COORDINATED CONTROL OF MIXED ROBOT AND SENSOR NETWORKS IN DISTRIBUTED AREA EXPLORATION

GIANLUCA BIANCHIN

Master's degree in Automation Engineering
Department of Information Engineering
University of Padova

– October 2014 –

Gianluca Bianchin: *Coordinated control of mixed robot and sensor networks in distributed area exploration*, Master's degree in Automation Engineering, © October 2014

SUPERVISOR:
Angelo Cenedese

COMMITTEE IN CHARGE:

Professor: Maria Elena Valcher	Professor: Mauro Migliardi
Professor: Andrea Cester	Professor: Loris Nanni
Professor: Simone Buso	Professor: Luca Palmieri

LOCATION:
Padova

TIME FRAME:
April - October 2014

This thesis is dedicated to my loving parents Achille and Nadia

for their love, endless support
and encouragement.

I am proud to be your son.
Thank you.

ABSTRACT

Recent advancements in wireless communication and electronics has enabled the development of low-cost, low-power, multifunctional sensor nodes that are small in size and communicate untethered in short distances. These tiny sensors, which consists of sensing, data processing and communicating components, leverage the idea of sensor networks. A sensor network is composed of a large number of sensors nodes that are densely deployed either inside the phenomenon or very close to it. Recently, large scale sensor networks are drawing an ever increasing interest in various technological fields.

In the near future, sensor networks will grow in size, therefore forthcoming steps in their evolution consist in combining large number of devices with different functions in order to create a heterogeneous network moving around the complexity of a standardized interface. Local interactions between sensor nodes allow them to reach a common goal and to deduce global conclusions from their data.

While potential benefits of sensor networks are clear, a number of open problem must be solved in order for wireless sensor network to become viable in practice. This problems include issues related to deployment, security, calibration, failure detection and power management.

In the last decade, significant advantages have been made in the field of service robotics, and robots have become increasingly more feasible in practical system design. Therefore, we trust that a number of open problems with wireless sensor networks can be solved or diminished by including mobility capabilities in agents composing the network.

The growing possibilities enabled by robotic networks in monitoring natural phenomena and enhance human capabilities in unknown environments, convey researchers recent interests to combine flexibility characteristics typical for distributed sensor networks, together with advantages carried by the mobility features of robotics agents. Teams of robots can often perform large-scale tasks more efficiently than single robots or static sensors; therefore the combination of mobility with wireless networks greatly enhanced the application space of both robots and sensor networks.

Some of the application areas for mixed robot and sensor networks are health, military and home. In military, for instance, the rapid deployment, self-organization and fault detection characteristics of sensor networks make them a very promising sensing technique for military command, control, communications, computing, intelligence, surveillance, reconnaissance and targeting systems. In health, sensor nodes can also be deployed to monitor ill patients and assist disabled patients.

The focus of this work is to study and propose theoretical approaches that pair together with the algorithm development regarding four open problems of both sensor and robot networks. These are:

- Each agent to localize itself;
- The group to set-up a localization infrastructure;
- The group to establish robust spatial patterns for sensing the environment;
- The group to perform a global measure for the mission-specific quantity of interest.

The approach we propose in this work to overcome open problems arising with sensor and robot network consists in exploiting the interaction between the two systems in order to efficiently perform common goals.

ACKNOWLEDGEMENTS

The successful completion of this thesis is the result of the help, cooperation, faith and support of my supervisor Professor *Angelo Cenedese*, that I would like to thank for all his technical teachings and who demonstrated considerable patience and willingness. I am grateful for the insightful discussions we had, for his guidance during last period of my graduate studies at Padova, and for his helpfully and support during the development of this work.
My deepest gratitude goes to my family for their unflagging love and unconditioned support throughout my studies. Special thanks goes to my sister Stefania, for every period we spent together in our childhood and in recent years.

I will always appreciate all you have done.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

WSNs Wireless Sensor Networks

EKF Extended Kalman Filter

ToA Time of Arrival

TDoA Time Difference of Arrival

AoA Angle of Arrival

RSS Received Signal Strength

RSSI Received Signal Strength Indicator

CCW Counter Clock Wise

WMRs Wheeled Mobile Robots

VTOL Vertical Take-Off and Landing

GPS Global Positioning System

CSMA Carrier Sense Multiple Access

SNR Signal to Noise Ratio

MRS Multi-Robot Systems

Part I

OVERVIEW ON THE PROBLEM

This introduction part contains a brief discussion in the topic of distributed systems, sensor and robot networks: description, potentiality, research open-problems, and applications of this kind of systems are resumed. An outline of the work then introduces considered problems and solutions proposed.

1 INTRODUCTION

Wireless Sensor Networks (WSNs) are large groups of spatially distributed electronic devices capable of sensing, computation and wireless communication. This type of networks are becoming very popular as they can offer access to huge quantity, and accurate quality of information that can revolutionize our ability to control the environment.
The variety of possible applications of WSNs to the real world is practically unlimited. It is important to underline that the application strongly affects the choice of the wireless technology to be used. Typical applications include:

- Habitat monitoring and environmental context: air pollution monitoring, forest fire detection, landslide detection, water quality monitoring [42];
- Industrial context: car tracking, industrial automation, machine surveillance and preventive maintenance;
- Surveillance and building environmental control in urban context [25]: traffic control, structural monitoring, video surveillance;
- Domestic context: home automation, domotics;
- Health care context: psychological data monitoring, home health care or assisted living, facilitation for disabled people, hospital management, allergy identification;
- Military context: battlefield surveillance, forces and equipment monitoring, enemy recognition, damages estimation, attack detection.

Recent technological improvements have enabled the deployment of small, inexpensive, low-power distributed devices which are capable of local processing and wireless communication. Each sensor node is capable of only a limited amount of processing, but when coordinated with the information from a large number of other nodes, they have the ability to manage even complex tasks.

Traditional large scale systems are usually characterized by a centralized architecture, while the recent trend is based on a distributed approach. Even if a centralized structure entails the advantage to be easy to be designed, it requires the employment of very reliable and expensive sensors, and it involves remarkable communication limitation. The choice of the distributed model should implicitly accounts for scalabilty and robustness to failures of both nodes and network. Moreover in large scale applications low-cost sensors are preferred in order to decrease costs.

The modern design of robotic and automation systems consider networked vehicles, sensors, actuators and communication devices. These developments enable researches and engineers to design new robots capable of performing tasks in a cooperative way. This new technology has been denominated Multi-Robot Systems (MRS) and includes:

- Physical embodiment: any MRS has to have a group of physical robot which incorporates hardware and software capabilities;
- Autonomous capabilities: a physical robot must have autonomous capabilities to be considered as a basic element of a MRS;
- Network-based cooperation: the robots, environment, sensors and humans must communicate and cooperate through a network;
- Environment sensors and actuators: besides sensing capabilities enhanced in robots, the framework must include other sensors, such as vision cameras, laser range finders, electronic sensors and other actuators, that often are dispatched by robotic agents ;
- Human-robot interaction: in order to consider a system as MRS, the system must have a human-robot related activity.

The initial motivation to this work was the implementation and evaluation of a self-configuring Multi-Robot System, able to provide self-adapting techniques in order to perform complex tasks in unknown mission spaces.

Mobile robots are increasingly used in flexible manufacturing industry and service environments. The main advantage of these vehicles is that they can operate autonomously in the workspace. To achieve this automation, mobile robots must include a localization - or positioning - system in order to estimate their own pose (positioning and orientation) as accurately as possible [28].

In particular, location-based applications are among the first and most popular applications for robotic nodes, since they could be employed to track people in wide outdoor areas or in extending to indoor environments the GPS approach for locating people and tracking mobile objects in large buildings (e.g. warehouses).

Positioning is a fundamental issue in mobile robot applications: indeed, a mobile robot that moves across its environment has to position itself before it can execute properly every kind of actions. Therefore mobile robots has to be equipped with some hardware and software, capable to provide a sensory feedback related to the environment .

Services provided by the system, included localization, can be improved through the exploitation of described encouraging capabilities of sensor networks. In this sense, capabilities of robotic and sensor networks are complementary and can be combined in order to produce a complete and robust platform.

The flexibility of the system can furthermore be improved through deployment capabilities for mobile agents: this feature ensure flexibility and adaptive characteristics to the system.

The performance of this kind of systems in terms of quality of the service provided is sensitive to the location of its agents in the mission space. This leads to the basic problem of deploying sensors and robots in order to meet the overall system objectives, which is referred to as the *coverage control* or *active sensing* problem [36] [9] [37].

In particular, sensors must be deployed and robot must be moved, so as to maximize the information extracted from the mission space - or the quality of service provided - while maintaining acceptable levels of communication and energy consumption. Coverage control problems have been studied in various context. In robotics, the principal goal is to move from one position to another so as to maximize the information gathered from the environment.

1.1 CONTRIBUTION

The goal of this thesis consists in examining the interaction between Wireless Sensor Networks and Multi-Robot systems: first of all introducing weaknesses of both systems taken separately and then proposing control strategies that enlarge systems capabilities through interaction.

In particular the problems of Localization, Beacons placement, and Coverage Control are taken into account and novel state-of-art solutions are proposed. These are mainly based on the mutual interplay between the robot and sensor networks: features of both infrastructures are joined in order to perform common tasks.

The work includes a comprehensive methods validation chapter, in which numerical simulations presents some typical issues arising in practice. Advantages and peculiar behaviors kept by agents are discussed.

The main contributions of this work are the derivation of a closed-form expression for estimated location's variance and the use of an intermittent Extended Kalman Filter (EKF) capable of combining odometry together with geometric localization data in order to improve localization quality. The closed-form expression for uncertainty has enabled a dynamical deployment of beacon nodes to dynamically enlarge infrastructure covered area, and to improve localization quality in the whole configurations space. An adaptive method for coverage control in unknown area exploration is described, capable of dynamically update informativeness and network's interests in non-visited areas. Results from simulations motivate coverage control strategy based on optimization, that enables robust environment coverage through Voronoi partitioning.

1.2 OUTLINE OF THE WORK

The remainder of the thesis is organized as follows.

- Chapter 2 presents an overview of robotic networks and WSNs. In spite of the diverse applications, sensoristic and robotic networks pose a number of unique technical challenges due to several factors.

The main troubles arising when both the systems are combined are introduced;

- Chapter 3 deals with the problem of locating mobile agents in unknown environments. In the first part, the most popular methods for robot localization are described and compared in terms of robustness and data required. The Geometric Triangulation method is discussed in detail and taken into account for its promising capabilities, and a novel algorithm based on this technique recently presented in literature is described. Starting from recent papers, an estimate for the variance of robot position is derived and exploited in order to combine the internal kinematics model of agents, together with triangulation data through Kalman filtering;

- Chapter 4 examines in detail the beacons placement problem for localization. Closed form location variance expressions, derived in previous chapter, are exploited to deduce optimal pattern and configuration for the sensor network providing localization. A novel method based on beacons cooperation and estimated location variance is developed in order to both extend the existing infrastructure, and improve localization quality;

- in Chapter 5 a distributed gradient-based algorithm maximizing the joint detection probabilities of random events is designed. A model for events taking place in the mission space is proposed using a density function representing the frequency of random events taking place. In the second part of the chapter, several improvements are proposed in order to provide global knowledge of the mission space to the network;

- Chapter 6 presents simulation results, which illustrates the effectiveness of the proposed schemes and compare network performances in several practical configurations.

- Chapter 7 and 8 concludes the thesis and describes directions for future works.

- Appendices report some theoretical results that can be developed in future works.

2 OVERVIEW

In the recent decades researchers focused their attentions in engineering systems composed by a large number of devices that can communicate and cooperate to achieve a common goal. Although complex large-scale monitoring and control systems are not new, as for example air traffic control or smart grids applications, a new architectural model is emerging, mainly thanks to the adoption of smart agents i. e. devices that are capable of cooperating and of taking autonomous decisions without any supervisory system. In fact, traditional large-scale systems have a centralized or at best a hierarchical architecture, which has the advantage to be relatively easy to be designed and has safety guarantees. However, these systems require reliable sensors and actuators and in generally are very expensive. Another relevant limitation related with centralized systems is that they do not scale well, due to communication and computation limitations.

The recent trend, in order to avoid these problems, is to substitute costly sensors, actuators and communication systems with a larger number of devices that can autonomously compensate potential failures and computation limitations through communication and cooperation.

2.1 LOCALIZATION

A common problem in mobile robotics deals with understanding where mobile agents are located in the 3D space. Localization is the process of finding both position and orientation of a vehicle in a given referential system. Navigation of mobile vehicles indoors and outdoors usually requires accurate and reliable methods of localization.
Localization is a complicated issue in the real world, as sensors are not perfect and measured quantities can result distorted by noise. Moreover environmental models are never fully complete, and robot's wheels can slip, causing errors in odometry[1] readings.

These limitations can be overcome by referencing the robot's location to landmarks whose locations are known. Unfortunately, there are two relevant issues that complicate landmarks-based navigation. First of all, the robot's orientation with respect to the world coordinate system is highly important (an incorrect measure of the robot's orientation will cause additional errors in locations as the robot moves), moreover the determination of robot's location and orientation is not a trivial issue.
For the reasons explained, a simple two landmark localization scheme is an insufficient solution. In fact, when navigating on a plane, three distinguishable beacons - at least - are required for the robot to localize itself (Figure 1). This is motivated by the fact that localization on a plane arises

1 Odometry is the use of data from motion sensors to estimate changes in position over time.

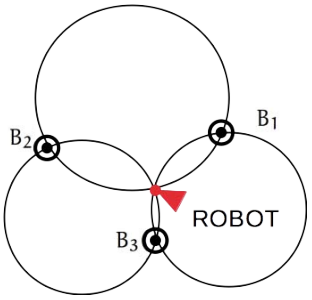


Figure 1: Localization results as the intersection of three distinguishable circles.

as the intersection of three distinguishable circles (each of them passing through a couple of landmarks and the robot). This is the reason why almost three beacons are required in order to localize mobile nodes. Usually, the use of more than three beacons results in redundancy.
On the other hand, using more than two landmarks for determining robots positions (i. e. location and orientation) is not a trivial problem: triangulation with three beacons is called *Three object triangulation*.

First of all, the solution to the problem can be obtained through a geometrical approach. Unfortunately, the geometry of the problem permits in general multiple solutions for valid robot locations and the geometry for computing the solution is in general not straight forward.

Secondly, an approach based on a mathematical derivation can be exploited. However, the computation of the solution is not a trivial problem. In particular the angle and distance between the landmarks is not the only information required to solve the problem; and the proper ordering of the landmarks is important. The geometric relationship between the landmarks and the robot must be considered and the uniqueness of the solution is not guaranteed in general.

2.2 BEACON DEPLOYMENT

As sensors are becoming inexpensive, deploying many sensors in a workspace to provide localization and other services is becoming feasible. Localization technologies based on anchor nodes provide a valuable alternative to on-board localization systems, because of their higher precision and avoidance of the unbounded growth of time integration errors with the distance traveled by the robot.

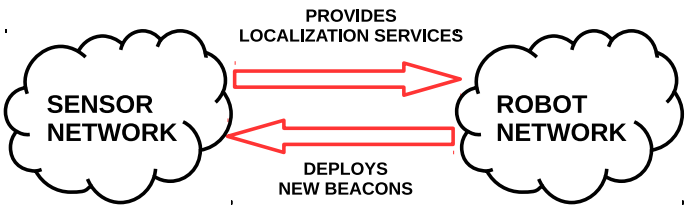


Figure 2: Mutual interplay between robot and sensor networks.

In particular, in scenarios where many specialized robots operate in the same workspace, it may be cheaper to install several beacon nodes and use them to localize all of the robots, rather then equipping mobile robots with extremely accurate odometric sensors. In fact, localization based on external beacons is in general robuster than on-board localization, moreover it could be the only alternative in scenarios where odometric sensors are not enough reliable. Localization systems based on anchor nodes have two main advantages over localization systems with no beacons. First, having beacons spatially distributed throughout the geographical region lets devices to compute their location in a scalable, decentralized manner. Second, even when the application permits offline centralized position estimation algorithms, both the coverage and estimation accuracy can be significantly improved by disposing of external fixed nodes as beacon.

The nature of the environment, such as indoors or outdoors and its features, temperature, pressure, weather, objects and various sources of interference influences not only the characteristics of sensors used, but also the magnitude and type of measurement errors. Traditionally, this types of issues have been addressed through extensive environment-specific calibration and configuration of the centrally controlled localization system [4]. This approach is, however, not suited for large scale sensor networks since the specific calibration does not fit the distributed approach. Recent studies have focused on self-configuring localization systems, that is, systems able to autonomously measure and adapt their properties to environmental conditions in order to achieve ad-hoc and robust deployment. The flexibility guaranteed by networks capable of self-deploying agents to perform tasks, is advantageous compared to pre-allocated systems: it provides robustness to agent failure, longevity to the entire network and allows to handle more complex tasks.

There are two major configuration and deployment concerns when beacons are used.

- Beacon configuration: each beacon needs to be configured with its spatial coordinates during deployment. Automating this process is important for large scale and highly dense beacon deployment. In an outdoor setting, it is possible assume that beacons can infer their position through GPS. However, in order to ensure robustness to the system, only a few beacons will need to have their positions assigned manually, the rest can exploit this structure in beacon placement to infer their coordinates.

- Beacon placement. The issue to understand how many beacons are need and where should they be placed regard the *beacon placement problem*. The beacon density and placement are important in influencing the overall localization quality. Uniformly dense placement is good and has its benefits, however in many cases it is not adequate.

2.3 COVERAGE CONTROL

Deploying multiple agents to perform tasks is advantageous compared to the single agent case: it provides robustness to agent failure and allows to handle more complex tasks. The single, heavily equipped vehicle may require considerable power to operate its sensor payload, it lacks robustness to vehicle failure and it cannot adapt its configuration to environmental changes. A cooperative network of sensors and vehicles equipped with sensor, has the potential to perform efficiently and reliably tasks in a more flexible and scalable way than single better-equipped agents. Therefore, distributed control can be employed by groups of robots to carry out tasks such as environmental monitoring, automatic surveillance of rooms, buildings or towns, search and rescue etc.

The performance of multi-robot and sensor network in distributed area exploration is sensitive to the location of agents in the mission space. In particular, sensors must be deployed so as to maximize the information extracted from the mission space. The goal is therefore to drive the sensors/agents to the position such that a given region is optimally covered by sensors. This causes the network to spread out over the environment while aggregating in areas of high sensory interest. Furthermore, robots do not know beforehand where areas of major interest are located: the network is required to learn this information online from sensors measurements. In this work, we consider a mobile sensing network composed of vehicles and static agents, both equipped with sensors to sample the environment; the problem of deploying agents is referred to as the *Coverage control* problem.

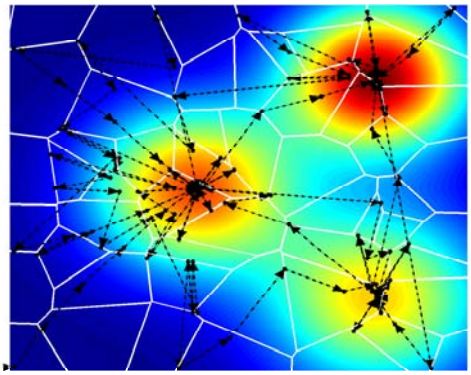


Figure 3: Coverage Control: the network spread out over the environment, while aggregating in areas of high sensory interest.

In particular, the problem of surveying a given arbitrarily-shaped mission space is addressed. The problem solution require the distribution of a fixed sensor network and a Multi-robot network in the domain to be sampled. The problem of optimizing sensor locations in fixed sensor networks has been extensively studied in the past, and are still open. In such

problems, the solution is a Voronoi partition, where the optimal sensor domain is a Voronoi cell in the partition and the optimal sensor location is a centroid of a Voronoi cell in the partition.

The approach proposed in this work is based on a distributed optimization that guide mobile agents towards points of maximum informativeness. The method is then extended exploiting Voronoi tessellation in order to keep track of the overall visited areas, this allows improvements in convergence time and in terms of overall knowledge.

Part II

THEORETICAL RESULTS

When sensors are deployed in unknown environments the main issues arising regards locating the sensor in order to handle spatial goals, understanding best locations for agents in order to gather more informative events, and designing locations for new nodes in order to improve quality of services provided. These are referred as Localization, Beacon Placement, Coverage Control issues for sensor and robot networks. The following chapters contain a comprehensive discussion regarding this topics and solutions for above mentioned problems.

3 LOCALIZATION

Localization of mobile targets in structured environments is helped, in general, by external elements that are called landmarks. Sensors are not perfect and an environmental model is never fully complete. Moreover robot's wheels can slip, causing errors in odometry readings [2] and this let self-localization to be very unreliable. For these reasons, in order to operate in unknown environments, robotic agents must be capable to acquire and to use knowledge, to estimate insides of the setting and to answer in real time for the situations that can occur.

3.1 OBJECTIVE AND MOTIVATION

Localization may be defined as the problem of estimating the spatial relationships among objects. In relative localization, dead-reckoning[1] methods ([23] and [7]) which consists of odometry and inertial navigation, are used to calculate the robot position and orientation from a known initial pose. Odometry is a widely used localization method because of it is low cost, shows high updating rate, and is reasonably accurate when used in short path.

However its unbounded growth of time integration errors with the distance traveled by the robot is unavoidable and represent a significant inconvenience [24]. Several approaches have been proposed to cope with odometric error propagation [46] and [44].

Conversely, absolute localization methods estimate the robot position and orientation by detecting particular features of a known environment. This could be particular landmarks accurately located, or pre-existing points already comprehended in the environment.
Among the methods proposed in literature, geometric triangulation is one of the most-widely used in the absolute localization thanks to the fact that it provides a closed-form solution, to the accuracy of solutions provided and to its ability of determining both position and orientation of the targets.
The accuracy of triangulation algorithms depends upon both landmarks arrangement and the position of the mobile robot among them. Localization systems using some beacons have two advantages over localization with no beacons [27].
Firstly, having beacons spatially distributed throughout the geographical region enables devices to compute their location in a scalable, decentral-

1 Dead-Reckoning is the process of calculating current position by using a previously determined position, and advancing that position based upon known or estimated speeds. Dead-Reckoning is a particular type of Odometry. A more detailed discussion about dead-reckoning is conduced in Section 6.1.1

ized manner. Secondly, even when the application permits offline, centralized position estimation algorithms, both the convergence and estimation accuracy can be significantly improved by having some nodes as beacons [11]. Beacons constitute the underlying infrastructure of the localization system.

3.1.1 *Problem definition*

This chapter presents a comprehensive solution to the problem of locating a mobile target, usually a robot, in a planar mission space, given the location of three or more distinguishable landmarks in the environment, and measured angles between them as viewed from the robot.
Locating, in this context, means determining robot's location and orientation uniquely. This situation arises when a mobile robot moves in a mission space where other localization technologies (such as Global Positioning System (GPS)) are not available.

Aspects of triangulation to consider in order to achieve optimal results for the robot pose[2] in practical situation are:

1. the sensitivity analysis of the algorithm;
2. the optimal placement of landmarks;
3. the selection of some landmarks among the available ones to compute the robot pose;
4. the knowledge of the true landmark location in the world and the true location of the angular sensor on the robot.

Having a good sensor that provides precise angle measurements as well as a good triangulation algorithm is not the only concern to get accurate positioning results. Indeed, the angle sensor could be subject to non linearities in measuring angle range. Moreover the beacon locations are often subject to inaccuracies, which directly affect the positioning algorithm.

The main objective of this chapter is to present a three object triangulation algorithm, based on geometric triangulation, capable of working in the entire mission space (except when beacons and robot are collinear) and for any beacon ordering. Moreover the number of trigonometric computation have to be minimized.

The second part of the chapter focuses on providing an accurate expression for variance of the position estimated by the algorithm. This is an indispensable requisite that enables to give a statistical description to the estimated position, in order to perform statistical filtering.

The final part of this chapter deal with combining the estimated position through geometric triangulation algorithm, with dead-reckoning data. As it will be shown, this approach guarantees a high-rate, energy-saving and robust approach to the problem.

In particular Kalman filtering is used as it is a convenient way to fuse triangulation estimates together with odometry data.

2 pose indicates both the location and orientation of a robot

3.1.2 *Previous work*

One of the first comprehensive work on localization has been carried out by Cohen and Koss. The work classifies the triangulation algorithms into four groups: Geometric Triangulation, Iterative methods, Geometric circle intersection, Multiple beacon triangulation.

Several authors have noticed that the second type methods(*Geometric circle intersection*) are the most popular in robotics [18] [35]. These methods compute the intersection of the three circles passing through a couple of beacons and the robot.

Esteves et al. in [15] extend the algorithm presented by Cohen and Koss to work for any beacon ordering and to work outside the triangle formed by the three beacons. In [19] a method, working for every beacon ordering, is presented. The method divides the whole plane into seven region and handles two specific configurations of the robot relatively to the beacons.

The most recent works [31] [39] propose novel methods to achieve a solution to the geometric triangulation relying on the idea of using the radical axis of circles. The methods still works in the entire plane naively. Furthermore it only uses basic arithmetic computations and two $\cot(\cdot)$ computation, leading to a drastic reduction in computational complexity.

This work aims at improving performances of triangulation algorithms by combining the position estimates with odometry data. Similar approaches, based on EKF, have already been proposed in previous works as in [17] and [16] but in these cases the filtering was used on the measures of angles between the robot and the beacons.

3.1.3 *Contribution*

In the first part of the chapter, arising from [39] we report a comprehensive description of ToTal algorithm proposed by Pierlot and Van Droogenbroeck. Particular attention is paid both to computational reduction brought with respect to previous works and to improvements in the robustness of the solution, introduced thanks to the exploitation of radical axes of circles employed.

In the second part of the chapter, a novel method capable of combining triangulation solution together with odometry is proposed.

We underline that more appreciated approaches in published works integrate a prediction phase, based on the odometric data and the robot kinematics, and a correction -or estimation- phase that takes into account external measurements. The approach used in this work is based on Kalman filtering [48]. In fact, Kalman filtering results as a suitable way to combine data arising from internal kinematic model together with triangulation data.

The chapter is completed with an overview on kinematic models for Wheeled Mobile Robots and Quadrotors.

3.2 NAVIGATION SYSTEMS: METHODS CLASSIFICATION

Landmark-based navigation of autonomous mobile robots or vehicles has been widely adopted in industry. In general, the methods for locating mobile robots in the real world are divided into two categories:

1. Relative positioning;
2. Absolute positioning.

In relative positioning, odometry (in particular dead-reckoning) and inertial navigation are commonly used to calculate the robot positions from a start reference point at a high updating rate. Odometry is one of the most popular techniques based on internal sensors for position estimation because of its ease of use in real time. However, a compromising disadvantage affects this method: it suffers of an unbounded accumulation of errors due to inaccuracies of sensors. Therefore, frequent corrections to the estimated position become necessary.

In contrast, absolute positioning relies on detecting and recognizing different features in the environment, in order for a mobile robot to reach a destination and implement a specified task. These environment features are normally divided into two types:

1. Natural landmarks;
2. Artificial landmarks.

Among these, natural landmark navigation is flexible as no explicit artificial landmarks are needed, however it may not work well when landmarks are sparse and often the environment must be a priori known. Although the artificial landmark and active beacon approaches are less flexible, the ability in finding landmarks is enhanced and the process of map building is simplified.

To make the use of mobile robots in daily deployment feasible, it is necessary to reach a trade-off between costs and benefits. Often, this discourages the use of expensive sensors such as vision systems and GPS in favor of cheaper sensing devices, for example laser or encoders, and calls for efficient algorithms that can guarantee real-time performance in the presence of insufficient or conflicting data.

Two scenarios could be hypothesized regarding nodes' behavior:

1. Anchor-based, in which only landmark (or anchor) nodes know their absolute position, while robotic nodes exploit landmarks to determine their absolute position;
2. Anchor-free, in which any node does not know its absolute location, in this case a relative coordinate reference is used.

The first type of methods enables to know the absolute location of nodes in an unique way, but this requires anchor nodes to know their absolute location (using for example GPS systems) and these should be quite densely

allocated in the configuration space, as they can not move. The second type of methods enable a more flexible arrangement of anchor, so nodes could be placed in a random way. Although anchor-free systems enable more flexibility in the deployment phase, the absence of anchor nodes produce an unbounded error propagation in the network.
Both in anchor-based and in anchor-free systems, the position computation is performed basing on the knowledge of relative distances (or angles) between nodes. Three types of techniques could be used to compute relative distance of a node with neighbors:

1. Range based;
2. Angle based;
3. Range free.

Range-based techniques make use of Euclidean distances, in fact nodes are equipped with sensors to estimate relative distances with neighbor nodes. Algorithm such as Min-Max, Trilateration and Multilateration belong to this family of methods [8] .

Angle-based techniques are based on angular distances between nodes, in fact robots are equipped with sensors capable of estimating relative angles with neighbor nodes. This type of techniques exploit geometric and trigonometric properties as motivated in [8]. Geometrical triangulation belong to this family of methods.

Range-free techniques are based on virtual coordinate references, and for this reason, they are independent from Euclidean distances.
A range-free method requires no distances or angles measurement among nodes, so they do not require additional hardware; and instead use properties of the wireless sensor network to obtain location information.

ESTIMATING RELATIVE DISTANCES FOR RANGE-BASED TECHNIQUES

The estimation of relative distances between nodes is usually referred as *Ranging technique*. There are three main types of ranging techniques:

- Time of Arrival (ToA);
- Time Difference of Arrival (TDoA);
- Received Signal Strength (RSS).

In the ToA, the distance estimation is obtained by exploiting the propagation time of signals in radio communication. A simple model for the time of flight T_f of wireless signals is:

$$T_f = \frac{d}{c} \qquad \text{where} \begin{cases} d & \text{is the distance betwen nodes} \\ c & \text{is the propagation speed} (c = 2,997 \cdot 10^5 \text{km/s}) \end{cases}$$

Two methods can be used: One-way ToA, and Two-way ToA.

In one-way ToA, node A sends signal at time t_1, the signal arrives to B at time t_2. In this way node B is able to estimate the relative distance just exploiting:
$T_f = t_2 - t_1$

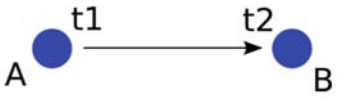


Figure 4: One-way ToA.

In two-way ToA, node A sends signal at time t_1 and the same arrives to node B at time t_2. Node B computes the message in t_d time and then sends a response to node A at time t_3. The same arrives at A at time t_4. The time of flight is then calculated using:
$T_f = \frac{(t_2-t_1)+(t_4-t_3)}{2}$.

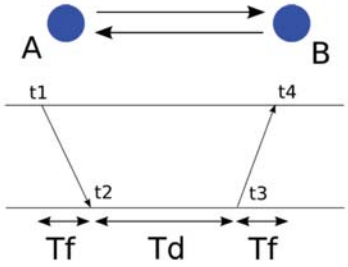


Figure 5: Two-way ToA.

In the TDoA technique, an estimation for the speed of propagation is performed in order to avoid errors linked to obstacles in unknown environments.
In particular, node A transmits two types of signals: RF (Radio Frequency signal) and US (Ultrasound Pulse signal), traveling at different speeds. A graphical explanation of signals sent is shown in the figure on the right. Receiver then estimates relative distances by exploiting the difference of times of arrival of the two signals.

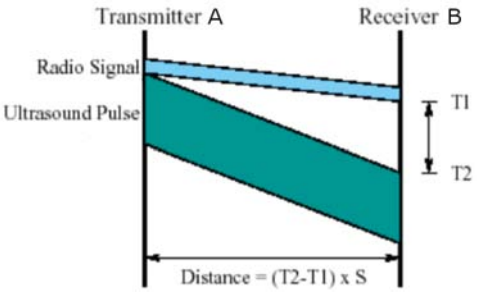


Figure 6: TDoA.

RSS techniques are based on power attenuation during the propagation of a transmitted signal. Every receiver is thus able to estimate relative distances by evaluating the strength of the signal received. The estimate of

a received signal (in dBm) is performed by the Received Signal Strength Indicator (RSSI). The main advantage of this method is that it does not require additional hardware, however it guarantees less precision than previous presented techniques. In fact, signals could be reflected, absorbed and attenuated by obstacles. This is mainly used in application where energy consumption is very relevant and no further hardware can be mounted on agents. In particular, the distance from the transmitter could be evaluated exploiting *Friis* equation:

$$P_R = P_T \frac{G_T G_R \lambda^2}{(4\pi)^2 d^n}$$

where:

- P_R, P_T: power of received and transmitted signals [Watt];
- G_R, G_T: receiver and transmitter antennae gain [ ];
- $\lambda = \frac{c}{f}$: wavelength [m];
- d: distance [m];
- n: signal propagation constant.

ESTIMATING RELATIVE ANGLES FOR ANGLE-BASED TECHNIQUES

AoA techniques provides estimations for relative angles between nodes. Agents exploiting AoA techniques are equipped with directional antennae in order to estimate the angle of arrival of signals. As explained by Figure 7, node u measures the angle of arrival of signals thanks to an angular measurements-system. This types of methods are the mainly used, as they enable to compute easily both the location and orientation of mobile robots. Unfortunately some aspects influence performance of these algorithm, such as inaccurate angle measurement, interference, obstacles.

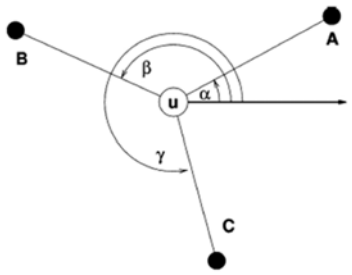


Figure 7: Angle of arrival AoA.

These methods are in general preferred to the other described in robotics application. This because accuracy of angles estimation does not degrade

with the increasing of distances between beacons and the target. In fact, this method results as one of the most reliable as accuracy of angles estimation depends only on the presence or absence of obstacles in the field of view.

3.2.1 *Methods overview*

One of the first comprehensive reviewing work has been carried out by Cohen and Koss [8]. In their paper, triangulation algorithms have been classified into four groups:

1. Geometric Triangulation;
2. Geometric Circle Intersection;
3. Iterative Methods (Iterative Search, Newton-Raphson, etc.);
4. Multiple Beacons Triangulation.

The first group makes an intensive use of trigonometric functions and exploits relative angles between landmarks as viewed from the robot. The solution can be computed in a closed form. The calculus is based on the geometry of the locations of landmarks and the location of the robot. Primitive types of these algorithms required properly ordered landmarks.

Algorithms of the second group determine parameters (radius and center) of the two circles passing through a couple of beacons and the robot. The robot position is then deduced by computing the intersection between these two circles and exploiting angles measured by the robot. This type is the most popular for solving the three object triangulation problem. Many variations and improvements have been proposed [35] [34] in order to reduce limitations of the original method such as working for any beacon ordering and to work outside the triangle formed by the three beacons.

The third group linearizes the trigonometric relations to converge to the robot position after some iterations, exploiting the Newton-Raphson method [50].

Iterative methods, based principally on *Iterative search* algorithm, which consists in searching the robot position through the possible space of orientation, and by using a closeness measure of the solution. The fourth group addresses the more general problem of finding the robot pose from more than three angle measurements, which is a overdetemined problem. This solution is preferable in presence of massive measurement noise in sensors.

In general, if the setup contains three beacons only, or if the robot has limited on-board processing capabilities, *Geometric Triangulation* and *Geometric circle intersection* are the best candidates. *Iterative methods* and *Multiple Beacons Triangulation* are appropriate if the application must handle

multiple beacons and if it can accommodate a higher computational cost. The main drawback of *Iterative methods* is the convergence issue, while *Multiple Beacons Triangulation* has the disadvantage of high computational costs. The drawback of the first and second group are usually a lack of precision related to: the consistency of the methods when the robot is located outside the triangle defined by the three beacons, the strategy to follow when falling into some particular geometrical cases, the reliability measure of the computed position. Simple methods of the first and second groups usually fail to propose a proper answer to all these concerns.

Therefore, the focus of this thesis is on the use of a specific hardware for localization based on AoA (radio, laser scanner etc.) and artificial landmarks for the position estimation of a mobile robot. Geometric triangulation is therefore the method preferred for the reasons just explained.

3.3 TRIANGULATION

It is difficult to compare all the above mentioned methods, because they operate in different conditions and have distinct behavior. In practice, the choice is dictated by the application requirements, and some compromises.

Thanks to the availability and accuracy of angle measurements systems, geometric triangulation has emerged as a widely used, robust, accurate, and flexible technique [15]. Furthermore, in geometric triangulation the robot can compute its orientation in addition to its position, so that the complete pose of the robot can be found. This feature is proper of geometric triangulation methods, and in general alternative methods presented do not guarantee this behavior. For the reasons presented, triangulation methods are the more suitable for the assumptions done in this work.

Figure 8 illustrates a typical triangulation setup. Here the problem is referred to a 2-D plane. Angles ϕ_1, ϕ_2 and ϕ_3 may be used by a triangulation algorithm in order to compute the robot position (x_R, y_R) and orientation θ.

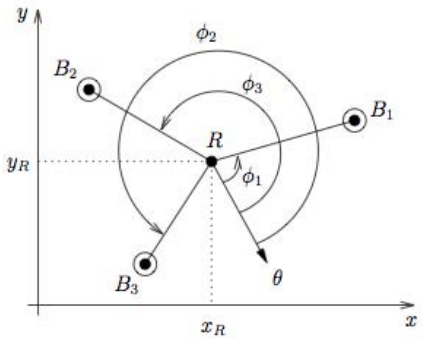


Figure 8: Triangulation setup in the 2-D plane. R denotes the robot, B_1, B_2 and B_3 are the beacons. ϕ_1, ϕ_2 and ϕ_3 are the angles for beacons, measured relatively from the robot orientation frame.

3.3.1 *ToTal algorithm*

This section deals with the description of a recently-proposed in literature algorithm based on geometric triangulation. This algorithm was firstly presented in [39] and named *ToTal*. Many are the advantages, with respect to previous approaches, presented by Pierlot and Van Droogenbroeck in their algorithm, including:

1. ToTal is independent of beacons ordering;
2. ToTal is independent of the relative position of the robot and the beacons;
3. the algorithm is faster and simpler with respect to previous presented in literature;
4. the simplified logic enables to determine a criterion to qualify the reliability of the computed position.

ToTal algorithm belongs to the family of *Geometric Circle Intersection* algorithms, and it could be divided into two main phases:

- first the parameters of the three circle passing through the robot and the three pairs of beacon are computed;
- in the second phase the intersection of these three circles is computed, by using all the three circles parameters.

The main contribution of the method presented, as motivated in the paper, is a simplification in the mathematical equations involved. Moreover, it is important to notice that these simplifications lead the algorithm to work properly in the entire plane.

3.3.1.1 *Description of the algorithm*

In this part, a detailed description of the equations and assumptions done in [39] is reported. It is important to notice that most of the considerations done in the following are common to almost every geometric triangulation algorithm.

Each mobile node, in order to localize itself exploiting three available beacons, requires:

- to measure relative angles with the three beacons;
- to be able to distinguish beacons among them;
- that angles measurements from the beacons are taken separately (in an independent way one from each other) and measured relatively to a reference angle θ (representing, for instance, the orientation of the mobile robot with respect to the world coordinate reference).

We start by determining the locus of the robot position R, that see two fixed beacons, B_1 and B_2.

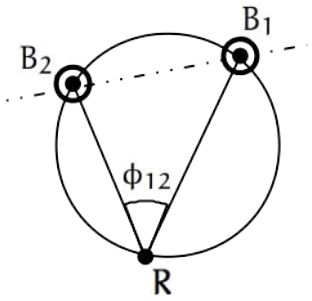


Figure 9: Locus of robot positions given the angle between beacons as seen from the robot.

Proposition 3.3.1 (Locus of robot positions)**.** *Given the angle between* B_1 *and* B_2 *as seen from the robot,* ϕ_{12}*, the locus of possible robot positions is an arc of the circle passing through* B_1*,* B_2 *and* R*.*

The proposition states that every point lying in the circle passing through the beacons and the robot is a valid location for the robot, given the angle ϕ_{12}. Moreover, it is important to notice that both cases $\phi_{12} < \pi$ and $\phi_{12} > \pi$ are acceptable values for the measured angle, as shown in Figure 10.

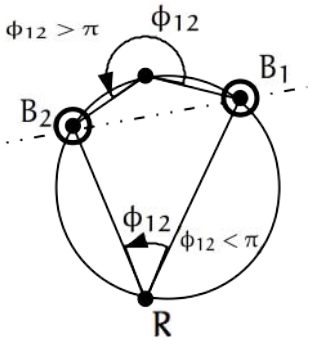


Figure 10: $\phi_{12} < \pi$ and $\phi_{12} > \pi$.

Proof. Let us consider the circumference passing through the two beacons and the robot as shown in Figure 11. Let A be a point lying on this circumference. Then every angle at the circumference $B_1\widehat{AB}_2$ has $B_1\widehat{OB}_2$ as angle at the center, for every choice of A.

At this point, to conclude, we just have to exploit the well-known theorem: *Angle at the center of circle is twice angle at circumference.*

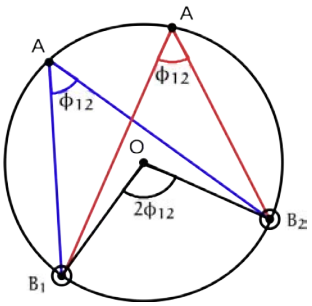


Figure 11: *Angle at the center of circle is twice angle at circumference.*

□

More precisely, this locus is composed of two arcs of circle, which are the reflection of each other through the line joining B_1 and B_2 as shown in Figure 12.

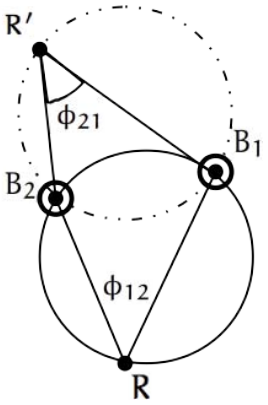


Figure 12: Arcs of circle is not uniquely determined.

This ambiguity may be avoided imposing that the measured angle between the couple of beacons B_1 and B_2, denoted as ϕ_{12} is computed as $\phi_{12} = \phi_2 - \phi_1$, natural choice for a Counter Clock Wise (CCW) rotating sensor. This is consistent with practical measurements and it removes the ambiguity about the locus; however it requires that beacons are properly ordered, and the robot must be capable to establish the index of every beacon.

As a result, the locus is a single circle passing through R, B_1 and B_2; and, in addition, the line joining B_1 and B_2 divides the circle into two parts: one for $\phi_{12} < \pi$ and the other for $\phi_{12} > \pi$ as shown in Figure 10.

The following steps presents the mathematical model for the locus of the possible positions of the robot.

The method proposed by Pierlot and Van Droogenbroeck makes use of the complex representation of 2-D points. This consists on expressing angles as the argument of complex numbers.

In this context, the angle ϕ_{12} can be written as:

$$\phi_{12} = \arg\left\{\frac{B_2 - R}{B_1 - R}\right\} \quad \Rightarrow \quad \phi_{12} = \arg\left\{(B_2 - R)(\overline{B_1 - R})\right\} \tag{1}$$

then, expressing R, B_1 and B_2 as complex numbers:

$$\begin{cases} R & \rightarrow (x + jy) \\ B_1 & \rightarrow (x_1 + jy_1) \\ B_2 & \rightarrow (x_2 + jy_2) \end{cases}$$

Equation 1 can be rewritten in the following way

$$arg\{(x_2 + jy_2 - x - jy)(x_1 - jy_1 - x + jy)\ e^{-j\phi_{12}}\} = 0$$

that is, extracting the argument to both member of equation:

$$\begin{aligned}[(x_2 - x)(x_1 - x)] \sin\phi_{12} + [(y_2 - y)(y - y_1)] \sin\phi_{12} + \\ + [(x_2 - x)(y - y_1)] \cos\phi_{12} + [(y_2 - y)(x_1 - x)] \cos\phi_{12} = 0\,. \end{aligned} \quad (2)$$

As previously introduced, the locus of possible position for the robot has the general form of a circumference in a plane. That is, it can be expressed in a implicit manner in the following form:

$$(x - x_{12})^2 + (y - y_{12})^2 = R_{12}^2$$

Equation 2 permits to derive the coordinates for the circle center:

$$x_{12} = \frac{(x_1 + x_2) + (y_1 - y_2)\cot\phi_{12}}{2} \quad (3)$$

$$y_{12} = \frac{(y_1 + y_2) - (x_1 - x_2)\cot\phi_{12}}{2} \quad (4)$$

and the squared radius:

$$R_{12}^2 = \frac{(x_1 - x_2)^2 + (y_1 - y_2)^2}{4\sin^2\phi_{12}}\,. \quad (5)$$

When we consider the generic circle passing through the more general beacons locations B_i and B_j, Equation 3 and Equation 4 modify as follow:

$$x_{ij} = \frac{(x_i + x_j) + (y_i - y_j)T_{ij}}{2} \quad (6)$$

$$y_{ij} = \frac{(y_i + y_j) - (x_i - x_j)T_{ij}}{2} \quad (7)$$

where T_{ij} denotes the $\cot(\cdot)$ of the bearing angle,

$$T_{ij} = \cot(\phi_{ij}) \quad (8)$$

Equation 5 becomes

$$R_{ij} = \frac{(x_i - x_j)^2 + (y_i - y_j)^2}{4\sin^2 \phi_{ij}} \,. \tag{9}$$

Equations (6), (7) c_{ij} and (9) completely describe the circle C_{ij} with center c_{ij} of coordinates (x_{ij},y_{ij}).

Each bearing angle ϕ_{ij} between beacons B_i and B_j, constraints the robot to be on a circle C_{ij}, passing through B_i, B_j and R as shown in Figure 13. Figure 13 shows a typical geometric triangulation setup in the 2-D plane.

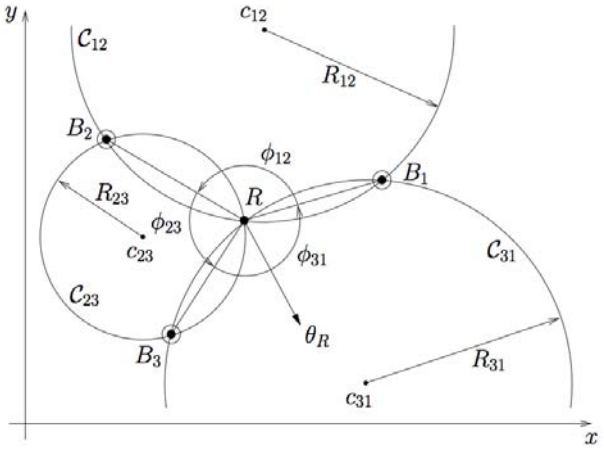


Figure 13: Triangulation setup in the 2-D plane.

It is important to underline that common methods usually are based on two separated steps:

1. initially only two of the three circles are considered: the first step consists on computing the intersection between them. The result is a couple of valid position for the robot;

2. The second step consists in intersecting the derived the couple of solutions with the the remaining circle in order to obtain a unique solution.

Unfortunately, this approach is quite expensive from a computational point of view, as it requires to solve a quadratic system. Moreover the choice of the two circle is arbitrary and usually fixed, whereas this choice could affect the accuracy of solutions.

The novel method proposed by Pierlot and Van Droogenbroeck, rather than classical geometrical triangulation methods, exploits all the three circles from the beginning, leading to a drastic reduction in the complexity of the problem involved and reducing the number of degrees of freedom in the choice of circles ordering. The main idea behind the above-mentioned method, consists in the notion of *radical axes* of a couple circles.

It is initially necessary to introduce the *power of a point relative to a circle* :

Definition 3.1. The *power of a point* p *relative to a circle* C, is defined by:

$$P_{C,p} = (x - x_c)^2 + (y - y_c)^2 - R^2$$

When the circles involved are two, the definition can be extended to *radical axes of a couple of circles*:

Definition 3.2. The *radical axis of a couple of circles* C_1 and C_2 is the locus of points having the same power with respect to both circles.

Example 3.3.1. Given a couple of circles, possible configurations are twofold:

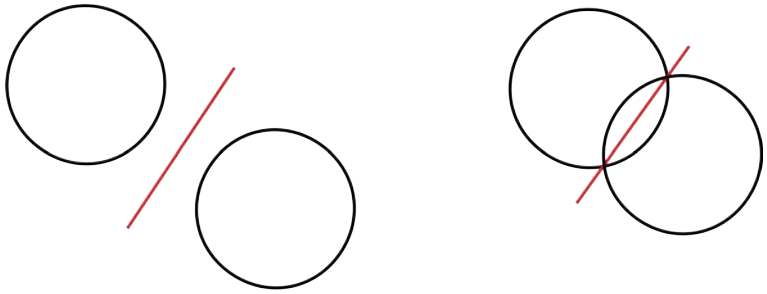


Figure 14: Radical axis of a couple of circles.

When three circles are considered, radical axes concerning every couple of them, intersect in a single point, as shown in Figure 15.

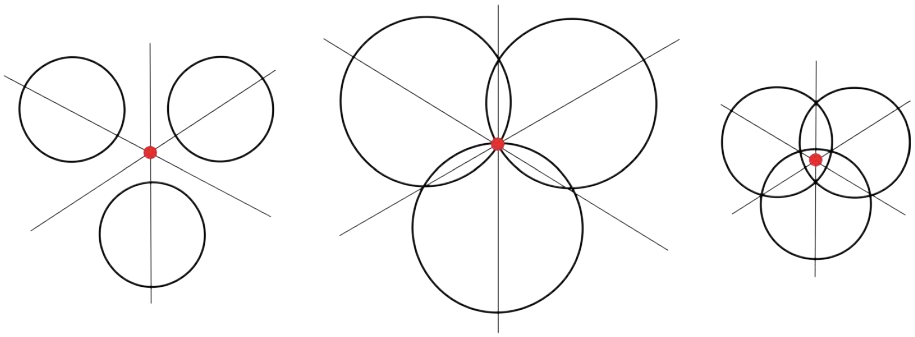


Figure 15: Radical axes concerning three circles. Possible arrangements are three. Red points represent intersections between radical axes.

It is important to underline that their intersection point is the unique point of the plane having the same power with respect to every single circle.

The example motivates the following:

Definition 3.3. The *Radical center of three circles* is the unique point of the plane in which the three radical axis, corresponding to each couple of circles, are concurring.

The central configuration shown in Figure 15 represent the situation in which the three radical axes intersect in a single point, corresponding exactly to the unique point of intersection between the circles. This case represent the ideal solution to the triangulation problem, i. e. when no noise affects measures.
On the other hand, when bearing angles are corrupted by noise, any configuration represented in Figure 15 is in general feasible. The configuration on the left, represent the circumstance in which circles does not present any intersection, while the figure on the right shows the case in which there exist multiple intersection points.

Although in presence of noise circumstances can be multiple, Figure 15 graphically clarify an important aspect: the (single) intersection between the three radical axis represent the point of minimal distance between the three circles, even in presence of noise. This property is a straight consequence of Definition 3.2. The observation just explained, motivates the choice of the intersection between radical axes as the most likely estimated location for the robot.

This fundamental observation is the reason behind the novel method proposed in ToTal algorithm. As will be shown, this is the main motivation that enables a computational reduction in the solution of the problem.

Assuming, as a preliminary hypothesis, absence of noise in measures, steps described in the following present the way proposed to solve the triangulation problem. It is important to remind that, according to the assumption just explained, the three circles intersect in a single point. Since only two of the three bearing angles are independent it is possible to explicate:

$$\phi_{31} = \phi_{32} + \phi_{21} \ .$$

By intersecting the three radical axes of the (three) couples of circles, we are able to obtain the radical center of the triplet:

$$\begin{aligned}
C_{12} \wedge C_{23} \quad &\longrightarrow \ (x-x_{12})^2 + (y-y_{12})^2 - R_{12}^2 = (x-x_{23})^2 + (y-y_{23})^2 - R_{23}^2 \\
C_{23} \wedge C_{31} \quad &\longrightarrow \ (x-x_{23})^2 + (y-y_{23})^2 - R_{23}^2 = (x-x_{31})^2 + (y-y_{31})^2 - R_{31}^2 \\
C_{31} \wedge C_{12} \quad &\longrightarrow \ (x-x_{31})^2 + (y-y_{31})^2 - R_{31}^2 = (x-x_{12})^2 + (y-y_{12})^2 - R_{12}^2
\end{aligned}$$

which leads to the following linear system:

$$\begin{cases}
(x_{12}-x_{23})x + (y_{12}-y_{23})y = k_{12} - k_{23} \\
(x_{23}-x_{31})x + (y_{23}-y_{31})y = k_{23} - k_{31} \\
(x_{31}-x_{12})x + (y_{31}-y_{12})y = k_{31} - k_{12}
\end{cases} \tag{10}$$

where the new introduced quantity k_{ij} is defined as follows:

$$k_{ij} = \frac{x_{ij}^2 + y_{ij}^2 - R_{ij}^2}{2} \ . \tag{11}$$

As a result, the solution to the geometrical triangulation problem presented - that was initially described as a geometrical and trigonometrical issue - has been re-conduced to the solution of a three linear equation-system. The coordinates of the power center, that is, the estimated robot position can then be obtained as the solution of the following linear system:

$$Ax = b \quad . \tag{12}$$

It is important to underline two important characteristics for the problem:

1. the linear system (12) consists of three equations and two unknowns;
2. any equation that appears in Equation 12 can be obtained as addition of the others.

It is important to notice that 2. mathematically proves that the three radical axis concur in an unique point, as previously introduced.

Linear system (12) can be expressed in matrix form, according to (12), in the following manner:

$$A = \begin{bmatrix} x_{12} - x_{23} & y_{12} - y_{23} \\ x_{23} - x_{31} & y_{23} - y_{31} \end{bmatrix}, \qquad b = \begin{bmatrix} k_{12} - k_{23} \\ k_{23} - k_{31} \end{bmatrix}$$

The computation of the solution through $(x_R, y_R) = A^{-1}b$, leads to:

$$x_R = \frac{(k_{12} - k_{23})(y_{23} - y_{31}) - (y_{12} - y_{23})(k_{23} - k_{31})}{\det(A)} \tag{13}$$

$$y_R = \frac{(x_{12} - x_{23})(k_{23} - k_{31}) - (k_{12} - k_{23})(x_{23} - x_{31})}{\det(A)} \tag{14}$$

where we have used:

$$\det(A) = (x_{12} - x_{23})(y_{23} - y_{31}) - (y_{12} - y_{23})(x_{23} - x_{31})$$

Therefore, Equation 13 and Equation 14 are the mathematical expressions that enable to compute actual position of the robot, when exploiting following data:

- the three circles centers coordinates $(x_{ij}, y_{ij}) \quad \forall\ i = 1,2,3\ ,\ j = 1,2,3 \quad i \neq j$;
- the three beacons locations $(x_i, y_i) \quad \forall\ i = 1,2,3$;
- the three bearing angles in order to compute $T_{ij} = \cot(\phi_{ij}) \quad \forall\ i = 1,2,3\ ,\ j = 1,2,3 \quad i \neq j$.

3.3.2 *Final version of ToTal algorithm and discussion*

It is possible to further simplify equations presented in order to produce an improved version for the algorithm.
First of all it is possible to replace expression Equation 9 for R_{ij}^2 in Equation 11, for k_{ij}, obtaining after some simplifications:

$$k_{ij} = \frac{x_i x_k + y_i y_j + T_{ij}(x_j y_i - x_i y_j)}{2} \, .$$

In addition, a further simplification (often used in many triangulation algorithm) consists in translating coordinates in such a way to locate the origin of the reference frame in the location of an arbitrarily-chosen beacon.
In this case, we arbitrarily choose B_2 as the origin, so other beacons coordinates become:

$$\begin{cases} B_1' = B_1 - B_2 = \{x_1 - x_2, y_1 - y_2\} \\ B_2' = B_2 = \{0, 0\} \\ B_3' = B_3 - B_2 = \{x_3 - x_2, y_3 - y_2\} \end{cases} \tag{15}$$

Then, noticing that the factor $\frac{1}{2}$ involved in (6) and (7), as well as in (11), for k_{ij}, cancels when used in robot position parameters, we can introduce modified circle center coordinates:

$$x_{ij}' = 2x_{ij},$$

$$y_{ij}' = 2y_{ij},$$

and modified parameter k_{ij}:

$$k_{ij}' = 2k_{ij} \, .$$

Therefore, robot position (x_R, y_R) can be computed exploiting:

$$x_R = x_2 + \frac{k_{31}'(y_{12}' - y_{23}')}{\det(A)} \tag{16}$$

$$y_R = y_2 + \frac{k_{31}'(x_{23} - x_{12})}{\det(A)} \tag{17}$$

a complete treatment of mathematical steps and a detailed description of equations involved can be found in [39].

Observation 1. There are two particular cases that require special treatment:

- infinite values of $\cot(\cdot)$ function, in (8);
- cases in which $\det(A) = 0$, in (13) and (14) that lead to invalid robot estimated position.

When a bearing angle ϕ_{ij} is equal to 0 or π, then the $\cot(\cdot)$ function assumes non-finite values. This is the case when the robot to be localized stands on the line joining a couple f beacons B_i and B_j.

Typical solutions very often exploited in literature consist on limiting the $\cot(\cdot)$ value to a minimum or maximum value, corresponding to a small angle that is far below the measurement precision. Typical values are $\pm 10^8$, which corresponds to an angle of about $\pm 10^{-8} \mathrm{rad}$.

More accurate solutions to some of these peculiar cases can be found in literature. For example, Pierlot and Van Droogenbroeck in [39] propose some adjustments in the algorithms, when one of the two angles measured is equal to 0 or π, which makes the method reliable.

However, it is important to underline that in such cases in which beacons and the robot are collinear, geometric triangulation is unable to find any solution to the problem as not enough parameters are available. In fact, this is the case in when the solution computed as intersection of the three circles is not a single point, but a locus. This problem is common to all geometric triangulation-based methods, nevertheless several solutions can be exploited as motivated in Observation 1.

In addition orientation of the robot θ_R may be determined by using beacons locations B_i and its corresponding angle ϕ_i, once the robot position is known, by exploiting:

$$\theta_R = \mathrm{atan2}(y_i - y_R, x_i - x_R) - \phi_i \tag{18}$$

where $\mathrm{atan2}(x, y)$ denotes the argument $\arg(x + jy)$ of the complex number $x + jy$.

Finally, it is important to underline that:

1. each localization iteration requires each mobile agent to transmit to every sensing beacon (at least three) a request;
2. each beacon sends a response containing its own position;
3. angles of arrival measurements are then taken by mobile agent;
4. the mobile robot is then able to perform geometric triangulation;

then each triangulation application requires the transmission of (at least) three data packet, the reception of as much packets, and an angle of arrival measurement.

Algorithm 1 proposes a detailed description for the algorithm in pseudo-code.

Algorithm 1 ToTal algorithm

Given the three beacon coordinates (x_1, y_1), (x_2, y_2), (x_3, y_3), and the three bearing angles ϕ_1, ϕ_2 and ϕ_3

1) Compute the modified beacon coordinates:

$x'_1 = x_1 - x_2, \qquad y'_1 = y_1 - y_2,$

$x'_3 = x_3 - x_2, \qquad y'_3 = y_2 - y_2,$

2) Compute the $\cot(.)$:

if $\phi_1 == \pi \vee \phi_2 == \pi$ **then**

$T_{ij} = -10e8$

else if $\phi_1 == 0 \vee \phi_2 == 0$ **then**

$T_{ij} = 10e8$

else

$T_{12} = \cot(\phi_2 - \phi_1)$

$T_{23} = \cot(\phi_3 - \phi_2)$

$T_{31} = \frac{1 - T_{12}T_{23}}{T_{12} + T_{23}}$

end if

3) Compute the modified circle center coordinates:

$x'_{12} = x'_1 + T_{12}y'_1, \qquad y'_{12} = y'_1 - T_{12}x'_1,$

$x'_{23} = x'_3 + T_{23}y'_3, \qquad y'_{23} = y'_3 - T_{23}x'_3,$

$x'_{31} = (x'_3 + x'_1) + T_{31}(y'_3 - y'_1),$

$y'_{31} = (y'_3 + y'_1) - T_{31}(x'_3 - x'_1)$

4) Compute k'_{31}:

$k'_{31} = x'_1x'_3 + y'_1y'_3 + T_{31}(x'_1y'_3 - x'_1)$

5) Compute $\det(A)$:

$\det(A) = (x'_{12} - x'_{23})(y'_{23} - y'_{31}) - (y'_{12} - y'_{23})(x'_{23} - x'_{31})$

if $\det(A) == 0$ **then**

Return with an error

end if

6) Compute the robot position (x_R, y_R)

$x_R = x_2 + \frac{k'_{31}(y'_{12} - y'_{23})}{\det(A)} \qquad y_R = y_2 + \frac{k'_{31}(x_{23} - x_{12})}{\det(A)}$

3.4 LOCATION VARIANCE DERIVATION

ToTal Algorithm proposed by Pierlot and Van Droogenbroeck provides a valuable way to obtain an estimation for robot position and orientation through a substantial reduction in the computation complexity of equations involved with respect to previously proposed methods. Exactly because of computational complexity of equations involved, general solutions to geometric triangulation problem do not allow to compute the accuracy of estimated position. An accurate estimation for the variance of the computed potion can be very useful in order to design agents and to understand the precision of estimation performed. Moreover, disposing of

location uncertain, as a function of beacons positions uncertain and angles of arrival uncertain, permits to create a statistical description for the model involved and to enforce statistical filtering to noisy measurements.

In [39] a sensitivity analysis of the computed position with respect to angles of arrival ϕ_1, ϕ_2 and ϕ_3 is performed. Although this leads an uncertain analysis of the computed position as function of the (non-linear) angles of arrival, it does not permit to take into account uncertain linked to non-exact positions of beacons.

The purpose of this work is to derive a detailed, closed-form expression for variance of computed position as a function of beacons locations uncertain and of angles of arrival uncertain. In order to accomplish this task, the *propagation of uncertainty law* has been exploited. Since functions involved in geometric triangulation are trigonometric, i. e. non-linear, they have to be linearized by approximation to a first-order *Taylor series* expansion.

Example 3.4.1. In general contexts, given the non-linear function:

$$f(x) : \mathbb{R}^n \to \mathbb{R}^m,$$

the Taylor expansion would be, assuming that x is concentrated around its mean $M(x)$:

$$f(x) \simeq f(M(x)) + \left[\frac{\partial f(x)}{\partial x}\right]_{M(x)} (x - M(x)) \, .$$

Since $f(M(x))$ is constant as function of x, it does not contribute to the error in f. Therefore, the covariance propagation follows the linear rule:

$$Cov_{f(x)} = \left[\frac{\partial f(x)}{\partial x}\right] Cov_x \left[\frac{\partial f(x)}{\partial x}\right]^T = J \cdot Cov_x \cdot J^T \, , \tag{19}$$

where J indicates the Jacobian matrix of $f(x)$:

$$J = \frac{\partial f(x)}{\partial x}$$

and Cov_x is the covariance matrix of vector x.

The derivation of a closed-form expression for uncertain, proposed in this work, makes use of geometric properties of circles employed in the three object triangulation problem. In the following we motivate our choice of exploiting the uncertain in the three circles areas to compute uncertain on estimated robot position.

Figure 16 presents a simple case in which we deal with uncertainty in the knowledge about position of one of the three beacons employed B_1. The yellow circle represents uncertain on location for node B_1, while it is located exactly in the mean value of its distribution, identified by black an white dot patterns. It is important to notice that this type of uncertain

could be associated to both uncertain on the actual position of the node, and to the uncertain on the measured angle performed by the robot.

Assuming that actual position of the beacon can be modeled as a Gaussian random variable:

$$(x_1, y_1) \sim \mathcal{N}((\overline{x_1}, \overline{y_1}), \mathbf{R}_{B1})$$

it is possible to state that actual beacon location is inside the yellow circle drawn in Figure 16 and described by:

$$C_{B1} = \{(x, y) : (\overline{x_1}, \overline{x_1}) - 3\mathbf{R}_{B1} \leqslant (x, y) \leqslant (\overline{x_1}, \overline{x_1}) + 3\mathbf{R}_{B1}\}$$

with probability $p = 0.99$ (Chebyshev's inequality).

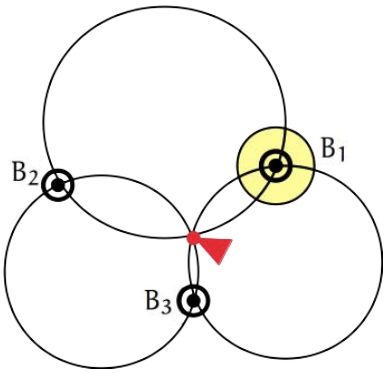


Figure 16: Triangulation when B_1 stands on the mean value of its position distribution.

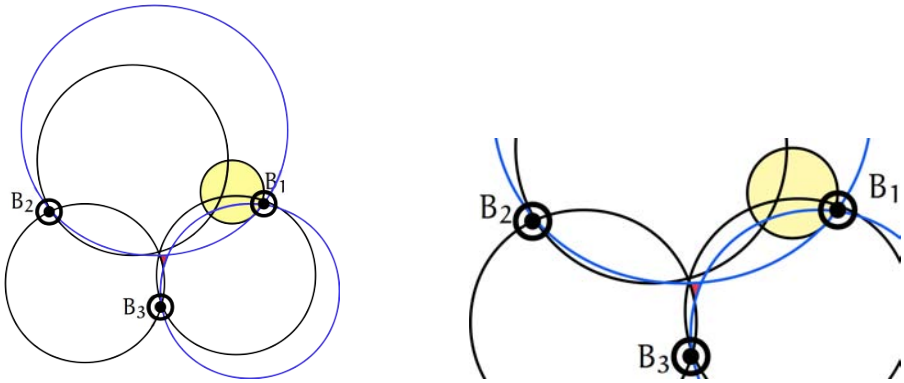
On the other hand Figure 17 represents the situation in which node B_1 does not stand exactly in the mean value of its distribution. In this case, as can be noticed from the figure, a variation in B_1 produces alterations in triangulation circle (circles passing through B_1) both in circle's center and in circle's area.

Furthermore, as shown in the zoom in Figure 17b, the three circles does not intersect in an unique point anymore. This graphical interpretation motivates the choice of using circles' areas as estimates for the uncertain in the location of the robot. In fact, noisy measurements leads to circles that do not intersect in an unique point, leading to multiple solutions for robot position.

In the following, it is reasonable to make following assumptions:

1. absolute orientation of the robot, θ_R is not affected by noise;
2. angles measurements taken between the robot and the three beacons are taken separately and relatively to a reference angle θ_R, i. e. beacon coordinates are independent one from the other.

The former can easily be removed, and its purpose is to reduce uncertainty on interpretation of results, while the latter ensures that beacons



(a) Triangulation when B_1 does not stand on the mean value of its position distribution.

(b) Zoom of Figure 17a. Note that the circles does not intersect in a unique point.

Figure 17: Triangulation when the position of a beacon is not deterministic. Yellow filled circle represent uncertain in knowledge of position of B_1. Black circles show the three object triangulation when B_1 is located exactly in the mean value of its distribution (same case as Figure 16). Blue circles shows how new circles alter when actual B_1 position change. Notice that in the second case the three circles does not intersect in an unique point.

location and their locations, as seen from the robot, are uncorrelated random variables , i. e. the joint covariance matrix is:

$$Cov_x = \begin{bmatrix} \sigma^2_{x_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{x_2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{y_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{x_3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{y_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{\phi_1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{\phi_2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \sigma^2_{\phi_3} \end{bmatrix}$$

According to these assumptions, modified beacons coordinates' variance can be computed exploiting Equation 15:

$$\begin{cases} \sigma^2_{x'_1} = \sigma^2_{x_1} + \sigma^2_{x_2} \\ \sigma^2_{y'_1} = \sigma^2_{y_1} + \sigma^2_{y_2} \\ \sigma^2_{x'_3} = \sigma^2_{x_3} + \sigma^2_{x_2} \\ \sigma^2_{y'_3} = \sigma^2_{y_3} + \sigma^2_{y_2} \end{cases}$$

Moreover, the expression for the variance of random variables T_{ij} employed in the second step of Algorithm 1, can be derived by using Equation 19:

$$Var(T_{12}) = \begin{bmatrix} \frac{\partial T_{12}}{\partial \phi_2} & \frac{\partial T_{12}}{\partial \phi_1} \end{bmatrix} \begin{bmatrix} \sigma^2_{\phi_2} & 0 \\ 0 & \sigma^2_{\phi_1} \end{bmatrix} \begin{bmatrix} \frac{\partial T_{12}}{\partial \phi_2} \\ \frac{\partial T_{12}}{\partial \phi_1} \end{bmatrix}$$

where,

$$\frac{\partial T_{12}}{\partial \phi_2} = \frac{\partial \cot(\phi_2 - \phi_1)}{\partial \phi_2} = -\frac{1}{\sin^2(\phi_2 - \phi_1)}$$
$$\frac{\partial T_{12}}{\partial \phi_1} = \frac{\partial \cot(\phi_2 - \phi_1)}{\partial \phi_1} = \frac{1}{\sin^2(\phi_2 - \phi_1)}$$

that leads to:

$$Var(T_{12}) = \frac{\sigma^2_{\phi_2}}{\sin^4(\phi_2 - \phi_1)} + \frac{\sigma^2_{\phi_1}}{\sin^4(\phi_2 - \phi_1)} := \sigma^2_{T_{12}}$$

In a similar way, we can obtain for T_{23}:

$$Var(T_{23}) = \frac{\sigma^2_{\phi_3}}{\sin^4(\phi_3 - \phi_2)} + \frac{\sigma^2_{\phi_2}}{\sin^4(\phi_3 - \phi_2)} := \sigma^2_{T_{23}} \quad .$$

Then, computing:

$$\frac{\partial T_{31}}{\partial T_{12}} = \frac{-T_{23} - (1 - T_{12}T_{23})}{(T_{12} + T_{23})^2} = \frac{-T_{23}(T_{12} - 1) - 1}{(T_{12} + T_{23})^2}$$
$$\frac{\partial T_{31}}{\partial T_{23}} = \frac{T_{12} - (1 - T_{12}T_{23})}{(T_{12} + T_{23})^2} = \frac{-T_{12}(T_{23} - T_{12}) - 1}{(T_{12} + T_{23})^2}$$

we obtain, for T_{31} :

$$\begin{aligned} Var(T_{31}) &= \begin{bmatrix} \frac{\partial T_{31}}{\partial T_{12}} & \frac{\partial T_{31}}{\partial T_{23}} \end{bmatrix} \begin{bmatrix} \sigma^2_{T_{12}} & 0 \\ 0 & \sigma^2_{T_{23}} \end{bmatrix} \begin{bmatrix} \frac{\partial T_{31}}{\partial T_{12}} \\ \frac{\partial T_{31}}{\partial T_{23}} \end{bmatrix} \\ &= \sigma^2_{T_{12}} \left[\frac{T_{23}(T_{12} - 1) - 1}{(T_{12} + T_{23})^2} \right] + \sigma^2_{T_{23}} \left[\frac{T_{12}(T_{23} - T_{12}) - 1}{(T_{12} + T_{23})^2} \right] := \sigma^2_{T_{31}} \quad . \end{aligned}$$

The third step of ToTal, provides for modified circle center coordinates. By using (19), it is possible to derive related variances:

$$\begin{aligned} \sigma^2_{x'_{12}} &= \begin{bmatrix} 1 & T_{12} & y'_1 \end{bmatrix} \begin{bmatrix} \sigma^2_{x'_1} & 0 & 0 \\ 0 & \sigma^2_{y'_1} & 0 \\ 0 & 0 & \sigma^2_{T_{12}} \end{bmatrix} \begin{bmatrix} 1 \\ T_{12} \\ y'_1 \end{bmatrix} \\ &= \sigma^2_{x'_1} + T^2_{12}\sigma^2_{y'_1} + y'^2_1 \sigma^2_{T_{12}} \quad , \end{aligned}$$

$$\sigma^2_{y'_{12}} = \begin{bmatrix} 1 & -T_{12} & -x'_1 \end{bmatrix} \begin{bmatrix} \sigma^2_{y'_1} & 0 & 0 \\ 0 & \sigma^2_{x'_1} & 0 \\ 0 & 0 & \sigma^2_{T_{12}} \end{bmatrix} \begin{bmatrix} 1 \\ -T_{12} \\ -x'_1 \end{bmatrix}$$
$$= \sigma^2_{y'_1} + T^2_{12}\sigma^2_{x'_1} + x^{2'}_1\sigma^2_{T_{12}}$$

and, in a similar way for $\sigma^2_{x'_{23}}$, $\sigma^2_{y'_{23}}$, $\sigma^2_{x'_{31}}$ and $\sigma^2_{y'_{31}}$:

$$\sigma^2_{x'_{23}} = \sigma^2_{x'_3} + T^2_{23}\sigma^2_{y'_3} + y'^2_3\sigma^2_{T_{23}}$$
$$\sigma^2_{y'_{23}} = \sigma^2_{y'_3} + T^2_{23}\sigma^2_{x'_3} + x'^2_3\sigma^2_{T_{23}}$$
$$\sigma^2_{x'_{31}} = \sigma^2_{x'_3} + \sigma^2_{x'_1} + T^2_{31}\sigma^2_{y'_3} + y'^2_3\sigma^2_{T_{31}} + T^2_{31}\sigma^2_{y'_1} + y^{2'}_1\sigma^2_{T_{31}}$$
$$\sigma^2_{y'_{31}} = \sigma^2_{y'_3} + \sigma^2_{y'_1} + T^2_{31}\sigma^2_{x'_3} + x'^2_3\sigma^2_{T_{31}} + T^2_{31}\sigma^2_{x'_1} + x^2_1\sigma^2_{T_{31}}$$

Then, an expression for the variance of the parameter k'_{31} can be computed:

$$\sigma^2_{k'_{31}} = x^{2'}_1\sigma^2_{x'_3} + x^{2'}_3\sigma^2_{x'_1} + y^{2'}_1\sigma^2_{y'_3} + y^{2'}_3\sigma^2_{y'_1} +$$
$$+ (x'_1y'_3)^2\sigma^2_{T_{31}} + (T_{31}y'_3)^2\sigma^2_{x'_1} + (T_{31}x'_1)^2\sigma^2_{y'_3} +$$
$$+ (x'_3y'_1)^2\sigma^2_{T_{31}} + (T_{31}y'_1)^2\sigma^2_{x'_3} + (T_{31}x'_3)\sigma^2_{y'_1}$$

and an expression for the variance of $\det(A)$:

$$\sigma^2_D = x^{2'}_{12}\sigma^2_{y'_{23}} + y^{2'}_{23}\sigma^2_{x'_{12}} + x^{2'}_{12}\sigma^2_{y'_{31}} +$$
$$+ y'_{31}\sigma^2_{x'_{12}} + x'_{23}\sigma^2_{y'_{23}} + y^{2'}_{23}\sigma^2_{x'_{23}} +$$
$$+ x^{2'}_{23}\sigma^2_{y'_{31}} + y^{2'}_{31}\sigma^2_{x'_{23}} + y^{2'}_{12}\sigma^2_{x'_{23}} + x^{2'}_{23}\sigma^2_{y'_{12}} +$$
$$+ y^{2'}_{12}\sigma^2_{x'_{31}} + x^{2'}_{31}\sigma^2_{y'_{12}} + y^{2'}_{23}\sigma^2_{x'_{23}} +$$
$$+ x^{2'}_{23}\sigma^2_{y'_{23}} + y^{2'}_{23}\sigma^2_{x'_{31}} + x^{2'}_{31}\sigma^2_{y'_{23}}$$

Finally, the variance of estimated position of the robot (x_R, y_R) can be derived. Let us define:

$$B = \begin{bmatrix} 1 & \frac{y'_{12}-y'_{23}}{\det(A)} & \frac{k'_{31}}{\det(A)} & -\frac{k'_{31}}{\det(A)} & -\frac{k'_{31}(y'_{12}-y'_{23})}{\det(A)} \end{bmatrix}$$

the variance for x-coordinate estimated position of the robot:

$$\sigma^2_{x_R} = B^T \begin{bmatrix} \sigma^2_{x'_2} & & & & \\ & \sigma^2_{k'_{31}} & & & \\ & & \sigma^2_{y'_{12}} & & \\ & & & \sigma^2_{y'_{23}} & \\ & & & & \sigma^2_D \end{bmatrix} B =$$
$$= \sigma^2_{x'_2} + \left(\frac{y'_{12}-y'_{23}}{\det(A)}\right)^2\sigma^2_{k'_{31}} + \sigma^2_{y'_{12}}\left(\frac{k'_{31}}{\det(A)}\right)^2 +$$
$$+ \sigma^2_{y'_{23}}\left(\frac{k'_{31}}{\det(A)}\right)^2 + \left(\frac{k'_{31}(y'_{12}-y'_{23})}{\det(A)}\right)^2\sigma^2_D \quad (20)$$

and for y-coordinate:

$$\sigma^2_{y_R} = \sigma^2_{y'_2} + \left(\frac{x'_{23} - x'_{12}}{\det(A)}\right)^2 \sigma^2_{k'_{31}} + \sigma^2_{x'_{12}} \left(\frac{k'_{31}}{\det(A)}\right)^2 + \\ + \sigma^2_{x'_{23}} \left(\frac{k'_{31}}{\det(A)}\right)^2 + \left(\frac{k'_{31}(x'_{23} - x'_{12})}{\det(A)}\right)^2 \sigma^2_D \quad (21)$$

Therefore, equations (20) and (21) provide expressions for variances attempted.

Example 3.4.2. For instance, let us consider three fixed beacons located in the vertexes of an equilateral triangle. This equilateral formation is to prefer as occlusions are minimized.
Let beacon positions $(x_{B,i}; y_{B,i})$, $i = 1, 2, 3$ be Gaussian random variables with covariance:

$$Var\left((x_{B,i}; y_{B,i})\right) = \begin{bmatrix} \sigma^2_B & 0 \\ 0 & \sigma^2_B \end{bmatrix}$$

with $\sigma^2_B = 0.01 m^2$. Figure 18 shows estimated position uncertain (in x and y directions) as a function of robot position in the whole mission space.
A more detailed discussion can be found in Chapter 6.

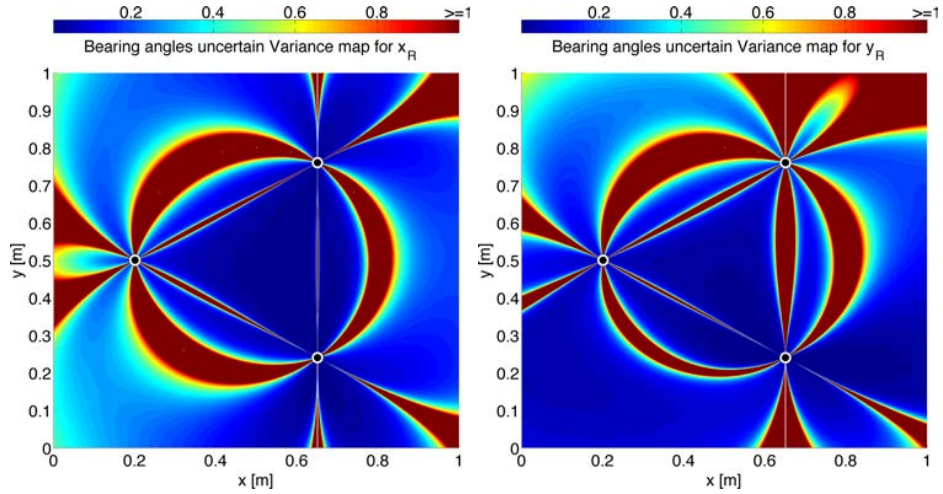


Figure 18: Robot position variance in m^2, when beacons positions are Gaussian random variables with variance $\sigma^2_B = 0.01 m^2$.

Notice that uncertain in the knowledge of all the three beacons positions coincides with uncertai on the bearing angle measured by the robot.

3.5 KALMAN FILTER-BASED NAVIGATION

Geometric triangulation-based algorithm presented in the previous sections suffers from several hypothesis that in many practical cases are quite

restrictive. For example, a mobile robot implementing the above-mentioned method, requires to dispose of (almost) three beacons at any given time. Unlikely this requirement is quite restrictive for purposes of this work, as our initial aim was to provide a reliable, high-rate and robust localization technique. Furthermore, each localization application requires the exchange of almost three data packets, with consequent substantial energy consumption.

According to energy-saving reasons just exposed, many practical cases requires the application to localize mobile agents without infra-agents cooperation. This motivates the use of independent localization algorithms. Therefore, this requires the robot to be equipped with hardware and software capable of providing a sensory feedback related to its environment.

One of the most famous relative positioning technique is the odometry, which consists of counting the number of wheel revolutions to compute relative offset to a known initial position. This method results quite accurate for small offset but it is not reliable because of the unbounded accumulation of errors over the time (due to wheel slippage, imprecision in wheels circumference, or wheels base). Furthermore odometry needs an initial position and fails when the robot is reset, since the reference position is unknown or modified. An absolute positioning system is thus required to re-calibrate the robot position periodically. These are the main reasons because odometry is not reliable in practical cases and methods that employs anchor nodes are usually preferable.

Nevertheless the flexibility and the simplicity of this methods enables to be employed in such cases in which not-structured environments are available, as described in the first part of this section.

Advantages described, and the duality that can be deduced between anchor-based methods and odometry, motivate the study of novel methods capable of combining the two types of approaches. In particular, initial motivations were to design and study a novel model:

- capable of working in the entire plane even when beacons rare;
- not affected by unbounded accumulation of error;
- that does not require additional hardware;
- energy consistent.

The innovative approach we propose, consists of combining geometric triangulation and odometry through through statistical filtering. In particular, the Kalman filter theory arise as a particularly suitable model for purposes just explained.
Figure 19 clarifies cooperation between odometry (kinematic model) and geometric triangulation, here exploited as (noisy measures) for the process modeled.

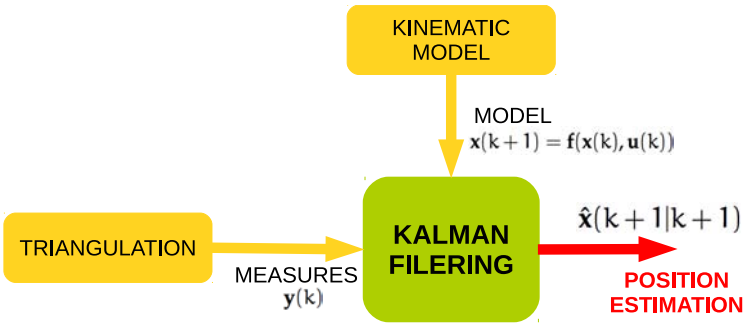


Figure 19: Novel approach proposed consists of combining geometric triangulation and odometry.

This approach has several advantages, such as robot's displacements are not required to be small and leads to an improvement in the hybrid use of the two methods, in particular triangulation could be used in bigger periods or only when the estimate uncertain overcome a fixed threshold, in this way obtaining advantages of both methods.

The Extended Kalman filter EKF has been used as it raises as a natural choice in this context since it provides a convenient way to fuse the data from multiple sensors i.e. odometry and triangulation. While in general the theory of Kalman filtering requires the process to be modeled using a linear dynamic model and a linear output model, the EKF enables the use of non-linear equations. In this work, the dynamic model representing the robot's wheels is nonlinear as better explained in Section 3.5.4.

In the following we propose, as an example, the modeling for a wheeled mobile robot and for a quadrotor. The model proposed for wheeled robot will then exploited in Chapter 6 to perform numerical simulations.

3.5.1 *Robot kinematic modeling*

WMRs provide flexible motion capabilities to robotic systems, in particular when we deal with reasonably smooth grounds and surfaces. Several mobility configurations (wheel number and type, their location and actuation, single-body or multi-body vehicle structure) can be chosen according to specific applications. The most common for single-body robots are *differential drive* and *synchro drive*, *tricycle* or *car-like drive*, *omnidirectional steering* etc.. Detailed reference on the analytic study of the kinematics of WMRs is [1].

Quadrotors also have become a widely used platform for many applications that include both indoor and outdoor. High performance combined with an increasing availability and reduced prices have led to a growing interest in using rotor based platforms.

In the following the kinematic models of two different types of widely used robots are presented. Kinematics is the study of geometry of motion. In the context of mobile robots, we are interested in determining the motion of the robot from the geometry of the constraint imposed by the

motion of the wheels. Kinematics helps the development of a model representing this types of systems in order to implement the control strategies.

3.5.2 *WMRs unicycle*

Mobile robots for operation on flat terrain permit several simplifying features that make them easier to model. In particular, many robots have two independently-driven coaxial wheels. The speed difference between both wheels results in a rotation of the vehicle about the center of the axle while the wheels act together in order to produce motion in the forward or reverse direction. Second, these robots are two-dimensional and lack suspensions. In general, suspensions compensates for vertical motion caused by the vehicles dynamics at high speeds. Mobile robots operate at relatively low speeds and, in the following, we assume absence of vertical motion. In the remaining part of this section, a model for a unicycle (Figure 21) is presented.

Consider a system with N particles, P_i ($i = 1, \ldots, N$), and their positions vector r_i in some reference frame. Each vector r_i can be written as a set of components (x_i, y_i, z_i). The 3N components specify the configuration of the system. The resulting Euclidean space:

$$C = \{X | X \in \mathbb{R}^{3N}, \quad X = [x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_N, y_N, z_N]\}$$

is called the *configuration space*.

The 3N scalar numbers are called configuration space variables or coordinates for the system. The trajectories of the system in the configuration space are always continuous. They may however have corners, double points, or points with multiple intersections. Corners are points at which the velocity is zero or discontinuous. Note that a velocity is discontinuous only when it is subject to an impulse.

Considering a system composed of N rigid bodies, B_i ($i = 1, \ldots, N$), each rigid body has three coordinates for planar systems, and six for spatial systems. Thus for planar systems, if each rigid body has coordinate (x_i, y_i, θ_i) in some reference frame, the configuration space can be written as the Cartesian product of the configuration spaces of individual rigid bodies. Note that θ_i belongs to a subset of $\mathbb{R}$, as it is an angle and only takes values in $[0, 2\pi)$ and sometimes denoted by S^1 (One-dimensional sphere). Thus, the configuration space for a single rigid body in the plane is called the *special Euclidean group* in two-dimensions, and denoted by $SE(2)$:

$$SE(2) = \mathbb{R} \times \mathbb{R} \times S^1 \quad .$$

The configuration space of N planar rigid bodies is:

$$C = \{X | X \in SE(2) \times SE(2) \times \cdots \times SE(2), X = [x_1, y_1, z_1, x_2, y_2, z_2, \ldots, x_N, y_N, z_N]\} \quad .$$

Consider a system of N planar rigid bodies. We have already seen that there is a 3N-dimensional configuration space associated with the system. However, when there are one or more configurations constraints (as in the

case of planar kinematic chains) not all of the 3N variables describing the system configuration are independent. The minimum number of variables to completely specify the configuration (position and orientation of every particle) of a system is called the number of degrees of freedom for that system. Thus, if there are m independent configuration constraints, the number of degrees of freedom is given by:

$$n = 3N - m \quad .$$

Constraints on the position of a system of particles are called *holonomic constraints*. The positions of the particles are constrained by holonomic equations. The system is constrained to move in a subset of the 3N-dimensional configuration space.

In contrast to holonomic constraints, in which the position of the particles are constrained, we may have constraints in which the velocities of the particles are constrained but the positions are not.

A rigid body can be described by the coordinates of a reference point C that is the single point of contact on the plane (x, y) and the angle (θ) between the longitudinal axes and the x-axes. If the body cannot slide in a lateral direction (the speed of the point C in lateral direction $(\mathbf{e}_l)$ must be zero), the velocity of the point C must be along the longitudinal axis (vector $\mathbf{e}_f$), where $\mathbf{e}_l$ and $\mathbf{e}_f$ have the meaning explained by Figure 20.

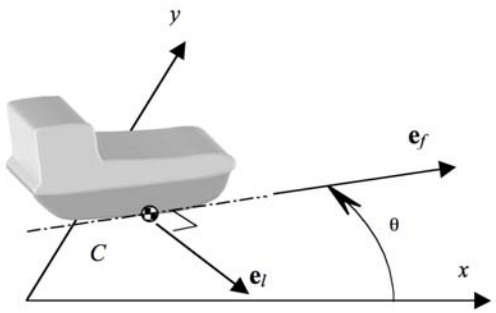


Figure 20: Position of a body in the plane. It is described by three coordinates: x, y and θ. If there is no slip in lateral direction, its velocity can only be along the vector e_f.

The position and orientation i.e. the *pose* of the robot is given by a 3×1 vector:

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

and differentiating this equation gives us the velocity:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$$

while the vectors e_f and e_l are defined by:

$$\mathbf{e}_f = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \quad \mathbf{e}_l = \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} \quad .$$

DYNAMIC MODEL

In order to derive a dynamic model for the system, and considering the reduction to the 2-D plane, the longitudinal component of velocity:

$$v_f = \mathbf{v}_c^T \mathbf{e}_f = \dot{x}\cos\theta + \dot{y}\sin\theta \tag{22}$$

and the lateral component :

$$v_l = \mathbf{v}_c^T \mathbf{e}_l = \dot{x}\sin\theta - \dot{y}\cos\theta \tag{23}$$

have to be considered as first. Notice that vectors has been reduced to two dimension i. e. $\mathbf{v}_c = [\dot{x}\ \dot{y}]^T$.

If there is no slip in lateral direction, they satisfy:

$$v_l = \mathbf{v}_c^T \mathbf{e}_l = \dot{x}\sin\theta - \dot{y}\cos\theta = 0 \quad .$$

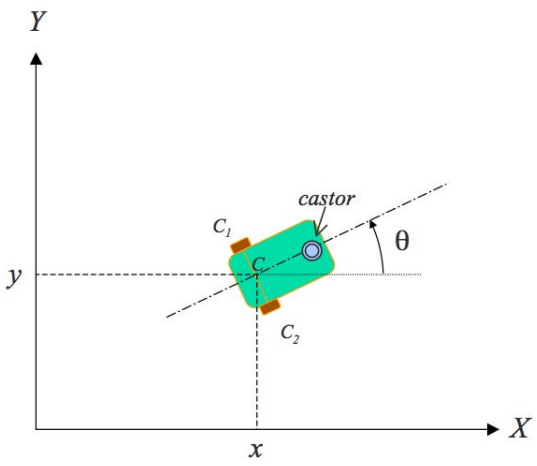


Figure 21: Differential drive robot composed of two fixed wheels plus a castor [SuperMARIO, MagellanPro]. The castor serve to support the weight of the vehicle and under ideal conditions do not affect the kinematics of the robot.

Referring to Figure 21, let us denote the centers of the wheels by C_1 and C_2 respectively and let their radius be r. Let the axis width (i. e. the length of the vector $\overrightarrow{C_1C_2}$) be l. Let (x_i, y_i, z_i) denote the position of center C_i and let ω_i denote the wheel speed of the i-th wheel. And let the component of velocity in the longitudinal direction, generated by the i-th wheel, be given by $v_{f,i}$. According to the definition of angular velocity, we have:

$$v_{f,i} = r\omega_i$$

while for the lateral speed:

$$v_{l,i} = 0 \quad .$$

From Equation 22 and Equation 23, we have:

$$v_{f,i} = \dot{x}_i \cos\theta + \dot{y}_i \sin\theta$$

$$0 = \dot{x}_i \sin\theta - \dot{y}_i \cos\theta \quad .$$

In order to derive the expression of the velocities of the center of mass of the robot, in function of the measured velocities of wheels, it is necessary to consider the coordinates of the point $C = (x, y)$ which is clearly half way between C_1 and C_2:

$$x = \frac{x_1 + x_2}{2} \qquad y = \frac{y_1 + y_2}{2}$$

thus, the velocity of point C, in the plane, is given by:

$$\mathbf{v}_C = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \frac{\dot{x}_1 + \dot{x}_2}{2} \\ \frac{\dot{y}_1 + \dot{y}_2}{2} \end{bmatrix} \quad .$$

The forward speed or the velocity component in the longitudinal direction can be obtained by projecting along $\mathbf{e}_l$:

$$\begin{aligned} v_f &= \mathbf{e}_f^T \mathbf{v}_C = \dot{x}\cos\theta + \dot{y}\sin\theta \\ &= \frac{\dot{x}_1 \cos\theta + \dot{y}_1 \sin\theta}{2} + \frac{\dot{x}_2 \cos\theta + \dot{y}_2 \sin\theta}{2} \\ &= \frac{v_{f,1} + v_{f,2}}{2} \\ &= \frac{r\omega_1 + r\omega_2}{2} \end{aligned} \tag{24}$$

Now if we consider the two points C_1 and C_2 which are rigidly attached to the axis of the mobile robot, the velocities of these two points are related by:

$$\mathbf{v}_{C_2} = \mathbf{v}_{C_1} + \dot{\theta}\hat{\mathbf{k}} \times \overrightarrow{C_1 C_2}$$

where

$$\mathbf{v}_{C_1} = v_{f,1}\mathbf{e}_f \qquad \mathbf{v}_{C_2} = v_{f,2}\mathbf{e}_f \tag{25}$$

Substituting l for $|\overrightarrow{C_1 C_2}|$ we get:

$$-r\omega_2 \mathbf{e}_f = -r\omega_1 \mathbf{e}_f + \dot{\theta}\hat{\mathbf{k}} \times l\mathbf{e}_l$$

and writing this, in the component along $\mathbf{e}$:

$$r\omega_2 = -r\omega_1 + l\dot{\theta} \qquad \Rightarrow \qquad \dot{\theta} = \frac{r\omega_1 - r\omega_2}{l} \quad . \tag{26}$$

Thus, Equation 24 and Equation 26 provide the relative velocity of the mobile robot center of mass in function of the velocity of the wheels.

KINEMATIC MODEL

In Figure 21, it is shown the schematics for the design of differential drive robot. The kinematics of the system is determined by the axle and the wheel radii.

In order to obtain a model which relates angular measured speeds with the Cartesian coordinates of the point representing the robot, Equation 24 and Equation 26 have to be translated via the usual transformation from Polar to Cartesian coordinate and we obtain:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r\cos\theta}{2} & \frac{r\cos\theta}{2} \\ \frac{r\sin\theta}{2} & \frac{r\sin\theta}{2} \\ r/l & -r/l \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} \tag{27}$$

3.5.3 *Quadrotor modeling*

The quadrotor is a Vertical Take-Off and Landing (VTOL) aircraft [33]. It consists of four propellers arranged on x-shape or +-shape. Every arm holds a propeller on its end that enables the system to fly. Several advantages distinguish the quadrotor from other aircraft, such as lower payload, simplicity of control and a great maneuvering attitude which can help in going into several areas that could not be accessed by traditional airplanes nor helicopters.

Figure 22: Quadrotor.

The symmetry of the quadrotor body gives simplicity to the controller design as it can be controlled through varying the speed of the propellers. Directional control is produced by individually altering the speed of the four motors Each two opposite propellers rotate in the same direction as

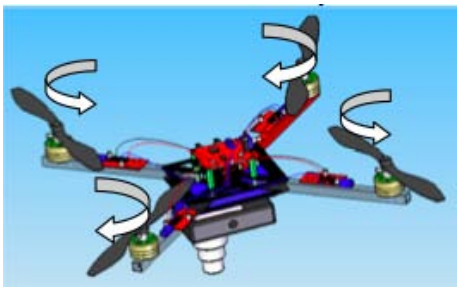


Figure 23: Quadrotor operation.

shown in Figure 23. A quadrotor consists of two fixed pitch clockwise

spinning propellers and two counter-clockwise spinning rotors which diagonally oppose each others. This results that the reactive force of each propeller being effectively canceled out by the diagonally opposite propeller's reactive component.

Quadrotor craft has further efficiency and mechanical advantages. Having small propellers reduces the torque on the system which means that the blades can be driven at higher velocities without producing additional mechanical vibrations and also increases motor efficiency.

The quadrotor aircraft is a highly non-linear, Multi-input Multi-output (MIMO) and strongly coupled system. In order to derive a suitable model for this system, it is necessary state some assumption to simplify the dynamics of the complex system. These assumptions are as follows:

- the quadrotor structure is rigid and symmetric;
- propellers are rigid;
- ground effect is neglected.

The Newtonian method is the most popular choice for modeling rigid bodies in six degrees of freedom and it enables to derive a consistent model for quadrotors.

Let consider Figure 24 in which is represented the force distribution on the quadrotor. Let f_i forces generated by propellers, ω_i the rotational

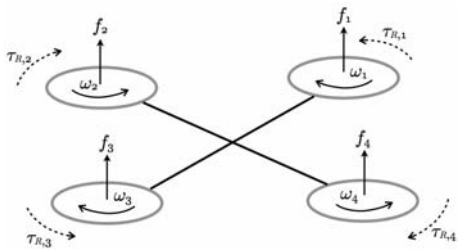


Figure 24: Force distribution.

speed of the propeller and $\tau_{R,i}$ the reaction torque of the system on the propeller. The physical laws that combines forces and torques with rotational speeds of propellers are:

$$f_i = b\omega_i^2$$

$$\tau_{R,i} = d\omega_i^2$$

where b is the thrust factor and d the drag factor (both depending on the rotor geometry and profile, its disk area and radius and on air density).

Let (x, y, z) be the position of the reference frame SR_B jointly liable to the quadrotor with respect to a universe reference frame O and (ϕ, θ, ψ) the angles of SR_B with respect to the universe reference frame as shown in Figure 25.

Let T be the resulting force:

$$T = f_1 + f_2 + f_3 + f_4 \quad ,$$

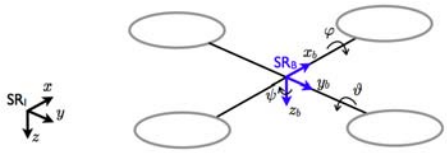


Figure 25: Universe and SR_B reference frame.

and

$$\tau_\phi = l(f_2 - f_4)$$

$$\tau_\theta = l(f_1 - f_3)$$

and τ_ψ the resulting torque on the $x-y$ plane:

$$\tau_\psi = -\tau_{R,1} + \tau_{R,2} - \tau_{R,3} + \tau_{R,4} \quad .$$

Therefore, the dynamics of the system is that of a rigid body with mass m subject to external forces applied to the center of mass according to Newton-Euler formula:

$$\sum F_i = \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} + f\left(\begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix}\right) + F_A + F_D$$

for the translational dynamics, where $f(\cdot)$ is a function of the actuation control, F_A is the aerodynamics force and F_D models disturbances. For the rotational dynamics:

$$\sum M_B = \begin{bmatrix} L_\phi \\ L_\theta \\ L_\psi \end{bmatrix} + \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} + \tau_A + \tau_D$$

where $\begin{bmatrix} L_\phi \\ L_\theta \\ L_\psi \end{bmatrix}$ models the gyroscopic effects, $\begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix}$ is the actuation control, τ_A are the aerodynamic effects and τ_D model disturbances.

A mathematical model of the system can be build. In particular, considering the (non-linear) state-space model representation:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u} \quad ,$$

$$\mathbf{x} = \begin{bmatrix} x & y & z & v_x & v_y & v_z & \phi & \theta & \psi & p & q & r \end{bmatrix}^T$$

$$\mathbf{u} = \begin{bmatrix} T & \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$$

the model:

$$\begin{cases} \dot{x} = v_x \\ \dot{y} = v_y \\ \dot{z} = v_z \\ \dot{v_x} = F_{A,x} - (\cos\psi \sin\theta \cos\phi + \sin\psi \sin\phi)\frac{T}{m} \\ \dot{v_y} = F_{A,y} - (\sin\psi \sin\theta \cos\phi + \sin\phi \cos\psi)\frac{T}{m} \\ \dot{v_z} = F_{A,x} + g - \cos\theta \cos\phi \frac{T}{m} \\ \dot{\phi} = p + \sin\phi \tan\theta q + \cos\phi \tan\theta r \\ \dot{\theta} = \cos\phi q - \sin\phi r \\ \dot{\psi} = \sin\phi \sec\theta q + \cos\phi \sec\theta r \\ \dot{p} = \tau_{A,x} + \frac{I_r}{I_x} q\Omega_r + \frac{I_y - I_z}{I_x} qr + \frac{\tau_\phi}{I_x} \\ \dot{q} = \tau_{A,y} + \frac{I_r}{I_y} p\Omega_r + \frac{I_z - I_x}{I_y} pr + \frac{\tau_\theta}{I_y} \\ \dot{r} = \tau_{A,z} + \frac{I_x - I_y}{I_z} pq + \frac{\tau_\psi}{I_z} \end{cases}$$

where have been used I_r for the blades inertia and Ω_r for the average blades rotation velocity.

3.5.4 *The Extended Kalman Filter*

In order to combine the data from odometry readings together with triangulation results, a kinematic model for the robot is necessary.

In the following, the model for a unicycle has been chosen for instance, but any other model for mobile robots (e. g. quadrotor) is suitable for the algorithm with no others complications, as in general all these models are non-linear. The model in state space representation for the process is:

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)) + \mathbf{w}(k) \\ \mathbf{y}(k) = \mathbf{h}(\mathbf{x}(k)) + \mathbf{v}(k) \end{cases} \tag{28}$$

where $\mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))$ is the nonlinear state transition function of the robot. In general $\mathbf{w}(k) \sim N(0, R_w)$ is a Gaussian noise with zero mean and covariance R_w independent from k. $\mathbf{h}(\mathbf{x}(k))$ is the nonlinear observation model and $\mathbf{v}(k)$ is a Gaussian noise with zero mean and covariance $R_v(k)$. It is important to notice that the model noise $\mathbf{w}(k)$ has covariance matrix independent from the current step k. In fact, while the model noise is in general a stationary random process, the measure noise $\mathbf{v}(k)$ has covariance function that depends on the position of the robot with respect to beacons position. For this reason, it is necessary to consider a covariance in function of the time step k.

Our approach consists in considering:

- odometry follows the kinematic model, that is the function $\mathbf{f}(\cdot)$ in (28);

- the position for the robot estimated through geometric triangulation provide measures $y(k)$.

Input vector to the model is composed of the two rotational velocities measured by encoders mounted on wheels, respectively the right and left one as motivated in Equation 27.

The state transition function of the robot is, according to Equation 27, for a unicycle:

$$\vec{f}(\mathbf{x}(k), \mathbf{u}(k)) = \begin{bmatrix} x(k) + v_f \cos\theta \\ y(k) + v_f \sin\theta \\ \theta(k) + \Delta\theta \end{bmatrix} \tag{29}$$

where the initial presented model has been discretized, i.e. v_f represents the mean speed of the robot in the time-step and $\Delta\theta$ represent the mean rotational speed in the time-step. The observational model is:

$$H = \begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & c_\theta \end{bmatrix}$$

i.e. it is a linear model, in which c_x, c_y and c_θ are parameters of the encoders installed on the robot.

Since the model (29) is nonlinear, the EKF must be used here to integrate the laser measurements and readings from optical encoders.

Note that the EKF is recursively implemented as follows:

STEP 1 - PREDICTION: predict the next position of the robot using odometry.

$$\hat{\mathbf{x}}(k+1|k) = f(\mathbf{x}(k), \mathbf{u}(k))$$

$$\mathbf{P}(k+1|k) = \nabla\mathbf{f}\, \mathbf{P}(k|k) \nabla\mathbf{f}^T + \mathbf{R}_w$$

where $\nabla\mathbf{f}$ indicates the Jacobean matrix of the transition function, and is obtained by linearization:

$$\nabla\mathbf{f} = \begin{bmatrix} 1 & 0 & -v_f(k)\sin\theta(k) \\ 0 & 1 & v_f(k)\cos\theta(k) \\ 0 & 0 & 1 \end{bmatrix}$$

STEP 2 - OBSERVATION: actual measurement are made.

$$y(k) = H\, \mathbf{x}(k)$$

STEP 3 - UPDATE: compare triangulation measurements with odometry prediction:

- Innovation:

$$\tilde{\mathbf{y}}_{k+1} = \mathbf{y}_k - H\hat{\mathbf{x}}(k+1|k)$$

- Innovation covariance:

$$\mathbf{S}_{k+1} = H\mathbf{P}(k+1|k)H^T + \mathbf{R}_\nu(k+1)$$

- Kalman gain:

$$K = \mathbf{P}(k+1|k)H\mathbf{S}_{k+1}^{-1}$$

- Updated state estimate:

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + K\tilde{\mathbf{y}}_{k+1}$$

- Updated estimate covariance:

$$\mathbf{P}(k+1|k+1) = (I - KH)\mathbf{P}(k+1|k)$$

The algorithm hides some very useful features: it produce the estimate of the current robot position at each cycle by integrating odometry data with only one angle measurement from the bearing sensor. Recursively, it combines every new measurement with measurements made in the past to estimate the robot position.

3.5.5 *Kalman Filtering with Intermittent Observations*

Previous equations clarify how triangulation results can be combined with odometry data in order to produce a robust position prediction and to improve localization performances. Nevertheless, it is easy to imagine that in some particular situations, triangulation estimate could not be available and, as consequence, observation $y(k)$ are not provided. These are, for instance:

- when less than three beacons are available to the robot;
- when occlusions occur;
- when data losses occur due to the unreliability of the network links;
- when beacons are not active due to energy constraints.

We can moreover decide to do not use triangulation results when their variance overcome a certain threshold as it could affect results. This leads to some troubles in the normal operation of the filter, since measures $y(\cdot)$ are not available.

In such cases, the robot could not exploit triangulation results and it needs to proceed using only odometry until anchor-based localization will be available again.

A similar problem has been approached in [40]: in particular Sinopoli et al. studied the asymptotic behavior of the filter in these situations. Although absence of observations, actually, corresponds to the limiting case in which the output noise has infinite variance, in the paper the approach is to derive Kalman filtering equations using a "dummy " observation with

a given variance when real observation does not arrive, and then take the limit as variance tents to infinity.

Defining the arrival of the observation at time k as a binary random variable γ_k, with probability distribution $p_{\gamma_k}(k) = \lambda$ and with γ_k independent of γ_s if $k \neq s$, the Kalman filter equations are modified as follows:

- the first step of prediction remains unchanged;
- the second step of observation is not performed as measures are not available;
- in the third step, the equations of the updated state estimate modify as follow:

  1. $$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \gamma_{k+1} K \tilde{\mathbf{y}}_{k+1}$$

  2. $$\mathbf{P}(k+1|k+1) = (I - \gamma_{k+1} KH)\mathbf{P}(k+1|k)$$

that is, the Kalman filter equation for intermittent observation corresponds exactly to propagating the previous state when the observation is not available at time k.

Thanks to the new formulation, derived from [40], the filtering phase $\hat{\mathbf{x}}(k+1|k+1)$ can be easily performed even when there is no observation update available at time $k+1$.

4 BEACON DEPLOYMENT FOR LOCALIZATION

Multiple are the advantages introduced thanks to the interplay between fixed and mobile agents: for instance, as introduced in previous section, static sensors can provide localization services, while at the same time robotic nodes deal with sensing the environment. Although many works in literature treat pre-configured networks of fixed nodes, because of pre-existing infrastructures or constraints due to physical issues, there exist some practical cases that envisage the network to automatically adapt to environments, a-priori unknown.
The *beacon deployment process* regards the problem of optimally deploying beacons, in order to provide mutual services to the robotic network.

Beacon displacement strongly affects quality of localization and many other services provided by the network. For instance, when considering the simple triangulation problem described in Chapter 3: the number of visible beacons and their placement geometry is crucial in order to reach reliable results. Each node in fact requires almost three beacons to be able to localize itself.

This chapter aims to study approaches to sensor node localization that uses sequentially deployed beacons in order to localize mobile agents.

4.1 OBJECTIVE AND MOTIVATION

In this section the main motivations to the work are introduced and the objective are described in detail. Moreover the key contributions are resumed. A detailed previous-work description concludes the section.

4.1.1 *Problem definition*

In this chapter the motivation, design, implementation and evaluation of a sequentially self-deployment system based on triangulation is presented. The main purposes of this work are threefold:

- to formalize the sensor placement problem;
- to address beacon deployment issues, following the idea of the novel concept of self-configuring beacon network;
- to propose a novel method capable of determining locations for new beacons to be deployed in an optimal way, by exploiting results presented in Chapter 3 regarding geometric triangulation variance derivation;

4.1.2 *Previous work*

Much of the research conduced thus far has mainly focused on localization techniques, while beacon placement issues have not been rather explored. Typical approaches taken in literature deal with pre-existing beacons displacement, thus focusing on optimal selection between available beacons.
An adaptive beacon placement algorithm has been proposed in [3], where beacons are deployed sequentially basing on empirically data of the perceived localization error. The perceived error is obtained by observations among neighboring beacons. When beacons are densely deployed over an area, the perceived localization error among them should reflect the error characteristics of the terrain or environment, which can then be applied to estimate the localization error of the actual sensor nodes.
Further work in [3] by Bulusu et al. provides a framework to realize the adaptive algorithm in the real world by proposing a distributed algorithm to disseminate the perceived localization error into a centralized location.
Isler describe a beacon deployment strategy with a different objective: minimizing the number of deployed beacons to localize mobile objects. The problem approached deals with finding the minimum number and the placement of cameras so that the error in localization is less then an error threshold at every point in the workspace.
In [41] the problem of controlling the configuration of a sensor team which employ triangulation for estimation is studied. Authors present a particle filter. The approach provide the optimal move at each time step by computing numerically an n-dimensional gradient. This work solves a local placement problem for a small number of robots at each time step.
In [32], the problem of relocating a sensor team whose members are restricted to lie on a circle and charged with jointly estimating the location of the target was studied. Efrat et al. in [14] study the problem of placing cameras in a polygonal workspace in such a way that for each point of interest, there are two cameras which can view the point at an acceptable angle. The work in [22] studies the problem of assigning disjoint pairs of sensors to targets.
In the work proposed by Tekdas and Isler, a solution framework based on integer linear programming is proposed. The work [22] proposes moreover an approximation algorithm with a constant factor performance guarantee, while the main focus addressed is the minimization of the number of sensors employed.

4.1.3 *Contributions*

The key contributions of this work consists in introducing optimal beacon deployment criteria and algorithms for geometric triangulation.
In the first part of this section, we propose a sequentially self-deployment system capable of optimally locate new beacons that provides localization services for robotic networks.
First of all, we introduce the problem by describing how a-priori designed

patterns can optimally cover the mission space by minimizing the number of beacons involved.
Subsequently, according to robustness and autonomy constraints, we proceed in developing mechanism that make the network capable of computing locations for new beacons to be deployed in an accurate manner.

Our method is based on cooperation between pre-existing beacons: thanks to local exchange of information, candidates location for new beacons are computed through local optimization. Computed positions are then transmitted to robotic agents for deployment.

4.1.4 *Problem formulation*

In this section, the sensor placement problem for triangulation-based localization is formulated.
Given a workspace Ω, which consists of all possible locations for the target, let s be a k-tuple representing related sensor parameters which can include, for example, location and orientation of agents. Let Q be the domain of s. Q represents the set of all possible placements of a single sensor in Ω.

Let us consider a function U:

$$U : Q \times Q \times Q \times \Omega \rightarrow \mathbb{R}$$

$U(s_i, s_j, s_k, \omega)$ returns the uncertainty in localizing a target positioned at ω ($\omega \in \Omega$) exploiting three sensors located and orientated as described by s_i, s_j and s_k respectively.
It is easy to deduce that the function U is specific to particular environment and sensing models. For example, $U(s_i, s_j, s_k, \omega)$ can be infinite if the environment causes an occlusion between ω and either s_i or s_j or s_k.

Let $S = \{s_1, \ldots, s_n\} \subseteq Q^n$ be a placement of sensors where the i^{th} sensor has parameters s_i. The quality of a placement in a workspace is determined as follows:

$$U(S, \Omega) = \max_{\omega \in \Omega} \min_{s_i, s_j, s_k \in S} U(s_i, s_j, s_k, \omega) \tag{30}$$

Equation 30 explains how, to establish the quality of services provided by a given beacons arrangement, the largest uncertain value over the entire workspace is taken.

The optimal sensor placement problem can be formulated as follow:

Definition 4.1. Given a workspace Ω, candidate sensor locations Q and an uncertain function U, find a placement S with minimum cardinality such that minimize U, i. e. :

$$\min_{S \subseteq Q^n} U(S, \Omega)$$

The definition clarify the twofold objectives:

- raising a placement that minimize the number of sensors employed;
- minimizing the uncertain in estimated position.

In addition, we underline that at low and medium beacon densities, the quality of localization suffers due to poor placement of beacons; while at high densities collisions in communications compromises the scalability of performances. Hereafter we present an example that clarify this seeming counter-sense.

4.1.5 *Sensor placement problem for triangulation-based localization is NP-Complete*

Unfortunately optimal solution to (30), i. e. optimal beacons displacement that both minimizes sensors employed and guarantee minimal localization uncertain is not trivial. It is furthermore possible to state that this kind of problems are NP-Complete[1].

This fact is usually demonstrated in literature [43] by establishing its relation to the well-known k-center problem, which is NP-Complete.

In the k-center problem, we are given a set of locations for centers and a set of targets along with a distance function $d(i, j)$ between the centers and the targets. The objective is to minimize the maximum distance between any target and the center closest to it. The converse problem, where the maximum distance from each vertex to its center is given and the number of centers is to be minimized, is also NP-Complete. Hence, the sensor placement problem is, at least, as hard as the converse k-center problem.

4.2 INTRODUCTION ON ADAPTIVE BEACON PLACEMENT

Intuitively, approaches such as uniform and very dense beacon placement are suitable for many practical approaches. However there exist some disadvantages that compromise their effectiveness.
First of all, uniform placement results as a good strategy in many cases in which we do not dispose of any detailed information about the work to accomplish or the environment to study. However, this model is insufficient due to the following reasons:

- Beacons may be perturbed during deployment. Consider for example, a terrain comprising hilltops. Air dropped beacon will roll over the hill compromising the overall network characteristics.
- Even when beacon placement is uniform, noise may affect the visibility of beacons that should be in range: in fact, uneven terrains and obstacles bring additional uncertainty.

Very dense placement may not be a viable solution too. This because of several reasons:

- Signals interference.

1 A decision problem is NP-Complete when it is both in NP and NP-Hard. The most notable characteristic of NP-Complete problems is that no fast solution to them is known. That is, the time required to solve the problem using any currently known algorithm increases very quickly as the size of the problem grows.

- Cost and power: cost of the beacons may preclude very dense placement. Power considerations may require that only a restricted subsets of the beacon network can be active at any given time.
- The environmental or terrain conditions may be such that even increasing the density uniformly will not overcome the problem. For instance, if the number of dropped beacons were doubled, the same situation would persist. Also, the terrain can already have a very high density of beacons and hence the new beacons deployed must be added in particular places to cope with noise.
- At very high densities, the phenomenon of self-interference emerges. In fact, the probability of collisions among signals transmitted by the beacons increases.

The fundamental limitation of the two proposed approaches is that they are basically fixed strategies, that do not take into account environmental conditions that cannot be known a-priori. It is practically impossible to pre-configure to such terrain and propagation uncertainties and compute an ideal beacon placement that uniformly achieves a desired quality of localization across the region.
Beacon placement needs to adapt to the noisy and unpredictable environmental conditions. Thus, recent studies addresses the problem of *Adaptive beacon placement*. The problem deals with understanding how should additional beacons be placed, given an existing field of beacons, in order to reach best localization performances.
Bulusu et al. in [3] has formalized two simple algorithms for adaptive beacon placement. The procedures differ in the amount of global knowledge and processing used to take decisions.

4.2.0.1 *Random*

The simplest algorithm for beacon deployment provide nodes to do not pay attention to the quality of localization at different areas of the mission space and simply selects a random point in the region as a candidate for adding a new beacon. The algorithm is structured as proposed in Algorithm 2.

Algorithm 2 Random beacon deployment algorithm

1) Select a random point (x^*, y^*)
2) Add a new beacon at (x^*, y^*)

The presented model is extremely simple, however it is usually taken into account as it is similar in character to uncontrolled airdrop of additional nodes. Its complexity is $O(1)$.

4.2.0.2 *Max*

The *Max* algorithm is described in Algorithm 3

Algorithm 3 Max algorithm

Let the terrain be a a square of s meters and each robot take measures in ρ meters.
1) Divide the terrain into $\rho \times \rho$ squares.
2) Measure localization error at each point in the terrain that corresponds to a square.
3) Add a new beacon at the point (x^*, y^*) that has the highest measured localization error among all points.

This algorithm is based on the assumption that points with high localization error are spatially correlated. The advantage of this algorithm is that it can be computed in a very straightforward way; however it may be influenced by random noise or propagation effects.
The complexity of the Max algorithm is linear in the number of square of the discretization.

Algorithm proposed beforehand, have been reported in order to provide an overview on fixed strategies. For a comprehensive discussion about uniform and very dense placements, please refer to [3].

4.3 ADAPTIVE BEACON DEPLOYMENT BASED ON TRIANGULATION VARIANCE MAP

It is possible to state that many of the techniques proposed in literature for landmarks positioning are specific of the localization method exploited, Moreover, as previously demonstrated, the problem of optimal beacon deployment for triangulation is NP-Complete.
These reasons have directed researchers to develop models that make use of pre-existing beacon networks, and directing their attentions on selecting the more appropriate tern of beacon to perform the triangulation, when mobile agents are capable of sensing more than three beacons.
This approach is based on a strong assumption: the beacon network is located in an off-line manner, and there is redundancy in beacons deployment. However we can deduce that this perspective is not suitable for such cases in which we cannot dispose a priori of information regarding the mission space. In such cases mobile agents are required to provide an adapting structure the the sensor network, through deploying beacons basing on sensed environment.

This chapter presents a novel approach for beacon deployment based on triangulation variance estimation described in previous Chapter 3.
In particular the uncertain about self actual location of new deployed beacons will provide a suitable and robust criteria for the iterative design of their locations.

In the following, two main assumption are done:

- triangulation is used as localization method. As previously introduced, the beacons deployment problem is specific for the localization methods used, and this assumption specify the framework.

- The main purpose is to localize every target located in the internal area formed by the three localization beacons.

The second assumption is motivated by considerations described in Chapter 6 and by the observation that triangulation algorithms perform better when the target is located in the internal polygon determined by the three beacons as vertexes.

Finally it is important to underline that the method we propose, although in this work is described as specific for geometric triangulation, could be extended to other methods capable of describing uncertain through numerical variance maps.

4.3.1 *High-uncertain areas of triangulation*

As motivated in Chapter 3, at least three beacons have to be available for the robot in order to to localize itself in a plane. All areas of a plane with less than three visible beacons are unsuitable for robot localization based on anchor nodes.

On the other and, when at least three beacons are available, the mobile node is capable of self-localizing in the entire plane except in two singular cases:

- when the robot and the beacons all lie in the same circumference;
- when the robot see an occlusion or a couple of beacons are collinear.

Targets cannot be localized when they are located on the circumference joining the three beacons because the intersection between the three circles of triangulation is an arc, not a single point.
Furthermore, triangulation cannot be performed when a couple of beacon is collinear, as two (of the three) circles of the algorithm coincides. Figure 26 explains graphically areas where the triangulation does not work.

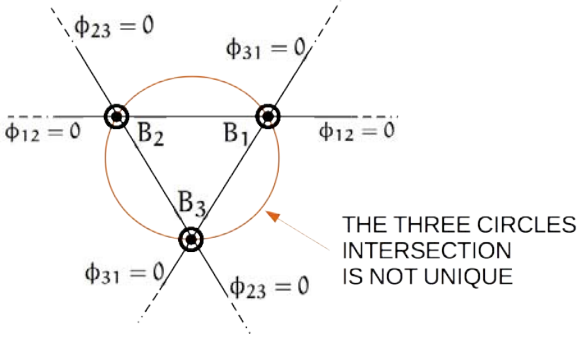


Figure 26: Regions of the plane where triangulation algorithms cannot localize targets.

Although ToTal algorithm proposed in previous section introduces several advantages with respect to previous proposed, among which the extension of geometric triangulation to entire mission space, it is important to underline that there still exist some confined areas of the plane where

triangulation is not capable of computing the position of the target. As motivated above, this region is composed by the three lines and the circle joining the beacons.
This singular behavior has to be taken into account during the infrastructure design phase: in particular beacons should be deployed in patterns that minimize this areas.

4.3.2 *Self-configuring minimal uncertain deployment*

When beacons for localization have to be deployed, an important parameter to take into account is the maximum distance at which a couple of them can be placed. This threshold arises as a consequence of signals attenuation and degradation as distances increase. Let d_{MAX} be this maximum distance. d_{MAX} represents, for example, the maximum distance at which bearing sensors mounted on mobile agents can detect beacons in a reliable manner.

Beacon deployment density ρ, describes the number of beacons per unit area:

$$\rho = \frac{N}{S} \quad ,$$

where S denotes the surface area of interest and N is the number of beacon located in the same area and respecting the d_{MAX} constraint. This index characterizes the fundamental aspect of understanding how dense are the nodes in a specific region, and helps in understanding whether new beacons are needed to improve triangulation.

When a new beacon have to be deployed, both the beacon deployment density ρ and enhancements in localization performances have to be taken into account. In fact, one of the main objectives to meet when a new beacon has to be deployed is to maximize the portion of mission space covered; while, on the other hand, new beacon deployed have to guarantee high precision in localization.
It is easy to deduce that the two purposes are conflicting. In fact, on one hand a low-density deployment guarantee the maximum surface covered, while on the other hand to reach optimal performances in terms of quality of localization, nodes have to deployed in a high-density pattern.

Example 4.3.1. In order to deploy beacons through maximizing the portion of mission space covered, the better pattern to employ is those represented in Figure 27.

In fact,

- every couple of nodes (B_i, B_j) satisfy $dist(B_i, B_j) = d_{MAX}$;
- every node in the tern is equally faraway from all its neighbors.

Therefore, the equilateral triangle-shape represents the optimal pattern for beacons disposal when the aim is to minimize beacon deployment density ρ.

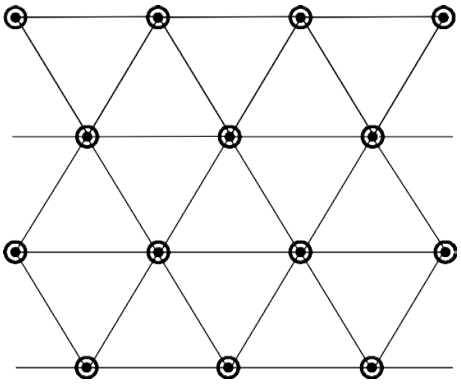


Figure 27: Equilateral triangle minimize the beacon deployment density.

The purpose of this chapter is to propose a novel method for localization beacon deployment, capable of meeting the following goals:

1. locations for new beacons have to be computed in a distributed manner, i. e. through exploitation of information gathered through local data exchange only;

2. the localization system has to self-deploy itself without any a priori knowledge. This mean that the network have to be capable of both computing candidates location and deploying sensors through information gathered during explorations only, since no a priori information is available;

3. new beacons located are required to compute their actual position by exploiting the pre-existing infrastructure;

4. basing on considerations presented Section 4.1.4, beacons have to be deployed is such a manner to minimize localization uncertain over the entire configurations space.

Requiring to novel beacons the ability of self-locating through the exploitation of the pre-existing infrastructure, lead strong robustness capabilities to the system.
Indeed, thanks to this feature, the system is capable of working in complete autonomy without the requirement of more restrictive assumptions, such as the support of GPS systems.
Therefore, since new-deployed beacons have to be capable of computing their actual position with lower uncertain, as a consequence of 3.4.2 they cannot be located in high-uncertain regions. That is, areas described in Section 4.3.1.

Example 4.3.2. Figure 28 shows (black, magenta and white dot patterns) three beacons composing a triplet for localization. This is referred as the

pre-existing localization infrastructure. The red dot represent the candidate position for the new beacon in order to build an equilateral-triangle shape. The equilateral triangle show, in its internal, the variance map computed by the tern of pre-existing beacons.

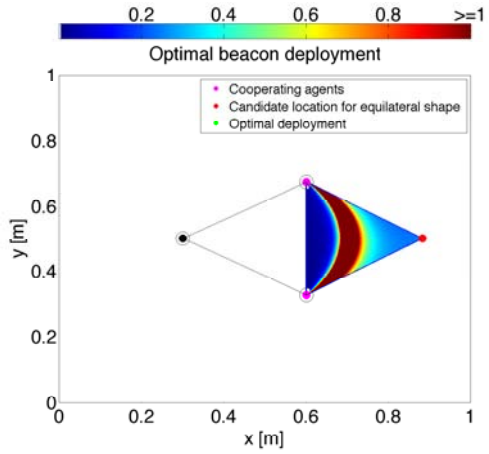


Figure 28: The problem of locating a new beacon can be stated as a constrained optimization issue.

Therefore, in order to meet the overall objective, that is both minimizing beacons density ρ and minimizing beacons location uncertain, a trade-off between the minimal-density and the minimal uncertain patterns have to be chosen.
The problem of locating a new beacon can then be formulated as a constrained optimization of the variance map in areas of the plane that satisfy the d_{MAX} constraint.

The novel method we propose is based on local cooperation between neighbor nodes, since each node requires only absolute positions of immediate neighbor.
Each node can compute candidate positions for a new beacons according to following steps:

1. Exploiting locations of two neighbors and its relative uncertain, compute the variance map $V(x, y)$ for the triplet of beacons;

2. Compute area of acceptable locations for the new node $\overline{\Omega}$, that is, location both respecting the d_{MAX} constraint and avoiding beforehand taken location. Notice that $\overline{\Omega} \subseteq \Omega$;

3. Compute ultimate candidate locations for new nodes solving the following constrained optimization problem:

$$\min_{\overline{\Omega}} V(x, y) \quad .$$

According to motivations just explained, in our algorithm we can identify two main phases that lead to the design of the ultimate location for new beacons:

1. design a candidate point according to density criteria;
2. adjust candidate location, reaching a trade-off, in order to obtain higher accuracy.

Figure 29 explains the deployment process performed by a tern of pre-existing nodes.

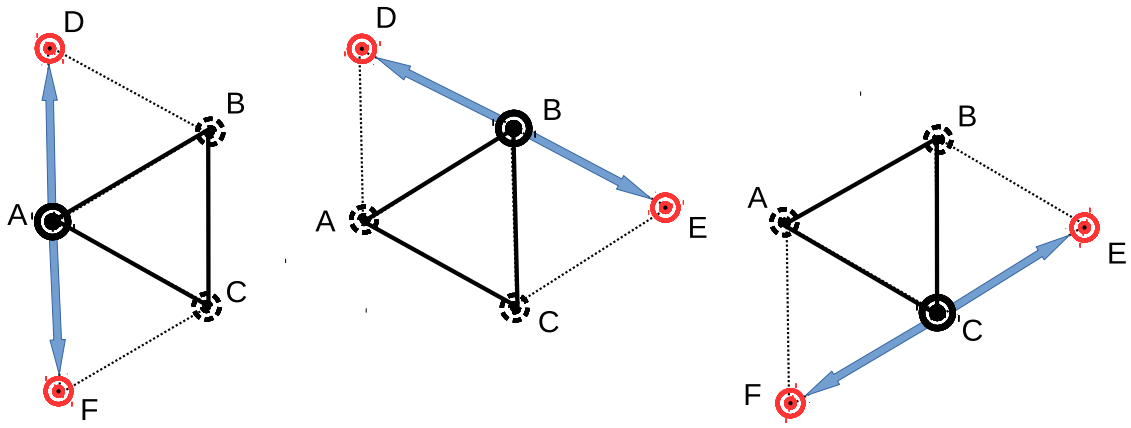


Figure 29: Each beacon is responsible for the deployment of a couple of new nodes.

The figure emphasize the fact that each beacon is responsible for the deployment of a couple of new nodes. Therefore each tern of pre-existing nodes can deploy three new beacons.

Another important aspect to underline, referring for instance to deployment of node D, is that the design of its position is performed by both node A and node B. In absence of noise, designed locations coincides. However, in practical applications, this process can affected by some unattended aspects, among which:

- nodes belief about self location can be altered by noise;
- triangles formed by beacons can present shapes different form equilateral;
- communications can be ruined by noise;
- obstacles or errors during the deployment phase can alter the actual position from the desired one.

This problem can be overcome through the introduction of a procedure based on average consensus, operated by each node together immediate neighbor, in order to design location for beacons in common.
Figure 30 schematizes the interaction, that produces ultimate location for the beacon.

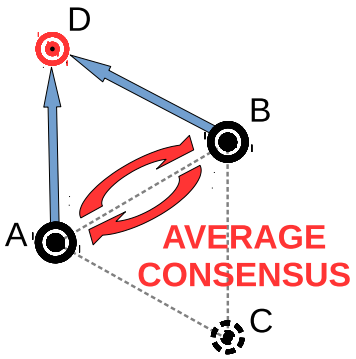


Figure 30: Node A and B design ultimate location for new node D through average consensus.

The final version for the algorithm we propose is reported in 4.
The procedure of new beacons deployment concludes with the transmis-

Algorithm 4 Iterative deployment algorithm

1) Send request to neighbors for their positions;
2) Compute acceptable positions $\overline{\Omega}$ respecting the d_{MAX} constraint;
3) Compute the variance map;
4) Solve the constrained optimization $\min_{\overline{\Omega}} V(x, y)$;
5) Compute ultimate candidate position through average consensus with immediate neighbor;

sion of the computed candidate location to robot network, that operates for the effective deployment of the sensor.

Once the new beacon has been deployed, it is requested to compute its actual position through geometric triangulation (Chapter 3) and the exploitation of three of its neighbors beacons.

The procedure is then repeated for all the beacons in the network, leading to a self-configuring deployment that extends to the whole mission space.

Figure 31 shows the optimal beacon deployment process performed by the infrastructure presented in Example 4.3.2.

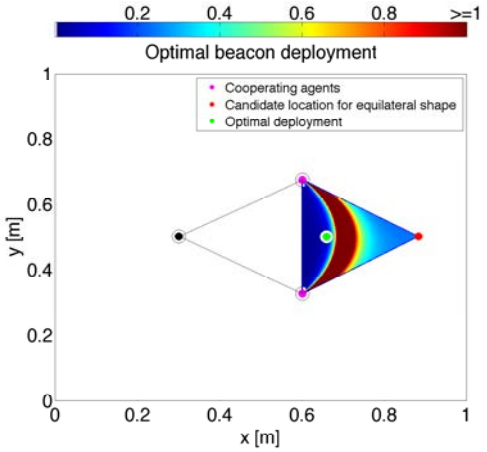


Figure 31: A couple of cooperating agents design a candidate location for a new beacon.

In particular, the cooperating agents are represented in magenta and black dot pattern. These are responsible for the computation of the subset $overline\Omega$, for the derivation of the Variance map and then for the design of the candidate point represented in green and white dot patterns.
The proposed procedure terminates with the deployment operated by the robot network.

5 COVERAGE CONTROL

In a sensor network, sensing devices can coordinate their actions through wireless communication and aim at performing tasks in a shared manner. Indeed, capabilities of sensor networks are strictly linked to the cooperation between their nodes, in opposition with the approach used for centralized controlled systems.
The performance of a sensor network, in terms of its specific tasks, is sensitive to the location of its nodes in the mission space. This leads to the basic problem of deploying sensors in order to meet the overall systems objectives, which is referred to as the *coverage control* or *active sensing* problem [5] [9].

In particular, sensors must be deployed so as to maximize the information extracted from the configurations space while maintaining acceptable levels of communication and energy consumption. As the number of sensors deployed in the target area increase, then a more complex and effective coverage method must be devised.

The static version of the solution to this problem involves positioning sensors without any further mobility; optimal location can be determined by an off-line scheme which is akin to the widely studied facility location problem.
The dynamic version allows the coordinated movement of sensors, which may adapt to changing conditions in the mission space, typically deploying them into geographical areas with the highest information density.

Advantages characterizing the two methods can be combined via mixed sensor and robot networks, i. e. a network composed of both mobile and fixed agents. In fact services provided by the two types of cooperating systems are often complementary. This allow the cooperation to provide noticeable results in many fields of employment.

5.1 AN OVERVIEW ON COVERAGE CONTROL

Thanks to mobility capabilities provided by robotic agents, the dynamic coverage control problem is in general managed by robots. However, it is important to underline that dynamic coverage control emerges as a result of the cooperation between robot and sensor networks. In fact, while the actual dynamic coverage control is performed by robots, fixed sensors still provide anchor-based localization services. Moreover fixed beacons, where needed, can be employed in order to improve coverage performances. This leads to a mixed dynamic and static version for coverage control.

When we deal with unknown mission spaces, i. e. when we do not dispose of any information a priori, the initial requirement for agents is to operate in order to provide an initial model for the mission space. To achieve

this goal, each robot needs to compute a path which allows itself to sense the most dynamic areas as first, then to perform a comprehensive estimation for the remaining part. Such a path is referred to as an *informative path*.

The key insight is to use a parameter function representing the rate of change of each point in the environment, while moving along a set of waypoints to optimize a cost function related to the informativeness of the path. The cost function attempts to place the path way-points at the centroids of their Voronoi cells, while also making sure the waypoints are not too far from one other. The robot moves along the path, marking the areas it observes as dynamic or static, and learning the environment model.

The collection of the desired information is, in many cases, constrained by the cost associated with obtaining it [26]. The cost of taking a measurement can be formulated in various manners. For many networks the most significant may be the energy wasting for reaching the interested area of the mission space.

The objective is to determine an optimal control strategy to take the set of measurement, while simultaneously trying to minimize the cost of taking those measurement, and this task must be fulfilled in a distributed way, that is, determining the optimal subdivision of explorations between the team of robots.

5.1.1 *Problem definition*

The goal of guaranteeing full awareness coverage of the mission space emerges in many practical applications, where some regions are of much more importance with respect to the other regions in the mission domain. Consequently, one needs to discriminate these two classes of regions. For example, in an application where a mobile sensor network is employed to survey a nuclear power plant persistently, nuclear reactors must be covered with full awareness while the the regions of less importance only need to be revisited frequently. Coverage control can generally be classified into:

- Static coverage control;
- Dynamic coverage control.

In static coverage control, the goal is to optimize the locations of fixed sensors in order to optimize the service provided by the entire network [10] [29]. Usually the static coverage problem can be referred as finding the minimum number of circles (circles represents the sensing area of agents) of radius R so that to completely cover the mission domain Ω. Usually this types of problems can be solved offline in a static manner.

When the mission space cannot be fully covered by any static configuration of the network, the problem of dynamic coverage arises; in this, each point in the configuration domain is sampled by the mobile sensors until

a prescribed coverage level is achieved [47] [51]. The movement of sensors not only impacts sensing performance, but it also influences other quality-of-service aspects in a sensor network, especially those related to wireless communication.
Because of limited on-board power and computational capacity of agents, a sensor network is not only required to sense but also to collect and transmit data as well in a reliable and optimal way. Particular attention has to be given to the path that agents plans to get from a location to another, as energy consumption constraints cold be limiting in many practical cases

For this reason, both sensing quality, communication performances and energy-wasting need to be jointly considered when controlling the deployment of sensors.

5.1.2 *Previous work*

The problem of optimizing sensor locations and sensor domains in fixed sensor networks has been extensively studied in the past: it can be considered as an optimization problem [38], the solution is a Voronoi partitioning [12], where the optimal sensor domain is a Voronoi cell and the optimal sensor location is a centroid of a Voronoi cell.

On the other hand, mobile sensor coverage is relatively new in literature and in [6] a survey of the most recent activities in control and design of robot static and dynamic sensor is presented. In [29], Li and Cassandras consider a probabilistic network model and a density function to represent the frequency of random events taking place. The authors develop an optimization problem that aims at maximizing coverage using sensor with limited ranges, while minimizing communication cost. Starting with initial sensor positions, authors developed a gradient-based algorithm to coverage to a (local) solution to the optimization problem.

A similar goal is the focus in [52], where the notion of virtual forces is used to enhance the coverage of a sensor network given an arbitrary initial deployment.

In [9] Cortes et al. address the same question, but in this case the trajectory converges to the centroid of a cell in a Voronoi partition of the search domain.

In [5] a coverage control scheme is developed which aims at the maximization of target exposure in some surveillance applications, while in [52] a heuristic algorithm is applied to enhance the coverage of a sensor network. Much of the active sensing literature [37] also concentrates on the problem of tracking specific targets using mobile sensors and the Kalman filter is extensively used to process observations and generate estimates.

5.1.3 *Contribution*

In this chapter, a distributed dynamic coverage control approach for cooperative sensing devices is developed and presented. Motivated by Li and

Cassandras in [29], the mission space has been modeled using a density function representing the frequency that events take place.

We assume that a mobile sensor has a limited range which is defined by a probabilistic model. A deployment algorithm is applied at each mobile node and it maximizes the joint detection probabilities of random events. We assume that the event density function is fixed and given. The extension to density function time-variant does not require any modification to the described approach as the adaptive relocation behavior naturally follows from the optimal coverage formulation.

In the work [29] presented by Li and Cassandras a novel method based on an optimization that aims to converge to a local solution to the optimization problem is considered. The model proposed has been extended in this work, in order to guarantee the obtainment of global optimal solutions.

The main contribution of this chapter is the development of a solution for the dynamic coverage control problem, through the maximization of a cost functional. In particular the problem have been formulated by exploiting optimal control theory. Moreover the proposed solution results suitable both for unknown environments and for common coverage control problems, where the mission space is a priori known. The method appears particularly interesting in order to perform an initial estimate of the state of the mission space.

Our approach is based on a distributed algorithm where each robot autonomously computes a Voronoi partitioning for its relative portion of mission space. Each Voronoi cell is then exploited for estimating information enclosed in each partition. This type of partitioning helps in both determining not yet visited areas and in defining their informativeness. The control scheme for mobile nodes, is then based on an optimization strategy, which aims to maximize information gathered.

5.1.4 *Optimal coverage formulation*

Let $\Omega \subseteq \mathbb{R}^2$ models the mission space, over which there exists an event density function

$$R(x), \quad x \in \Omega \tag{31}$$

that models the frequency or density that a specific random event takes place. Being a density function, $R(x)$ satisfies:

- $R(x) \geqslant 0 \quad \forall x \in \Omega$;
- $\int_\Omega R(x) = 1$.

$R(x)$ can, for instance, model the frequency that a certain type of event appears at x, or it could be the probability that a certain measured value exceeds a specific threshold at x.

In the mission space Ω, there are N mobile sensors located at $s = \{s_1, \ldots, s_N\}$, $s_i \in \mathbb{R}^2$ $i = 1, \ldots, N$.

When an event occurs at point x, it emits a signal and this is observed by a sensor at location s_i. The received signal strength generally decays with $\|x - s_i\|$, i. e. the distance between the source and the sensor. This degradation can be modeled by a monotonically decreasing diffentiable function $p_i(x)$, which expresses the probability that sensor i detects the event occurring at x.

For instance, a sensor model with a simply form may be:

$$p_i(x) = p_{0i} e^{-\lambda_i \|x - s_i\|}$$

that expresses the idea that the detection probability declines exponentially with distance.
p_{0i} and λ_i are determined by physical characteristics of the sensor. The optimal coverage problem can be formulated as maximizing the probability that events are detected. When an event takes place at x, and it is observed by the sensors, the joint probability that this event is detected can be expressed by:

$$P(x, s) = 1 - \prod_{i=1}^{N} [1 - p_i(x)] \tag{32}$$

where we have assumed that sensors take observations independently.

Definition 5.1. The optimal coverage problem can be formulated as an optimization problem that maximizes the expected event detection frequency by the sensors, over the entire mission space Ω:

$$\max_s \int_\Omega R(x) P(x, s) dx \quad := \max F(s) \tag{33}$$

In (33) controllable variables are the locations of mobile sensors s. The result of optimization (33), computed with respect to s_i, gives new positions for mobile robots s_i, $i = 1, \ldots, N$ that maximize the number of detected events.

In general, this kind of problem can be solved by applying a non-linear optimizer with an algorithm which can evaluate integrals numerically: in such a case, a centralized controller with intensive computational capacity is required.

5.2 DISTRIBUTED SOLUTION TO THE OPTIMAL COVERAGE PROBLEM

While in general terms, the solution to the optimal coverage problem is provided through centralized systems due to both the requirement of global knowledge of robots positions and the computational complexity required, this section proposes a distributed method to solve the optimal coverage problem.

Taking partial derivatives with respect to s_i, $i = 1, \ldots, N$, in order to solve Equation 33, we have:

$$\frac{\partial F(s)}{\partial s_i} = \int_\Omega R(x) \frac{\partial P(x, s)}{\partial s_i} dx \quad . \tag{34}$$

Using Equation 32, the partial derivative (34) can be rewritten as:

$$\frac{\partial F(s)}{\partial s_i} = \int_{\Omega} R(x) \prod_{k=1, k\neq i}^{N} [1 - p_k(x)] \frac{dp_i(x)}{ds_i(x)} dx \quad . \tag{35}$$

In general, (35) cannot be computed directly by mobile sensors, since it requires global information such as the value of $R(x)$ over the whole mission space and the exact location of all other sensors in the network. In addition, the evaluation of integrals remains a significant task for cheapest sensors.

If partial derivative might be evaluated locally by each mobile sensor, then a gradient method could be applied which directs agents towards locations that maximize $F(s)$.

In order to render distributed the optimal coverage problem, mobile nodes must be capable of computing the optimization (33) with the availability of local knowledge only. This mean that only local exchange of information between neighbor nodes, is needed.

Basing on the physical observation that when $d_i(x) >> 1$, $p_i(x) \simeq 0$, we assume:

$$p_i(x) = 0, \quad \forall x \quad \text{s.t.} \quad \|x - s_i\| > D \tag{36}$$

where D denotes the sensing radius. Thus $\|x - s_i\|$ defines the region of coverage of sensor i. Since $p_i(x) = 0$ for all $x \notin \Omega_i$, Ω can be replaced by Ω_i in Equation 35, where $\Omega_i = \{x : \|x - s_i\| < D\}$.
Moreover, indexes taken by variable k in the product can be limited in such a way to consider only nodes in the neighboring B_i, i. e. nodes that have their detectable area in common with the detect area of node i: $B_i = \{k : \|s_i - s_k\| < 2D, k = 1, \ldots, N, k \neq i\}$.
Thus, Equation 35 can be rewritten as:

$$\frac{\partial F(s)}{\partial s_i} = \int_{\Omega_i} R(x) \prod_{k\in B_i} [1 - p_k(x)] \frac{dp_i(x)}{ds_i(x)} dx \quad . \tag{37}$$

In conclusion we have obtained a useful and simply way to solve in a distributed manner the problem of the optimal coverage by introducing a simply simplification that takes into account the reduced sensing capacities of real sensors.

As a consequence of the local knowledge only required to solve the maximization, in general Equation 37 provides solutions for local maximum for the function $R(x)$.

In order to reduce computation costs in making Equation 37, the integral evaluation can be discretized [29]. This type of solution is often taken into account in many practical cases and could be evaluated in future works.

Example 5.2.1. Figure 32 shows simulation results about a network composed of a couple of mobile agents exploiting a control strategy according to Equation 37. Events density $R(x)$ has been modeled with three peaks located in $(0.4, 0.5)$, $(0.8, 0.2)$ and $(0.8, 0.8)$.

As displayed by the figure, the control strategy leads agent to move along the gradient direction until a local minimum is reached. Moreover it is important to underline that Robot1, according to control law, described by Equation 33, when enters in the sensing area of Robot2, modify its local view using its neighbors location. This modify its trajectory, that instead of moving along the gradient direction get the node to leave in a different direction.

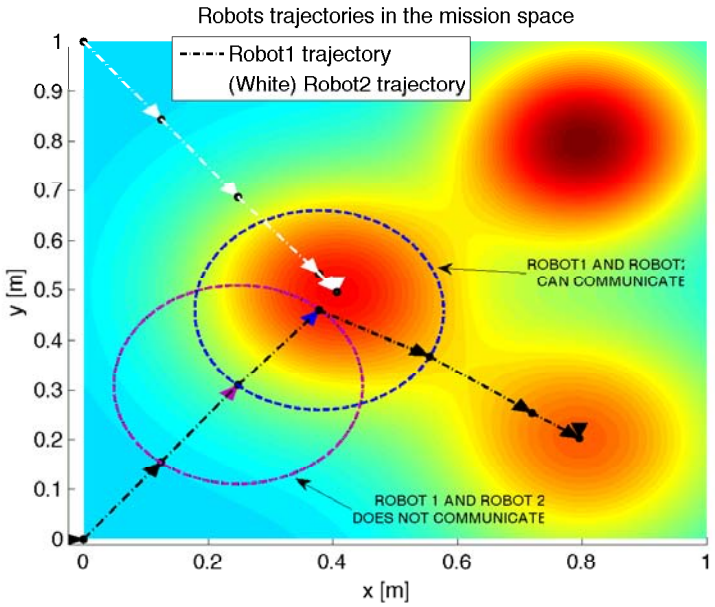


Figure 32: Area-discovering process by a team composed of a couple of robots.

5.3 CONFIGURATIONS SPACE PARTITIONING

Voronoi partitioning is one of the most attractive techniques that has been used to solve the problem of partitioning environments by a set of robot, so that each robot performs its operations in the region within the corresponding cell.

For a successful distributed implementation of the Voronoi partitioning problem, each robot should be able to compute the corresponding Voronoi cell autonomously. Usually, to calculate the Voronoi cell, each robot first uses the positions of all other robots in the environment to compute the entire Voronoi partition, and then, extracts its Voronoi cell. Computing the entire Voronoi partition is not very efficient as each robot unnecessarily calculates Voronoi cell for every other robot. In most practical situations, robots may not have information about the positions of all other robots in the environment, e. g. , when subsets of robots are outside each others' communication range. In such a situation, it is not possible to compute the exact Voronoi calls in a distributed manner. Therefore, it makes sens to study the problem of distributed Voronoi partitioning in a sensor range constrained scenario.

5.3.1 *Problem formulation*

Considering a team of robotic agents, the problem of partitioning the configurations space consists in dividing it into a number of regions, each of them could then be assigned to single agents. The approach exploited could be *static* or *dynamic*.
When a static approach is used, the partitioning is static through time and both sizes and locations of the partitions are in general fixed. This perspective is easier to implement as it can be studied during the setup phase of the network.
On the other hand, when a dynamic approach is exploited, partitions are periodically re-updated in terms of their sizes and locations, following the updates recorded by measuring and needs of the entire network.

Assuming to dispose of an initial sets of points, representing the sets of "centers" of the partitions involved we can proceed as follow. Let Ω be an open convex set representing the mission space, $\Omega \in \mathbb{R}^2$, and let $\overline{\Omega}$ the internal set of Ω; we can define:

Definition 5.2. The set $\{V_i\}_{i=1}^N$ is called a *tessellation* of Ω if:

- $V_i \cap V_j = \emptyset$ for $i \neq j$;
- $\cup_{i=1}^N V_i = \overline{\Omega}$

Definition 5.3. Given a set of points $\{\mathbf{z}_i\}_{i}^N$, belonging to $\overline{\Omega}$, the *Voronoi* region V_i corresponding to the point $\mathbf{z}_i$ is defined by:

$$V_i = \{\mathbf{x} \in \Omega \ : \ \|\mathbf{x} - \mathbf{z}_i\| < \|\mathbf{x} - \mathbf{z}_j\| \text{ for } j = 1, \ldots, N, j \neq i\} \tag{38}$$

where $\|\cdot\|$ denotes the Euclidean norm on $\mathbb{R}^2$

The points $\{\mathbf{z}_i\}_{i=1}^N$ are called *generators*.
The set $\{V_i\}_{i=1}^N$ is a *Voronoi tessellation* or *Voronoi diagram* of Ω, and each V_i is referred to as the *Voronoi region* corresponding to $\mathbf{z}_i$.

Given a region $V_i \in \mathbb{R}^2$ and a density function ρ defined in V_i, the *mass centroids* $\mathbf{z}*$ of V_i is defined by:

$$\mathbf{z}* = \frac{\int_{V_i} x\rho(x)dx}{\int_{V_i} \rho(x)dx} \tag{39}$$

Let $\mathbf{f}$ be an increasing function of distances between elements of Ω $f : \mathbb{R}_+ \to \mathbb{R}_+$ and let $\boldsymbol{\phi}$ be a weight function $\phi : \Omega \to \mathbb{R}_+$ that gives a distinct priority to various areas of the mission space. Function $f(\cdot)$, could for example represent the energy consumption of the mobile node in moving in the mission space, while the function $\phi(\cdot)$ can be chosen proportional to the objective function, in such case a greater number of sensors would be deployed in regions of major interests.

The optimal partitioning problem, that minimizes distances between actual locations of every nodes and considering the weight function Φ can be solved by minimizing the function:

$$H := \sum_{i=1}^{N} \int_{V_i} f(\|\mathbf{s}_i - \mathbf{x}\|)\boldsymbol{\phi}(s)d\mathbf{s} \tag{40}$$

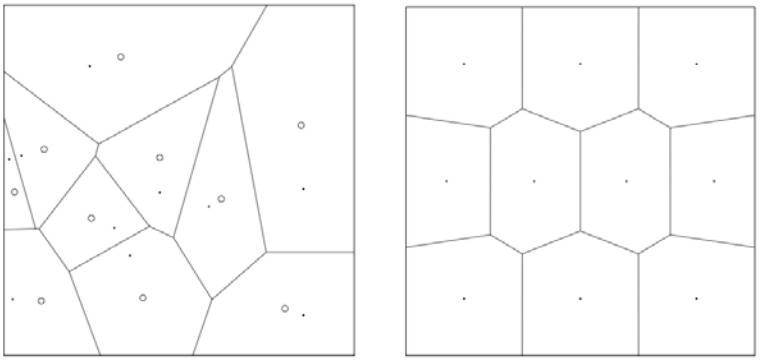


Figure 33: On the left, the Voronoi regions corresponding to 10 randomly selected points in a square, using a uniform density function. The dots are the Voronoi generators (38) and the circles are the centroids (39) of the corresponding Voronoi regions . On the right, a 10 point centroidal Voronoi tessellation. In this case, the dots are simultaneously the generators for the Voronoi tessellation and the centroids of the Voronoi regions.

where $x \in \Omega$.

5.3.2 *Voronoi partitioning*

Proposition 5.3.1. *Optimal regions, obtained by minimizing Equation 40 corresponds to the* Voronoi regions, *weighted by the cost function* $\phi(s)$ *and are defined as follow:*

$$V_i := \{\mathbf{s} \in \Omega : \ \|\mathbf{q} - \mathbf{s}_i\| \leqslant \|\mathbf{q} - \mathbf{s}_i\|, \forall i \neq j\}$$

in particular, the weight function assumes the minimum value:

$$H_{VOR} = \int_{\Omega} \min_{i \in \{1,\dots,N\}} f(\|\mathbf{s}_i - \mathbf{q}\|)\phi(\mathbf{q})d\mathbf{q}$$

In the following, we assume the specific form for the function $f(\cdot)$:

$$f(\|\mathbf{s}_i - \mathbf{q}\|) := \|\mathbf{s}_i - \mathbf{q}\|^2 \quad .$$

Given N points z_i, $i = 1, \dots, N$,representing the Voronoi generators (38), we can define their associated Voronoi regions V_i, $i = 1, \dots, N$. Furthermore, given the regions V_i, $i = \dots, N$, we can define their mass centroids z_i^*, $i = 1, \dots, N$ using Equation 39.

On the other hand, we can consider the following problem: given a region $\Omega \subseteq \mathbb{R}^2$, a positive integer N and a density function ρ defined for every x in Ω; find N points $z_i \in \overline{\Omega}$ and N regions V_i that tessellate Ω, such that for each i, V_i is the Voronoi region for z_i and z_i is the mass centroid of V_i.

The described problem, corresponds to many practical cases in which robots are initially located in the same initial positions, or when we are interested in finding an optimal tessellation of the configurations space Ω without considering the initial positions of mobile nodes.

In the context described, we are interested in situations where:

$$\mathbf{z}_i = \mathbf{z}_i^*, \quad i = 1, \dots, N$$

i. e. the positions z_i that serve as generators for the Voronoi regions V_i are themselves the mass centroids of those regions. We call such a tessellation a *Centroidal Voronoi tessellation*.

Example 5.3.1. The solution to this problem is in general not unique. For example, consider the case of $N = 2$, $\Omega \subseteq \mathbb{R}^2$ a square, and $\rho \equiv 1$.

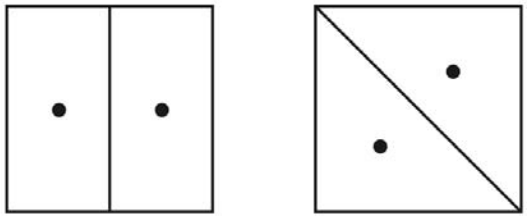


Figure 34: Two Voronoi tessellation of a square.

Two solutions to the problem described are showed in Figure 34, others may be obtained through rotations.

It is important to notice that uniform choice for generators $\{\mathbf{z}_i\}_{i=1}^N$ is an optimal choice only for squared mission space.

5.3.3 *Deterministic approach for determining centroidal Voronoi tessellations*

In this section, we discuss a deterministic approach for the determination of centroidal Voronoi tessellations of a given mission space. We refer to [12] for a comprehensive discussion and references on algorithms.

Given a discrete, finite-dimensional set of points $W = \{\mathbf{x}_l\}_{l=1}^m$ belonging to $\Omega \subseteq \mathbb{R}^2$, an integer $k > 1$ and an initial set of cluster centers $\{\mathbf{z}_i\}_i^N$, then for each $\mathbf{x} \in W$,

1. find the z_i that is closest to $\mathbf{x}$; denote the index if that $\mathbf{z}_i$ by i^*;
2. assign $\mathbf{x}$ to the cluster corresponding to z_{i^*};
3. recompute the cluster center $\mathbf{z}_{i^*}$ to be the mean of the points belonging to the corresponding cluster.

The three steps above presented and performed for each $\mathbf{x} \in W$ permit to determine a centroidal Voronoi tessellation.

If no particular constraints have to be observed, the initial set of cluster centers $\{\mathbf{z}_i\}_{i=1}^N$ can be chosen randomly.

Other deterministic approaches for determining Voronoi tessellations are presented in [45] [49] . For a comprehensive description of the deterministic and the approaches for the determination of centroidal Voronoi tessellation see [12].

5.3.4 *Extension to Voronoi partitioning in discretized environments*

During the whole previous discussions, the restricting assumption that the set Ω has to be a convex set were done. A set Ω is said convex if, all couples of points within the set, every points located in the straight line segment that join the pair of points is also within Ω. This assumption can be quite limiting in many practical cases, for example if we refer to indoors rooms or to presence of mountains and hills in outdoor explorations. In such cases the proposed methods cannot be applied. Figure 35 report an example of an environment composed of 5 rooms, representing a non-convex set.

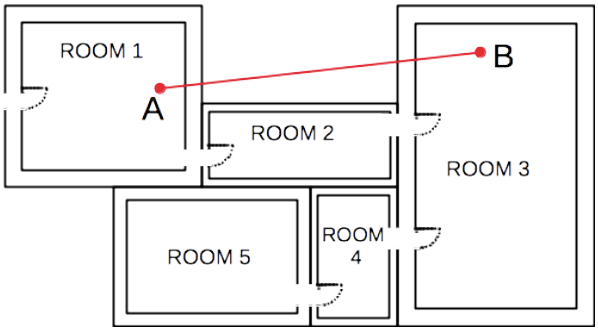


Figure 35: Example of an indoor environment that cannot be modeled as a convex set.

In [13], Durham et al. introduce a method capable of extending the solution to non-convex configurations space Ω. This is based on a discretization of the mission space. Ω has been divided according to a uniform grid and to each cell has been assigned a value representing the presence or not of obstacles.
Figure 36 shows the discretization of the environment presented in Figure 35.

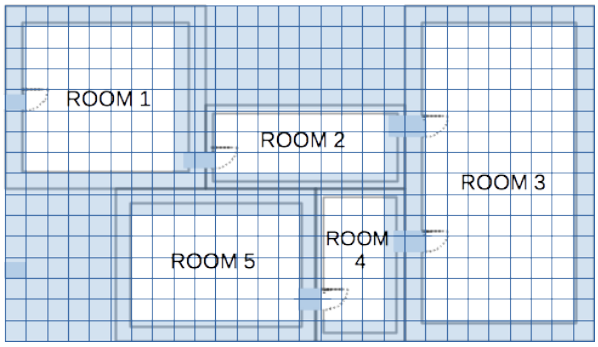


Figure 36: Discretized environment of Figure 35. Note that colored squares model the presence of obstacles while white squares represent the walkable environment after discretization.

The definitions regarding Voronoi partitioning for continuous environments, needs now to be extended to discretized ones. First of all, it is necessary to extend the definition of function $f(\|s_i - x\|)$ for discretized

environments. This can be done through the definition of the distance between a couple of cells $d(h,k)$.
In typical applications where the size of the cells tends to zero and in absence of further constraints, it is reasonable to define the distance $d(h,k)$ as the Euclidean distance between the centers of the squared cells.

In this terms, the cost function Equation 40 for discretized environments, assumes the form:

$$H_D := \sum_{i=1}^{N} \sum_{k \in V_i} d(h(i),k)\phi(k)$$

where we have supposed that agent i is initially located at $h(i)$.

5.4 DISTRIBUTED COVERAGE CONTROL IMPROVED BY PARTITIONING

As shown in Figure 32, representing a simulation framework, some troubles arises when Equation 33 is employed in unknown environments.
In particular, it is easy to deduce that the proposed solution based on optimization provides acceptable results only in cases where the considered function $R(x)$ presents a single maximum, i. e. maximum is absolute.
In fact, the proposed approach plans to move along the gradient direction of $R(x)$ until reaching the closest maximum for the considered events function.

This leads mobile agents to converge in minimum number of steps to the location of major probability of events taking place. However, this approach leads to a non-optimal examination of the configurations space. In fact, since agents has no a-priori knowledge about $R(x)$, that is their knowledge about the mission space is local and restricted to the portion they are actually observing, then the optimization they perform could be merely local. Therefore a local optimization for Equation 33 lead to sub-optimal solutions to the problem.

So motivated by circumstances just explained it emerges the need to provide a global (instead of a local) estimate for the function $R(x)$. In total absence of a global knowledge about $R(x)$, then mobile agents have to employ in order to provide an initial estimate for the function describing events taking place.

The main idea behind the method we propose, relies on giving a global (instead of a local) knowledge of $R(x)$ to each single agent. In order to guarantee a global view of the area to monitor in absence of information a priori, agents have to be imposed to perform a complete inspection of the whole mission space.

The main contributions in this approach are threefold:

- agents are controlled to perform an initial inspection of the whole mission space. This produce a global estimation for the function $R(x)$;
- a partitioning for the entire mission space is introduced, this provides a suitable method in order to approximate $R(x)$ in areas not visited yet;

- interest in non-visited partitions is updated over time.

Moreover, the following assumptions are considered:

- although an accurate model for the mission space requires the function $R(x)$ changing over time, i. e. $R(x) = R(x,t)$, it reasonable to assume function $R(x,t)$ slowly changing over time; this justify the following:
- when a partition get visited at time t^* by one agent of the network, then the agent has a detailed knowledge of that area, Hence in $R(x,t)$ the dependence on t disappear.

Let us denote by $P_v(x, t-k)$ the probability that point x have been visited by almost one agent of the network during last $(t-k)$ steps. Let the set $\{V_i\}_{i=1}^N$ be a tessellation of Ω.

The optimal coverage problem presented in Section 5.1.4 can be extended, leading agents to build a global knowledge of the function $R(x,t)$ for each agent, by enhancing the weight function $F(s)$ in Equation 33 as described by the following:

$$F_1(s) = \int_{\Omega_i} R(x)P_i(x,s)dx + \sum_{i=1}^{N} \int_{V_i} (1 - P_v(x, t-k))dx \,. \tag{41}$$

Notice that $(1 - P_v(x, t-k))$ represents the probability that x has not been visited during last $t-k$ steps by any agent of the network.
The additional term introduced, with respect to Equation 33, induces an increment in the function $F(s)$ in correspondence of partitions that have a lower probability of being already visited by any agent. The relative increase of $F(s)$ in correspondence of these points, can be interpreted as an enhancement in the interest that the network has in visiting this points, as they have never been visited before.

It is important to underline that the tessellation $\{V_i\}_{i=1}^N$ has to be shared among all the agents composing the network.

In order to insert dynamics induced by non-visited areas, Equation 41 can be improved introducing the grows in interest in non-visited partitions:

$$F_2(s,t) = \int_{\Omega_i} R(x)P_i(x,s)dx + \sum_{i=1}^{N} \int_{V_i} [1 - P_v(x, t-k)]\rho(x, t-k)dx \,. \tag{42}$$

where $\rho(x, t-k)$ is a monotonic increasing function of time t.
The function $[1 - P_v(x, t-k)]\rho(x, t-k)$ indicates the probability that x have not been visited during the last $(t-k)$ steps, and this should a monotonic increasing function over time t to produce growth in interest in non-visited cells.

Example 5.4.1. For instance, typical models for the probability $P_v(x)$ in regular (squared) mission spaces are:

$$P_v(x) \sim \quad \mathcal{N}(\mu, \sigma) \quad .$$

In fact, we can suppose that central points of the mission space are examined more frequently, while peripheral areas are less frequently visited.

In addition, more sophisticated agents, can be equipped with memories containing previously visited partitions. In this type of systems, the function $P_v(x)$ degenerates to the deterministic case.

This kind of approach is implemented and described in detail simulation results in Chapter 6

Proposition 5.4.1. *Given a robot network operating in Ω, comprehending one robotic agent implementing weight function described by (47).*
Let $R(x)$ be the probability density function describing frequencies at which events takes place in Ω and let $\rho(t)$ be a monotonic increasing function of t in $[0, t]$.
Let the set $\{V_i\}_{i=1}^{N}$ be a tessellation of Ω.

Then the whole mission space is being visited in a time t^* such that:

$$\rho(t^*) = N \cdot \frac{\overline{R}|\Omega|}{|V_i|} \quad . \tag{43}$$

where $\overline{R} = \max_x R(x)$ and $|\cdot|$ denotes the area of the polygon described by its argument.

Proof. In order to fix ideas, let us set $k = 0$, i. e. we consider the entire time frame $[0, t]$. The proposition can be demonstrated, by showing that holds:

$$\int_{\Omega_i} R(x) P_i(x, s) dx < \int_{V_i} [1 - P_v(x, t)] \rho(x, t) dx \cdot$$

where V_i represent the most interesting partition of the mission space. Since $R(x)$ is a probability density function,

$$\int_{\Omega} R(x) dx = 1$$

then it is also a limited function i. e.

$$\exists \overline{R} \; R(x) < \overline{R} \quad .$$

Since $P(x, s)$ represents a probability, i. e. $P(x, s) \in [0, 1]$ we can consider the following inclusions:

$$\int_{\Omega_i} R(x) P_i(x, s) dx \leqslant \overline{R} \int_{\Omega_i} 1 dx \leqslant \overline{R}|\Omega| \tag{44}$$

where $|\Omega|$ represent the area of the polygon described by Ω.

Let $\rho(x, t)$ be a constant function over the entire partition V_i, i. e. :

$$\rho(x, t) = \rho(t)$$

In a similar manner, for all the partitions V_j that have not been visited yet, we can write:

$$\int_{V_i} [1 - P_v(x,t)]\rho(x,t)dx \geqslant \int_{V_i} 1dx \cdot \rho(t)\cdot \tag{45}$$

since:

- $1 - P_v(x,t) = 1$, since partition i have not been visited;
- the function $\rho(t)$ can be brought outside the sign of integral as it does not depend on x.

Denoting $\int_{V_i} 1dx$ as $|V_i|$, and combining Equation 44 and Equation 45, together with Equation 47 we obtain:

$$\overline{R}|\Omega| < |V_i| \cdot \rho(t)\cdot$$

Assuming $\rho(t)$ be a monotonic increasing function of t, we can state that partition V_i will surely be visited when

$$\rho(t) > \frac{\overline{R}|\Omega|}{|V_i|} \tag{46}$$

In addition, it is important to underline that condition (43) guarantees the visit of partition i as $\rho(t)$ is a monotonic increasing function.

□

Starting from this assumption, each mobile agent can then move to other partitions in order to create a global estimate of the whole space.

5.4.1 *Adding distances cost*

Although Equation 42 arises as a robust and effective technique, capable of exploring the whole configurations space in a upper-bounded time, some improvements can be introduced in order to improve energy consumption. In fact, motivated by simulations results proposed in Chapter 6, it is easy to imagine that trajectories performed by the agents are not optimal, since the weight function considers only informativeness of partitions, while energy consumption needed to reach the partition is neglected.

More efficient results in terms of distance traveled by agents, can be obtained by weighting the additional term proposed in Equation 42 with the distance that the agent needs to travel in order to reach the partition of interest. In particular, the Euclidean distance $||z_i - z_j||$ between the centroid of the actual Voronoi cell and the centroid of the partition of interest is used.

Therefore, the weight function considered is:

$$F_3(s,t) = \int_{\Omega_i} R(x)P_i(x,s)dx + \\ + \sum_{i=1}^{N} \int_{V_i} [1 - P_v(x,t-k)]\rho(x,t-k)dx \cdot ||z_i - z_j||^{-1}_{i\neq j} \tag{47}$$

where z_j denotes the centroid of the currently partition in which the agent resides, and z_i indicates the generic centroid of partition i.

5.4.2 *Iterative update of mission space model*

Control strategies proposed in previous sections provide suitable and effective method in order to

- perform an initial estimation for the function $R(x)$, when configurations spaces are unknown;
- perform optimal coverage control, when $R(x)$ is exactly known.

Indeed, once an initial estimation for $R(x)$ is made, then control strategy defined by (33) provides a robust solution to the coverage control problem.

Therefore we deal with two different control strategies, both based on optimization, the former capable of estimating $R(x)$ while the other capable of performing optimal coverage control. Our purpose for this section, is to study and present how these two methods can be combined in order to provide a system able to self-update and to provide optimal coverage control in time-variant mission spaces.

When the actual function $R_0(x) = R_0(x,t)$ i. e. the function that describe actual frequency of events taking place is variable over time, we are interested in providing a strategy that continuously update our estimated model $R(x,t)$, in order to get an updated model describing the mission space.
It is important to underline that in this process of initialization for $R(x,t)$, areas rapidly changing over time i. e. areas described by peaks for $R(x,t)$ requires to be visited more frequently.

In the following, we propose an iterative method capable of fulfilling control objectives just explained, through combining optimal control described weight functions (33) and (42). Our method, is based on an iterative application of the two control laws, as described in detail by the following flux diagram:

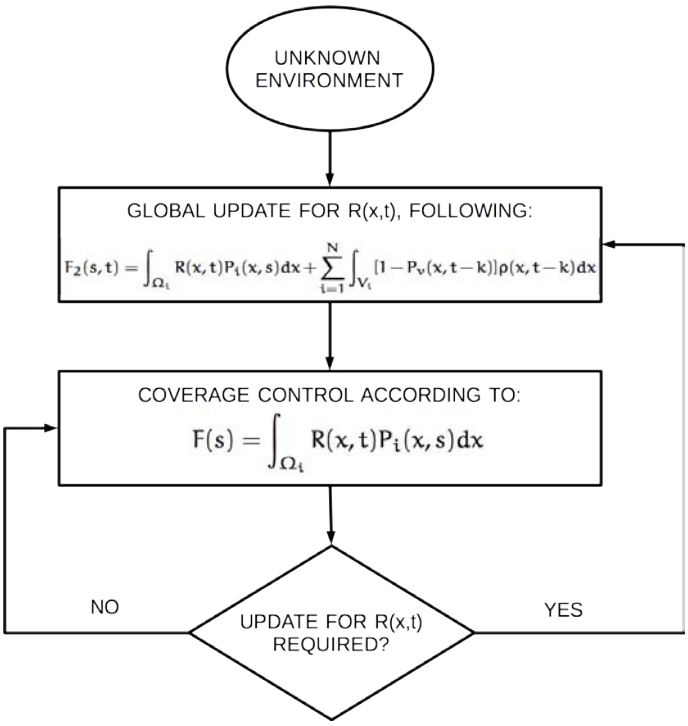


Figure 37: Iterative update of mission space.

As explained by the figure, the approach we propose is based on an alternation of the two phases:

1. mission space model update;
2. coverage control exploiting the global knowledge acquired during the previous phase.

The frequency at which the control can require global updates for $R(x,t)$ can depend on:

- average updating rare for $R_0(x,t)$, that is, mission space slower changing over time can be updated with lower rates;
- accuracy required for the estimation $R(x,t)$;
- fixed sensors dislocation and tightness.

5.4.3 *Maps merging*

When single agents are included in networks composed of a variety of sensors, they needs to be equipped with control policies that enable cooperation. This problem is referred as the *cooperative coverage control*.

Although in previous sections we have already overcome the problem of redundancy of areas monitored, as described by Equation 32, we are now interested in extending cooperation policies even during mission space initialization.

In fact, according to control policies described in previous sections, agents perform initial estimation $R(x)$ according to Equation 42 in an autonomous manner, that is, without exploiting cooperation with neighbors.

In this context, cooperation policies forces agents to combine their knowledge about the environment that has been acquired during the initialization process, whenever an encounter occur.

This approach can be explained with the aid of the following figure:

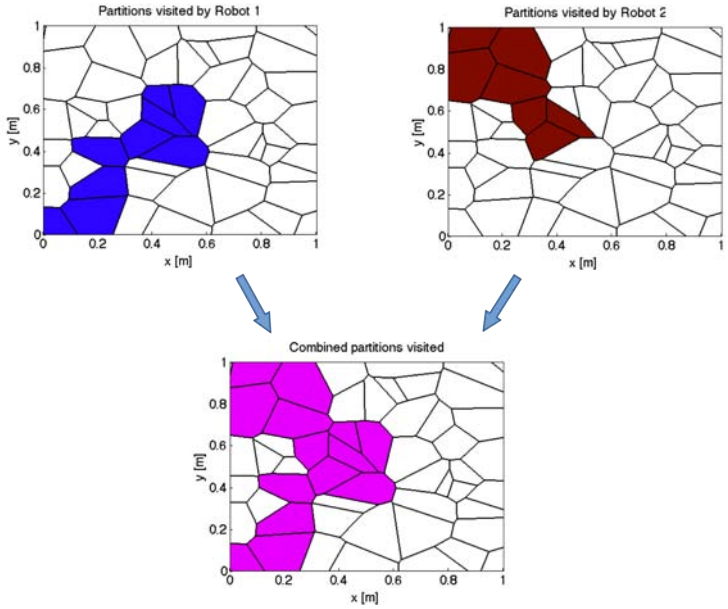


Figure 38: Single agents knowledge are combined, whenever an encounter occur.

where, as a consequence of the meeting between Robot1 and Robot2, their local map have been combined in order to build an overall knowledge for the environment. Subsequently both agent dispose of the combined knowledge of the configurations space.

5.5 BEACON PLACEMENT FOR COVERAGE CONTROL

In *dynamic coverage control*, agents are capable of moving in order to increase the area covered with time, until every point in the given area has been covered with some prescribed coverage level. Thanks to mobility of agents employed, the network can adapt for changing conditions in the mission space, and enable the network to monitor widely extended areas, even when employing a reduced number of sensors.

On the other hand, the static approach to coverage control involves positioning sensors without any further mobility. Thus, *Static coverage control* deals with achieving an arrangement for agent that minimizes the probability of undetected events. The sensors or agents determine optimal positions based on gradient climbing methods, until the optimal configuration is reached. The optimal configuration maximize the collective reward function.

Advantages introduced by Static coverage control, athwart dynamic approach are twofold:

- persistent sensing of interested areas;
- higher coverage levels;

These aspect cannot be in general guaranteed by the dynamic version, since nodes are required to travel across the configurations space in order to cover larger extension areas. On the other hand, in such cases in which $R(x,t)$ is highly changing over time, fixed sensors become unnecessary since their position cannot be determined in optimal manner.

Motivations just explained, justify a *mixed coverage control*, that is, the exploitation of both fixed and robotic sensor in order to perform a robust and reliable coverage control when $R(x,t) \approx R(x)$.

In the following, our purposes are twofold:

- providing a given covered threshold for areas with the highest information density;
- maintaining complete covering of the remaining areas of the mission space.

5.5.1 *Static sensor coverage model*

Assume the network is a squared sensor field Ω and k sensors have to be deployed. Each sensor has a detection range r_s, and sensor s_i is deployed at (x_i, y_i). For any point P at (x,y), we denote the Euclidean distance between s_i and P as:

$$d(s_i, P) = (x_i - x)^2 (y_i - y)^2 \quad . \tag{48}$$

A typical sensor model expresses the coverage $C_{xy}(s_i)$ of the sensor s_i as:

$$C_{xy}(s_i) = \begin{cases} 1 & \text{if } d(s_i, p) < r_s \\ 0 & \text{otherwise} \end{cases} \tag{49}$$

for all $P \in \Omega$. It assumes that sensor readings have no associated uncertainty in reality, and sensor detection are imprecise.

When the coverage $C_{xy}(s_i)$ needs to be expressed in probabilistic terms, hence a precise detection model is introduced:

$$C_{xy}(s_i) = \begin{cases} 0 & \text{if } d(s_i, P) \geqslant r + r_l \\ e^{-\lambda \alpha^\beta} & \text{if } r - r_l < d(s_i, P) < r + r_l \\ 1 & \text{if } d(s_i, P) \leqslant r - r_l \end{cases} \tag{50}$$

where $\alpha = d(s_i, P) - (r - r_l)$ and r_l ($r_l < r$) is a measure of uncertainty in the sensor detection, α and β are parameters that measure detection

probability when a target is at distance greater than r_l but within maximum from the sensor. This model reflects the behavior of range sensing devices such as infrared and ultrasound sensors. Typical values for parameters involved are: $r = 10$, $r_l = 5$, $\beta = 0.5$ and $\lambda = 0.5$.

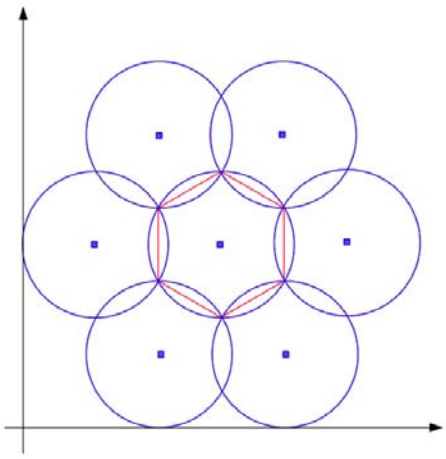


Figure 39: Optimal hexagon-based sensor distribution.

The choice for parameter r_l lead to two different behaviors:

- when $r_l \simeq 0$ i. e. r_l can be neglected, we are using the binary sensor detection model, attempting to prevent the detection region of two sensors from overlapping. This optimal distribution of sensor nodes is illustrated in Figure 39.
- if $r_l > 0$ the probabilistic sensor detection is used. Due to uncertainty in sensor detection responses, grid points are not uniformly covered with the same probability. Some grid points will have lower coverage rate if they are covered only by one sensor and far from other sensors. In this case, it is necessary to overlap the sensor detection area to compensate for the low detection probability.

The probability of the point (x, y) being covered by a couple of sensors s_i, s_j is denoted as,

$$C_{xy}(s_i, s_j) = 1 - (1 - C_{xy}(s_i))(1 - C_{xy}(s_j)) \quad .$$

Definition 5.4. Let C_{th} be the desired coverage threshold for all grid points. The problem of maintaining a given covered threshold can be formulated as finding locations for sensing devices s_i, s_j such that, for all points (x, y) in the mission space:

$$\min_{x,y}\{C_{xy}(s_i, s_j)\} \geqslant C_{th} \quad . \tag{51}$$

It can also be extended to a region which is overlapped by a set of k sensors. That is $\{s_1, s_2, \ldots, s_k\}$. The coverage in this case is given by:

$$C_{xy}(k) = 1 - \prod_{j=1,\ldots,k} (1 - C_{xy}(s_j)) \tag{52}$$

5.5.2 *Coverage control through mixed robot and sensor networks*

The method we propose is based on a mixed network, i. e. on a group composed of both fixed and robotic agents. In particular, our control scheme involves:

- fixed sensors for monitoring areas with the highest information density;
- robotic agents for monitoring remaining areas of the mission space.

Thanks to the cooperation between the dynamic and Static coverage control schemes, our method is able to guarantee a persistent sensing in highly informative areas, while the remaining areas of mission space continue in being completely covered thanks to mobile agents. Let R_{MIN} be a threshold that discriminates between highly an lower informative areas, that is:

- if $R(x) > R_{MIN} \Rightarrow x$ is considered a highly-informative point
- if $R(x) < R_{MIN} \Rightarrow x$ is considered a lower-informative point

and let

$$\hat{R}(x) := \{x : R(x) > R_{MIN}\}$$

the region containing highly-informative points.

In the following, it is reasonable to assume the following:

1. robotic nodes are capable of deploying fixed sensors in the mission space.

It is moreover important to recall the previously-stated assumption $R(x,t) \approx R(x)$. This enables to state that function $R(x)$, and in particular its peaks, are almost not varying over time. This assumption is fundamental in order to compute fixed sensors optimal locations; in fact this kind of agents are capable of guaranteeing higher coverage levels but their do not provide flexibility in terms of mobility.

Exploiting global knowledge of the mission space own by robotic agent, the design of candidate location for fixed sensors can be performed as a global optimization problem. Therefore mobile agents can compute locations for fixed sensors nodes through:

$$s_i = \{x | \max_x \hat{R}(x)\}$$

mobile agents are then capable of deploying fixed sensors directly thanks to assumption (2).

This approach arises as a suitable method capable of monitoring with a high update rate areas where events takes place frequently, while the remaining parts of the configurations space continues to maintain acceptable levels of coverage thanks to mobile agents.

For a detailed description on persistent coverage control, please refer to [9] and [20].

Part III

SIMULATION RESULTS

Simulations results allow to assess novel solutions proposed in the theoretical previous chapters. Methods described in previous chapters to improve network task quality through interplay between sensor and robotic network are validated and discussed through Matlab simulation software.

6 SIMULATION RESULTS IN MATLAB

This chapter presents numerical simulations performed in order to validate theoretical approaches described in previous chapters. The simulation setup comprises a unit square mission space (1m x 1m), with three fixed beacons employed to perform localization. Robotic agents are represented with colored triangles. Triangular-shape displays both position and orientation of the agent. Beacons are in general represented by black and white dot patterns.

Simulations are organized in sections, each of them comprehend results regarding previous chapters; in particular:

- Section 6.1 presents simulations results regarding Chapter 3;
- Section 6.2 presents simulations results regarding Chapter 4;
- Section 6.3 presents simulations results regarding Chapter 5;

6.1 LOCALIZATION

In the former part of this Section, odometric techniques are described and analyzed in terms of quality of target estimated position, when noise affects sensors readings.
In the second part, closed-form expressions for variance of computed locations are exploited in order to graphically derive the uncertain behavior as function of target's location; results obtained are then discussed and compared in terms of both localization error and variance of estimated position. The part moreover introduce a comparison between methods in typical agent trajectories.
The third part of the Section concludes describing performances of statistical filtering in simulation results. The implementation of the Kalman filter is discussed in detail and design choices are motivated and discussed.

6.1.1 *Dead reckoning*

Dead-reckoning is the most widely odometry-based method employed for determining position and orientation of a specific target. In many practical applications dead-reckoning provides easily and accessible real-time positioning information. In general, the frequency at which the measurements and the algorithm have to be performed depends to a large degree on the requested accuracy for the system.

Dead-reckoning is a relative positioning method, and it consists of evaluating the position of a mobile robot by using velocity and angles measured by encoders attached to the robot's wheels. This method estimates relative

position of the target by exploiting information about its initial location and then integrating measured gathered by sensors. The main advantages are that it is simple, low-cost and has an easier time in estimating the position in real time, compared to absolute positioning methods.
However, since it is based on the odometric information arising from wheels, this kind of methods are subject to major accumulation of errors caused by wheel slippage, mechanical tolerances and surfaces roughness that could let the robot fail to keep track of its true location over long distances.

When measuring dead-reckoning errors, one must distinguish between:

- SYSTEMATIC ERRORS, which are caused by kinematic imperfections of the mobile robot e. g. unequal wheel diameters;
- CASUAL ERRORS, which can not be removed; in some cases these can be caused by wheel-slippage or irregularities of the floor.

In general, systematic errors are a consequence of errors in the model of the robot itself, and they lead to drifting issues; on the other hand non-systematic errors are in general function of the characteristics of the floor and in general cannot be recognized or removed. In the following, we assume absence of systematic errors. This can be achieved, for instance, through adequate sensors calibration.
In order to discuss troubles arising when targets are located through odometry-based methods, some preliminary simulations have been conduced and discussed.

In Figure 40a is presented an example of robot trajectory in the mission space. The robot position is represented by a colored dot, while its orientation is explicated by the adjacent triangle. This path will be considered in following a typical trajectory performed by the robot in order to perform a specific task. The path shown can be described as follow:

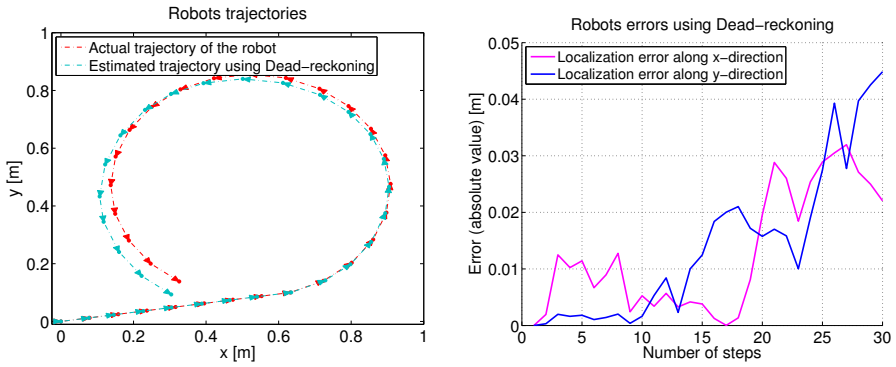
- the robot initially perform a rectilinear path (for the first 10 steps) in each of them the average speed is $v_f = 0.8m/s$;
- the second part of the trajectory resembles a circular path: each step is characterized by a variation of phase of $\Delta\theta = 0.25rad$ and an average speed of $v_f = 1m/s$;

and the trajectory contains a total amount of 30 steps. Note that initially the robot is located in $(x_0, y_0) = (0, 0)m$ and its orientation is $\theta_0 = 0rad$.

Figure 40b displays the dead-reckoning localization error corresponding to the path described. Notice that errors in the two directions x and y, have been computed as difference between the exact position of the robot and the estimated one.
Simulation has been conduced introducing additive Gaussian noise of variance $R_w = 0.01m^2$ for modeling casual errors generated by sensors.

Figure 40b highlights the increasing trend for localization error caused by the continuous integration of casual errors, as previously introduced.

Effects of errors integration can moreover be noticed in the graphical trajectories plot (Figure 40a): the result is an always increasing deviation between actual and estimated trajectories.

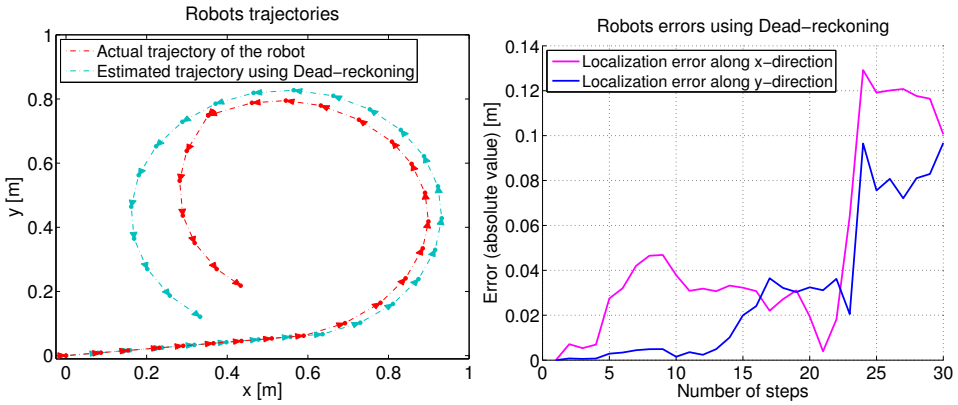


(a) Graphical representation of the trajectory of the robot in the mission space.

(b) Error evolution along x and y direction.

Figure 40: Robot behavior when performing dead reckoning, in presence of casual errors with variance $R_w = 0.01m^2$.

Figure 41 shows typical errors caused when wheels slippage occurs. In particular the wheel slippage has been simulated at iterations $t = 23$ and $t = 24$. This behavior is representative of typical practical cases in which robots travel across different types of terrains, and the wheels grip is not always guaranteed. The graphical representation of error as function of the time clarify how the estimated location drastically get worse when this type of situations occur.



(a) Graphical representation of the trajectory of the robot in the mission space.

(b) Error evolution along x and y direction.

Figure 41: Robot behavior when performing dead reckoning, in presence of casual errors with variance $R_w = 0.01m^2$ and wheel slippage at iteration $t = 23$ and $t = 24$.

6.1.2 *Geometric triangulation: Variance map*

Advantages introduced by anchor-based localization methods enable to overcome problems previously introduced regarding odometric techniques. Moreover, thanks to this kind of cooperation between nodes, anchor-based methods enable to obtain an estimation for reliability of estimated position. Variance estimate, as discussed in Section 3.4, depends both on disposition of beacons and on the relative position of the robot with respect to them. In order to derive and discuss the behavior of uncertain, this section proposes some graphical plot regarding variance estimation as a function of robot position in the mission space.

The simulation setup comprises three beacons disposed in a static configuration: beacons are equally distributed on a circle of radius $r = 0.3m$ and $120°$ out of phase. The three beacons form an equilateral triangle, whose vertices are located in positions reported in Table 1.

The configurations space has been discretized for numerical purposes

	x [m]	y [m]
Beacon 1	$0.5 + 0.3\cos(\pi/3)$	$0.5 + 0.3\sin(\pi/3)$
Beacon 2	$0.5 + 0.3\cos(-\pi/3)$	$0.5 + 0.3\sin(-\pi/3)$
Beacon 3	$0.5 + 0.3\cos(\pi)$	$0.5 + 0.3\sin(\pi)$

Table 1: Location of beacons for the first configuration.

into a matrix composed of unit squares of length $l = 0.001m$. Each of this squares represent a (discretized) possible position for the robot. The orientation is arbitrarily set to $0°$.

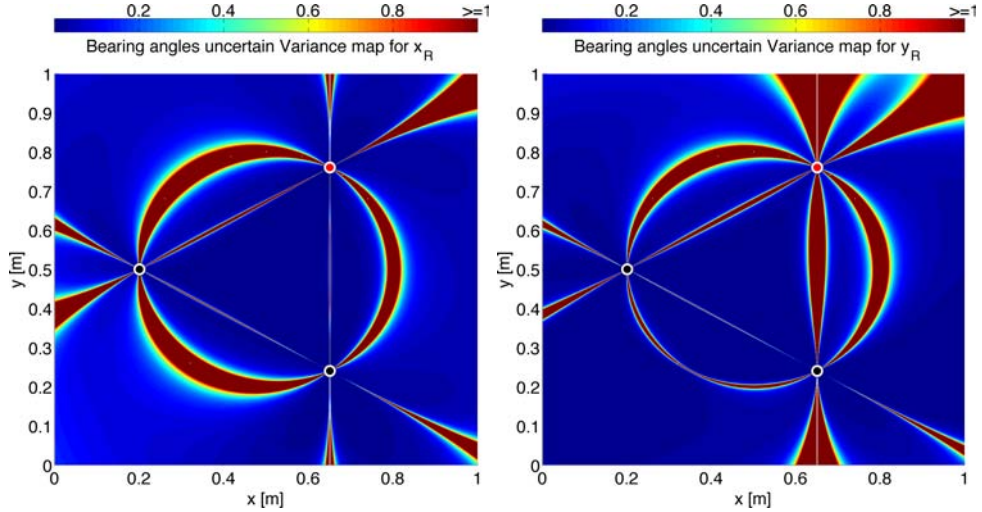
Simulations have been conduced through considering two types of uncertainty affecting localization:

- uncertain on beacons positions;
- uncertain on measured bearing angles.

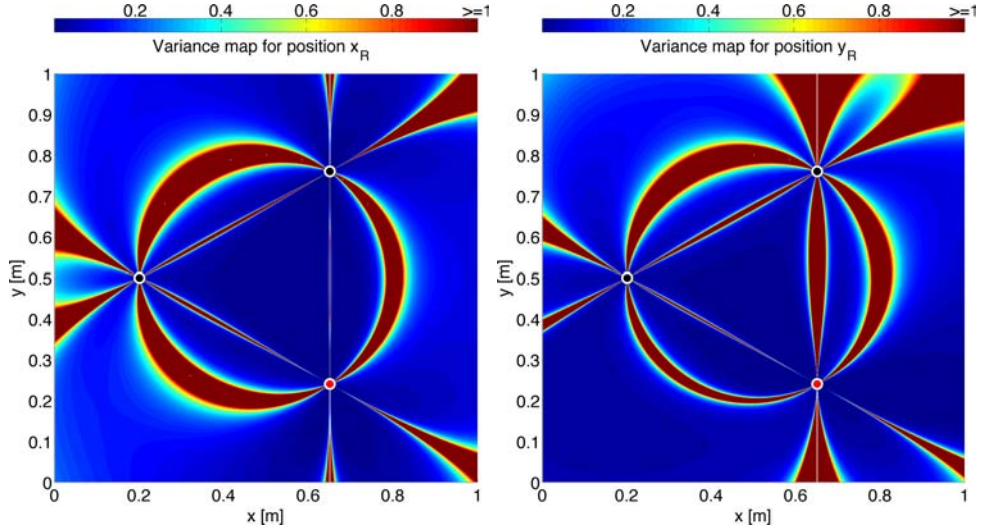
The former models uncertain on actual position of beacons; while the latter is in general caused by sensors inaccuracies.

6.1.2.1 *Uncertain on beacons positions*

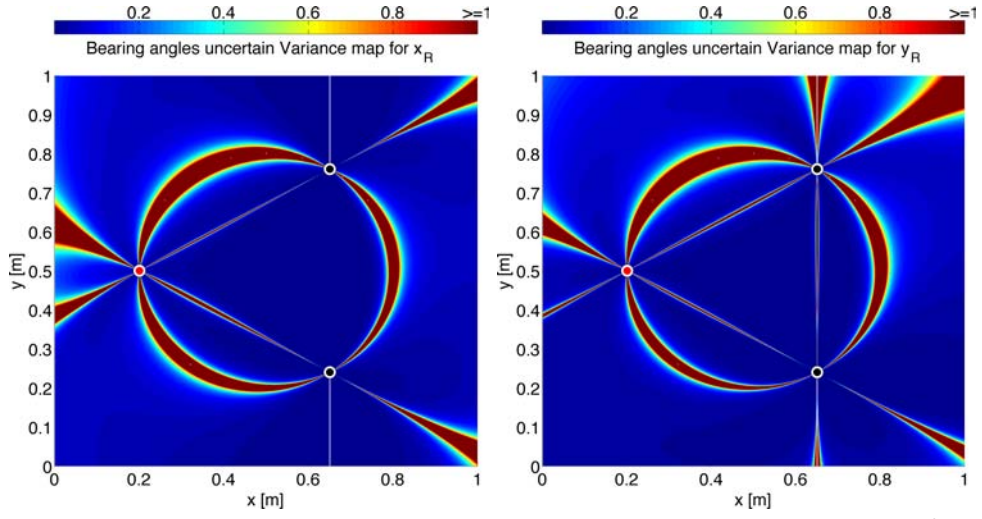
In many practical cases, in particular when adopting very unreliable sensors or in such cases when GPS is not available or its estimate is not reliable, the absolute position of anchor nodes is not exactly known, i. e. it has to be modeled through a random variable. Effects of this kind of uncertain have to be taken into account to compute variance of final estimated location of the robot.



(a) Variance map when location of B1 (Red) is a random variable with variance $\sigma_B^2 = 0.01m^2$.



(b) Variance map when location of B2 (Red) is a random variable with variance $\sigma_B^2 = 0.01m^2$.



(c) Variance map when location of B3 (Red) is a random variable with variance $\sigma_B^2 = 0.01m^2$.

Figure 42: Robot position triangulation variance in m^2, when beacons positions are Gaussian random variables with standard deviation $\sigma_B^2 = 0.01m^2$. Notice that the function has been saturated to $1m^2$ for graphical reason.

Formally, beacons positions (x_i, y_i) $i = 1, 2, 3$, are assumed to be random variables, with Gaussian distribution characterized by zero mean and variance $\sigma_B^2 \in \mathbb{R}^2$. It is reasonable to state that this formulation models casual errors due to uncertain on the knowledge of exact position of beacons.
Figure 42 presents variance behavior as a function of robots positions in the mission space. In the following, this kind of graphical representation is referred as *Variance Map*. The beacons' locations are represented by black and white or red dot patterns. We underline that the comparison proposed in Figure 42 helps in evaluating effects of uncertain in locations of the three beacons, however it does not represent a realistic scenario since in general locations of all three beacons are modeled as random variables.
As a preliminary observation, it can be noticed by the figure that the variance of estimated position is flat and close to zero almost everywhere; except for areas corresponding to lines joining the three beacons and the region corresponding to the circle joining the three beacons. Indeed, as previously motivated in Section 4.3.1, these regions arises as areas where geometric triangulation cannot determine the solution. Effectively,

- when the robot and beacons all lie in the same circumference, the three circles of triangulation coincide;
- when a couple of beacons are collinear, two of the three triangulation circles coincides, therefore estimation cannot be performed.

The figure induces another important observation: uncertain of $0.01m^2$ in actual beacons positions leads to variances of the same order of magnitude in targets localization, as shown by the colorbar in the figure. This is true for the largest parts of areas of the configurations space; while in correspondence of above-cited critic regions, variance rapidly diverges overcoming the unacceptable value of $1m^2$.
This is an important observation concerning the reliability of the system: errors in actual positions of beacons arise in errors of the same order of magnitude in targets estimated locations for the largest part of regions composing the configurations space.

Moreover, the figure highlights the symmetrical features of the variance function. Considering, for instance Figure 42a(left), it is possible to observe that the equilateral-triangle presents one of its symmetrical axes parallel to x-axis. The result is a perfectly-symmetric shape for the variance function with respect to the horizontal line described by $y = 0.5m$. This is an important observation and ensures that the uncertain of the estimate depends only on the relative positions of beacons respectively to the robot, and this estimate is independent from robot's orientation.
However, it can be noticed that Figure 42a(right) does not exhibit the same symmetry: this behavior is reasonable since the equilateral triangle considered does not hold any symmetry axis parallel to the y-axis.

Another important aspect that can be noticed from Figure 42c, is that the triangulation algorithm proposed, thanks to arrangements included and described in in Chapter 3, is capable of locating targets situated in the lines joining couples of beacons. However, as can be noticed from the

comparison between Figure 42c left and right, this computation exhibits different behaviors. Indeed, it is important to notice that in the left-figure, targets can be accurately located when situated in the vertical line joining B_1 and B_2; on the other hand this accuracy is not guaranteed in Figure 42b. This aspect is a consequence of the beacons disposal: indeed, when the target is located in the considered line bearing measured angles $\phi_1 = 0$, $\phi_2 = 0$, while $\phi_3 \neq 0$. As a consequence, the target is located in the line joining B_1 and B_2 therefore the x-coordinate can be computed with low uncertain. On the other hand, the same argumentation does not hold for y-coordinate as the computed solution strongly depend on B_3.

From the comparison between Figure 42 (a) (b) and (c), it is possible to discuss effects of errors related to inaccuracies in the knowledge of actual locations of beacons.

It is preliminary possible to establish the similarity between Figure 42a and Figure 42c that show the variance map when B_1 and B_3 respectively are random variables of variance $\sigma_B^2 = 0.01m^2$. On the other hand Figure 42b shows greater uncertainty levels with respect to above-mentioned figures.

This fact can be motivated through analytically analysis of the function describing the variance. Recalling equations Equation 20 and Equation 21, that express uncertain function:

$$\sigma_{x_R}^2 = \sigma_{x_2'}^2 + \left(\frac{y_{12}' - y_{23}'}{\det(A)}\right)^2 \sigma_{k_{31}'}^2 + \sigma_{y_{12}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \\ + \sigma_{y_{23}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \left(\frac{k_{31}'(y_{12}' - y_{23}')}{\det(A)}\right)^2 \sigma_D^2 \quad (53)$$

$$\sigma_{y_R}^2 = \sigma_{y_2'}^2 + \left(\frac{x_{23}' - x_{12}'}{\det(A)}\right)^2 \sigma_{k_{31}'}^2 + \sigma_{x_{12}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \\ + \sigma_{x_{23}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \left(\frac{k_{31}'(x_{23}' - x_{12}')}{\det(A)}\right)^2 \sigma_D^2 \quad (54)$$

Let us initially motivate Figure 42a. Indeed, when B_1 is a random variable, while locations of other beacons are deterministic, Equation 53 and Equation 54 can be rewritten as follow:

$$\sigma_{x_R}^2 \approx \sigma_{y_{12}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \sigma_{y_{23}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2$$

$$\sigma_{y_R}^2 \approx \sigma_{x_{12}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2 + \sigma_{x_{23}'}^2 \left(\frac{k_{31}'}{\det(A)}\right)^2$$

as motivated by following observations:

- $\sigma_{x_2'}^2 = 0$ and $\sigma_{y_2'}^2 = 0$ since the location of B_2 is deterministic;

- $\left(\frac{y'_{12}-y'_{23}}{\det(A)}\right)^2 \sigma^2_{k'_{31}} = 0$ and $\left(\frac{x'_{23}-x'_{12}}{\det(A)}\right)^2 \sigma^2_{k'_{31}} = 0$ since $\sigma^2_{k'_{31}}$ depends only on bearing angles variances, and these are assumed deterministic;
- $\left(\frac{k'_{31}(y'_{12}-y'_{23})}{\det(A)}\right)^2 \sigma^2_D$ and $\left(\frac{k'_{31}(x'_{23}-x'_{12})}{\det(A)}\right)^2 \sigma^2_D$ can be neglected.

Equations highlight the strict dependence of estimate position from variances of two of the three triangulation circles centers: $(\sigma^2_{x'_{12}}; \sigma^2_{y'_{12}})$ and $(\sigma^2_{x'_{23}}; \sigma^2_{y'_{23}})$. Figure 43 explains circles involved in the solution of the triangulation problem.

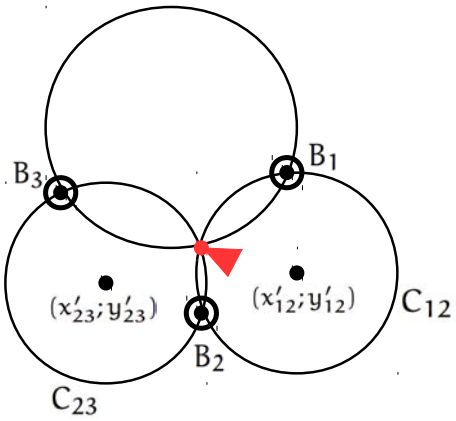


Figure 43: Circle centers of only two triangulation circles are involved in Equation 53 and Equation 54.

As can be noticed by the figure, uncertain on the knowledge on position of B_1 lead to uncertain in the circle C_{12} center only, since C_{13} is not involved in Equation 53 and Equation 54. Analogous considerations can be conduced for Figure 42c.

On the other hand, when we consider Figure 42b, it is easy to notice that uncertainty in the knowledge of location of the node B_2 leads uncertain in the knowledge of both C_{12} and C_{23} circles centers coordinated. Moreover in this case, additional terms $\sigma^2_{x'_{12}}$ and $\sigma^2_{y'_{12}}$ in Equation 53 and Equation 54 cannot be neglected. This observation motivates the greater uncertain levels in Figure 42b.

Finally it is important to underline the key role played by beacon B_2 that differs from other beacons. This is a consequence of the choice of translating the reference frame in the coordinates of B_2. Therefore, a good project choice is to locate fundamental reference frame on the more precise beacon. That is, B_2 has to be chosen the more precise node.

6.1.2.2 *Uncertain on bearing angles*

In practical applications for localization we have to deal with non-ideal measured bearing angles between the robot and beacons. This behavior can be caused, for instance, by finite resolution of bearing sensors mounted

on mobile agents.
Formally, bearing angles ϕ_{ij} measured by the robot could be affected by random Gaussian noise with variance σ^2_A, that models casual measures errors .
Figure 44 presents variance behavior as a function of target position in the entire mission space when $\phi_i \sim \mathcal{N}(0, \sigma^2_A)$, $i = 1, 2, 3$, $\sigma^2_A = 0.01 rad^2$.

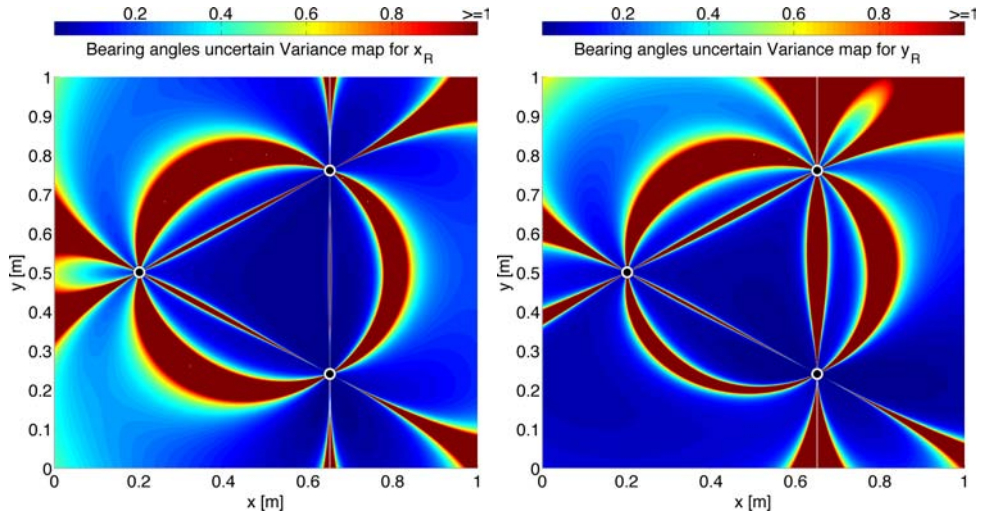


Figure 44: Variance behavior as a function of robot position when $\phi_i \sim \mathcal{N}(0, \sigma^2_A)$, $i = 1, 2, 3$, $\sigma^2_A = 0.01 rad^2$. Notice that the function has been saturated to $1 m^2$ for graphical reasons.

An important observation arise from the comparison with Figure 42:

- high uncertain regions increase in size;
- remaining parts of the configuration space manifest in general higher variances.

This worsening in localization quality can be motivated by the fact that uncertain on bearing measured angle can be interpreted as uncertain on the knowledge of the exact location of both the three beacons. Thus is reasonable that the whole mission space present degraded values of variance.

BEARING ANGLES WITH VARIANCE $\sigma^2_A = 0.001 rad^2$

One could be interested in understanding improvements arising when using more accurate bearing sensors to measure angles. In Figure 45 is shown the Variance map when $\sigma^2_A = 0.001 rad^2$.
From the comparison with Figure 44 it is possible to observe that right uncertainty regions are still present; however their size is reduced with respect to Figure 44.

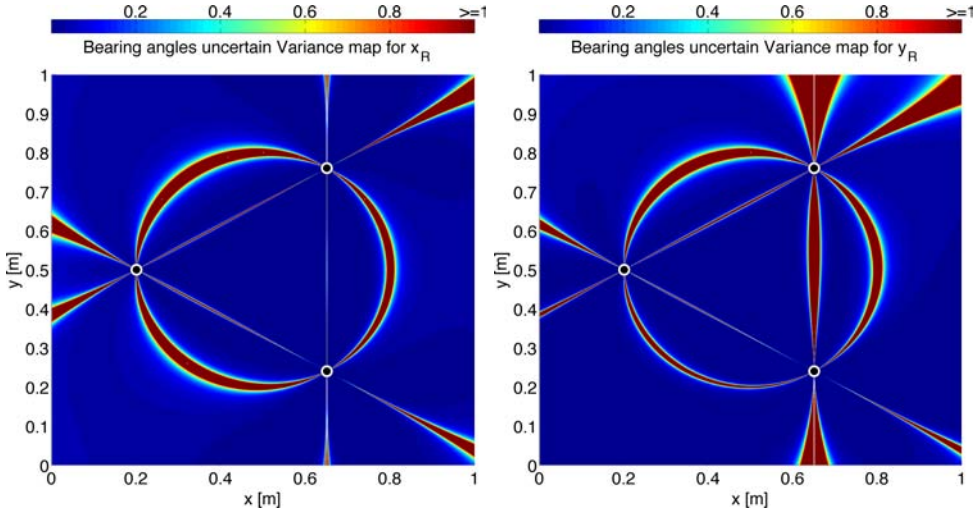


Figure 45: Variance behavior as a function of robot position when $\phi_i \sim \mathcal{N}(0, 0.001 m^2)$, $i = 1, 2, 3$.

The result is better quality of localization in the entire mission space. Therefore the comparison suggested that more precise bearing sensors lead to higher precision in location estimations and more precise sensors are preferred when available.

6.1.2.3 *Uncertain and triangulation error on a typical trajectory*

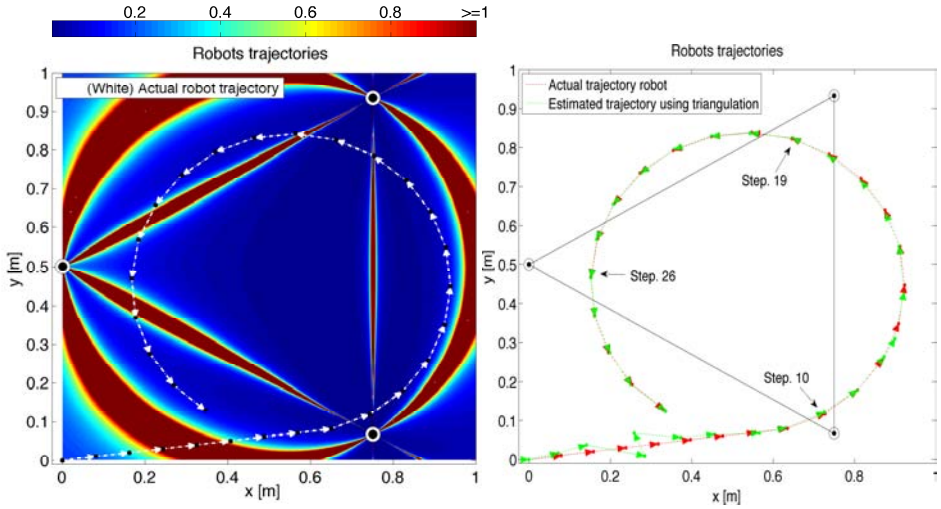
In this section the localization error and related estimated variance have been examined when the robot perform a typical trajectory in the mission space. The path followed by target is the same described in Section 6.1.1. Beacons have been located in positions described in Table 2. In the following we have assumed uncertain on bearing angles $\sigma_A^2 = 0.001 rad^2$ since it models realistic scenarios.

	x [m]	y [m]
Beacon 1	$0.5 + 0.3\cos(\pi/3)$	$0.5 + 0.5\sin(\pi/3)$
Beacon 2	$0.5 + 0.3\cos(-\pi/3)$	$0.5 + 0.5\sin(-\pi/3)$
Beacon 3	$0.5 + 0.3\cos(\pi)$	$0.5 + 0.5\sin(\pi)$

Table 2: Location of beacons for the second configuration.

Figure 46 shows the behavior of the actual and estimated through ToTal algorithm trajectories. Preliminarily it is important to observe that thanks to the exploitation of anchor nodes, geometric triangulation overcomes typical issue previously observed for dead-reckoning. These are, for instance, the increasing trend over time for the localization error and errors caused by wheels slippage.
In this terms, advantages introduced by geometric triangulation are consistent: indeed localization error does not depend on the number of iterations steps performed, but only on the relative position of the robot with

respect to anchor nodes. The localization error is shown in Figure 47a and its related variance in Figure 47a.



(a) Variance map and actual trajectory. (b) Actual and estimated robot trajectories.

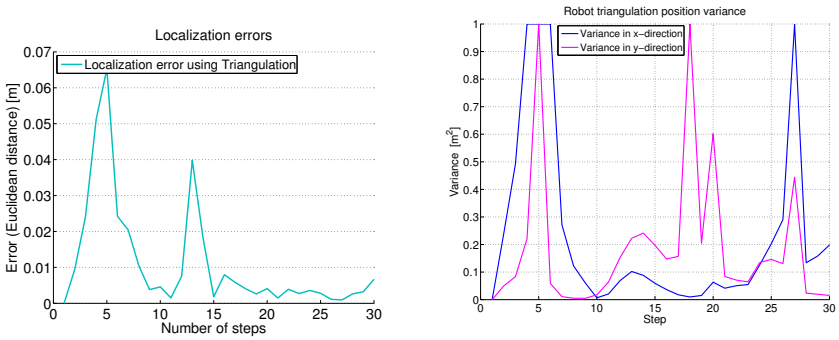
Figure 46: Actual and estimated trajectories through ToTal, when measured bearing angles are affected by Gaussian white noise with variance $\sigma^2_{\lambda} = 0.001 rad^2$.

During the initial linear path of the navigation phase, the mobile agent crosses the high-uncertain area determined by the circle passing through the three beacons. During this crossing phase, the estimated location is characterized by both high localization error and high variance as shown in Figure 47 at steps $3-6$. As can be noticed in Figure 46, in this phase the estimated location for the mobile target is definitely unreliable.

In the second phase of the navigation path, the mobile agent performs a circular trajectory internal to the circle above-mentioned. As highlighted by Figure 46, the mobile target repeatedly crosses lines connecting the beacons. As a consequence, these regions arise high uncertain in the estimated position, as can be noticed in Figure 47b at steps $17-20$ and $25-27$. However, as shown in Figure 47a, although the uncertain in estimated position increase, localization error remains low. This is a consequence of the adaptation of the algorithm for collinear beacons described in Chapter 3. Although the algorithm provides a feasible way in order to compute robot's position even when beacons are collinear through an approximation of $\cot(\cdot)$ functions, this estimation is characterized by high variance. Another important aspect to notice is the growth of estimation error at iteration $13-14$. As can be noticed by Figure 46, these iterations corresponds to regions close to the circle passing through the three beacons.

Finally, according to previous-given interpretations of error and variance, it is possible to resume simulation results in the following manner:

- the region of the mission space corresponding to the circle joining the three beacons leads both to high localization error and high variance



(a) Behavior of triangulation error (Euclidean distance between actual and estimated position).

(b) Behavior of triangulation variance in function of steps.

Figure 47: Triangulation error and variance when measured bearing angles are affected by Gaussian white noise with variance $\sigma^2_A = 0.001 rad^2$.

values; therefore these results as unreliable regions for geometric triangulation;

- regions of the mission space corresponding to lines joining couple of beacons provide lower values for uncertain error thanks to corrections introduced in ToTal algorithm; however the reliability of estimated position is lower.

Observations just conduced introduces the sensitivity of geometric triangulation technique on specific areas of the mission space, and motivates the study of mixed triangulation approaches (Section 3.5).

6.1.3 *Kalman filtering*

In the following, theoretical approach described in Section 3.5 and based on Kalman filtering is discussed. In particular, localization errors and variances are compared among methods; moreover our approach based on mixed triangulation and dead reckoning is described and project choices are motivated. As introduced in Section 3.5, Kalman Filter equations provide, in this context, a suitable way to combine data originated by on-board sensors together with geometric triangulation estimated position for the robot. Simulations have been conduced modeling on-board sensors imprecision with addictive Gaussian white noise of variance $R_w = 0.01 m^2$; triangulation is then affected by noisy angles measurement, modeled by white Gaussian noise of variance $\sigma^2_A = 0.001 rad^2$ as described in Section 6.1.2.3.

Figure 48 shows the behavior of localization error when mobile agent performs the typical path, graphically displayed in Figure 49.The plot compares errors when dead-reckoning, ToTal algorithm and Kalman filtering-based navigation methods are employed.

Notice that vertical magenta lines define steps in which the Kalman filter takes measures i. e. filtered position arise as a combination between

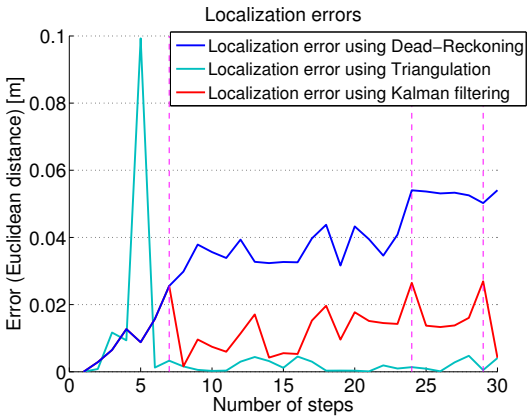


Figure 48: Behavior of triangulation error (Euclidean distance from actual position of the robot). Comparison between Dead-reckoning, ToTal triangulation algorithm estimation and Kalman-filtered estimated position. Parameters employed: $R_w = 0.01m^2$, $\sigma_A^2 = 0.001rad^2$, $threshold = 0.02$.

dead-reckoning and geometric triangulation. On the other hand, in steps not marked with magenta lines, the filter produces estimated location basing on kinematic model only; i. e. follows odometric state-space model re-initialized last iteration at which measures were available.

It can be noticed by Figure 48 that error behavior for dead-reckoning and triangulation corresponds to previously-motivated trends (Section 6.1.1 and 6.1.2.3 respectively).

As introduced in Section 3.5.5, the Kalman filter dispose for intermittent observations. This assumption has been done in order to render more robust the algorithm. Indeed it is reasonable to assume that in many practical cases (because of occlusions, limited-range wireless capabilities, errors in communications etc.), mobile agents does not dispose for three available beacons at any given time. Furthermore this assumption enable to overcome issues related to high-uncertain areas of the mission space described in Section 4.3.1. On the other hand, one of the main purposes of this work was to provide a high-rate localization system, capable of working in the entire mission space and overcoming triangulation unreliable regions. The combination between a high-rate location system such as odometry, together with reliable properties of geometric triangulation, lead to a system capable of combining both characteristics.

The approach we propose, envisage to:

- exploit dead-reckoning to provide high-rate estimated location;
- when perceived localization error overcome a given threshold, triangulation is used in order to improve estimated location.

Simulations have been conduced assuming the given threshold equal to $threshold = 0.02m$.
It is important to underline that in practical situations, agents in general do not dispose of perceived localization error, (on which the considered

threshold is based). Further works can improve this approach, for instance, by exploiting dead-reckoning variance.

By examining Figure 48, it is possible to notice that up to 7^{th} step, Kalman-estimated location coincides exactly with actual position computed through dead-reckoning, when initialized with initial robot location. The magenta vertical line shown at step 7 highlight the measure-update performed by the filter. This mean that location computed through geometric triangulation is exploited in Kalman filter equations. As shown by the figure, this leads to a reduction in the localization error of filtered location.

In following steps(8-23), Kalman estimation follows internal model only, thus filtered-position error trend resemble the dead-reckoning one. Notice that the two behavior are the same, except the fact that Kalman estimate has been re-initialized at the 7-th step thanks to measure availability. Analogous considerations can be conduced for succeeding steps. Indeed at iteration 24 and 29 novel measure are available, and the result is a reduction in localization error analogous to iteration 7.

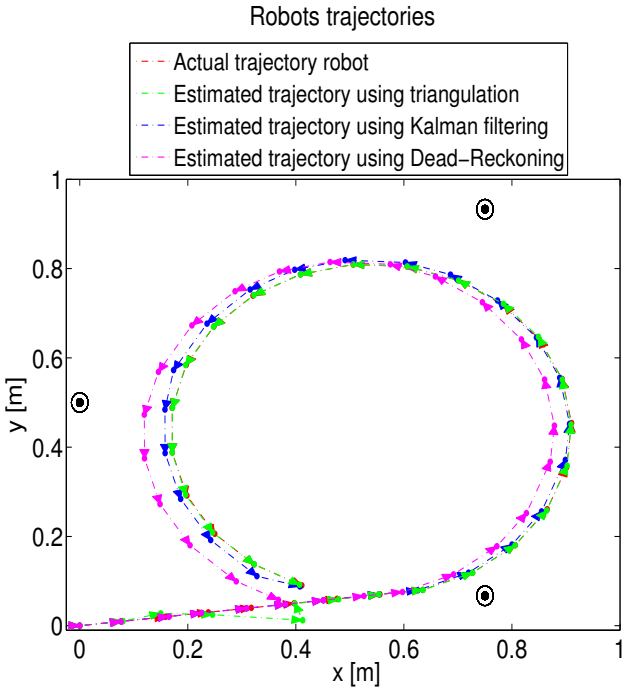


Figure 49: Actual and estimated trajectories. Comparison between Dead-reckoning, ToTal triangulation algorithm and Kalman estimate to merge the two estimate. Parameters employed: $R_w = 0.01m^2$, $\sigma_A^2 = 0.001rad^2$, $threshold = 0.02m^2$.

Figure 49 shows graphically the path performed by robot. It is important to underline agent's behavior at iteration 7: the robot is indeed located in the region corresponding to the circle joining the three beacons: estimated location through triangulation is therefore inaccurate and dead-reckoning is preferred.

Finally, some observations arise as conclusion of this section:

- geometric triangulation provide accuracy in estimated positions and does not suffer of integration-over-time drifts;
- although geometric triangulation provides reliable estimated locations in much regions of the configuration space, there exist some confined regions where this method is unacceptable;
- dead-reckoning provides simple and high-rate update locations estimates, however this method suffers from unbounded error integration that confer unreliability to estimated location;
- the approach we propose is capable of combining advantages of both methods, through apposite combination of both model and geometric triangulation.

6.2 BEACON DEPLOYMENT FOR LOCALIZATION

In the former part of this section are discussed advantages introduced by Variance map in deriving the quality of localization when we deal with a pre-existing beacon disposal in the configurations space. In particular, triangular patterns are described in detail and troubles arising when the number of beacons scales are discussed. Through simulations results is then presented the capability of the network in deploying new beacons according to equilateral triangle shapes . During the second part of the section, *Self-configuring minimal uncertain deployment* algorithm described in Section 4.3.2 is discussed and simulation results are presented. It is important to highlight that Variance maps plots considered in the following, represent the average variance between x-axis and y-axis.

6.2.1 *Variance map with equilateral-triangle grid*

Closed-form expressions for estimated locations uncertain provide a useful tool in order to understand quality of localization in various regions of the mission space. This kind of analysis is actually helped through Variance map graphical representations, that enable to graphically visualize the quality of localization when we deal with pre-existing beacons patterns. Indeed, thanks to Variance map, regions characterized by poor localization quality can easily be individuated and beacon disposal could then be improved.

As introduced in 4.3.1, triangle disposal for terns of beacon represent the optimal pattern when the aim consists in minimizing beacons deployment density ρ. Figure 50 presents numerically computed variance map when beacons are equally separated in x-axis and in y-axis of $d_{max} = 0.25m$.

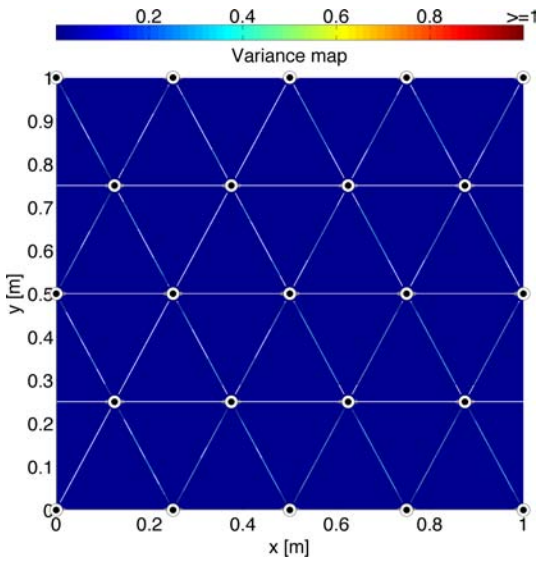


Figure 50: Variance map for triangle-shapes. The Variance map shown is the average between x-axis and y-axis.

Notice this beacons pattern does not contain redundancy, since each mobile agent is capable of sensing at least (and no more then) three beacons at any given time (i. e. because of sensing radius related constraints). As shown by the figure, the variance of estimated location remains lower than the threshold of $0.1m^2$ for most regions of the configurations space, except for locations corresponding to lines joining beacons. First of all it is interesting to notice that high-uncertain regions corresponding to circles joining terns of beacons do not appear in the presented map. This is a consequence of the fact that each target is always located inside the triangle formed by the beacons. Therefore, in absence of obstacles, Figure 50 represent both the minimal-uncertain and minimal-density beacons pattern.

However, in presence of obstacles or when the mission space cannot be considered of infinite size, beacons disposal presents redundancy and a beacon selection criteria has to be designed. This requirement arise since mobile agent could deal with more than three sensed beacons. Typical selection criteria often employed consist in choosing the three closest beacons to compute location.

Figure 51 show a triangle-pattern when the configurations space is limited in size, that is, when beacons are required to satisfy the d_{MAX} constraint at any given point of the mission space. Comparing Figure 50 with Figure 51 it can be noticed that the d_{MAX} constraints requires additional beacons at borders.

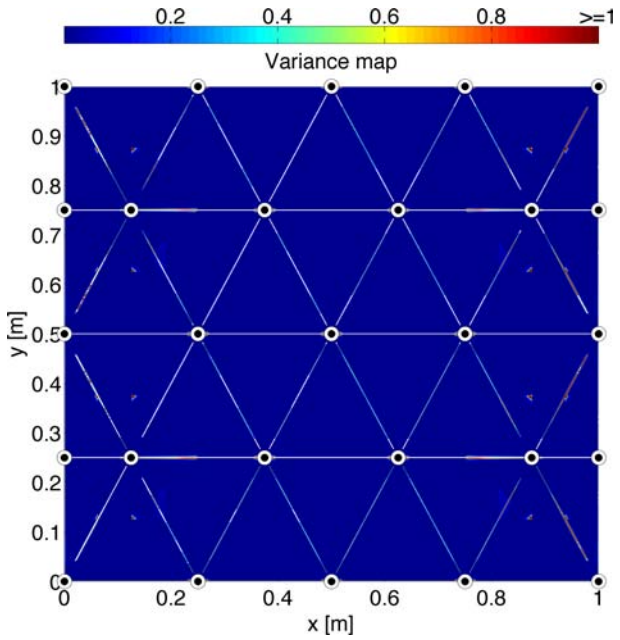


Figure 51: Variance map for triangle-shapes and borders effects. Variance in m^2.

As a consequence of redundancy at borders combined with sub-optimal selection criteria, the Variance map exhibit peculiar behavior in regions close to borders. As highlighted by the figure, new asymmetric peaks appears inside triangles and variance in lines joining couple of beacons increase.

In conclusion, observations conduced lead to infer that in presence of beacons redundancy, beacons-selection criteria plays a fundamental role in variance behavior.

6.2.2 *Self-configuring beacon deployment for equilateral-triangle grid*

In Section 4.3.2 has been described the self-configuring beacon deployment algorithm, capable of deploying new beacons in an adapting manner and following triangular patterns. The purpose of this section is to clarify, through simulations, how couples of beacons can cooperate in order to set up a complete localization beacons grid.

In Section 4.3.2 we have already noticed that each candidate location arise from the cooperation of a couple of neighbor nodes. Figure 52 clarify how the computation of a candidate location requires a couple of existing beacons to cooperate for its design. In particular in the simulation exhibited, nodes 1 and 3 cooperate to design the location for a new agent in such a manner to form an exact equilateral triangle.

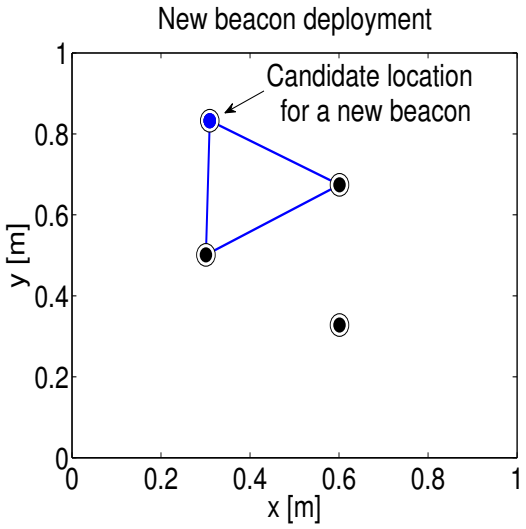
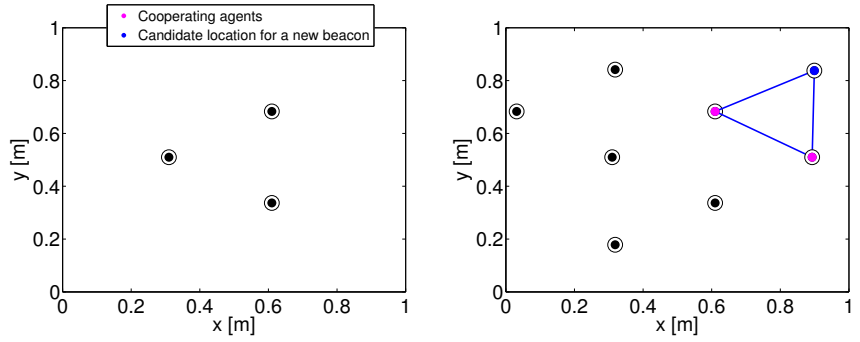


Figure 52: Candidate location for a new beacon arise from cooperation between a couple of pre-existing nodes.

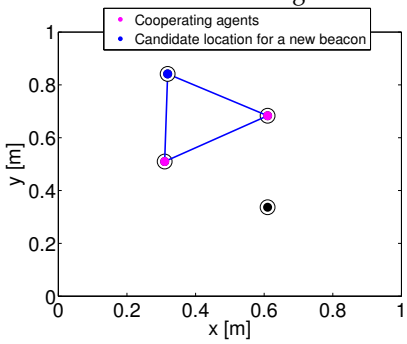
The beacon deployment process begins from couples of pre-existing nodes, as shown in Figure 53. In particular, the figure clarify the process of deployment for the entire beacons networks in such a manner to cover the whole mission space. The resulting deployment correspond exactly to equilateral-triangle shape presented in Section 6.2.1.

Finally is important to underline that the new beacon deployment process concludes with the transmission of the new designed location to robotic network for the effective deployment.

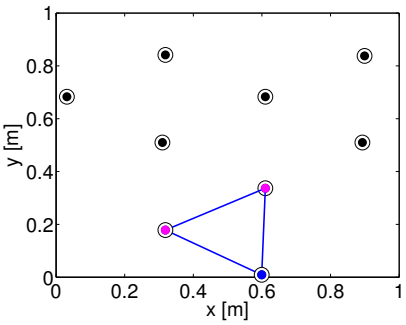


Initial beacons arrangement.

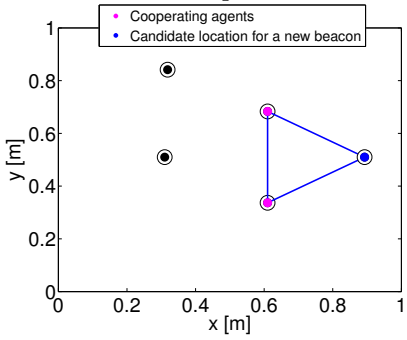
Step6.



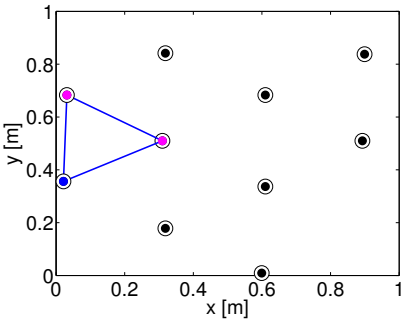
Step 2.



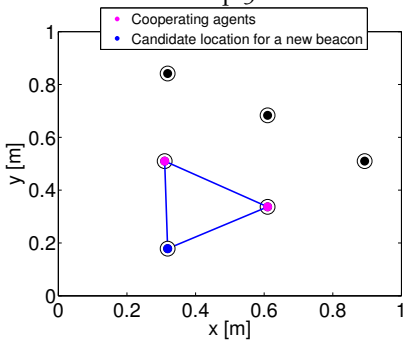
Step 7.



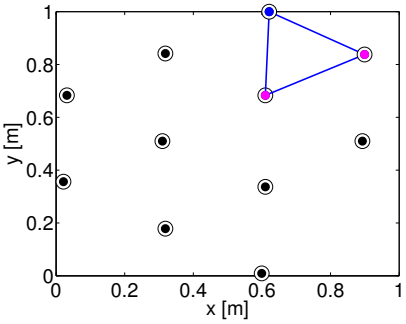
Step 3.



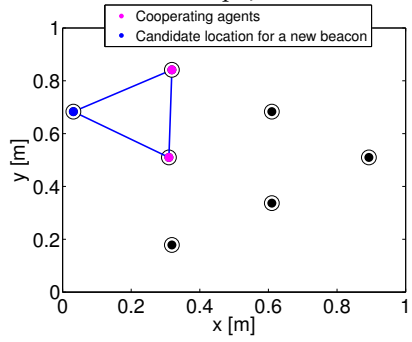
Step 8.



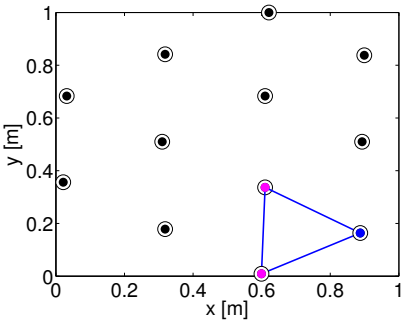
Step 4.



Step 9.



Step 5.



Step 10.

Figure 53: Beacons deployment process. Magenta colored beacons represents cooperating agents that compute novel beacon location. New candidate location is represented in blue.

6.2.3 *Self-configuring minimal uncertain deployment*

When the final objective consists in both minimizing beacons density ρ and minimizing beacons locations uncertain, triangle shapes have to be modified and the computed candidate location arise as the solution of an optimization problem. In this section, we propose simulations results when implementing Algorithm 4.
Figure 54 proposes a typical situation in which locating new beacons in correspondence of the vertex of the equilateral triangle gives rise to unacceptable uncertain in new beacon's location. Indeed, candidate location corresponding to vertex of equilateral triangle, correspond to a high-uncertain region since it lies on a line joining a couple of beacons.

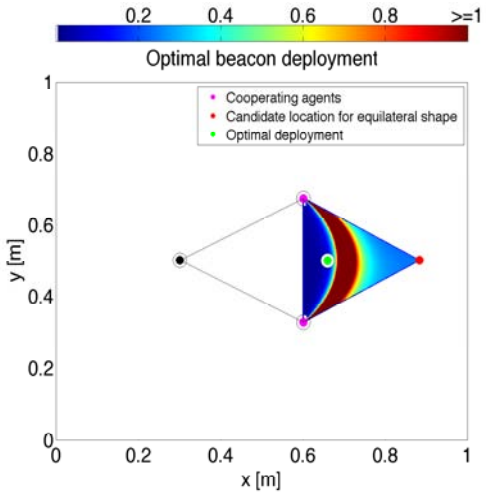


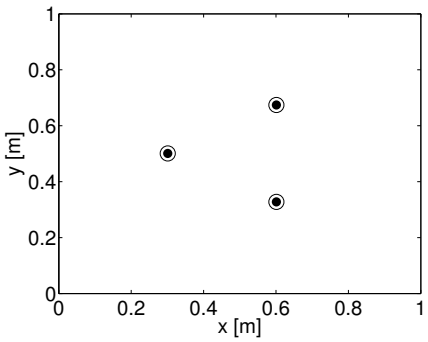
Figure 54: Minimal uncertain deployment. Ultimate candidate location is represented in green colour.

As highlighted by the figure, the optimization process enable to avoid this undesired behavior by modifying candidate location.
Figure 55 show the minimal uncertain deployment process when nodes employ Algorithm 4.
In order to lead the network to maintain a certain triangle-like structure, the optimization was performed weighing distances between candidate location and actual triplet. That is, following the algorithm:

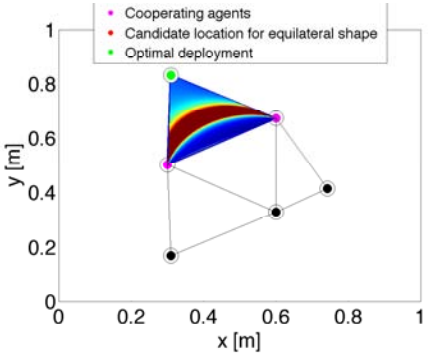
Algorithm 5 Iterative deployment algorithm weighted on distances

1) Send request to neighbors for their positions;
2) Compute acceptable positions $\overline{\Omega}$ respecting the d_{MAX} constraint;
3) Compute the variance map;
4) Solve the constrained optimization:
$$\min_{\overline{\Omega}} V(x,y) \cdot \mathrm{dist}\{(x,y),(x^*,y^*)\}^{-1};$$
5) Compute ultimate candidate position through average consensus with immediate neighbor.

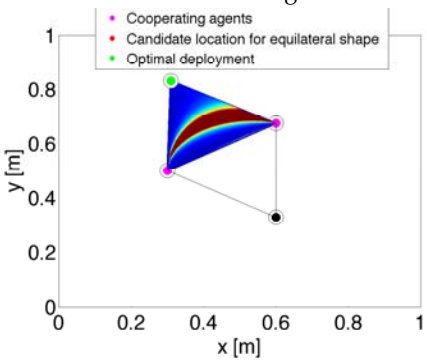
where (x^*, y^*) represent current beacon position.



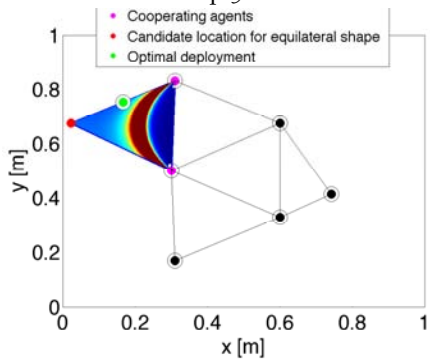
Initial beacons arrangement.



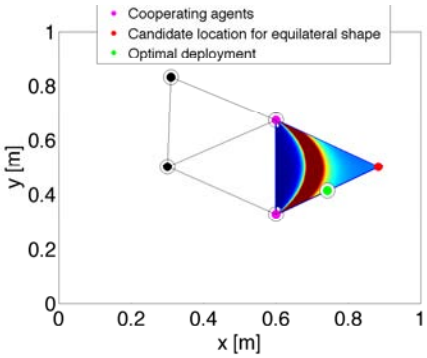
Step 5.



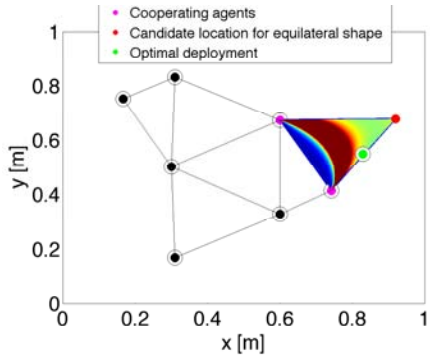
Step 2.



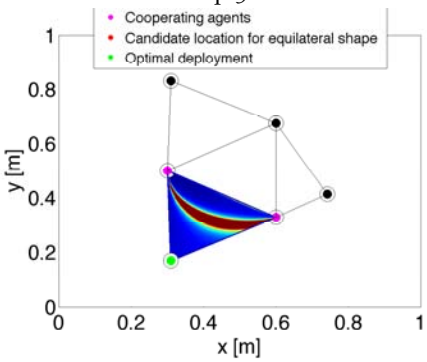
Step 6.



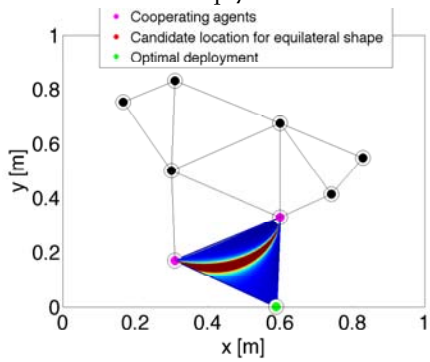
Step 3.



Step 7.



Step 4.



Step 8.

Figure 55: Optimal beacon deployment process. Notice that variance map plotted represents the average variance in x and y directions.

As can be noticed by the figure, when Variance map exhibit unacceptable variance values (recognizable by lighter colors), such as in Step 3, Step 6 and Step 7 the algorithm proposes a candidate location that is closer to cooperative nodes in order to improve localization quality. Finally it is important to underline that candidate location arise as solution of an optimization problem and then is modified through a consensus procedure.

6.3 COVERAGE CONTROL

The deployment algorithms described in Chapter 5 has been implemented in a simulation environment based on Matlab. The behavior of mobile agent located in the mission space will be analyzed in detail when the control strategies described in Chapter 5 are implemented.

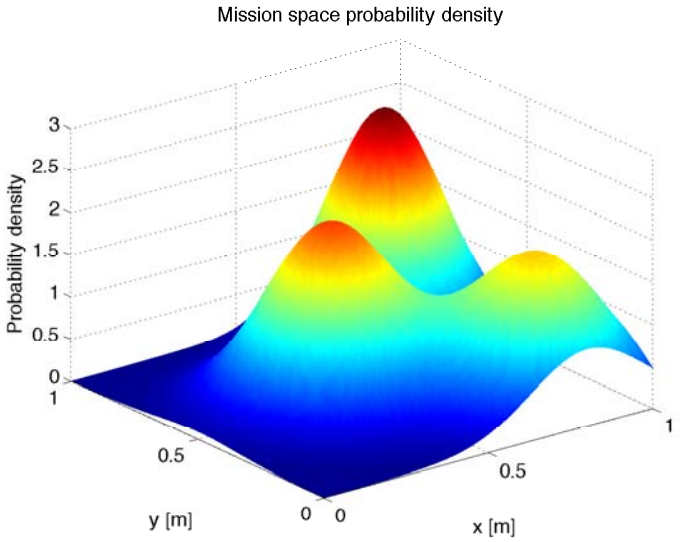


Figure 56: Graphical representation of the density function describing events.

As shown in Figure 56, a unit (meter) square mission space has been considered i. e. $\Omega = \{[0,1]x[0,1]\}m$. Recalling that the density function $R(x)$, described by Equation 31 models the frequency that specific random events takes place at x, in this context has been assumed:

$$R(x) = \frac{R_1(x) + R_2(x) + R_3(x)}{3}, \tag{55}$$

where

$$\begin{cases} R_1(x) & \sim & \mathcal{N}(\begin{bmatrix} 0.8 \\ 0.8 \end{bmatrix}, \begin{bmatrix} 0.02 & 0 \\ 0 & 0.02 \end{bmatrix}) \\ R_2(x) & \sim & \mathcal{N}(\begin{bmatrix} 0.4 \\ 0.5 \end{bmatrix}, \begin{bmatrix} 0.025 & 0 \\ 0 & 0.025 \end{bmatrix}) \\ R_3(x) & \sim & \mathcal{N}(\begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}, \begin{bmatrix} 0.03 & 0 \\ 0 & 0.03 \end{bmatrix}) \end{cases} \tag{56}$$

i. e. events to monitor takes place as modeled by the sum of three Gaussian density functions located at $P_1 := (0.8; 0.8)$, $P_2 := (0.4; 0.5)$ and $P_3 := (0.8; 0.2)$.
Notice that in general cases the density function $R(x)$ can be time-variant: $R(x) = R(t, x)$. In such cases Equation 55 models events taking place for a fixed time step (i. e. $t = t^*$). In the following, it is reasonable to assume that

$R(x,t)$ be slowing changing over time with respect to dynamics of mobile nodes.

Each mobile node is equipped with a sensor whose detection probability is modeled by:

$$p_i(x) = p_{0i} e^{-\lambda_i ||x - s_i||} \tag{57}$$

where $p_{0i} = 1$, $\lambda_i = 1 \cdot 10^{-4}$ for all agents considered.

6.3.1 *Simulation results on the distributed solution to the optimal coverage problem*

In the following, theoretical results presented in Section 5.2 will be simulated in a Matlab-based environment. Our purpose it to understand coverage quality and node behavior when agents dispose for local information only.

The sensing radius (36) considered for simulations is $D = 0.2$ m, as illustrated by dashed circles in Figure 57.
Figure 57 presents behavior of a couple of agents during the area-discovering process.

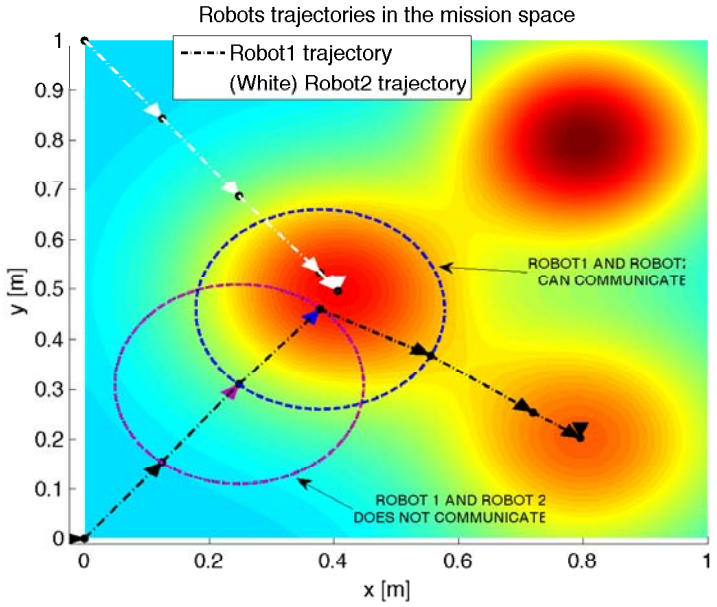


Figure 57: Area-discovering process by a team composed of a couple of robots.

It has been assumed that agents have no a priori knowledge about the function $R(x)$. Therefore, as discussed in Section 5.2, Equation 37 can be computed in a distributed manner, by exploiting local knowledge gathered through on-board sensors and the knowledge of neighbor sensing model density function (32).

Figure 58a-d presents several snapshots taken during the area-discovering process of Figure 57, representing local knowledge of nodes regarding the events density function $F(x)$ (defined in Equation 33) gathered through on-board sensors.

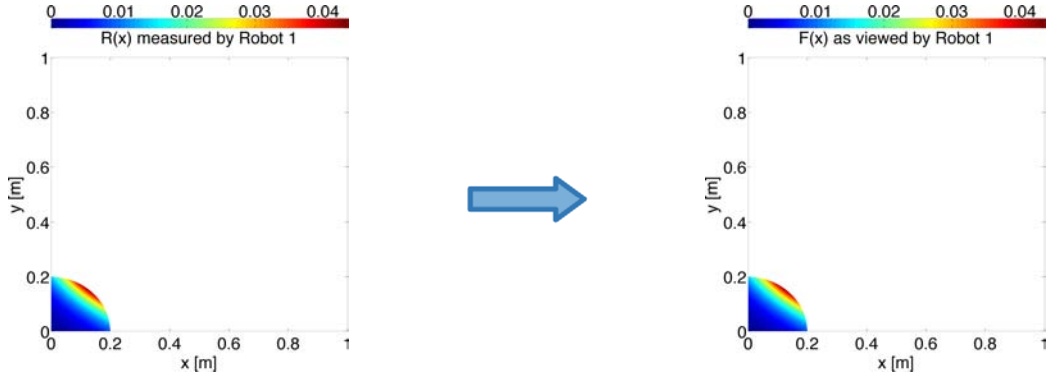
Moreover, Figure 58a-d describes how the estimated function $R(x)$ gets altered by the algorithm to compute the cost function $F(s)$ necessary to deduce optimal moving direction.

As can be observed by Figure 58a, 58b and 58c, during the initial phase of the area-discovering process, since relative distances between robots is over the sensing radius, $R(x)$ is not altered to compute $F(x)$. Therefore, since local $R(x)$ is monotonic the solution to the optimization problem conduces the mobile agent along the direction of maximum growth of the function $\int_{\Omega} R(x) \cdot p_{0i} e^{-\lambda_i ||x - s_i||} dx$.

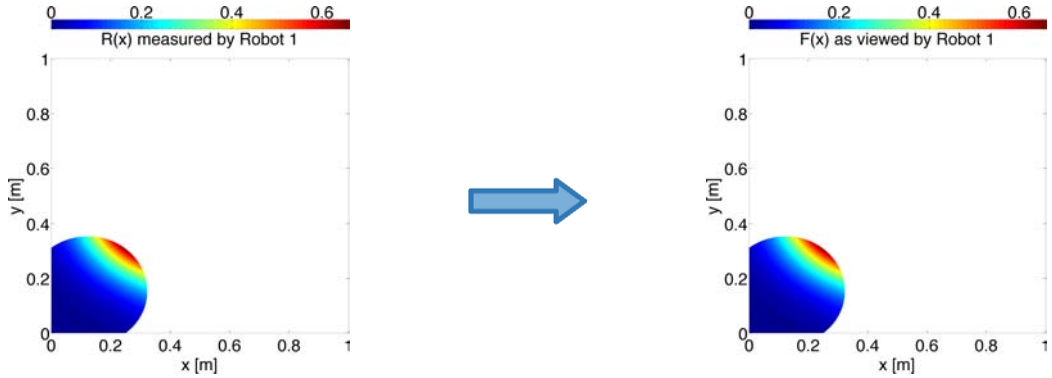
On the other hand, as shown in Figure 57d and 57e, when distance between the couple of nodes is below the sensing radius, estimated $R(x)$ get attenuated in order to model neighbor effective sensing. In mathematical terms this leads to an abatement in the amplitude of the initial peak of $R(x)$ (located at $(0.4, 0.5)$) that turns into a minimum for the informative function $F(x)$. Therefore agent 1 gets guided towards other peaks of $R(x)$.

Considerations just explained leads to a further and more general interpretation for $F(x)$: this function can be interpreted as an information function (defined in the whole configuration space) that models the informativeness that locations of the plane can lead whether visited.

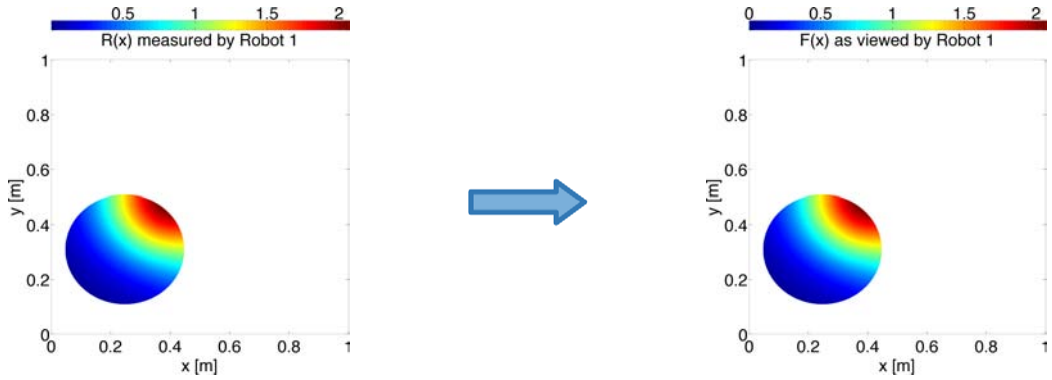
In general terms we can state that informative locations i. e. maximum for the function $R(x)$ become minimum for $F(x)$ whether other agents of the network are located in correspondence of this maximum.



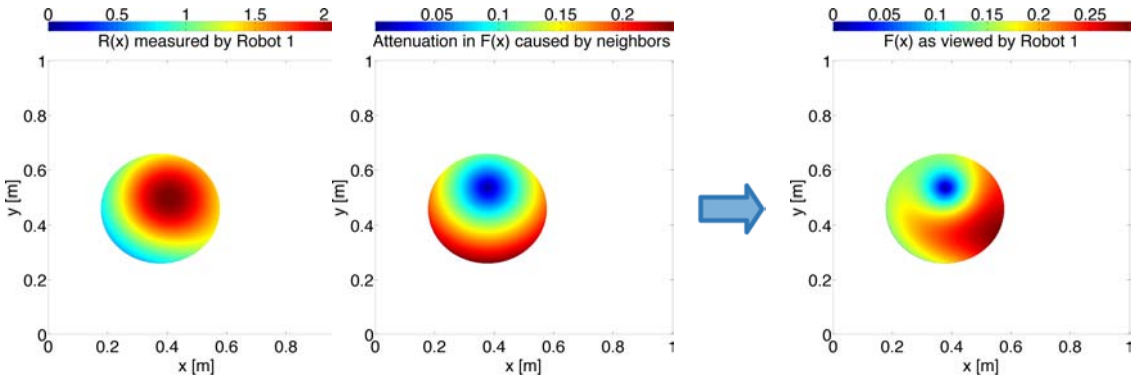
(a) Step 1.



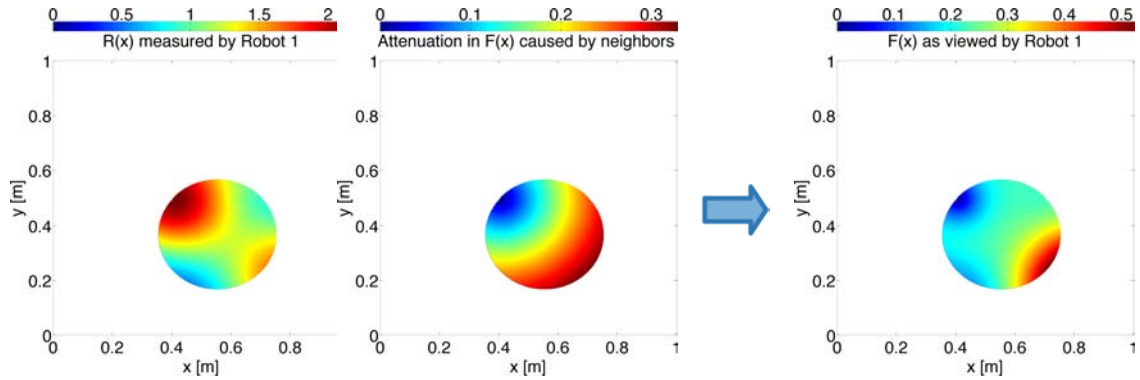
(b) Step 2.



(c) Step 3.



(d) Step 4. $R(x)$ has been reduced to produce $F(x)$ as an effect of nodes proximity.



(e) Step 5. $R(x)$ has been reduced to produce $F(x)$ as an effect of nodes proximity.

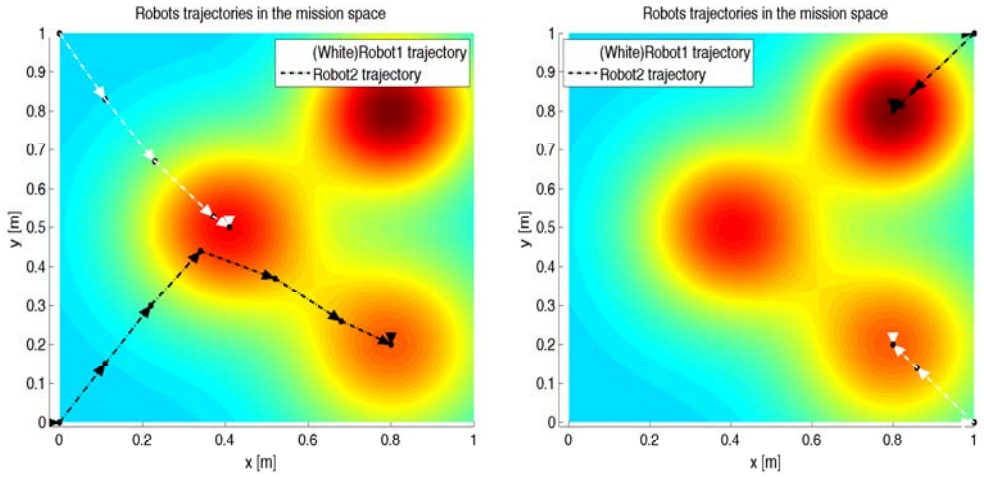
(f) Step 6.

Figure 56: Local knowledge $R(x)$ of Robot1, and local weight function computed $F(x)$.

6.3.2 *Distributed coverage and partitioning*

The comparison proposed in Figure 57 shows the strong dependence of nodes trajectories and convergence location from the initial positions of robotic agents. Indeed, as can be noticed from the figure, different initial locations for the couple of robots, lead to different convergence points although the mission space model remains unchanged. This aspect arises as a consequence of the exploitation of local information only; indeed the local knowledge of $R(x)$ possessed by agents does not permit the computation of global maximum, but solutions computed in general are local.

Figure 57a explains in detail this sub-optimal solution to the problem: both agents converges to a local maximum for the function $F(s)$ while the absolute maximum located at P_1 is neglected[1]. It is important to underline that this undesired behavior arise as a consequence of the lack in knowledge of the entire mission space.
Moreover, Figure 57b show the change in convergence points when initial locations are varied.



(a) Agents behavior when initial conditions are $(x,y_0)_{R1} = (0,0)m$ and $(x,y_0)_{R2} = (0,1)m$. (Same as Figure 57)

(b) Agents behavior when initial conditions are $(x,y_0)_{R1} = (1,0)m$ and $(x,y_0)_{R2} = (1,1)m$.

Figure 57: Agents trajectories are different when initial conditions are different.

Problems just explained motivated the work presented in Section 5.3.4: which main purpose consists in proposing improvements in the cost function Equation 33in order to provide a global knowledge of $R(x)$ to robotic network.
In this section, improvements introduced by the exploitation of the cost function Equation 42 are presented and discussed through simulations re-

1 $P_1 = (0.8; 0.8)m$

sults. For further completeness, we report Equation 42 in the following :

$$F_2(s,t) = \int_\Omega R(x)P(x,s)dx + \sum_{i=1}^{N} \int_{V_i} (1 - P_v(x))\rho(t)dx \,. \tag{58}$$

that is, the cost function to be maximized. It is important to mention parameters signification:

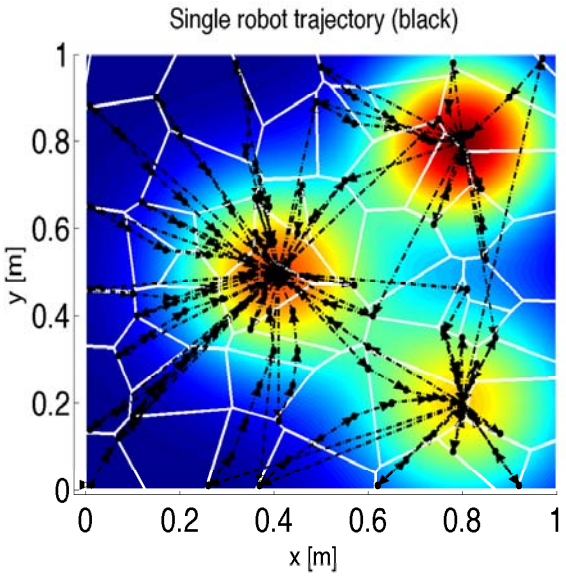
- $\{V_i\}_{i=1}^N$ is a Voronoi tessellation of the whole mission space, satisfying
$$\max\{dim(V_i)\} < dim(B_i) \quad \forall \quad i = 1, 2, ..N$$
i. e. each Voronoi partition can be completely measured by on-board sensors when agents is located on its centroid;
- $P_v(x) = \begin{cases} 1 & \text{if } x \text{ has been visited in the last } k \text{ iterations} \\ 0 & \text{otherwise} \end{cases}$
- $\rho(t) = t - k \quad .$

The modified cost function taking into account distances weight is

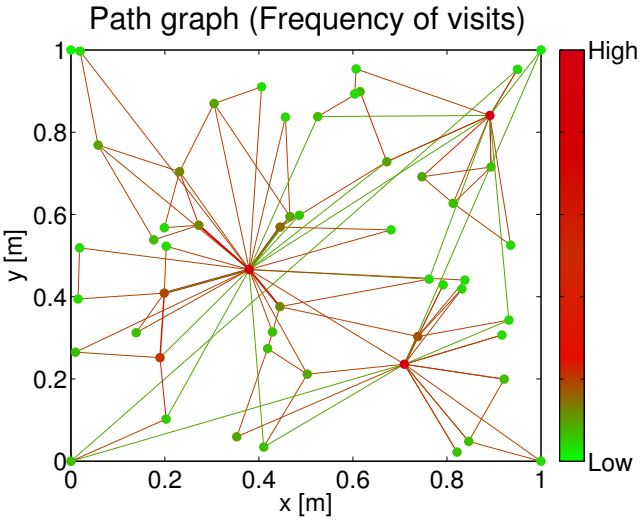
$$F_3(s,t) = \int_{\Omega_i} R(x)P_i(x,s)dx + \\ + \sum_{i=1}^{N} \int_{V_i} [1 - P_v(x, t-k)]\rho(x, t-k)dx \cdot \|z_i - z_j\|_{i\neq j}^{-1} \tag{59}$$

Moreover, in order to dynamically vary the value of the variable k, it has been chose to update k every time the agents begin visiting a new partition.

It is important to notice that a Voronoi tessellation has been used in this context, rather then an uniform discretization of the configurations space since it provides a suitable and robust manner to discretize non-regular environments.
Moreover the tessellation enables the network to dynamically update partitions whether new areas are discovered by agents. Another very important advantage introduced by Voronoi's technique is that centroids can be dynamically updated during the life of the network, optimizing shapes of the tessellation in such a manner to adapt their size according to the informativeness of the associated part of mission space.



(a)



(b)

Figure 58: (a) Single robot implementing Equation 58 and performing 1000 iterations. Notice that every Voronoi location have been visited almost once and that more rapidly changing locations have been visited more often than other areas. (b) Related graph.

In Figure 58 is shown the behavior of a single robot when implementing Equation 59. Notice that Equation 59 takes into account distances between centroids. The plot representing robot's paths in the mission space is enhanced by a graph representing frequency at which Voronoi cells are being visited; nodes represent Voronoi centroids while edges models sequences in visits. The mission space model is the same in (56).
Figure 58a clarifies the fact that the algorithm leads the robot to visit the whole mission space, through imposing the visit of every Voronoi cell. This leads the robot to build-up a comprehensive knowledge of the configuration space and in particular of its model $R(x)$.

The graph in Figure 58a graphically show the behavior just explained. Indeed, nodes and edges corresponding to regions with peaks in $R(x)$ results more frequently visited than remaining centroids of the configurations space. As a consequence, disposing of a complete knowledge about

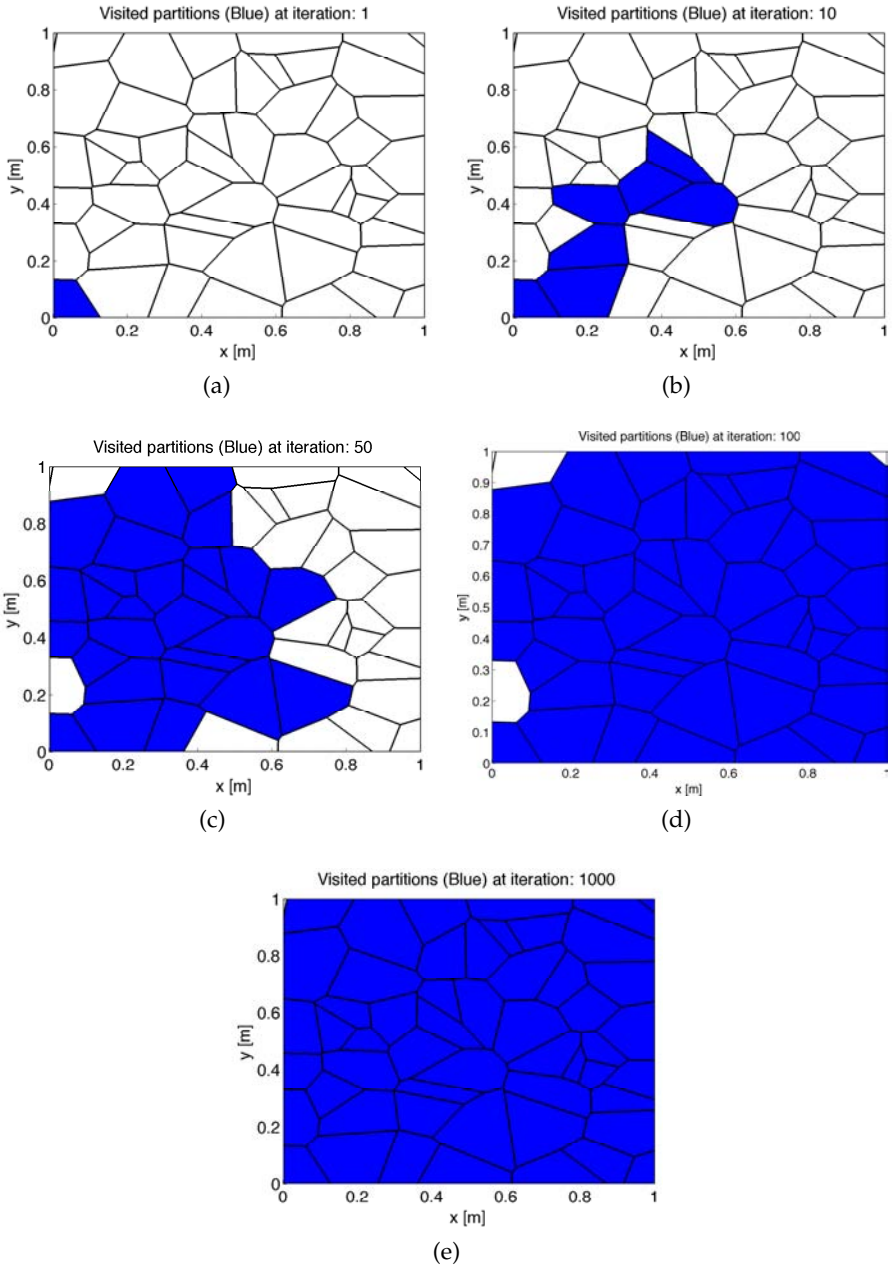


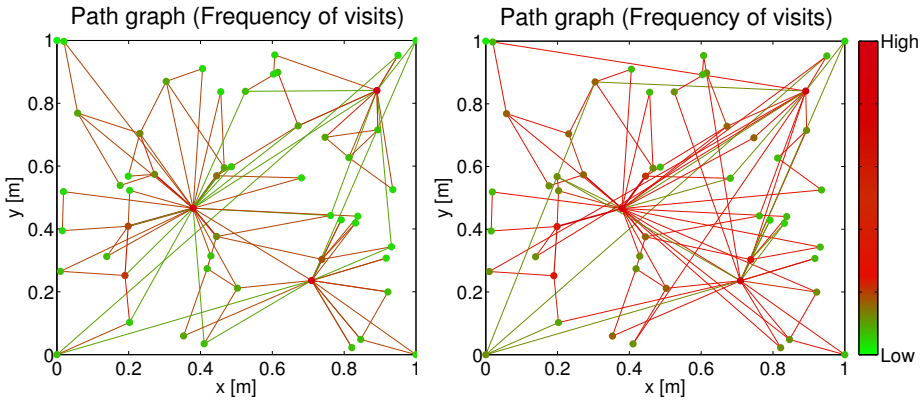
Figure 59: Visited partition when exploiting Equation 59.

the mission space, mobile agents are capable of detecting and reaching global maximum for $R(x)$. Another important aspect to notice is that high informative locations are visited more often than border regions. This behavior is motivated observing that once the robot moves to visit a new partition, then it moves climbing informativeness gradient so reaching

the closest maximum. Therefore we can state that high-informativeness regions are being monitored with higher frequencies.

Figure 59 show visited partitions over time by the robot when performing paths shown in Figure 58. It can be noticed by the figure that the control strategy in about 100 iterations leads the robot to visit almost all given partitions.

In order to validate effects introduced by weighting the cost function with distances that agents are required to travel in order to reach next Voronoi cell, the purpose is then to compare (58) and Equation 59.

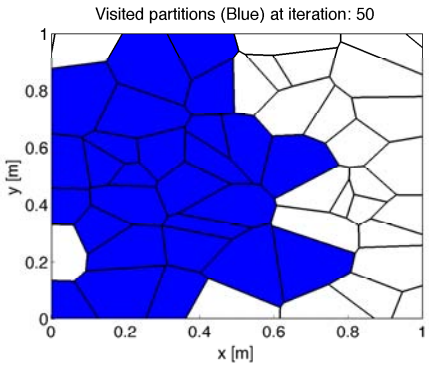


(a) Considering distances costs. (b) Neglecting distances costs.

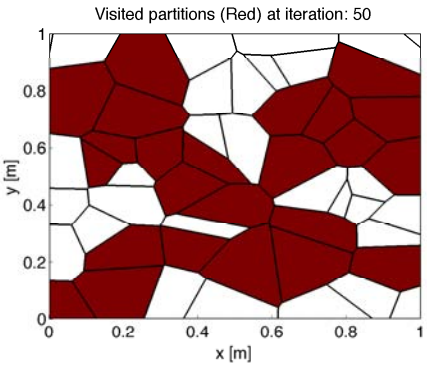
Figure 60: Graph comparison.

The comparison proposed by Figure 60 highlights advantages in exploiting distances costs in weight function. First of all, it can be noticed that graph exploiting (58) is highly connected: this is a consequence of the higher distances traveled by agents to reach far away locations in the mission space. Furthermore the comparison between colors of nodes highlights how Equation 59 enable the agent to visit more frequently high informativeness regions.

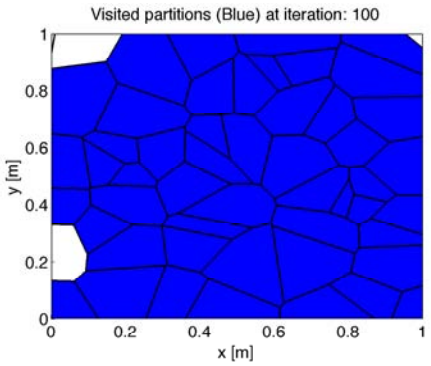
In Figure 61 evolution of visited partitions over time is compared when using Equation 59 and (58). From the comparison it is possible to notice that when exploiting (58), the robot is requested to travel longer distances in order to initially visit more interesting partitions. However, although more interesting partitions get visited earlier, the visiting procedure requires longer times to converge, since there exist a wasting in time to travel these distances.



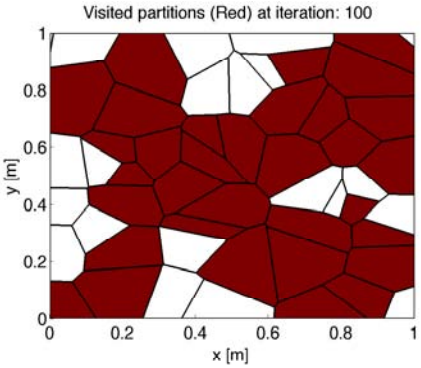
(a) Covered regions at iteration 50 when using (59).



(b) Covered regions at iteration 50 when using (58).



(c) Covered regions at iteration 100 when using (59).



(d) Covered regions at iteration 100 when using (58).

Figure 61: Comparison between (58) and (59) .

Part IV

CONCLUSIONS AND APPENDIX

The following chapters summarize the conclusions of the thesis an future works. Appendixes containing connected open-problems conclude the part.

7

CONCLUSIONS

Sensor networks are considered a key technology for the future. Their use has enlarged rapidly in last decade in view of a future that is increasingly automated. Scientific efforts are above all focusing on the development of smart agent networks, i. e. networks composed by devices able to autonomously sense environment and adapt network features upon measured quantities.

Multi-Robot systems are considered as a promising technology in environmental monitoring, since they permit to enhance capabilities of sensor networks by introducing mobility features to sensors employed. However, to achieve this automation, mobile robots must include a localization system. The growing possibilities enabled by robotic networks in the monitoring of natural phenomena, combined with flexibility features typical for distribute sensor networks, enable a number of potential improvements in quality of service provided in distributed area exploration. This work has aimed at exploring the fields of work of both sensor and robot networks through combining complementary features of these two systems.

This thesis addresses first the problem of locating mobile agents in unknown environments. In this kinds of frameworks, odometric techniques are the mainly used because of their low cost, high-update rate and accuracy in short paths. Unlikely, this methods resulted unacceptable in long paths because of an unbounded growth of time integration error with the traveled distance. Solutions based on anchor nodes, such as Geometric Triangulation, provides a promising way in order to improve localization quality in longer paths. However, since no a priori knowledge regarding the environment is possessed by the network and anchors cannot be located offline, agents are required to be equipped with mechanisms capable of deploying beacons that can serve for localization. In this terms, we have designed a localization system capable of self-deploying the static infrastructure composed by beacons. Characteristics of Geometric triangulation have been described in detail, and the method has been taken into account since it provides a promising method in anchor-based localization. Our contribution leads to derive a closed-form expression for the variance of estimated target location.

Considering the variance of target estimated location as a function of relative position of the target with respect to beacons, the mission space could be divided into disjoint regions where the variance is almost zero and regions where the variance presents a divergent behavior. In order to provide a robust, high precision, and high-updating rate system, we proposed an hybrid localization system based on Kalman filter in order to combine odometric readings together with estimated location through geometric

triangulation. Through simulation results we have probed the robustness of the method, even in typical noisy trajectories.

Deploying capabilities held by mobile agents give rise to the problem of understanding the optimal locations for new beacons in order to extend existing infrastructure together with improving the quality of the service provided. This problem has resulted to be NP-Hard. The distributed self-deployment algorithm proposed, is based on local cooperation between only two beacons nodes to design new candidate locations. The main feature of the system is the robustness guaranteed by the capability in computing actual location of the novel deployed beacon. This lead the system to work both in indoor and outdoor environments, since no global positioning systems are required.

The performance of sensor and robot systems in terms of quality of the service provided is sensitive to locations of agents in the mission space. In particular sensors are required to spread over the covered area and to aggregate in high-informativeness regions. This requirement, in context where the mission space model is changing over time, involve a dynamical deployment for senors. This feature can be met through the exploitation of mobile vehicles in areas rapidly changing over time, while fixed sensors can be employed in static areas. This kind of coverage control problems are usually solved through the exploitation of a parametric function modeling frequencies of events taking place in the configurations space. A novel distributed approach based on local optimization of cost function is proposed for coverage the control problem. Simulations motivate the introduction of a novel weigh function, capable of providing a global knowledge of the configurations space through an initialization process. Simulations results and a mathematical rigorous approach demonstrated the capability of the cost function in accurately modeling unknown areas of the mission space ad in leading agents to visit the entire mission space without requiring accurate path planning.

8

FUTURE WORK

The application suggests a number of interesting directions for further work. First of all, it would be interesting to analyze the performances of Geometric Triangulation in presence of obstacles, occlusions, or in presence of further sources of noise in measured bearing angles. In this sense, the derivation of a Variance map can be extended in presence of obstacles and occlusions. In order to overcome this kind of problems, we suggest to derive a closed form expression for uncertain of estimated position as a function of the power or the integrity of measured signals.

In Section 6.1.3 the Kalman filter with intermittent observations was implemented in order to improve localization quality. However, we have supposed to exploit observation whenever the perceived localization error overcomes a fixed threshold. It would be interesting to analyze in detail this problem; in particular to derive a mathematical model for the perceived localization error and propose alternative conditions.
One relevant assumption done in Chapter 3 was the absence of noise in estimated robot orientation. Motivated by many practical approaches where robot's orientation is corrupted by noise, future works would require to overcome this assumption. Another relevant issue to threat in localization would be the the extension of proposed algorithm in order to exploit redundancy in available beacons. The direction for future works we propose is based on an iterative application of Geometric Triangulation algorithm that exploits all combination without repetitions of terns of available beacons.

In the beacon deployment process, future works could involve an analysis of the quality of service provided as a function of beacons density ρ. Moreover, alternative beacons patterns could be considered, in comparison with the equilateral-triangle one proposed in this work.

The distributed coverage control problem was analyzed through the exploitation of a Voronoi tessellation in order to provide a global knowledge about the environment to the network. It would be interesting to analyze the problem through a dynamical tessellation of the mission space. In particular centroids can be located, and regions can be adapted in size so as to consider $R(x,t)$ as a key factor in the tessellation process. Moreover, further works motivated by practical applications could involve mission spaces model $R(x)$ rapidly changing over time: typical applications are, for instance, the tracking of mobile targets.

A

APPENDIX

A.1 COMMUNICATIONS COSTS

Besides sensing and collecting data from the mission space Ω, another important issue of a sensor network is to forward fields data to a basestation b, in order to provide real-time responsiveness of the network for external users. Most current sensor networks assume a two-layer structure [30], in which all sensor nodes form the first layer, and the second layer consists of a unique basestation, which is the common destination for all data.

In order to ensure reliable data forwarding, a wireless link must preserve a certain channel quality, which is measured by its Signal to Noise Ratio (SNR). To preserve a given SNR, the power of the transmitter have to be taken into account, that is a monotonically increasing function of the length of the current link. For example, for a single hop wireless link, the energy needed to transmit (E_{tx}) and receive (E_{rx}) a bit follows:

$$E_{tx} = \gamma + \beta d^n$$

$$E_{rx} = \eta$$

where a $\frac{1}{d^n}$ path loss has been assumed and γ is the energy/bit consumed by the transmitter electronics, β accounts for energy dissipated in the transmit op-amp and η is the energy/bit consumed by a node to receive the bit. In this terms, a relay that receives a bit and then transmits it a distance d is:

$$e(d) = \gamma + \beta d^n + \eta = (\gamma + \eta) + \beta d^n + \alpha + \beta d^n \quad .$$

Typical parameters for current radio transmission are $\gamma = 180nJ/bit$, $\beta = 10pJ/bit/m^2$ $n = 2$.

Considering a mobile sensor network with N sensors, each located at s_i, $i = 1, \ldots, N$, and a single basestation b located at $s_0 \in \mathbb{R}^2$. Let $r_i(s_i)$ be the data rate originated by the i-th sensor. Note that r_i is a function of s_i because the amount of data forwarded at i is determined by the number of events detected, that is on the sensor location. It is reasonable to assume that $r_i(s_i)$ is proportional to the frequency at which events are detected, i. e. :

$$r_i(s_i) = k \int_\Omega R(x) p_i(x) dx$$

where k [bit/detection] is the average amount of data forwarded when the sensor detects an event. Data originated at each sensor have to be finally delivered to the basestation. Let $c_i(s)$ be the total power consumed by the network in order to deliver a bit of data from sensor i to the basestation b.

In this terms, the optimal coverage problem can be revised by combining sensing coverage and communication costs:

$$\max_{s} \left\{ \omega_1 \int_{\Omega} R(x)P(x,s)dx - \omega_2 \sum_{i=1}^{N} r_i(s_i)c_i(s) \right\}$$

where ω_1, ω_2 are weighting factors. Denoting the communication cost by

$$G(s) = \sum_{i=1}^{N} r_i(s_i)c_i(s),$$

the overall objective function can be rewritten as:

$$J(s) = \omega_1 F(s) - \omega_2 G(s) \quad .$$

BIBLIOGRAPHY

[1] James C Alexander and John H Maddocks. On the kinematics of wheeled mobile robots. In *Autonomous robot vehicles*, pages 5–24. Springer, 1990.

[2] Johann Borenstein and Yoram Koren. A mobile platform for nursing robots. *Industrial Electronics, IEEE Transactions on*, (2):158–165, 1985.

[3] Nirupama Bulusu, John Heidemann, and Deborah Estrin. Adaptive beacon placement. In *Distributed Computing Systems, 2001. 21st International Conference on.*, pages 489–498. IEEE, 2001.

[4] Nirupama Bulusu, John Heidemann, Deborah Estrin, and Tommy Tran. Self-configuring localization systems: Design and experimental evaluation. *ACM Transactions on Embedded Computing Systems (TECS)*, 3(1):24–60, 2004.

[5] Mihaela Cardei and Jie Wu. Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer communications*, 29(4):413–420, 2006.

[6] Christos G Cassandras and Wei Li. Sensor networks and cooperative control. *European Journal of Control*, 11(4):436–463, 2005.

[7] Chee-Yee Chong and Srikanta P Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, 2003.

[8] Charles J Cohen and Frank V Koss. Comprehensive study of three-object triangulation. In *Applications in Optical Science and Engineering*, pages 95–106. International Society for Optics and Photonics, 1993.

[9] Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1327–1332. IEEE, 2002.

[10] Jorge Cortes, Sonia Martinez, and Francesco Bullo. Spatially-distributed coverage optimization and control with limited-range interactions. *ESAIM: Control, Optimisation and Calculus of Variations*, 11 (04):691–719, 2005.

[11] Lance Doherty, Laurent El Ghaoui, et al. Convex position estimation in wireless sensor networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1655–1663. IEEE, 2001.

[12] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: applications and algorithms. *SIAM review*, 41(4):637–676, 1999.

[13] Joseph W Durham, Ruggero Carli, Paolo Frasca, and Francesco Bullo. Discrete partitioning and coverage control for gossiping robots. *Robotics, IEEE Transactions on*, 28(2):364–378, 2012.

[14] Alon Efrat, Sariel Har-Peled, and Joseph SB Mitchell. Approximation algorithms for two optimal location problems in sensor networks. In *Broadband networks, 2005. BroadNets 2005. 2nd international conference on*, pages 714–723. IEEE, 2005.

[15] João Sena Esteves, Adriano Carvalho, and Carlos Couto. Generalized geometric triangulation algorithm for mobile robot absolute self-localization. In *Industrial Electronics, 2003. ISIE'03. 2003 IEEE International Symposium on*, volume 1, pages 346–351. IEEE, 2003.

[16] Josep Maria Font and Joaquim A Batlle. Dynamic triangulation for mobile robot localization using an angular state Kalman filter. In *Proceedings of the European Conference on Mobile Robots, Ancona*, pages 20–25, 2005.

[17] J Font Llagunes, Joaquim Agulló Batlle, et al. Mobile robot localization. Revisiting the triangulation methods. In *International Federation of Automatic Control (IFAC). Symposium on Robot Control*, volume 8, 2012.

[18] Josep M Font-Llagunes and Joaquim A Batlle. Consistent triangulation for mobile robot localization using discontinuous angular measurements. *Robotics and Autonomous Systems*, 57(9):931–942, 2009.

[19] Hatem Hmam. Mobile platform self-localization. In *Information, Decision and Control, 2007. IDC'07*, pages 242–247. IEEE, 2007.

[20] Peter F Hokayem, Dušan Stipanovic, and Mark W Spong. On persistent coverage control. In *Decision and Control, 2007 46th IEEE Conference on*, pages 6130–6135. IEEE, 2007.

[21] Volkan Isler. Placement and distributed deployment of sensor teams for triangulation based localization. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 3095–3100. IEEE, 2006.

[22] Volkan Isler, Sanjeev Khanna, John Spletzer, and Camillo J Taylor. Target tracking with distributed sensors: The focus of attention problem. *Computer Vision and Image Understanding*, 100(1):225–247, 2005.

[23] Thomas Judd and Robert W Levi. Dead reckoning navigational system using accelerometer to measure foot impacts, December 10 1996. US Patent 5,583,776.

[24] Alonzo Kelly. Linearized error propagation in odometry. *The International Journal of Robotics Research*, 23(2):179–218, 2004.

[25] Michael Kintner-Meyer. Opportunities of wireless sensors and controls for building operation. *Energy engineering*, 102(5):27–48, 2005.

[26] Andreas Krause, Carlos Guestrin, Anupam Gupta, and Jon Kleinberg. Near-optimal sensor placements: Maximizing information while minimizing communication cost. In *Proceedings of the 5th international conference on Information processing in sensor networks*, pages 2–10. ACM, 2006.

[27] John J Leonard and Hugh F Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *Robotics and Automation, IEEE Transactions on*, 7(3):376–382, 1991.

[28] Cornelius T Leondes. *Mechatronic Systems Techniques and Applications*, volume 2. CRC Press, 2000.

[29] Wei Li and Christos G Cassandras. Distributed cooperative coverage control of sensor networks. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 2542–2547. IEEE, 2005.

[30] Wei Li and Christos G Cassandras. A minimum-power wireless sensor network self-deployment scheme. In *Wireless Communications and Networking Conference, 2005 IEEE*, volume 3, pages 1897–1902. IEEE, 2005.

[31] Marcin Ligas. Simple solution to the three point resection problem. *Journal of Surveying Engineering*, 139(3):120–125, 2013.

[32] Sonia MartíNez and Francesco Bullo. Optimal sensor placement and motion coordination for target tracking. *Automatica*, 42(4):661–668, 2006.

[33] Vicente Martinez. Modelling of the flight dynamics of a quadrotor helicopter. *A MSc Thesis in Cranfield University*, 2007.

[34] Clare D McGillem and Theodore S Rappaport. Infra-red location system for navigation of autonomous vehicles. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1236–1238. IEEE, 1988.

[35] Clare D Mcgillem and Theodore S Rappaport. A beacon navigation method for autonomous vehicles. *Vehicular Technology, IEEE Transactions on*, 38(3):132–139, 1989.

[36] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. Exposure in wireless ad-hoc sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 139–150. ACM, 2001.

[37] Lyudmila Mihaylova, Tine Lefebvre, Herman Bruyninckx, Klaas Gadeyne, and Joris De Schutter. A comparison of decision making criteria and optimization methods for active robotic sensing. In *Numerical Methods and Applications*, pages 316–324. Springer, 2003.

[38] Atsuyuki Okabe and Atsuo Suzuki. Locational optimization problems solved through Voronoi diagrams. *European Journal of Operational Research*, 98(3):445–456, 1997.

[39] Vincent Pierlot and Marc Van Droogenbroeck. A new three object triangulation algorithm for mobile robot positioning. 2014.

[40] Bruno Sinopoli, Luca Schenato, Massimo Franceschetti, Kameshwar Poolla, Michael I Jordan, and Shankar S Sastry. Kalman filtering with intermittent observations. *Automatic Control, IEEE Transactions on*, 49 (9):1453–1464, 2004.

[41] John R Spletzer and Camillo J Taylor. Dynamic sensor planning and control for optimally tracking targets. *The International Journal of Robotics Research*, 22(1):7–20, 2003.

[42] Robert Szewczyk, Eric Osterweil, Joseph Polastre, Michael Hamilton, Alan Mainwaring, and Deborah Estrin. Habitat monitoring with sensor networks. *Communications of the ACM*, 47(6):34–40, 2004.

[43] Onur Tekdas and Volkan Isler. Sensor placement for triangulation-based localization. *Automation Science and Engineering, IEEE Transactions on*, 7(3):681–685, 2010.

[44] Yojiro Tonouchi, Takashi Tsubouchi, and Suguru Arimoto. Fusion of dead-reckoned positions with a workspace model for a mobile robot by bayesian inference. In *Intelligent Robots and Systems' 94.'Advanced Robotic Systems and the Real World', IROS'94. Proceedings of the IEEE/RSJ/GI International Conference on*, volume 2, pages 1347–1354. IEEE, 1994.

[45] SJ Wan, SK Michael Wong, and P Prusinkiewicz. An algorithm for multidimensional data clustering. *ACM Transactions on Mathematical Software (TOMS)*, 14(2):153–162, 1988.

[46] C Ming Wang. Location estimation and uncertainty analysis for mobile robots. In *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, pages 1231–1235. IEEE, 1988.

[47] Yue Wang and Islam I Hussein. Awareness coverage control over large-scale domains with intermittent communications. *Automatic Control, IEEE Transactions on*, 55(8):1850–1859, 2010.

[48] Greg Welch and Gary Bishop. An introduction to the Kalman filter, 1995.

[49] Kuo-Lung Wu and Miin-Shen Yang. Alternative c-means clustering algorithms. *Pattern recognition*, 35(10):2267–2278, 2002.

[50] Tjalling J Ypma. Historical development of the Newton-Raphson method. *SIAM review*, 37(4):531–551, 1995.

[51] Minyi Zhong and Christos G Cassandras. Distributed coverage control and data collection with mobile sensor networks. *Automatic Control, IEEE Transactions on*, 56(10):2445–2455, 2011.

[52] Yi Zou and Krishnendu Chakrabarty. Sensor deployment and target localization based on virtual forces. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 2, pages 1293–1303. IEEE, 2003.