DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y AUTOMÁTICA
ESCUELA SUPERIOR DE INGENIERÍA
UNIVERSIDAD DE SEVILLA

Range-only SLAM schemes exploiting robot-sensor network cooperation

por

Arturo Eugenio Torres González

Ingeniero de Telecomunición

PROPUESTA DE TESIS DOCTORAL
PARA LA OBTENCIÓN DEL TÍTULO DE
DOCTOR POR LA UNIVERSIDAD DE SEVILLA
SEVILLA, 2016

Directores
Dr.-Ing. José Ramiro Martínez de Dios, Profesor Titular
Dr.-Ing. Aníbal Ollero Baturone, Catedrático

UNIVERSIDAD DE SEVILLA

Memoria para optar al grado de Doctor por la Universidad de Sevilla

Autor:	**Arturo Eugenio Torres González**
Título:	**Range-only SLAM schemes exploiting robot-sensor network cooperation**
Departamento:	**Departamento de Ingeniería de Sistemas y Automática**

V° B° Director:

José Ramiro Martínez de Dios

V° B° Director:

Aníbal Ollero Baturone

El autor:

Arturo Eugenio Torres González

If it is only after
that we understand what has come before,
then we understand nothing

Abstract

Simultaneous localization and mapping (SLAM) is a key problem in robotics. A robot with no previous knowledge of the environment builds a map of this environment and localizes itself in that map. Range-only SLAM is a particularization of the SLAM problem which only uses the information provided by range sensors.

This PhD Thesis describes the design, integration, evaluation and validation of a set of schemes for accurate and efficient range-only simultaneous localization and mapping exploiting the cooperation between robots and sensor networks.

This PhD Thesis proposes a general architecture for range-only simultaneous localization and mapping (RO-SLAM) with cooperation between robots and sensor networks. The adopted architecture has two main characteristics. First, it exploits the sensing, computational and communication capabilities of sensor network nodes. Both, the robot and the beacons actively participate in the execution of the RO-SLAM filter. Second, it integrates not only robot-beacon measurements but also range measurements between two different beacons, the so-called inter-beacon measurements. Most reported RO-SLAM methods are executed in a centralized manner in the robot. In these methods all tasks in RO-SLAM are executed in the robot, including measurement gathering, integration of measurements in RO-SLAM and the Prediction stage. These fully centralized RO-SLAM methods require high computational burden in the robot and have very poor scalability. This PhD Thesis proposes three different schemes that works under the aforementioned architecture. These schemes exploit the advantages of cooperation between robots and sensor networks and intend to minimize the drawbacks of this cooperation.

The first scheme proposed in this PhD Thesis is a RO-SLAM scheme with dynamically configurable measurement gathering. Integrating inter-beacon measurements in RO-SLAM significantly improves map estimation but involves high consumption of resources, such as the energy required to gather and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements. The objective of this scheme is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a centralized mechanism running in the robot that adapts measurement gathering.

The second scheme of this PhD Thesis consists in a distributed RO-SLAM scheme based on the Sparse Extended Information Filter (SEIF). This scheme reduces the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate its measurements in SLAM. Moreover, it inherits the scalability of the SEIF.

The third scheme of this PhD Thesis is a resource-constrained RO-SLAM scheme based on the distributed SEIF previously presented. This scheme includes the two mechanisms developed in the previous contributions –measurement gathering control and distribution of RO-SLAM Update stage between beacons– in order to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements. This scheme exploits robot-beacon cooperation to improve SLAM accuracy and efficiency while meeting a given resource consumption bound. The resource consumption bound is expressed in terms of the maximum number of measurements that can be integrated in SLAM per iteration. The sensing channel capacity used, the beacon energy consumed or the computational capacity employed, among others, are proportional to the number of measurements that are gathered and integrated in SLAM.

The performance of the proposed schemes have been analyzed and compared with each other and with existing works. The proposed schemes are validated in real experiments with aerial robots.

This PhD Thesis proves that the cooperation between robots and sensor networks provides many advantages to solve the RO-SLAM problem. Resource consumption is an important constraint in sensor networks. The proposed architecture allows the exploitation of the cooperation advantages. On the other hand, the proposed schemes give solutions to the resource limitation without degrading performance.

Acronyms

AMCL Adaptive Monte-Carlo Localization

BAN Body Area Network

COTS Commercial Off-The-Shelf

DMP Direct Measurement Probability

EIF Extended Information Filter

EKF Extended Kalman Filter

GPS Global Positioning System

IMP Inter-beacon Measurement Probability

IMU Inertial Measurement Unit

LAN Local Area Network

LNB List of Neighbor Beacons

LQI Link Quality Indicator

MAV Micro Aerial Vehicle

NH Number of Hops

OS Operating System

PF Particle Filter

POMDP	Partially Observable Markov Decision Process
PRR	Packet Reception Rate
RBF	Recursive Bayesian Filter
RBPF	Rao-Blackwellized Particle Filter
RC	Radio Controlled
ROS	Robot Operating System
RO-SLAM	Range-Only Simultaneous Localization and Mapping
RSSI	Received Signal Strength Indicator
SEIF	Sparse Extended Information Filter
SLAM	Simultaneous Localization and Mapping
SMD	Surface Mounted Device
SN	Sensor Network
TDOA	Time Difference of Arrival
TOA	Time of Arrival
TOF	Time of Flight
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
USAR	Urban Search And Rescue
WSN	Wireless Sensor Network

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Simultaneous Localization And Mapping (SLAM) is a key problem in robotics (Durrant-Whyte and Bailey, 2006). In SLAM, a robot builds a map of an unknown environment and localizes itself in that map. The robot has no previous knowledge of the map. The solutions of this problem will allow to have truly autonomous robots. Reliable solutions to the SLAM problem have been applied in successful robot deployments in many application domains, particularly when an external location reference such as a Global Positioning System (GPS) is not available. The possible applications include urban search and rescue, environment reconstruction, underwater surveillance, underground mining and planetary exploration. Most SLAM algorithms do not intend to perfectly estimate the map and the robot localization but to be operational. Published approaches are employed in self-driving cars, unmanned aerial vehicles, autonomous underwater vehicles, planetary rovers, newly emerging domestic robots and even inside the human body.

Different solutions have been proposed in the last years. One key difference between them is the type of sensors used. These sensors can be single or stereo cameras (Murillo et al., 2006), laser range finders (Guivant et al., 2000), RGB-D cameras (Engelhard et al., 2011), ultrasound range sensors (Chong and Kleeman, 1999) or radio-based range sensors (Kurth et al., 2003), among others. Most SLAM solutions rely on laser range finders and cameras (single or stereo) because they provide information about the physical environment which could be used for navigation and obstacle avoidance. However, the recent development of ubiquitous computing systems and technologies such as sensor networks has attracted some attention to solve the SLAM problem with the help of networked systems.

We live in a fully connected world. In the last years a variety of technological fields have emerged in the context of ubiquitous computing systems. Ubiquitous computing is the concept where computing may appear anytime and anywhere (Weiser, 1991, 1993). The computer can be in the classical form of a laptop or desktop computer, in more modern and mobile ways such as a smartphone or a tablet, or in everyday objects like a washing machine or a bracelet. Different concepts and research topics fall within the scope of ubiquitous computing, such as distributed computing, context-aware computing, artificial intelligence, sensor networks and one of the most popular nowadays, the Internet of Things (IoT) (Atzori et al., 2010; Zanella et al., 2014). Our research is focused on the cooperation between autonomous robots and sensor networks.

Sensor networks consist of a high number of low-cost nodes equipped with sensing, actuating, computing and communication capabilities that organize autonomously into networks to achieve a global mission (Akyildiz et al., 2002). Sensor networks are usually used as spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, among others, and to cooperatively pass their data through the network to a main location (García-Hernández et al., 2007). These autonomous sensors are also known as nodes.

Robots and sensor networks have enabled great potentials and a large space for ubiquitous applications. Robotics and sensor networks have usually been considered as separate research fields and little work has investigated the union between these two domains. However, they share several features, enable common cyberphysical applications and provide complementary support to each other (Marron et al., 2011; Kantor et al., 2006). For example, by incorporating intelligent and mobile robots, a sensor network can perform various tasks more efficiently and reliably, such as network deployment, data collection, network connectivity, sensing coverage or network maintenance. On the other hand, using sensor networks allow mobile robots to be remotely monitored and controlled, to navigate in unknown spaces, and to localize themselves. Moreover, mobile robots can help the sensor network to estimate the location of its nodes if unknown.

Sensor network nodes can usually take range measurements to each other, and then to a node attached to a mobile robot. These measurements can be used to localize the robot if the nodes location is known, or to localize the nodes if the robot location is known. In many cases, the locations of the robot and of the sensor network nodes are unknown, or at least not known with enough accuracy. Then, SLAM solutions based on these range measurements are necessary in several scenarios.

The cooperation between robots and sensor networks would then benefit both fields. The sensor network nodes need to know their location in order to perform the tasks they are programmed to do. On the other hand, the robots need to know where they are located in order to be able to navigate in the environment. Thus, they can cooperate so the robot can build a map of sensor network nodes and localize itself in this map.

1.2 Approach

This PhD Thesis deals with schemes for range-only simultaneous localization and mapping (RO-SLAM) in which the map is a geometric representation of the locations of the sensor network nodes deployed in the environment. As these nodes will be considered static, we are going to refer to them also as beacons. We are focused on the use of range measurements directly taken by the robot or taken between two beacons as well as odometry based measurements provided by the robotic platform. The development of dead reckoning, odometry or methods based on inertial measurements are out of our scope.

Scalability, efficiency in the use of resources and robustness are main issues in ubiquitous computing technologies in general and in RO-SLAM in particular. Accomplishing them three is a main objective in the architecture and the schemes developed in this PhD Thesis.

The adopted architecture has two main characteristics. First, it exploits the sensing, computational and communication capabilities of sensor network nodes. The sensor network nodes actively participate in the execution of the RO-SLAM filter. Second, it integrates not only robot-beacon measurements but also range measurements between two different beacons, the so-called inter-beacon measurements. This PhD Thesis describes three RO-SLAM schemes that follow this architecture adopting different approaches to fulfill the accuracy and efficiency objectives.

As it will be proved later, integrating inter-beacon measurements has many advantages in accuracy of the estimations and in the convergence speed of the algorithms. However, taking more measurements involve a higher resource consumption. These resources include the energy required to take and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements, among others. The schemes developed in this PhD Thesis propose different ways of dealing with this increment in resource consumption while still taking advantage of inter-beacon measurements.

The proposed schemes focus on accuracy and resource consumption efficiency, as well as on the cooperation between robots and sensor networks. Many different RO-SLAM methods have been developed in the last years, but very few of them exploit this cooperation. This PhD Thesis intends to provide schemes for improving accuracy and efficiency in the use of resources in RO-SLAM with cooperation between robots and sensor networks.

1.3 Research value

1.3.1 Contributions

- **Contribution1: Design of an architecture for flexible and robust RO-SLAM.** The architecture is modular and flexible in the methods used and in the type of sensors that can be integrated. As it will be shown, RO-SLAM methods can be divided into: the front-end, where sensor measurements are processed and potentially spurious readings are identified; and the back-end, which solves the estimation problem. In this architecture, both, the robot and the beacons, actively participate performing the front-end and the back-end tasks. The architecture makes use of two main properties: integration of inter-beacon measurements and involvement of beacons in the computation of RO-SLAM algorithm. Changing the use of these front-end and back-end parts in both robot and beacons leads to different variations, called "schemes" in this PhD Thesis. This contribution is developed in Chapter 3.

- **Contribution2: Development of a RO-SLAM scheme with dynamically configurable measurement gathering.** Integrating inter-beacon measurements in RO-SLAM significantly improves map estimation but involves high consumption of resources, such as the energy required to gather and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements. The objective of this scheme is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a centralized mechanism running in the robot that adapts measurement gathering. This contribution is also known as **Scheme1** and it is developed in Chapter 4.

- **Contribution3: Development of a distributed SEIF-based RO-SLAM with inter-beacon measurements.** This scheme reduces the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate its measurements in SLAM. The SLAM filter is based on the Sparse Extended Information Filter (SEIF) and inherits its efficiency and scalability. Its distributed approach shares resource consumption, reducing robot CPU burden, and at the same time naturally integrates inter-beacon measurements, which improves map and robot localization accuracies. This contribution is also known as **Scheme2** and it is developed in Chapter 5.

- **Contribution4: Development of a resource-constrained SEIF-based RO-SLAM with inter-beacon measurements.** This scheme includes the ideas of the two mechanisms developed in the previous contributions –measurement gathering control of **Contribution2** and distribution of RO-SLAM Update stage between beacons of **Contribution3**– in order to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements. This scheme exploits robot-beacon cooperation to improve SLAM accuracy and efficiency while meeting a given resource consumption bound. The resource consumption bound is expressed in terms of the maximum number of measurements that can be integrated in SLAM per iteration. The sensing channel capacity used, the beacon energy consumed or the computational capacity employed, among others, are proportional to the number of measurements that are gathered and integrated in SLAM. This contribution is also known as **Scheme3** and it is developed in Chapter 6.

Figure 1.1 summarizes the contributions of this PhD Thesis.

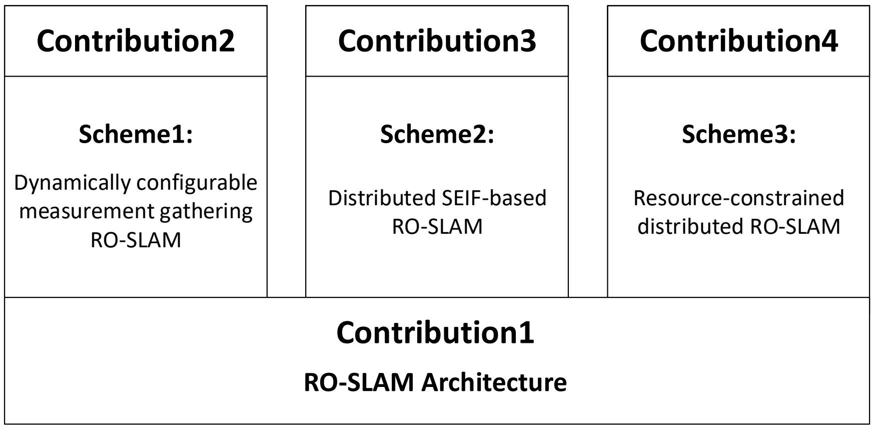

Contribution2
Contribution3
Contribution4
Scheme1:
Dynamically configurable measurement gathering RO-SLAM
Scheme2:
Distributed SEIF-based RO-SLAM
Scheme3:
Resource-constrained distributed RO-SLAM
Contribution1
RO-SLAM Architecture

Figure 1.1: Relation between the contributions and the proposed architecture and schemes of this PhD Thesis.

All the aforementioned contributions have been validated in simulations and experiments. Chapter 7 analyzes the performance and robustness of the proposed schemes and compares them with each other and with other methods in the literature.

1.3.2 Relevant publications

Books

- J.R. Martinez-de Dios, A. de San Bernabé, A. Torres-González, A. Ollero. *Cluster-based Localization and Tracking in Ubiquitous Computing Systems.* Springer, 2016. (In press)

Book chapters

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Exploiting multi-hop inter-beacon measurements in RO-SLAM* in *Cooperative Robots and Sensor Networks 2015*, 101-119, Springer.

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Robot-WSN Cooperation for Scalable Simultaneous Localization and Mapping* in *Cooperative Robots and Sensor Networks 2014*, 25-41, Springer.

Journals

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Robot-Beacon Distributed Range-only SLAM for Resource-Constrained Operation.* Sensors. (Submitted).

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Robot-Sensor Network Distributed SEIF Range-only SLAM.* Autonomous Robots, Springer. (Submitted).

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *An Adaptive Scheme for Robot Localization and Mapping with Dynamically Configurable Inter-Beacon Range Measurements.* Sensors 14 (5), 7684-7710.

- A. Torres-González, J.R. Martinez-de Dios, A. Jiménez-Cano, A. Ollero. *An Efficient Fast-Mapping SLAM Method for UAS Applications Using Only Range Measurements.* Unmanned Systems 4 (02), 155-165, 2016.

- J.R. Martinez-de Dios, K. Lferd, A. de San Bernabé, G. Núñez, A. Torres-González, A. Ollero. *Cooperation Between UAS and Wireless Sensor Networks for Efficient Data Collection in Large Environments.* Journal of Intelligent & Robotic Systems 70 (1-4), 491-508. April, 2013.

International conferences

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Accurate fast-mapping Range-only SLAM for UAS applications.* IEEE Intl. Conf. on Unmanned Aircraft Systems (ICUAS), 2015, 453-550.

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Efficient Robot-Sensor Network Distributed SEIF Range-only SLAM.* IEEE Intl. Conf. on Robotics and Automation, ICRA 2014, 1319-1326.

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Exploiting Multi-hop Inter-beacon Measurements in RO-SLAM.* In the 5th Intl. Conf. on Ambient Systems, Networks and Technologies (ANT-2014). Procedia Computer Science, Volume 32, 2014, 1101-1107

- A. Torres-González, J.R. Martinez-de Dios, A. Ollero. *Integrating Internode Measurements in Sum of Gaussians Range Only SLAM.* In ROBOT2013: First Iberian Robotics Conference (pp. 473-487). Springer International Publishing. January, 2014.

1.4 Context

This PhD Thesis has been developed within the context of a number of R&D projects funded by the European Commission in FP7 and H2020 and by the Spanish Ministry for Science and Innovation. In all of them the localization of an aerial robot in GPS-denied scenarios was an important part. The PhD candidate designed and implemented the developments presented in this dissertation mainly within the Robotics, Vision and Control group at the University of Seville.

- AEROARMS [1] (H2020-ICT-644271). AEROARMS proposes the development of the first aerial robotic system with multiple arms and advanced manipulation capabilities to be applied in industrial inspection and maintenance (I&M). New perception methods, with ability to adapt to changing illumination conditions, are required for accurate local mapping and localization in denied GPS industrial environments and for grabbing and manipulation operations. The schemes developed in this PhD Thesis provide a basis for developing these new perception methods.

- Manipulation system using aerial robots for maintenance in the generation and distribution of energy. Application to wind turbines (AEROMAIN) (DPI2014-59383-C2-1-R). AEROMAIN proposes the development of the worldwide first aerial robotic system with advanced manipulation capabilities to be applied in inspection and maintenance of energy systems and particularly in the maintenance of wind turbines, including contact inspection (i.e. ultrasonic inspection) and blade repairing of surface damages or even “materials lost” of impacted areas (leading edge).

- Estimation and Control for SAFE wireless high MOBILity cooperative industrial systems (EC-SAFEMOBIL)[2] (FP7-ICT-288082). The objectives are to design and develop estimation and control methods for networked robotics in key industrial applications. EC-SAFEMOBIL is devoted to the development of accurate common motion estimation and control methods and technologies in order to reach levels of reliability and safety to facilitate unmanned vehicle deployment in a broad range of applications. It also includes the development of a secure architecture and the middleware to support the implementation.

- Aerial Robot Co-Worker in Plant Servicing (ARCOW)[3]. This is a challenger project in the third challenge (Plant Servicing and Inspection, PSI) of the European Robotic Challenges (EuRoC)[4] (FP7-ICT-608849). EuRoC proposes

[1]`https://aeroarms-project.eu`
[2]`http://www.ec-safemobil-project.eu`
[3]`http://www.euroc-project.eu/index.php?id=grvc-catec`
[4]`http://www.euroc-project.eu`

to launch three industry-relevant challenges. PSI challenge aims at targeting the open problems in existing MAV (Micro Aerial Vehicle) solutions (especially in multicopters) to enable their deployment in real life scenarios. ARCOW project focuses on the implementation and validation of enabling technologies to safely introduce aerial robots collaborating with humans in aircraft manufacturing plants. The main objectives are twofold. The first, to develop an aerial robot able to deliver light goods (small tools, bag of rivets, seals, etc.) to the different working stations while navigating within an aircraft manufacturing plant. The second objective is the development of a low-cost localization system for costly tools or portable machinery into the plant in order to build an improved Foreign Object Debris (FOD) monitoring system. Safety, reliability and human-aware operation will be key factors in the project.

- Cooperating Objects Network of Excellence (CONET)[5] (FP7-ICT-224053). The objective of CONET was to build a strong community in the area of Cooperating Objects including research, public sector and industry partners from the areas of embedded systems, ubiquitous computing and wireless sensor networks. The first steps of this PhD Thesis were tested in the CONET Integrated Testbed, one of the main results of the project.

1.4.1 Other projects

- PLAtform for the deployment and operation of heterogeneous NETworked objects (PLANET)[6] (FP7-ICT-257649). Its objective is to design, develop and validate an integrated planning platform that enables the deployment, operation and maintenance of heterogeneous large-scale systems of networked Unmanned Aerial Systems (UAS), Unmanned Ground Vehicles (UGVs) and cooperating with Wireless Sensors and Actuators Networks (WSAN). Target tracking is a critical component in PLANET. The preliminary tests that lead to the schemes developed in this PhD Thesis were performed in the context of PLANET.

[5]`http://www.cooperating-objects.eu`
[6]`http://www.planet-ict.eu`

- Aerial Robotics Cooperative Assembly System (ARCAS) [7] (FP7-ICT-287617). The ARCAS robotic system involves transportation of parts by means of one and several (joint transportation) flying robots, and the precise placement and assembly of the parts with appropriate manipulation devices to build a structure or to assembly an object. In order to achieve this objective, the flying robots need accurately know the location of themselves and of the parts to be assembled. The real outdoor experiments of this PhD Thesis were made with platform developed for this project, AMUSE.

- Integrated system for identification, localization and monitoring of personnel in working centers (SILCAE) (PI-1339/2014). SILCAE is a technology transfer project which aims to develop an automatic system for identifying, locating and monitoring people in poorly structured environments of intense activity. The system will be used for monitoring activities and staff present in a wide range of scenarios, from building sites and factories, to supermarkets and superstores. The system must be robust to the variabilities of the stage, must require reduced infrastructure and must be flexible to adapt to the needs of each type of scenario.

1.5 Structure of the document

This PhD Thesis describes the design and evaluation of a set schemes for efficient and accurate RO-SLAM. This document is structured as follows.

Chapter 2 describes existing work related to this PhD Thesis. More concretely, it first briefly presents the SLAM problem and the probabilistic tools most widely employed for solving it. Secondly, it introduces sensor networks, summarizes the state of the art of the current technologies in range sensors and reviews localization and tracking methods for sensor networks. Finally, describes the SLAM methods and schemes with range-only sensors, their characteristics and how they deal with the partial observability problem.

[7] `http://www.arcas-project.eu`

Chapter 3 presents the general architecture for RO-SLAM, **Contribution1** of this PhD Thesis. The adopted architecture has two main characteristics. First, it exploits the sensing, computational and communication capabilities of sensor network nodes (beacons). The beacons actively participate in the execution of the RO-SLAM filter. Second, it integrates not only robot-beacon measurements but also range measurements between two different beacons, the so-called inter-beacon measurements. This chapter also analyzes and discusses the implications of integrating these inter-beacon measurements in RO-SLAM.

Chapter 4 presents a RO-SLAM scheme with dynamically adaptive measurement gathering, **Contribution2** of this PhD Thesis. This scheme adopts a mechanism that dynamically modifies the rate and variety of range measurements that are integrated in RO-SLAM in order to reduce the increment in resource consumption caused by the integration of inter-beacon measurements. The scheme is analyzed and evaluated in simulations and real 2D experiments performed in the CONET Integrated Testbed.

Chapter 5 presents a scheme that improves resource consumption efficiency by distributing the computation of RO-SLAM between the robot and the different beacons in the surroundings of the robot, **Contribution3** of this PhD Thesis. It uses a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate them in SLAM. This scheme is based on the Sparse Extended Information Filter (SEIF) (Thrun et al., 2004), and inherits its efficiency and scalability. Its distributed approach shares resource consumption, reducing robot CPU burden, and at the same time naturally integrates inter-beacon measurements, which improves map and robot localization accuracies. The scheme is validated in simulations and real 2D experiments performed in the CONET Integrated Testbed.

Chapter 6 presents a scalable robot-beacon distributed RO-SLAM scheme for resource-constrained operation, **Contribution4** of this PhD Thesis. This scheme efficiently combines a distributed SEIF SLAM method, that integrates robot-beacon and inter-beacon measurements, together with a distributed information-driven tool that selects the measurements to be integrated in SLAM balancing uncertainty improvement and resource consumption. The scheme has a robot-beacon distributed approach where beacons actively participate in measurement selection, gathering and integration in SLAM. This scheme ensures resource-constrained operation with static or dynamic bounds, showing significant flexibility. It achieves higher accuracy and lower beacon initialization times than conventional SLAM methods. Besides, it can be executed in almost constant time regardless of the map size. Its performance was evaluated in 3D SLAM experiments.

Each chapter describing a scheme includes some experiments with mobile ground robots or aerial robots, in order to better explain the scheme or to discuss about specific properties or parameters of the corresponding scheme. **Chapter 7** analyzes the performance of the proposed schemes in sets of 3D SLAM experiments performed with aerial robots. It evaluates the performance of the schemes under the same conditions and compares them with each other and with the well-known EKF SLAM.

Finally, **Chapter 8** summarizes the conclusions of this PhD Thesis and introduces future lines of research.

Chapter 2

Related work

2.1 Introduction

This chapter briefly summarizes existing works, methods and technologies related to those developed and used in this PhD Thesis. The chapter can be divided into three parts. The first one briefly presents the SLAM problem and the probabilistic tools most widely employed for solving it. The second, introduces sensor networks. The third part deals with range-only SLAM, the actual core of this PhD Thesis.

RO-SLAM is a particularization of the general SLAM problem. Simultaneous localization and mapping is the problem in which a robot without any prior information about the environment and its own location, generates a map of this environment and simultaneously localizes in this map. The solutions to this problem allow the development of truly autonomous robots, which could work ideally in any environment. These solutions are based on probabilistic tools such as Bayesian filters.

In RO-SLAM the robot builds the map and localizes employing only range measurements. Thus, RO-SLAM assumes that a number of devices (beacons) endowed with range sensing capabilities have been deployed in the environment at random locations. One of the advantages of RO-SLAM is that the devices used to take measurements usually can do many other things. Besides range sensing, these devices usually have computing and communication capabilities and can self-organize into networks. For instance, these sensor network nodes can be used to gather measurements of the environment, filter them, obtain statistics and transmit them to a Base Station for environmental monitoring. Sensor network technologies have attracted many researchers in the last years and a very wide variety of techniques where the sensor network nodes use range measurements for localization, tracking and mapping have been developed.

Solving the SLAM problem integrating different types of sensors require different techniques. The most common case is the range-bearing SLAM, which employs measurements from rangefinder scanners, stereo cameras or RGB-d cameras. Also, there are bearing-only SLAM and range-only SLAM. The first type of techniques use bearing sensors such as single cameras. The second, range-only SLAM, use range measurements obtained with laser, radio or acoustic signals. In both range-only and bearing-only SLAM, one single measurement is unsuitable to constrain the location of a feature in the map. This is known as the partial observability problem, which will be discussed in the following.

The chapter is structured as follows. First, in Section 2.2, the SLAM problem is analyzed in a general way, presenting its formulations and fundamental properties and briefly describes some of the probabilistic tools which will be used later in this PhD Thesis. Second, Section 2.3 introduces sensor networks, summarizes the state of the art of the current technologies in range sensors, reviews localization and tracking methods for sensor networks and presents the mapping problem in sensor networks and how it has been solved. Finally, Section 2.4 describes the SLAM methods and schemes with range-only sensors.

2.2 Simultaneous localization and mapping

Simultaneous localization and mapping (SLAM) problem asks if it is possible for a mobile robot to be placed at an unknown location in an unknown environment and then to incrementally build a consistent map of this environment while simultaneously determining its location within this map. Reliable solutions to the SLAM problem have been applied in successful robot deployments in many application domains, particularly when an external location reference such as a global positioning system

(GPS) is not available. The possible applications include urban search and rescue, environment reconstruction, underwater surveillance, underground mining and planetary exploration.

The objective of SLAM is to estimate the robot location and the map of the surroundings as the robot moves in an unknown environment. One way to represent the map is using a set of geometric features. In (Dissanayake et al., 2001) it was demonstrated that the Extended Kalman Filter (EKF) based SLAM algorithm can converge as the robot continues to gather information about its environment. Essential convergence properties of the algorithm with direct practical impact were proved in the linear case. These convergence results were extended to nonlinear EKF in (Huang and Dissanayake, 2007), where the nonlinear model was approximated by its linearization using the Taylor expansion. This linearization assumes that the Jacobians are evaluated using the ground truth, which is not available in the SLAM problem. Thus, they are evaluated with the SLAM state, involving estimation errors that perturb the SLAM performance. Furthermore, one of the fundamental limitations of EKF-based SLAM algorithm is the possibility of resulting in overconfident estimates. This overconfidence limitation was observed by a number of researchers (Bailey et al., 2006a) and theoretically proved in (Huang and Dissanayake, 2007). Another fundamental limitation of the EKF-based SLAM algorithm is its quadratic computational complexity, which is associated to the presence of a dense covariance matrix.

The two aforementioned limitations have motivated research in other variations of Bayesian filters for SLAM algorithms such as those based on the Extended Information Filter (EIF) (Thrun et al., 2005; Frese, 2005a; Walter et al., 2007; Wang et al., 2007), more efficient implementations of EKF SLAM methods (Guivant and Nebot, 2001), EKF-based SLAM algorithms with improved consistency (Huang et al., 2010), and local submap joining algorithms (Williams, 2001; Estrada et al., 2005; Huang et al., 2008).

Recent works used nonlinear optimization-based approaches to solve the SLAM problem. In these approaches all the robot poses from where the measurements were gathered are involved in the estimation of the state vector and an utility function based on Maximum Likelihood (ML) is optimized, either using the odometry and the robot to feature observation information (Dellaert and Kaess, 2006), or using the relative pose information obtained from scan/image registration (Lu and Milios, 1997; Ila et al., 2010). These optimization based methods can improve the consistency of SLAM results and can be made efficient by exploiting the sparseness of the information matrix. Nowadays the SLAM problem consisting of a few thousands robot poses and a few million robot-to-feature observations can be solved efficiently using optimization based approaches (Kümmerle et al., 2011).

Quite recently, it has been shown that there are some special underlying nonlinear structures present in point feature-based SLAM (Wang et al., 2013) and pose-graph SLAM (Wang et al., 2012; Carlone et al., 2014). This has resulted in the discovery of some interesting properties and algorithms for SLAM (Carlone et al., 2014; Zhao et al., 2013).

2.2.1 SLAM formulations

There are mainly two SLAM formulations: the so-called feature-based SLAM and the pose-graph SLAM.

Feature-based SLAM problem

In feature-based SLAM the environment is assumed to consist of stationary features. These features can be points, lines or in general anything in the environment perceptible by the sensors. The robot pose (including position and orientation) at time t is denoted by $X_r(t)$, whereas the parameters used to describe features (e.g. the position of a point feature) are denoted by $X_f(t) = X_f$. A process model that relates the robot pose at time $t+1$ with the robot pose at time t is usually described by:

$$X_r(t+1) = f(X_r(t), u(t), v(t)), \tag{2.1}$$

where $u(t)$ is the control action at time t and $v(t)$ is the process noise at time t.

The observation at time t is a function of $X(t)$ and X_f and is given by:

$$z(t) = h(X_r(t), X_f, w(t)), \tag{2.2}$$

where $w(t)$ is the observation noise at time t.

The feature-based SLAM problem aims to obtain an estimate of feature locations X_f and robot pose $X(1), \cdots, X(t)$ using the information gathered from the sensors. In this PhD Thesis we are interested in feature-based SLAM where the nodes of the sensor network act as features (anchors) in the environment map.

Pose-graph SLAM problem

In the pose-graph SLAM formulation the observations of the environment are used to first estimate the relationship between the poses from which the observations are acquired. The relative pose between pose i and pose j is denoted as z_{ij} and the covariance matrix of z_{ij} is given by P_{ij}. The state vector for pose-graph SLAM includes all the robot poses:

$$X(t) = [Xr(1) \ \cdots \ Xr(t)]^T , \tag{2.3}$$

and the aim of SLAM here is to find the state vector which minimizes the weighted sum of the squared error of all the relative pose information, i.e. to minimize:

$$\sum_{ij} [z_{ij} - g(X_r(i), X_r(j))]^T P_{ij}^{-1} [z_{ij} - g(X_r(i), X_r(j))] , \tag{2.4}$$

where $g(X_r(i), X_r(j))$ is the nonlinear function that represents the relative pose between pose $X_r(i)$ and $X_r(j)$.

2.2.2 Fundamental properties of SLAM

This section summarizes the main properties required in a SLAM scheme.

Observability

The first important question to ask is whether the SLAM problem is solvable. That is, if the information available is sufficient to obtain an estimate of the current state $X(t)$. In control theory observability means the ability to infer the system state from a sequence of control actions and observations. In feature-based SLAM the system state consists of two parts. The first part is the state of the map, which model is assumed static since the environment is assumed steady. The second part is the location of the robot, which evolves in time according to the robot kinematic model. As the observations available to SLAM only relate the robot location to the map of the unknown environment and the robot odometry only contains the relative position between the consecutive robot poses, the initial location of the robot cannot be obtained using the observations and robot odometry. Thus, feature-based SLAM problem is not observable from the control theoretic point of view (Andrade-Cetto and Sanfeliu, 2004; Lee et al., 2006).

To address this issue most SLAM techniques and schemes either assume that there is some knowledge about the initial robot location available or, more usually, assign the origin of the coordinate frame of the map to be the initial pose of the robot. The latter is equivalent to assuming perfect knowledge of the initial robot location and is used in most of the practical SLAM implementations. Of course, the map built in the latter case is local.

Observability of SLAM given the robot model

The robot state is observable when the robot model and measurements of the appropriate control inputs are available. This is because the current robot state can be directly computed using the initial robot state and the sequence of control inputs. Of course, the uncertainty of the robot state computed this way will increase over time because of the process and control noises.

When the robot poses at different time steps are all known, SLAM problem becomes a mapping problem. Therefore, the conditions for the observability of the SLAM problem are similar to that of the mapping problem, which depend on the sensor observation model.

Clearly, when a range sensor and a bearing sensor are available to observe the environment, feature-based SLAM is fully observable. However, in case of having bearing-only or range-only sensors, SLAM is partially observable. Moreover, depending on each sensor other observability problems may arise. For instance, the bearing-only case (e.g. when a monocular camera is used) the point feature position may not be observable in some special cases such as with zero parallax. Also, in case of using a range-only sensor, the observability of point feature position depends on the motion of the robot/sensor. For example, if the robot is moving on a straight line, there will always be an ambiguity of the point feature location no matter how many range observations are integrated in SLAM (Blanco et al., 2008a).

Observability of SLAM when the robot model is not available

There are some cases in which the robot model is not available. In these cases the observability problem in SLAM becomes more interesting since the estimate of robot poses $Xr(1), \cdots, Xr(t)$ has to be derived from the observations of the environment. A robot pose becomes observable through the ability of the on-board sensors to capture multiple features in the environments that overlap with observations from one or more other robot poses.

For example, for 2D point feature-based SLAM, observing two common point features is sufficient to obtain an estimate of the relative pose. When the robot is moving in a long corridor, the motion along the corridor cannot be determined through the matching of two laser scans. For monocular SLAM without any robot/camera motion model, image observations of at least five common 3D point features are needed to obtain an estimate of the relative rotation and translation between the two poses. Even then the scale parameter is not observable. Observability issues that arise with monocular cameras have been well addressed in computer vision literature through projective geometry (Hartley and Zisserman, 2003).

Convergence

The convergence problem is that of asking if the uncertainty of the SLAM estimated state converges to a finite value while the time t goes to infinity. In feature-based SLAM thanks to the stationary assumption of the features, the observations from the robot will make the feature location uncertainty to monotonically decrease. This convergence property is proved when EKF is used in SLAM for both the linear case (Dissanayake et al., 2001) and the nonlinear case (Huang and Dissanayake, 2007).

In practice, approaching the lower bounds to the estimation uncertainty requires controlling the robot to move in a specific manner and gather a sufficient number of observations to the same features.

Consistency

Consistency is another important criterion for evaluating a SLAM solution. For a dynamic estimation problem a solution is said to be consistent if the estimate is unbiased and the estimated covariance matrix matches the real mean square error (Bar-Shalom et al., 2004). It has been demonstrated that both, the EKF-based SLAM solution and the SLAM solution based on Particle Filters, can be inconsistent in some scenarios (Huang and Dissanayake, 2007; Bailey et al., 2006b,c). However, it is unlikely that inconsistency can be completely avoided as the SLAM problem is inherently nonlinear. From this point of view, it is important to compare the tendency for inconsistency in different algorithms.

Recently it has become clear that the overconfident estimate for a SLAM algorithm results from by the fact that the estimation process is fed with incorrect information that is originated when the Jacobian matrices of observation/odometry functions with respect to the same feature/pose gets evaluated at different feature/pose location estimates (Huang and Dissanayake, 2007; Huang et al., 2008). This issue can be avoided by using a full nonlinear least squares optimization but its computational complexity may be unacceptable for very large-scale SLAM problems.

2.2.3 Probabilistic tools

This PhD Thesis makes extensive use of probabilistic estimation tools to solve the SLAM problem. In the large related work, Bayesian filtering have been shown as a very useful mathematically-founded framework for estimation under uncertainty. In this section the main tools used in SLAM are summarized very briefly. A more detailed description of the principles of the Bayes filter and the probabilistic filters derived from it can be found in Appendix C.

Kalman Filters

Since its development, Kalman Filters (KFs) (Thrun et al., 2005) have been the subject of extensive research and application. Their success is originated by their simplicity and robustness. KFs have become one of the most common methods used for localization and mapping in ubiquitous computing systems. KF is a parametric Recursive Bayesian Filter (RBF) that implements an optimal estimator that minimizes the covariance of the estimated error. It represents the belief $bel(x_t)$ at time t by the mean μ_t and the covariance Σ_t. All the distributions involved in KF are Gaussian.

The operation of KFs is based on two basic stages: the Prediction stage and the Update stage. In the Prediction stage, an estimation of the system state for the next instant is obtained. On the other hand, the Update stage integrates the new measurements in order to improve the estimation of the state vector.

KF assumes linear state and measurements with additive Gaussian noise, and this is not usually met in many applications. For example, the observation model of range measurements is nonlinear, and the robot's motion model is also nonlinear. The Extended Kalman Filter (EKF) overcomes the assumption of linearity. Here the assumption is that the next state probability and the measurement probabilities are governed by nonlinear functions. The key idea in the EKF is linearization. Given the nonlinear functions f and h, the linearization of the functions is obtained by Taylor expansion.

Information Filters

Information Filters (IFs) are the dual to Kalman Filters. The belief is also represented by a Gaussian. Whereas in the Kalman Filter family of algorithms, Gaussians are represented by their moments, mean and covariance, in Information Filters, Gaussians are represented in their canonical form, comprised of an information vector (ξ) and an information matrix (Ω).

IFs have two main interesting properties that make them more suitable in some cases. First, the Prediction stage in IFs is additive and thus, easily implemented in a parallel manner, enabling distributed implementations where each entity gathers its measurement and integrates its measurement in the IF. The second advantage is that although KFs and IFs have the same computational burden, the greater computational cost is in KFs is in the Update stage while in IFs, the greater computational cost is in the Prediction stage. Hence, IFs is significantly more efficient than KFs in problems with simple Prediction stage and where many measurements are integrated in the Update stage. These and other advantages of IFs will be discussed in detail in Chapter 5.

Particle Filters

Particle Filters (PFs) are nonparametric implementations of the RBFs. The key idea of PFs is to represent the belief $bel(x_t)$ by a set of random state samples. Instead of representing the distribution by a parametric form, PFs represent a distribution by a set of samples drawn from this distribution. Such a representation is approximate, but it is not parametric, and therefore can represent a much broader space of distributions than only Gaussians.

In PFs the samples of a posterior distribution are called *particles* and each one is a hypothesis as to what the true world state may be at a concrete time.

2.3 Sensor networks

The techniques and methods presented in this PhD Thesis exploit the cooperation between robots and sensor network nodes to improve SLAM performance. This section briefly summarizes the issues of sensor network technologies that are maybe more related to the methods and schemes developed in this PhD Thesis. The objective is not to be exhaustive but to give a general description of the most related topics, which can be of interest to readers from the robotics community. More detailed descriptions of algorithms and applications of Sensors Networks and cooperating objects can be found in the literature (Banatre et al., 2008; Marron et al., 2011).

Sensor networks (SNs) consist of a high number of low-cost nodes equipped with sensing, actuating, computing and communication capabilities that organize autonomously into networks to achieve a global mission. SNs are usually used as spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, among others, and to cooperatively pass their data through the network to a main location. They can also be used to gather range measurements to a target carrying a sensor and then localize and track it. These autonomous sensors are also known as nodes. Thus, sensor network nodes are usually endowed with sensing, computing and communication capabilities.

SNs have high potentialities for many applications in scenarios such as military target tracking and surveillance (Simon et al., 2004) and (Yick et al., 2005), natural disaster relief (Castillo-Effer et al., 2004), biomedical health monitoring (Gao et al., 2006) and (Lorincz et al., 2004), and hazardous environment exploration and seismic sensing (Werner-Allen et al., 2006). In military target tracking and surveillance, a SN can assist in intrusion detection and identification. Specific examples include spatially-correlated and coordinated troop and tank movements.

Unlike traditional networks, a SN has its own design and resource consumption constraints, which include a limited amount of energy, short communication range, low bandwidth, and limited processing and storage in each node. Design constraints are usually application dependent and are based on the monitored environment. The environment plays a key role in determining the size of the network, the deployment scheme, and the network topology. The size of the network varies with the monitored environment. For indoor environments, fewer nodes are required to form a network in a limited space whereas outdoor environments may require more nodes to cover a larger area. An ad hoc deployment is preferred over pre-planned deployment when the environment is inaccessible by humans or when the network is composed of hundreds to thousands of nodes. Obstructions in the environment can also limit communication between nodes, which in turn affects the network connectivity (or topology).

In the following the most related issues regarding the techniques presented in this PhD Thesis are summarized. First, in Section 2.3.1 the main types of range sensors are presented. While the SLAM problem has not been much researched within the SN domain, a very high number of localization and tracking and mapping methods have been developed. The main methods are briefly summarized in Section 2.3.2 and Section 2.3.3. Again the objective of these sections is to illustrate the wide range of localization, tracking and mapping techniques developed in the SN domain. The description presented in this section will be complemented with deeper and more focused descriptions in each of the required chapters.

2.3.1 Range sensors

Range sensors can measure the distance between two sensor network nodes. These measurements can be used in tasks such as localization, tracking and mapping. Three main type of ranging technologies are widely used in ubiquitous computing: radio signal metrics, time of arrival and differential time of arrival.

Radio signal and link quality

These measurement relies on the fact that radio signals interact with the environment as they propagate. Different radio signal metrics can be used to estimate range. Among them there are hardware-based estimators, where the rage measurement that can be directly obtained from the radio module, requiring negligible computation overhead, delay or energy consumption. They include the Link Quality Indicator (LQI), the Received Signal Strength Indicator (RSSI), and the Signal-to-Noise Ratio (SNR). On the other hand, software-based estimators, such as the Packet Reception Rate (PRR), require to estimate the success ratio in radio transmission (Baccour et al., 2009).

The radio signal metric most widely-used in localization and tracking in ubiquitous systems is RSSI. RSSI relies on the fact that the strength of the radio signals attenuates with distance. A node receiving a packet is able to measure its signal strength (RSSI) and use it to estimate the distance to the emitter of the packet. Equation (2.5) shows a widely used model that relates the received power strength $P(d)$ in *dBm* based on the distance to the transmitter (Seidel and Rappaport, 1992):

$$P_r(d) = P_0(d_0) - 10 n_p \log_{10}\left(\frac{d}{d_0}\right) + X_\sigma, \tag{2.5}$$

where $P_0(d_0)$ is the strength of the transmitter, n_p is the path loss exponent, that measures the rate at which the radio strength decreases with distance. X_σ is a

Gaussian random variable with zero mean and standard deviation σ, that represents the random effect caused by the fading. Both n_p and σ depend on the environment surrounding the emitter and receiver nodes, and because of this the precision of this computation depends on the particular conditions of the surroundings. Moreover, multi-path, shadowing and path-loss effects affect RSSI measurements (Zanella, 2016).

The RSSI model can also be obtained experimentally by fitting using regression the RSSI and range measurements. In these cases the above model is simplified, as in (Kumar et al., 2009), which adopts the following expression:

$$rssi(d) = A \log d + B, \tag{2.6}$$

where A and B are the parameters of the model obtained by regression. Figure 2.1 shows an example of a experimental RSSI-range model.

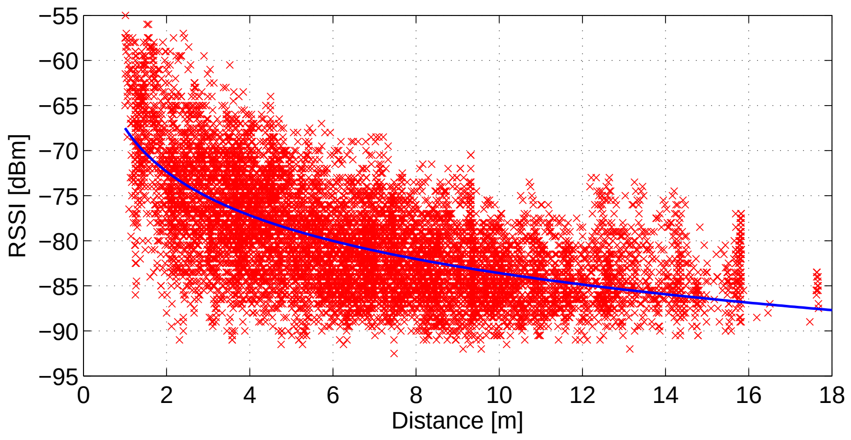


Figure 2.1: Example of an experimental RSSI-range model.

RSSI-based tracking has been extensively studied in the recent years. Its main advantage is that it does not require any additional hardware because it can be measured by the radio module of most ubiquitous computing nodes. RSSI-based techniques are energetically and economically inexpensive compared to other techniques. However, RSSI measurements are rather inaccurate. Reflections and other interactions with the environment, such as multi-path propagation, make RSSI measurements very dependent on the setting. This low accuracy perturbs its use in RO-SLAM, where an accurate initialization is needed. Thus, very few RO-SLAM techniques use RSSI measurements.

Despite the disadvantages of RSSI, it has been used in many localization, tracking and mapping techniques.

Time of Arrival (TOA)

These techniques compute the range through the propagation delay between a transmitter and a receiver ("time of flight"), assuming that the propagation speed is known. This technique can be further classified between TOA one-way ranging (2.7) and TOA two-ways ranging or RTT (Round Trip Time) (2.8). The former requires perfect synchronization between transmitter and receiver, while the latter, measuring the delay between the transmission of a signal and the reception of a signal's answer, does not require synchronization (Guvenc and Chong, 2009). Apart from radio signals, lasers and ultrasounds are very commonly used in TOA-based localization methods.

$$distance = speed \times time \tag{2.7}$$

$$distance = \frac{speed \times (t_{RT} - \triangle t)}{2} \tag{2.8}$$

where t_{RT} is total time spent by signal during the round-trip travel and $\triangle t$ is the processing time taken by both nodes.

An alternative system that can provide the accuracy and robustness needed by indoor positioning systems and having an advantage of low power and low cost is the Ultra-Wideband (UWB) technology. UWB allows up to a few centimeters ranging accuracy ranging, and involve short discrete transmission pulses instead of continuously modulating a code into a carrier signal (Roy et al., 2004). This technology provides high data rates for radio communications, extremely high accuracy for localization systems and good resolution for radars, which using an inherently low cost architecture and only milli-watts of power (Yang and Giannakis, 2004).

A valuable aspect of UWB technology is the ability for a UWB radio system to determine the "time of flight" of the transmission at various frequencies. This helps to overcome multipath propagation, as at least some of the frequencies have a line-of-sight (LOS) trajectory. With a cooperative symmetric two-way measuring technique, distances can be measured to high resolution and accuracy by compensating for local clock drifts and stochastic inaccuracies. Furthermore, UWB signals can even penetrate through walls and grounds, allowing to have Line-of Sight (LOS) access where other technologies do not.

Time Difference of Arrival (TDOA)

These techniques take advantage of the combination of ultrasound/acoustic and radio signals to estimate distance by determining the time difference between the arrival of these signals. Each device must be equipped with a speaker and a microphone.

The idea of TDOA ranging is graphically explained in Figure 2.2. A transmitter node sends a radio signal, waits a fixed time t_{delay} and sends an acoustic or ultrasound signal pattern with its speaker. If a receiver receives the radio signal then if registers the current time, t_{radio} and turns on its microphone. When the receiver hears the acoustic signal, it registers the current time, t_{sound}. The distance between the transmitter and the receiver can be computed as:

$$d = \frac{\nu_{radio}\nu_{sound}}{\nu_{radio} - \nu_{sound}}(t_{sound} - t_{radio} - t_{delay}), \tag{2.9}$$

where ν_{radio} and ν_{sound} are respectively the speed of propagation of radio and sound signals. Radio propagation is much faster than sound propagation, so (2.9) can be simplified to $d = \nu_{sound}(t_{sound} - t_{radio} - t_{delay})$.

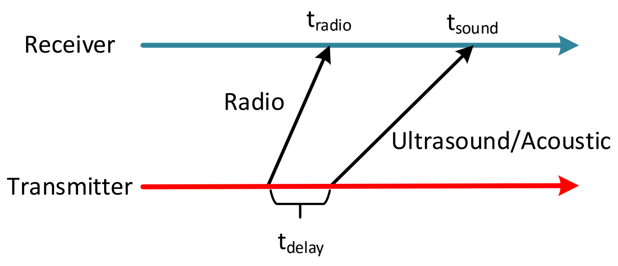


Figure 2.2: TDOA ranging.

TDOA techniques can achieve high accuracies. For example, works such as (Savvides et al., 2001) and the cricket system (Priyantha et al., 2000) claim accuracies of few centimeters over ranges of several meters. TDOA techniques require line of sight communication between the transmitter and the receiver, which is not always possible in many environments. Also, each node should be equipped with a speaker and a microphone.

Discussion

Table 2.1 summarizes the characteristics of the different kind of range sensors explained before. RSSI provides very noisy measurements but every radio transceiver can measure the strength of the received signal, which makes it the cheapest way to have a range estimation. On the other hand, TOA and TDOA perform much better in accuracy. The only problem is that the better the accuracy, the more expensive it becomes.

	RSSI	TOA	TDOA
Accuracy	Low	Medium/high	High
Prize	Cheap	Expensive	Expensive

Table 2.1: Characteristics of range sensing technologies.

2.3.2 Range-based localization and tracking in sensor networks

Localization in sensor networks is achieved by deploying nodes with range sensors at known locations. Having this “map” of nodes the network can estimate the location of a target in the scenario.

Most localization methods in sensor networks use RSSI as range sensor. In RSSI-based localization and tracking techniques the target exploits the attenuation properties of the radio signals to estimate its location. RSSI-based techniques have been traditionally classified in range-based, which use the distance between nodes computed from RSSI measurements, or range-free, which use the RSSI measurements to establish geometry relations. They also can be classified in active, in which require the target carries one node and collaborate with anchor nodes, or passive methods, which track targets relying on the disturbances on RSSI originated by the presence of the target.

Below the main localization and tracking methods are summarized.

Multilateration

Multilateration (Wang et al., 2009; Wessels et al., 2010) is one of the simplest methods to determine the target location having the distance to several anchor nodes. The target measures the range between itself and the anchor nodes. The location of the target is obtained by calculating the intersection point of the circles centered at the anchor nodes with radii equal to the estimated distances, as shown in Figure 2.3.

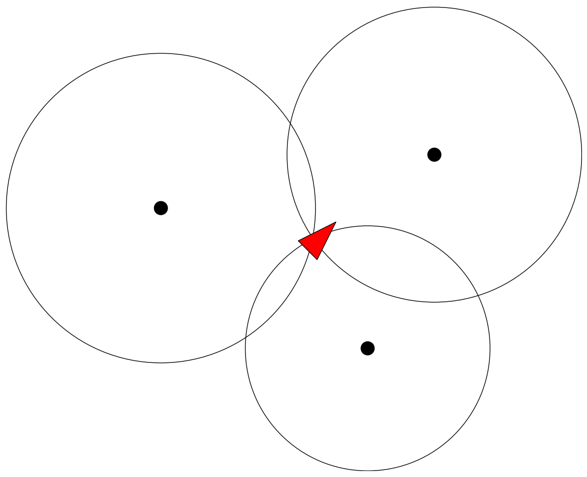


Figure 2.3: Lateration example.

However, the result of multilateration is a unique position as long as the distance measurements are perfect (noiseless). There are implementations for non perfect range measurements that analyze target location solutions obtained using pairs of reference anchor nodes. Each pair provides two intersections: one correct and one incorrect. All the pairs will create a set of intersections. The correct solutions distribute in a cluster that can be easily identified. The final estimation is the mean of the coordinates of the intersections in the identified cluster.

Least squares

The least squares method is an approach to find an approximate solution in overdetermined systems of equations. It can be applied to range-based localization as follows. Let $X = [x, y]^T$ be the target 2D position to be determined and $X_l = [x_l, y_l]^T$ the known coordinates of the lth anchor node, $l = 1, 2, \ldots, L$, where L is the number of anchor nodes receiving beacons from the target. The distance between the target and anchor node l, denoted by d_l, is

$$d_l = \sqrt{(x - x_l)^2 + (y - y_l)^2} \tag{2.10}$$

Introducing a new variable, range R, $R = x^2 + y^2$ and squaring both sides of (2.10) it results:

$$-2x_l x - 2y_l y + R = d_l^2 - x_l^2 - y_l^2 \tag{2.11}$$

Expression (2.11) can be represented in matrix form as $A\theta = b$, where:

$$A = \begin{bmatrix} -2x_1 & -2y_1 & 1 \\ -2x_2 & -2y_2 & 1 \\ \vdots & \vdots & \vdots \\ -2x_L & -2y_L & 1 \end{bmatrix} \qquad \theta = \begin{bmatrix} x & y & R \end{bmatrix}^T \qquad b = \begin{bmatrix} d_1^2 - x_1^2 - y_1^2 \\ d_2^2 - x_2^2 - y_2^2 \\ \vdots \\ d_L^2 - x_L^2 - y_L^2 \end{bmatrix} \tag{2.12}$$

Thus, the least square estimate $\hat{\theta}$ can be computed as follows:

$$\hat{\theta} = (A^T A)^{-1} A^T b \tag{2.13}$$

Thereby, from $\hat{\theta}$ the estimated location of the target $\hat{X} = [\hat{x}, \hat{y}]$ can be easily obtained. Least squares is optimum in case of range measurements with Gaussian noise. However, RSSI noise is not Gaussian (Lee et al., 2007) and other methods can obtain higher accuracies in this case.

MinMax

MinMax (Langendoen and Reijers, 2003) is a very popular localization algorithm, due to its simple implementation. Like in multilateration, the estimated distance between the target and the anchor nodes is needed. A pair of horizontal lines and a pair of vertical lines are drawn around each anchor node at the estimated distance to the target. Then, each anchor is at the center of a square. The estimated localization of the target is the center of the overlapping area of all the squares, see Figure 2.4.

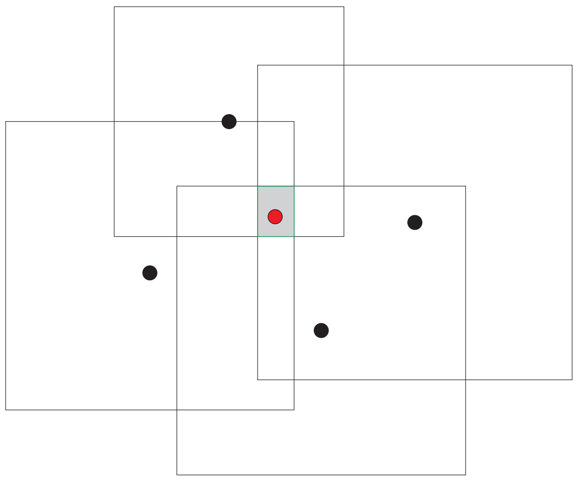


Figure 2.4: Example of the operation of the MinMax algorithm.

Intuitively, the accuracy is better if the intersection area is smaller. Thus, a certain error is unavoidable even if the ranging is perfect.

ROCRSSI

ROCRSSI (Liu et al., 2004) is a range free algorithm that relies on the assumption that RSSI decreases with the distance between transmitter and receiver. Assume a set of anchor nodes with known locations. Each anchor node receives packets from the target and from the other anchor nodes. Then, the measurements coming from anchor nodes are divided in two sets: the first one with RSSI values lower than the RSSI of the target and the second with RSSI values greater than the RSSI of the target.

The maximum RSSI value in the first set and the minimum RSSI value in the second one are assumed to be close to the target, defining an inner ring of radius $R1$ and an outer ring of radius $R2$. Drawing the rings for all the anchor nodes a set of overlapping ring pairs are created. The estimated location of the target is in the centroid of these overlapping ring pairs.

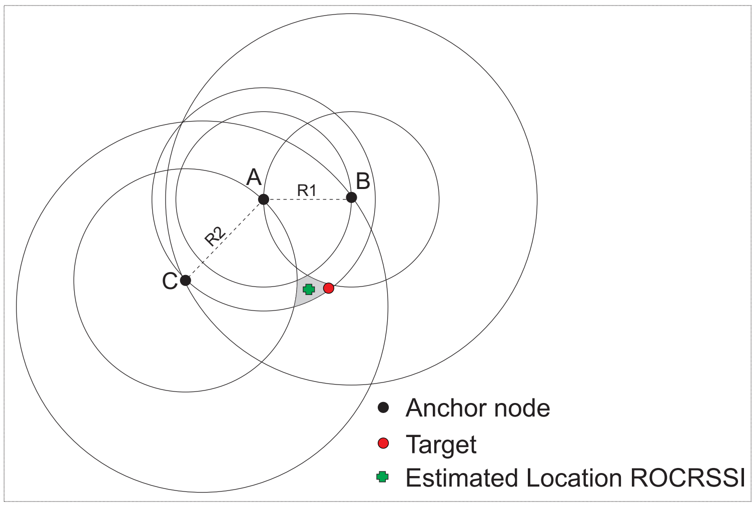


Figure 2.5: Example of the operation of the ROCRSSI algorithm.

Figure 2.5 shows an example. Anchor node A reads RSSI values from B, C and from the target T, as $RSSI_{AB}$, $RSSI_{AC}$ and $RSSI_{AT}$. They are ordered as $RSSI_{AB} < RSSI_{AT} < RSSI_{AC}$. B is in a first set of anchor nodes and C, in the second set. The maximum value in the first set is $RSSI_{AB}$ and the distance between A and B is $R1$. Similarly, the minimum value in the second set is $RSSI_{AC}$ and the distance between A and C is $R2$. The target is assumed to be between the two circles defined by $R1$ and $R2$. Repeating this process for anchor nodes B and C, the estimated target location lies in the centroid of the intersection of the rings.

This algorithm is range free, i.e. it does not need to estimate the distances among nodes, but just to compare between the received RSSI measurements.

Weighted centroid localization

WCL (Weighted Centroid Localization) (Blumenthal et al., 2007) exhibits high robustness against noise in the RSSI measurements. (Blumenthal et al., 2007) demonstrated that although other methods like, for example Least Squares (LS), are optimal with noiseless RSSI measurements, the performance of WCL is better with realistic (non-Gaussian) levels of noise.

Assume that the target receives packets from anchor nodes located at known positions and it reads their RSSI. In this algorithm the location of the target node i is computed using the expression:

$$L_i = \frac{\sum_{j=1}^{n}(\omega_{ij} L_j)}{\sum_{j=1}^{n}\omega_{ij}}, \tag{2.14}$$

where n is the size of the data set of RSSI measurements received by the target and ω_{ij} are weighting factors that depend on the distance from the anchor node to the target, computed through the received signal strength:

$$\omega_{ij} = \frac{1}{(D_{ij})^p}, \tag{2.15}$$

where p is an exponent to modify the influence of distance in the weights. Higher p gives more relevance to measurements from nearby static anchor nodes. RSSI-range models become flat –*i.e.*, insensitive to range– as range increases. The measurements from distant anchor nodes provide less useful information and are more affected by noise.

In (Clemente et al., 2012) experimental tests to determine the best values of p were performed. The number of anchor nodes, m, used in the localization method was also analyzed. Figure 2.6 shows the mean localization errors obtained. A minimum of three anchor nodes were necessary to obtain reasonable localization errors. The results also revealed that taking into account distant anchor nodes frequently perturbs the localization performance. Measurements from distant anchor nodes often include low information and high noise.

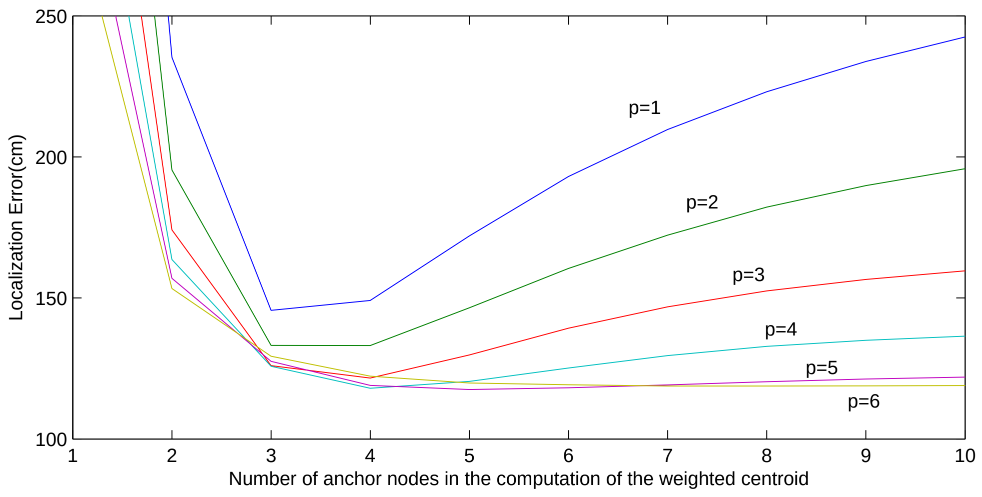


Figure 2.6: Evaluation of the effect of p and the number of anchor nodes in the WCL method.

Maximum likelihood

The Maximum Likelihood (ML) (Patwari et al., 2001) localization technique is based on classical statistical inference theory. Given the vector of RSSI measurements $r = \{r_1, r_2, \ldots, r_n\}$ that the target received from n anchor nodes with coordinates $X = \{x_1, x_2, \ldots, x_n\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$, the ML algorithm computes the a priori probability of receiving r for each potential position $[x, y]$ of the target. The position that maximizes the probability is then selected as the estimated target location.

The Maximum Likelihood method is more complex than other methods like WCL or MinMax, but it minimizes the variance of the estimation error as the number of observations, i.e, of anchor nodes, grows. However, in most realistic cases the number of anchor nodes are very limited, and its performance can be rather unsatisfactory.

RSSI map-based algorithms

RSSI map-based algorithms differ from other localization principles. They determine the location of the target by comparing the obtained RSSI values to a radio map. The radio map is constructed in an off-line phase and it contains the measured RSSI patterns at certain locations. The characteristics of the signal propagation in the environments are captured and the modeling of the complex signal propagation is avoided. However, the extraction of the radio maps is quite laborious and is not robust to changes in the environment. There are two different approaches: active and passive.

- **Active:** Active RSSI map-based localization and tracking is often called fingerprinting (Honkavirta et al., 2009). In fingerprinting a target receives signals from anchor nodes or vice versa, the RSSI of the signals is measured and compared to a radio map to obtain a location.

  The construction of the radio map begins by dividing the area of interest into cells. RSSI values of the radio signals are collected in points inside the cells and stored in a radio map. The ith element in the radio map is:

  $$M_i = \left(B_i, \{\vec{a}_{ij} \mid j \in N_i\}\right), i = 1, \ldots, M, \tag{2.16}$$

  where B_i is the ith cell, whose center is p_i. Vector $\vec{a}_{ij}$ holds the RSSI values measured from anchor node j and N_i is the set of anchor nodes whose signals can be received in cell i.

  The radio map can be modified or preprocessed before applying it in the location estimation phase. The motivation can be the reduction of the memory requirements of the radio map or the reduction of the computational cost of location estimation.

  Given the radio map, the objective of the location estimation phase is to infer the location of the target from the received measurements vector y which includes RSSI samples y_j from several anchors. In some cases several RSSI measurements from the anchor are collected before the state estimate is computed and the mean of the values is used to reduce sensitivity to noise.

  The estimation of the state can be made in a deterministic way, assuming that the state x is a non random vector, through the weighted K-nearest neighbor method (Bahl and Padmanabhan, 2000). Or if the state x is assumed to be a random variable, with the maximum likelihood estimate or with RBFs like KFs or PFs.

- **Passive:** Passive RSSI map-based localization and tracking (Youssef et al., 2007) relies on the fact that radio signals are affected by changes on the environment. By continuously recording and analyzing the signal strength this method can detect the changes in the environment and correlate them with entities and their locations.

  It has been used to identify and locate the presence of a human body in a room. The radio frequency used by the nodes is 2.4 GHz. The human body contains more than 70% of water and it is known that the resonance frequency of water is 2.4 GHz. Thus, the human body reacts as an absorber attenuating the wireless signal.

  The system consists of several pairs of nodes, acting as transmitter and receptor, deployed in an indoor environment, for example one floor in a building. To construct the radio map a person stands at different points and the signal strength characteristics are recorded. For every point the signal strength histogram of each pair of nodes is obtained. Based on the constructed radio map, a Bayesian inversion-based inference algorithm is used to compare the RSSI vector, that contains an entry for every pair of nodes, to the radio map.

Recursive Bayesian filtering

Localization and tracking can be done with RBFs. Knowing the location of anchor nodes and taking range measurements to them from the target and/or vice versa, it is simple to implement one of these tools in order to locate the target. Examples and details of RBFs have been given in Section 2.2.3.

2.3.3 Range-based mapping in sensor networks

The mapping problem is dual to localization. A mobile node (e.g. a node carried by a mobile robot) has knowledge of its pose in a scenario where a set of SN nodes have been deployed at unknown locations. Using range measurements this mobile node can build a “map” of static SN nodes. In environmental monitoring applications the SN measurements are meaningless without knowing the location from where the measurements were obtained. Moreover, having this map allows location estimation of other mobile nodes.

Building a map of nodes using a mobile robot can be achieved by classic approaches like Kalman Filters. The main advantage of using SN nodes for mapping is that SNs can naturally transmit their ID, solving the data association problem, for instance to establish the correspondence between measurements and features in the map, which is a big problem in mapping using other sensors.

Besides, in the SN domain many researchers worked in the idea of self-localization in SN (Wei et al., 2015; Niculescu, 2004; Meertens and Fitzpatrick, 2004; Shang et al., 2003). In this case the static SN nodes build their own map using range measurements between them. One could think it is the perfect solution for SN mapping, since it only needs the own SN nodes for building the map. However, these solutions always need some anchor nodes with previously known locations.

Self-localization in SN has attracted many authors and different and interesting solutions have been proposed. In particular, some authors came up with solutions based on nonparametric belief propagation (NBP) (Ihler et al., 2005), a variant of the popular belief propagation (BP) algorithm (Pearl, 2014). BP can be formulated as an iterative, local message passing algorithm, in which each node computes its “belief” about its associated variable, communicates this belief to and receives messages from its neighbors, then updates its belief and repeats.

2.4 Simultaneous localization and mapping with range-only measurements

This sections is devoted to RO-SLAM techniques, the main topic of this PhD Thesis. It describes the RO-SLAM problem, the main steps in its solutions and the most widely-used techniques for each of the steps. Although this chapter is dealt with more details, again the objective is not to be exhaustive but to give a general survey of the reported approaches and techniques. The description presented in this section will be complemented with deeper and more focused descriptions in each of the required chapters.

Figure 2.7 shows a general block diagram for RO-SLAM. Most reported SLAM techniques make a distinction between the front-end, where sensor information is processed and potentially spurious readings identified, and the back-end that solves the estimation problem. Many publicly available SLAM algorithms only contain the back-end as the front-end tends to be very much sensor and robot specific. In the most simple RO-SLAM, the front-end comprises a *measurement gathering* block, which is responsible of how and when the range measurements are gathered. On the other hand, the back-end is comprised of two blocks: *initialization tool* and the core *SLAM filter*.

RO-SLAM relies only on range measurements, which inherently cause the problem of partial observability: only one measurement is insufficient to constrain one location. Thus, RO-SLAM methods require the robot to move and integrate measurements from different positions in order to initialize the locations of the beacons. Two basic approaches have been used to solve beacon initialization: directly introducing the measurements using a multi-hypothesis SLAM filter –the so-called undelayed SLAM approach, or combining the SLAM filter with tools for initializing beacons –the delayed SLAM approach. Examples of tools for delayed initialization are multilateration, probability grids and Particle Filters (PFs).

Like in the Range-Bearing SLAM, the EKF is probably the most commonly adopted estimation tool in RO-SLAM. However, there are many different approaches used in the literature and these methods are combined with distinct initialization tools.

The following sections are devoted to describing the main landmark initialization tools and the main core estimation tools used in reported RO-SLAM solutions. Measurement gathering, how and when range measurements are gathered, is a key issue in the presented works developed in this PhD Thesis and are analyzed in detail in each of the chapters.

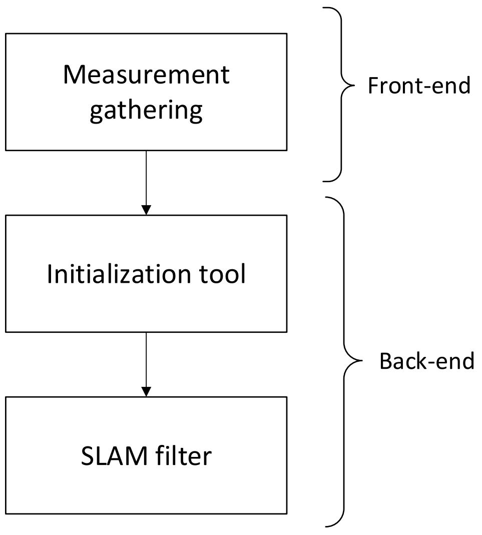


Figure 2.7: General block diagram of RO-SLAM techniques.

2.4.1 SLAM beacon initialization

Beacon initialization is a critical step in RO-SLAM and has very high impact on SLAM accuracy and performance. As said before, there are two main types of beacon initialization. First, the most natural way is a delayed initialization, in which an external tool provides the SLAM back-end with estimates of the beacons location after integrating a number of range measurements. On the other hand, in the undelayed initialization performs the idea is to introduce measurements in the SLAM core filter directly without waiting for any initialization. This can be done in case of using multi-hypothesis filters in the SLAM back-end.

The most used initialization tools are detailed in the following. First, three methods for delayed initialization: multilateration, probability grids and Particle Filters. Last, Gaussian mixtures for undelayed initialization.

Multilateration

Multilateration is the straightforward initialization tool. It is based on the same idea used for SN localization, see Section 2.3.2. The robot gathers measurements of one beacon from different locations, and then tries to estimate the pose of the beacon through a simple least squares optimization. Least squares is needed in order to deal with the noise of the measurements. Figure 2.8 shows an example with a robot –represented as red triangle in the figure– taking four measurements to one beacon –represented as a blue star.

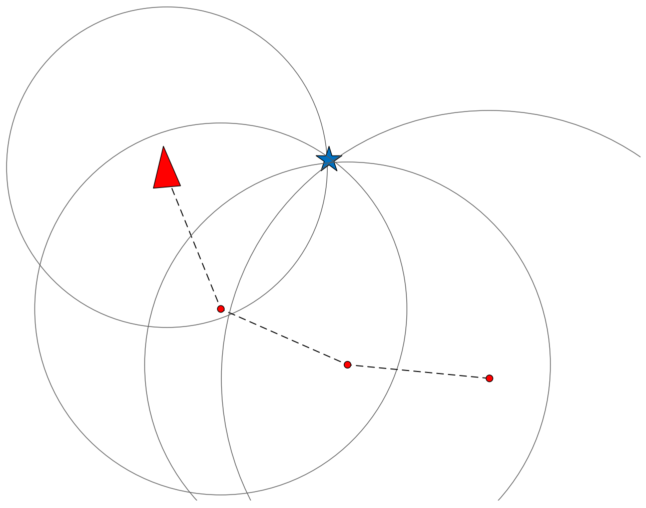


Figure 2.8: Example of multilateration beacon initialization.

Multilateration is probably the simplest beacon initialization tool. It is computationally efficient, but lacks robustness. It is very sensible to measurement noise and outliers, and it can cause bad beacon initialization, which can lead to significant SLAM estimation errors.

Probability grids

Probability grids provide more robustness than multilateration but it is not scalable to large scenarios and the accuracy is related to the size of the cells in the grid. Thus, implementation in 3D SLAM would largely increase the amount of cells and then augment the computational burden.

The idea behind probability grids is the discretization of the physical world into a 2D (or 3D) grid, with each grid cell corresponding to a rectangular (or cube) area . Each measurement "votes" for its possible solutions. Ideally, solutions that are near each other should end up in the same cell, even in the presence of noise. This can be accomplished by choosing a grid size that matches the total uncertainty. Once that all votes have been added to the accumulator, one can search for the cell with the greatest number of votes. In order to find a solution, a vote ratio must be defined. Higher ratios increase confidence but they can delay making a decision. Figure 2.9 shows an example of a probability grid initialization.

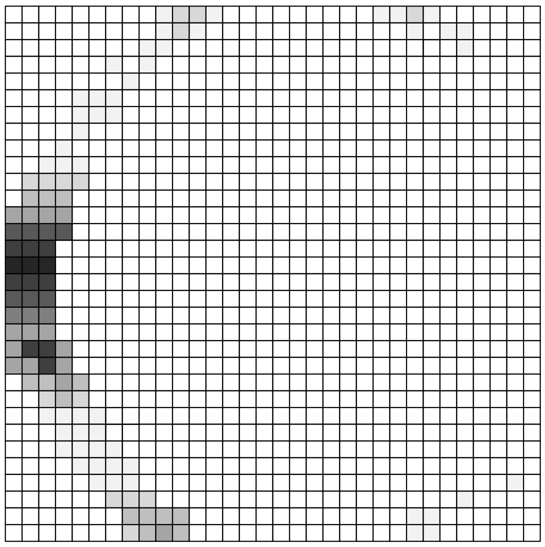


Figure 2.9: Example of probability grid initialization. Darker cells have more votes and then have more probability of being the chosen beacon initialized location.

Particle Filters

Particle Filters are widely-adopted beacon initialization tools in SLAM. PFs, see Section 2.2.3, can represent any probability distribution. In this case the probability distribution has the shape of a ring (in 2D). As more range measurements are integrated the distribution of the particles tend to converge to a Gaussian like distribution. When the covariance of the distribution of the particles is lower than a certain bound, it is said that the PF has converged and the beacon initialization is completed.

PFs are computationally hard, but they can provide a pure probabilistic solution to the beacon initialization problem. The accuracy is related to the number of particles, but of course having more particles increases the resource consumption. One advantage of using PFs for beacon initialization is that it is a well known tool which has been very much researched. Thus, there are different versions of PFs which can improve the drawbacks of the basic PF implementation.

Figure 2.10 shows an example of the PF initialization. The particles are initially deployed in a ring. After integrating some range measurements in the PF from different robot locations, the particles tend to converge to the beacon location.

Gaussian mixture

A probability mixture model is a probability distribution that is a convex combination of other distributions. Mixture models are a semi-parametric alternative to nonparametric probability distribution (as Particle Filters) and provide greater flexibility and precision in modeling the underlying statistics of sample data.

Gaussian mixtures are a type of density model which comprises a number of Gaussian functions. These component functions are combined to provide a multi-modal density. Thus, if we assume that the discrete random variable X is a mixture of k component discrete Gaussian variables $\mathcal{N}(\mu_i, \sigma_i)$, then, the probability mass function of X, $f_X(x)$, is a weighted sum of its component distributions:

$$f_X(x) = \sum_{i=1}^{k} \omega_i \mathcal{N}(\mu_i, \sigma_i) \tag{2.17}$$

where $0 \leq \omega_i \leq 1$ and $\sum_{i=1}^{k} \omega_i = 1$.

Gaussian mixtures can represent any distribution as a linear combination of Gaussian distributions. For beacon initialization they provide a multi-hypothesis scheme in which only one of the Gaussians will be the chosen one and the other will be discarded. Figure 2.11 shows an example of Gaussian mixture initialization.

The main advantage is using Gaussian mixtures is that all these Gaussians can be added to the SLAM filter from the very beginning, e.g. in case of having a Gaussian filter, e.g. EKF. It is not necessary to wait until only one Gaussian hypothesis lasts. That is why it can be an undelayed initialization.

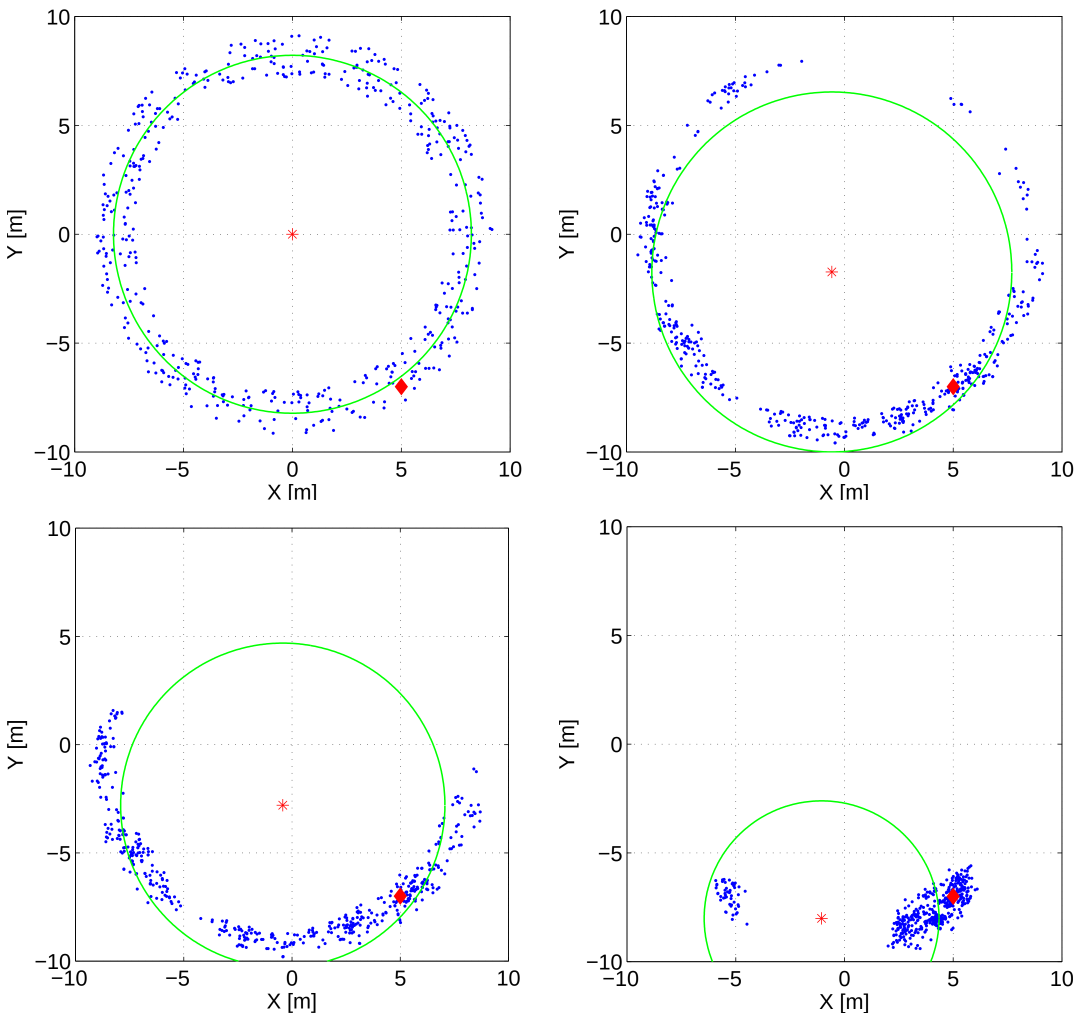


Figure 2.10: Distribution of particles at different times as more measurements are integrated in the PF. Legend: robot (red stars), beacon location (red diamond), particles (blue points) and measurements (green circles centered in the robot with radii z_t).

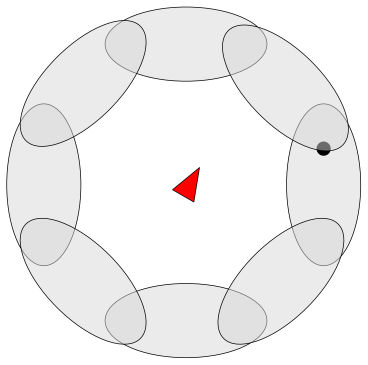


Figure 2.11: Example of Gaussian mixture initialization.

2.4.2 RO-SLAM methods

Once the techniques of Bayes filtering have been presented in Section 2.2.3, it is turn to take a look into the different implementations of these techniques for solving the RO-SLAM problem. There are two main types of solutions in reported works, one using an EKF as the core SLAM estimator tool and the other using a non-parametric filter similar to PFs.

EKF-SLAM

The EKF-SLAM algorithm is the most widely-used method in RO-SLAM. It is an EKF implementation in which the state vector is comprised of the robot location and the locations of the beacons in the environment:

$$x = [x_r, x_1, x_2, \cdots, x_n]^T, \tag{2.18}$$

where x_r is the location $[x, y, z]$ of the robot and x_i is the location $[x, y, z]$ of beacon b_i.

The Prediction and Update stages perform as described in Section 2.2.3. The observation model for range-only measurements from beacon b_i is the Euclidean distance between the beacon and the robot locations:

$$h_t(\mu_t) = \sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2}, \tag{2.19}$$

where $\delta_x = \mu_x^r - \mu_x^i$, $\delta_y = \mu_y^r - \mu_y^i$ and $\delta_z = \mu_z^r - \mu_z^i$. $h_t(\mu_t)$ is nonlinear and its Jacobian is:

$$H_t = \frac{\partial h_{r,i}}{\partial \mu} = \begin{bmatrix} \frac{\delta_x}{h_t} & \frac{\delta_y}{h_t} & \frac{\delta_z}{h_t} & \cdots & \frac{-\delta_x}{h_t} & \frac{-\delta_y}{h_t} & \frac{-\delta_z}{h_t} & \cdots \end{bmatrix} \tag{2.20}$$

Many works implemented EKF-SLAM for range-only sensors. The difference between them is the initialization method. EKF-SLAM has been combined with multilateration (Menegatti et al., 2009), probability grids (Djugash et al., 2006; Olson et al., 2004), PF-based initialization (Menegatti et al., 2010) and Gaussian mixtures (Caballero et al., 2010; Torres-González et al., 2014).

FastSLAM

The FastSLAM algorithm (Montemerlo et al., 2002) marked a fundamental conceptual shift in the design of recursive probabilistic SLAM. Previous efforts focused on improving the performance of EKF-SLAM, while retaining its essential linear Gaussian assumptions. FastSLAM, with its basis in recursive MonteCarlo sampling, or particle filtering, was the first to directly represent the nonlinear process model and non-Gaussian pose distribution.

The high dimensional state-space of the SLAM problem makes direct application of Particle Filters computationally infeasible. However, it is possible to reduce the sample space by applying Rao-Blackwellization (R-B), whereby a joint state is partitioned according to the product rule $P(x_1, x_2) = P(x_2|x_1)P(x_1)$ and, if $P(x_2|x_1)$ can be represented analytically, only $P(x_1)$ need be sampled $x_1^{(i)} \sim P(x_1)$. The joint distribution, therefore, is represented by the set $\{x_1^{(i)}, P(x2|x_1^{(i)})\}_i^N$ and statistics such as the marginal

$$P(x_2) \approx \frac{1}{N} \sum_i^N P\left(x_2|x_1^{(i)}\right) \tag{2.21}$$

can be obtained with higher accuracy than when sampling over the joint space.

The joint SLAM state may be factored into a vehicle component and a conditional map component:

$$P\left(X_{0:k}, m|Z_{0:k}, U_{0:k}, x_0\right) = P\left(m|X_{0:k}, Z_{0:k}\right) P\left(X_{0:k}|Z_{0:k}, U_{0:k}, x_0\right) \tag{2.22}$$

Here the probability distribution is on the trajectory $X_{0:k}$ rather than the single pose x_k because, when conditioned on the trajectory, the map landmarks become independent. This is a key property of FastSLAM and the reason for its efficiency: the map is represented as a set of independent Gaussians, with linear complexity, rather than a joint map covariance with quadratic complexity.

The essential structure of FastSLAM, then, is a Rao-Blackwellized state, where the trajectory is represented by weighted samples and the map is computed analytically. Thus, the joint distribution at time k is represented by the set $\{w_k^{(i)}$, $X_{0:k}^{(i)}$, $P(m|X_{0:k}^{(i)}, Z_{0:k)}\}_i^N$, where the map accompanying each particle is composed of independent Gaussian distributions:

$$P\left(m|X_{0:k}^{(i)}, Z_{0:k}\right) = \prod_{j}^{M} P\left(m_j|X_{0:k}^{(i)}, Z_{0:k}\right) \tag{2.23}$$

Recursive estimation is performed by particle filtering for the pose states and, by the EKF, for the map states.

Updating the map for a given trajectory particle $X_{0:k}^{(i)}$ is trivial. Each observed landmark is processed individually as an EKF measurement update from a known pose. Unobserved landmarks are unchanged. Propagating the pose particles, on the other hand, is more complex, since it requires the execution of the following steps: 1) draw a sample; 2) weight samples according to the importance function; 3) if necessary, perform resampling; 4) for each particle, perform an EKF update on the observed landmarks as a simple mapping operation with known vehicle pose. Steps 1 to 3 are the PF update, described in Section 2.2.3.

Rao-Blackwellization has been used in RO-SLAM (Blanco et al., 2008b,a). Both works proposed an Rao-Blackwellized Particle Filter (RBPF) as the SLAM back-end algorithm. The difference is that the first one uses a PF-based delayed initialization while the second one implements a Gaussian mixture for each beacon. Work (Hai et al., 2010) introduced another RBPF with auxiliary PF initialization. Once a beacon is initialized, its PF turns into an EKF in order to increase efficiency.

Work (Sun et al., 2009) developed *RSLAM*, which is a variant of FastSLAM algorithm, extended for range-only measurements and the multi-robot case.

Others

There are some other works presenting different approaches in RO-SLAM than the "classical" EKF-SLAM and FastSLAM.

Work (Kuai et al., 2010) divides the RO-SLAM problem. It localizes the robot by triangulation and uses a two-step PF for map building.

Work (Djugash et al., 2008) proposes a decentralized mapping algorithm which can then be used to localize robots. This is achieved by an EKF in polar space. The decentralization is provided by a message-passing algorithm based on belief propagation.

Work (Lourenço et al., 2013) presents a globally asymptotically stable solution to RO-SLAM. It designs a nonlinear sensor-based system with its dynamics augmented so that the proposed formulation can be considered as linear time-varying (LTV) for the purpose of observability analysis. Then a standard discrete-time LTV Kalman Filter is designed to solve the established system.

2.4.3 Discussion

SLAM front-end has not been treated in literature, but it will be an important part of this PhD Thesis. On the other hand, the back-end has received many different proposals both for beacon initialization and for the core SLAM filter.

Landmark initialization is an fundamental part of RO-SLAM since a bad initialization could lead to inconsistent mapping. Multilateration methods, although simple and efficient, are very (too) sensitive to measurement noise and outliers.

Probability grids provide better beacon initialization, but their accuracy depends on the size and resolution of the grid. Particle Filters (PFs) are maybe the most widely used beacon initialization tools in RO-SLAM. They provide better accuracy and a good number of mechanisms have been developed to reduce their computational burden. PFs are the best choice for delayed initialization. On the other hand, it performs undelayed initialization, in which a number of hypotheses are directly introduced in SLAM filter. It is usually achieved by Gaussian mixtures. Table 2.2 summarizes the initialization methods and their characteristics.

	Delayed/Undelayed	**Burden**	**Accuracy**
Multilateration	Delayed	Low	Low
Probability grids	Delayed	High (not scalable)	Medium
Particle Filters	Delayed	High	High
Gaussian mixture	Undelayed	Medium	High

Table 2.2: Comparison of the different beacon initialization methods in RO-SLAM.

RO-SLAM filters commonly used in the literature are divided between EKF and PF-based SLAM. EKF-SLAM is by far the most popular method, it is a well-known algorithm and is efficient with small-medium sized maps. Later in this PhD Thesis we will propose an EIF-based algorithm which main features are its efficiency and scalability. PF-based algorithms like FastSLAM have become popular in visual SLAM and it also has its implementation in RO-SLAM.

It is interesting to point out that most existing RO-SLAM techniques consider beacons as passive devices ignoring their sensing, computational and communication capabilities. Many of them use sensor network nodes as beacons but only integrate in SLAM robot-beacon measurements. This PhD Thesis proposes RO-SLAM techniques that use SN nodes as beacons and exploit the sensing, computational and communication capabilities of SN nodes. In particular, the analysis of the exploitation of beacon capabilities in RO-SLAM is part of **Contribution1** of this PhD Thesis and it will be presented in Chapter 3.

2.5 Conclusions

This chapter summarized the current state of the art in range-only SLAM and explained the basis on which this PhD Thesis lies. The SLAM problem and its characteristics and most common solutions were presented. A review of the sensor network technology was provided, including a good number of localization and tracking methods with range-only sensors that could serve as an introduction to the RO-SLAM solutions.

Most RO-SLAM methods in the literature ignore the potential synergies resulting from the cooperation between the robot and sensor networks, which, as we will analyzed further in this PhD Thesis, has a good number of advantages and possibilities. Moreover, most of reported works focused on the development of back-end SLAM algorithms ignoring the possibilities of the front-end. In this PhD Thesis we will develop schemes that exploit the front-end and the sensing, processing and communication capabilities of beacons in order to improve the estimation and efficiency of RO-SLAM.

Chapter 3

Architecture for accurate and efficient range-only SLAM

3.1 Introduction

This chapter presents the general architecture for accurate and efficient RO-SLAM that is proposed in this PhD Thesis. This architecture exploits the capabilities of sensor network nodes and is **Contribution1** of this PhD Thesis. This chapter also introduces different RO-SLAM schemes that follow this architecture adopting different approaches to fulfill the accuracy and efficiency objectives.

The adopted architecture has two main characteristics. First, it exploits the sensing, computational and communication capabilities of sensor network nodes. The sensor network nodes actively participate in the execution of the RO-SLAM filter. Second, it integrates not only robot-beacon measurements but also range measurements between two different beacons, the so-called inter-beacon measurements. Further in this chapter the implications of integrating in SLAM inter-beacon measurements, its advantages and also its drawbacks are discussed. Summarizing, integrating inter-beacon measurements improves estimation accuracies and convergence speed, but gathering and integrating these “extra” measurements involves higher resource consumption, which is often constrained in the envisioned applications. Thus, the RO-SLAM schemes proposed in this PhD intend to benefit from the inter-beacon measurements promoting resource consumption efficiency using different approaches.

This chapter is structured as follows. Section 3.2 starts specifying the requirements of the architecture, presenting the general architecture and briefly describing the main modules and methods of the architecture. In Section 3.3 different RO-SLAM schemes that follow the presented architecture are presented. They adopt different approaches in order to fulfill the accuracy and efficiency objectives. They will be described in detail in further chapters. Section 3.4 describes the characteristics, gathering protocol and integration in SLAM of inter-beacon measurements. The conclusions are in the final section.

3.2 The presented RO-SLAM architecture

3.2.1 Problem statement and requirements

Consider a GPS-denied scenario where a large number of sensor nodes have been deployed. The location of the nodes is assumed unknown. For instance, they have been placed at random locations for real-time monitoring of an industry accident, or they are used for monitoring an industrial facility and their exact location was not registered during deployment. Each sensor node gathers measurements and transmits them to a Monitoring Station. Thus, nodes are endowed with sensing, computing and communication capabilities. We assume that each sensor network node can measure the distance to other nodes within its sensing region. This is not a hard requirement, in fact most Commercial Off-The-Shelf (COTS) sensor network nodes can measure the radio signal strength (RSSI) of incoming packets and estimate the range to the emitting node (Banatre et al., 2008).

We are interested in RO-SLAM techniques that use sensor nodes as radio beacons –landmarks– to online estimate the locations of sensor nodes and of the robot. Accurate mapping and robot localization is critical for the navigation of the robot in complex GPS-denied environments. Also, the localization of sensor nodes –mapping– is necessary to geolocate the measurements collected. Besides, knowing the location of robots and nodes enables robot-sensor node cooperative missions of interest in these scenarios such as sensor node transportation and deployment (Corke et al., 2006; Maza et al., 2011) or collection of data from sensors (Martinez-de Dios et al., 2013). The potentialities of robot-sensor network collaboration has originated significant interest in RO-SLAM techniques that employ sensor nodes as beacons, see e.g. (Challa et al., 2005; Menegatti et al., 2009; Nogueira et al., 2010). However, most reported SLAM techniques consider beacons as passive landmarks and do not exploit the capacities they are actually endowed with.

The above problem is rather general. The characteristics of sensor networks and ubiquitous computing systems and technologies impose a basic set of requirements in our problem. Ubiquitous computing systems consist of a high number of low-cost nodes equipped with sensing, actuating, computing and communication capabilities that organize autonomously into networks in order to achieve a global mission. Nodes are designed to be low cost, i.e. engineered to have low energy consumption, equipped with low sensing/actuating resources and low computing capability. Below, the main requirements in our problem are briefly discussed:

- **Accuracy.** RO-SLAM estimations should be as accurate as possible both in robot localization and also in beacons mapping. Improving accuracy often requires higher consumption of resources. The architecture should allow establishing trade-offs between accuracy and efficiency.

- **Efficiency.** RO-SLAM computation involves significant resource consumption. Low energy consumption, low computational capability and low communication ranges are inherent constraints in ubiquitous computing nodes. The architecture and techniques involved should consume as low resources as possible. These resources include energy, computational burden and memory and communications bandwidth, among others.

- **Modularity.** The architecture and the techniques developed should have a clear modular approach that allow their scalability and expandability.

- **Robustness.** RO-SLAM estimations should be robust to noise perturbations, node failures and other unexpected events. Sensor network nodes are designed to be low cost. However, in contrast to the individual fragility of each node, the strength of the architecture should originate from the cooperation between many nodes. The architecture and methods to be implemented should be robust to failures of individual nodes and errors in the wireless communications.

- **Scalability.** The proposed RO-SLAM architecture should have good scalability with the problem size and particularly with the number of nodes in order to enable its use in large scenarios.

3.2.2 Assumptions

SLAM assumes that there is no previous knowledge neither in the robot nor in the sensor nodes. The robot has no knowledge of its pose or the locations of the static sensor nodes. Even, each node has no knowledge of its own position. The only assumption is that the robot kinematic model and the sensor observation models are known. Nodes do not have any knowledge of any other node in the setting. If node A needs parameters or data from another node B, it has to 'ask' node B using a certain protocol.

The proposed architecture assumes that the scenario and node locations are realistic. Most of the assumptions considered are implemented in the greater majority of Commercial Off The Shelf (COTS) devices and do not involve any practical constraint.

It is assumed that nodes and communications can fail. Measurements from sensors are subject to noise. It is also assumed that the scenario conditions can change. For instance, radio propagation depends on the physical configuration of the environment. Besides, beacons –i.e., sensor nodes– have a realistic energy consumption model. In fact, we took the energy parameters from datasheets of COTS devices including nodes such as the *Nanotron nanoPAN* (see Appendix B).

3.2.3 RO-SLAM general scheme

As presented in Section 2.4, RO-SLAM methods can be divided into: the front-end, where sensor measurements are processed and potentially spurious readings identified; and the back-end, which solves the estimation problem. Most works in RO-SLAM focused only on the back-end as it can be seen as the core of the SLAM problem, but the fact is that managing the measurement gathering can provide several benefits to the final result.

Most reported RO-SLAM methods are executed in a centralized manner in the robot. In these methods all tasks in RO-SLAM are executed in the robot, including measurement gathering, integration of measurements in RO-SLAM and the Prediction stage. These fully centralized RO-SLAM methods require high computational burden in the robot and have very poor scalability. In these methods beacons act as passive devices disregarding the sensing, computational and communication capabilities COTS sensor nodes are actually endowed with.

The proposed general architecture for RO-SLAM is shown in Figure 3.1. Both, the robot and the beacons actively participate performing the RO-SLAM front-end and the SLAM back-end tasks. The architecture makes use of two main properties: integration of inter-beacon measurements and involvement of beacons in the computation of RO-SLAM.

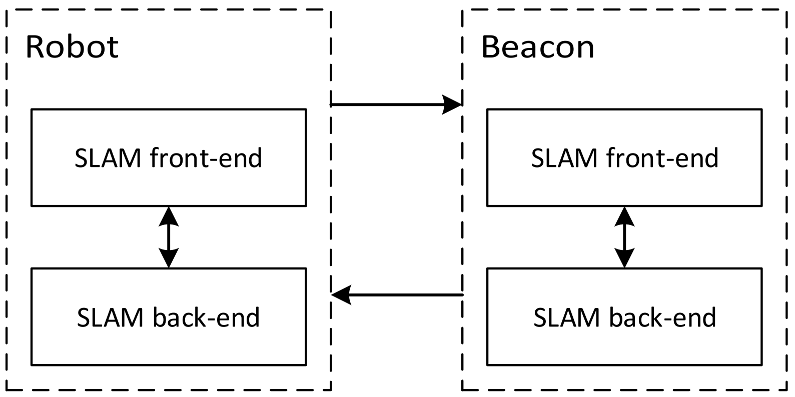


Figure 3.1: General block diagram of the RO-SLAM architecture proposed in this PhD Thesis.

Most of the reported RO-SLAM methods integrate only direct robot-beacon range measurements. The proposed architecture exploits the capability of the beacons of taking range measurements to other beacons and transmitting them using *ad hoc* protocols. It integrates in SLAM not only direct measurements but range measurements between beacons, called inter-beacon measurements. Integration of inter-beacon measurements significantly improves map estimation. However, it involves higher consumption of resources, such as the energy required to take and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements.

The proposed architecture exploits the capability of the beacons of executing SLAM tasks reducing the robot resource consumption. As a result the architecture has the following properties:

- Accuracy. Integrating inter-beacon measurements directly improves the mapping estimation uncertainty and, indirectly, the localization uncertainty.
- Efficiency. Decentralization helps to reduce the computational burden and in general to reduce resource consumption.
- Robustness. Decentralization makes RO-SLAM more robust to failures of nodes.
- Scalability. Decentralization improves scalability with the problem size and with the number of nodes.

3.3 Developed RO-SLAM schemes

The presented architecture has been used to develop three main different RO-SLAM schemes, which are **Contribution2**, **Contribution3** and **Contribution4** of this PhD Thesis. They satisfy the main properties of the architecture (integration of inter-beacon measurements and involvement of beacons in RO-SLAM computation) but implement them adopting different approaches. The differences between these schemes are in the way they extend the RO-SLAM front-end both in the robot and in the beacons. They make use of two main mechanisms: measurement gathering control and distribution of RO-SLAM Update stage between beacons.

Measurement gathering control

Integration in RO-SLAM of inter-beacon measurements significantly improves map estimation but involves high consumption of resources, such as the energy required to gather and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements. This mechanism includes a supervision module that monitors the SLAM performance and dynamically modifies the gathering of robot-beacon measurements and inter-beacon measurements balancing estimation accuracy and resource consumption.

Distribution of RO-SLAM Update stage between beacons

In most RO-SLAM methods beacons act as passive devices. The robot gathers range measurements to beacons, integrates the measurements and performs the rest of the SLAM tasks. In the proposed architecture the beacons are actively involved in the computation of the SLAM tasks, reducing the number of actions performed by the robot. In the simplest of the RO-SLAM schemes the beacons gather measurements to other beacons. In the most complex, the beacons select the range measurements to collect, gather these measurements and execute in a distributed manner the RO-SLAM Update stage in order to integrate its measurements in SLAM.

The above two mechanisms are implemented following different approaches and resulting in three RO-SLAM schemes that are presented in the following.

3.3.1 Scheme1: RO-SLAM with dynamically configurable measurement gathering

The objective of **Scheme1** is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a centralized mechanism running in the robot that adapts measurement gathering. A simple representation of this scheme is shown in Figure 3.2. It actuates over the measurement gathering by intentionally modifying the rate and variety of range measurements that are gathered and integrated in the SLAM filter. It exploits the fact that sensor network nodes can execute flexible *ad hoc* measurement gathering strategies. The SLAM front-end is based on another two main modules: Measurement Gathering and SLAM Supervisor.

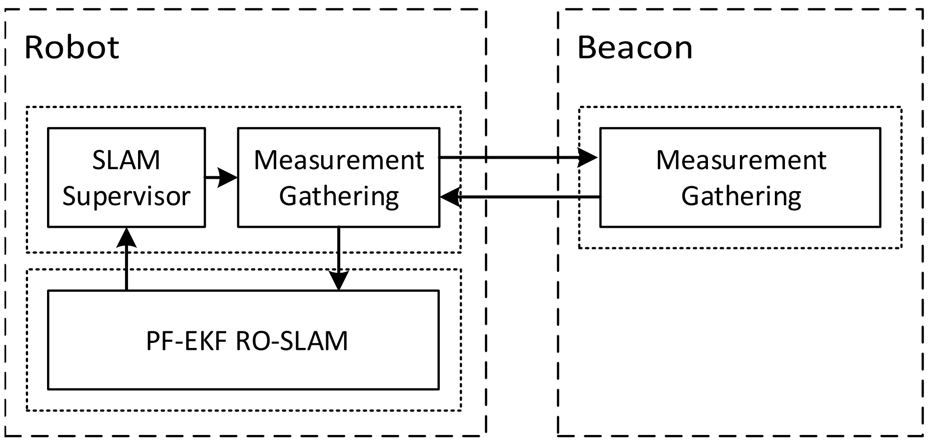


Figure 3.2: General diagram of **Scheme1**, dynamically configurable RO-SLAM scheme with inter-beacon range measurements.

The Measurement Gathering module enables gathering and collecting direct robot-beacon, as well as inter-beacon measurements. It performs a controlled flooding protocol in which each beacon gathers range measurements to its nearby beacons and transmits them back to the robot, naturally avoiding repeated measurements. Measurement gathering can be configured to be performed at different rates and also with different inter-beacon depth levels, so that the robot can integrate measurements of beacons that are distant from the robot's sensing region.

The SLAM Supervisor receives as input metrics of the SLAM performance and dynamically selects the most suitable measurement gathering mode for the current conditions.

Scheme1 is general and can be applied to any back-end SLAM filter. In this chapter it has been applied to a EKF RO-SLAM filter that is combined with PF for beacon initialization. EKF-PF RO-SLAM is probably one of the best known and most widely researched RO-SLAM methods.

Scheme1 is the **Contribution2** of this PhD Thesis and will be presented in Chapter 4.

3.3.2 Scheme2: Distributed SEIF-based RO-SLAM with inter-beacon measurements

The objective of **Scheme2** is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate its measurements in SLAM. This scheme is shown in Figure 3.3.

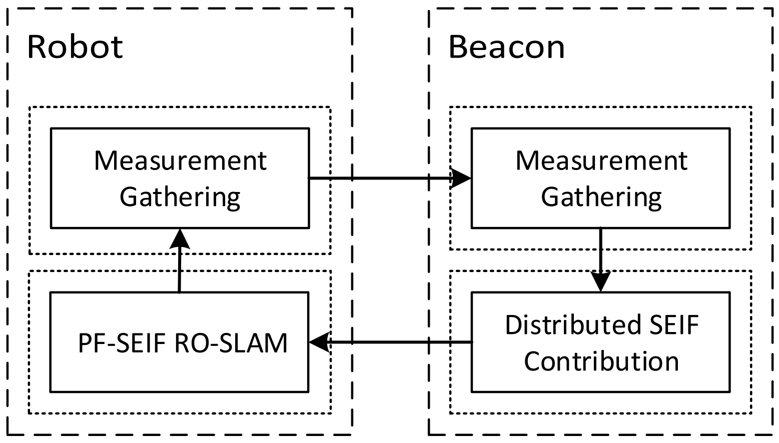


Figure 3.3: General diagram of **Scheme2**, distributed SEIF-based RO-SLAM scheme with inter-beacon measurements.

Scheme2 exploits the robot-sensor network collaboration by distributing measurement gathering and integration in SLAM between the beacons around the robot. The SLAM back-end is based on Sparse Extended Information Filter (SEIF) (Thrun et al., 2004) –and inherits its efficiency and scalability– combined with PFs for beacon initialization. Its distributed approach shares resource consumption, reducing robot CPU burden, and at the same time naturally integrates inter-beacon measurements, which improves map and robot localization accuracies. In Chapter 5 it is proven that **Scheme2** accumulates higher amount of information than the conventional SEIF SLAM and, at the same time, it preserves the sparsity of the information matrix and its constant time property.

Scheme2 is the **Contribution3** of this PhD Thesis and will be presented in Chapter 5.

3.3.3 Scheme3: Resource-constrained SEIF-based RO-SLAM with inter-beacon measurements

Scheme3 includes the two aforementioned mechanisms –measurement gathering control and distribution of RO-SLAM Update stage between beacons– in order to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements. This scheme is shown in Figure 3.4.

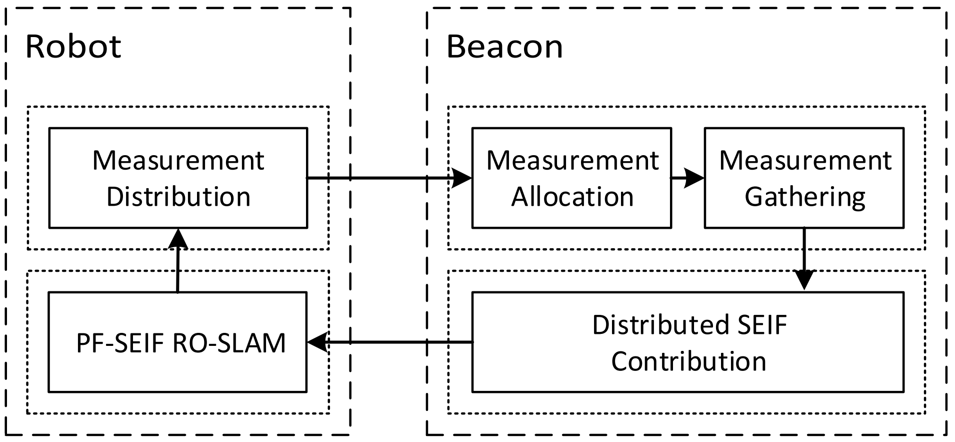


Figure 3.4: General diagram of **Scheme3**, resource-constrained SEIF-based RO-SLAM scheme with inter-beacon measurements.

Scheme3 exploits robot-beacon cooperation to improve SLAM accuracy and efficiency while meeting a given resource consumption bound. The resource consumption bound is expressed in terms of the maximum number of measurements that can be integrated in SLAM per iteration. The sensing channel capacity used, the beacon energy consumed or the computational capacity employed, among others, are proportional to the number of measurements that are gathered and integrated in SLAM. **Scheme3** can meet static and dynamic bounds, e.g. determined by an online resource allocation tool, enabling high flexibility, which can be of interest in many applications. It employs a distributed Sparse Extended Information Filter (SEIF) SLAM method combined with PFs for beacon initialization, in which each beacon gathers and integrates in the SLAM Update stage robot-beacon and inter-beacon measurements.

It also comprises a distributed tool that uses greedy gain-cost analysis to dynamically select the most informative measurements to be integrated in SLAM. This tool includes two modules: Measurement Distribution, which executes in the robot, and Measurement Allocation, which is executed in each beacon. This RO-SLAM scheme has a robot-beacon distributed approach were beacons actively participate in measurement selection, gathering and integration in SLAM.

Scheme3 is the **Contribution4** of this PhD Thesis and will be presented in Chapter 6.

3.4 Inter-beacon measurements

3.4.1 Introduction

The architecture proposed in this PhD Thesis integrates robot-beacon as well inter-beacon range measurements. This section describes how the inter-beacon measures are collected and integrated in the RO-SLAM filters.

Integrating in SLAM inter-beacon range measurements involves a number of interesting advantages: they reduce map and robot estimation errors and accelerate beacon initialization. These advantages are highlighted in previous works from the authors, such as (Torres-González et al., 2014). However, despite these advantages, very few methods exploiting inter-beacon measurements have been proposed. The general idea of using inter-beacon measurements was given in (Djugash et al., 2006), in which different ways for incorporating inter-beacon measurements were proposed, mainly by using virtual nodes and adopting off-line map improvement using multidimensional scaling (MDS). MDS with inter-beacon measurements was also used in (Bardella et al., 2012). These off-line approaches are not suitable for most applications, which require online map and robot locations.

This section studies the impact of inter-beacon measurements in RO-SLAM. First, it proposes a protocol to collect inter-beacon measurements using controlled flooding with a configurable number of hops. With high number of hops it can collect measurements between two beacons that are further beyond the robot's sensing range. Increasing the hop number increases the speed and accuracy of beacon initialization, which indirectly improves robot estimation.

Second, a simple integration of inter-beacon measurements in an EKF RO-SLAM with PF initialization is presented. Then, its performance is analyzed with different noise levels.

3.4.2 Inter-beacon measurements collection protocol

This section describes the general protocol for collection of inter-beacon measurements implemented in the schemes proposed in this PhD Thesis. Once collected, the measurements can be integrated in the RO-SLAM back-end. Next section will analyze the effects of integrating inter-beacon measurements in an EKF RO-SLAM with PF initialization.

Each measurement collection event is divided in two parts: the forward stage and the backward stage. The forward stage performs a cascade-like measuring of inter-beacon distances. The origin of the cascade is the robot and its depth is NH, the hop number. All messages in this stage include a field nh representing the number of hops remaining until the end of the forward stage. The backward stage orderly collects measurements from the involved beacons using backward messages. All messages include a sequence number Seq that identifies the measurement event. Each beacon tracks the Seq of the last measurement event it was involved in.

The forward stage starts when the robot broadcasts a forward message with $nh = NH$ to the beacons within its sensing region. Each beacon b_i receiving the message checks if it is a new measurement event. If it is not, the message is ignored. Otherwise, beacon i updates $nh = nh - 1$, measures its distance to all the beacons in its sensing region and buffers the measurements in ms_i, the measurement set for beacon b_i. If $nh > 0$, it broadcasts a forward message with the new nh. The beacon keeps the ID of its parent beacon –from which it received the forward message– and starts its backward stage. If $nh = 0$, that forward message reached its hop limit and it is not retransmitted: the forward stage ends and the backward stage starts. Then, beacon i creates a backward message with ms_i and sends it to its father. In the backward stage each beacon updates its measurement set. When beacon b_i receives a backward message with a suitable Seq, it adds to ms_i the measurements contained in the message. Each beacon keeps in backward stage until its timeout expires. Then, it sends a backward message containing ms_i to its parent beacon.

Figure 3.5 illustrates its operation with different NH. With $NH = 0$ –implemented by measurement collection in traditional RO-SLAM methods– the robot does not broadcast the forward message and only collects robot-beacon measurements $\{z_{r,1}, z_{r,2}, z_{r,3}\}$. With $NH = 1$, the robot collects the same measurements that with $NH = 0$ and also collects the following measurements $\{z_{1,r}, z_{2,r}, z_{2,3}, z_{2,5}, z_{3,r}, z_{3,2}, z_{3,4}\}$. With $NH = 2$, among others the robot collects measurements between beacons that are beyond its sensing range such as $z_{4,6}$. This protocol can dynamically change NH. Also, notice that it prevents flooding cycles, naturally avoiding repeated measurements and canceling the need for additional filtering.

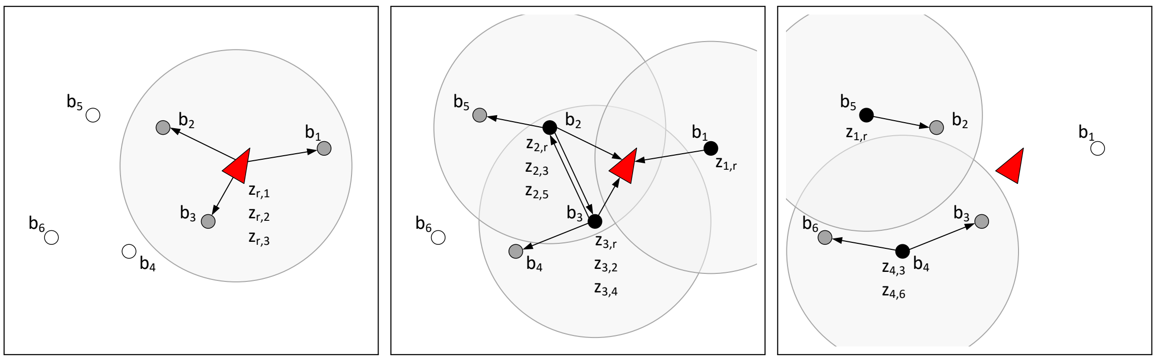


Figure 3.5: Examples of measurements collection with: $NH = 0$ (left), $NH = 1$ (center) and $NH = 2$ (right). Grey circles represent the sensing regions.

With $NH = 1$, the robot broadcasts a forward message with $nh = 1$. Beacon b_2 receives the message, updates $nh = 0$ and measures distances $z_{2,j} \forall j \in SR_2$, being

SR_2 the sensing region of beacon $ID = 2$. Since $nh = 0$, it sends a backward message with ims_2 to the robot -its father. At the end of the backward stage the robot has collected the following measurements $\{z_{r,1}, z_{r,2}, z_{r,3}, z_{1,r}, z_{2,r}, z_{2,3}, z_{2,5}, z_{3,r}, z_{3,2}, z_{3,4}\}$.

The protocol collects two measurements for each inter-beacon distance except for the deepest beacons, of which one is collected. Figure 3.5-right shows its operation with $NH = 2$ and the measurements collected by each beacon and the robot: it collects measurements between beacons that are beyond its sensing range. In this example beacon b_2 received two forward messages. The first one was sent by the robot. It also received one from beacon b_3 but it was ignored since it had non-new Seq values.

3.4.3 Integration in EKF-PF RO-SLAM

This section describes how inter-beacon measurements are integrated in EKF SLAM. Range measurements have the problem of partial observability. The EKF is combined with an auxiliary tool for beacon initialization. We adopt Particle Filters (PFs) as initialization tool. When the robot takes the first measurement from beacon b_i, the initialization of b_i is started and then beacon b_i enters at the "initialization phase". It keeps in this stage until the auxiliary tool converges and an initial estimation of the location of b_i is computed. Then, it is added to the EKF state vector. From now on we say that b_i is at the "state vector phase".

With $NH = 0$, the measurement collection protocol takes only robot-beacon measurements: they are integrated as in Section 2.4.2. In general with $NH > 0$ the protocol takes measurements between two beacons, being at least one of them at NH or less hops from the robot. The observation model for inter-beacon measurement $z_{i,j}$ between beacons b_i and b_j used is similar to that in (2.19) but adapted to consider the range between the estimated locations of both beacons.

$$h_{i,j}(\mu_t) = \sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2}, \tag{3.1}$$

where $\delta_x = \mu_x^i - \mu_x^j$, $\delta_y = \mu_y^i - \mu_y^j$ and $\delta_z = \mu_z^i - \mu_z^j$.

If both beacons are in the "state vector phase", $z_{i,j}$ is used to update the EKF using the following observation Jacobian:

$$H_{i,j} = \left[\ldots 0, \frac{\delta_x}{h_{i,j}}, \frac{\delta_y}{h_{i,j}}, \frac{\delta_z}{h_{i,j}}, 0 \ldots 0, \frac{-\delta_x}{h_{i,j}}, \frac{-\delta_y}{h_{i,j}}, \frac{-\delta_z}{h_{i,j}}, 0 \ldots \right] \tag{3.2}$$

All the terms in $H_{i,j}$ are zero except those for the entries corresponding to the locations of beacons b_i and b_j.

$z_{i,j}$ is integrated in SLAM differently depending on the phase of the involved beacons. Figure 3.6 summarizes the operation of the algorithm when a new inter-beacon measurement is taken. If only one of the beacons, e.g. beacon b_j, is in the "state vector phase", $z_{i,j}$ is used to update or initialize the PF of beacon b_i. If neither beacon i nor beacon b_j are in the "state vector phase", the measurement is kept for future use until one of the PFs, either PF_i or PF_j, converges. Instead of buffering all measurements, for simplification and efficiency, the robot only keeps the number –$nm_{i,j}$– and mean of the measurements. Assuming Gaussian noise, the average value is considered as a measurement but with variance $nm_{i,j}$ times lower than that of the range measurements.

As a result, the integration of inter-beacon measurements helps the convergence of other PFs and enables a chain reaction PF convergence effect. This effect drastically reduces the PF convergence times and helps to anticipate the deployment of other PFs.

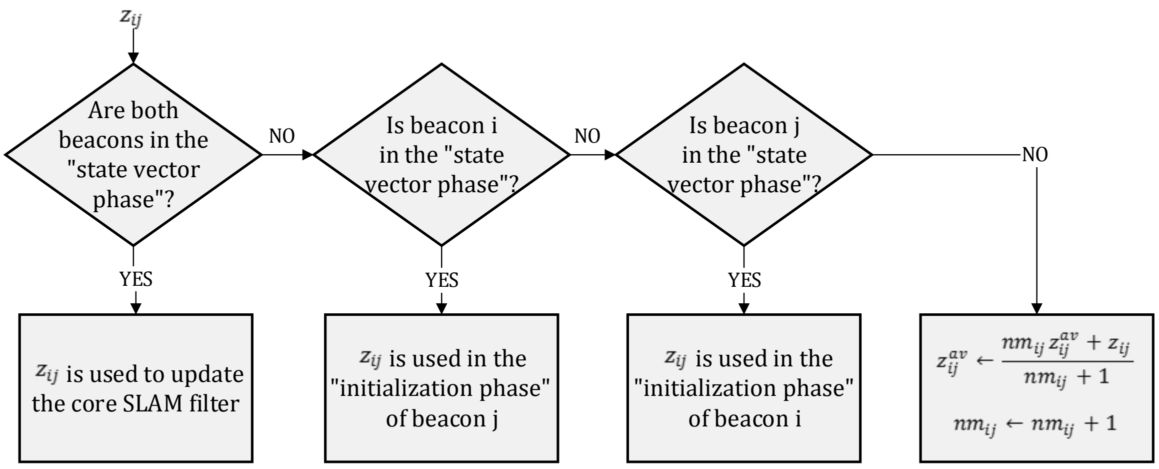

z_{ij}
Are both beacons in the "state vector phase"?
NO
Is beacon i in the "state vector phase"?
NO
Is beacon j in the "state vector phase"?
NO
YES
YES
YES
z_{ij} is used to update the core SLAM filter
z_{ij} is used in the "initialization phase" of beacon j
z_{ij} is used in the "initialization phase" of beacon i
$z_{ij}^{av} \leftarrow \frac{nm_{ij} z_{ij}^{av} + z_{ij}}{nm_{ij} + 1}$
$nm_{ij} \leftarrow nm_{ij} + 1$

Figure 3.6: Integration of inter-beacon measurements in the SLAM filter.

3.4.4 Results with EKF-PF RO-SLAM

This section shows the evaluation of EKF-PF RO-SLAM with inter-beacon measurements in real experiments performed in the CONET Integrated Testbed (Jiménez-González et al., 2011). A mobile robot is used to perform 2D RO-SLAM. A total of 43 beacons were deployed in the experiment. Nanotron nodes equipped with ToA sensors were used as beacons. The experimental characterization of these nodes can be found in Appendix B. The measurement noise follows a Gaussian distribution with an standard deviation of $\sigma_m = 1m$. In these experiments the robot will follow random trajectories within a scenario of 50x70 m. The auxiliary PFs were set with 300 particles. More details of the testbed and of these experiments can be found in Appendix A.

This section analyzes the performance of RO-SLAM with a very large number of experiments assuming a wide range of measurement and odometry noise levels. Hybrid experiment-simulation tests were performed. The real trajectories performed by the robot in the scenario were used but simulated noise level was added to the range measurements and odometry values obtained in the experiments. This section is divided in two parts. The first evaluates the effects of using different values of NH whereas the second one analyzes its performance with different odometry and measurement error levels.

In order to assess the performance of the inter-beacon measurements integration, series of 300 tests were performed with NH=0, NH=1 and NH=2. Each of them considered random beacon settings and robot trajectories. Figure 3.7 shows the mean error in the location of each static beacon in the three cases. The map error reduced significantly with higher NH.

Figure 3.8 compares the evolution of the map along one experiment with a EKF-PF filter that integrates only robot-beacon measurements (top) and a filter that integrates inter-beacon measurements with $NH = 1$. The particles of the PFs corresponding to beacons in the initialization phase are shown in magenta. When the PF of a beacon converges, the beacon location estimation is considered initialized and its estimation is represented as a green ellipse using the 3σ bound. The map estimation evolution is shown in three times: (left) $t = 24s$, when in the traditional SLAM the PFs of 10%

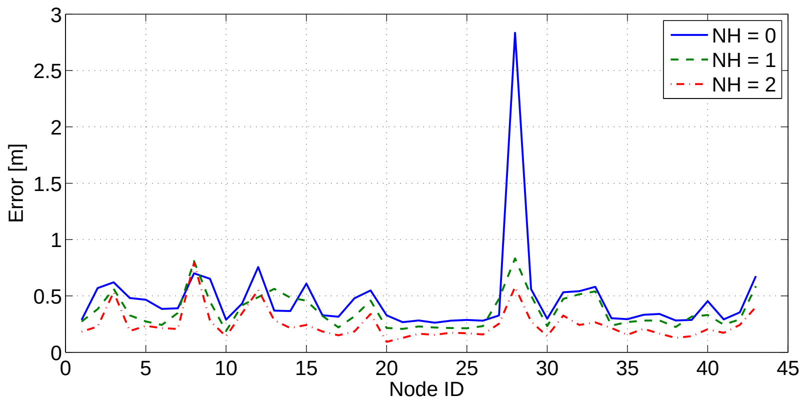


Figure 3.7: Mean error in the location of each static beacon with different NH.

of the beacons have converged; (center) $t = 26.4s$, when 90% of the beacons have converged; and (right) $t = 201s$, at the end of the experiment. It can be seen that integrating inter-beacon measurements enables faster and more accurate map building. Along the experiment the number of initialized beacons was significantly higher than when integrating only direct robot-beacon measurements. Also, the uncertainty of initialized beacons was lower, which can be noticed analyzing the size of the ellipses.

Figure 3.9 shows the mean time when the PF of each beacon were deployed (red color) and converged (blue). PFs converged in average at time $t = 175.15$s with $NH = 0$, at $t = 132.85$s with $NH = 1$ and at $t = 76.6$s with $NH = 2$. The PF convergence chain reaction effect is emphasized with higher NH. The improvement can be noticed in almost all beacons and is particularly evident in beacons that are distant from the robot initial position. With $NH = 2$ the multi-hop flooding protocol allows the robot to integrate measurements between two beacons beyond the robot's sensing range. Thus, some PFs converge even before the robot takes a first direct measurement for that beacon and these robot-beacon measurements are used directly in the EKF improving the beacon and robot estimations. This improvement can also be noticed in the PF deployment times. The average deployment time for $NH = 0$ was time $t = 57.1$s, while it was $t = 43.05$s for $NH = 1$ and $t = 30.45$s for $NH = 2$.

The overall number of measurements integrated in each simulation were: $\overline{nm} = 10046$ for $NH = 0$, $\overline{nm} = 22378$ for $NH = 1$ and $\overline{nm} = 31958$ for $NH = 2$. In order

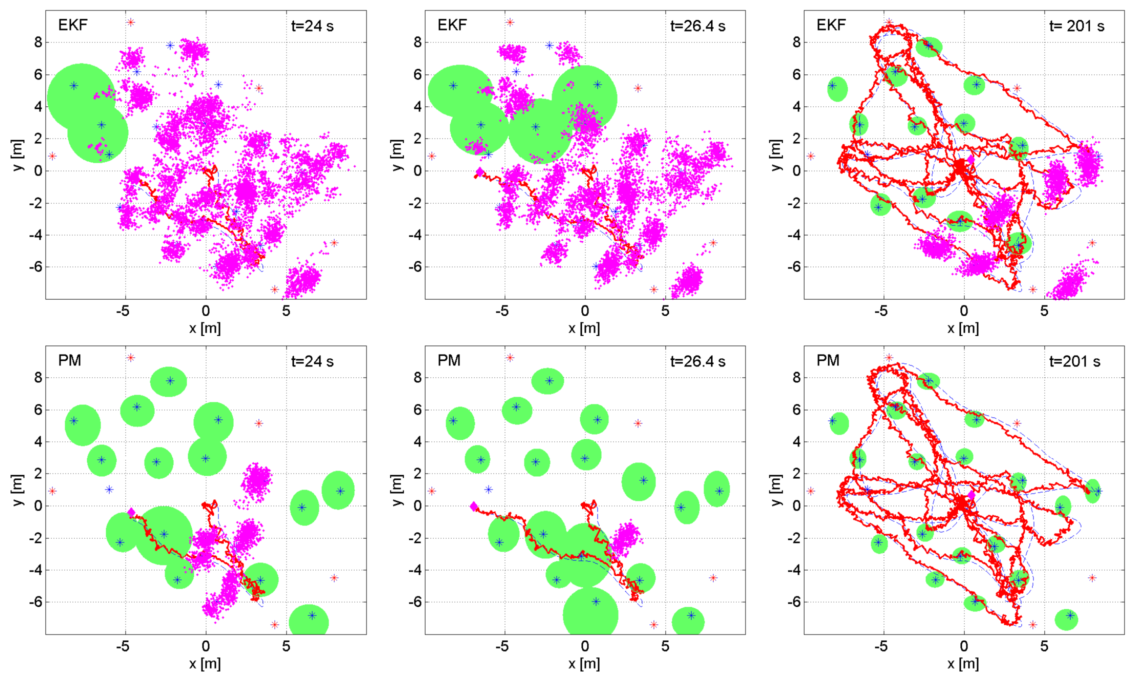


Figure 3.8: Evolution of map building for EKF-PF SLAM integrating only direct robot-beacon measurements (top) and integrating also inter-beacon measurements (bottom). Beacon PF particles are represented in magenta. The estimation of the initialized beacon locations is represented with ellipses using the 3σ bound. The map status at three different times is shown: (left) $t = 24s$, when the PFs of 10% of the beacons have converged in the traditional SLAM; (center) $t = 26.4s$, when 90% of the beacons in the EKF-PF SLAM method integrating inter-beacon measurements have converged; and (right) $t = 201s$, at the end of the experiment.

to have an estimate of the resource consumption, the computing times until 90% of the PFs had converged have been calculated. They were $\bar{t} = 9.32s$ ($NH = 0$), $\bar{t} = 8.38s$ ($NH = 1$) and $\bar{t} = 8.01s$ ($NH = 2$). These computing times have been calculated through *Matlab Profiler*, running in a i7-3630QM computer. The large increase in the number of measurements with higher NH was compensated with shorter PF convergence times, resulting in overall computational burden savings.

Tables 3.1 and 3.2 compare the robustness of using different NH values assuming three odometry error levels –good ($\sigma_{o,1} = 0.05m/s$), average ($\sigma_{o,2} = 0.15m/s$) and bad ($\sigma_{o,3} = 0.25m/s$)– and four measurement inaccuracies –good ($\sigma_{m,1} = 0.1m$), average ($\sigma_{m,2}$=0.5m), bad ($\sigma_{m,3}$=1m) and very bad ($\sigma_{m,4}$=1.5m). The tables summarize the

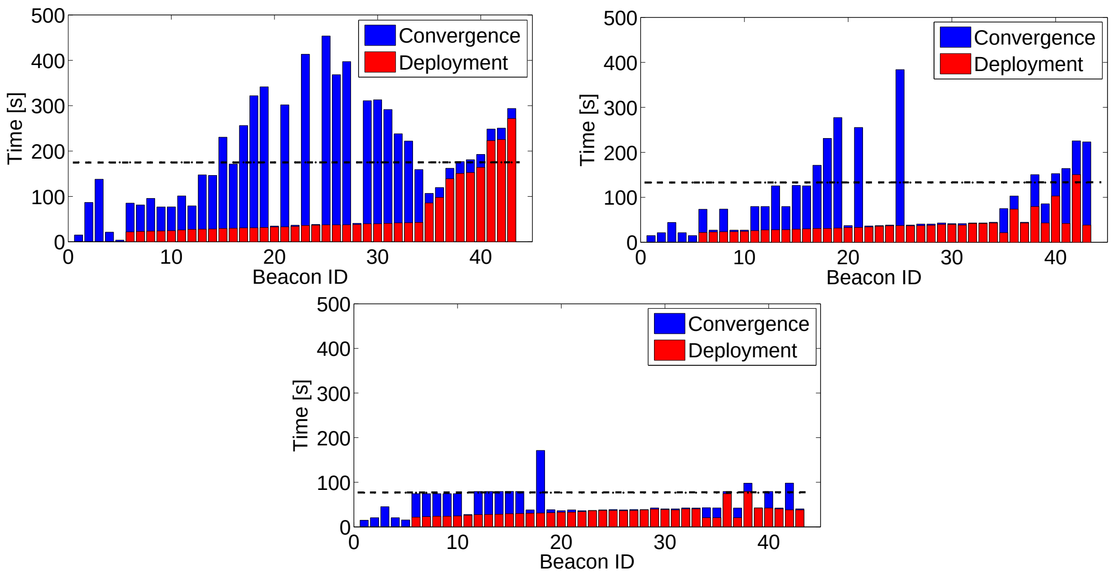


Figure 3.9: Deployment (red) and convergence (blue) times of each beacon PF with $NH = 0$ (top-left), $NH = 1$ (top-right) and $NH = 2$ (bottom). Black dashed lines represent the average convergence time for each value of NH.

improvement in performance originated from using inter-beacon measurements with NH=1 and NH=2 w.r.t. the traditional approach –NH=0. They show the mean results from 200 tests with different robot paths and randomly deployed beacons.

It is shown that RO-SLAM performance and robustness to noise level improves when increasing the hop number of inter-beacon measurements: the performance increment w.r.t. $NH = 0$ is higher with $NH = 2$ than with $NH = 1$. The use of inter-beacon measurements reduce the PF convergence times approximately in the same way (about 60% with $NH = 1$ and 70% with $NH = 2$) despite the odometry and measurement error levels. Also, the improvement in map estimation accuracy is more evident than in robot localization. This is attributed to the fact that inter-beacon measurements directly update landmark estimations and influence indirectly on the robot estimation.

It can also be noticed that the use of multi-hop inter-beacon measurements provides higher improvements w.r.t. $NH = 0$ with lower measurement noise levels and worse odometry. In this case the integration of inter-beacon measurements highly improves

	$\sigma_{m,1}$	$\sigma_{m,2}$	$\sigma_{m,3}$	$\sigma_{m,4}$	
Mean map error	33.5%	32.9%	30.2%	27.3%	
Mean robot error	9.7%	9.2%	8.6%	7.2%	$\sigma_{o,1}$
Mean init. time	60.1%	59.9%	56.8%	55.9%	
Mean map error	41.7%	40.3%	38.4%	34.1%	
Mean robot error	11.2%	10.8%	10.3%	9.4%	$\sigma_{o,2}$
Mean init. time	60.5%	59.7%	57.8%	56.2%	
Mean map error	45.6%	43.8%	42.2%	37.9%	
Mean robot error	12.4%	11.9%	11.1%	10.0%	$\sigma_{o,3}$
Mean init. time	60.5%	60.0%	57.1%	56.0%	

Table 3.1: Performance evaluation integrating inter-beacon measurements with $NH = 1$ w.r.t. integrating only robot-beacon measurements, $NH = 0$.

	$\sigma_{m,1}$	$\sigma_{m,2}$	$\sigma_{m,3}$	$\sigma_{m,4}$	
Mean map error	48.2%	46.8%	43.5%	39.7%	
Mean robot error	17.6%	16.9%	16.1%	14.9%	$\sigma_{o,1}$
Mean init. time	72.3%	71.2%	70.4%	66.7%	
Mean map error	54.8%	52.9%	50.1%	46.3%	
Mean robot error	19.5%	19.0%	18.1%	16.8%	$\sigma_{o,2}$
Mean init. time	72.6%	71.2%	70.6%	66.9%	
Mean map error	65.1%	63.5%	61.2%	57.4%	
Mean robot error	20.7%	20.1%	19.3%	18.2%	$\sigma_{o,3}$
Mean init. time	72.7%	71.4%	70.9%	67.2%	

Table 3.2: Performance evaluation integrating inter-beacon measurements with $NH = 2$ w.r.t. integrating only robot-beacon measurements, $NH = 0$.

the estimation. With higher measurement noise levels and better odometry, inter-beacon measurements are less useful and cause lower improvements.

3.4.5 Discussion

We noticed that the convergence of a PF triggers the integration of low-variance inter-beacon measurements, helping the convergence of other PFs and enabling a chain-reaction PF convergence effect. This effect drastically reduces the PF convergence times and helps to anticipate the deployment of other PFs. The effect is larger with higher numbers of inter-beacon relations, i.e. the number of beacons of which measurements can be taken by each beacon. Higher NH also enables the robot to collect measurements from a higher number of inter-beacon relations.

It is clear that integrating more measurements will reduce the uncertainty in the estimations. The importance of these new measurements is not only that they are more but that they are different. Inter-beacon measurements establish relationships between two points of the space (where the beacons are located) without the need of the robot to be located at these points. Thus, inter-beacon measurements will improve the overall estimation accuracy.

However, as it will be deeply discussed in further chapters of this PhD Thesis, taking and integrating much more measurements has a cost. Sometimes the cost of taking these measurements can be higher than the reward. The thing is that gathering and integrating multi-hop inter-beacon measurements is not for free, and it could be necessary to develop techniques which could balance between the advantages and trade-offs of integrating these measurements in RO-SLAM.

3.5 Conclusions

This chapter presented the general RO-SLAM architecture proposed as **Contribution1** of this PhD Theses. This architecture is based on two properties:

- Measurement gathering control. This mechanism includes a supervision module that monitors the SLAM performance and dynamically modifies the gathering of robot-beacon measurements and inter-beacon measurements balancing SLAM estimation accuracy and resource consumption.

- Distribution of RO-SLAM Update stage between beacons. Beacons are actively involved in the computation of the SLAM tasks, reducing the number of actions performed by the robot.

They are implemented following different approaches, resulting in the following three RO-SLAM schemes:

- **Scheme1: RO-SLAM with dynamically configurable measurement gathering**. This scheme is the **Contribution2** of this PhD Thesis and will be presented in Chapter 4.
- **Scheme2: Distributed SEIF-based RO-SLAM with inter-beacon measurements**. This scheme is the **Contribution3** of this PhD Thesis and will be presented in Chapter 5.
- **Scheme3: Resource-constrained SEIF-based RO-SLAM with inter-beacon measurements**. This scheme is the **Contribution4** of this PhD Thesis and will be presented in Chapter 6.

The proposed RO-SLAM architecture and all the proposed schemes integrate robot-beacon as well inter-beacon range measurements. This chapter also described how the inter-beacon measurements are collected and integrated in the RO-SLAM. The integration of inter-beacon measurements helps the convergence of other PFs and enables a chain reaction PF convergence effect. This effect drastically reduces the PF convergence times and helps to anticipate the deployment of other PFs. Besides, integrating inter-beacon measurements helps to improve map accuracy and indirectly, robot location estimation accuracy.

Chapter 4

Adaptive measurement gathering for efficient RO-SLAM

4.1 Introduction

As shown in Chapter 3, integrating inter-beacon measurements significantly improves RO-SLAM performance. However, gathering and integrating more measurements also involves higher consumption of limited resources such as energy, bandwidth and computational burden. This chapter presents a RO-SLAM scheme with dynamically adaptive measurement gathering. Its objective is to reduce the increment in resource consumption caused by the integration of inter-beacon measurements by adopting a mechanism that dynamically modifies the rate and variety of range measurements that are integrated in RO-SLAM. This chapter describes **Scheme1**, **Contribution2** of this PhD Thesis, the first of the RO-SLAM schemes that adopts the architecture presented in Section 3.3.

Scheme1 will be presented and tested using an EKF as the SLAM filter but it is a general approach which would work with any other implementation. Besides the SLAM filter, this scheme is based on another two main modules: Measurement Gathering and SLAM Supervisor.

The chapter is structured as follows. Section 4.2 introduces some related work specific on active SLAM. Section 4.3 describes the proposed **Scheme1**. Section

4.4 presents the SLAM Supervisor and Measurement Gathering modules and the algorithms involved. Section 4.5 shows a proof of concept and some experiments that highlight the main characteristics of **Scheme1** and help to understand how it operates. Conclusions of this chapter are in Section 4.6.

4.2 Related work

In the last few years several SLAM methods have been combined with active perception tools. These tools evaluate different sensing actions that the robot can perform in order to dynamically select the action that maximizes the information acquired by the robot. Active perception tools could be used in SLAM in order to accelerate the convergence of the uncertainty of an algorithm, in order to improve estimation accuracies or to increase the area explored by the robot.

Almost all active SLAM works have been focused on visual SLAM. Work (Davison and Murray, 2002) published one of the first general active vision systems for autonomous localization, addressing issues such as uncertainty-based measurement selection, automatic map maintenance and goal-directed steering. Work (Vidal-Calleja et al., 2006) actuated by moving the robot and the camera in order to optimize both localization and mapping uncertainties. Work (Frintrop and Jensfelt, 2008) used an active camera to track landmarks and explore unknown environments in visual SLAM.

A very low number of active RO-SLAM methods have been reported. Work (Merino et al., 2010) developed a method that actuated over the steering of a robot in order to maximize uncertainty reduction during beacon initialization.

Almost all active SLAM methods actuate over the robot trajectory. In fact, the term ‘active SLAM’ (Leung et al., 2006) is usually used to refer to the integration of trajectory planning in SLAM. The scheme described in this chapter does not actuate in the robot trajectory but in measurement gathering. The objective is to dynamically adapt the measurements that are gathered and integrated in SLAM in order to optimize uncertainty reduction-resource consumption trade-offs.

4.3 RO-SLAM scheme with dynamically configurable measurement gathering

Scheme1 meets the requirements described in Section 3.2. The accuracy is reached by the integration of inter-beacon measurements. It benefits from inter-beacon measurements in a dynamic and adaptive way. Figure 3.2 shows its block diagram. Its objective is to improve resource consumption efficiency by employing self-adaptation mechanisms that dynamically modify the number and variety of range measurements that are integrated in the RO-SLAM.

Measurement Gathering is responsible for gathering and collecting direct robot-beacon and also inter-beacon measurements. When a measurement gathering event is triggered, the robot starts a controlled flooding protocol in which each beacon gathers range measurements to its neighbor beacons and transmits the measurements back to the robot as described in Section 3.4.2.

Measurement Gathering can be configured with different inter-beacon depth levels NH, so that it can collect measurements of beacons that are distant from the robot's sensing region. High inter-beacon depth levels allow collecting more measurements and of more distant beacons than with low-depth levels. The higher the inter-beacon depth level, the higher the amount of information that can be integrated in RO-SLAM, reducing uncertainty. In contrast, a higher inter-beacon depth level involves higher resource consumption including the energy used by beacons in measuring and communication, the bandwidth required for transmitting measurements and the computational burden needed to integrate measurements in the SLAM filter. Furthermore, Measurement Gathering can be configured to be performed with different probabilities of taking the measurements, involving a similar trade-off between uncertainty reduction and resource consumption.

Measurement Gathering can be configured in different modes, each of them defined by the measurement gathering probabilities and the inter-beacon depth level. For notation purposes, we will use NH to denote the inter-beacon depth level. The probability of gathering direct robot-beacon measurements is denoted by DMP. IMP is the probability of gathering inter-beacon measurements.

On the other hand the objective of the SLAM Supervisor is to analyze the SLAM performance and select the measurement gathering mode most suitable for the current conditions.

The proposed **Scheme1** is summarized in Algorithm 4.1.

Algorithm 4.1 Algorithm of **Scheme1**, RO-SLAM with dynamically configurable measurement gathering.

```
Require: $\Sigma_{t-1}, \mu_{t-1}, u_t, NH, DMP, IMP$
 1: $[\Sigma_t^-, \mu_t^-] = EKF\_Predict(\Sigma_{t-1}, \mu_{t-1}, u_t)$
 2: $\{z_k\} = MeasurementGathering(NH, DMP, IMP)$
 3: $[\Sigma_t, \mu_t] = EKF\_Update(\Sigma_t^-, \mu_t^-, \{z_t\})$
 4: $[NH, DMP, IMP] = SLAM\_Supervisor(\Sigma_t)$
 5: return $\Sigma_t, \mu_t, NH, DMP, IMP$
```

4.3.1 Measurement Gathering

Measurement Gathering makes use of the protocol described in Section 3.4.2. It collects measurements from every beacon at a hop-distance NH or less. NH is the inter-beacon depth level and allows the collection of inter-beacon measurements from beacons beyond the robot sensing region. NH is a dynamic parameter that can change in any moment.

The difference with the general collection protocol described in Section 3.4.2 is that in the general collection protocol all the involved measurements are collected whereas now the involved measurements will be gathered with a given probability. This probability is assigned by the SLAM Supervisor.

Different probabilities are assigned to direct robot-beacon measurements and to inter-beacon measurements. DMP is the probability of a direct robot-beacon measurement to be gathered. Changing this probability is equivalent to changing the measurement gathering rate. In general DMP will be 1, i.e. we would like to collect every possible robot-beacon measurement. On the other hand, IMP is the probability of gathering an inter-beacon measurement. If $IMP = 0.5$ only one half of the inter-beacon measurements will be gathered and integrated in RO-SLAM. If $DMP = 0.5$ is similar to divide the measurement collection rate by two.

4.4 SLAM Supervisor

The SLAM Supervisor receives as input metrics of the SLAM performance and dynamically selects the most suitable measurement gathering mode for the current conditions. Measurement gathering with high inter-beacon depth levels allows reducing map initialization times and improving of map estimation accuracy. However, once the map has been created, it is not interesting to reduce robot map error if we consider the increment in resource consumption they involve. On the other hand, changing measurement rates allows balancing between robot localization error and measurement and/or odometry noise levels. The following three measurement gathering modes can be clearly identified.

Mapping mode (MM): The objective is to accurately initialize the map as soon as possible. When SLAM starts, the map is assumed unknown, hence rapidly creating an accurate estimation of the map is very important. In fact, SLAM methods for rapid map initialization have attracted significant interest in recent years, as pointed out in Section 4.2. Creating a map requires initializing a good percentage of the available beacons. Measurement gathering with high NH drastically reduces beacon initialization times and improves beacon localization accuracy. In the *Mapping* mode, the measurement gathering is configured with high NH and with the highest possible measurement rate. In this mode every robot-beacon measurement is needed so $DMP = 1$. Inter-beacon measurements will be taken with high probability $IMP = P_{MM}$. The *Mapping* mode should be selected while the ratio between the number of non-initialized beacons and the number of initialized beacons is above a certain threshold.

Localization mode (LM): Its objective is to allow accurate robot self-localization in cases with higher measurement and/or odometry noise levels. The map has been already created, and Measurement Gathering is configured with *NH = 1* but with low probability ($IMP = P_{LM}$) and robot-beacon measurements will be collected at the highest rate ($DMP = 1$), in order to improve robot localization. This mode is suitable in cases where good map and robot accuracies are desired, of course at the expense of having a higher consumption of resources than the *Relaxed* mode. SLAM Supervisor should select the *Localization* mode, while the robot error is above a certain threshold. When the robot error falls below the threshold, the mode is changed to *Relaxed* in order to save resources.

Relaxed mode (RM): In this mode an accurate map has already been created and the robot can self-locate with low error. Thus, Measurement Gathering is configured with *NH = 0* and lower probability for robot-beacon measurements ($DMP = P_{RM}$) in order to save resources. The *Relaxed* mode is suitable with low odometry noise levels; the measurement rate can be reduced without impacting much on the robot location error. It is also suitable when the robot is at densely beacon-populated areas, where measurement gathering events collect a good number of measurements that allow accurate robot localization. On the other hand, the *Relaxed* mode is not suitable with high odometry noise levels, since high measurement rates should be used to compensate for the errors. SLAM Supervisor should select the *Relaxed* mode while the robot error is below a certain threshold. Of course, the ground-truth is assumed not available, and mode selection should be based on robot error estimation, as described below.

Table 4.1 summarizes the parameters of Measurement Gathering in the different modes of **Scheme1**.

	Mapping mode	**Localization mode**	**Relaxing mode**
NH	$NH_{MM} \geq 1$	1	0
DMP	1	1	P_{RM}
IMP	P_{MM} (high)	P_{LM} (low)	0

Table 4.1: Proposed configurations for Measurement Gathering.

The SLAM Supervisor module is implemented by a simple finite state machine with three states, one per each measurement gathering mode, see Figure 4.1. The robot starts at the *Mapping* state and remains at this state until *Cond1* is satisfied, *i.e.*, until the ratio between the number of non-initialized beacons and the number of initialized beacons is below threshold T_1. When the map initialization finishes, the state is changed to *Localization.* Of course, at any time, the robot can move to an unexplored area where new beacons are discovered. If *Cond2* is satisfied, *i.e.*, the ratio between the number of non-initialized beacons and the number of initialized beacons is above threshold T_2, the state machine changes to *Mapping*, regardless of its previous state. *Cond1* and *Cond2* are expressed as:

$$Cond1: \qquad \frac{\#\text{non-initialized beacons}}{\#\text{initialized beacons}} < T_1 \tag{4.1}$$

$$Cond2: \qquad \frac{\#\text{non-initialized beacons}}{\#\text{initialized beacons}} > T_2 \tag{4.2}$$

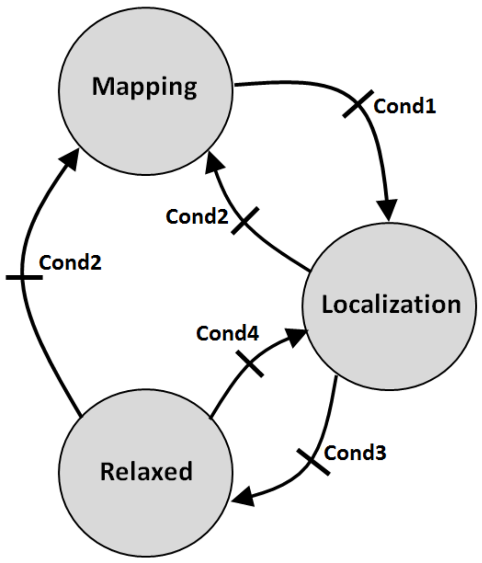


Figure 4.1: Diagram of the finite state machine implemented in the SLAM Supervisor.

States *Localization* and *Relaxed* operate similarly. *Localization* tries to maintain the highest accuracy for robot location and gathers inter-beacon measurements with low probability. On the other hand, *Relaxed* does not take any inter-beacon measurement. That makes it suitable only in cases with very good odometry. The state machine stays at *Relaxed*, saving resources until *Cond4* is satisfied, *i.e.*, until the robot error is above threshold T_4. In that case, the state changes to *Localization*, which involves a higher measurement rate in order to reduce the robot error. The state machine stays at *Localization* until the robot error is below T_3 (*Cond3*). When *Cond3* is satisfied, it changes to *Relaxed* to save resources.

Cond3 and *Cond4* use estimations of the robot location error. A wide variety of metrics have been used to measure the uncertainty in a statistical distribution. Some are based on the covariance matrix, while others are based on the information matrix. EKF SLAM filters use the moment representation of Gaussian distributions; hence, covariance-based metrics are more convenient. A number of metrics based on the determinant, maximum eigenvalues and trace of the covariance matrix have been developed. They were compared in (Wenhardt et al., 2007) concluding that all of them perform properly.

In **Scheme1** the selected metric is the trace of the robot location covariance matrix. Due to the marginalization properties of the moment representation of Gaussian distributions, it is straightforward to extract the robot location covariance matrix from the EKF SLAM covariance matrix, Σ_t:

$$\Sigma_{t,robot} = \begin{bmatrix} \Sigma_{t,11} & \Sigma_{t,12} & \Sigma_{t,13} \\ \Sigma_{t,21} & \Sigma_{t,22} & \Sigma_{t,23} \\ \Sigma_{t,31} & \Sigma_{t,32} & \Sigma_{t,33} \end{bmatrix}, \tag{4.3}$$

where $\Sigma_{t,ij}$ is the component in row i and column j of Σ_t. Therefore, *Cond3* and *Cond4* are expressed as:

$$Cond3: \qquad tr(\Sigma_{t,robot}) < T_3 \tag{4.4}$$

$$Cond4: \qquad tr(\Sigma_{t,robot}) > T_4 \tag{4.5}$$

A hysteresis H is added between T_3 and T_4 in order to prevent ping-pong effects between states, $T_4 = T_3 + H$. The selection of settings and parameters of the SLAM Supervisor used in the validation experiments can be found in the next section.

4.5 Evaluation

This section is divided in two parts. The first one is a proof of concept that will help to fully understand the operation of **Scheme1**. The second one evaluates the performance of the scheme.

4.5.1 Proof of concept

Figure 4.2 shows the result of executing **Scheme1** in one experiment performed in the CONET Integrated Testbed, which description is summarized in Appendix A. Figure 4.3 details its operation during the execution of the experiment. It shows: (top) the values of the smoothed robot trace used as robot error estimation; (center) the measurement gathering state provided by the SLAM Supervisor; and (bottom) the robot and map location errors at each time.

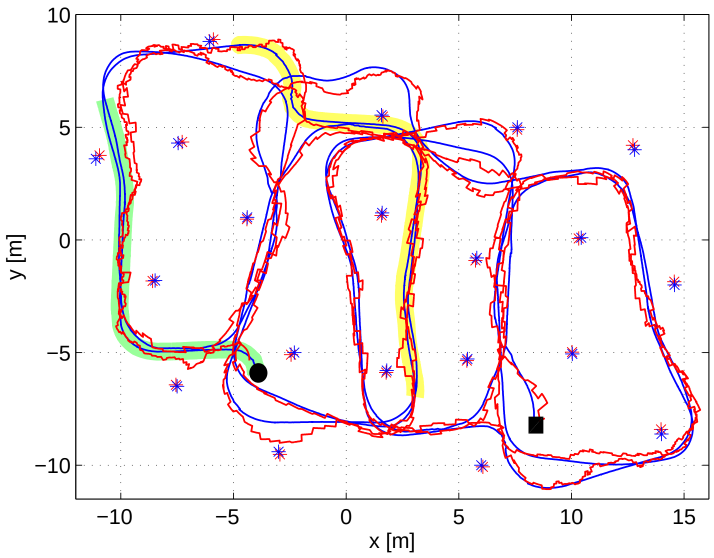


Figure 4.2: Results of **Scheme1** in one experiment. The ground-truth locations of the robot and beacons are in blue, while the resulting estimations are in red. The robot initial and final locations are marked in the figure with a black circle and a black rectangle, respectively.

It starts at $t = 0$ s at state *Mapping* and remains at that state (see Figure 4.3) until the PFs of all the discovered beacons converge, which occurred at $t = 55.6$ s.

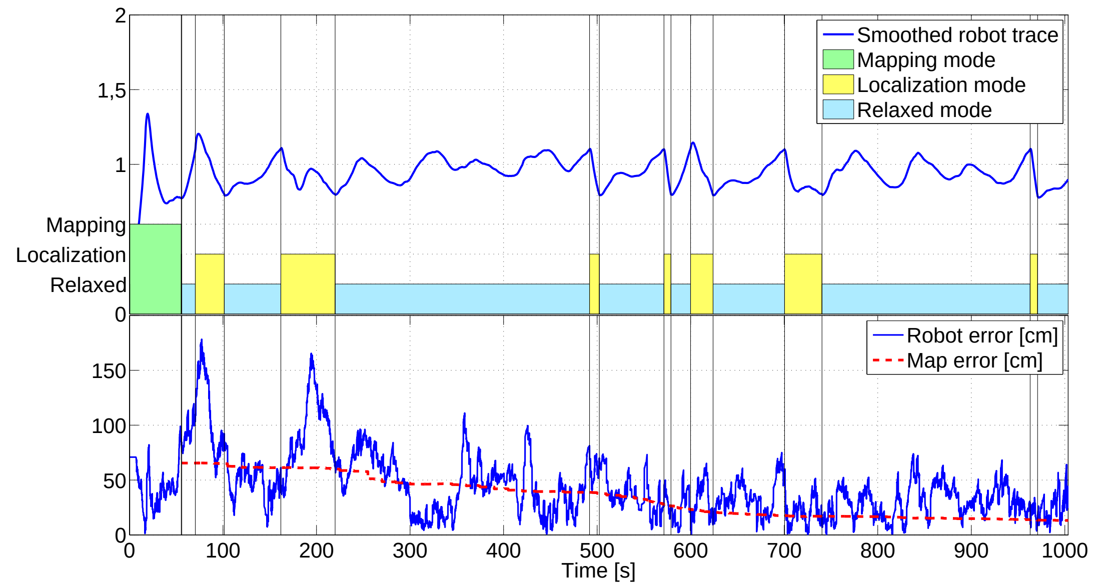


Figure 4.3: Details of the operation of **Scheme1** along the experiment: (**top**) values of the smoothed robot trace; (**center**) the measurement gathering state; (**bottom**) robot and map location errors.

Figure 4.2 shows in green the path followed by the robot while **Scheme1** is at the *Mapping* state. Due to the sensor sensing range, the robot was capable of integrating direct robot-beacon measurements only from 55% of the 20 beacons deployed in the scenario. The Particles Filters of the beacons distant from the robot's sensing range (45% of the beacons) converged using only inter-beacon measurements. As expected, the *Mapping* state, with $NH = 2$ and $IMP = 1$, rapidly created an accurate map. At $t = 55.6$ s, the last PF converged; the map was initialized with a map error of 65.5 cm, and the state machine changed to *Localization*. From that time on, the map error is represented in Figure 4.3.

After the *Mapping* state, the smoothed robot trace was very low (meaning accurate estimation), fulfilling *Cond3*, and thus, when the last PF converged, the state machine changed to *Localization* and immediately changed again to *Relaxed*.

The state machine remained at *Relaxed* until $t = 70.4$ s, in which the smoothed robot trace grew above T_4, satisfying *Cond4*: the state machine changed to *Localization*. The robot trace grew, because the beacons that surrounded the robot at those times had significantly high errors, higher than the mean map error. Integrating their measurements increased the robot error. After changing to the *Localization* state, the increase in the number of measurements integrated in SLAM –originated by changing the measurement rate from $DMP = 0.33$ (*Relaxed*) to $DMP = 1$ (*Localization*)– rapidly decreased the smoothed robot trace. At $t = 101$ s, it fell below T_3, fulfilling *Cond3*, and the state changed again to *Relaxed*.

The *Relaxed* state remained until $t = 162$ s. The robot motion had two consecutive curves, which suddenly degraded the robot odometry. This degradation in odometry originated a sudden increase in the robot trace. At $t = 162$ s, the smoothed robot trace became above T_4, fulfilling *Cond4*, and the state machine changed to *Localization*. With a higher measurement gathering rate, the robot location uncertainty improved; and at $t = 219.8$ s, the smoothed robot trace fell below T_3, and the state changed again to *Relaxed*.

In the rest of the experiment the smoothed robot trace increased and reduced as the robot followed its path. The state was changed when necessary reacting and adapting to the changing conditions. At the end of the experiment, the robot and map errors w.r.t. the ground-truth were, respectively, 32.4 cm and 14.3 cm, enough for a wide variety of applications. In the full experiment, the state machine was at the *Mapping* state during 7.60% of the total time, at *Localization* during a total of 17.24% and at *Relaxed* during a total of 75.16%. It was at the *Relaxed* state during the greater majority of the time involving the low consumption of resources. The same experiment was repeated, obtaining similar results. A video illustrating another experiment can be seen at `http://youtu.be/GxnPV96fnBM`.

4.5.2 Performance analysis

In this section **Scheme1** is compared to the following RO-SLAM methods. *Method1* is the conventional EKF SLAM method. During the whole experiment, it integrates only direct robot-beacon measurements ($NH = 0$) taken with $DMP = 1$. *Method2* is a non-adaptable method that exploits the advantages of integrating inter-beacon measurements. It configures measurement gathering with $NH = 2$ and $P_{MM} = 1$ during the whole experiment. *Method3* is an adaptable RO-SLAM scheme that configures measurement gathering with $NH = 2$ and $P_{MM} = 1$ during map initialization and, with $NH = 0$ and $P_{RM} = 0.33$, once the map has been initialized. Tables 4.2 and 4.3 summarize their configuration.

	During Mapping	**After Mapping**
Method1	Localization mode ($NH = 0$)	Localization mode ($NH = 0$)
Method2	Mapping mode	Mapping mode
Method3	Mapping mode	Relaxed mode
Scheme1	Mapping mode	Localization/Relaxed modes

Table 4.2: Description of the methods' configuration.

Parameter	**Value**
NH_{MM}	2
P_{MM}	1
P_{LM}	0.005
P_{RM}	0.33

Table 4.3: Parameters for simulation.

It should be noticed that *Method1*, *Method2* and *Method3* could be very easily implemented by **Scheme1** simply by changing the state transitions parameters in the SLAM Supervisor. For instance, *Method1* could be implemented by enforcing that the finite machine is always at state *Localization*. Furthermore, *Method3* is a particular case of **Scheme1** with only states *Mapping* and *Relaxed*. From this point of view, **Scheme1** generalizes existing techniques, adding flexibility and reactivity.

All –**Scheme1**, *Method1*, *Method2* and *Method3*– were executed with each experiment in the validation data set with exactly the same parameters. All the experiments were performed in the CONET Integrated Testbed. Figure 4.4 shows the mean beacon location errors (top) and mean robot location errors (bottom) in 50 experiments. Figure 4.5 shows the mean beacon initialization times (top) and the computation times required (bottom) in each of the 50 experiments.

Method2, which uses $NH = 2$ and $IMP = 1$ during the whole experiment, obtained the lowest map and robot mean errors. *Method1*, which uses $NH = 0$ and $DMP = 1$ during the whole experiment, obtained the worst map and robot errors. *Method3* and

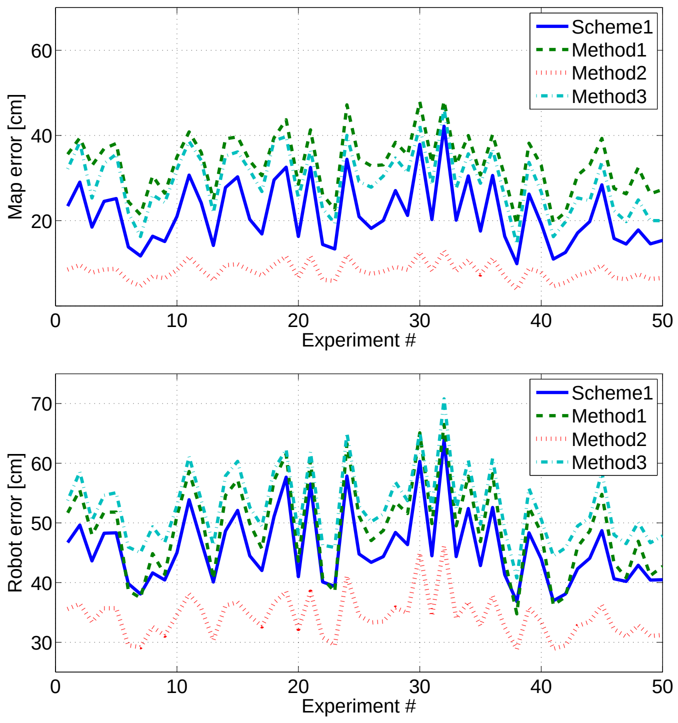


Figure 4.4: Mean beacon location errors (**top**) and mean robot location errors (**bottom**) in 50 experiments performed with *Method1*, *Method2*, *Method3* and the proposed **Scheme1**.

the proposed method obtained intermediate map and robot errors. The errors in the proposed method were lower than in *Method3*. Besides the *Mapping* state, which is the same for both, the proposed method has two states, *Localization* and *Relaxed*, that allow flexibility to balance between accuracy and burden, while *Method3* only has one state. The *Localization* state enables the proposed method to reduce errors when necessary, involving very low extra costs.

Method2, *Method3* and the proposed method had similar mean beacon initialization times, since they all operate with $NH = 2$ and $IMP = 1$ during map initialization. In contrast, *Method1* uses $NH = 0$ obtained beacon initialization three times higher. The robot computational burden of *Method2* was three times higher than that of *Method1*, *Method3* or the proposed method. It should be noticed that both, *Method3* and **Scheme1**, require lower robot burden than *Method1*, the traditional method.

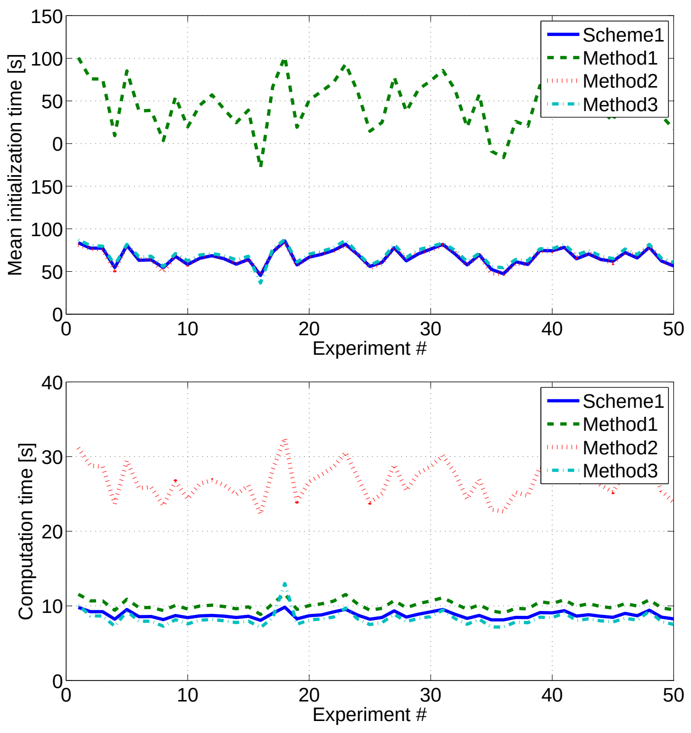


Figure 4.5: Mean beacon initialization times (**top**) and computational times (**bottom**) required in each of the 50 experiments performed with *Method1*, *Method2*, *Method3* and the proposed **Scheme1**.

Although both consume more resources during map initialization, they rapidly obtain an accurate map estimation that allows them to be at the *Relaxed* state, saving resources, during large periods of the experiment. As a result, in total, *Method3* and **Scheme1** obtain lower robot and map errors with lower burden than *Method1*. Furthermore, the computational burden of the proposed method is a bit higher (6.5%) than that of *Method3*, since **Scheme1** goes in the *Localization* mode from time to time in order to the reduce robot error, while *Method3* keeps at the *Relaxed* mode during all the time after map initialization.

Method2 obtained the lowest map and robot mean errors, but required a very high robot computational burden. The best trade-offs between performance and burden were *Method3* and the proposed **Scheme1**. Both overtook *Method1*. In particular, the proposed method had lower map (34%) and robot error (14%) than *Method1*, requiring also lower computational burden (16%). When comparing to *Method3*, the proposed method had lower map and robot error than *Method3* (19% and 31%, respectively) requiring only a bit higher computational burden (6.5%).

The following analyzes the energy consumed by beacons. **Scheme1** is also efficient in terms of the energy consumed by the beacons. We obtained a consumption model using the information provided by the manufacturer (see Appendix B). The radio module is responsible for the greater part of the consumption. Each measurement takes 12 ms, during which, the emitter and the receiver interchange the measurement request and response packets. Both beacons transmit during 6 ms (consuming 210 mA) and receive during 6 ms (consuming 51 mA). The consumption of both beacons is the same and depends on the number of measurements in which they are involved. Table 4.4 shows the mean number of measurements in which beacons are involved in the 50 experiments analyzed. It also shows the mean beacon energy consumptions in these experiments. As expected, *Method2* consumed 360% more than the rest. The consumption of **Scheme1** was a bit higher than *Method1* (3.9%) and than *Method3* (1.7%). However, these small differences are compensated by the significant improvements in accuracy.

	Scheme1	*Method1*	*Method2*	*Method3*
# of measurements involving a beacon	1830	1760	8300	1800
Mean beacon energy consumption (J)	11.53	11.09	52.29	11.34

Table 4.4: Comparison of beacon energy consumption of *Method1*, *Method2*, *Method3* and the proposed **Scheme1** during the experiments.

4.6 Conclusions

This chapter presented **Scheme1**, **Contribution2** of this PhD Thesis and the first of the RO-SLAM schemes that adopts the architecture presented in Section 3.3. The objective of this scheme is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a mechanism that dynamically modifies measurement gathering. Measurement Gathering implements a protocol by means of which it can take and collect at configurable rates direct robot-beacon and inter-beacon range measurements of configurable depth levels. It also includes a Supervisor that monitors the SLAM performance and dynamically selects the configuration of Measurement Gathering.

Scheme1 exploits the capability of beacons of taking range measurements and transmitting them using *ad hoc* protocols. Besides, it exploits the flexibility of dynamically modifying the measurement gathering rate and inter-beacon depth level, allowing its adaptation to changing conditions.

SLAM Supervisor implements a simple finite state machine with three states, *Mapping*, *Localization* and *Relaxed*, each of them with different inter-beacon depth levels and measurement collection probabilities. The SLAM performance metrics selected are the percentage of discovered, but not initialized, beacons and the estimation of the robot error using the trace of the robot location covariance matrix.

The proposed **Scheme1** and its modules are general and can be applied to any SLAM filter. In this chapter it has been applied to a PF-EKF SLAM filter, which is maybe one of the most common RO-SLAM methods. It has been validated in simulations and experiments performed in the CONET Integrated Testbed, and its performance has been compared to the traditional methods based on integrating only direct robot-beacon measurements. The experiments show how SLAM Supervisor dynamically changes the state, changing the configuration of measurement gathering, impacting on which measurements are integrated in the SLAM filter. Its evaluation evidences the advantages of its adaptation and configuration capabilities. In these experiments it obtained map and robot location errors significantly lower (e.g., 34% and 16%, respectively) than traditional RO-SLAM techniques requiring 16% lower computational burden and similar beacon energy consumption. Besides, it exhibited better robustness against measurement and odometry noise levels.

Scheme1 allows high flexibility, being able to implement a high number of methods by simply modifying the parameters of the SLAM Supervisor. Thus, it can be easily adapted to particular problems or applications.

Chapter 5

Distributed Sparse Extended Information Filter for RO-SLAM

5.1 Introduction

Scheme1, which was presented in Chapter 4, reduces the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a mechanism that dynamically adapts measurement gathering. This chapter presents a RO-SLAM scheme that improves resource consumption efficiency by distributing the computation of RO-SLAM between the robot and the different beacons in the surroundings of the robot. It uses a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate them in SLAM. This chapter describes **Scheme2**, **Contribution3** of this PhD Thesis and the second of the RO-SLAM schemes that adopts the architecture presented in Section 3.3.

Scheme2 is based on Sparse Extended Information Filter (SEIF) (Thrun et al., 2004), and inherits its efficiency and scalability. Its distributed approach shares resource consumption, reducing robot CPU burden, and at the same time naturally integrates inter-beacon measurements, which improves map and robot localization accuracies.

It is proven in the chapter that the proposed scheme accumulates higher amount of information than the conventional SEIF SLAM and at the same time, it preserves the sparsity of the information matrix and its constant time property.

The chapter is divided in the following sections. Section 5.2 presents specific related work relative to the canonical form of the Gaussian distribution and its advantages on efficiency and scalability. It also summarizes the SEIF SLAM algorithm, which is the base of the proposed **Scheme2**. Section 5.3 describes the proposed **Scheme2**. Section 5.4 analyzes the building of the information matrix and proofs that **Scheme2** preserves the main properties of SEIF SLAM. Section 5.5 validates and evaluates **Scheme2**. The chapter ends with the conclusions in Section 5.6.

5.2 Related work

Different strategies have been developed in the past years to improve efficiency and scalability in SLAM. The majority of them use the canonical form of the Gaussian distribution. Thin Junction Tree Filters (Paskin, 2003) employ an special data structure –called junction tree– to represent the information matrix. Work (Paskin, 2003) proposes approximations to keep these junction trees simple –thin– regardless of the map size. Treemap Filters (Frese, 2005c) hierarchically divide the map into local regions.

Dual to Kalman Filters, Information Filters (IFs) represent the state by its canonical form based on the information vector and the information matrix. In Extended Information Filter (EIF) SLAM the update stage is very efficient –additive– but the prediction stage requires operations with the whole information matrix, involving bad scalability with the map size. Besides, EIF is a full SLAM solution, i.e. it solves the SLAM problem after all information has been gathered, not being suitable for on-line problems.

Sparse Extended Information Filter (SEIF) (Thrun et al., 2004; Liu and Thrun, 2003) solves on-line SLAM maintaining a sparse representation of the information matrix, which simplifies the matrix operations, improving efficiency and scalability. It has been demonstrated in (Thrun et al., 2004; Frese, 2005b) that many of the off-diagonal elements of the information matrix are relatively close to zero. SEIF SLAM enforces the sparsity of the information matrix, enabling efficient algorithms for motion update and state recovery. As a result, it can be executed in constant time regardless of the map size. SEIF for feature-based SLAM has been researched in some works. In (Eustice et al., 2005) a modification for ensuring the consistency of the global map estimate was proposed. SEIF SLAM for multi-robot systems was used in (Thrun and Liu, 2005).

In this chapter we propose an efficient robot-beacon distributed SEIF scheme where beacons actively participate in gathering and integrating measurements in SLAM, sharing resource consumption and ensuring constant time execution. Work (Djugash et al., 2008) presented a decentralized method to self-localize a sensor network with the help of a mobile robot. It is based on Loopy Belief Propagation (LBP) (Frey and MacKay, 1998), hence it requires the sensor network to be sparse –not fully connected– which is an important limitation in many applications. In case of having loops in the network some information can be counted twice, making it more difficult to accurately recover the exact state representation.

5.2.1 Range-only SEIF SLAM in a nutshell

This section briefly summarizes the SEIF SLAM algorithm as an introduction to the proposed **Scheme2**. For brevity, most expressions have been omitted. For notation simplicity, time subindex t has been also omitted. Refer to (Thrun et al., 2005) for further details. The adopted state vector x is comprised of the robot position (x_r) and the location of all the beacons in the map ($x_1, x_2, \dots, x_N$). SEIF SLAM is based on Extended Information Filter (EIF). Dual to EKF, EIF represents the state vector by the information vector $\xi = \Sigma^{-1}\mu$ and the information matrix $\Omega = \Sigma^{-1}$, where μ is the mean of the state vector and Σ is the covariance matrix. Ω, see (5.1), is symmetric and positive-semidefinite. Each off-diagonal entry of Ω –called *link* (Thrun et al., 2004)– represents the relation between two elements in x.

$$\Omega = \begin{pmatrix} \Omega_{r,r} & \Omega_{r,1} & \cdots & \Omega_{r,N} \\ \Omega_{1,r} & \Omega_{1,1} & \cdots & \Omega_{1,N} \\ \vdots & \vdots & \ddots & \vdots \\ \Omega_{N,r} & \Omega_{N,1} & \cdots & \Omega_{N,N} \end{pmatrix} \tag{5.1}$$

At any time in the SLAM operation some of the off-diagonal elements of Ω are zero meaning lack of information between the involved elements; some of them have high values –strong links– meaning high information; and a number of them have values close to zero –weak links. Weak links have much lower impact on the estimation than strong links but both involve similar computational burden. SEIF maintains a sparse representation of Ω by keeping the number of active beacons –beacons with non-zero links to the robot– bounded by a threshold. Every time the number of active beacons is above the bound, the sparsification step is performed deactivating the beacons with the weakest links.

The Update stage in information filters modifies only the entries of Ω corresponding to the elements involved in the measurement. Factorizing Ω allows efficient Update stage regardless of the map size. Also, the sparsity of Ω significantly reduces the computational burden required in SEIF for the Prediction stage. For linearizing the prediction and observation models it is required to recover the state estimate μ from the predicted $\bar{\Omega}$ and $\bar{\xi}$. The whole state is not needed, only the states of the robot and active beacons are required. Of course, enforcing sparseness in Ω involves an approximation error in the estimations obtained by SEIF. Work Thrun et al. (2005) suggests using sparsification bounds in the range $[4-10]$ in order to balance accuracy degradation and burden reductions.

The observation model we adopted is the Euclidean distance between the source and the destiny of the measurement –see Eqs. (2.19) and (2.20). For each measurement the predicted observations $h_{r,i}(\mu)$ and Jacobian $H_{r,i}$ are computed and used in the measurement Update stage. The information provided by each measurement is added to the predicted information matrix and information vector:

$$\xi = \bar{\xi} + \sum_j H_{r,i}^T R^{-1}[z_{r,i} - h_{r,i}(\mu) + H_{r,i}\mu] \tag{5.2}$$

$$\Omega = \bar{\Omega} + \sum_j H_{r,i}^T R^{-1} H_{r,i} \tag{5.3}$$

5.3 Robot-sensor network distributed SEIF SLAM

In conventional RO-SLAM techniques the robot gathers range measurements to the beacons within its sensing region and integrates these measurements in the Update stage. In **Scheme2**, measurement gathering and integration in SLAM is distributed among the beacons. The beacons involved at time t can be classified into two sets: direct beacons and indirect beacons. Direct beacons are those that at time t are within the robot sensing region (SR_r). The set of direct beacons at time t is:

$$DBS_t = \{b_i : b_i \in SR_r\} \tag{5.4}$$

Indirect beacons are those that are outside SR_r but within the sensing region of a direct beacon. The set of indirect beacons at t is:

$$IBS_t = \{b_j : b_j \notin SR_r,\ b_j \in SR_i\ \ \forall b_i \in SR_r\}, \tag{5.5}$$

where SR_i is the sensing region of beacon b_i.

The operation of **Scheme2** is as follows. The RO-SLAM Prediction stage is executed in the robot. The robot keeps LNB, the list with the direct beacons it has detected. Then, the robot broadcasts an *UpdateReq* message that includes LNB and the predicted state. Each direct beacon b_i receives the message and extracts LNB. If b_i finds itself in LNB, it gathers a range measurement to the robot ($z_{i,r}$) and to each of the beacons within its sensing region SR_i. This measurement set is designed as $MS_i = \{z_{i,r}, z_{i,j}\ \ \forall b_j \in SR_i\}$. Then, each direct beacon b_i computes with the measurements in MS_i its contribution to the Update stage and transmits it to the robot in an *UpdateResp* message.

The Update stage in SLAM represented in the information form is additive. Thus, the robot reconstructs the SLAM updated state by adding the contributions it received. Figure 5.1 summarizes the main tasks and messages interchanged in **Scheme2**.

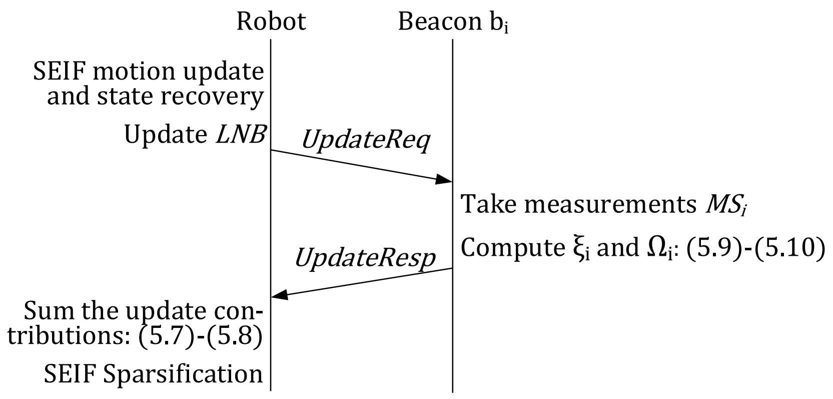

Robot
Beacon b_i
SEIF motion update and state recovery
Update LNB
UpdateReq
Take measurements MS_i
Compute ξ_i and Ω_i: (5.9)-(5.10)
UpdateResp
Sum the update contributions: (5.7)-(5.8)
SEIF Sparsification

Figure 5.1: Main tasks and messages interchanged between the robot and beacon b_i in **Scheme2**.

Measurement gathering in **Scheme2** is illustrated in Fig. 5.2. The red and gray circumferences represent the sensing regions of the robot and beacons, respectively. Direct beacons are represented in black color and, indirect in gray: $DBS_t = \{b_1, b_5, b_6\}$ and $IBS_t = \{b_2, b_3\}$. In conventional RO-SLAM the robot gathers and integrates measurements to direct beacons, $\{z_{r,1},\ z_{r,5}, z_{r,6}\}$ in Fig. 5.2. In **Scheme2** each direct beacon b_i gathers and integrates the measurements in its MS_i, e.g. b_5 gathers $MS_5 = \{z_{5,r},\ z_{5,1}, z_{5,6}\}$. Thus, when the robot completes the SLAM Update stage it has integrated measurements $\{z_{1,r}, z_{1,2}, z_{1,3}, z_{1,5}, z_{5,r}, z_{5,1}, z_{5,6}, z_{6,r}, z_{6,5}\}$.

Scheme2 integrates all robot-beacon and inter-beacon measurements that involve one or more direct beacons, naturally avoiding repeated measurements. It gathers one measurement for all these distances except for those between two direct beacons, e.g. b_1 and b_5. In these cases two different measurements are gathered, e.g. $z_{1,5}$ and $z_{5,1}$. The method is also robust to message loss, as is shown in Section 5.5.

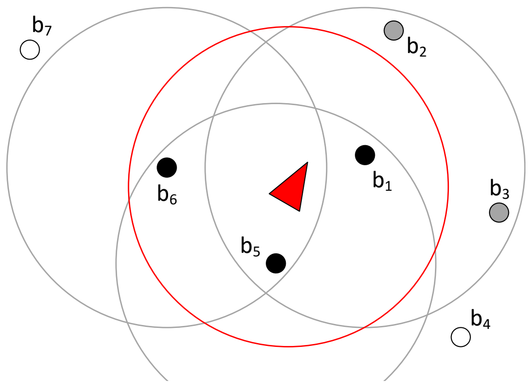


Figure 5.2: Measurement gathering in a conventional RO-SLAM and in **Scheme2**. The red triangle represents the robot. Direct beacons are represented with black circles and indirect, with gray circles. Circumferences represent the sensing regions of the robot (red) and of beacons (gray).

5.3.1 Operation of the robot

The operation of the robot in one prediction-update cycle is summarized in Alg. 5.1. First, the robot computes the SLAM Prediction stage. We assume static beacons and a robot with nonlinear kinematic model. Thus, its Jacobian should be computed at each time, which requires recovering μ. **Scheme2** uses the efficient algorithm described in (Thrun et al., 2005) for motion update and state recovery (step 1 in Alg. 5.1). This algorithm computes the predicted information vector $\bar{\xi}$, the information matrix $\bar{\Omega}$ and also the recovered predicted estimate μ.

Algorithm 5.1 Summary of the operation of the robot. ξ and Ω are the information vector and matrix. *LNB* is the list of the direct beacons detected by the robot.

```
Require: $\xi_{t-1}, \Omega_{t-1}$
1: SEIF SLAM motion update and state recovery
2: Update *LNB* and create *UpdateReq* message
3: Broadcast *UpdateReq* message
4: Receive *UpdateResp* messages from beacons
5: Sum SLAM Update contributions to $\bar{\xi}$ and $\bar{\Omega}$ as in (5.7) and (5.8)
6: SEIF SLAM Sparsification
7: return $\xi_t, \Omega_t$
```

Next, the robot broadcasts an *UpdateReq* message that includes the predicted estimate μ. Transmitting the whole μ is not suitable in cases with large numbers of beacons: it would require broadcasting large –or several– messages, increasing message losses. Besides, the greater part of μ is not of interest for direct beacons. Only the elements in μ required for each direct beacon in *LNB* are transmitted. Each direct beacon b_i gathers range measurements to the robot and to the beacons $b_j \in SR_i$. For integrating them it needs μ_i, μ_r and μ_j $\forall b_j \in SR_i$. ev_i is the vector with the estimates required for direct beacon b_i to perform its SLAM Update stage:

$$ev_i = \begin{bmatrix} \mu_r & \mu_i & \mu_j \end{bmatrix}^T \tag{5.6}$$

At the beginning *LNB* is assumed empty. If a direct beacon that received the *UpdateReq* message does not find itself in *LNB*, it sends a *BeaconDiscovery* message to the robot with its ID and the IDs of the beacons that are within its sensing region. Then, the robot will add it to *LNB*. When the robot does not receive update contributions from beacon b_i within *LNB* in a number of consecutive times, it is interpreted that b_i is currently outside SR_r and is deleted from *LNB*.

UpdateResp message from b_i contains its contribution to the SLAM Update stage (ξ_i and Ω_i). The robot receives the *UpdateResp* messages and when a timeout expires, it reconstructs the updated state (ξ and Ω) adding the predicted $\bar{\xi}$ and $\bar{\Omega}$ to the contributions it received:

$$\xi = \bar{\xi} + \sum_i F_i^T \xi_i, \tag{5.7}$$

$$\Omega = \bar{\Omega} + \sum_i F_i^T \Omega_i F_i, \tag{5.8}$$

where F_i is the projection matrix that implements operations that allocate ξ_i and Ω_i at the correct entries for ξ and Ω.

Finally, the robot checks if the updated Ω satisfies the sparsification bound θ_x. If not, the active beacons with the strongest links are selected and the weakest links are deactivated. Note that measurements from both active and non-active beacons are integrated in the Update stage. Integrating measurements from a non-active beacon gives this beacon the possibility of being selected in the sparsification step.

5.3.2 Operation of beacons

The operation of the beacons is summarized in Alg. 5.2. Each beacon b_i that received an *UpdateReq* message first measures the range to the beacons in its sensing range –its measurement set is MS_i. The operation of b_i is different if it is at the "initialization phase" or at the "state vector phase". If b_i is at the "state vector phase", it computes its SLAM Update contribution integrating the measurements in MS_i as follows:

$$\xi_i = \sum_{k \in MS_i} H_{i,k}^T R^{-1} [z_{i,k} - h_{i,k}(ev_i) + H_{i,k} ev_i] \tag{5.9}$$

$$\Omega_i = \sum_{k \in MS_i} H_{i,k}^T R^{-1} H_{i,k} \tag{5.10}$$

where $h_{i,k}(ev_i)$ and $H_{i,k}$ are the predictions and Jacobians for each measurement in MS_i either if it is a robot-beacon measurement or an inter-beacon measurement. R is the covariance matrix of the measurement noise.

Algorithm 5.2 Summary of the operation of beacon b_i. ξ_i and Ω_i are the contributions of beacon b_i to the information vector and matrix. *LNB* is the list of the direct beacons detected by the robot. MS_i is the set of measurements gathered by beacon b_i.

```
1: Receive UpdateReq message. Extract LNB
2: if ($b_i$ is present in LNB) then
3:    Take measurements $MS_i$
4:    if ($b_i$ is at "state vector phase") then
5:       Compute $\xi_i$ and $\Omega_i$ with $MS_i$ as in (5.9) and (5.10)
6:       Send $\xi_i$ and $\Omega_i$ to robot in an UpdateResp message
7:    else
8:       Send $MS_i$ to robot in an UpdateResp message
9:    end if
10: else
11:    Send to robot a BeaconDiscovery message
12: end if
```

Next, each beacon b_i transmits the resulting ξ_i and Ω_i to the robot in an *UpdateResp* message, as in step 6 in Alg. 5.2.

If b_i is the "initialization phase", the measurements in MS_i cannot be used for the update of the SLAM state vector: they are used for the initialization of beacon b_i. Two versions of **Scheme2** were developed:

- *Scheme2v1*: The initialization of each beacon b_i is computed by the robot. In this case b_i transmits to the robot an *UpdateResp* message containing MS_i. The robot will use MS_i for the initialization of b_i. When beacon initialization is finished, the robot computes the estimation of the location of b_i and adds it to the SLAM state vector x.

- *Scheme2v2*: Each beacon computes its own initialization: both SLAM Update and beacon initialization are decentralized. b_i integrates MS_i in its own initialization tool. When the beacon initialization is finished, b_i sends the robot its estimation in an *UpdateResp* message so that it adds the initialized beacon to x.

The selection between *Scheme2v1* or *Scheme2v2* depends on the beacon computational capabilities and on the initialization tool. For instance, with beacons implemented using wireless sensor networks (WSN) technology, *Scheme2v2* is suitable in case of using trilateration initialization but would require *Scheme2v1* if using PFs. Beacons implemented with embedded PCs (*RaspberryPi* as in the experiments), smatphones or PDAs can execute *Scheme2v2* using PFs as initialization tools.

5.4 Analysis of Scheme2

If we analyze the building of the information matrix in **Scheme2** we can obtain the following conclusions:

- The integration of inter-beacon measurements is responsible for obtaining higher values of Ω, and hence lower robot localization and map uncertainties, than schemes that integrate only robot-beacon measurements. We believe that this conclusion is clear and for brevity it is not proven.

- **Scheme2** preserves the sparsity of the information matrix of the conventional SEIF SLAM. Under common assumptions, both create the same links of the information matrix. This will be proven in Section 5.4.2.
- **Scheme2** preserves the constant time property of SEIF SLAM, which integrates only robot-beacon measurements. It is clear that the Prediction stage and the distributed Update stage of **Scheme2** are constant time. In Section 5.4.3 it will be proven that state recovery in **Scheme2** is also constant time.

Finally, in Section 5.4.4 the assumptions of the demonstrations used in Sections 5.4.2 and 5.4.3 are relaxed and the above conclusions are confirmed using simulations. Next, the building of the information matrix and links is described.

5.4.1 Building of the information matrix

The information matrix and its structure was briefly presented in Section 5.2.1. Each off-diagonal entry of the information matrix constraints two elements in the state vector –each on-diagonal entry constraints one. The off-diagonal entry $\Omega_{r,i}$ links together the estimations of the robot pose and of the location of beacon b_i. Entry $\Omega_{i,j}$ $\forall i \neq j$ links together the localization estimations of b_i and b_j.

Two types of links can be distinguished: robot-beacon links and inter-beacon links. A robot-beacon link relates the robot with a beacon. They are created or reinforced (the value of the link is incremented) when a robot-beacon measurement is integrated in the SLAM Update stage. Integrating $z_{r,j}$ affects $\Omega_{r,r}$, $\Omega_{j,j}$ and the robot-beacon links between the robot and b_j, i.e. $\Omega_{r,j}$ and its symmetric $\Omega_{j,r}$. Inter-beacon links relate two beacons. They are created or reinforced when an inter-beacon measurement is integrated. Integrating $z_{i,j}$ affects $\Omega_{i,i}$, $\Omega_{j,j}$ and inter-beacon links $\Omega_{i,j}$ and $\Omega_{j,i}$. Inter-beacon links are created/reinforced also when integrating the robot motion in the SLAM Prediction stage.

Figure 5.3 illustrates how the information matrix is built in the conventional SEIF SLAM (a-c) and in **Scheme2** (d-f) in a simple example. We assume that b_1 and b_2 are currently direct beacons (within SR_r) and that b_1 and b_2 are within the sensing region one another. b_3 is within the sensing region of b_2. We assume that initially $\Omega_{r,1}$=$\Omega_{r,2}$=

$\Omega_{1,2}$=0. First, we describe the operation of the conventional SEIF SLAM. The SEIF SLAM Update stage integrates measurements $z_{r,1}$ and $z_{r,2}$, which creates robot-beacon links $\Omega_{r,1}$ and $\Omega_{r,2}$ –and their symmetric links, see Fig. 5.3-a. Next, the robot motion in the SEIF SLAM Prediction stage transfers some low amount of information from the robot-beacon links –$\Omega_{r,1}$ and $\Omega_{r,2}$ in Fig. 5.3-b– to the inter-beacon links between the involved beacons –$\Omega_{1,2}$. Link $\Omega_{1,2}$ is created. At that time, $\Omega_{r,1}$ and $\Omega_{r,2}$ are strong (have high value) whereas $\Omega_{1,2}$ is weak. Next, the sparsification step in this example deactivates b_1 and hence, removes link $\Omega_{r,1}$, see Fig. 5.3-c.

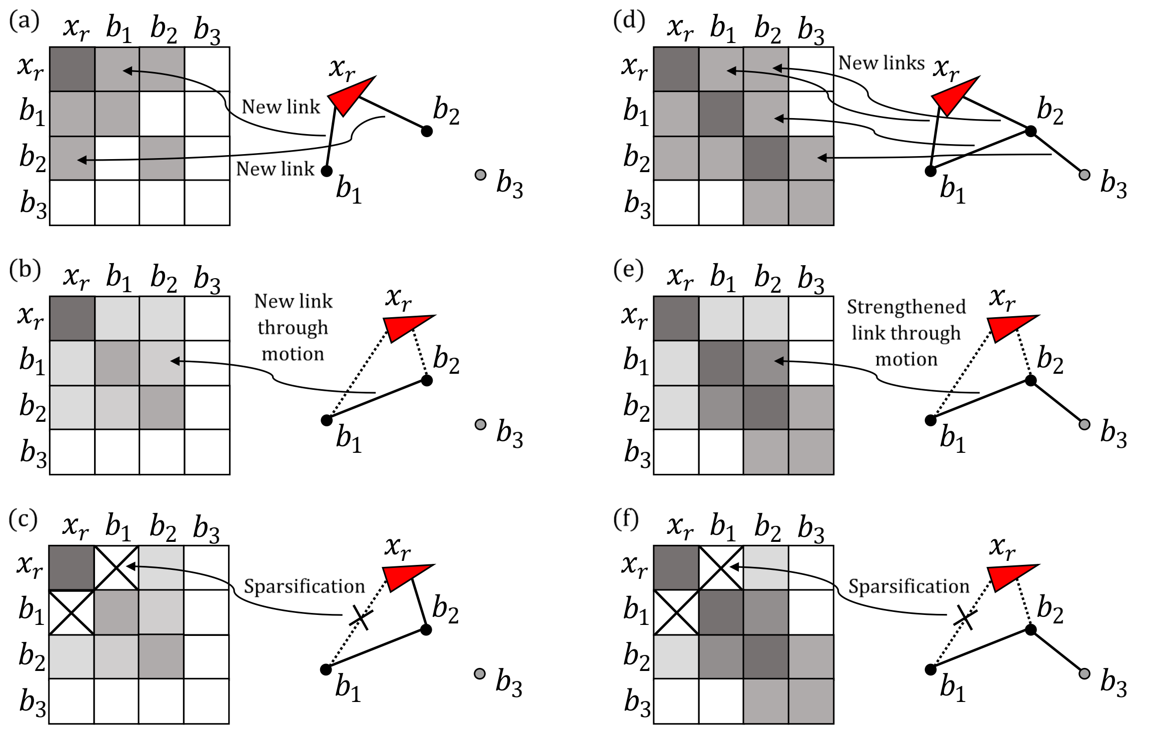


Figure 5.3: (Left) Effect of the steps in the conventional SEIF SLAM on the information matrix: (a) measurement integration, (b) Prediction stage and (c) sparsification. (Right) Effect of the steps in **Scheme2** on the information matrix : (d) measurement integration, (e) Prediction stage and (f) sparsification.

In **Scheme2** the Update stage integrates $z_{r,1}$, $z_{r,2}$, $z_{1,2}$ and $z_{2,3}$, which strengthens robot-beacon links between the robot and all direct beacons –$\Omega_{r,1}$, $\Omega_{r,2}$– and also the involved inter-beacon links –$\Omega_{1,2}$ and $\Omega_{2,3}$, see Fig. 5.3-d. The Prediction stage transfers some low amount of information from $\Omega_{r,1}$ and $\Omega_{r,2}$ to $\Omega_{1,2}$, see Fig. 5.3-e. At that time, links $\Omega_{r,1}$, $\Omega_{r,2}$, $\Omega_{1,2}$ and $\Omega_{2,3}$ are strong. Finally, the sparsification step deactivates b_1 and removes $\Omega_{r,1}$, Fig. 5.3-f.

Scheme2 creates exactly the same robot-beacon links that SEIF SLAM and exactly with the same value –strength. The main difference between both is how they treat inter-beacon links. In SEIF SLAM the only way to create/reinforce inter-beacon links is through the robot motion in the Prediction stage. Besides, in **Scheme2** they are created/reinforced also when integrating inter-beacon measurements. As a result inter-beacon links accumulate significantly more information than in SEIF SLAM.

5.4.2 Preservation of the sparsity of the information matrix

The conventional SEIF SLAM imposes sparsification bounds both in the number of robot-beacon links, θ_x, and in the number of inter-beacon links, θ_y (Thrun et al., 2004). In (Thrun et al., 2004) it was proven that keeping the number of robot-beacon links below θ_x automatically constrains the number of inter-beacon links below θ_y. One could think that in **Scheme2** integrating inter-beacon measurements may increase the number of inter-beacon links, violating the sparsification bound. In this section we proof that this is not the case: we demonstrate that in many scenarios it creates exactly the same inter-beacon links that the conventional SEIF SLAM. **Scheme2** accumulates higher amount of information than the conventional SEIF SLAM, but this increment is not used to create new links but to increase the strength of existing links, enabling lower map and robot uncertainty with a compact description.

Without loss of generality we assume that: (1) every beacon is within the robot sensing range at some time in the robot path and; (2) every pair of neighbor beacons in the environment (beacons which are within the sensing region one another) fall simultaneously within SR_r at some time in its path. These assumptions do not involve practical constraints. They are met in cases where the robot follows dense-navigation paths, which are typically adopted in SLAM in mapping or surveillance applications. Besides, in Section 5.4.4 it is shown that sparsity preservation is confirmed also when these assumptions are relaxed.

In **Scheme2** inter-beacon links are created through the robot motion and also by the integration of inter-beacon measurements. In the following the sets of inter-beacon links created in the full experiment through both mechanisms are analyzed. The integration of $z_{i,j}$ creates/ reinforces inter-beacon link $\Omega_{i,j}$. Assuming dense-navigation robot paths, the robot will integrate inter-beacon measurements of every pair of neighbor beacons $\forall\, b_i, b_j : b_i \in SR_j$. Thus, in the full robot path the set of inter-beacon links created by inter-beacon measurements are:

$$L_{IB} = \{\Omega_{i,j} > 0 \quad \forall\, b_i, b_j :\; b_i \in SR_j\} \tag{5.11}$$

Besides, recall that the robot motion creates an inter-beacon link between any pair of beacons in which both beacons fall simultaneously within SR_r. The robot path is assumed dense enough such that every pair of neighbor beacons b_i and b_j in the environment –$\forall b_i, b_j :\; b_i \in SR_j$– fall simultaneously within SR_r at some time in the robot path. Thus, the set of inter-beacon links between neighbor beacons created through the robot motion in the full robot path is:

$$L_R^1 = \{\Omega_{i,j} > 0 \quad \forall\, b_i, b_j :\; b_i \in SR_j\} \tag{5.12}$$

The robot motion creates inter-beacon links between neighbor beacons but, also between beacons that are not neighbors one another but both are simultaneously within SR_r at a certain time t. The set of inter-beacon links between non-neighbor beacons will be called L_R^2. In the general case $L_R^2 \neq \emptyset$ and the set of inter-beacon links created through the robot motion is $L_R = L_R^1 \cup L_R^2$. From Eqs. (5.11) and (5.12) it is easy to notice that $L_{IB} \subseteq L_R$.

In the conventional SEIF SLAM inter-beacon links are created only via the robot motion: the set of inter-beacon links is $L_{SEIF} = L_R$. In **Scheme2** they are created through both mechanisms: the set of inter-beacon links is $L_{prop} = L_{IB} \cup L_R = L_R$. Thus, **Scheme2** creates exactly the same inter-beacon links as SEIF SLAM, preserving the sparsity of the information matrix. From Section 5.4.1 it is clear that both create the same robot-beacon links. Thus, it is concluded that under the above assumptions both create the same robot-beacon and also inter-beacon links.

If the dense-navigation robot path assumption is not met, inter-beacon measurements could create some links in L_{IB} that are not present in L_R. Section 5.4.4 evaluates through simulation the effects of relaxing this assumption.

5.4.3 Preservation of constant time property

Constant time execution is a well known property of SEIF SLAM. **Scheme2** distributes measurement gathering and the computation of the Update stage, making them independent of the map size. This section analyzes if the state/map recovery in **Scheme2** preserves constant time execution.

A critical key for SEIF SLAM being constant time is that it only recovers the state of the robot and of all the active direct beacons. As described in Section 5.3.1 **Scheme2** recovers ev_i for each direct beacon b_i currently within SR_r. Recalling (5.6), state/map recovery for direct beacon b_i involves computing: μ_r, μ_i and the state of the beacons neighbors of b_i. This should be done for each direct beacon at time t. Thus, considering all direct beacons it is easy to notice that at time t state/map recovery should recover: the robot state; the states of currently active direct beacons and; the state of active indirect beacons at time t –recall the definition in Section 5.3. In **Scheme2** recovering the states of indirect beacons involves an increment in the computational cost over that of SEIF SLAM. However, as proven below state/map recovery is kept constant time.

Let n_{DB} and n_{IB} be the number of direct and indirect beacons in the surrounding of the robot at time t. SEIF SLAM recovers the states of the robot and of direct beacons (in black in Fig. 5.2). **Scheme2** recovers also the state of indirect beacons (in gray in Fig. 5.2). It uses the state recovery algorithm presented in (Thrun et al., 2005), which complexity is linear with the number of beacons states recovered: $O(n_{DB})$ for SEIF SLAM and $O(n_{DB} + n_{IB})$ for **Scheme2**.

The complexity of state/map recovery for the proposed method is as bounded as it is for SEIF SLAM. The computational burden of state/map recovery in **Scheme2** –and also in SEIF SLAM– depends on beacon density, and not on the total number of beacons present in the environment. Highly inhomogeneous local beacon densities are not very useful for RO-SLAM. It is more interesting if beacons are deployed in densities with some homogeneity. In these cases $n_{DB} + n_{IB}$ will have similar values along the robot path.

5.4.4 Simulation relaxing assumptions

This section analyzes using simulations the conclusions in Sections 5.4.2 and 5.4.3 when the assumptions are relaxed. A $70x70m$ scenario with 50 randomly deployed beacons was considered. The range measurement noise is assumed Gaussian with zero mean and standard deviation of $\sigma_m = 0.8m$. The robot trajectories were different in every experiment and did not meet the dense-navigation assumption. The sparsification bound was $\theta_x = 10$, in the range suggested in (Thrun et al., 2005). The proposed method was compared with EKF SLAM and SEIF SLAM. All the methods used PFs with 500 particles for beacon initialization and of course, all were set with exactly the same parameters.

It is easy to notice that **Scheme2** integrates more information than EKF SLAM and SEIF SLAM. Table 5.1 compares their performance in a set of 50 simulations with different beacon settings and robot paths. Two main metrics are analyzed: the average amount of information integrated in Ω and the average number of inter-beacon links created. The amount of information is measured by the logarithm of the determinant of the covariance matrix (or the inverse information matrix), which is proportional to the entropy of a multivariate Gaussian. **Scheme2** creates the same robot-beacon links than SEIF SLAM and are not compared. In average, the proposed method integrated 66% more information than EKF SLAM and 67% more than the conventional SEIF SLAM. Besides, in average EKF SLAM created 1042 inter-beacon links, SEIF SLAM created 631 and **Scheme2**, 645. These scenarios do not meet the dense-navigation robot path assumption and **Scheme2** created only 2.2% more inter-beacon links but integrated 67% more information than the conventional SEIF SLAM.

The proposed method creates the same inter-beacon links as the conventional SEIF SLAM in case of having dense-navigation robot paths. If not, it creates links involving beacons that are out of the sensing range of the robot in its path. These links could

	EKF SLAM	SEIF SLAM	**Scheme2**
$-\log(\lvert\Omega\rvert)$	197.37	197.03	328.41
inter-beacon links	1042	631	645

Table 5.1: Comparison of the average amount of information accumulated and the number of inter-beacon links created by EKF SLAM, SEIF SLAM and **Scheme2** in a set of experiments.

not be created by SEIF SLAM and are useful to improve map estimation completeness. For instance, Fig. 5.4 shows the inter-beacon links created by SEIF SLAM and the proposed method in one example. In **Scheme2** all the beacons created links with other beacons. That is not the case with SEIF SLAM, which cannot create links with some beacons originating incomplete maps.

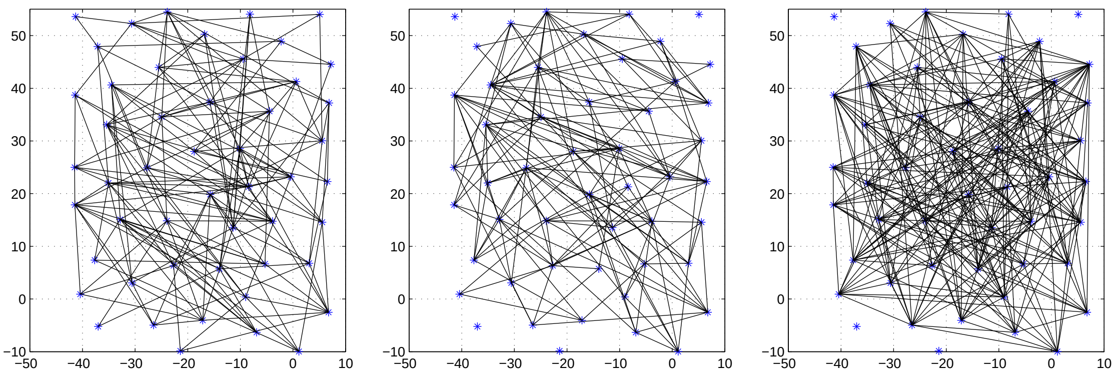


Figure 5.4: Inter-beacon links created in one simulation: (left) the proposed method; (center) SEIF SLAM; and (right) EKF SLAM. Only the 25% strongest links are shown for better visibility.

Scheme2 creates a relatively low number of strong inter-beacon links between nearby beacons which provide high amount of information of the map. As an example, Fig. 5.5-center shows the values of the inter-beacon links created in each case in the simulation in Fig. 5.4. The links resulting in each case are ordered –stronger links first– for visualization. Similar conclusions were obtained in the rest of scenarios. The complexity of state/map recovery in SEIF SLAM and **Scheme2** are respectively $O(n_{DB})$ and $O(n_{DB} + n_{IB})$. The values of n_{DB} and n_{IB} along the example in Fig. 5.4 are shown in Fig. 5.5-bottom. Similar conclusions were obtained in the rest of

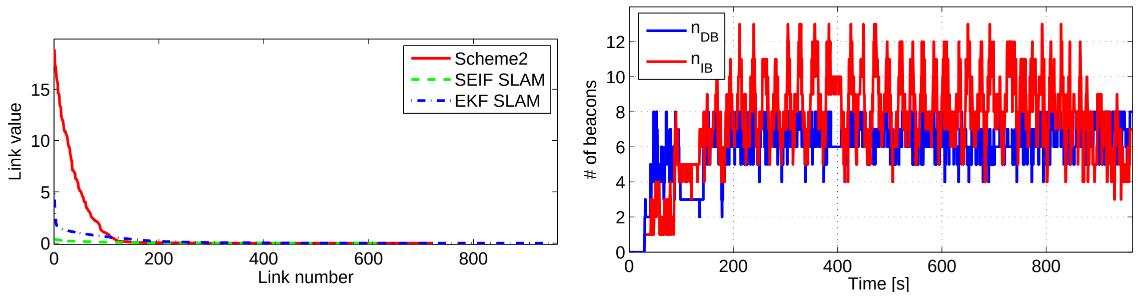


Figure 5.5: (Left) Values of the inter-beacon links created by the proposed method, SEIF SLAM and EKF SLAM, ordered –stronger links first– for visualization. (Right) Values of n_{DB} and n_{IB} along the simulation.

simulations. It can be noticed that n_{IB} is higher when n_{DB} is higher, and lower when n_{DB} is lower, exhibiting some proportionality. n_{DB} took values between 4 and 8 during most of the time, and n_{IB}, between 5 and 13. The lowest values took place when the robot was located at the borders of the scenario, where few beacons were within SR_r. The highest occurred the robot was at places with high local beacon densities. These results depend on beacon density and not on map size.

5.5 Evaluation and comparison

This section is divided in two parts. The first one evaluates the performance of **Scheme2** compared with other methods. The second discusses on three aspects: robustness against message errors, consumption of resources of beacons and scalability.

The proposed scheme has been validated in series of experiments performed in the CONET Integrated Testbed, which description is summarized in Appendix A. The preliminary experiments shown here use a network of 8 *Nanotron nanoPAN* devices equipped with range sensors, see Appendix B for more details. As it will be detailed later, scalability analysis is performed by simulating bigger maps (higher number of beacons) under the same conditions of the real experiments.

5.5.1 Performance analysis

Figure 5.6 shows the results of the proposed **Scheme2** in one experiment. It was compared with EKF and traditional SEIF in series of 25 experiments with different maps and robot paths, see Table 5.2. The improvement in the map error is 38.5% w.r.t. SEIF and 36% w.r.t. EKF. The improvement in the robot localization is lower but it is still a 16% more accurate than traditional methods. The testbed allowed performing the same experiment many times in the same conditions, which confirmed the repeatability of results.

	EKF	SEIF	**Scheme2**
Map RMS error [m]	0.25	0.26	0.16
Robot RMS error [m]	0.49	0.50	0.42
PF convergence times [s]	49.84	49.96	10.52

Table 5.2: Mean map and robot errors and auxiliary PF convergence time in series of 25 experiments.

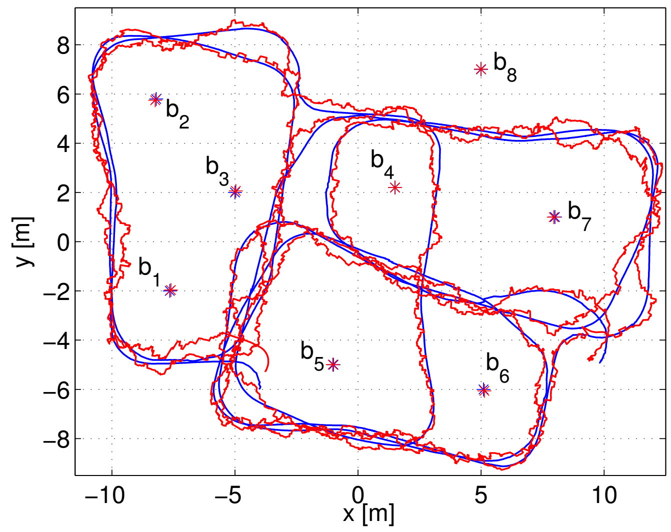
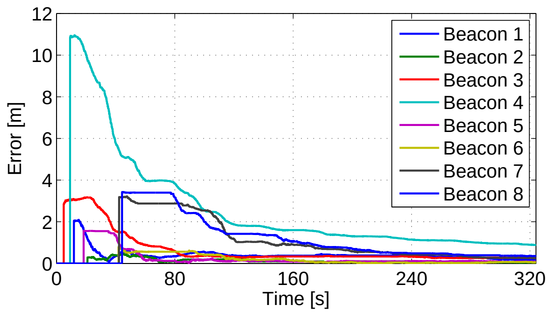


Figure 5.6: Results of the proposed *Method2*. Estimations are in red color and ground truth in blue.

Figure 5.7 presents the evolution of the location error for each beacon in the proposed **Scheme2** (right) and the traditional SEIF (left). The drawing for each beacon starts when its auxiliary PF converged. In the proposed method the beacon errors are significantly lower and beacon PFs converge significantly sooner. In average in the traditional SEIF all PFs converged at $t = 49.84s$ and in our method they converged at $t = 10.52s$ (78% earlier).



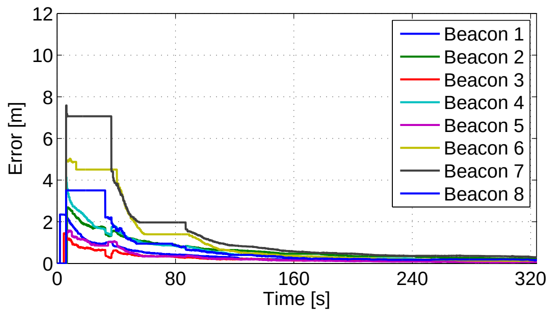


Figure 5.7: Evolution of the location error for each beacon for traditional SEIF (left) and the proposed **Scheme2** (right).

In these experiments, the proposed method created exactly the same inter-beacon links –143– as SEIF SLAM. The sensing range of *Nanotron* sensors was higher than 100 m and hence, the robot dense-navigation assumption was met. Figure 5.8 analyzes the evolution of the values of each inter-beacon link along the experiment in **Scheme2** (red) and in SEIF SLAM (green). For better illustrating the tendency, the envelope that groups the 90% central curves is shown for each case. Links start taking non-zero values in **Scheme2** sooner than in SEIF SLAM, which is originated by the shorter beacon PF convergence times. The values of inter-beacon links in **Scheme2** grow at higher constant rate –and are higher at any time– than in SEIF SLAM. At any time along the experiment the proposed method has accumulated more information in Ω, involving lower mapping uncertainty.

5.5.2 Discussion

This section analyzes three aspects of **Scheme2**: (1) robustness against message errors, (2) consumption of resources of beacons and (3) scalability.

The impact of message loss was evaluated using the measurements collected in the real experiments and simulating transmission errors with Packet Reception Rates

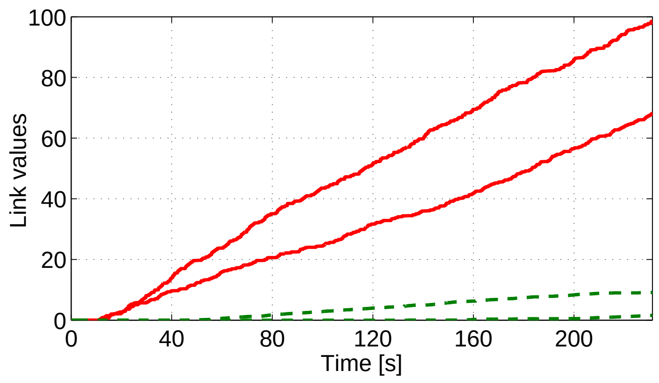


Figure 5.8: Evolution of the values of inter-beacon links along the experiment in SEIF SLAM (green) and in **Scheme2** (red). For better illustrating the tendency, the envelope that groups the 90% central curves is shown.

(PRRs) in the range [60%-99%]. Figure 5.9 shows the map and robot localization errors obtained with EKF SLAM, SEIF SLAM and the proposed *Scheme2v2*. The first two are not influenced by transmission errors. The Update stage of *Scheme2v2* is additive, which makes it rather robust to message loss. Map RMS error for *Scheme2v2* is lower than for EKF SLAM and SEIF SLAM for any PRR level. *Scheme2v2* had the lowest robot error for PRR levels higher than 70%. Although the proposed method makes extensive use of transmissions, it is significantly robust to transmission errors.

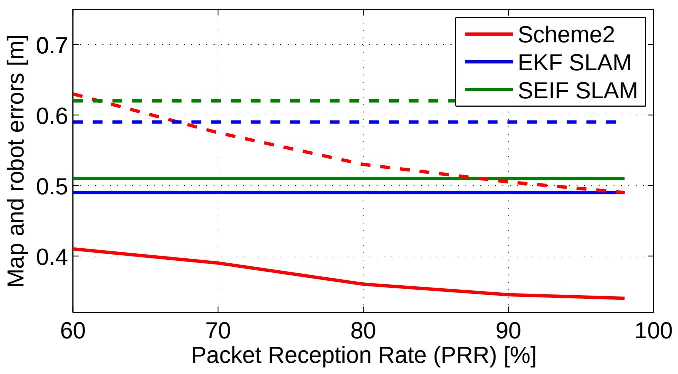


Figure 5.9: Impact of PRR on map –full lines– and robot –dashed– errors for the proposed *Scheme2v2*, EKF SLAM and SEIF SLAM.

In order to analyze scalability we performed series of simulations, with the same parameters of the real experiments. 25 experiments were performed for each map size with between 20 and 200 beacons –the beacon density was kept steady. Figure 5.10-a shows the average robot computational burden of the EKF SLAM, SEIF SLAM and the proposed *Scheme2v1* and *Scheme2v2*. *Scheme2v2* is the most efficient for maps with more than 50 beacons. For any map size *Scheme2v2* consumes lower robot CPU time than *Scheme2v1* and, *Scheme2v1*, lower than SEIF SLAM. Figure 5.10-b shows the map and robot errors obtained. The proposed *Scheme2v2* obtains the lowest errors. *Scheme2v1* and *Scheme2v2* obtain similar errors: only *Scheme2v2* is shown for clarity. The advantage is more evident with larger maps. Inter-beacon measurements enforce map consistency and, in absence of inter-beacon measurements, the estimations are more influenced by the robot odometry noise.

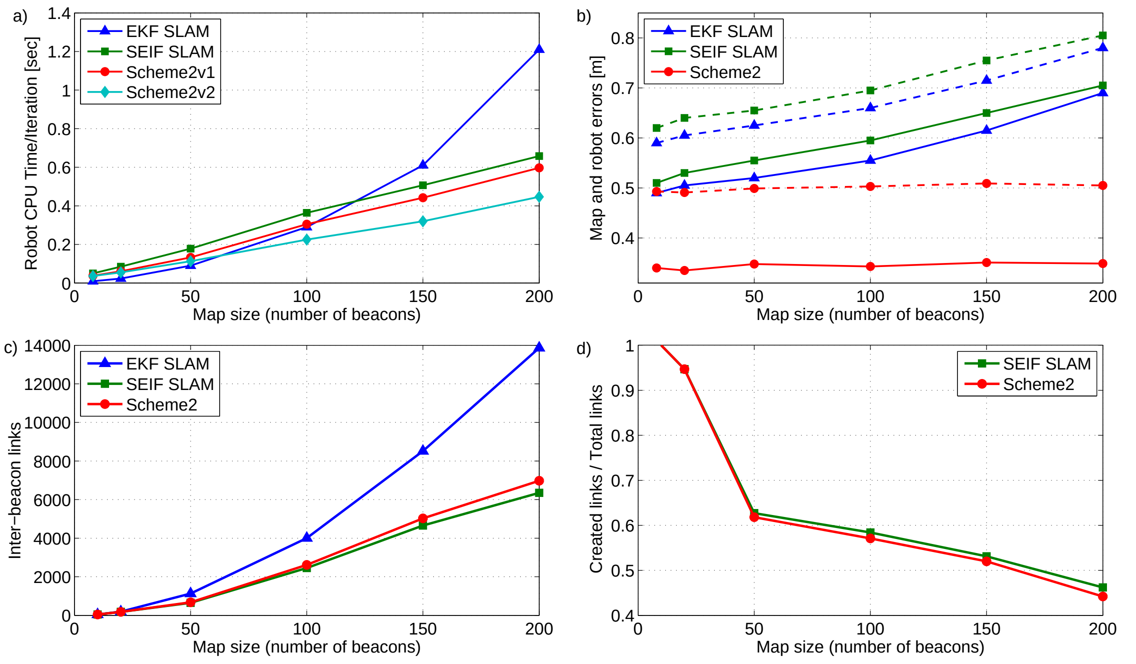


Figure 5.10: Performance analysis of the proposed *Scheme2v1*, *Scheme2v2*, EKF SLAM and conventional SEIF SLAM with different map sizes: (a) robot computational burden measured using MATLAB profiler; (b) map –represented with full lines– and robot –dashed lines– errors; (c) number of inter-beacon links created by each method; (d) sparsity of the information matrix in terms of the ratio between created links and total possible links.

Figure 5.10-c shows the average number of inter-beacon links created at the end of each simulation by each method. Figure 5.10-d shows the sparsity of the information matrix expressed as the ratio between the inter-beacon links created in the experiment and the total number of possible inter-beacon links. The sparsity of the information matrix increases with the map size. They both perform similarly in terms of the absolute number of links and sparsity of the information matrix. Notice that although with higher map sizes the number of links created in **Scheme2** is a bit higher than in SEIF SLAM, its information matrix is sparser. Integrating inter-beacon measurements allows **Scheme2** to detect more beacons and add them to the state vector, improving map completeness, but these new beacons create very few links, increasing the sparsity of the information matrix. For instance, with maps of 190 beacons, the state vector in **Scheme2** contained 185 beacons, created 7048 inter-beacon links out of the total of 17020 possible links, whereas in SEIF SLAM the state vector contained 167 beacons and created 6349 inter-beacon links out of 13861 possible links. **Scheme2** builds more complete maps involving larger state vectors and information matrices than SEIF SLAM but also the information matrices are sparser and the robot CPU burden is lower as shown in Figure 5.10-a.

5.6 Conclusions

This chapter described **Scheme2**, **Contribution3** of this PhD Thesis and the second of the RO-SLAM schemes that adopts the architecture presented in Section 3.3.

This chapter proposed a distributed SEIF SLAM scheme for robot-sensor network cooperation applications. This cooperation allows the distribution between the robot and the sensor network nodes (beacons) of tasks like measurement gathering and integration in SLAM. Beacons take range measurements to the robot and to their neighbor beacons, estimate their contribution to the SLAM Update stage and send it to the robot. Besides robot-beacon measurements, the proposed **Scheme2** naturally integrates inter-beacon measurements, resulting in more accurate map and robot estimations and faster convergence of the beacon initialization PFs, as demonstrated in Chapter 3. Distributing measurement integration in SLAM reduces the robot burden.

The impact of **Scheme2** on the building of the information matrix was analyzed in detail. It was proven that: (1) it accumulates higher amount of information than SEIF SLAM; (2) it creates a very similar number of links than SEIF SLAM, preserving the sparsity of the information matrix and; (3) it preserves the constant time property of SEIF SLAM. **Scheme2** integrates more information than SEIF SLAM preserving its compact description, efficiency and scalability.

Chapter 6

Resource-constrained SEIF-based RO-SLAM with inter-beacon measurements

6.1 Introduction

This chapter presents a RO-SLAM scheme that is capable of exploiting robot-beacon cooperation in order to improve SLAM accuracy and efficiency while meeting a given resource consumption bound. This is **Scheme3**, **Contribution4** of this PhD Thesis and the third of the RO-SLAM schemes that adopts the architecture presented in Section 3.3.

In this scheme the resource consumption bound is expressed in terms of the maximum number of measurements that can be integrated in SLAM per iteration. The sensing channel capacity used, the beacon energy consumed or the computational capacity employed, among others, are proportional to the number of measurements that are gathered and integrated in SLAM. The proposed **Scheme3** can meet static and dynamic bounds, e.g. determined by an online resource allocation tool, enabling high flexibility, which can be of interest in many cases.

Scheme3 is based on the distributed SEIF SLAM used in **Scheme2** (see Chapter 5), in which each beacon gathers and integrates in the SLAM update stage robot-beacon and inter-beacon measurements. It also comprises a distributed tool that uses greedy gain-cost analysis to dynamically select the most informative measurements to be integrated in SLAM. Thus, **Scheme3** distributes between beacons not only the SEIF Update stage but the measurement gathering control. This scheme has a robot-beacon distributed approach were beacons actively participate in measurement selection, gathering and integration in SLAM.

The proposed **Scheme3** has the following main properties: (1) adaptive resource-constrained operation, since it dynamically adapts to satisfy the given resource consumption bound; (2) accuracy, since it integrates inter-beacon measurements, significantly improving map and robot localization accuracies and speeding up beacon initialization; (3) efficiency, since it gathers and integrates the most informative measurements and; (4) scalability, since all the involved tasks are executed in a distributed manner between the robot and beacons.

The chapter is organized as follows. Section 6.2 describes the proposed **Scheme3**. The details of robot and beacon operations are in Sections 6.3 and 6.4. Evaluation of the scheme and discussion of its main parameters and characteristics are in Section 6.5. Conclusions are the last section.

6.2 Resource-constrained SEIF-based RO-SLAM

Efficiency in the use of resources is very important in robot-sensor network cooperation. In most cases the SLAM algorithm is executed simultaneously with other tasks, all of them sharing the available resources. Also, radio beacons gathering range measurements such as RSSI, Time of Arrival (ToA) or Differential ToA, make use of some kind of communication, which requires using a channel with a certain (constrained) capacity. In fact the capacity of the sensing channel is one of the most relevant constraints in settings with a high number and density of deployed beacons. In our problem resource consumption can be expressed in terms of the maximum number of measurements that are gathered and integrated in SLAM at each iteration. The use of the sensing channel, the consumption of beacons energy and of beacons computational capacity are proportional to the number of measurements that are gathered and integrated in SLAM. Hence, bounding the number of measurements constrains the consumption of the main resources involved.

A diagram of the proposed **Scheme3** with its main modules is shown in Fig. 3.4. **Scheme3** combines (1) a distributed SEIF SLAM method in which beacons gather and integrate in SLAM robot-beacon and inter-beacon measurements, and (2) a distributed information-driven measurement selection tool that dynamically selects the measurements that are integrated in SLAM in order to improve performance while fulfilling the bound in the total number of measurements. Both modules are executed in a distributed manner by the robot and by the beacons. Each beacon maintains a local version of the SLAM state whereas the global state is maintained only by the robot. The message interchange and the operation of the robot and beacons is summarized in Fig. 6.1.

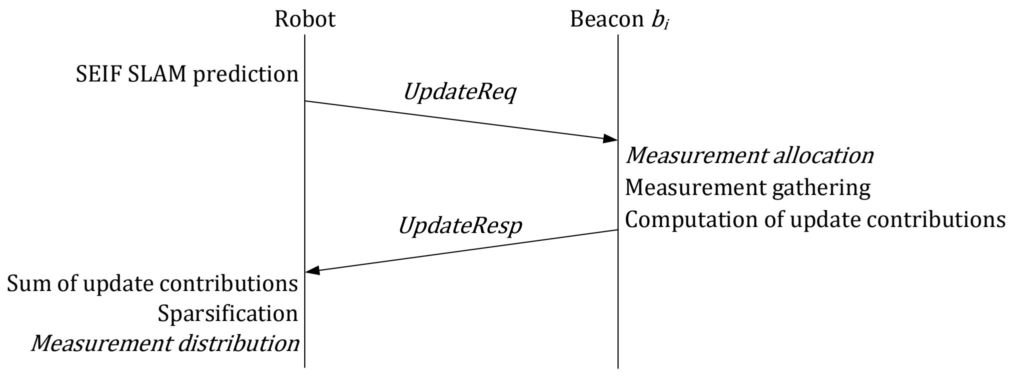


Figure 6.1: Operation and message interchange in **Scheme3**.

Methods that select the measurements that best reduce the uncertainty in the SLAM global state are necessarily centralized and have to deal with the information matrix of the global state. These methods incur in high computational burden with large maps and scale badly with the map size. **Scheme3** approximates this centralized measurement selection by a robot-beacon distributed tool that preserves the constant time execution and scalability. In Section 6.5 it is experimentally shown that the adopted distributed measurement selection tool is almost as accurate as the centralized measurement selection –difference in map and robot RMS errors lower than 3%.

The distributed measurement selection tool is performed in two steps: *measurement distribution* and *measurement allocation*. In *measurement distribution* the robot dynamically decides the number of measurements that are assigned to each beacon within the robot sensing region using expectations of uncertainty reductions. In *measurement allocation* each beacon b_i decides the actual measurements that it will gather and integrate in SLAM analyzing the cost and expected uncertainty improvement of integrating each measurement.

The bound in the number of measurements that can be gathered in each iteration is NM_{max}. As example where NM_{max} has a high value is shown in Fig. 6.2-left: each beacon within the robot sensing region gathers and integrates a measurement to each beacon within its sensing region. An example with a low NM_{max} is shown in Fig. 6.2-right. b_1 and b_2 are assigned with only one measurement and gather $z_{1,r}$ and $z_{2,r}$. b_3 is assigned with two measurements and besides $z_{3,r}$, it gathers $z_{3,2}$, the measurement that achieves the best expected uncertainty improvement-cost trade-off.

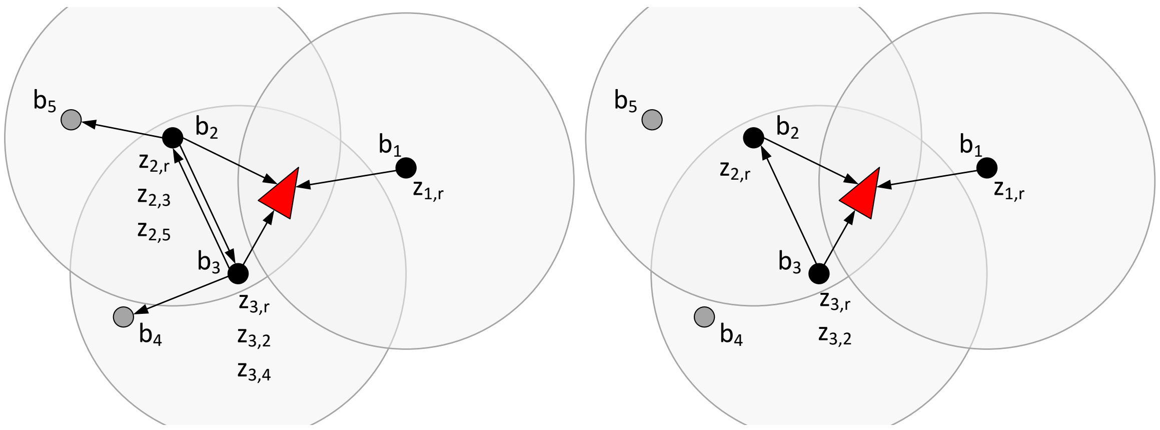


Figure 6.2: (Left) Measurement gathering in **Scheme3**. (Right) Measurement gathering with a low value of NM_{max}. b_1 gathers $z_{1,r}$. b_2 gathers $z_{2,r}$. b_3 gathers $z_{3,r}$ and $z_{3,2}$. Gray circles represent the sensing regions of beacons b_1, b_2 and b_3.

6.3 Operation of the robot

The operation of the robot in **Scheme3** can be decomposed in four main tasks: (1) computation of the SEIF SLAM Prediction stage; (2) reconstruction of the updated state using the contributions received by the robot; (3) computation of the sparsification step; and (4) *measurement distribution*. For brevity, most SEIF equations have been omitted. Refer to (Thrun et al., 2005) for further details.

The robot operation in **Scheme3** is as follows, see Alg. 6.1. First, the robot computes the SEIF SLAM prediction. Static beacons and nonlinear robot kinematic model are assumed. The robot Jacobian is computed at each time, which requires recovering the state. **Scheme3** uses the efficient algorithm described in (Thrun et al., 2004) for motion update and state recovery. This algorithm computes the predicted information vector $\bar{\xi}_t$ and information matrix $\bar{\Omega}_t$ and recovers the predicted μ_t.

Algorithm 6.1 Summary of the operation of the robot in **Scheme3**. ξ and Ω are the information vector and matrix. LM is the list of measurements assigned to each beacon. ξ_i and Ω_i are the contributions of beacon b_i to the information vector and matrix.

```
Require: $\xi_{t-1}, \Omega_{t-1}, NM_{max}, LM$
1: SEIF motion update and state recovery
2: Create and broadcast UpdateReq message
3: Receive UpdateResp messages
4: Extract $\xi_i$, $\Omega_i$ and $ui_i$ from UpdateResp messages
5: Compute $\xi_t$ and $\Omega_t$ as in (6.2)-(6.3)
6: SEIF Sparsification
7: Measurement distribution. Create $LM$
8: return $\xi_t, \Omega_t, LM$
```

As the robot moves beacons go in and out of the robot sensing region. The robot maintains BS_r, a list with the beacons that are currently within its sensing region. At each time t the robot broadcasts an *UpdateReq* message that includes μ_t and LM, a list created by the robot at t-1 that contains the number of measurements that have been assigned to each beacon $b_i \in BS_r$. Transmitting the whole μ_t in the *UpdateReq* message is not suitable in cases with large maps. Only the elements in μ_t required for each beacon are transmitted. Let ev_i be the vector with the estimates required by beacon b_i to compute its contribution to the update stage:

$$ev_i = [\mu_r \quad \mu_i \quad \mu_j]^T, \tag{6.1}$$

where μ_r is the estimation of the robot current location, μ_i is the estimation of the location of beacon b_i and μ_j represents the estimations of the location of every beacon b_j within the sensing region of b_i.

When b_i receives the *UpdateReq* message, it performs as described in Section 6.4 and transmits to the robot an *UpdateResp* message with $\xi_{i,t}$ and $\Omega_{i,t}$, its contribution to the SLAM update stage. The robot reconstructs the updated state –ξ_t and Ω_t– using $\bar{\xi}_t$ and $\bar{\Omega}_t$ and the update contributions it received:

$$\xi_t = \bar{\xi}_t + \sum_i F_i^T \xi_{i,t}, \tag{6.2}$$

$$\Omega_t = \bar{\Omega}_t + \sum_i F_i^T \Omega_{i,t} F_i, \tag{6.3}$$

where F_i is the projection matrix that implements the operations necessary to allocate $\xi_{i,t}$ and $\Omega_{i,t}$ at the suitable entries in ξ_t and Ω_t.

Next, the robot checks if Ω_t satisfies the SEIF sparsification bound. If not, the beacons with the weakest links are deactivated as described in (Thrun et al., 2004).

The final step performed by the robot is to distribute the number of measurements NM_{max} between $b_i \in BS_r$. NM_{max} is considered an input to **Scheme3**. It can be static or dynamic, computed by an online resource allocation tool, for instance analyzing the capacity of the channel using link quality estimators, see e.g. (Baccour et al., 2012). Measurement distribution is performed proportionally to $IG_{i,t}$, the usefulness of the measurements from beacon b_i to reduce the uncertainty of the SLAM state. b_i has impact on the SLAM state only if its update contribution reaches the robot, i.e. if b_i receives the *UpdateReq* message sent by the robot and if the robot receives the *UpdateResp* message with the update contribution from b_i. These two events are statistically independent. Taking $p_{r,i}$ as the Packet Reception Rate (PRR) from the robot to b_i and assuming symmetric PRRs, $IG_{i,t}$ can be estimated as:

$$IG_{i,t} = p_{r,i}^2 \, ui_{i,t}, \tag{6.4}$$

where $ui_{i,t}$ estimates the capability of the measurements gathered by b_i to reduce the uncertainty in the SLAM state. Transmission errors in sensor networks are not infrequent. This criterion naturally assigns more measurements not only to the most informative beacons, but also to those with better communication with the robot.

Each b_i computes its own $ui_{i,t}$ –described in Section 6.4– and transmits it to the robot in an *UpdateResp* message. The robot can measure $p_{r,i}$ to each $b_i \in BS_r$ by simply analyzing message transmission success. Next, the robot allocates the NM_{max} measurements among beacons $b_i \in BS_r$ proportionally to $IG_{i,t}$ and creates LM, the list with the number of measurements assigned to each beacon $b_i \in BS_r$.

6.4 Operation of beacons

The operation of beacons in **Scheme3** is summarized in Alg. 6.2. Once beacon b_i has received the *UpdateReq* message it performs as follows: (1) executes *measurement allocation* and selects the most informative measurements; (2) gathers and integrates in SLAM the selected measurements and; (3) transmits to the robot its update contribution in an *UpdateResp* message.

6.4.1 Measurement allocation

UpdateReq messages include LM. Once beacon b_i has received an *UpdateReq* message, it extracts LM_i, the number of measurements it was assigned with. Let BS_i be the set of beacons b_j within the sensing region of b_i. In *measurement allocation* each beacon b_i selects which measurements $z_{i,j}, b_j \in BS_i$ it should gather and integrate in SLAM. We adopt a common approach in information-driven measurement selection and formulate the problem as the greedy optimization of a utility function that establishes a trade-off between information gain and resource consumption:

$$J_{i,j} = r_{i,j} - \alpha c_{i,j} \tag{6.5}$$

Algorithm 6.2 Summary of the operation of beacon b_i in **Scheme3**. ξ_i and Ω_i are the contributions of beacon b_i to the information vector and matrix. LM_i is the number of measurements assigned to beacon b_i. ev_i is the state vector required by beacon b_i. $J_{i,j}$ is the utility function for measurement selection. ui_i is the estimated improvement in the uncertainty if b_i takes measurements to every neighbor. MS_i is the set of measurements gathered by beacon b_i.

```
1: Receive *UpdateReq* message. Extract $ev_i$ and $LM_i$
2: *Measurement allocation.* Compute $J_{i,j}$ and $ui_i$
3: Gather the $LM_i$ measurements with the highest $J_{i,j}$. Create $MS_i$
4: **if** ($b_i$ is at “state vector phase”) **then**
5:     Compute $\xi_{i,t}$ and $\Omega_{i,t}$ with $MS_i$ as in (6.11)-(6.12)
6: **else**
7:     Use $MS_i$ to update the PF of $b_i$
8: **end if**
9: Create *UpdateResp* message and transmit it to the robot
```

$c_{i,j}$ is the cost of the resources consumed in gathering and integrating measurement $z_{i,j}$. In sensor networks, energy is maybe the most constrained resource. We take $c_{i,j}$ as the energy consumed by b_i in gathering and integrating $z_{i,j}$. $c_{i,j}$ could be different for each beacon, e.g. depending on the remaining energy in its batteries. For simplicity, $c_{i,j}$ was assumed the same for all measurements. The reward $r_{i,j} = ui_{i,j}$ is the expected SLAM uncertainty improvement resulting after integrating $z_{i,j}$. α is a weighting factor that balances the cost the reward. Its effects will be evaluated in Section 6.5.2.

The reward $r_{i,j} = ui_{i,j}$ is determined as follows. In **Scheme3** each beacon maintains its own local state. $\overline{\Omega}_{i,t}$ is the predicted information matrix for time t of the local state of b_i. $\overline{\Omega}_{i,t}$ was computed by b_i at t-1. It is easy to notice that the updated information matrix of the local state of b_i that would result after integrating measurement $z_{i,j}$ is:

$$\Omega`_{i,t} = \overline{\Omega}_{i,t} + \Omega_{i,j,t} \tag{6.6}$$

where $\Omega_{i,j,t}$ is the expected contribution of $z_{i,j}$ and is computed as follows:

$$\Omega_{i,j,t} = H^T_{i,j,t} R^{-1} H_{i,j,t}, \tag{6.7}$$

where R is the covariance matrix of the measurement noise and $H_{i,j,t}$ is the Jacobian of the observation model of measurement $z_{i,j}$ computed with ev_i, just received from the robot in the *UpdateReq* message.

On the other hand, in case of not integrating $z_{i,j}$, the updated information matrix for b_i would be $\Omega^n_{i,t} = \overline{\Omega}_{i,t}$. The uncertainty improvement $ui_{i,j}$ is the difference of the uncertainty in $\Omega`_{i,t}$ and in $\Omega^n_{i,t}$. Entropy is maybe the most widely-used metric for the uncertainty in a probability distribution. It is adopted in **Scheme3**. Entropy can be used to measure the uncertainty of beacons in the "state vector phase" and also of beacons in the "initialization phase", giving the same treatment to both cases. If beacon b_i is in the "state vector phase", its state follows a Gaussian probability distribution and its entropy can be computed using an exact expression. In this case $ui_{i,j}$ is as follows:

$$ui_{i,j} = \frac{1}{2}\log\left(\frac{|\overline{\Omega}_{i,t}|}{|\overline{\Omega}_{i,t} + \Omega_{i,j,t}|}\right) \tag{6.8}$$

If b_i is in the "initialization phase", i.e. its PF has not converged, its probability distribution is approximated by the set of PF particles. In this case, there is not an exact expression and each b_i computes its $ui_{i,j}$ using the approximate calculation described in (Boers et al., 2010).

It should be noticed that if b_j is in the "initialization phase", it is still not in the state vector of its neighbor b_i. Hence, $z_{i,j}$ is not useful to update the local map of b_i, either if b_i is in the "initialization phase" or in the "state vector phase". Hence, the uncertainty improvement $ui_{i,j}$ is taken as zero.

Long-term optimization of $J_{i,j}$ involves high computational burden and bad scalability. We adopted a simple but efficient greedy approach: at each time b_i selects the LM_i beacons $b_j \in BS_i$ that achieve the highest value in $J_{i,j}$. Of course, measurements with negative gain-cost utility, $J_{i,j} < 0$, are not selected.

Each beacon b_i receiving the *UpdateReq* message also computes ui_i, which will be used by the robot in *measurement distribution.* ui_i estimates how good it is to assign measurements to b_i, i.e. the expected improvement in the uncertainty of the local state of b_i if b_i integrates one measurement to each beacon $b_j \in BS_i$. Similarly, the updated information matrix for b_i that would result after integrating one measurement to each beacon $b_j \in BS_i$ is:

$$\Omega``_{i,t} = \overline{\Omega}_{i,t} + \sum_{j \in BS_i} \Omega_{i,j,t} \tag{6.9}$$

As above, if no measurement is integrated, the updated information matrix is $\Omega_{i,t-1}$. Thus, ui_i is computed as follows:

$$ui_i = \frac{1}{2} \log \left(\frac{|\overline{\Omega}_{i,t}|}{|\overline{\Omega}_{i,t} + \sum_{j \in BS_i} \Omega_{i,j,t}|} \right) \tag{6.10}$$

6.4.2 Integration of measurements

At this step, beacon b_i has already gathered one measurement to the robot and to each of the beacons selected in *measurement allocation*. Let MS_i be the set of gathered measurements. The next step is to integrate them. If beacon b_i is in the "initialization phase", it updates its PF with the measurements in MS_i. If b_i is in the "state vector phase", it integrates them in its local state and computes its update contribution as follows:

$$\xi_{i,t} = \sum_{j \in MS_i} H_{i,j,t}^T R^{-1} [z_{i,j} - h_{i,j}(ev_i) + H_{i,j,t} ev_i], \tag{6.11}$$

$$\Omega_{i,t} = \sum_{j \in MS_i} H_{i,j,t}^T R^{-1} H_{i,j,t}, \tag{6.12}$$

where $h_{i,j}(ev_i)$ and $H_{i,j,t}$ are respectively the predictions and Jacobians for each measurement in MS_i, either robot-beacon or inter-beacon measurement. Finally, b_i transmits an *UpdateResp* message to the robot with its contribution to the SEIF update ($\xi_{i,t}$ and $\Omega_{i,t}$) and to *measurement distribution* (ui_i).

6.5 Evaluation

This section is divided in two parts. The first one details the operation of **Scheme3**. The second discusses on its scalability and several parameters of the scheme.

The experiments shown in this section were performed with AMUSE aerial robotic manipulator and the experimental setting will be detailed in Section 7.2.2. In this section we concentrate on describing the operation and results of **Scheme3**. A more detailed description of the experiment setting and scenario can be found in Section 7.2.2.

6.5.1 Performance analysis

The result of **Scheme3** in one 3D SLAM experiment in XY (left) and 3D (right) views is shown in Fig. 6.3. Blue lines and stars represent respectively the ground truth trajectory and beacon locations. Red lines and triangles are the resulting estimates. The total number of measurements integrated at each iteration along the experiment is shown in Figure 6.4-a. At the beginning all beacons gathered all the measurements they were assigned with: in total $NM_{max} = 80$ between all the beacons. As the experiment advanced the beacon local states had lower and lower uncertainty and inter-beacon measurements became less and less informative. From $t = 108s$ on, some measurements achieved $J_{i,j} < 0$ –their reward was lower than the cost– and were not gathered anymore. In average the number of measurements per iteration in this experiment was 61, lower than NM_{max}. **Scheme3** ensures the given NM_{max} bound avoiding reward-cost inefficient measurements.

The evolution of beacon localization errors along the experiment is shown in Figure 6.4-b. The drawing for each beacon starts when its PF converged. The majority of the PFs converged between $t = 6s$ and $t = 16s$, shortly after the start of the experiment. The UAS localization errors in the three coordinates are shown in Figure 6.5. The red dashed lines represent the 3σ bounds showing the consistency of the estimations.

The cumulative number of inter-beacon measurements gathered by three beacons along the experiment is shown in Fig. 6.4-c. Similar curves were obtained for all beacons. The shape of each curve is a ramp with almost constant slope until the beacon

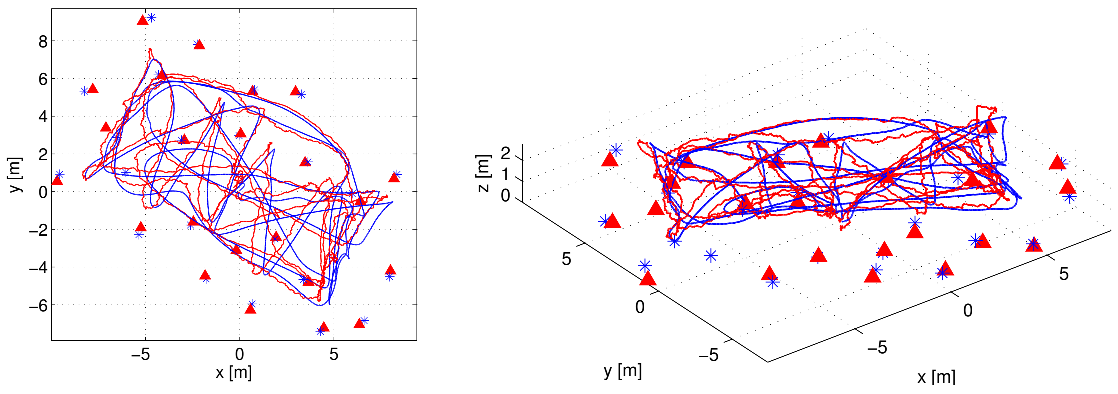


Figure 6.3: Results of **Scheme3** in a 3D SLAM experiment with AMUSE UAS: XY (left) and 3D views (right).

stops gathering measurements. This evolution is useful to analyze the performance of *measurement distribution* and *measurement allocation*. At the beginning each beacon gathers all the measurements it is assigned with. *Measurement distribution* assigns measurements to b_i proportionally to IG_i. More measurements are assigned to beacons with higher uncertainty. Hence, *measurement distribution* naturally balances the values of IG_i of all the beacons. Figure 6.4-d shows that the three beacons represented in Figure 6.4-c have similar evolution in IG_i. This can be observed for all beacons. As a result all beacons are assigned with a similar number of measurements, resulting in similar slopes in Figure 6.4-c. Beacon b_i gathers and integrates $z_{i,j}$ as long as $J_{i,j} > 0$. The uncertainty of b_i will be lower as it integrates more measurements. After a while, the measurements gathered by b_i will not satisfy $J_{i,j} > 0$ and it will stop taking measurements –slope becomes zero in Figure 6.4-c. Each beacon has its own different situation (number of neighbors, time of PF convergence, etc.) hence, they will reach zero-slope at different times.

Scheme3 can dynamically adapt to different values of NM_{max}. The previous experiment was repeated simulating that during interval $t \in [90, 105]$ the number of measurements was bounded by $NM_{max} = 30$, see Figure 6.6. In this case the robot RMS error was $0.516m$, very similar to that with $NM_{max} = 80$ along the entire experiment, which was $0.51m$. The difference in the map error was even smaller. in **Scheme3** selects the most informative measurements reducing the impact of changes

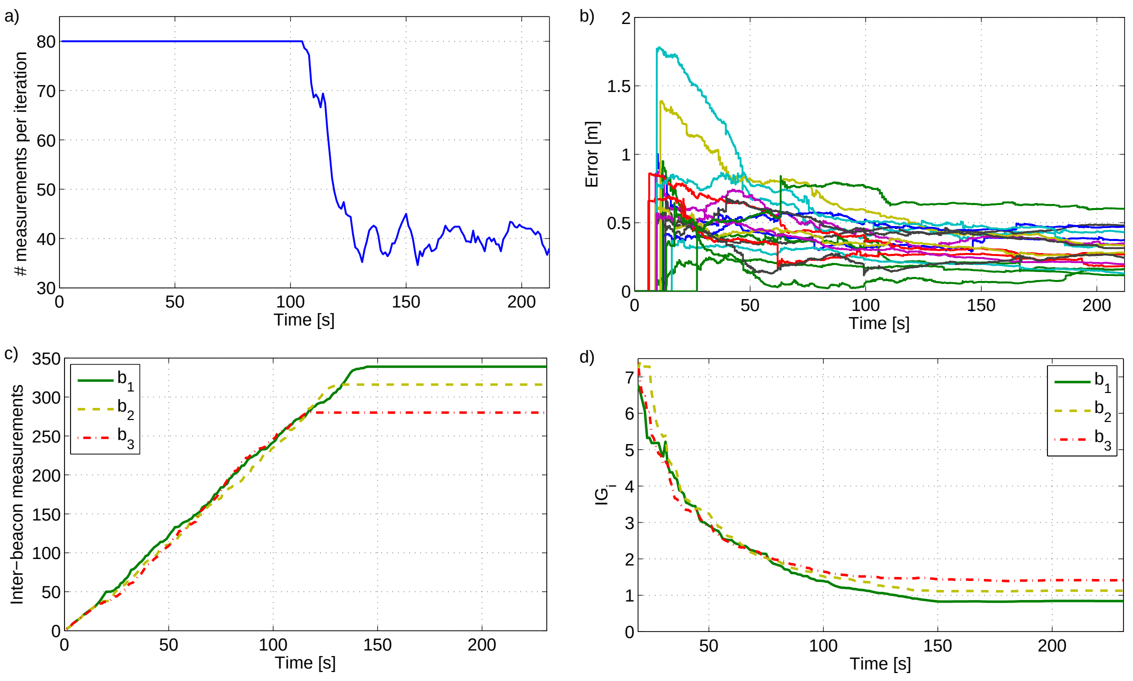


Figure 6.4: (a) Number of measurements integrated at each iteration along the experiment. (b) Evolution of beacon localization errors. (c) Number of measurements gathered by three beacons along the experiment. (d) Values of IG_i along the experiment for the beacons in c).

in NM_{max}. The only effect is that beacons stop gathering inter-beacon measurements –reach $J_{i,j} < 0$– later in the experiment. They keep gathering measurements and at the end of the experiment the number of measurements integrated are the same in both cases.

6.5.2 Discussion

In the following we discuss on the scalability of **Scheme3** and analyze the impact of transmission errors and of the parameters of the method: NM_{max} and α.

In Chapter 5 it was shown that robot-beacon distributed SEIF RO-SLAM preserves the scalability of SEIF. *Measurement distribution* involves only the beacons within the robot sensing region, whereas *measurement allocation*, performed by each beacon b_i, involves only the beacons within the sensing region of b_i. Thus, its computational

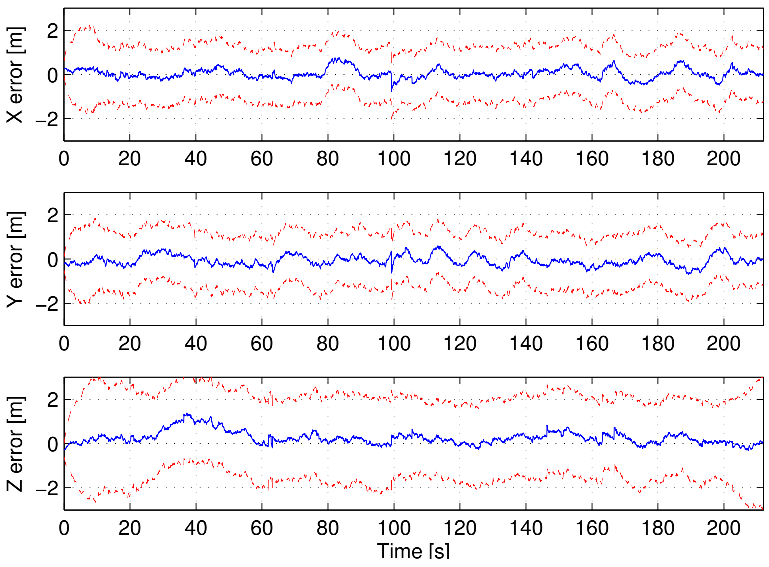


Figure 6.5: Evolution of UAS localization error in the experiment.

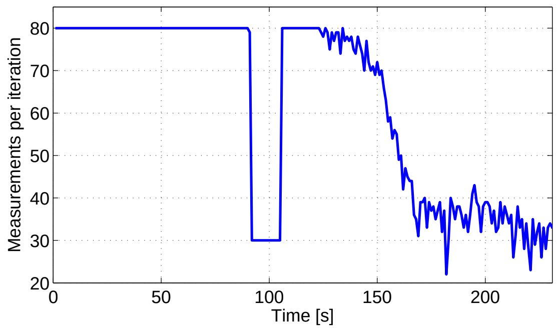


Figure 6.6: Experiment in Figure 6.4 taking $NM_{max} = 30$ during $t \in [90, 105]$.

complexity depends on the beacon density, not on the map size. Beacons are used as landmarks in RO-SLAM: highly inhomogeneous local beacon densities are not suitable in RO-SLAM. It is often more interesting if beacons are deployed in densities with some homogeneity, leading to constant time execution regardless of the map size.

NM_{max} is taken as an input to **Scheme3**. The performance of this scheme with different values of NM_{max} is summarized in Table 6.1. The measurements from all the experiments were logged and **Scheme3** was offline executed with NM_{max}. The integration of measurements is critical for PF convergence and low values of NM_{max} decelerate PF convergence. On the other hand, the estimation accuracy was only slightly affected, which is attributed to its capability to select informative measurements. With $NM_{max} = 80$ the average number of measurements actually integrated in SLAM was similar to that with $NM_{max} = 60$. The explanation is the role of α. In the experiments all measurements are assumed to have the same cost. Thus, α acts as a threshold since measurement $z_{i,j}$ is gathered only if $J_{i,j} = r_{i,j} - \alpha c_{i,j} > 0$.

	$NM_{max} = 40$	$NM_{max} = 60$	$NM_{max} = 80$
Map RMS error [m]	0.35	0.346	0.34
Robot RMS error [m]	0.52	0.51	0.51
PF convergence times [s]	15.8	9.5	5.7
# of measurements/iteration	40	56.5	61.7

Table 6.1: Average performance of **Scheme3** with different values of NM_{max}.

The performance of **Scheme3** with different values of α is summarized in Table 6.2. α allows setting **Scheme3** to prevent integrating measurements that are not very informative. With $\alpha = 1.5$, almost all measurements satisfy $J_{i,j} > 0$ and the number of measurements integrated in SLAM is almost NM_{max}. With $\alpha = 15$, many measurements do not satisfy $J_{i,j} > 0$ soon in the experiments and in average only 49.3 measurements were integrated per iteration. These were the two extremes in the range of α, an intermediate value $\alpha = 7.5$ was used. Despite the difference in the number of measurements, the value of α affects accuracy very slightly as shown in Table 6.2.

	$\alpha = 1.5$	$\alpha = 7.5$	$\alpha = 15$
Map RMS error [m]	0.34	0.34	0.37
Robot RMS error [m]	0.51	0.51	0.52
PF convergence times [s]	5.7	5.7	5.9
# of measurements/iteration	78.9	61.7	49.3

Table 6.2: Average performance of **Scheme3** with different values of α.

Scheme3 needs communication between the robot and beacons. In this sense the transmission errors in sensor networks cannot be ignored. Its performance assuming different PRR levels is summarized in Table 6.3. Our method explicitly considers PRR in the estimation of IG_i and assigns more measurements to the beacons that have better link quality with the robot. As expected, it exhibits good robustness to

PRR. Even with PRR=40% –transmission error rate of 60%– its performance is very slightly perturbed.

	PRR=40	PRR=60	PRR=80	PRR=100
Map RMS error [m]	0.4	0.37	0.35	0.35
Robot RMS error [m]	0.57	0.53	0.52	0.51
PF convergence times [s]	9.6	7.1	6.4	5.7
# of measurements/iteration	44.8	51.4	57.1	61.7

Table 6.3: Average performance of **Scheme3** with different PRR levels.

6.6 Conclusions

This chapter presented **Scheme3**, a scalable robot-beacon distributed RO-SLAM scheme for resource-constrained operation. The objective is to improve SLAM performance while meeting a given resource consumption bound expressed as the maximum number of measurements that can be integrated in SLAM per iteration. In our problem the number of measurements is a good metric for resource consumption since it directly impacts the sensing channel capacity used, the beacon energy consumed and the computational capacity employed, among others.

Scheme3 efficiently combines a distributed SEIF SLAM method, that integrates robot-beacon and inter-beacon measurements, together with a distributed information-driven tool that selects the measurements to be integrated in SLAM balancing uncertainty improvement and resource consumption. The scheme has a robot-beacon distributed approach where beacons actively participate in measurement selection, gathering and integration in SLAM. Our scheme ensures resource-constrained operation with static or dynamic bounds, showing significant flexibility. It achieves higher accuracy and lower beacon initialization times than conventional SLAM methods. Besides, it can be executed in constant time regardless of the map size.

Its performance was evaluated in 3D SLAM experiments. Robustness analysis confirmed its stable and predictable performance against transmission errors and different values of its parameters.

Chapter 7

Experiments

7.1 Introduction

Previous chapters of this document shown simulations and first experiments of the different schemes proposed in this PhD Thesis. This chapter will evaluate the performance of these schemes in real experiments with aerial robots in two different scenarios.

This chapter is structured as follows. Section 7.2 describes the experimental settings and specifications in the preliminary indoor scenario and the final outdoor scenario. Sections 7.3, 7.4 and 7.5 show the results of each of the three proposed schemes in the outdoor scenario. Section 7.6 compares and discusses the performance of the proposed schemes. The chapter ends with conclusions in Section 7.7.

7.2 Experimental settings

7.2.1 Preliminary scenario

A set of preliminary experiments were performed in controlled conditions in the CATEC[1] and ETHZ[2] testbeds in the scope of the EuRoC[3] project (European Robotics

[1]Center for Advanced Aerospace Technologies: `http://www.catec.aero/en`
[2]Swiss Federal Institute of Technology in Zurich: `https://www.ethz.ch/en.html`
[3]`http://www.euroc-project.eu/`

Challenge). The aerial platform used in these experiments was the AscTec Neo, an hex-rotor helicopter with 9” blades, see Figures 7.1 and 7.2. The platform is equipped with an AscTec VI-Sensor, a stereo camera looking in frontal-downward direction, with additional cameras with wide field of view (one looking upwards and one downwards) and with an onboard computer, the Intel NUC with an Intel Core i7-5557U processor.

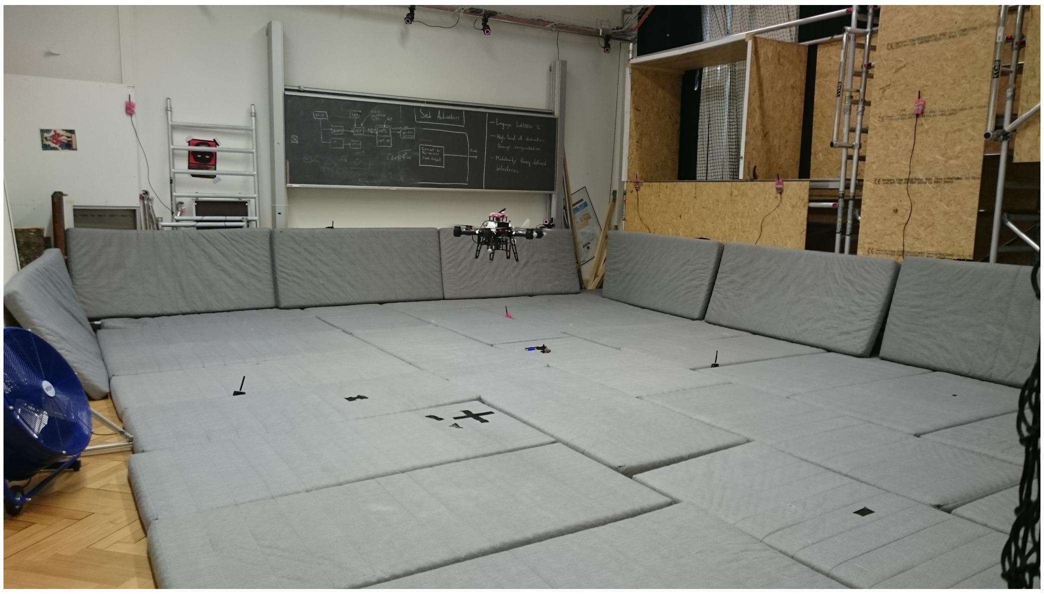


Figure 7.1: AscTec Neo flying in an experiment performed in the ETHZ testbed.

A total of 14 beacons were deployed at random locations in both indoor testbeds. Each beacon was comprised of a *RaspberryPi* running Linux connected through USB to a Nanotron nanoPAN 5375 range sensor and to a WiFi USB adapter, all powered by an external battery (see Appendix B). One range sensor was mounted on top of the AscTec Neo, see Figure 7.2.

The UAS odometry was obtained through the stereo camera images. Visual odometry computes associations between features detected in consecutive images and fits a transformation matrix that models the 6DoF change in the UAS pose between both images. The ground truth in indoor scenarios was provided by a Vicon MoCap motion capture system with sub-millimeter accuracy.

These controlled experiments were taken as preliminary tests to evaluate the performance of the proposed schemes in 3D SLAM. The small size of the testbeds

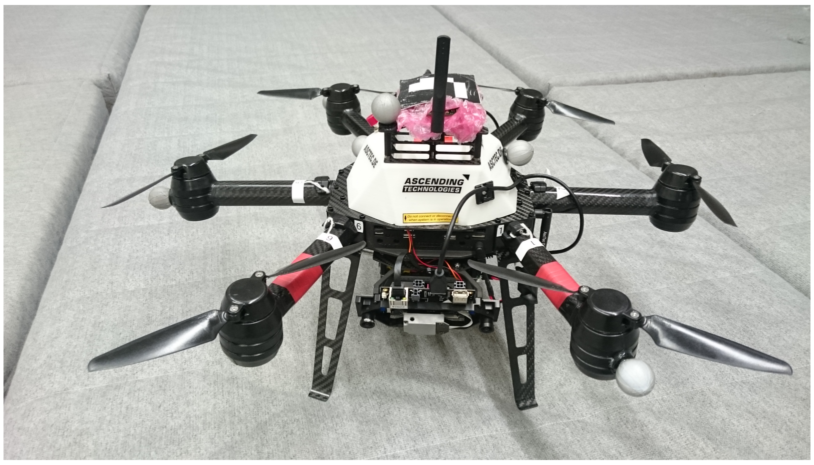


Figure 7.2: AscTec Neo with a Nanotron range sensor.

insufficient to perform scalability tests, the high beacon density insufficient to make evident the advantages of the proposed schemes and the high number of non-Gaussian outliers in range measurements originated by the influence of the scenario obstacles recommended to perform experiments in larger outdoor scenarios.

7.2.2 Outdoor scenario

The outdoor experiments were performed with AMUSE UAS, an octorotor developed in the UE-FP7 ARCAS[4] project for maintenance and repairing of industrial facilities (Heredia et al., 2014), see Figure 7.3. Maintenance of industrial facilities is currently performed using human inspection or sensor nodes that gather measurements for process monitoring and anomaly detection. In these complex scenarios GPS is often unavailable or has bad quality. UAS are suitable tools for confirming and eventually repairing the anomalies detected but they require accurate localization. The proposed RO-SLAM schemes are very interesting in this application. Besides the typical technological constraints of UAS and beacons, in these scenarios there are often a high number of sensors and wireless devices involving significant bandwidth limitations. Besides, the energy consumed by nodes –beacons– is constrained in order to avoid frequent battery replacements.

[4]`http://www.arcas-project.eu/`

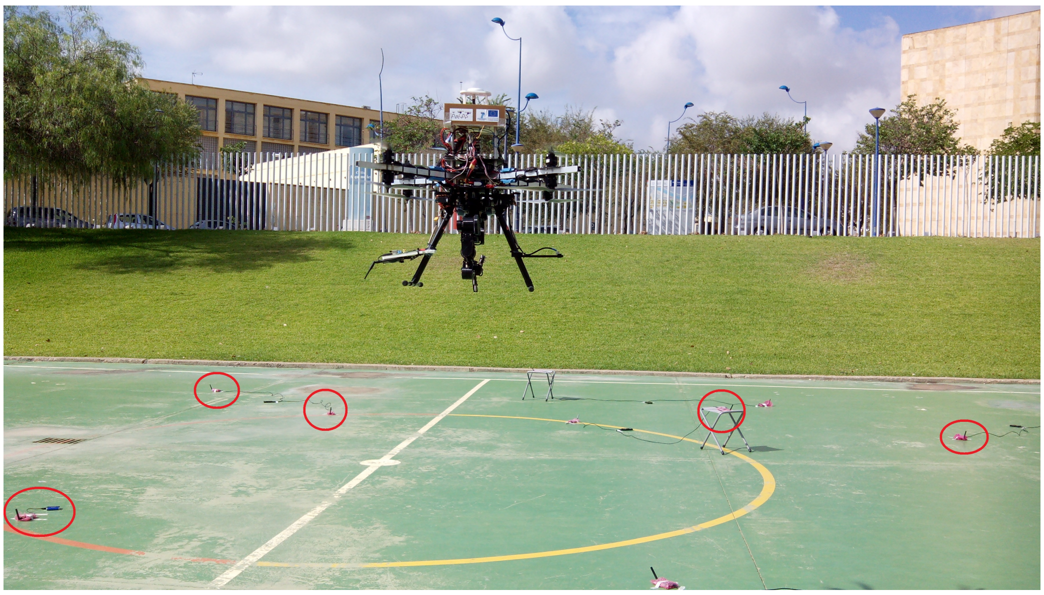
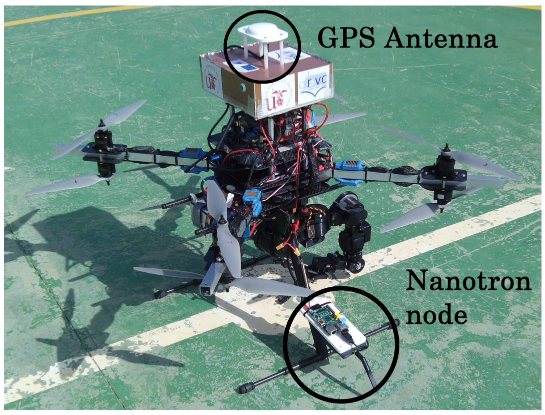


Figure 7.3: AMUSE UAS flying during one experiment at the School of Engineering of Seville. Beacons are marked with red circles.

A total of 24 beacons were deployed at random locations and different heights in a 20x20 m scenario (beacons are marked in Figure 7.3). Each beacon was comprised by the same elements ad described before: a *RaspberryPi* running Linux, a Nanotron nanoPAN 5375 range sensor and a WiFi USB adapter, all powered by a battery, see Figure 7.4-right. The performance of Nanotron sensors in outdoors was characterized experimentally (see Appendix B). Each beacon runs an independent ROS (Robot Operating System) node. The ROS node implements the beacon algorithm, gathers range measurements with the Nanotron range sensor using an ad-hoc developed ROS driver and communicates with the other beacons using WiFi. One beacon was mounted on the landing skid of AMUSE, see Figure 7.4-left.

In these experiments AMUSE was in manual flight. The objective was not to use the proposed scheme for real-time navigation. AMUSE is equipped with a *Novatel OEM6* RTK GPS unit with $2cm$ accuracy, which is used only as ground truth for accuracy assessment. SLAM provides the generated map and robot location in a local coordinate frame.



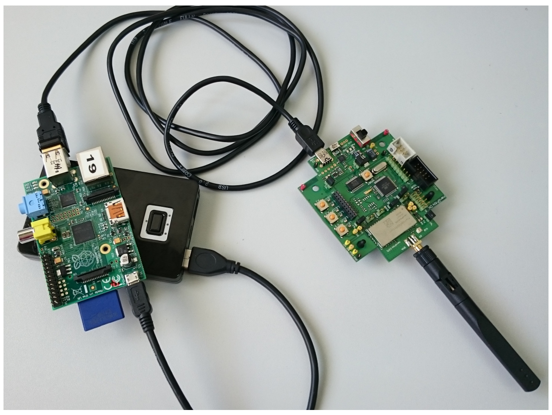


Figure 7.4: (Left) Picture of AMUSE octorotor during one experiment. (Right) Picture of a beacon comprised of a Nanotron range sensor connected through USB to a *RaspberryPi* module powered by a battery.

The UAS odometry was obtained from its Inertial Measurement Unit (IMU). However, IMU-based odometry is often too noisy to be used directly in SLAM as shown below. The adopted UAS kinematic model is presented in the following.

UAS kinematic model

A quadrotor can be considered as a rigid body and its dynamics can be derived using the Newton–Euler formalism. The dynamic equations of the quadrotor at the center of mass in body coordinates can be expressed in matrix form as:

$$m\dot{V} + \Omega \times (mV) = F, \tag{7.1}$$

$$I\dot{\Omega} + \Omega \times (I\Omega) = \tau, \tag{7.2}$$

where V is the UAS linear speed and Ω is the angular speed, both in body frame coordinates. F and τ are respectively the external forces and torques applied to the quadrotor, while m is the mass of the system. Focusing on the translational equations of the quadrotor model, (7.1) can be developed by:

$$\ddot{x} = (\cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi)\frac{T}{m}, \tag{7.3}$$

$$\ddot{y} = (\sin\psi \sin\theta \cos\phi - \cos\psi \sin\phi)\frac{T}{m}, \tag{7.4}$$

$$\ddot{z} = -g + (\cos\theta \cos\phi)\frac{T}{m}, \tag{7.5}$$

where $[\phi, \theta, \psi]$ are the Euler angles and T the total thrust generated by the rotors.

The model presented by Eqs. (7.3-7.5) is a simplification of the complex dynamic interactions. To enforce more realistic behavior of the quadrotor, drag force generated by the air resistance and other several aerodynamic effects should be included in the model (Bouabdallah and Siegwart, 2007). These aerodynamic effects are difficult to be measured or estimated in outdoor experiments. Therefore, these effects could not be included in the model used for the position estimation.

However, the position estimation using the above model achieved poor results due to the inability to measure the rotors drag effects and the aerodynamic effects in outdoor experiments. In other works (Leishman et al., 2014; Allibert et al., 2014), the authors demonstrated improvements in estimation accuracy obtained through a proper understanding of accelerometer measurements and modeling rotor drag effects. These works present results of experiments in indoor scenarios with a controlled environment and it could not be compared with outdoor experiments results.

Figure 7.5 shows the IMU-based odometry estimation in X and Y axis compared with the RTK GPS measurements obtained, which were taken as ground truth. It can be seen that IMU-based estimation is not a good estimation, as expected due to the influence of noise. In order to emulate the integration of a better odometry, we use a few beacons as anchors for correcting the IMU estimation.

Figure 7.6 shows the sonar measurements compared to the RTK GPS altitude measurements, taken as ground truth. In general the sonar measurements follow very well the ground-truth. The occasional large errors in the height estimation of the sonar are caused by elements or obstacles in the scenario such as chairs. Figure 7.7 shows the experimental model of the error of sonar measurements. The measurement error model was approximated by a Gaussian with mean $\mu_s = 0.0447m$ and standard deviation $\sigma_s = 0.09m$.

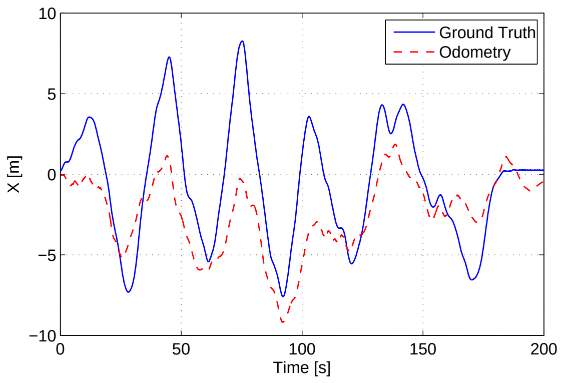


Figure 7.5: IMU-based odometry versus RTK GPS –assumed ground truth– in one experiment: (Left) X axis; (Right) Y axis.

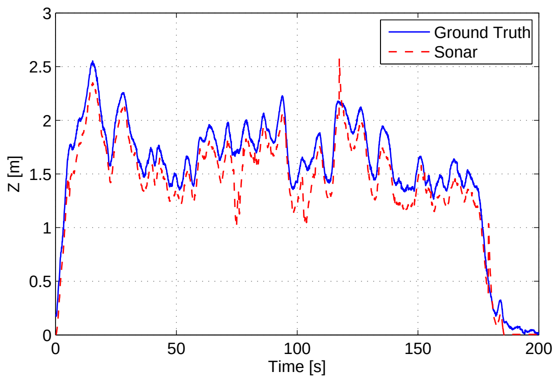


Figure 7.6: Sonar measurements compared to RTK GPS measurements taken as ground truth.

To overcome the high noise level in IMU-based odometry, like other works in 3D RO-SLAM, e.g. (Fabresse et al., 2014), we opted for using some beacons –5 in the experiments performed– as anchors for correcting the UAS localization using the UAS kinematic model and compensate for the high noise level in the IMU-based odometry. AMUSE is also equipped with a *Maxbotics MB1230* sonar for altitude correction.

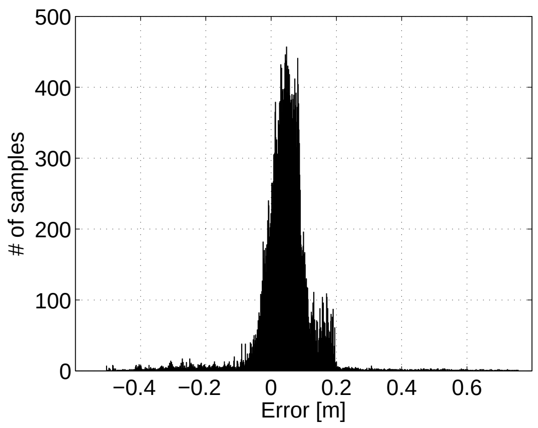


Figure 7.7: Experimental model of sonar measurement error.

7.3 Scheme1: RO-SLAM with dynamically configurable measurement gathering

The proposed **Scheme1** –see Chapter 4– was experimentally evaluated in sets of outdoor experiments using the AMUSE platform with different trajectories. In 3D SLAM with aerial robots, we need the highest accuracy in robot localization. Hence, in these experiments we avoided using the *Relaxed* mode of the SLAM Supervisor of **Scheme1**, recall Figure 4.1. Only *Mapping* and *Localization* modes were used in the experiments. The parameters of these modes used in these experiments are shown in Table 7.1.

Parameter	**Value**
T_1	0.1
NH_{MM}	1
P_{MM}	1
P_{LM}	0.005

Table 7.1: Parameters of **Scheme1** used in the real experiments.

Figure 7.8 shows the results of **Scheme1** in two different experiments in the outdoor scenario. Figure 7.9 shows the error in the estimations of the beacon locations for the EKF SLAM (left) and for the proposed **Scheme1** (right) in one of the experiments. The drawing of the error for each beacon starts when the beacon PF converges.

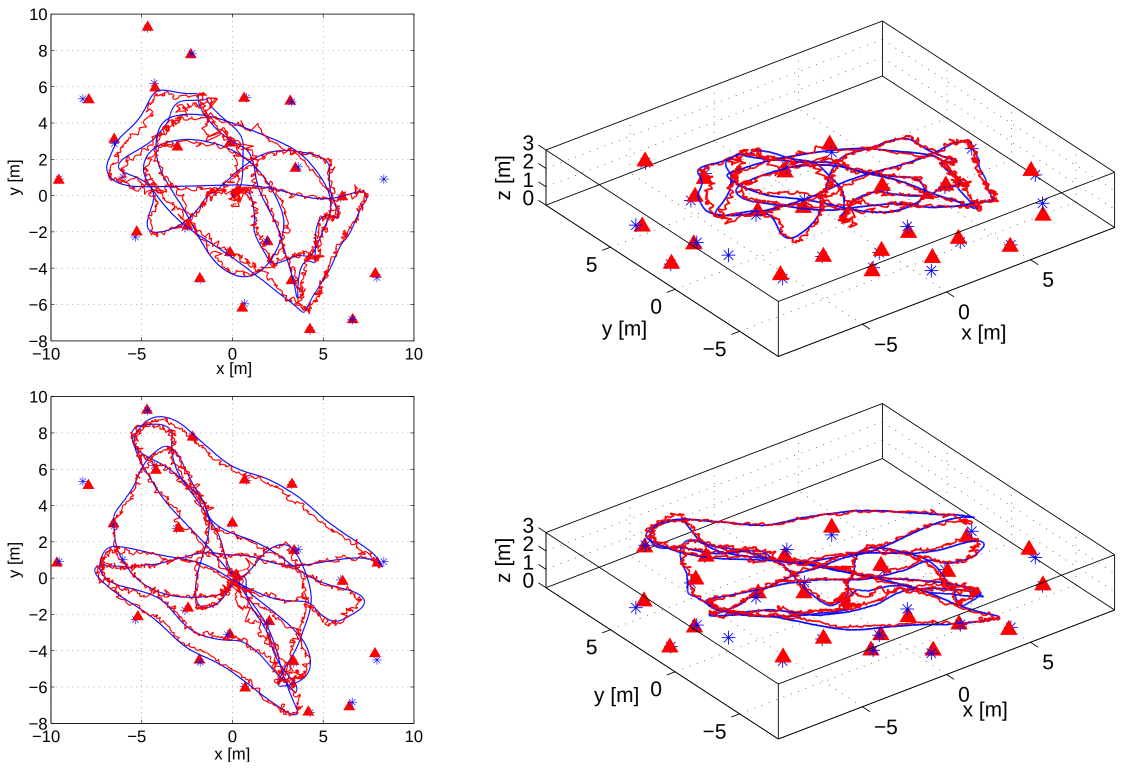


Figure 7.8: Result of executing **Scheme1** 3D SLAM in two outdoor experiments: (left) XY view, (right) 3D view. Blue lines and stars represent respectively the ground truth of the UAS trajectory and beacon locations. Red lines and triangles are the estimation of the filter.

The improvement in map accuracy leads consequently to improving the UAS localization. Figure 7.10 shows the evolution of the estimation error of the UAS using EKF SLAM and the proposed **Scheme1**. It is clear that the map and robot localization estimation errors obtained with **Scheme1** are significantly lower than those obtained with the EKF SLAM.

Table 7.2 compares the average performance of the EKF SLAM, the proposed **Scheme1** and a version of this scheme but without the supervisor module that stops the gathering of inter-beacon measurements, i.e. a full inter-beacon method, called "Full Int." in Table 7.2. All the methods were set with exactly the same parameters. When compared to the EKF SLAM, the proposed **Scheme1** largely decreases map building times, in 60%. This is a significant advantage to increase safety levels for UAS

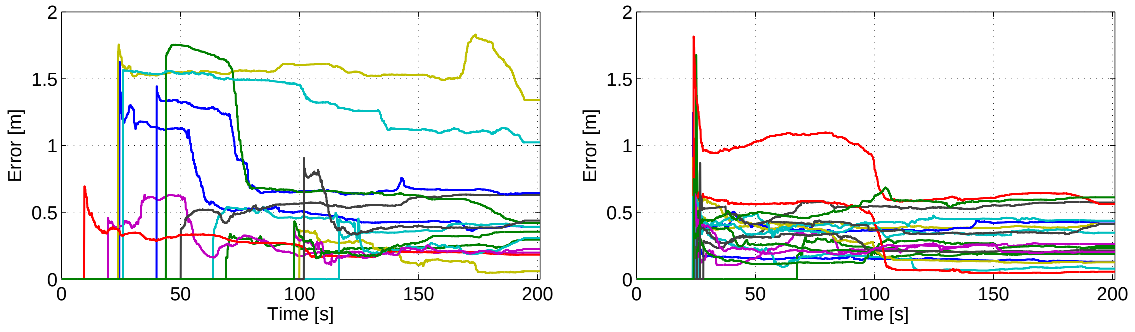


Figure 7.9: Estimation error of the beacon locations for the EKF SLAM (left) and the proposed **Scheme1** (right).

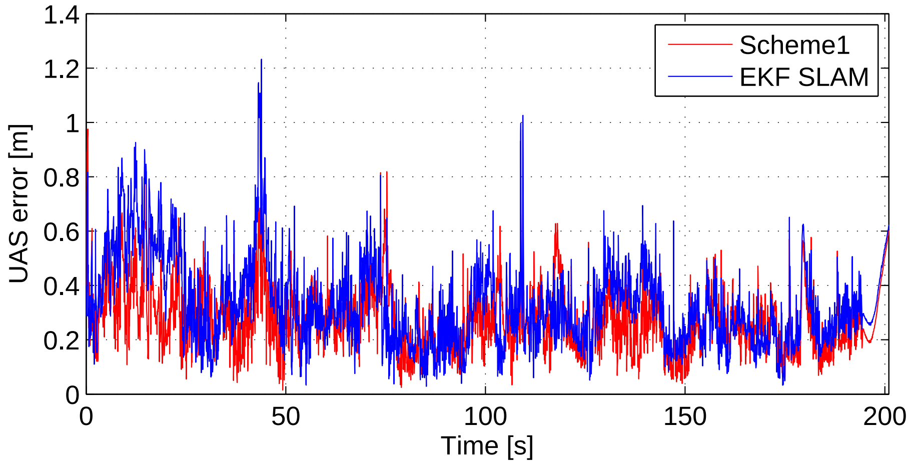


Figure 7.10: Evolution of the error in the UAS localization estimation in one experiment.

navigation in unstructured environments. It should be noticed that, despite the higher number of measurements, faster PF convergence times reduce the computational burden. Besides, the performance of **Scheme1** exhibits an improvement of the trade-off between accuracy and efficiency reducing the map and UAS estimation accuracies w.r.t. the EKF SLAM method in a 25% and 12%, respectively. The full inter-beacon scheme behavior is very similar to that of **Scheme1**. The accuracies and map building times are almost the same but the computer burden is 252% higher than **Scheme1**.

Using sensor network nodes has many advantages but one has to be aware of the energy consumption, since these nodes are supposed to be working unattended for

	EKF	**Scheme1**	Full int.
Map error [m]	0.45	0.34	0.335
UAS location error [m]	0.51	0.45	0.45
PF convergence times [s]	39.0	15.6	15.5
UAS CPU time/iteration [ms]	4.65	3.55	12.5

Table 7.2: Performance of **Scheme1** versus the traditional EKF SLAM method and full inter-beacon method.

long periods of time. Radio transmissions are responsible for the greatest part of the energy consumption. Sending and listening to the channel in order to receive radio messages consume the majority of the energy. Table 7.3 shows a comparison of the total number of measurements used and the average energy consumption of each beacon deployed in the scenario. **Scheme1** used more energy than EKF SLAM since each node gather higher number of measurements. However, it used only 68% more energy than the EKF while the full inter-beacon scheme used 715% more energy. Besides, taking more measurements leads to sending more radio messages and then using more bandwidth. Thus, **Scheme1** will use more bandwidth than EKF SLAM but still a lot less than the full inter-beacon RO-SLAM scheme. Thus, the proposed **Scheme1** has the quick beacon convergence and accuracy of the full inter-beacon scheme with a significant lower energy and bandwidth consumption.

	EKF	**Scheme1**	Full int.
# of measurements used	8024	13520	65343
Beacon energy consumption [J]	50.4	84.9	411.0

Table 7.3: Performance of the proposed **Scheme1** versus the EKF SLAM method and full inter-beacon method.

7.4 Scheme2: Distributed SEIF-based RO-SLAM with inter-beacon measurements

The proposed **Scheme2** was evaluated and compared with other methods in 20 sets of real experiments with different beacon settings and UAS trajectories. Figure 7.11

shows the result of **Scheme2** –proposed in Chapter 5– in one experiment in the outdoor scenario in XY view (left) and 3D view (right). The resulting average map and robot errors were 0.34 and 0.49 m, respectively. The errors in XY were 0.19 and 0.28 m. In Z, they were 0.29 and 0.35 m.

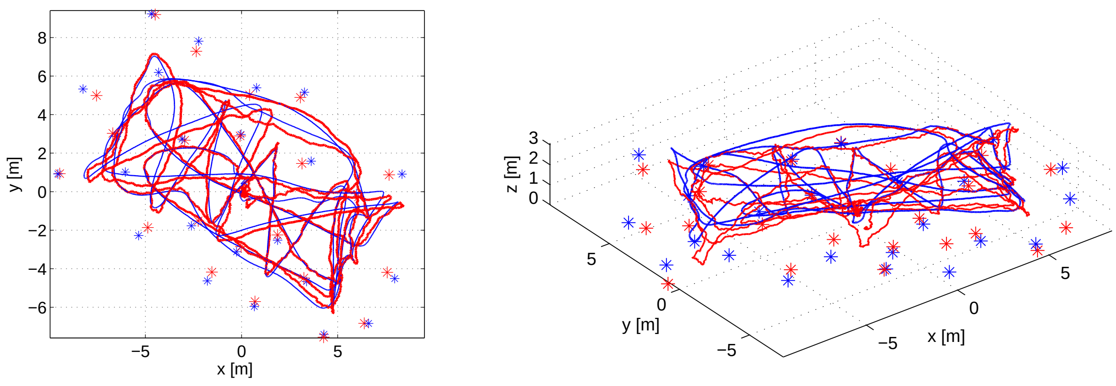


Figure 7.11: Result of **Scheme2** in one experiment: (left) XY view, (right) 3D view. The mean map and robot errors were 0.34 and 0.46 m, respectively. The mean errors in XY were 0.19 and 0.28 m.

Table 7.4 compares the proposed **Scheme2** with EKF SLAM and SEIF SLAM. All the methods were set with exactly the same parameters. This table shows the resulting average map and robot localization RMS errors and PF convergence times. The mean robot and map RMS errors for **Scheme2** were 13.5% and 28% lower than for SEIF SLAM, and 12% and 25.5% lower than for EKF SLAM. The mean PF convergence time for **Scheme2** was 62% lower than for SEIF SLAM. Besides, the robot CPU time for **Scheme2** was 35.3% lower than for SEIF SLAM. EKF SLAM also has lower robot CPU time than SEIF SLAM: it is efficient with low number of beacons but scales badly with the map size as described in Section 5.5.2.

Figures 7.12-left and 7.12-right show the evolution of the location error for each beacon in **Scheme2** and SEIF SLAM –EKF SLAM performs similarly to SEIF SLAM and is not shown. The drawing for each beacon starts when its auxiliary PF converged. In the proposed **Scheme2** beacon PFs converge significantly sooner and with lower error than in SEIF SLAM.

	EKF SLAM	SEIF SLAM	**Scheme2**
Map RMS error [m]	0.45	0.47	0.34
Robot RMS error [m]	0.51	0.52	0.45
PF conv. time [s]	39.0	39.2	14.8
robot CPU time (% of SEIF)	73	100	64.7

Table 7.4: Performance comparison of EKF SLAM, SEIF SLAM and the proposed **Scheme2**.

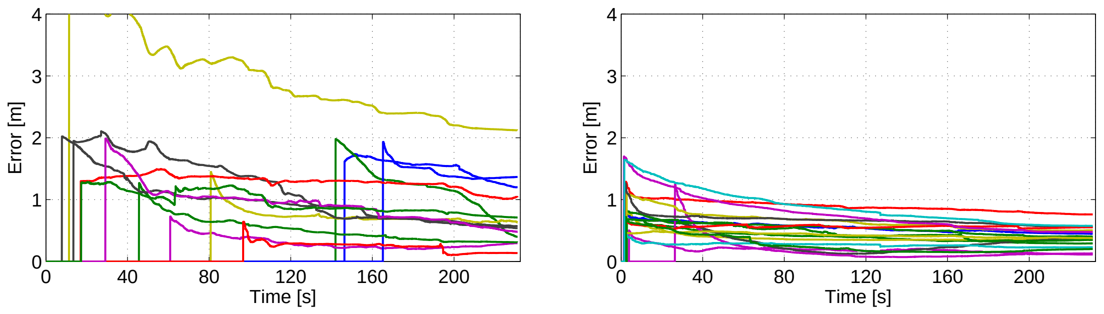


Figure 7.12: Evolution of the location error for each beacon in SEIF SLAM (left) and in the proposed **Scheme2** (right).

Scheme2 exploits beacon capabilities involving higher consumption of beacon resources than conventional methods. Table 7.5 shows the average energy consumed by the beacons in the experiments. It was estimated with the number of measurements gathered and the number of messages interchanged in the experiments, using the power consumption characteristics in the manufacturers data-sheet. The beacons energy consumption using **Scheme2** is almost 7 times higher than with EKF SLAM and SEIF SLAM. It seems interesting to develop strategies to reduce the number of inter-beacon measurements without losing their advantages: map and robot accuracy and fast map initialization.

EKF SLAM	SEIF SLAM	**Scheme2**
50.4	50.7	345.7

Table 7.5: Average beacon energy consumption [J].

7.5 Scheme3: Resource-constrained SEIF-based RO-SLAM with inter-beacon measurements

The proposed **Scheme3** –see Chapter 6– was evaluated and compared with other methods in 20 sets of real experiments with different beacon settings and UAS trajectories. Figures showing the performance of this scheme in the outdoor scenario were placed in Section 6.5.

Next, the proposed **Scheme3** will be compared to different methods in order to fully evaluate its performance. Method *M1* is a conventional SEIF SLAM scheme that integrates only robot-beacon measurements. Method *M2* is the distributed SEIF SLAM also known as **Scheme2** of this PhD Thesis. It integrates robot-beacon and all inter-beacon measurements. Method *M3* is *M2* combined with a tool that selects the NM_{max} measurements that best improve the uncertainty in the global state: this tool is necessarily centralized at the robot. In the proposed distributed scheme each beacon selects the best measurements to improve its local uncertainty. Comparing with method *M3* allows evaluating how far our distributed measurement selection is from the centralized selection. The data from the sets of experiments was logged and the four methods were executed offline with the same parameters. Their performance is compared in Table 7.6, which analyzes robot and map RMS errors, convergence times of auxiliary PFs, number of measurements actually integrated per iteration, average energy consumed by beacons and average robot CPU time consumed evaluated in percentage w.r.t. that of *M1*. Recall that the number of measurements integrated per iteration is proportional to the energy consumed by beacons (shown in the table) and to the beacon computational time required for measurement integration (not shown in the table).

M1 does not integrate inter-beacon measurements and hence had the poorest errors and PF convergence times. *M2* integrates inter-beacon measurements, which significantly reduces PF convergence times –62%– and map and robot RMS errors –28% and 13.5%, respectively– over *M1*. On the other hand, *M2* gathered and integrated 586% more measurements, which largely increased beacon energy consumption. *M2* distributes computation between the robot and beacons and hence reduces the robot

	M1	*M2*	*M3*	**Scheme3**
Map RMS error [m]	0.47	0.34	0.35	0.35
Robot RMS error [m]	0.52	0.45	0.46	0.47
PF convergence times [s]	39.2	14.8	15.6	15.7
# of measurements/iteration	35.4	242.9	80	61.7
Beacon energy consumption [J]	50.4	345.7	113.8	87.8
Robot CPU time (% of *M1*)	100	64.7	265.5	60.1

Table 7.6: Comparison of the proposed **Scheme3** versus methods *M1*, *M2* and *M3*.

CPU times over *M1*. *M3* is *M2* combined with a centralized tool that selects the NM_{max} measurements that best improve the uncertainty of the global state. *M3* integrated 80 measurements per iteration, 67% lower than *M2*, and achieved similar RMS errors (difference $< 3\%$) and PF convergence times ($< 6\%$). However, it required much larger robot CPU times: it uses the information matrix of the global state to select the most informative measurements and computing determinants has $O(n^3)$ complexity.

The proposed **Scheme3** obtained similar RMS errors, PF convergence times and robot CPU times to *M2* requiring 75% less measurements and 75% lower beacon energy consumption. Besides, our scheme achieved similar RMS errors and PF convergence times to *M3* but required 23% less measurements (and beacon energy consumption). Each beacon uses the information matrix of its local state for measurement selection, requiring 77% lower robot CPU burden than *M3*. Besides, in our scheme each beacon maintains a local version of its map, which can be useful in some cases. Once beacons have built their local map, they can transmit it to any robot, which can immediately recover the full map applying map-joining techniques.

7.6 Discussion

A comparison of the average performance of the three proposed schemes in the outdoor experiments is presented in Table 7.7. The proposed schemes are compared between each other and with EKF SLAM, taken as baseline for performance comparison. All four methods use the same PFs for beacon initialization and all use exactly the same parameters.

	EKF SLAM	**Scheme1**	**Scheme2**	**Scheme3**
Map RMS error [m]	0.43	0.34	0.34	0.35
Robot RMS error [m]	0.51	0.45	0.45	0.47
PF conv. time [s]	39.0	15.6	14.8	15.7
Beacon energy cons. [J]	50.4	84.9	345.7	87.8
Robot CPU time (% of *EKF*)	100	93.4	88.6	73.6

Table 7.7: Performance comparison of EKF SLAM and the proposed **Scheme1**, **Scheme2** and **Scheme3**.

All the proposed schemes outperform the basic EKF SLAM. The improvement in accuracy and initialization times is mostly due to the integration of inter-beacon measurements. As it has been demonstrated through this PhD Thesis, these measurements help to accelerate the beacon initialization and the mapping accuracy, involving improvements in the robot localization estimation. Taking these measurements also has a cost in terms of CPU time, bandwidth and energy. We have presented different ways to limit the impact on these resources.

Scheme1 dynamically changes the measurement gathering rate and decides to take inter-beacon measurements or not. The beacon energy consumption is only 68% higher than in the EKF SLAM, but having a great improvement in convergence time and accuracy. **Scheme1** uses an EKF as SLAM filter, and this has some advantages. EKF does not need any approximation in order to obtain the estimation like SEIF. That is why it achieved the best performance. On the other hand, it is not as scalable as **Scheme2** and **Scheme3** with practical performance impact in large scenarios and maps.

Scheme2 distributes measurement gathering and part of the computation between the beacons surrounding the robot and uses a different SLAM filter to improve scalability. This helps to reduce the computational burden of the robot but still each beacon integrates all direct and inter-beacon measurements gathered. This scheme has the lowest PF convergence time and has the same accuracy as **Scheme1**, but at a high cost. It integrates too many inter-beacon measurements and then the beacon energy consumption is four times higher than that of **Scheme1**.

Scheme3 finally combines the distribution of **Scheme2** with the measurement gathering control of **Scheme1**. Furthermore, measurement gathering control is performed in a way more flexible than **Scheme1** and each beacon selects the measurements that it gathers. The PF convergence times and beacon energy consumption is very similar to those of **Scheme1**. The robot CPU time is lower but the accuracy is poorer than in **Scheme1**. The accuracy reduction is due to the approximations performed in the SEIF in order to efficiently obtain the estimation (mean of the state vector) from the information vector and the information matrix. This effect is lower in **Scheme2** because it integrates many measurements and compensates the effect. However, the accuracy is still better than in the EKF SLAM.

Table 7.8 shows a qualitative comparison of the proposed schemes and the EKF SLAM. This will help to understand the advantages and disadvantages of these schemes and the differences between them. Mostly the same characteristics as quantitative analysis are now evaluated. We used values from “low” to “very high”. Referring to map and robot accuracies and also to scalability, higher values are the preferred. On the other hand, PF convergence time and energy consumption aim to have lower values.

	EKF SLAM	**Scheme1**	**Scheme2**	**Scheme3**
Map accuracy	Low	High	High	High
Robot loc. accuracy	Low	High	High	Medium-High
PF conv. time	High	Low	Low	Low
Beacon energy cons.	Low	Medium	Very high	Medium
Scalability	Low	Low	High	High

Table 7.8: Qualitative comparison of EKF SLAM and the proposed **Scheme1**, **Scheme2** and **Scheme3**.

7.7 Conclusions

This chapter evaluated the performance of the proposed schemes in real experiments with aerial robots. The schemes were validated and compared with relevant methods in order to show its advantages and drawbacks. Two different scenarios were presented, extracting the same conclusions.

Finally, the proposed schemes were compared to each other and the results were discussed. All of them outperform EKF SLAM mainly due to the integration of inter-beacon measurements. Each of the schemes has its own way to deal with the drawbacks of integrating these inter-beacon measurements. **Scheme1** includes a central supervisor that dynamically selects the measurements that are integrated in SLAM. **Scheme2** distributes computation and measurement gathering with the surrounding beacons. **Scheme3** combines these two ideas and provide a distributed, flexible and scalable scheme.

Chapter 8

Conclusions and future work

8.1 Conclusions

This PhD Thesis described the design, integration, evaluation and validation of a set of schemes for accurate and efficient range-only simultaneous localization and mapping exploiting the cooperation between robots and sensor networks.

The first contribution of this PhD Thesis, **Contribution1**, is a general architecture for RO-SLAM with cooperation between robots and sensor networks. The adopted architecture has two main characteristics. First, it exploits the sensing, computational and communication capabilities of sensor network nodes. Both, the robot and the beacons actively participate in the execution of the RO-SLAM filter. Second, it integrates not only robot-beacon measurements but also range measurements between two different beacons, the so-called inter-beacon measurements. Most reported RO-SLAM methods are executed in a centralized manner in the robot. In these methods all tasks in RO-SLAM are executed in the robot, including measurement gathering, integration of measurements in RO-SLAM and the Prediction stage. These fully centralized RO-SLAM methods require high computational burden in the robot and have very poor scalability. The proposed architecture provides the framework to design RO-SLAM schemes that exploit the robot-sensor network cooperation in order to improve accuracy, efficiency and scalability.

Contribution2 proposes a RO-SLAM scheme with dynamically configurable measurement gathering, **Scheme1**. Integrating inter-beacon measurements in RO-SLAM significantly improves map estimation accuracy but involves high consumption of resources, such as the energy required to gather and transmit measurements, the bandwidth required by the measurement collection protocol and the computational burden necessary to integrate the larger number of measurements. The objective of this scheme is to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a centralized mechanism running in the robot that adapts measurement gathering. This scheme allows high flexibility, being able to implement a high number of methods by simply modifying the parameters of the SLAM Supervisor. Thus, it can be easily adapted to particular problems or applications.

Contribution3 is a distributed RO-SLAM scheme based on the SEIF, **Scheme2**. This scheme reduces the increment in resource consumption resulting from the integration of inter-beacon measurements by adopting a distributed SLAM filter in which each beacon is responsible for gathering its measurements to the robot and to other beacons and computing the SLAM Update stage in order to integrate its measurements in SLAM. Thus, it efficiently integrates inter-beacon measurements distributing the costs between the surrounding beacons. Moreover, this scheme inherits the scalability of the SEIF.

Contribution4 is a resource-constrained RO-SLAM scheme based on the distributed SEIF previously presented. This scheme, **Scheme3**, includes the two mechanisms developed in the previous contributions –measurement gathering control and distribution of RO-SLAM Update stage between beacons– in order to reduce the increment in resource consumption resulting from the integration of inter-beacon measurements. This scheme exploits robot-beacon cooperation to improve SLAM accuracy and efficiency while meeting a given resource consumption bound. The resource consumption bound is expressed in terms of the maximum number of measurements that can be integrated in SLAM per iteration. The sensing channel capacity used, the beacon energy consumed or the computational capacity employed, among others, are proportional to the number of measurements that are gathered and integrated in

SLAM. The scheme has a robot-beacon distributed approach where beacons actively participate in measurement selection, gathering and integration in SLAM. This scheme ensures resource-constrained operation with static or dynamic bounds, showing significant flexibility. It achieves higher accuracy and lower beacon initialization times than conventional SLAM methods. Besides, it can be executed in constant time regardless of the map size.

Finally, the performance of the schemes developed in this PhD Thesis have been analyzed and compared with each other and with existing works. The proposed schemes were validated in real experiments with aerial robots. It has been proved that it is operational to integrate inter-beacon measurements in RO-SLAM.

This PhD Thesis proved that the cooperation between robots and sensor networks provide many advantages to solve the RO-SLAM problem. One important limitation in sensor networks is the resource consumption. The proposed architecture allows the exploitation of the cooperation advantages. On the other hand, the proposed schemes give solutions to the resource limitation without degrading performance.

8.2 Future work

The research described in this PhD Thesis presents a set of schemes for RO-SLAM with robots and sensor networks. Nevertheless, these solutions should be considered as a first step in a long-term research effort. The application of the developed techniques and schemes for their use in applications environment generate new questions and open new lines of research. Current and future research lines derived from this PhD Thesis can be summarized as follows:

1. **Multi-robot RO-SLAM**. The schemes presented in this PhD Thesis considered only one single robot. However, it is becoming more common to have teams of robots collaborating to achieve a common goal instead of having only one. Many works in the recent years studied the multi-robot SLAM problem. It would be very interesting to apply these multi-robot techniques with the sensor networks cooperative tools developed in this PhD Thesis.

2. **Integration with other type of sensors**. Nowadays, mobile and aerial robots are usually equipped with a high number of different sensors. Lasers, cameras, IMUs and radio sensors provide measurements which could be combined to reach better accuracy in any situation. For example, radio range measurements are good for large-scale outdoors environments while cameras perform better in well illuminated small or medium sized scenarios.

3. **Implementation in large scale real experiments**. Some schemes developed in this PhD Thesis claim to be scalable. This scalability is proven theoretically and in simulation experiments. Unfortunately, we could not validate the scalability of the proposed schemes in real experiments, so it would be great to see how they perform in big scenarios with hundreds or thousands of beacons.

4. **Hardware improvement**. In this PhD Thesis the range sensors used for validation are TOA commercial sensors using their own protocols. Nevertheless, the future of sensor networks and ubiquitous systems rely on standard communication protocols like WiFi or Bluetooth. Nowadays, almost everybody has one or more devices with wireless connections. Thus, future real applications should obtain range measurements not only from specific radio range sensors but also from existing WiFi or Bluetooth signals.

5. **Security**. A real-world application needs to be secure. The schemes developed in this PhD Thesis do not consider techniques or protocols to protect the information exchanged between the robot and the sensor network nodes. The deployment of sensor network nodes in unattended environments makes them vulnerable to a high variety of potential attacks. Thus, it is of high relevance to protect the network with techniques or protocols that ensure data integrity and confidentiality.

6. **Robustness**. In the real world, range sensors could fail or lose their calibration. They can also be located at bad places for the electromagnetic point of view, i.e. surrounded by metallic structures. Beacons providing wrong measurements can critically perturb the SLAM estimation and are particularly harmful for beacon

initialization. It would be interesting to have tools which can detect when a certain beacon is giving bad measurements, and then calibrate it, ignore it or even replace it depending on the source of the errors.

Appendix A

CONET Integrated Testbed

A.1 Description of the testbed

The CONET Integrated Testbed (`http://conet.us.es`) (Jiménez-González et al., 2011) is a remote tool to assess and compare methods combining robot and sensor networks.

It is composed of Pioneer 3-AT robots and hundreds of sensor network nodes of different types (TelosB, Mica2, Iris, MicaZ, etc.). It also has Nanotron nanoPAN nodes, which provide better range measurements using TOA technologies (see Appendix B). The robot can self-localize using AMCL (Adaptive Monte-Carlo Localization) that integrates laser range measurements with low error ($< 0.01m$), which has been considered as ground-truth for the experiments.

The testbed uses an open and modular architecture and is installed since 2010 at the basement of the building of the School of Engineering of Seville (Spain). It is set in a room of more than 500 m^2 ($22m$ x $24m$). Figure A.1 shows a picture taken during the experiments.

A.2 Pioneer 3-AT mobile robot

The Pioneer 3-AT is a highly versatile four wheel drive robotic platform. These robots can reach speeds of .8 meters per second and carry a payload of up to 12 kg. In the

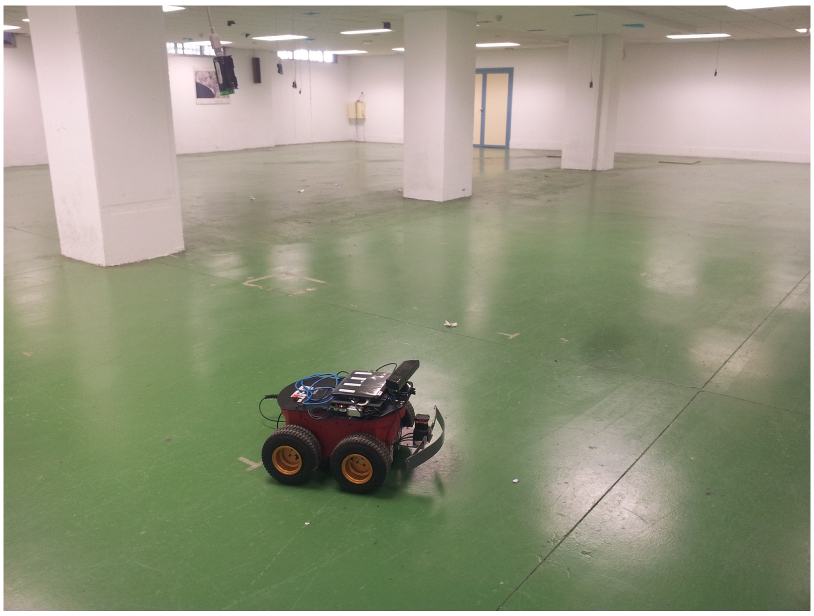


Figure A.1: Picture taken during the experiments carried out in the *CONET Robot-WSN Integrated Testbed*.

CONET Integrated Testbed they are equipped with extra sensors which makes it even more versatile. Each one has a Hokuyo laser range finder, a Microsoft Kinect, a GPS and an inertial measurement unit. The robots have a computer which integrates and manage all the sensors and they can also bring a sensor network node (Martinez-de Dios et al., 2014).

It has skid-steer configuration and then, the kinematic model of the robot in the experiments is the following:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} x_{t-1} + T_t V_t \sin \theta_{t-1} \\ y_{t-1} + T_t V_t \cos \theta_{t-1} \\ \theta_{t-1} + T_t \alpha_t \end{bmatrix}, \tag{A.1}$$

where $[x_t, y_t, \theta_t]^T$ is the robot state and V_t and α_t are, respectively, the odometry linear and steering velocities. T_t is the differential time between t and $t-1$.

Appendix B

Nanotron nanoPAN range sensors

Among others, the CONET Integrated Testbed also includes a network of Nanotron nanoPAN 5375 range sensors (`http://nanotron.com/EN/PR_ic_modules.php#03`).

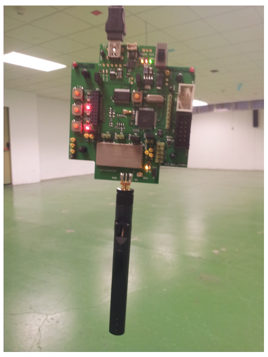


Figure B.1: Picture of one nanoPAN sensor in the testbed.

These sensors can measure the distance between two of them thanks to TOA technology. Table B.1 shows the specifications of the nanoPAN 5375 RF module.

The measurement error of these sensors can be approximated by a Gaussian with zero mean. Its standard deviation will depend on where they will be located. The manufacturer says that the standard deviation is $\sigma_m = 1m$ outdoors and $\sigma_m = 2m$ indoors. However, experimental characterization showed that these covariances are lower in reality. Figure B.2 shows both indoors and outdoors experimental models. The estimated standard deviations were finally $\sigma_m = 0.6m$ outdoors and $\sigma_m = 0.9m$ indoors.

Data rates	250 kbps, 1 Mbps
Ambient temperature range	-40 to +70 °C
Supply voltage	2.5 ± 0.2 V
TX current	typ. 210 mA
HC Ranging	(80 MHz, 1 Mbps)
RX sensitivity	typ. -89 dBm
RX current	typ. 51 mA
TX output power	typ. +20 dBm
TX power range	typ. 35 dB
LD Ranging	(80 MHz, 250 kbps)
RX sensitivity	typ. -95 dBm
RX current	typ. 69 mA
TX output power	typ. +20 dBm
TX power range	typ. 35 dB
R Comm	(22 MHz, 250 kbps)
RX sensitivity	typ. -96 dBm
RX current	typ. 34 mA
TX output power	typ. +20 dBm
TX power range	typ. 35 dB
Antenna load impedance	nom. 50 Ohm
SPI clock frequency	up to 24 Mbps
Real Time Clock frequency	32.768 kHz
Hardware accelerated encryption	128 bit
Certifications	FCC, ETSI

Table B.1: Product Specification nanoPAN 5375 RF Module.

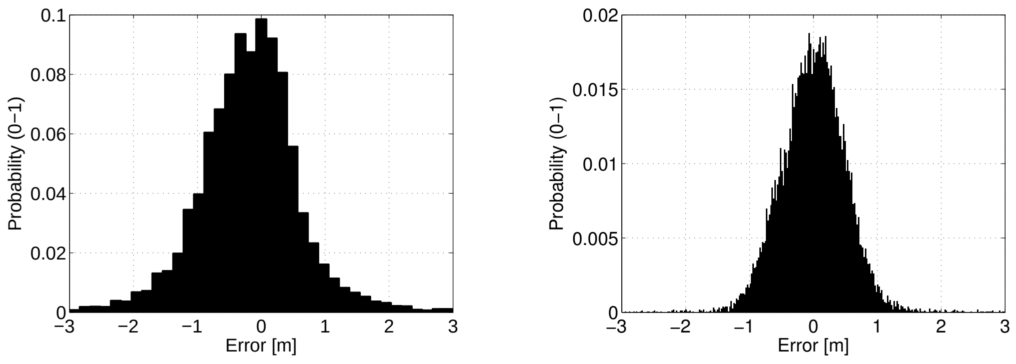


Figure B.2: Experimental model of the nanoPAN sensors in indoors (left) and outdoors (right).

Appendix C

Recursive Bayesian Filtering

C.1 Introduction

The objective of this Annex is not to be exhaustive in the description but to give a general overview that will be complemented with deeper and more focused descriptions in each of the required chapters.

The concept of *belief* is key in localization and mapping in probabilistic robotics (Thrun et al., 2005). In SLAM a belief reflects the system internal knowledge about the state of the target and the map. In localization, the position of the target can not be measured directly, instead the system must infer the position from data. The same occurs with the mapping problem. Therefore the true state is distinguished from its internal *belief*, or *state of knowledge* with regards to that state.

Probabilistic methods represent beliefs through conditional probability distributions. A belief distribution assigns a probability (or density value) to each possible hypothesis with regards to the true state. Belief distributions are posterior probabilities over state variables conditioned on the available data. We will denote belief over a state variable x_t by $bel(x_t)$, which is an abbreviation for the posterior:

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \tag{C.1}$$

This posterior is the probability distribution over the state x_t at time t, conditioned on all past measurements $z_{1:t}$ and all past controls $u_{1:t}$.

Occasionally, it is useful to calculate a posterior before incorporating z_t. Such a posterior will be denoted as follows:

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) \tag{C.2}$$

This probability distribution is often known as Prediction in the context of probabilistic filtering. This terminology reflects the fact that $\overline{bel}(x_t)$ predicts the state at time t based on the previous state posterior, before incorporating the measurement at time t, just after executing control action u_t. Calculating $bel(x_t)$ from $\overline{bel}(x_t)$ is called measurement update or simply Update.

The Bayes Filter algorithm is the most general way for calculating beliefs. This algorithm calculates the belief distribution bel recursively from measurements and controls actions. Algorithm C.1 shows the basic Bayes Filter in pseudo-algorithmic form. The Bayes Filter is recursive, that is, belief $bel(x_t)$ at time t is calculated from belief $bel(x_{t-1})$ at time $t-1$. Its input also includes the most recent control u_t and measurement z_t. Its output is the belief $bel(x_t)$ at time t. Algorithm C.1 only depicts a single step of the Bayes Filter algorithm: the *update rule*. This update rule is applied recursively.

Algorithm C.1 The general algorithm for Bayes filtering.

```
1: Algorithm Bayes Filter($bel(x_{t-1}), u_t, z_t$)
2: for all $x_t$ do
3:     $\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1}) bel(x_{t-1}) \delta x$
4:     $bel(x_t) = \eta p(z_t|x_t) \overline{bel}(x_t)$
5: end for
6: return $bel(x_t)$
```

The Bayes Filter algorithm has two essential steps. In Line 3, it processes the control action u_t. It does so by calculating a belief over the state x_t based on the prior belief over state x_{t-1} and the control action u_t. In particular, the belief $\overline{bel}(x_t)$ that the robot assigns to state x_t is obtained by the integral (sum) of the products of two distributions: the prior assigned to x_{t-1} and the probability that control u_t induces a transition from x_{t-1} to x_t. The similarity of this update step to Equation C.1 is evident. As mentioned above this step is called the motion update or Prediction.

The second step of the Bayes Filter is the so-called measurement update. In Line 4 the Bayes Filter algorithm multiplies belief $\overline{bel}(x_t)$ by the probability that measurement z_t may have been observed conditioned on state x_t. That is, being at state x_t, the probability of observing z_t. As will become apparent in the following when actually deriving the basic filter equations, the resulting product is generally not a probability, that is, it may not integrate to 1. Hence, the result is normalized, by virtue of the normalization constant η. This leads to the final belief $bel(x_t)$, which is returned in Line 6 of the algorithm.

To compute the posterior belief recursively the algorithm requires an initial belief $bel(x_0)$ at time $t = 0$ as boundary condition. If one knows the value of x_0 with certainty, $bel(x_0)$ should be initialized with a point mass distribution that centers all probability mass on the correct value of x_0 and assigns zero probability anywhere else. If one is entirely ignorant about the initial value x_0, $bel(x_0)$ may be initialized using a uniform distribution over the domain of x_0.

The Bayes Filter algorithm can only be implemented in the form stated here for very simple estimation problems. In particular, we either need to be able to carry out the integration in Line 3 and the multiplication in Line 4 in closed form, or we need to restrict ourselves to finite state spaces, so that the integral in Line 3 becomes a (finite) sum.

Since the Bayes Filter is not a practical algorithm and cannot be implemented on a digital computer, probabilistic algorithms use tractable approximations. There are two popular families of recursive state estimation techniques that are both derived from the Bayes Filter: Gaussian techniques, including Kalman Filters and its derivatives, which assume a Gaussian probabilistic distribution of the belief and; non-parametric filters, including Particle Filters, which approximate the belief by a finite number of samples of the state.

C.2 Kalman Filters

Since its development, Kalman Filters (KFs) (Thrun et al., 2005) have been the subject of extensive research and application. Their success is originated by their simplicity and robustness. KFs have become one of the most common methods used for localization and mapping in ubiquitous computing systems. KF is a parametric Recursive Bayesian Filter (RBF) that implements an optimal estimator that minimizes the covariance of the estimated error. It represents the belief $bel(x_t)$ at time t by the mean μ_t and the covariance Σ_t. These distributions are Gaussian if the three following properties, in addition to the Markov hypothesis, are satisfied:

1. The probability of the next state $p(x_t|u_t, x_{t-1})$ must be a linear function with additive Gaussian noise. This is represented as:

   $$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t, \tag{C.3}$$

   where x_t is the state vector and u_t the control action vector. A_t and B_t are matrices that represent the linear model of the system. ϵ is a Gaussian noise with zero mean and covariance Q_t.

2. The probability of the measurement $p(z_t|x_t)$ should be also linear with additive Gaussian noise.

   $$z_t = H_t x_t + \delta_t, \tag{C.4}$$

   where H_t is the matrix that represents the observation model of the sensor. δ is a Gaussian noise with zero mean and covariance R_t.

3. Finally, the initial distribution should be a normal distribution.

In a system modeled by (C.3) and (C.4) the operation of KFs is based on two basic stages: the Prediction stage and the Update stage. In the Prediction stage, an estimation of the system state for the next instant is obtained. On the other hand, the Update stage integrates the new measurements in order to improve the estimation of the state vector. Figure C.1 shows an scheme with the operation of the Kalman Filter.

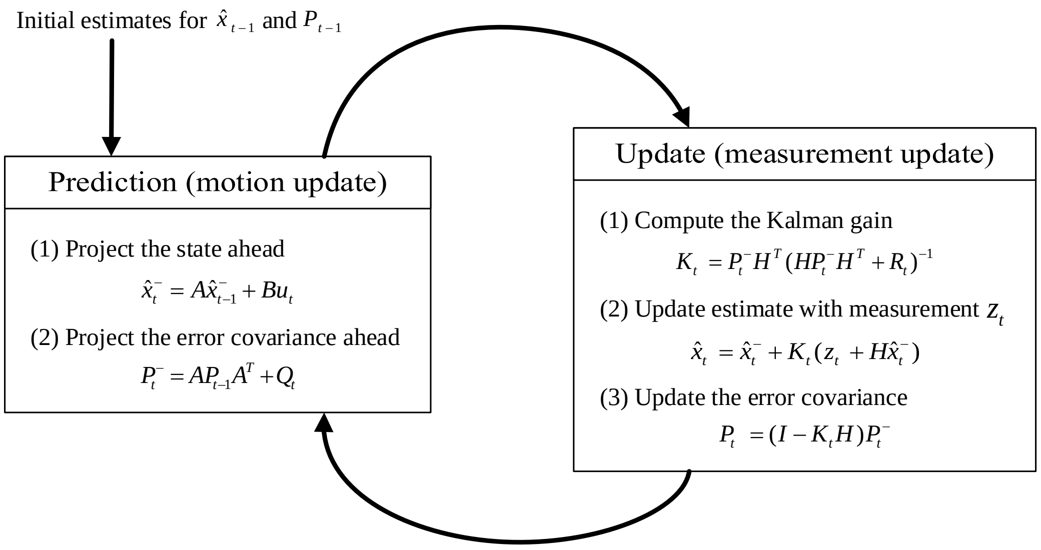

Initial estimates for $\hat{x}_{t-1}$ and P_{t-1}
Prediction (motion update)
(1) Project the state ahead
$\hat{x}_t^- = A\hat{x}_{t-1}^- + Bu_t$
(2) Project the error covariance ahead
$P_t^- = AP_{t-1}A^T + Q_t$
Update (measurement update)
(1) Compute the Kalman gain
$K_t = P_t^- H^T (HP_t^- H^T + R_t)^{-1}$
(2) Update estimate with measurement z_t
$\hat{x}_t = \hat{x}_t^- + K_t(z_t + H\hat{x}_t^-)$
(3) Update the error covariance
$P_t = (I - K_t H)P_t^-$

Figure C.1: KF algorithm.

However, the assumptions of linear state and measurements with additive Gaussian noise are not usually met in many applications. For example, the observation model of range measurements is nonlinear, and the robot's motion model is also nonlinear. The Extended Kalman Filter (EKF) overcomes the assumption of linearity. Here the assumption is that the next state probability and the measurement probabilities are governed by nonlinear functions f and h, respectively:

$$x_t = f(u_t, x_{t-1}) + \epsilon_t, \tag{C.5}$$

$$z_t = h(x_t) + \delta_t \tag{C.6}$$

Function f replaces matrices A and B in (C.3) and h replaces matrix H in (C.4). However, as f and h are non linear, the estimation is not Gaussian. The EKF computes an approximation of the estimation distribution. It represents that approximation with a Gaussian with its mean and covariance. Thus, the EKF can behave similarly to the Kalman Filter, except that the estimation distribution is not exact but approximate.

The key idea in the EKF is linearization. Given the nonlinear functions f and h, the linearization of the functions is obtained by Taylor expansion. In EKF the Jacobian matrices of f and h take the role of A and H:

$$F = \frac{\partial f}{\partial x}, \qquad H = \frac{\partial h}{\partial x} \tag{C.7}$$

If noise is also non-linear two more Jacobians are needed.

$$W = \frac{\partial f}{\partial \epsilon}, \qquad V = \frac{\partial h}{\partial \delta} \tag{C.8}$$

Figure C.2 summarizes the operation of the extended EKF algorithm.

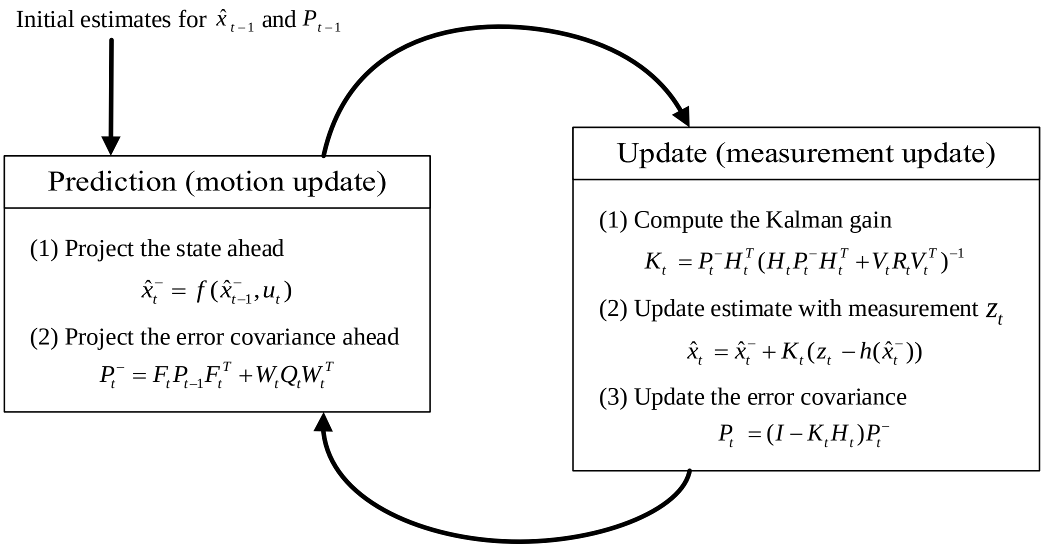

Initial estimates for $\hat{x}_{t-1}$ and P_{t-1}
Prediction (motion update)
(1) Project the state ahead
$\hat{x}_t^- = f(\hat{x}_{t-1}^-, u_t)$
(2) Project the error covariance ahead
$P_t^- = F_t P_{t-1} F_t^T + W_t Q_t W_t^T$
Update (measurement update)
(1) Compute the Kalman gain
$K_t = P_t^- H_t^T (H_t P_t^- H_t^T + V_t R_t V_t^T)^{-1}$
(2) Update estimate with measurement z_t
$\hat{x}_t = \hat{x}_t^- + K_t (z_t - h(\hat{x}_t^-))$
(3) Update the error covariance
$P_t = (I - K_t H_t) P_t^-$

Figure C.2: EKF algorithm.

C.3 Information Filters

Information Filters (IFs) are the dual to Kalman Filters. The belief is also represented by a Gaussian. Whereas in the Kalman Filter family of algorithms, Gaussians are represented by their moments, mean and covariance, in Information Filters, Gaussians are represented in their canonical form, comprised of an information vector (ξ) and an information matrix (Ω).

$$\Omega = \Sigma^{-1} \tag{C.9}$$

$$\xi = \Sigma^{-1}\mu \tag{C.10}$$

IFs have two main interesting properties that make them more suitable than KFs in some cases. First, the Prediction stage in IFs is additive and thus, easily implemented in a parallel manner, enabling distributed implementations where each entity gathers its measurements and integrates them in the IF. The second advantage is that although KFs and IFs have the same computational burden, the greater computational cost in KFs is in the Update stage while in IFs, the greater computational cost is in the Prediction stage. Hence, IFs are significantly more efficient than KFs in problems with simple Prediction stage and where many measurements are integrated in the Update stage.

Algorithm C.2 describes the Extended Information Filter (EIF). It uses the same robot motion and observation models than the EKF (f and h). F and H are then the Jacobians of these non-linear functions. Lines 2 to 4 perform the Prediction stage, and lines 5 to 6 implement the Update stage of the EIF. The main difference is in line 1, where EIF recovers the mean of the state from the information vector and information matrix. This step requires significant computational burden for large state vectors. Thus, different ways of dealing with this problem have been developed such as the Sparse Extended Information Filter (SEIF). SEIF and other characteristics of Information Filters are discussed in detail in Chapter 5.

Algorithm C.2 Extended Information Filter.

```
Require: $\xi_{t-1}, \Omega_{t-1}, u_t, z_t$
1: $\mu_{t-1} = \Omega_{t-1}^{-1}\xi_{t-1}$
2: $\bar{\Omega}_t = (F_t\Omega_{t-1}^{-1}F_t^T + Q_t)^{-1}$
3: $\bar{\xi}_t = \bar{\Omega}_t f(u_t, \mu_{t-1})$
4: $\bar{\mu}_t = f(u_t, \mu_{t-1})$
5: $\Omega_t = \bar{\Omega}_t + H_t^T R_t^{-1} H_t$
6: $\xi_t = \bar{\xi}_t + H_t^T R_t^{-1}[z_t - h(\bar{\mu}_t) + H_t\bar{\mu}_t]$
7: return $\xi_t, \Omega_t$
```

C.4 Particle Filters

Particle Filters (PFs) are nonparametric implementations of the RBFs. The key idea of PFs is to represent the belief $bel(x_t)$ by a set of random state samples. Instead of representing the distribution by a parametric form, PFs represent a distribution by a set of samples drawn from this distribution. Such a representation is approximate, but it is not parametric, and therefore can represent a much broader space of distributions than only Gaussians.

In PFs the samples of a posterior distribution are called *particles* and are denoted:

$$X_t := x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[M]} \tag{C.11}$$

Each particle $x_t^{[m]}$ (with $1 \leq m \leq M$) is a concrete instantiation of the state at time t, that is, a hypothesis as to what the true world state may be at time t. Here M denotes the number of particles in the particle set X_t, often a large number. PFs approximate the belief $bel(x_t)$ by the set of particles X_t. Ideally, the likelihood for a state hypothesis x_t to be included in the particle set X_t shall be proportional to its Bayes Filter posterior $bel(x_t)$:

$$x_t^{[m]} \sim p(x_t|z_{1:t}, u_{1:t}) \tag{C.12}$$

As a consequence of (C.12), the denser a subregion of the state space is populated by particles, the more likely is that the true state falls into this region. This property holds only asymptotically for $M \uparrow \infty$ for the standard Particle Filter algorithm. For finite M, particles are drawn from a slightly different distribution. In practice, this difference is negligible as long as the number of particles is not to small.

Just like all other RBFs algorithms, the PF algorithm maintains the belief $bel(x_t)$ recursively from the belief $bel(x_{t-1})$ one time step earlier. Since beliefs are represented by sets of particles, this means that PFs construct the particle set X_t recursively from the set X_{t-1}.

The basic PF algorithm is detailed in Algorithm C.3. Line 3 is like the Prediction stage of Kalman Filters, it estimates the change on the state based on the control inputs and a certain model. Line 4 is the *Importance Sampling*, and it calculates the *importance factor* also known as the *weight* of each particle. The set of weighted particles represent the Bayes filter posterior $bel(x_t)$. Lines 7 to 10 perform which is known as *resampling*. It consists in drawing with replacement the particles of the temporary set X_t with probability proportional to the weight of each particle. This step is the most important because it can remove the less probable particles and keep the most probable ones.

Algorithm C.3 Particle Filter.

```
Require: $X_{t-1}, u_t, z_t$
 1: $\bar{X}_t = X_t = \emptyset$
 2: for $m = 1$ to $M$ do
 3:     sample $x_t^{[m]} \sim p(x_t|z_{1:t}, u_{1:t})$
 4:     $\omega_t^{[m]} = p(z_t|x_t^{[m]})$
 5:     $\bar{X}_t = \bar{X}_t + \langle x_t^{[m]}, \omega_t^{[m]} \rangle$
 6: end for
 7: for $m = 1$ to $M$ do
 8:     draw $i$ with probability $\propto \omega_t^{[i]}$
 9:     add $x_t^{[i]}$ to $X_t$
10: end for
11: return $X_t$
```

Bibliography

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). A survey on sensor networks. *Communications magazine, IEEE*, 40(8):102–114.

Allibert, G., Abeywardena, D., Bangura, M., and Mahony, R. (2014). Estimating body-fixed frame velocity and attitude from inertial measurements for a quadrotor vehicle. In *Control Applications (CCA), 2014 IEEE Conference on*, pages 978–983. IEEE.

Andrade-Cetto, J. and Sanfeliu, A. (2004). The effects of partial observability in slam. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 1, pages 397–402. IEEE.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.

Baccour, N., Koubâa, A., Ben Jamaa, M., Youssef, H., Zuniga, M., and Alves, M. (2009). A comparative simulation study of link quality estimators in wireless sensor networks. In *Modeling, Analysis & Simulation of Computer and Telecommunication Systems, 2009. MASCOTS'09. IEEE International Symposium on*, pages 1–10. IEEE.

Baccour, N., Koubâa, A., Mottola, L., Zúñiga, M. A., Youssef, H., Boano, C. A., and Alves, M. (2012). Radio link quality estimation in wireless sensor networks: a survey. *ACM Transactions on Sensor Networks (TOSN)*, 8(4):34.

Bahl, P. and Padmanabhan, V. N. (2000). Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of* *the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee.

Bailey, T., Nieto, J., Guivant, J., Stevens, M., and Nebot, E. (2006a). Consistency of the ekf-slam algorithm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3562–3568. IEEE.

Bailey, T., Nieto, J., Guivant, J., Stevens, M., and Nebot, E. (2006b). Consistency of the ekf-slam algorithm. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3562–3568. IEEE.

Bailey, T., Nieto, J., and Nebot, E. (2006c). Consistency of the fastslam algorithm. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 424–429. IEEE.

Banatre, M., Marron, P., Ollero, A., and Wolisz, A. (2008). *Cooperating Embedded Systems and Wireless Sensor Networks*. Wiley.

Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons.

Bardella, A., Danieletto, M., Menegatti, E., Zanella, A., Pretto, A., and Zanuttigh, P. (2012). Autonomous robot exploration in smart environments exploiting wireless sensors and visual features. *annals of telecommunications-annales des télécommunications*, 67(7-8):297–311.

Blanco, J.-L., Fernandez-Madrigal, J.-A., and Gonzalez, J. (2008a). Efficient probabilistic range-only slam. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*, pages 1017 –1022.

Blanco, J.-L., Gonzalez, J., and Fernandez-Madrigal, J.-A. (2008b). A pure probabilistic approach to range-only slam. In *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA'08*, pages 1436 –1441.

Blumenthal, J., Grossmann, R., Golatowski, F., and Timmermann, D. (2007). Weighted centroid localization in zigbee-based sensor networks. In *Intelligent*

Signal Processing, 2007. WISP 2007. IEEE International Symposium on, pages 1–6. IEEE.

Boers, Y., Driessen, H., Bagchi, A., and Mandal, P. (2010). Particle filter based entropy. In *Information Fusion (FUSION), 2010 13th Conference on*, pages 1–8. IEEE.

Bouabdallah, S. and Siegwart, R. (2007). Full control of a quadrotor. In *Intelligent robots and systems, 2007. IROS 2007. IEEE/RSJ international conference on*, pages 153–158. IEEE.

Caballero, F., Merino, L., and Ollero, A. (2010). A general gaussian-mixture approach for range-only mapping using multiple hypotheses. In *IEEE Intl. Conf. on Robotics and Automation, ICRA 2010*, pages 4404 –4409.

Carlone, L., Aragues, R., Castellanos, J. A., and Bona, B. (2014). A fast and accurate approximation for planar pose graph optimization. *The International Journal of Robotics Research*, page 0278364914523689.

Castillo-Effer, M., Quintela, D. H., Moreno, W., Jordan, R., and Westhoff, W. (2004). Wireless sensor networks for flash-flood alerting. In *Devices, Circuits and Systems, 2004. Proceedings of the Fifth IEEE International Caracas Conference on*, volume 1, pages 142–146. IEEE.

Challa, S., Leipold, F., Deshpande, S., and Liu, M. (2005). Simultaneous localization and mapping in wireless sensor networks. In *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2005*, pages 81–87.

Chong, K. S. and Kleeman, L. (1999). Feature-based mapping in real, large scale environments using an ultrasonic array. *The International Journal of Robotics Research*, 18(1):3–19.

Clemente, A. D. S. B., Martínez-de Dios, J. R., and Baturone, A. O. (2012). A wsn-based tool for urban and industrial fire-fighting. *Sensors (Basel, Switzerland)*, 12(11):15009.

Corke, P., Hrabar, S., Peterson, R., Rus, D., Saripalli, S., and Sukhatme, G. (2006). Deployment and connectivity repair of a sensor net with a flying robot. In *Experimental Robotics IX*, volume 21 of *Springer Tracts in Advanced Robotics*, pages 333–343. Springer.

Davison, A. and Murray, D. (2002). Simultaneous localization and map-building using active vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):865–880.

Dellaert, F. and Kaess, M. (2006). Square root sam: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203.

Dissanayake, M., Newman, P., Clark, S., Durrant-Whyte, H. F., and Csorba, M. (2001). A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on*, 17(3):229–241.

Djugash, J., Singh, S., and Grocholsky, B. (2008). Decentralized mapping of robot-aided sensor networks. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 583–589. IEEE.

Djugash, J., Singh, S., Kantor, G., and Zhang, W. (2006). Range-only slam for robots operating cooperatively with sensor networks. In *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA*, pages 2078 –2084.

Durrant-Whyte, H. and Bailey, T. (2006). Simultaneous localization and mapping: part i. *Robotics & Automation Magazine, IEEE*, 13(2):99–110.

Engelhard, N., Endres, F., Hess, J., Sturm, J., and Burgard, W. (2011). Real-time 3d visual slam with a hand-held rgb-d camera. In *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden*, volume 180.

Estrada, C., Neira, J., and Tardós, J. D. (2005). Hierarchical slam: Real-time accurate mapping of large environments. *Robotics, IEEE Transactions on*, 21(4):588–596.

Eustice, R., Walter, M., and Leonard, J. (2005). Sparse extended information filters: Insights into sparsification. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*, pages 3281–3288.

Fabresse, F., Caballero, F., Maza, I., and Ollero, A. (2014). Localization and mapping for aerial manipulation based on range-only measurements and visual markers. In *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA 2014*, volume 31.

Frese, U. (2005a). A proof for the approximate sparsity of slam information matrices. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 329–335. IEEE.

Frese, U. (2005b). A proof for the approximate sparsity of slam information matrices. In *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA*, pages 329–335.

Frese, U. (2005c). Treemap: An o (log n) algorithm for simultaneous localization and mapping. In *Spatial Cognition IV. Reasoning, Action, Interaction*, pages 455–477. Springer.

Frey, B. J. and MacKay, D. J. (1998). A revolution: Belief propagation in graphs with cycles. *Advances in neural information processing systems*, pages 479–485.

Frintrop, S. and Jensfelt, P. (2008). Attentional landmarks and active gaze control for visual slam. *Robotics, IEEE Transactions on*, 24(5):1054–1065.

Gao, T., Greenspan, D., Welsh, M., Juang, R. R., and Alm, A. (2006). Vital signs monitoring and patient tracking over a wireless network. In *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 102–105. IEEE.

García-Hernández, C., Ibarguengoytia-Gonzalez, P., and Perez-Diaz, J. (2007). Wireless sensor networks and applications: a survey. *IJCSNS International Journal of Computer Science and Network Security*, 7(3):264–273.

Guivant, J., Nebot, E., and Baiker, S. (2000). Autonomous navigation and map building using laser range sensors in outdoor applications. *Journal of robotic systems*, 17(10):565–583.

Guivant, J. E. and Nebot, E. M. (2001). Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *Robotics and Automation, IEEE Transactions on*, 17(3):242–257.

Guvenc, I. and Chong, C.-C. (2009). A survey on toa based wireless localization and nlos mitigation techniques. *Communications Surveys & Tutorials, IEEE*, 11(3):107–124.

Hai, D., Li, Y., Zhang, H., and Li, X. (2010). Simultaneous localization and mapping of robot in wireless sensor network. In *Proc. IEEE Intl. Conf. on Intelligent Computing and Intelligent Systems*, volume 3, pages 173 –178.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Heredia, G., Jiménez-Cano, A., Sánchez, M., Llorente, D., Vega, V., Braga, J., Acosta, J., and Ollero, A. (2014). Control of a multirotor outdoor aerial manipulator. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS*.

Honkavirta, V., Perala, T., Ali-Loytty, S., and Piché, R. (2009). A comparative survey of wlan location fingerprinting methods. In *Positioning, Navigation and Communication, 2009. WPNC 2009. 6th Workshop on*, pages 243–251. IEEE.

Huang, G. P., Mourikis, A. I., and Roumeliotis, S. I. (2010). Observability-based rules for designing consistent ekf slam estimators. *The International Journal of Robotics Research*, 29(5):502–528.

Huang, S. and Dissanayake, G. (2007). Convergence and consistency analysis for extended kalman filter based slam. *Robotics, IEEE Transactions on*, 23(5):1036–1049.

Huang, S., Wang, Z., and Dissanayake, G. (2008). Sparse local submap joining filter for building large-scale maps. *Robotics, IEEE Transactions on*, 24(5):1121–1130.

Ihler, A. T., Fisher, J. W., Moses, R. L., and Willsky, A. S. (2005). Nonparametric belief propagation for self-localization of sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):809–819.

Ila, V., Porta, J. M., and Andrade-Cetto, J. (2010). Information-based compact pose slam. *Robotics, IEEE Transactions on*, 26(1):78–93.

Jiménez-González, A., Martínez-de Dios, J. R., and Ollero, A. (2011). An integrated testbed for cooperative perception with heterogeneous mobile and static sensors. *Sensors*, 11(12):11516–11543.

Kantor, G., Singh, S., Peterson, R., Rus, D., Das, A., Kumar, V., Pereira, G., and Spletzer, J. (2006). *Distributed Search and Rescue with Robot and Sensor Teams*, pages 529–538. Springer Berlin Heidelberg, Berlin, Heidelberg.

Kuai, X., Yang, K., Fu, S., Zheng, R., and Yang, G. (2010). Simultaneous localization and mapping (slam) for indoor autonomous mobile robot navigation in wireless sensor networks. In *Proc. Intl. Conf. on Networking, Sensing and Control (ICNSC), 2010*, pages 128 –132.

Kumar, P., Reddy, L., and Varma, S. (2009). Distance measurement and error estimation scheme for rssi based localization in wireless sensor networks. In *Wireless Communication and Sensor Networks (WCSN), 2009 Fifth IEEE Conference on*, pages 1–4. IEEE.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. (2011). g 2 o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE.

Kurth, D., Kantor, G., and Singh, S. (2003). Experimental results in range-only localization with radio. In *Intelligent Robots and Systems, 2003.(IROS 2003).*

Proceedings. 2003 IEEE/RSJ International Conference on, volume 1, pages 974–979. IEEE.

Langendoen, K. and Reijers, N. (2003). Distributed localization in wireless sensor networks: a quantitative comparison. *Computer Networks*, 43(4):499–518.

Lee, H., Cerpa, A., and Levis, P. (2007). Improving wireless simulation through noise modeling. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 21–30. IEEE.

Lee, K. W., Wijesoma, W. S., and Javier, I. G. (2006). On the observability and observability analysis of slam. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3569–3574. IEEE.

Leishman, R. C., Macdonald, J. C., Beard, R. W., and McLain, T. W. (2014). Quadrotors and accelerometers: State estimation with an improved dynamic model. *Control Systems, IEEE*, 34(1):28–41.

Leung, C., Huang, S., and Dissanayake, G. (2006). Active slam using model predictive control and attractor based exploration. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 5026–5031.

Liu, C., Wu, K., and He, T. (2004). Sensor localization with ring overlapping based on comparison of received signal strength indicator. In *Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on*, pages 516–518. IEEE.

Liu, Y. and Thrun, S. (2003). Results for outdoor-slam using sparse extended information filters. In *Proc. IEEE Intl. Conf. on Robotics and Automation, ICRA*, volume 1, pages 1227–1233.

Lorincz, K., Malan, D. J., Fulford-Jones, T. R., Nawoj, A., Clavel, A., Shnayder, V., Mainland, G., Welsh, M., and Moulton, S. (2004). Sensor networks for emergency response: challenges and opportunities. *Pervasive Computing, IEEE*, 3(4):16–23.

Lourenço, P., Batista, P., Oliveira, P., Silvestre, C., and Chen, C. P. (2013). Sensor-based globally asymptotically stable range-only simultaneous localization and mapping. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 5692–5697. IEEE.

Lu, F. and Milios, E. (1997). Globally consistent range scan alignment for environment mapping. *Autonomous robots*, 4(4):333–349.

Marron, P. J., Karnouskos, S., Minder, D., and Ollero, A. (2011). *The emerging domain of Cooperating Objects*. Springer Science & Business Media.

Martinez-de Dios, J., Lferd, K., de San Bernabe, A., Nunez, G., Torres-Gonzalez, A., and Ollero, A. (2013). Cooperation between uas and wireless sensor networks for efficient data collection in large environments. *Journal of Intelligent & Robotic Systems*, 70(1-4):491–508.

Martinez-de Dios, J. R., Jimenez-Gonzalez, A., San Bernabe, A., and Ollero, A. (2014). *A remote integrated testbed for cooperating objects*. Springer.

Maza, I., Caballero, F., Capitan, J., Martinez-de Dios, J., and Ollero, A. (2011). A distributed architecture for a robotic platform with aerial sensor transportation and self-deployment capabilities. *Journal of Field Robotics*, 28(3):303–328.

Meertens, L. and Fitzpatrick, S. (2004). The distributed construction of a global coordinate system in a network of static computational nodes from inter-node distances. *Kestrel Institute TR KES. U*, 4.

Menegatti, E., Danieletto, M., Mina, M., Pretto, A., Bardella, A., Zanconato, S., Zanuttigh, P., and Zanella, A. (2010). Autonomous discovery, localization and recognition of smart objects through wsn and image features. In *Proc. IEEE GLOBECOM 2010*, pages 1653 –1657.

Menegatti, E., Zanella, A., Zilli, S., Zorzi, F., and Pagello, E. (2009). Range-only slam with a mobile robot and a wireless sensor networks. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, pages 8–14.

Merino, L., Caballero, F., and Ollero, A. (2010). Active sensing for range-only mapping using multiple hypothesis. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 37–42, Taipei (Taiwan).

Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al. (2002). Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Aaai/iaai*, pages 593–598.

Murillo, A., Guerrero, J. J., and Sagues, C. (2006). Robot and landmark localization using scene planes and the 1d trifocal tensor. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2070–2075. IEEE.

Niculescu, D. (2004). Positioning in ad hoc sensor networks. *IEEE Network*, 18(4):24–29.

Nogueira, M., Sousa, J., and Pereira, F. (2010). Cooperative autonomous underwater vehicle localization. In *IEEE OCEANS - Sydney*, pages 1–9.

Olson, E., Leonard, J., and Teller, S. (2004). Robust range-only beacon localization. In *Proc. IEEE/OES Autonomous Underwater Vehicles*, pages 66 – 75.

Paskin, M. A. (2003). Thin junction tree filters for simultaneous localization and mapping. In Gottlob, G. and Walsh, T., editors, *Proc. of the 18th Intl. Joint Conf. on Artificial Intelligence, IJCAI*, pages 1157–1164, San Francisco, CA. Morgan Kaufmann Publishers.

Patwari, N., O'Dea, R. J., and Wang, Y. (2001). Relative location in wireless networks. In *Vehicular Technology Conference, 2001. VTC 2001 Spring. IEEE VTS 53rd*, volume 2, pages 1149–1153. IEEE.

Pearl, J. (2014). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. (2000). The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM.

Roy, S., Foerster, J. R., Somayazulu, V. S., and Leeper, D. G. (2004). Ultrawide-band radio design: The promise of high-speed, short-range wireless connectivity. *Proceedings of the IEEE*, 92(2):295–311.

Savvides, A., Han, C.-C., and Strivastava, M. B. (2001). Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179. ACM.

Seidel, S. Y. and Rappaport, T. S. (1992). 914 mhz path loss prediction models for indoor wireless communications in multifloored buildings. *Antennas and Propagation, IEEE Transactions on*, 40(2):207–217.

Shang, Y., Ruml, W., Zhang, Y., and Fromherz, M. P. J. (2003). Localization from mere connectivity. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*, MobiHoc '03, pages 201–212, New York, NY, USA. ACM.

Simon, G., Maróti, M., Lédeczi, Á., Balogh, G., Kusy, B., Nádas, A., Pap, G., Sallai, J., and Frampton, K. (2004). Sensor network-based countersniper system. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 1–12. ACM.

Sun, D., Kleiner, A., and Wendt, T. (2009). Multi-robot range-only slam by active sensor nodes for urban search and rescue. In Iocchi, L., Matsubara, H., Weitzenfeld, A., and Zhou, C., editors, *RoboCup 2008: Robot Soccer World Cup XII*, volume 5399 of *Lecture Notes in Computer Science*, pages 318–330. Springer Berlin Heidelberg.

Thrun, S., Burgard, W., Fox, D., et al. (2005). *Probabilistic robotics*, volume 1. MIT press Cambridge.

Thrun, S. and Liu, Y. (2005). Multi-robot slam with sparse extended information filers. In *Robotics Research*, pages 254–266. Springer.

Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z., and Durrant-Whyte, H. (2004). Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716.

Torres-González, A., Martinez-de Dios, J., and Ollero, A. (2014). Integrating internode measurements in sum of gaussians range only slam. In *ROBOT2013: First Iberian Robotics Conference*, pages 473–487. Springer International Publishing.

Vidal-Calleja, T., Davison, A., Andrade-Cetto, J., and Murray, D. (2006). Active control for single camera slam. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1930–1936.

Walter, M. R., Eustice, R. M., and Leonard, J. J. (2007). Exactly sparse extended information filters for feature-based slam. *The International Journal of Robotics Research*, 26(4):335–359.

Wang, H., Hu, G., Huang, S., and Dissanayake, G. (2012). On the structure of nonlinearities in pose graph slam. In *Robotics: Science and Systems*.

Wang, H., Huang, S., Frese, U., and Dissanayake, G. (2013). The nonlinearity structure of point feature slam problems with spherical covariance matrices. *Automatica*, 49(10):3112–3119.

Wang, X., Bischoff, O., Laur, R., and Paul, S. (2009). Localization in wireless ad-hoc sensor networks using multilateration with rssi for logistic applications. *Procedia Chemistry*, 1(1):461–464.

Wang, Z., Huang, S., and Dissanayake, G. (2007). D-slam: A decoupled solution to simultaneous localization and mapping. *The International Journal of Robotics Research*, 26(2):187–204.

Wei, M., Aragues, R., Sagues, C., and Calafiore, G. C. (2015). Noisy range network localization based on distributed multidimensional scaling. *IEEE Sensors Journal*, 15(3):1872–1883.

Weiser, M. (1991). The computer for the 21st century. *Scientific american*, 265(3):94–104.

Weiser, M. (1993). Ubiquitous computing. *Computer*, 26(10):71–72.

Wenhardt, S., Deutsch, B., Angelopoulou, E., and Niemann, H. (2007). Active visual object reconstruction using d-, e-, and t-optimal next best views. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7.

Werner-Allen, G., Lorincz, K., Ruiz, M., Marcillo, O., Johnson, J., Lees, J., and Welsh, M. (2006). Deploying a wireless sensor network on an active volcano. *Internet Computing, IEEE*, 10(2):18–25.

Wessels, A., Wang, X., Laur, R., and Lang, W. (2010). Dynamic indoor localization using multilateration with rssi in wireless sensor networks for transport logistics. *Procedia Engineering*, 5:220–223.

Williams, S. B. (2001). *Efficient solutions to autonomous mapping and navigation problems*. PhD thesis, Citeseer.

Yang, L. and Giannakis, G. B. (2004). Ultra-wideband communications: an idea whose time has come. *Signal Processing Magazine, IEEE*, 21(6):26–54.

Yick, J., Mukherjee, B., and Ghosal, D. (2005). Analysis of a prediction-based mobility adaptive tracking algorithm. In *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on*, pages 753–760. IEEE.

Youssef, M., Mah, M., and Agrawala, A. (2007). Challenges: device-free passive localization for wireless environments. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 222–229. ACM.

Zanella, A. (2016). Best practice in rss measurements and ranging. *IEEE Communications Surveys Tutorials*, PP(99):1–1.

Zanella, A., Bui, N., Castellani, A., Vangelista, L., and Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things Journal*, 1(1):22–32.

Zhao, L., Huang, S., and Dissanayake, G. (2013). Linear slam: A linear solution to the feature-based and pose graph slam based on submap joining. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 24–30. IEEE.