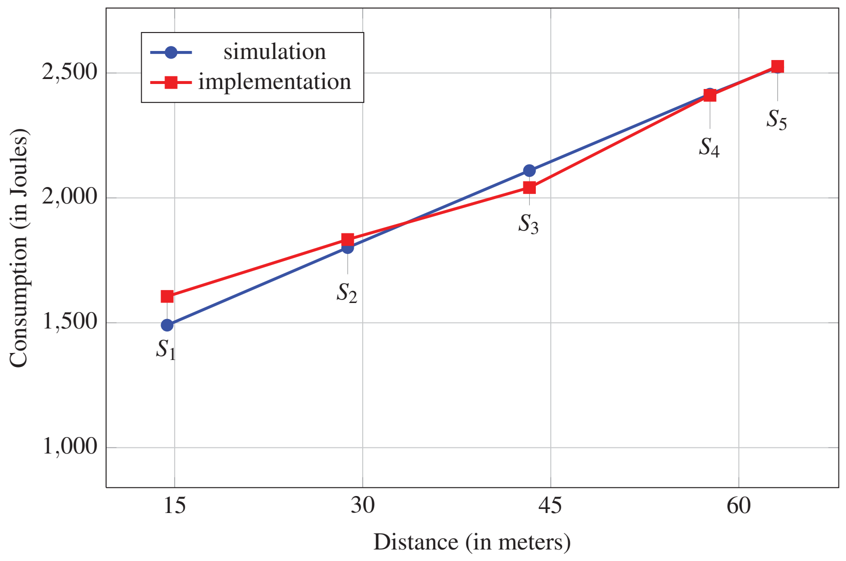


Figure 4.30: Simulation and implementation results for total consumption with relation to the robot position.

ing the connectivity with a fixed base station. The algorithm is distributed, needs only local information at each sensor, does not require synchronization and is divided into direction computation, connectivity maintenance and movement. The connectivity preservation all along the deployment is done by using the properties of the Relative Neighborhood Graph. The performance of the algorithm is evaluated regarding the number of sensors that cover the PoI, deployment speed and energy consumption. Formal proofs for connectivity preservation, algorithm termination and the shape of the resulting graph are provided as well. Furthermore, the algorithm is implemented using middleware architecture for Wifibots.

4.3 PoI discovery and coverage with mobile sensors

Very often environmental monitoring applications require the knowledge of both the position of the PoI to cover and the characteristics of the monitored area. Obtaining all the necessary information about the environment is not an easy task, especially if the dynamic nature of the observed processes is taken into account. In this section, the goal is to develop a mobile sensor deployment algorithm that combines environment and PoI discovery along with coverage and connectivity preservation. However, the PoI discovery and coverage are opposing demands if the same set of devices is used for both operations. In order to maximize the PoI coverage in the field of interest, mobile sensors have to self-deploy in a certain manner and to adjust their positions according to the placement of PoIs that still need to be discovered, which excludes the application of any standard environment exploration techniques.

The approach described in this section is based on the continuous and variable speed movement of mobile sensors, which follow concentric circular paths to explore and cover the field of interest. By constantly moving, sensors execute the environment discovery task and, by adjusting the movement velocity, they satisfy the constraints on PoI coverage and connectivity with the data sink. The algorithm that runs on all the mobile sensors is distributed and introduces a new technique of velocity calculation based on the information available from the sensors in one-hop neighborhood. The complete design of the algorithm is intended to be used with the middleware architecture described in Chapter 3. The contribution of this particular work is twofold:

- analytical expressions for mobility parameters are derived under the constraint of circular movement with the purpose of discovering and covering several PoI, while maintaining intermittent but bounded connectivity with the data sink;

- the approach is evaluated by extensive simulation campaigns in different simulation scenarios.

4.3.1 General assumptions

Constant improvements in mobile sensor technology allows us to use different types of mobile sensors. In recent years, flying drones are more and more used in military and civil applications for environmental monitoring. An interesting property of this type of mobile devices is that they consume approximately the same amount of energy while moving or staying still. Thus, it is possible to develop different dynamic deployment strategies for different types of expected coverage [89]. By using this kind of mobile sensors (which we also refer to as sensors and flying drones) that are able to move freely and without energy consumption constraints, a novel approach to solve the aforementioned problems is presented. The assumption is that the static sink and mobile nodes are placed or dropped in the field of interest and since the sink is static, it is considered as the center of concentric paths that will be followed by mobile sensors.

Mobile sensor velocity is calculated in a distributed manner and it is a function of the available information about PoIs, sensor characteristics and network topology. This mobile sensor movement scenario is based on concentric circular paths with the data sink in the barycenter (Figure 4.31). Mobile robots move by following the predefined paths. Circular movement is used for the sake of simplicity, but the approach can be equally applied onto the assumption of different closed curve configuration.

Another assumption is that no global knowledge concerning the set of PoIs is available and that sensors are able to deploy themselves in order to get to their starting positions on the circular paths before the main algorithm execution. Regarding the starting positions, two cases are analyzed: configuration with only *one* sensor per path and configuration with *multiple* sensors on the path.

The primary goal of deployment approach is the coverage of multiple PoIs whose positions are not known at the beginning of the deployment. By moving on the concentric paths, mobile sensors achieve PoI *discovery*, multiple PoI *coverage* and *connectivity* with the data sink.

4.3.1.1 Formal definitions

The Unit Disk Graph (UDG) network model is used to formally describe the deployment and coverage model of a sensor S [18]. In this model, sensor's communication and sensing ranges are represented with r_c and r_s, respectively. Without loss of generality, it is assumed that $r_c > 2r_s$ (Figure 4.32(a)). The ratio between the communication and sensing range, r_c/r_s, is referred to as *path density* (D_L).

Definition 6 The path *is the closed curve that is followed by mobile sensors, denoted by L_n, where n represents the number of the path or* path level. *Paths L_n and L_{n+1}, as well as L_{n-1} and L_n, are called* neighboring paths.

In order to ease the calculation process, without loss of generality, it is assumed that all the paths are circles. The distance between neighboring paths is set to $2r_s$, since this distance allows the sensors on neighboring paths to cover the area in between without overlapping (Figure 4.32(b)). The distance of the first path from the sink is set to r_s. Therefore, the distance of all the points of the path L_n from the barycenter is $d_n = (2n-1)r_s$ and the path length is $l_n = 2d_n\pi = 2(2n-1)r_s\pi$.

Definition 7 The stripe *is the annulus referred to as ξ_n, $2(n-1)r_s$ and $2nr_s$ are its inner and outer radius, respectively. It represents the region covered by the sensor that moves following the path L_n, with the area $4(2n-1)r_s^2\pi$.*

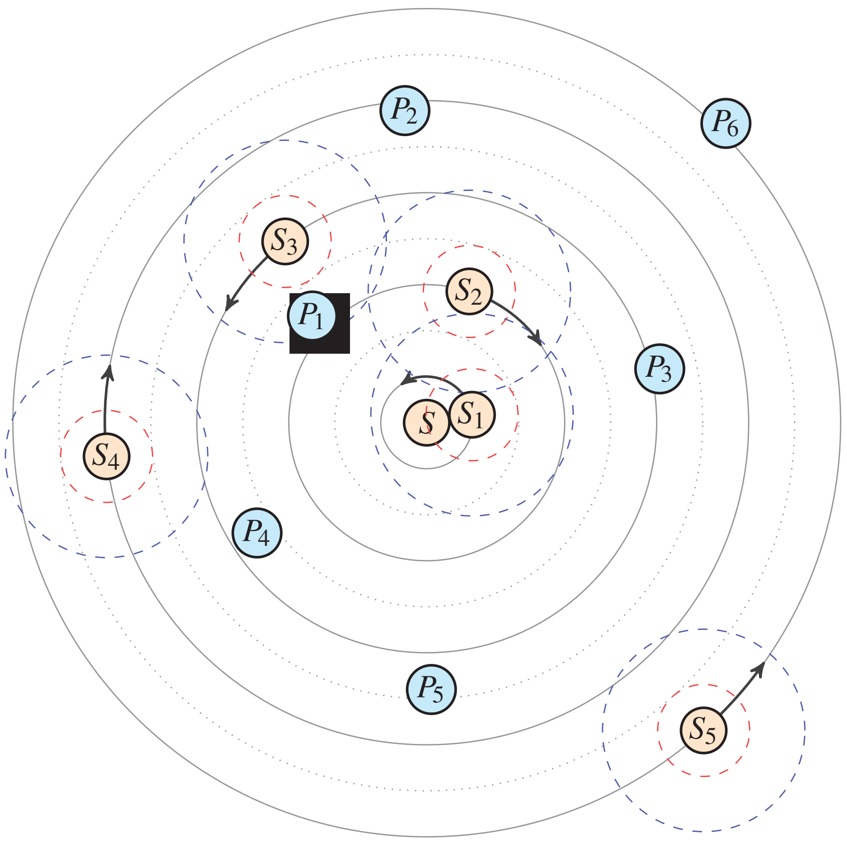


Figure 4.31: Mobile sensors (S_i) follow the concentric circular paths and cover the PoIs (P_i). Node S represents the data sink.

Definition 8 *The* Point of Interest *(PoI) is the point in the area of interest where the event occurs (denoted by P_i in general case), defined with its polar coordinates: the distance from the data sink r_P as the radial coordinate and the angle from the predefined direction Θ_P as the angular coordinate.*

The assumption is that there is an unknown number of PoIs in the area of interest and that the PoIs are placed in unknown locations (r_{P_i}, Θ_{P_i}). Moreover, without loss of generality, it is assumed that all the PoIs have equal dynamics of alternation between active and dormant states, described by the period T_P. A specific PoI is considered to be covered if it lays within the sensing range r_s of a sensor for a period of time greater than T_{sens}, that is the period of time in which a sensor can get all the necessary information about the event occurring at the PoI. After the information about the PoI is retrieved by the sensor, it has to be transferred to the sink in a multi-hop fashion. Similarly to T_{sens}, T_{comm} is defined as the period of time necessary to provide successful communication between two sensors. It is assumed that $T_{sens} > T_{comm}$.

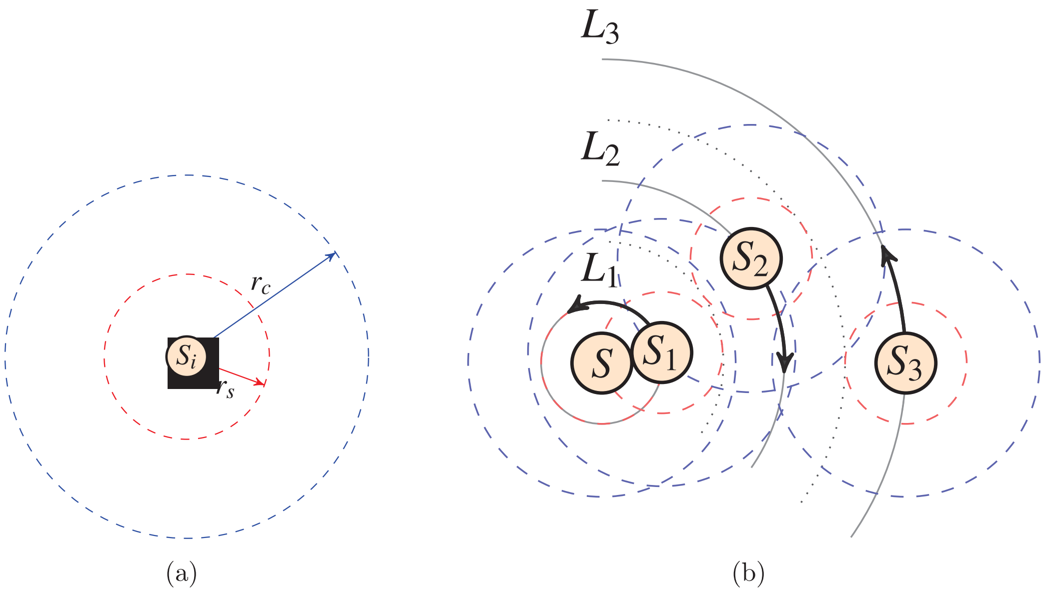


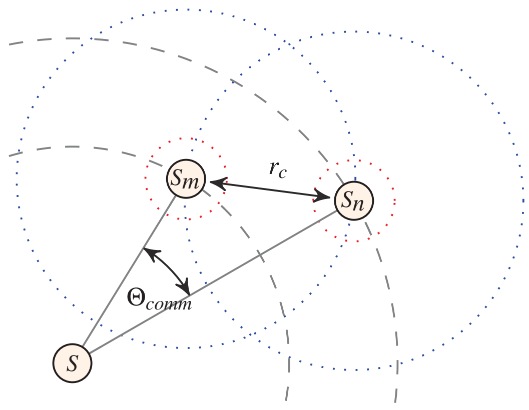
Figure 4.32: (a) Mobile sensor with its communication (r_c) and sensing (r_s) ranges. The ratio between communication and sensing range is referred to as path density, D_L. (b) The distance between neighboring paths is set to $2r_s$ in order to minimize the overlap of sensing ranges.

Due to the PoI dynamics, it is necessary to transfer the PoI data from a sensor to the sink as fast as possible, and therefore, the critical period of time T_{data} is introduced, in which the information about the PoI should be delivered to the sink from the sensor. The complete network is considered to be connected if all the messages from the sensors containing PoI information are transferred to the sink with a delay lower than T_{data}. In order to capture all the information regarding the PoI, a mobile sensor has to cover it with period lower than the period of PoI's dynamics T_P.

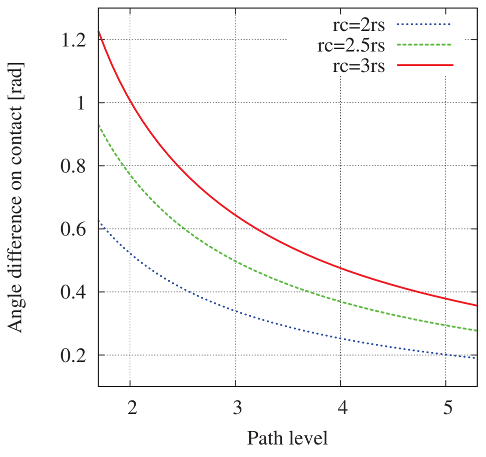
In the remainder of the work, the sensor moving on the path L_n will be referred to as *sensor* S_n or n^{th} *sensor*. Moreover, sensors following neighboring paths are referred to as *neighboring sensors*.

4.3.2 Circular movement analysis

In order to formally describe the movement of the sensors and the conditions of their interconnection, it is first needed to determine the angle between the barycenter and the positions of two sensors in the moment when they enter (or exit) each other's communication range ($\Theta_{comm}(n, m)$, shown in Figure 4.33(a)).



(a)



(b)

Figure 4.33: (a) The angle between the barycenter and two sensors on neighboring paths in the moment when they enter (or exit) each other's communication ranges. (b) Communicating sensors' angle difference between n and $n-1$ path levels ($\Theta_{comm}(n, n-1)$), for different values of the path density $D_L = r_c/r_s$.

Lemma 5 *The difference between angular coordinates of sensors on paths L_m and L_n ($n > m$) in the moment of entering (or exiting) each other's communication range, denoted by Θ_{comm}, is expressed as:*

$$\Theta_{comm}(n, m) = \arccos \frac{(2m-1)^2 + (2n-1)^2 - D_L^2}{2(2m-1)(2n-1)} \tag{4.3}$$

Proof 5 *The application of the cosines law to the triangle SS_nS_m gives that $r_c^2 = ((2m-1)r_s)^2 + ((2n-1)r_s)^2 - 2(2m-1)(2n-1)r_s^2 \cos \Theta_{comm}(n, m)$. Therefore, it is possible to deduce the expression for $\Theta_{comm}(n, m)$.*

In the case of sensors on neighboring paths, where $m = n-1$, $\Theta_{comm}(n, m)$ can be simplified to $\Theta_{comm}(n)$, as presented in Equation 4.4. Figure 4.33(b) is the plot of function $\Theta_{comm}(n)$ for sensors on different paths (from path 2 to 5, since the sensor on first path is constantly connected to the data sink) and 3 arbitrarily chosen path densities ($D_L \in \{2, 2.5, 3\}$). It is clear that a higher path density permits a larger sensor movement while preserving the connection with the communicating sensor on the neighboring path.

$$\Theta_{comm}(n) = \frac{4n^2 + 1 - D_L^2}{4n^2 - 1} \tag{4.4}$$

Further in the text, $\Theta_{comm}(n)$ is used for the calculation of minimal intercontact time between mobile sensors on neighboring paths. In the following sections, a reader can find the analysis of sensor velocity during the communication $v_{comm}(n)$, inter-contact period of time $T_{inter}(n)$ for both cases of movements, in the same and in opposite directions, $T^{+}_{inter}(n)$ and $T^{-}_{inter}(n)$, respectively, and PoI coverage velocity $v_{sens}(n)$.

4.3.2.1 Movement in the same direction

In the general case, sensors move at their maximal movement velocity v_{max}, and eventually slow down in the case of contact with other sensors or PoIs. Therefore, in order to satisfy the given communication period T_{comm} and keep the connectivity, sensors in mutual contact have to travel at the communication velocity $v_{comm}(n)$. If the calculated velocity is greater than cruising velocity v_{max}, then the sensor keeps v_{max} as the movement velocity, otherwise it slows down to a velocity $v_{comm}(n) < v_{max}$.

The angular velocity ω of the sensor during the movement is calculated as the fraction of linear velocity v and the circular movement radius r.

Lemma 6 *In order to satisfy the minimum communication period T_{comm} during the movement on paths L_n and L_{n-1}, sensors must limit their maximal velocity to:*

$$v_{comm}(n) = \frac{(2n-1)(2n-3)r_s\Theta_{comm}(n)}{T_{comm}} \tag{4.5}$$

Proof 6 *Let us consider the movement of the sensors in the same direction on paths L_n and L_{n-1}. In order to satisfy the T_{comm} constraint, they have to move with velocity $v_{comm}(n)$. Thus, their angular velocities are $\omega_n = v_{comm}(n)/((2n-1)r_s)$ and $\omega_{n-1} = v_{comm}(n-1)/((2(n-1)-1)r_s)$. The condition for keeping the connection between the sensors is $\omega_{diff}T_{comm} = 2\Theta_{comm}(n)$, where $\omega_{diff} = \omega_{n-1} - \omega_n$. Therefore, the maximal movement velocity that allows both sensors to keep their connection during the period T_{comm} is given in Equation 4.5.*

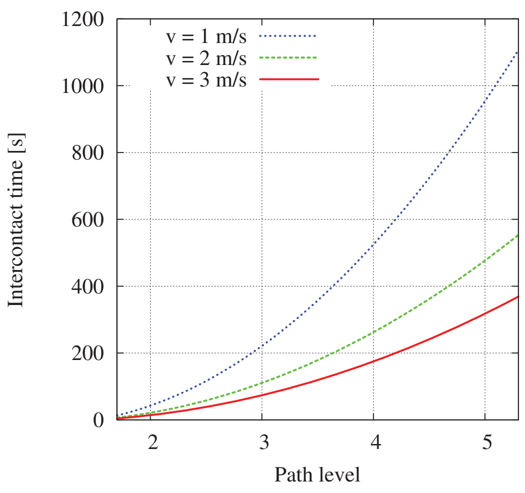
After the communication with sensor S_{n-1} is done, sensor S_n restores the maximal velocity v_{max} in order to minimize the inter-contact time with sensor S_{n-1}.

Lemma 7 *Under the assumption that sensors move at velocity v_{max} after losing their connection, the needed time to establish a new connection is:*

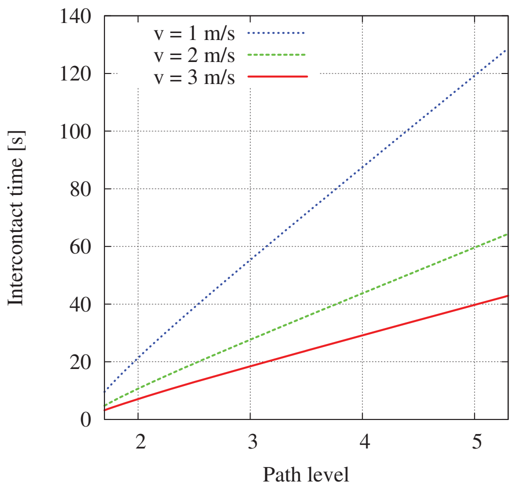
$$T^{+}_{inter}(n) = \frac{(2n-1)(2n-3)r_s}{2v_{max}}(2\pi - \Theta_{comm}(n)) \tag{4.6}$$

Proof 7 *Inter-contact time can be calculated by using the difference in neighboring sensors' angular coordinates, in other words, inter-contact time is the time needed to lower the sensors' angular difference to the value of* $\Theta_{comm}(n)$*. Neighboring sensors* n *and* $n-1$ *are both moving in the same direction at the velocity* v_{max}*, thus their angular velocities are* $\omega_n = v_{max}/((2n-1)r_s)$ *and* $\omega_{n-1} = v_{max}/(((2(n-1)-1)r_s)$*, respectively. The inter-contact time* $T^+_{inter}(n)$ *is the time needed for a sensor moving at the angular speed* $\omega_{diff} = \omega_{n-1} - \omega_n$ *to travel the angle* $2\pi - \Theta_{comm}(n)$*. Since* $\omega_{diff} = 2v_{max}/((2n-1)(2n-3)r_s)$*, then* $T^+_{inter}(n) = (2n-1)(2n-3)r_s(2\pi - \Theta_{comm}(n))/2v_{max}$*.*

Figure 4.34(a) shows the inter-contact time as a function of the number of paths involved, for 3 different maximal movement velocities that are chosen arbitrarily. As a consequence of the longer distance that sensors on higher paths have to travel and the nature of the movement in the same direction in the same linear velocity, inter-contact periods are unacceptably high.



(a) Movement in the same direction.



(b) Movement in opposite directions.

Figure 4.34: Inter-contact time for sensors on neighboring paths in the case of single sensor per path.

4.3.2.2 Movement in opposite directions

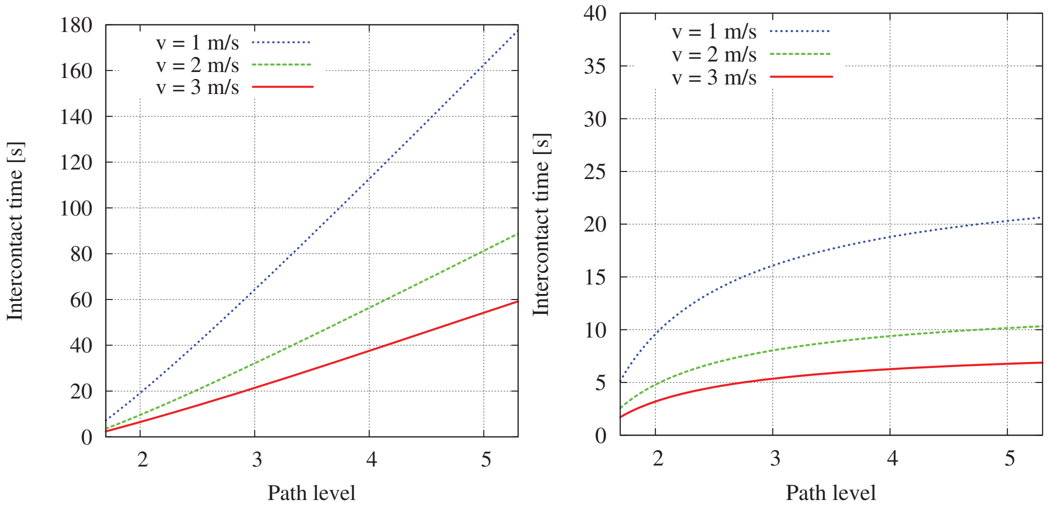
To lower the inter-contact period, movement in opposite directions on neighboring paths is used. Since the connection time between two sensors on neighboring paths should be at least T_{comm}, in this case, their velocity should be lower or equal to $v_{comm}(n)$, where $v_{comm}(n)$ in this case is calculated similarly to Lemma 6:

$$v_{comm}(n) = \frac{(2n-1)(2n-3)r_s\Theta_{comm}(n)}{4(n-1)T_{comm}} \tag{4.7}$$

Following the reasoning from the proof of Lemma 7, it is easy to deduce the inter-contact time for the opposite direction movement for sensors on neighboring paths (Figure 4.34(b)):

$$T^{-}_{inter}(n) = \frac{(2n-1)(2n-3)r_s(2\pi - \Theta_{comm}(n))}{4(n-1)v_{max}} \tag{4.8}$$

4.3.2.3 Multiple sensors on the path



(a) Movement in the same direction. (b) Movement in opposite directions.

Figure 4.35: Inter-contact time for sensors on neighboring paths in the case of multiple sensors per path.

Further step in lowering the inter-contact time is the case where multiple sensors are placed on the same path (at equidistant positions). In this case, the number of sensors on the path is equal to path level. Similarly to Equations 4.6 and 4.8, expressions for inter-contact time are derived, both for movements in the same (Equation 4.9) and in the opposite directions (Equation 4.10), for the case where there are multiple sensors deployed on a circular path. It is worth noting that sensors on the same path *always* move in the same direction, differences in direction are related to the sensors on neighboring paths. The inter-contact time as a function of path number for different movement velocities is shown in Figure 4.35(b).

$$T^{+}_{inter}(n) = \frac{(2n-1)(2n-3)r_s}{2v_{max}}\left(\frac{2\pi}{n} - \Theta_{comm}\right) \tag{4.9}$$

$$T^{-}_{inter}(n) = \frac{(2n-1)(2n-3)r_s}{4(n-1)v_{max}}(\frac{2\pi}{n} - \Theta_{comm}) \tag{4.10}$$

In case of possible contact between the sensors following the same path, hence the sensors that move in the same direction, the sensor that senses its neighboring sensor from the same path lowers the speed thus increasing the distance between them and avoiding the collision.

Figures 4.34 and 4.35 show a significant decrease in neighboring sensor inter-contact time in the favor of movement in opposite directions with multiple sensors on a path, and therefore, this scenario is considered in approach evaluation (Section 4.3.3).

4.3.2.4 Detection of a PoI

Similarly to the case where the sensors interconnect, a sensor detecting the PoI has to change its movement velocity in order to satisfy the condition for T_{sens}.

Lemma 8 *Depending on the PoI distance from the path L_n (d_P), the angular coordinate difference Θ_{sens} (Figure 4.36) for the case when the PoI is on the boundary of the sensing range is:*

$$\Theta_{sens}(n) = \arccos \frac{(d_n + d_P)^2 + d_n^2 - r_s^2}{2d_n(d_n + d_P)}, d_n = (2n-1)r_s \tag{4.11}$$

Proof 8 *Similarly to the proof of Lemma 5, the case that comprises the sensor S_n, the base station S and the PoI P_i is considered. Applying the law of cosines to the triangle SP_iS_n gives that $|S_nP_i|^2 = |SS_n|^2 + |SP_i|^2 - 2|SS_n||SP_i|\cos\Theta_{sens}(n)$. Since $|SS_n| = (2n-1)r_s$, $|SP_i| = (2n-1)r_s + d_P$, $d_P = r_{P_i} - (2n-1)r_s$ and $|S_nP_i| = r_s$, it is therefore $r_s^2 = ((2n-1)r_s)^2 + ((2n-1)r_s + d_P)^2 - 2(2n-1)r_s((2n-1)r_s + d_P)\cos\Theta_{sens}(n)$.*

Lemma 9 *The maximal velocity of sensor movement needed to satisfy the sensing period T_{sens} is:*

$$v_{sens} = \frac{2(2n-1)r_s\Theta_{sens}(n)}{T_{sens}} \tag{4.12}$$

Proof 9 *The length of the path that sensor S_n can travel while covering P_i is $l = 2\Theta_{sens}(n)(2n-1)r_s$. Satisfying T_{sens} as the sensing constraint, the maximal movement velocity during the PoI sensing is, therefore, $v_{sens} = l/T_{sens}$.*

It is worth noting that if a sensor has to choose between several values for its actual movement velocity, it chooses the smallest value, which is also the least energy consuming, as shown in Section 4.2.7.

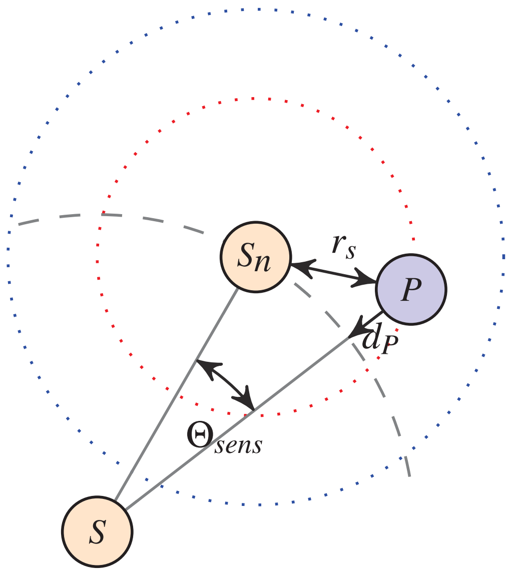


Figure 4.36: The difference between angular coordinates of sensor S_n and the PoI P_i in the moment when sensing begins.

4.3.3 Approach evaluation

This section provides the reader with the evaluation results of the deployment properties illustrated with the PoI discovery, network connectivity and network coverage results. Similarly to Section 4.2, the simulator used is WSNet. Table 4.3 provides the parameters that are used in simulations. In all the simulations, three different movements velocities are tested: 1, 2 and 3m/s. Values for T_{comm} and T_{sens} are arbitrarily chosen and set to $2s$ and $3s$ respectively.

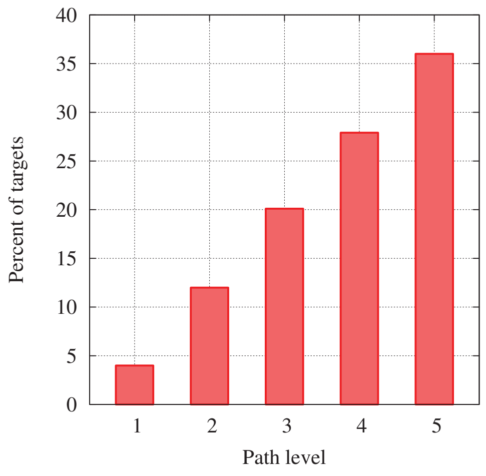
Table 4.3: Summary of the simulation parameters in PoI discovery campaigns

$r_c[m]$	$r_s[m]$	$T_{comm}[s]$	$T_{sens}[s]$	$T_{data}[s]$
11	5	2	5	100
$v_{max}[m/s]$	Area $[m^2]$	Duration [s]	Number of sensors	Number of PoIs
1,2,3	2500π	600	15	50

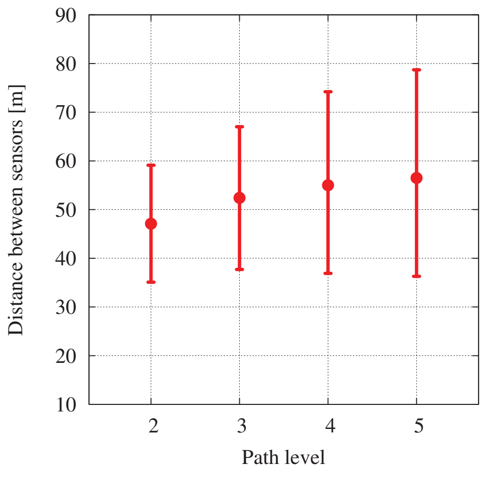
Table 4.4: PoI distance from the data sink and its covering sensor path level

PoI distance [m]	$0-10$	$10-20$	$20-30$	$30-40$	$40-50$
Path level	1	2	3	4	5

The positions of PoIs are chosen randomly in the given circular area, which gives the specific PoI distribution on circular paths as shown in Figure 4.37(a). The multiple sensor case is taken into consideration in the simulation scenarios. Specifically, 15 sensors are placed on the paths as follows: one sensor is placed on the first, two on the second, three on the third, four on the fourth and five sensors on the fifth path. For the sake of clarity, Table 4.4 provides the reader with the relation between the PoI distance from the data sink and its covering sensor path level.



(a)



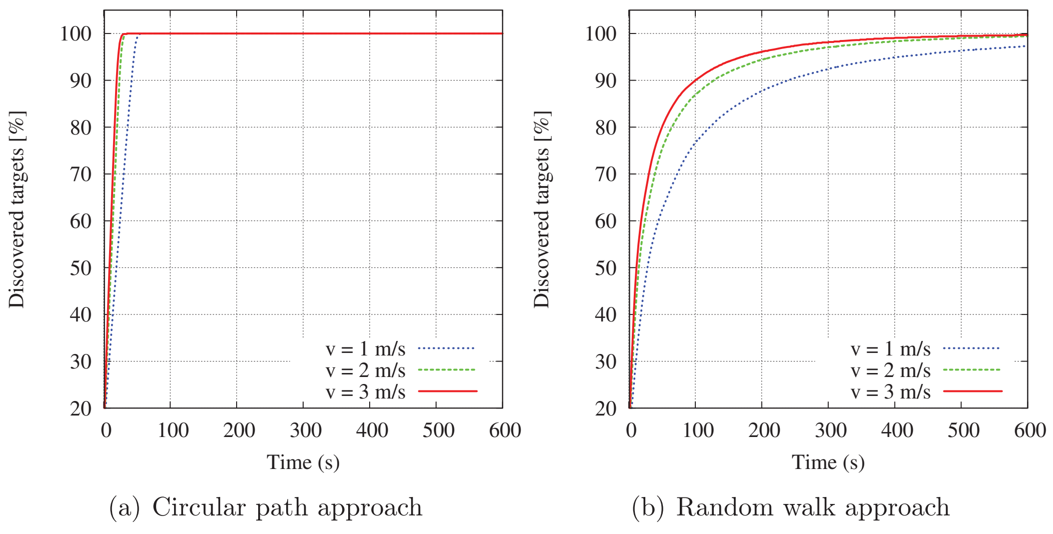
(b)

Figure 4.37: (a) Distribution of PoIs with relation to the path covering it and (b) the distance between the sensors on the path during the movement.

Sensors on the same path are distributed equidistantly in the beginning of the deployment, however, these distances change due to the contact with other sensors and PoIs in the field. Figure 4.37(b) shows the average, minimal and maximal distances between sensors on different paths.

4.3.3.1 PoI discovery

The PoI discovery is the first property of proposed movement scheme. It is necessary to measure the time needed to discover all the PoIs in the field of interest and to compare it with the random walk approach, with the same number of sensors and their initial placement. Random walk approach refers to the successive set of sensor movements with random distance and random direction. In each simulation run, 50 PoIs are randomly deployed in the field with the area of $2500\pi m^2$, while the maximal sensor movement velocity varies from 1 to $3m/s$ for both random walk and our circular paths approach.



(a) Circular path approach (b) Random walk approach

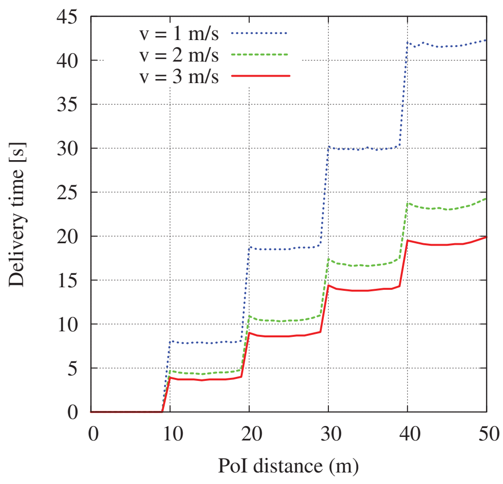
Figure 4.38: Percent of PoIs discovered after a certain time for both approaches and different movement velocities.

Figure 4.38 represents the percent of PoIs discovered after a certain time, which gives the measure of the reactivity of the network. It shows that the approach (Figure 4.38(a)) outperforms the random walk approach (Figure 4.38(b)), and that all the PoIs are discovered after $t = 35s$ and $t = 56s$ for $v_{max} = 3m/s$ and $v_{max} = 1m/s$, respectively. Random walk approach achieves the 99% coverage after $384s$, but fails to discover all the PoIs in the field during the simulated period of time.

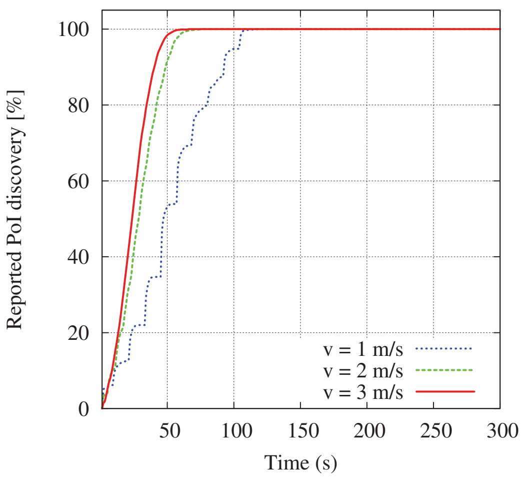
4.3.3.2 Network connectivity

After the PoI information has been captured, sensors deliver this information to the sink node by passing it to the neighboring sensor on lower path. Thus, the message delivery time can vary depending on inter-contact time between communicating sensors on neighboring paths and their initial placement on the path. Figure 4.39(a) shows the average message delivery time for different movement velocities when the distance from the PoI and the sink varies. Stepwise delivery function that depends on distance reflects the effect of different sensors on discrete levels.

The way to observe network connectivity is to evaluate the complete PoI discovery from the sink node's point of view. In contrast to Figure 4.38, the PoI is considered as discovered only when the report about it arrives on the sink node. Figure 4.39(b) shows that the time needed to report the information about the whole field is $118s$ for $v_{max} = 1m/s$ and $66s$ for $v_{max} = 3m/s$.

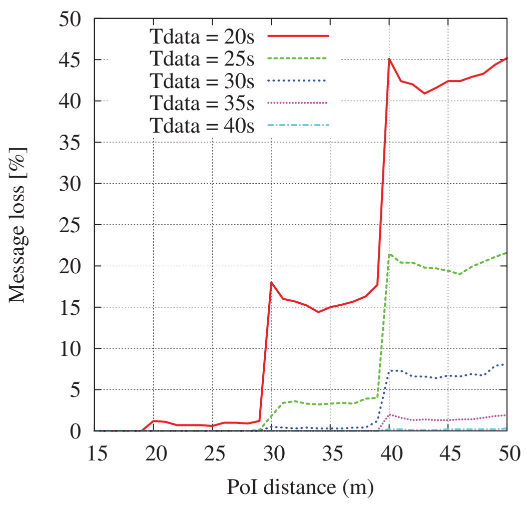


(a) PoI information delivery time

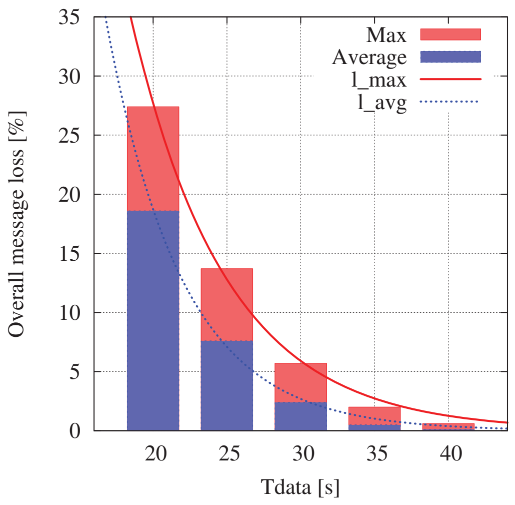


(b) Percent of reported PoIs

Figure 4.39: PoI information delivery time and the percent of reported PoIs to the sink node for circular path approach.



(a) Message loss with relation to the PoI distance from the sink



(b) Overall message loss with relation to T_{data}

Figure 4.40: Message loss related to T_{data} in circular path approach.

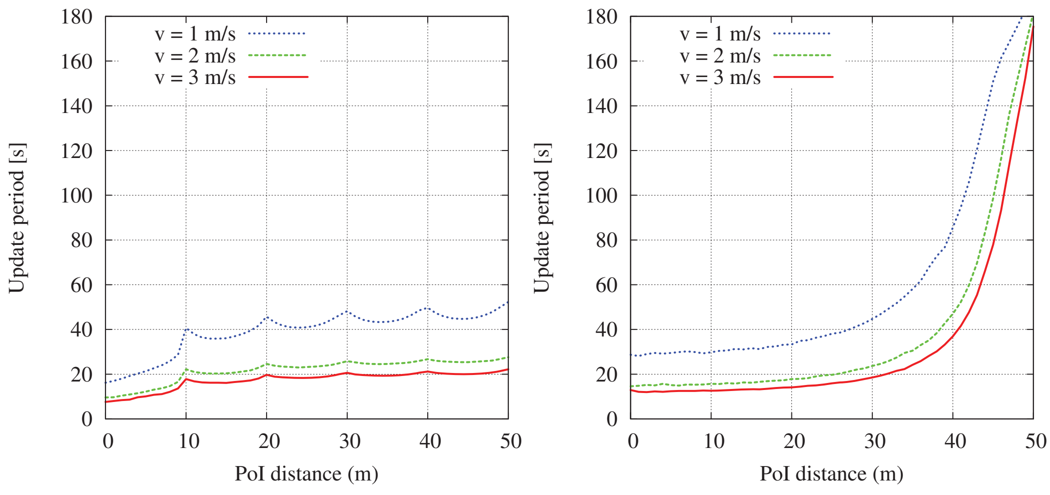
Another way to evaluate the connectivity of the sensors with the sink, and thus the connectivity of the complete network, is to verify the fraction of lost messages due to the expiration of the message lifetime timer T_{data}. In this set of simulations, the message loss with relation to the PoI distance from the sink is observed (Figure 4.40(a)) and the overall message loss during the simulation time (Figure 4.40(b)) for different values of $T_{data} \in \{20, 25, 30, 35, 40\}$ is analyzed. Figure 4.40(a) shows that fourth and fifth level sensors are critical for all the values of T_{data}, where the message loss reaches 17% for L_4 and 45% for L_5. The maximal movement velocity is assumed to be $v_{max} = 3m/s$.

The relation between the overall message loss and T_{data} is presented in Figure 4.40(b), from which it is possible to deduce the minimal value of T_{data} needed to obtain the message delivery to the sink node with certain message loss threshold. Furthermore, expressions for overall average l_{avg} and maximal message loss l_{max} are provided, and these expressions can be used for message loss prediction for a given T_{data} (Equation 4.13).

$$l_{avg} = 906.679e^{-(T_{data}/5.1541)} \quad l_{max} = 608.074e^{-(T_{data}/6.4732)} \tag{4.13}$$

4.3.3.3 PoI coverage

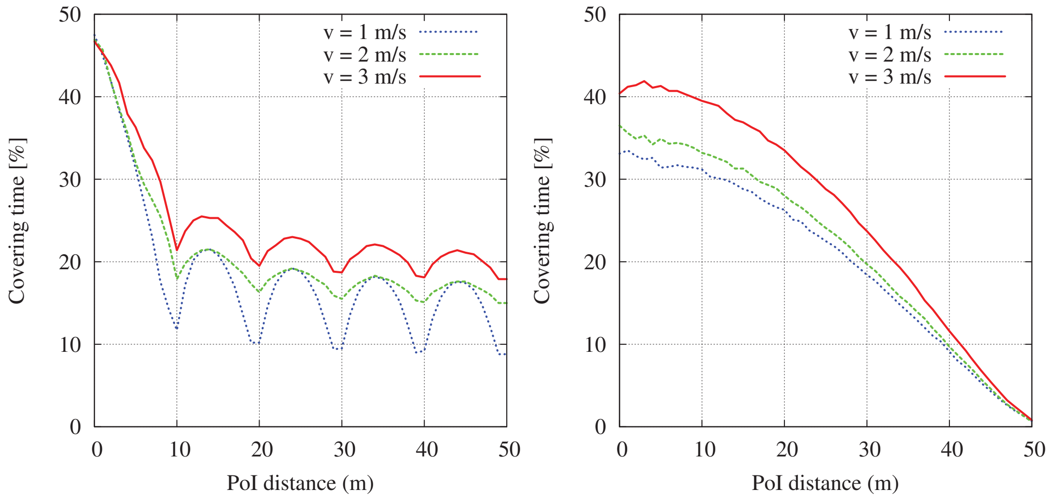
The PoI coverage is evaluated by analyzing the PoI update periods and the overall coverage time (depending on the PoI distance from the sink). The update period is the time interval between two consecutive detections of the same PoI. Figure 4.41 shows the comparison of PoI update period with relation to PoI distance from the sink, for both random walk and circular scenario. It is worth noting that circular approach achieves short update periods. Thus, the frequency of PoI visits and reactivity of the whole network increase. It is also worth noting here that PoIs included into consideration are only discovered PoIs (that means all the PoIs for circular approach, and only a fraction of all the PoIs for the random walk approach).



(a) Circular path approach (b) Random walk approach

Figure 4.41: PoI update time with relation to different movement velocities.

Local maximums in Figure 4.41(a) are the update periods for the PoIs located on the edge of sensors' sensing range. Lower PoI update periods, in the same figure, come from PoIs located next to the center of the path described by the sensors. The update time for the random walk approach (Figure 4.41(b)) increases with the PoI distance from the sink node. Low update periods for PoIs located closed to the sink are the result of the sensors that move randomly and thus statistically more often traverse the central area of the region of interest.

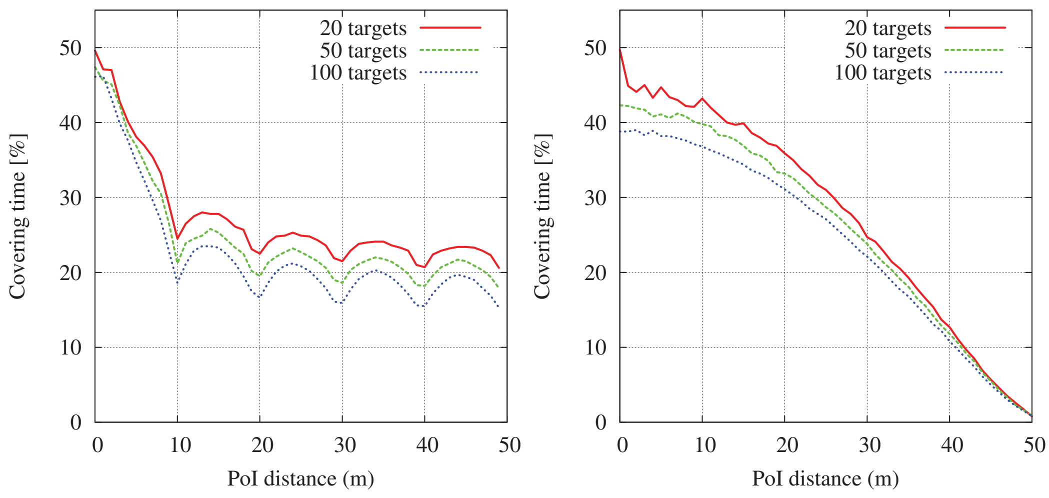


(a) Circular path approach (b) Random walk approach

Figure 4.42: PoI coverage with relation to different movement velocities.

Figure 4.42 presents the PoI coverage evaluation, the percent of time spent while covering the PoI with relation to the PoI distance from the sink for three sensor movement velocities, $v_{max} \in \{1, 2, 3\}$. Due to the T_{sens} condition for covering the PoIs on the edge of the path, other PoIs are covered longer since the covering sensors covers them at the same time as the PoI on the edge. This case results in local maximums shown in Figure 4.42(a). Coverage time for all the PoIs is the highest for the highest movement speed, $v_{max} = 3$m/s. Random walk approach coverage decreases with the PoI distance from the sink, as opposing to the update period graphs. It is worth noting that the fraction of discovered PoIs during the simulation time in the random walk approach is lower than that in the circular path approach.

The percentage of time covered also depends on the number of PoIs in the field of interest. This relation is shown in Figure 4.43(a) for circular paths and Figure 4.43(b) for random walk approach. Figures show that the percentage of time spent in PoI coverage decreases as the number of PoIs increases due to the number of PoIs that need to be processed during the movement. A higher PoI density requires slower sensor movement and thus produces an output



(a) Circular path approach (b) Random walk approach

Figure 4.43: Covering time with relation to different number of PoIs.

similar to the movement with a lower velocity (shown in Figure 4.42).

4.3.4 Conclusion

This section described a novel approach to integrate PoI discovery, multiple PoI coverage and data report to the sink. The motivation for this work is the application of flying mobile sensors for the environmental monitoring, where there is a need to gather as much information as possible while covering the events that occur in the field of interest. By constantly moving, sensors execute the environment discovery task, and by adjusting the movement velocity, they satisfy the constraints regarding the PoI coverage and T_{data} connectivity with the data sink in order to report the PoI data. The effectiveness of the proposed approach is analyzed analytically, while extensive simulation results prove the feasibility of the proposed concept. Future work on this topic will be based on different mobile sensor deployment algorithms and topologies that can be deduced from the proposed one. Furthermore, more realistic network models will be included. Finally, the proposed approach will be implemented on real mobile sensors based on the middleware architecture described in Chapter 3, where the movement direction does not depend on the set of neighboring sensors but only on the path shape and where the neighborhood discovery block sets the movement velocity in the motor driver through the connectivity block.

Chapter 5
General conclusion

Contents

5.1 Summary of contribution 93
5.2 Perspectives 96

5.1 Summary of contribution

The contribution of the work presented in this thesis is twofold. First, it is the conceptual proposal of middleware application dedicated for the use with mobile robots in the context of mobile wireless sensor networks. More precisely, it is intended to be implemented on Wifibot mobile robots, and therefore, some of the blocks that the general architecture comprises are specific to Wifibot mobile robots. Second, these are the algorithms for the deployment and self-deployment of mobile sensors in the context of mobile wireless sensor network applications.

5.1.1 Middleware for networked robots in WSN

As it is already stated, the first contribution of this thesis is the conceptual middleware architecture for networked robots that is implemented on Wifibots. The proposed middleware allows the user to easily implement different types of deployment algorithms for various applications of WSN. It includes the central base-station application that allows a user to gather all the information sensed by the fleet of robots, as well as to send commands to a group or an individual robots, thus introducing the manual control in the robotic network if necessary. Although it is dedicated to be used with Wifibot mobile robots, it can be easily adjusted in order to be used with other robotic platforms.

Generally speaking, the middleware embedded in networked robots achieves several tasks. Firstly, it interacts with the robot firmware in order to drive the wheel motors and collect the information regarding the sensor output and the battery state. Secondly, it manages the communication with other robots and the base-station in the network. Thirdly, it processes both sensed information about the environment and the messages received from the neighbors in the network. Finally, it reacts in a fast and reliable way to the events in the environment. Although many robotic networks assume the existence and reliability of the global network infrastructure that can be used for the communication and information collection among the robots, this scenario cannot hold in the WSN applications. Therefore, the robotic network relies on ad-hoc network infrastructure that allows any-to-any communication paradigm among robots.

Mobile WSN applications implemented with the use of this robot middleware show its feasibility, flexibility and ease of use.

5.1.2 Mobile sensor deployment applications

The advances in mobile robotics allow us today to add the mobility concept into many different classes of wireless sensor and/or actuator networks. The deployment of mobile sensors is possible and useful in many application scenarios, ranging from the environmental monitoring, public safety applications, to the industry and military applications. Manual sensor deployment represents a rather difficult task to achieve in such type of applications due to dangerous or inaccessible deployment environment. The use of mobile robots equipped with sensory capabilities, allows us to overcome these difficulties by deploying the sensor network in a random manner and applying the self-repositioning of self-deploying techniques. Due to the unpredictability of the deployment environment, followed by the uncertainties regarding the robot robustness, the deployment algorithm fulfills basic requirements. It is localized, distributed, efficient and self-reconfigurable.

Second contribution of this thesis is the set of solutions for three applications: the problem of improving the quality of service with the use of mobile robotic networks, coverage of the point of interest with mobile robots, and point of interest discovery and coverage with the use of mobile robots. These mobile robot deployment algorithms proved their feasibility through numerous simulation campaigns as well as in the practical implementation relying on the proposed middleware.

5.1.2.1 Improving the QoS with mobile sensors

First application that is analyzed is the approach to controlling the mobility of wireless mobile robots in order to improve the performances of multimedia video transmission. The proposed robot movement algorithm is based on local, but intrusive, measurements and uses the delays and percentage of packet losses of pings sent to previous and next node on the path between the video source and destination. The effectiveness of the proposed movement strategy is evaluated on a real multimedia testbed.

Obtained results show that it is possible to use a simple algorithm for autonomous mobile robots in order to reduce the video transmission loss by changing the node positions in a multi-hop transmission link. Despite the use of ping delay and loss data as the input to the movement algorithm, it still proves to be feasible. However, these experimental scenarios imply the use of certain network metrics obtained by intrusive methods. Furthermore, concerning the repositioning algorithm execution time, the use of different network metrics (such as received signal strength, signal to noise ratio, throughput, etc.) is suggested.

5.1.2.2 PoI coverage with mobile sensors

Second application of mobile WSN that is analyzed is the PoI coverage with mobile wireless sensors. Mobile robots have the objective to cover the PoI while maintaining the connectivity with a fixed base station.

The proposed deployment algorithm that achieves PoI coverage is distributed, needs only local information at each sensor, does not require synchronization and is divided into direction computation, connectivity maintenance and movement. The connectivity preservation all along the deployment is achieved by using the properties of the Relative Neighborhood Graph. The performance of the algorithm is evaluated regarding the number of sensors that cover the PoI, deployment speed and energy consumption. Formal proofs for connectivity preservation, algorithm termination and the shape of the resulting graph are provided as well. Furthermore, the algorithm is implemented using middleware architecture for Wifibots.

5.1.2.3 PoI discovery and coverage with mobile sensors

Third application of mobile sensors in the context of WSN is a novel approach that integrates PoI discovery, multiple PoI coverage and data report to the data sink. The motivation for this work is the application of flying mobile sensors for the environmental monitoring, where there is a need to gather as much information as possible while covering the events that occur in the field of interest. By constantly moving, sensors execute the environment discovery task, and by adjusting the movement velocity, they satisfy the constraints regarding the PoI coverage and connectivity with the data sink in order to report the information about the PoI. The effectiveness of the proposed approach is analyzed analytically, while extensive simulation results prove the feasibility of the proposed concept.

5.2 Perspectives

Perspectives of the mobile robot middleware and application development in the context of wireless sensor networks are mainly focused on issues of deployment algorithm implementation, the problems of scalability and heterogeneity, as well as the concept of network robustness.

5.2.1 Short-term

Due to the time limitations of the work on this thesis, not all of the proposed concepts could be implemented on real hardware, and that should be the first step in the work to come. On the other hand, as already pointed out, one of the biggest issues in the domain of robotics in general is the problem of precise *localization*. In order to guarantee the precision and accuracy of the network deployment, especially in the case of deployment algorithms based on geometry, it is necessary to provide the robots themselves and the neighboring set of robots with the accurate localization information. Proposed middleware architecture should include more precise localization techniques that could minimize deployment failures induced by the imprecise localization technique. One of the possible localization techniques is the use of Wifibot camera in order to achieve precise landmark-based localization for indoor applications.

Practical implementations of a robotic network demand the use of different types of robots in the network. Depending on the specific application, it is indeed necessary to combine different types of robots that can serve different purposes at the same time, all in the context of the same network. Issues of *scalability* and *heterogeneity* have not been thoroughly discussed in this thesis from the point of view of the middleware architecture, although their importance and necessity in practical implementations deserves more attention. Short term perspectives of the middleware architecture development, as well as the work on deployment algorithms, will be focused on providing the solutions that cover the aforementioned problems.

5.2.2 Mid-term

A significant number of mobile WSN applications require a high level of system robustness. The issue of robustness has many different aspects, and one of the most important ones is the problem of energy consumption reflected in the network lifetime. Especially in the field of mobile robotics, the energy consumption caused by the movement represents one of the most important problems that receives a lot of attention.

Mid-term perspectives of the work proposed in this thesis, are to provide an auto-adaptable network of mobile sensors that will be self-sustainable in terms of energy and network lifetime. In other words, it is necessary to provide an automated deployment and/or redeployment algorithm that will take into account the energy state of individual network nodes and take advantage of provided charging stations in order to provide a completely self-sustainable mobile WSN that is still capable of achieving its initial deployment goals.

5.2.3 Long-term

Finally, from the other point of view on network robustness, the problem that gets a lot of attention is the question of system security. The problem of system security in the context of robotic networks applies all aspects of the network - hardware, software and communication. Hardware security issues focus on the problem of robot construction, its sturdiness and ability to cope with physical challenges during the practical application in hostile deployment environments.

From the networking point of view, much more perfidious are the issues arising from the security breaches in the communication part. Indeed, insecure connection can lead to unwanted confidential data leak. Furthermore, an attacker can even insert false commands in the network and thus completely change the behavior of the individual robots and take the global control of the network, making it useless or even counterproductive. Being one of the essential problems in the robotic network, the communication security issues represent the long-term perspectives of multi-robot middleware and robot deployment algorithm development.

Bibliography

[1] I. Akyildiz, T. Melodia, and K. Chowdhury. Wireless multimedia sensor networks: Applications and testbeds. *Proceedings of the IEEE*, 96(10):1588–1605, 2008. 15

[2] X. Bai, S. Kumar, D. Xuan, Z. Yun, and T. H. Lai. Deploying wireless sensors to achieve both coverage and connectivity. In *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 131–142, New York, NY, USA, 2006. ACM. 16, 19

[3] D. Bakken. Middleware. *Encyclopedia of Distributed Computing*, 2001. 11

[4] M. A. Batalin and G. S. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *in Proc. of 6th International Symposium on Distributed Autonomous Robotic Systems*, pages 373–382, 2002. 16, 31

[5] M. A. Batalin and G. S. Sukhatme. Coverage, exploration and deployment by a mobile robot and communication network. In *Proceedings of the International Workshop on Information Processing in Sensor Networks*, pages 376–391, 2003. 19

[6] I. Bekmezci. *Wireless Sensor Networks: A Military Monitoring Application*. VDM Verlag, 2009. 6

[7] S. Biaz and S. Wu. Rate adaptation algorithms for ieee 802.11 networks: A survey and comparison. In *Computers and Communications, 2008. ISCC 2008. IEEE Symposium on*, pages 130–136, 2008. 46

[8] D. Blank, D. Kumar, and B. M. College. Pyro: A python-based versatile programming environment for teaching robotics. *ACM Journal on Educational Resources in Computing*, 3(4):1–15, 2003. 13

[9] J. Borenstein, H. Everett, and L. Feng. Where am i? sensors and methods for mobile robot positioning. *University of Michigan*, 119:120, 1996. 27

[10] O. Burdakov, P. Doherty, K. Holmberg, J. Kvarnström, and P.-M. Olsson. Relay positioning for unmanned aerial vehicle surveillance*. *Int. J. Rob. Res.*, 29(8):1069–1087, July 2010. 3

[11] W. Burgard, M. Moors, and F. Schneider. Collaborative exploration of unknown environments with teams of mobile robots. In *Revised Papers from the International Seminar on Advances in Plan-Based Control of Robotic Agents,*, pages 52–70, London, UK, UK, 2002. Springer-Verlag. 4

[12] J. Carle and D. Simplot-Ryl. Energy-efficient area monitoring for sensor networks. *Computer*, 37(2):40–46, 2004. 16

[13] S. Chellappan, W. Gu, X. Bai, D. Xuan, B. Ma, and K. Zhang. Deploying wireless sensor networks under limited mobility constraints. *Mobile Computing, IEEE Transactions on*, 6(10):1142–1157, Oct. 2007. 16

[14] Q. Chen. *Studies in autonomous ground vehicle control systems: structure and algorithms.* PhD thesis, Columbus, OH, USA, 2007. AAI3335623. 3

[15] W. Cheng, M. Li, K. Liu, Y. Liu, X. Li, and X. Liao. Sweep coverage with mobile sensors. In *IPDPS*, pages 1–9, 2008. 17

[16] S. Cho, S. Ramasubramanian, O. Turkcu, and S. Subramaniam. Throughput and delay analysis of multi-channel wireless infrastructure networks. *Ad Hoc Netw.*, 10(3):373–387, May 2012. 5

[17] H. Choset. Coverage for robotics – a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):113–126, May 2001. 18

[18] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Math.*, 86(1-3):165–177, 1990. 53, 78

[19] T. H. Collett, B. A. MacDonald, and B. P. Gerkey. Player 2.0: toward a practical robot programming framework. *Proceedings of the Australasian Conference on Robotics and Automation (ACRA '05)*, 2005. 13

[20] CORBA. Common object request broker architecture. 12

[21] C. Costanzo, V. Loscrì, and E. Natalizio. Distributed virtual-movement scheme for improving energy efficiency in wireless sensor networks. In *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, MSWiM '09, pages 297–304, New York, NY, USA, 2009. ACM. 18

[22] C. Côté, Y. Brosseau, D. Letourneau, C. Raievsky, and F. Michaud. Robotic software integration using marie. *International Journal of Advanced Robotic Systems*, 3(1):55–60, 2006. 13

[23] L. Cragg, H. Hu, and N. Voelker. Modularity and mobility of distributed control software for networked mobile robots. In *Software Engineering for Experimental Robotics*, volume 30 of *Springer Tracts in Advanced Robotics*, pages 459–484. Springer Berlin Heidelberg, 2007. 12

[24] T. A. Dahlberg, A. Nasipuri, and C. Taylor. Explorebots: a mobile network experimentation testbed. In *Proceedings of the 2005 ACM SIGCOMM workshop on Experimental approaches to wireless network design and analysis*, E-WIND '05, pages 76–81, New York, NY, USA, 2005. ACM. 15

[25] S. Diamond and M. Ceruti. Application of wireless sensor network to military information integration. In *Industrial Informatics, 2007 5th IEEE International Conference on*, volume 1, pages 317–322, 2007. 1

[26] L. Doitsidis, S. Weiss, A. Renzaglia, M. W. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. *Auton. Robots*, 33(1-2):173–188, Aug. 2012. 3

[27] F. Ducatelle, G. A. Di Caro, and L. M. Gambardella. Cooperative self-organization in a heterogeneous swarm robotic system. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, GECCO '10, pages 87–94, New York, NY, USA, 2010. ACM. 19

[28] A. Elkady and T. Sobh. Robotics middleware: A comprehensive literature survey and attribute-based bibliography. *Journal of Robotics*, 2012. 14

[29] R. B. Fierro, A. K. Das, J. R. Spletzer, J. M. Esposito, V. Kumar, J. P. Ostrowski, G. J. Pappas, C. J. Taylor, Y. Hur, R. Alur, I. Lee, G. Z. Grudic, and B. Southall. A framework and architecture for multi-robot coordination. *I. J. Robotic Res.*, 21(10-11):977–998, 2002. 12

[30] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart. Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325 –1339, july 2006. 17

[31] M. Garetto and E. Leonardi. Restricted mobility improves delay-throughput tradeoffs in mobile ad hoc networks. *Information Theory, IEEE Transactions on*, 56(10):5016 –5029, oct. 2010. 18

[32] G. R. Gerhart and B. A. Abbott, editors. *Robotic and Semi-Robotic Ground Vehicle Technology*. Society of Photo-Optical Instrumentation Engineers (SPIE), Bellingham, WA, USA, 1998. 3

[33] A. Ghosh and S. K. Das. Coverage and connectivity issues in wireless sensor networks: A survey. *Pervasive and Mobile Computing*, 4(3):303 – 334, 2008. 18

[34] C. Gifford, R. Webb, J. Bley, D. Leung, M. Calnon, J. Makarewicz, B. Banz, and A. Agah. Low-cost multi-robot exploration and mapping. In *Technologies for Practical Robot Applications, 2008. TePRA 2008. IEEE International Conference on*, pages 74 –79, nov. 2008. 17

[35] J. Gross, J. Klaue, H. Karl, and A. Wolisz. Cross-layer optimization of ofdm transmission systems for mpeg-4 video streaming. *Comput. Commun.*, 27(11):1044–1055, July 2004. 45

[36] S. Guilbalt and A. Pelc. Gathering asynchronous oblivious agents with local vision in regular bipartite graphs. In *Proc. 18th international Colloquium on Structural Information and Communication Complexity (SIROCCO 2011)*, pages 162–173, 2011. 17

[37] V. Gungor, E. Natalizio, P. Pace, and S. Avallone. Challenges and issues in designing architectures and protocols for wireless mesh networks. In E. Hossain and K. Leung, editors, *Wireless Mesh Networks*, pages 1–27. Springer US, 2007. 14

[38] S. Han, H. Lim, and J. Lee. An efficient localization scheme for a differential-driving mobile robot based on rfid system. *Industrial Electronics, IEEE Transactions on*, 54(6):3362–3369, 2007. 27

[39] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*, MobiSys '04, pages 270–283, New York, NY, USA, 2004. ACM. 1

[40] J. L. Hill and D. E. Culler. Mica: A wireless platform for deeply embedded networks. *IEEE Micro*, 22(6):12–24, Nov. 2002. 15

[41] S. Hong, J. Lee, H. Eom, and G. Jeon. The robot software communications architecture (rsca): Embedded middleware for networked service robots. In *Proceedings of the IEEE International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS)*, pages 25–26, 2006. 12

[42] D. Johnson, T. Stack, R. Fish, D. Flickinger, L. Stoller, R. Ricci, and J. Lepreau. Mobile emulab: A robotic wireless and sensor network

testbed. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, 2006. 15

[43] M.-O. Killijian, M. Roy, and G. Séverac. Arum: A cooperative middleware and an experimentation platform for mobile systems. In *WiMob*, pages 442–449, 2010. 12

[44] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, IPSN '07, pages 254–263, New York, NY, USA, 2007. ACM. 1

[45] J. Klaue, B. Rathke, and A. Wolisz. Evalvid – a framework for video transmission and quality evaluation. In P. Kemper and W. Sanders, editors, *Computer Performance Evaluation. Modelling Techniques and Tools*, volume 2794 of *Lecture Notes in Computer Science*, pages 255–272. Springer Berlin Heidelberg, 2003. 44

[46] J. Ko, C. Lu, M. Srivastava, J. Stankovic, A. Terzis, and M. Welsh. Wireless sensor networks for healthcare. *Proceedings of the IEEE*, 98(11):1947–1960, 2010. 1

[47] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, pages 2–10, 0-0 2006. 16

[48] P. Kulkarni, D. Ganesan, P. Shenoy, and Q. Lu. Senseye: a multi-tier camera sensor network. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 229–238, New York, NY, USA, 2005. ACM. 15

[49] S. Kumar, T. H. L., and A. Arora. On k-coverage in a mostly sleeping sensor network. In *MobiCom '04: Proceedings of the 11th annual international conference on Mobile computing and networking*, 2004. 17

[50] T. P. Lambrou and C. G. Panayiotou. Collaborative area monitoring using wireless sensor networks with stationary and mobile nodes. *EURASIP J. Adv. Signal Process*, 2009:7:1–7:16, Jan. 2009. 3

[51] S. H. Lee, S. Lee, H. Song, and H.-S. Lee. Wireless sensor network design for tactical military applications : Remote large-scale environments. In *Military Communications Conference, 2009. MILCOM 2009. IEEE*, pages 1–7, 2009. 1

[52] F. Li, Y. Wang, X.-Y. Li, A. Nusairat, and Y. Wu. Gateway placement for throughput optimization in wireless mesh networks. *Mob. Netw. Appl.*, 13(1-2):198–211, Apr. 2008. 14

[53] X. Li, H. Frey, N. Santoro, and I. Stojmenovic. Strictly localized sensor self-deployment for optimal focused coverage. *Mobile Computing, IEEE Transactions on*, pages 1520 – 1533, dec. 2010. 17

[54] X. Li, A. Nayak, D. Simplot-Ryl, and I. Stojmenovic. *Sensor Placement in Sensor and Actuator Networks*, pages 263–294. Wiley, 2010. 16

[55] D. Massaguer, C. liang Fok, N. Venkatasubramanian, G. catalin Roman, and C. Lu. Exploring sensor networks using mobile agents. In *In Proc. of the 5th Intlernataional Joint Conference on Autonomous agents and Multiagent systems*, 2006. 17

[56] I. Maza, F. Caballero, J. Capitán, J. R. Martínez-De-Dios, and A. Ollero. Experimental results in multi-uav coordination for disaster management and civil security applications. *J. Intell. Robotics Syst.*, 61(1-4):563–585, Jan. 2011. 3

[57] P. Medagliani, J. Leguay, G. Ferrari, V. Gay, and M. Lopez-Ramos. Energy-efficient mobile target detection in wireless sensor networks with random node deployment and partial coverage. *Pervasive Mob. Comput.*, 8(3):429–447, June 2012. 8

[58] Y. Mei, Y.-H. Lu, C. Lee, and Y. Hu. Energy-efficient mobile robot exploration. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 505 –511, may 2006. 17

[59] N. Mohamed and J. Al-Jaroodi. Characteristics of middleware for networked collaborative robots. In *Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on*, pages 524–531, 2008. 13

[60] D. Moltchanov. Survey: Performance models for wireless channels. *Comput. Sci. Rev.*, 4(3):153–184, Aug. 2010. 5

[61] M. Mouad, L. Adouane, P. Schmitt, D. Khadraoui, and P. Martinet. Control Architecture for Cooperative Mobile Robots using Multi-Agent based Coordination Approach. In *6th National Conference on Control Architectures of Robots*, page 15 p., Grenoble, France, 2011. INRIA Grenoble Rhône-Alpes. 12

[62] R. Murphy, J. Kravitz, S. Stover, and R. Shoureshi. Mobile robots in mine rescue and recovery. *Robotics Automation Magazine, IEEE*, 16(2):91–103, 2009. 4

[63] T. Nadeem and S. Parthasarathy. Mobility control for throughput maximization in ad hoc networks: Research articles. *Wirel. Commun. Mob. Comput.*, 6:951–967, November 2006. 18

[64] E. Natalizio, P. Pace, F. Guerriero, and A. Violi. A reactive and dependable transport protocol for wireless mesh networks. *J. Parallel Distrib. Comput.*, 70(5):431–442, May 2010. 15

[65] I. A. D. Nesnas, R. Simmons, and D. Gaines. Claraty: Challenges and steps toward reusable robotic software. *International Journal of Advanced Robotic Systems*, 3(1):23–30, 2006. 13

[66] A. A. Nimbalkar and D. Pompili. Reliability in underwater inter-vehicle communications. In *Proceedings of the third ACM international workshop on Underwater Networks*, WuWNeT '08, pages 19–26, New York, NY, USA, 2008. ACM. 3

[67] K. Ohara, T. Suzuki, B. K. N. Ando, K. Ohba, and K. Tanie. Distributed control of robot functions using rt middleware. *Proceedings of the International Joint Conference (SICE-ICASE '06)*, pages 2629–2632, 2006. 13

[68] L. E. Parker. On the design of behavior-based multi-robot teams. *Advanced Robotics*, 10(6):547–578, 1995. 12

[69] S. Poduri, S. Pattem, B. Krishnamachari, and G. S. Sukhatme. Using local geometry for tunable topology control in sensor networks. *IEEE Transactions on Mobile Computing*, 8(2):218–230, 2009. 17

[70] D. Pompili, T. Melodia, and I. F. Akyildiz. Deployment analysis in underwater acoustic wireless sensor networks. In *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*, pages 48–55, New York, NY, USA, 2006. ACM. 16

[71] H.-T. Roh and J.-W. Lee. Joint relay node placement and node scheduling in wireless networks with a relay node with controllable mobility. *Wirel. Commun. Mob. Comput.*, 12(8):699–712, June 2012. 15

[72] ROS. Robot operating system, www.ros.org. 13

[73] A. Sanfeliu, N. Hagita, and A. Saffiotti. Network robot systems. *Robotics and Autonomous Systems*, 56(10):793–797, 2008. 13

[74] C. Shen, W. Cheng, X. Liao, and S. Peng. Barrier coverage with mobile sensors. In *ISPAN '08: Proceedings of the The International Symposium on Parallel Architectures, Algorithms, and Networks*, pages 99–104, Washington, DC, USA, 2008. IEEE Computer Society. 17

[75] J.-P. Sheu, K.-Y. Hsieh, and P.-W. Cheng. Design and implementation of mobile robot for nodes replacement in wireless sensor networks. *Journal of Information Science and Engineering*, 2008. 4

[76] D. Simplot-Ryl, I. Stojmenovic, and J. Wu. Energy-efficient backbone construction, broadcasting, and area coverage in sensor networks. *Handbook of Sensor Networks*, pages 43–380343–380, 2005. 16

[77] P. Soetens. Rtt: Real-time toolkit, 2010. 13

[78] A. Somov, A. Baranov, A. Savkin, M. Ivanov, L. Calliari, R. Passerone, E. Karpov, and A. Suchkov. Energy-aware gas sensing using wireless sensor networks. In G. Picco and W. Heinzelman, editors, *Wireless Sensor Networks*, volume 7158 of *Lecture Notes in Computer Science*, pages 245–260. Springer Berlin Heidelberg, 2012. 1

[79] W.-Z. Song, R. Huang, M. Xu, B. Shirazi, and R. LaHusen. Design and deployment of sensor network for real-time high-fidelity volcano monitoring. *IEEE Trans. Parallel Distrib. Syst.*, 21(11):1658–1674, Nov. 2010. 8

[80] A. Srinivas and E. Modiano. Joint node placement and assignment for throughput optimization in mobile backbone networks. *Selected Areas in Communications, IEEE Journal on*, 30(5):975–985, 2012. 15

[81] A. Srinivas, G. Zussman, and E. Modiano. Construction and maintenance of wireless mobile backbone networks. *IEEE/ACM Trans. Netw.*, 17(1):239–252, Feb. 2009. 14

[82] C. Stachniss, O. Martínez Mozos, and W. Burgard. Efficient exploration of unknown indoor environments using a team of mobile robots. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):205–227, Apr. 2008. 4

[83] Z. Tafa and V. Milutinovic. Detectability of static and moving targets in randomly deployed military surveillance networks. *Ad hoc and Sensor Wireless Networks*, 13(3-4):291–312, 2011. 17

[84] G. Tan, S. A. Jarvis, and A.-M. Kermarrec. Connectivity-guaranteed and obstacle-adaptive deployment schemes for mobile sensor networks. *IEEE Transactions on Mobile Computing*, 8:836–848, 2009. 17

[85] A. Tiwari, F. L. Lewis, and S. S. Ge. Wireless sensor networks for machine condition based monitoring. In *In Proeedings of the. Int. Conf. Control, Automation, Robotics, and Vision*, pages 461–467. Lewis, 2004. 1

[86] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261 – 268, 1980. 53, 58

[87] V. Tsaoussidis and S. Wei. Qos management at the transport layer. In *Information Technology: Coding and Computing, 2000. Proceedings. International Conference on*, pages 312–317, 2000. 14

[88] H. Utz, S. Sablatnog, S. Enderle, and G. Kraetzschmar. Miro – middleware for mobile robot applications. *IEEE Transactions on Robotics and Automation*, 18(4):493–497, 2002. 12

[89] B. Wang, H. B. Lim, and D. Ma. A survey of movement strategies for improving network coverage in wireless sensor networks. *Computer Communications*, 32(13-14):1427 – 1436, 2009. 16, 18, 77

[90] Q. Wang, X. Wang, and X. Lin. Mobility increases the connectivity of k-hop clustered wireless networks. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, pages 121–132, New York, NY, USA, 2009. ACM. 18

[91] Y.-C. Wang and C.-C. Hu. Efficient placement and dispatch of sensors in a wireless sensor network. *IEEE Transactions on Mobile Computing*, 7(2):262–274, 2008. 16

[92] Z. Wang. Formation control in mobile actuator/sensor networks. In *Proceedings of SPIE*, pages 706–717. Spie, 2005. 17

[93] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh. Monitoring volcanic eruptions with a wireless sensor network. In *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, pages 108–120, 2005. 1

[94] Wifibot. Mobile robots, www.wifibot.com. xi, 22, 27

[95] M. Winkler, M. Street, K.-D. Tuchs, and K. Wrona. Wireless sensor networks for military purposes. In *Autonomous Sensor Networks*, volume 13 of *Springer Series on Chemical Sensors and Biosensors*, pages 365–394. Springer Berlin Heidelberg, 2013. 1

[96] P. Winter. Steiner problem in networks: a survey. *Networking*, 17(2):129–167, 1987. 67

[97] WSNet. An event-driven simulator for large scale wireless sensor networks, wsnet.gforge.inria.fr. 60

[98] M. Xi, Y. Qi, K. Wu, J. Zhao, and M. Li. Using potential to guide mobile nodes in wireless sensor networks. *Ad hoc and Sensor Wireless Networks*, 12(3-4):229–251, 2011. 17

[99] G. Xing and V. B. Misic. Wireless ad hoc and sensor networks. *International Journal of Parallel, Emergent and Distributed Systems*, 25(6):437–438, 2010. 16

[100] M. Yarvis, A. Kushalnagar, H. Singh, Y. Liu, and S. Singh. Exploiting heterogeneity in sensor networks. In *in Proc. of the IEEE Infocom*, 2005. 16

[101] S. Yoon and C. Qiao. Cooperative search and survey using autonomous underwater vehicles (auvs). *IEEE Trans. Parallel Distrib. Syst.*, 22(3):364–379, Mar. 2011. 3

[102] M. Younis and K. Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Networks*, 6(4):621 – 655, 2008. 8, 16, 18

[103] L. Yu, N. Wang, and X. Meng. Real-time forest fire detection with wireless sensor networks. In *Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005 International Conference on*, volume 2, pages 1214–1217, 2005. 1

[104] J. Yuh. Design and control of autonomous underwater robots: A survey. *Auton. Robots*, 8(1):7–24, Jan. 2000. 3

[105] P. Zhou, X. Wang, B. Manoj, and R. Rao. On optimizing gateway placement for throughput in wireless mesh networks. *EURASIP Journal on Wireless Communications and Networking*, 2010(1):368423, 2010. 14

List of Publications

[1] M. Erdelj, V. Loscri, E. Natalizio, and T. Razafindralambo. Multiple Point of Interest Discovery and Coverage with Mobile Wireless Sensors. *Ad Hoc Networks*, 2013.

[2] M. Erdelj, T. Razafindralambo, and D. Simplot-Ryl. Covering points of interest with mobile sensors. *IEEE Transactions on Parallel and Distributed Systems*, 2013.

[3] M. Erdelj and T. Razafindralambo. Design and implementation of architecture for multi-robot cooperation in the context of WSN. *PE-WASUN*, 2013.

[4] M. Erdelj, N. Mitton, and E. Natalizio. Applications of Industrial Wireless Sensor Networks. *Industrial Wireless Sensor Networks: Applications, Protocols, and Standards*. CRC Press, 2013.

[5] M. Erdelj and T. Razafindralambo. Multiple Target Discovery and Coverage with Mobile Wireless Sensors. In *14èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel)*, 2012.

[6] M. Erdelj, E. Natalizio, and T. Razafindralambo. Multiple Point of Interest Discovery and Coverage with Mobile Wireless Sensors. In *Workshop on Mobility and Communication for Cooperation and Coordination (MC3) at International Conference on Computing, Networking and Communications (ICNC)*, 2012.

[7] M. Erdelj, T. Razafindralambo, and D. Simplot-Ryl. Points of Interest Coverage with Connectivity Constraints using Wireless Mobile Sensors. In *Networking*, 2011.