Université de Liège
Faculté des Sciences Appliquées
Département d'Électricité, Électronique et Informatique
Institut Montefiore

Design, performance analysis, and implementation of a positioning system for autonomous mobile robots

Vincent Pierlot

Thèse de doctorat présentée en vue de l'obtention du grade de
Docteur en Sciences de l'Ingénieur (électricité et électronique)

Année Académique 2012-2013

Marc	Van Droogenbroeck	*Promoteur*
Bernard	Boigelot	*Examinateur*
Josep Maria	Font Llagunes	*Examinateur*
João Miguel	Sena Esteves	*Examinateur*
Olivier	Gillieaux	*Examinateur*
Benoît	Vanderheyden	*Président*

Abstract

Positioning is a fundamental issue in mobile robot applications, and it can be achieved in multiple ways. Among these methods, triangulation based on angle measurements is widely used, robust, and flexible. In this thesis, we present an original beacon-based angle measurement system, an original triangulation algorithm, and a calibration method, which are parts of an absolute robot positioning system in the 2D plane. Also, we develop a theoretical model, useful for evaluating the performance of our system.

In the first part, we present the hardware system, named BeAMS, which introduces several innovations. A simple infrared receiver is the main sensor for the angle measurements, and the beacons are common infrared LEDs emitting an On-Off Keying signal containing the beacon ID. Furthermore, the system does not require an additional synchronization channel between the beacons and the robot. BeAMS introduces a new mechanism to measure angles: it detects a beacon when it enters and leaves an angular window. This allows the sensor to analyze the temporal evolution of the received signal inside the angular window. In our case, this feature is used to code the beacon ID. Then, a theoretical framework for a thorough performance analysis of BeAMS is provided. We establish the upper bound of the variance and its exact evolution as a function of the angular window. Finally, we validate our theory by means of simulated and experimental results.

The second part of the thesis is concerned with triangulation algorithms. Most triangulation algorithms proposed so far have major limitations. For example, some of them need a particular beacon ordering, have blind spots, or only work within the triangle defined by the three beacons. More reliable methods exist, but they have an increasing complexity or they require to handle certain spatial arrangements separately. Therefore, we have designed our own triangulation algorithm, named ToTal, that natively works in the whole plane, and for any beacon ordering. We also provide a comprehensive comparison between other algorithms, and benchmarks show that our algorithm is faster and simpler than similar algorithms. In addition to its inherent efficiency, our algorithm provides a useful and unique reliability measure, assessable anywhere in the plane, which can be used to identify pathological cases, or as a validation gate in data fusion algorithms.

Finally, in the last part, we concentrate on the biases that affect the angle measurements. We show that there are four sources of errors (or biases) resulting in inaccuracies in the computed positions. Then, we establish a model of these errors, and we propose a complete calibration procedure in order to reduce the final bias. Based on the results obtained with our calibration setup, the angular RMS error of BeAMS has been evaluated to $0.4\,deg$ without calibration, and to $0.27\,deg$, after the calibration procedure. Even for the uncalibrated hardware, BeAMS has a better performance than other prototypes found in the literature and, when the system is calibrated, BeAMS is close to state of the art commercial systems.

Résumé

Le positionnement constitue un problème fondamental dans les applications de robots mobiles. Il peut être réalisé selon plusieurs méthodes. Parmi celles-ci, la triangulation basée sur des mesures d'angles est largement utilisée, car robuste et flexible. Dans cette thèse, nous présentons un système de mesure d'angles original reposant sur des balises, un algorithme de triangulation original et un procédé de calibration qui font partie d'un système de positionnement absolu dans le plan 2D. En outre, nous développons un cadre théorique utile afin d'évaluer les performances de notre système.

Dans la première partie, nous présentons le système matériel, nommé BeAMS, lequel introduit plusieurs innovations. Un simple récepteur infrarouge constitue le capteur principal pour les mesures d'angles, les balises sont de simples diodes infrarouges émettant un signal tout ou rien contenant un identifiant. En outre, le système ne nécessite aucun canal de synchronisation entre les balises et le robot. BeAMS introduit aussi un nouveau mécanisme pour mesurer les angles: il détecte une balise quand il entre et sort d'une fenêtre angulaire. Ceci permet au capteur d'analyser l'évolution temporelle du signal à l'intérieur de la fenêtre angulaire. Dans notre cas, ce mécanisme est utilisé pour coder l'identifiant de la balise. Ensuite, nous développons un cadre théorique pour une étude approfondie des performances de BeAMS. Nous établissons la limite supérieure de la variance ainsi que son évolution exacte en fonction de la fenêtre angulaire. Enfin, nous validons notre théorie au moyen de simulations et de résultats expérimentaux.

La deuxième partie de la thèse porte sur les algorithmes de triangulation. La majorité des algorithmes de triangulation proposés jusqu'ici présentent des limites importantes. Par exemple, certains d'entre eux nécessitent un agencement particulier des balises ou fonctionnent seulement à l'intérieur du triangle défini par les trois balises. Des méthodes plus fiables existent, mais celles-ci présentent une complexité accrue ou impliquent la gestion de certains agencements spatiaux séparément. Par conséquent, nous avons conçu notre propre algorithme de triangulation, nommé ToTal, qui fonctionne nativement dans tout le plan et pour tout agencement de balises. Nous fournissons également une comparaison exhaustive avec d'autres algorithmes et des tests montrent que notre algorithme est plus rapide et plus simple que ses concurrents. En plus de son efficacité inhérente, notre algorithme fournit une mesure de fiabilité unique et très utile, évaluable n'importe où dans le plan, et qui peut être utilisée soit pour identifier des cas pathologiques, soit en tant que seuil de validation pour les algorithmes de fusion de données.

Enfin, dans la dernière partie, nous nous concentrons sur les erreurs qui affectent les mesures d'angles. Nous montrons qu'il existe quatre sources d'erreurs (ou biais) entrainant des imprécisions sur les positions calculées. Ensuite, nous établissons un modèle de ces erreurs et proposons une procédure de calibration afin de réduire l'erreur finale. Sur base des résultats obtenus avec notre dispositif de calibration, l'erreur quadratique moyenne affectant les angles fournis par BeAMS a été évaluée à $0.4\,deg$ sans calibration, et à $0.27\,deg$, après calibration. Même pour la version non calibrée, BeAMS a une meilleure performance que les autres prototypes décrits dans la littérature et, lorsque le système est calibré, BeAMS est proche des systèmes commerciaux définissant l'état de l'art.

Acknowledgments

First, I would like to sincerely thank my supervisor, Marc Van Droogenbroeck, for the confidence he has placed in me, his encouragement, and his pertinent suggestions. I thank also everyone who reads this thesis, and in particular the members of the jury.

I am also grateful to Bernard Boigelot for the EUROBOT project he started at the Montefiore institute. This project has been an extraordinary opportunity to apply my ideas to real robots, to work with a great team, and to stay in touch with electronics, mechanics, and many other interesting fields.

I would like to thank Sébastien Piérard, who has always helped me with various informatics fields, especially with C programming and Latex/Lyx editing. I also appreciate the so numerous and helpful conversations I had with him. In the same way, I thank Jean-François Degbomont for the numerous interesting discussions we had during lunch time. Thanks also to Guy Lejeune for the mate breaks and lively discussions.

I would like to thank Olivier Barnich, Sébastien Piérard, and Antoine Lejeune for collaborating with me over the past years. I also thank the students I have supervised throughout their master thesis for the good times spent with them, in particular Laurent Sibila, Pierre Tassin, Maxime Urbin-Choffray, and Thi Tuyet Minh Nguyen.

I also thank the members of the Montefiore institute for the good atmosphere. In particular, I thank the other members of the *eurobot Montefiore team* for the very good times I spent with them. I think about Guy Lejeune, Etienne Michel, Stéphane Lens, Maxime Urbin-Choffray, and Matthieu Remacle.

Thanks to Pascal Harmeling and Angel Calderon Jimenez for their help in the design of the hardware components.

Thanks also to my family and my friends for their encouragements. A special thanks to Aline and Marylou for reading over parts of this thesis.

And last but not least, I thank Alexandra for always supporting me and encouraging me during the last years of my thesis.

List of Symbols

$f_X(x)$	probability density function of the random variable X
$f_{XY}(x, y)$	joint probability density function of the random variables X and Y
$U_{(a,b)}(x)$	uniform probability density function between a and b
$\delta(x)$	DIRAC delta function
$E\{X\}$, μ_X	expectation of the random variable X
$var\{X\}$, σ_X^2	variance of the random variable X
$E\{X, Y\}$	joint expectation of the random variables X and Y
$C\{X, Y\}$	covariance of the random variables X and Y
B_i	beacon i
$\{x_i, y_i\}$	coordinates of the beacon i
ϕ_i	angle for beacon i
ϕ_{ij}	bearing angle between beacons B_i and B_j
T_{ij}	$\cot(\phi_{ij})$
$\mathcal{C}_{ij}$	circle passing through B_i, B_j, and the robot position
R_{ij}	radius of the circle $\mathcal{C}_{ij}$
$\{x_{ij}, y_{ij}\}$	coordinates of the center of $\mathcal{C}_{ij}$
$\{x_R, y_R\}$	coordinates of the robot position
θ, θ_R	orientation of the robot
$P_{IR}(\phi)$	infrared power collected at the receiver
P_{th}	infrared power threshold
Φ_r	random variable of the first rising edge
Φ_f	random variable of the last falling edge
Φ_b	random variable of the beacon angle
Φ_w	random variable of the angular window
T_b	duration of a bit
N_0	number of zeros in a code
N_1	number of ones in a code
N_b	number of bits in a code
N_c	number of codes
C_i	code number i
ω	angular speed of the turret
T_0	OFF duration
ϕ_0	OFF angle
p_0	probability of 0 symbols
p_1	probability of 1 symbols

Abbreviations

AGC	Automatic Gain Control
AGV	Autonomous Guided Vehicle
CCW	Counterclockwise
CW	Clockwise
EKF	Extended Kalman Filter
GDOP	Geometric Dilution of Precision
IR	Infrared
OOK	On-Off Keying
PDF	Probability Density Function
RMS	Root Mean Square
SNR	Signal to Noise Ratio
SVD	Singular Value Decomposition
BeAMS	BEacon Angle Measurement System
ToTal	Three Object Triangulation ALgorithm

Contents

Chapter 1

Introduction

Mobile robot positioning

Over the past years, the use of mobile robots has increased in various fields. Most of the time, they are used to transport materials in workstations, manufacturing industry, warehouses, harbors, airports, etc. These mobile robots are generally named "Autonomous Guided Vehicles" (AGVs). In order to be totally autonomous, navigate, avoid obstacles, and execute their actions correctly, mobile robots need some form of positioning, which is the process of determining the robot position in a given reference frame. Therefore, positioning is a crucial issue and an essential component in mobile robot applications. Depending on the authors, the concept of positioning may also includes the computation of the robot orientation. In this thesis, positioning refers to the process of determining both the position and orientation (pose) in the 2D plane.

Some fundamental papers, such as the ones of Betke and Gurvits [13], Borenstein *et al.* [15], Briechle and Hanebeck [18], Font-Llagunes and Batlle [34], Hu and Gu [43], Shimshoni [75], discuss mobile robot positioning. In particular, Betke and Gurvits [13] and Esteves *et al.* [32] highlight that sensory feedback is essential in order to position the robot in its environment. Some surveys (see Borenstein *et al.* [16], Demetriou [24], Gu *et al.* [38], or Muthukrishnan [60]) discuss several techniques used for positioning: odometry, inertial navigation, magnetic compasses, active beacons, natural landmark navigation, map-based positioning, and vision-based positioning.

We can identify two main families encompassing these methods [15]: (1) relative positioning (or dead-reckoning), and (2) absolute positioning (or reference-based). Relative positioning is the process of determining the pose with respect to its last estimate by using internal measurements, while absolute positioning uses measurements with respect to some references in the environment. Techniques belonging to the first family mainly operate by odometry, which consists of counting the number of wheel revolutions (*e.g.* with optical encoders) and integrating them to compute the offset from a known position. Relative positioning based on odometry is accurate for small offsets, but can lead to an increasing drift resulting from the unbounded accumulation of errors over time (due to the integration step, uncertainty about the wheelbase, wheel slippage, etc). Another important relative positioning technique is the inertial navigation, which is based on gyroscopes (measuring the angular velocity) and accelerometers (measuring the linear acceleration). Therefore, the measurements have to be integrated to compute the position, and as a result, errors accumulate over time. As for the

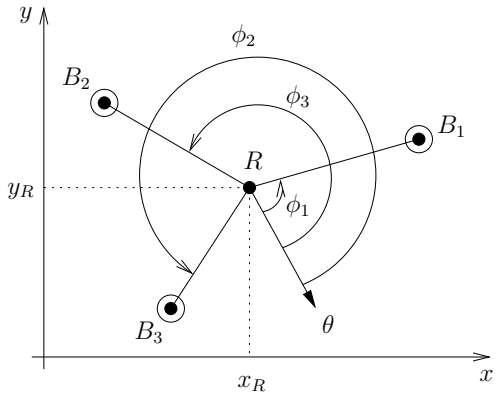

Figure 1.1: Triangulation setup in the 2D plane. R denotes the robot. B_i are the beacons. ϕ_i are the angle measurements for B_i, relative to the robot reference orientation θ. These angles may be used by a triangulation algorithm in order to compute the robot position $\{x_R, y_R\}$ and orientation θ.

odometry, inertial sensors have the advantage to be self-contained, to the contrary of absolute positioning. Also, inertial sensors do not need wheels to estimate the pose and therefore, they are used in applications such as submarines, ships, aircrafts, spacecrafts, etc. A good survey about inertial sensors and applications can be found in the work of Borenstein *et al.* [15]. Since relative positioning techniques can lead to an increasing drift, an absolute positioning system is thus generally required to recalibrate the position of the robot periodically (in order to maintain the positioning error to a low level). This is why both methods are essential and complementary to each other [5, 13, 17]. These two informations are generally combined in a Kalman filter or other data fusion algorithm [22, 33, 44]. For example, in [37], odometry with a gyroscope is combined with GPS data. Odometry is also merged with range measurements from ultrasound sensors in [48, 87], with angle measurements from a laser scanner in [6, 33, 34, 74], with angle measurements from a rotating camera in [26], and with range and angle measurements from a panoramic camera in [44].

As explained by Borenstein *et al.* [16], no universal indoor positioning system exists, contrasting with the widespread use of GPS for outdoor applications. This explains the large variety of existing systems depending on the target application and constraints such as cost, accuracy, available volume, coverage area, and usable technologies. Most absolute positioning techniques rely on beacons, whose locations are known, and perform positioning by triangulation (see Figure 1.1) or trilateration with respect to these beacons. Note that, most of the time, the robot evolves on a 2D plane, and therefore, it is characterized by its position and its orientation. The combination of the position and the orientation is generally termed *pose*. In the context of absolute positioning, a beacon is a discernible object in the environment, which may be natural or artificial, passive or active. Discernible means that it contrasts with the surrounding environment. Natural beacons are generally termed landmarks and are associated to particular geometrical shapes of the environment (*e.g.* horizontal or vertical lines, doors, stairs, etc). Artificial means that the environment has been modified on purpose to increase the contrast at a particular location. Active means that the beacon needs a source of energy (*e.g.* lights), to the contrary of passive beacons (*e.g.* retroreflectors).

Triangulation is the process of determining the robot pose (position and orientation) based

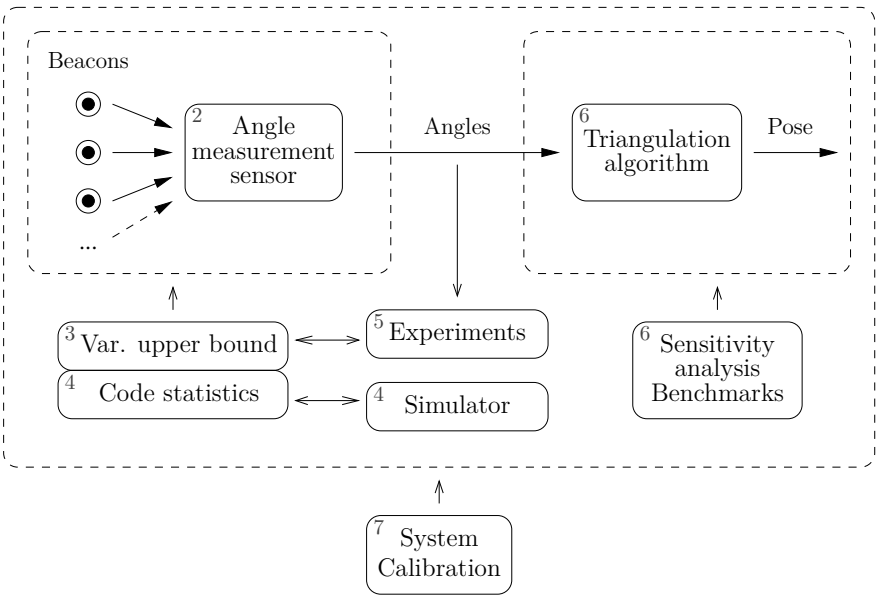

Figure 1.2: Framework of the thesis. Each element of the scheme is a part of an absolute robot positioning system in the 2D plane. The corresponding chapter is indicated at the upper left corner of each box.

on angle measurements (see Figure 1.1). This contrasts with the *trilateration* technique which determines the robot position by measuring distances from the robot to known locations [83]. While the techniques differ in principle, it has been shown by Niculescu and Nath [61] that any triangulation problems of size n (n measured angles) can be reduced to a trilateration problem of size $\binom{n}{2}$, which requires the computing of the intersection of at least three circles in the 2D plane. Because of the availability of angle measurement systems, its robustness, accuracy, and flexibility, triangulation with active beacons has emerged as a widely used, robust, accurate, and flexible technique for mobile robots [30, 75]. Another advantage of triangulation versus trilateration is that the robot can compute its orientation (heading) in addition to its location [32, 34, 67], so that the complete *pose* of the robot can be found. Computing the orientation can be as important as the robot position for many applications. Moreover, the orientation is especially downgraded by the odometry [17].

Overview

In this thesis, we present an original angle measurement system, as well as original methods and algorithms, which are parts of a complete robot positioning system in the 2D plane. The framework of the thesis is illustrated in Figure 1.2, in which the number of the corresponding chapter is indicated at the upper left corner of each box. The content of each chapter is detailed hereafter:

Chapter 2 presents our original angle measurement system, named BeAMS. We detail our specific application and explain our motivation for the design of a new system. BeAMS is based on a rotating receiver located on the robot, and beacons sending On-Off Keying (OOK) infrared signals. We detail all the hardware components and explain the angle

measurement principle. Also, we discuss the parameters and trades-off involved in our system, as well as a more general and practical deployment of BeAMS.

Chapter 3 provides a theoretical framework to analyze the errors affecting the measured angles, due to the use of an OOK modulation mechanism. In particular, we establish the theoretical upper bound of the variance affecting the angle measurements.

Chapter 4 complements the results of Chapter 3 by going into further details related to the code statistics of modulated signals in general, with an emphasis on BeAMS. We also present simulated results, in order to validate the theory established in both Chapters 3 and 4.

Chapter 5 compares the theoretical model to simulation data and real measurements. Values for the actual precision (variance) and for the accuracy (bias) of the measured angles are provided. Then, we discuss the relevance of several comparison criteria, and compare our system to other similar systems.

Chapter 6 presents our original three object triangulation algorithm, named ToTal. We discuss the limitations of the existing algorithms and explain our motivation for a new algorithm. Also, we provide a comprehensive comparison with similar algorithms, and benchmarks show that our algorithm is faster and simpler.

Chapter 7 presents a calibration method, dedicated to positioning systems based on three beacons. We identify and detail four sources of errors (biases) resulting in inaccuracies in the computed positions. We establish models of these biases, and propose a complete calibration procedure in order to improve the accuracy of the computed positions and orientations.

Chapter 8 concludes our thesis. We summarize the results of each chapter, and detail our main contributions. We also discuss the performance of BeAMS, as well as a larger system deployment. Finally, we detail how our works take part into a complete positioning system, and we give some hints for future improvements.

Publications

In this thesis, we present original works, which are parts of a complete robot positioning system in the 2D plane. Some of these works have been published in conference articles: (1) the description of the angle measurement system [65] (Chapter 2), (2) a part of the theoretical framework and performance analysis [66] (Chapters 3 and 5), and (3) our three object triangulation algorithm [67] (Chapter 6). The whole theoretical framework (Chapters 3 and 4) has been published as an internal report, since it is mainly composed by mathematical developments. An extended version of the angle measurement system and its theoretical analysis, as well as an extended version of the triangulation algorithm are under publication in journal papers. The complete list of publications may be found on page 157. Finally, the C source code of all triangulation algorithms, including ToTal, as well as the code for the benchmarks, are made available to the scientific community[1].

[1] `http://www.ulg.ac.be/telecom/triangulation`

Chapter 2

Design of BeAMS, a new angle measurement sensor

2.1 Motivation

Our motivation for the design of a new angle measurement sensor has been initiated by our involvement in the EUROBOT contest[1]. This contest opposes two autonomous mobile robots in a playing field of $(2 \times 3)\ m^2$ area. Although the rules change every year, the background is always the same: each robot must execute specific actions, or pick up some objects and place them in some containers, the winner being the one who accumulates the most points in $90\,s$.

In this kind of contest, positioning is also a critical issue, as the robots have to plan trajectories, avoid obstacles, and grab objects. Moreover, the EUROBOT contest is a harsh environment for robot positioning, as collisions between robots and shocks are numerous. As explained in Chapter 1, the odometry is mandatory but not sufficient, and each robot should implement some kind of absolute recalibration.

To help the absolute positioning, three beacon supports per robot are available around the playing field. Therefore, this naturally allows the use of the triangulation or trilateration techniques. Also, the precise knowledge of the robot orientation is an asset for this kind of contest, and for robot positioning in general. Therefore, as the triangulation technique is also able to compute the robot orientation in addition to its position, we decided to use an angle-based positioning system.

While there are many angle measurement systems, none of them was suited for our application, because of the numerous constraints imposed by the EUROBOT contest. As a consequence, we decided to create our own angle measurement system, named BeAMS[2]. The hardware of BeAMS consists of a sensor located on the robot, and several active beacons emitting infrared light in the horizontal plane, located at known positions. In order to identify beacons and to increase robustness against noise, each beacon sends out a unique binary sequence encoded as an On-Off Keying (OOK) amplitude modulated signal. The principle is illustrated in Figure 2.1.

Note that this chapter describes our new angle measurement system, independently of any positioning algorithm. We do not describe a complete positioning system, just a sensor *that can be used* in a positioning algorithm. The description of an algorithm that uses angle

[1] `http://www.eurobot.org`

[2] BeAMS: **Be**acon **A**ngle **M**easurement **S**ystem.

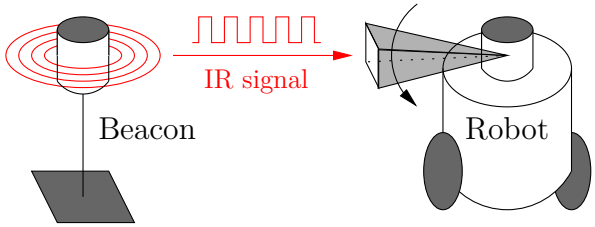

Figure 2.1: Presentation of BeAMS, a new angle measurement system for mobile robots based on two principles: (1) beacons send On-Off Keying coded infrared signals, and (2) the receiver on the robot turns at constant speed to measure angles of beacons. These angles can be combined to compute the robot position.

measurements to compute a position or to navigate can be found in many papers, but is not our focus in this chapter. Triangulation methods using three angle measurements can be found in [21,28,32,34,59,67], and methods using more than three angle measurements are described in [13, 75]. More than three angles can be used to increase precision in some pathological geometrical setups or to deal with harsh environments where some beacons might be out of sight of the sensor [13, 53, 75, 89]. Finally, a method for navigating with angle measurements without computing the position of the robot is described in [11]. For our application, we developed a new triangulation algorithm, described in Chapter 6.

This chapter is organized as follows. Section 2.2 presents some of the numerous angle measurement systems developed for robot positioning. The hardware of our new angle measurement system is described in Section 2.3. The angle measurement principle is explained in Section 2.4. Then, we discuss the parameters and trades-off involved in our system in Section 2.5, and discuss a practical system deployment. Finally, we conclude the chapter in Section 2.6.

2.2 Related work

As explained by Borenstein *et al.* [16], no universal indoor positioning system exists, contrasting with the widespread use of GPS for outdoor applications. This explains the large variety of existing systems depending on the target application and constraints such as cost, accuracy, available volume, coverage area, usable technologies, and safety (*e.g.* laser class). Some surveys on indoor positioning systems may be found in [2, 3, 15, 16, 38, 60]. Technologies used in these systems may be as varied as lasers [69,89], IR [1,85], Ultrasound [14,36], RF [8,64,70] including RFID [57], WLAN, Bluetooth and UWB [25], magnetism, vision-based [13, 44, 55], and even audible-sound [50]. In this study, we concentrate mostly on angle measurement systems, although some of the systems include other forms of measurement, such as range. Hereafter, we present a selection of popular commercial systems and then some "home-made" systems found in the literature, all based on beacons.

2.2.1 Commercial systems

Most commercial systems are described by Borenstein *et al.* in [15], and also by Zalama *et al.* in [89] (NAMCO *LaserNet*, DBIR *LaserNav*, TRC *Beacon Navigation System*, SSIM *RobotSense*, MTI Research *CONAC*, SPSi *Odyssey*, *LS6* from Guidance Control Systems

Ltd., NDC *LazerWay*, and SICK *NAV200*). Almost all of these systems use an on-board rotating laser beam sweeping the horizontal plane to illuminate retro-reflective beacons. The horizontal sweeping is generally performed with a fixed laser emitter and receiver combined with a 45 *deg* tilt mirror mounted on a motor. The angular position of the motor is given by an angular encoder attached to the motor shaft. The beacons are generally simple passive retro-reflectors reflecting the light back to the sensor on the mobile robot. Systems using passive retro-reflectors cannot differentiate between beacons, which makes the task of positioning harder. Furthermore a positioning algorithm working with indistinguishable beacons needs an initial position in order to work properly [43, 89]. In addition, if the beacons are not identifiable, the algorithm could fail in the following two cases: at the wake-up (robot start up or reboot) or when the robot is kidnapped (*i.e.* displaced).

To overcome these issues, some systems use variants such as bar-coded reflective tapes to identify the beacons (for example *LaserNav*, as used by Loevsky and Shimshoni [53], or robot *HILARE* [10]). Another technique for identifying beacons consists of using networked active beacons with an additional communication channel (typically an RF channel). When they are hit by the rotating laser, the beacons communicate to compute the angles between them and send the angles back to the mobile through an RF link (MTI Research *CONAC*). The difficulty with this system is the setup of the networked beacons. The SPSi *Odyssey* system (used by Beliveau *et al.* [12]) is different, since it can position a mobile in 3D. The beacons are laser transmitters and the receiver is located on the mobile. This system is not able to compute the heading of the mobile (unlike on-board angle measuring systems), except while the system is moving (as in the case of the GPS). Moreover, the field of view of the emitters is limited to 120 *deg* horizontally and 30 *deg* vertically, which makes the positioning possible within a limited volume of space. Nowadays, positioning systems have a full 360 *deg* coverage, except for the *Odyssey* and the older *LaserNet* (90 *deg*) systems.

It turns out that most commercial systems use rotating lasers combined with retro-reflective beacons. They generally have a good accuracy and working range, but they cannot differentiate between beacons, apart from the *LaserNav* system, which is no longer manufactured. Finally, they are expensive and take up too much space, which makes them inappropriate for small educational robots. We will now describe home-made systems found in the literature.

2.2.2 Non-commercial systems

2.2.2.1 Rotating lasers

One particular famous non-commercial system is the *Imperial College Beacon Navigation System* [69]. The principle involved in this system appears to be exactly the same as for the *CONAC* system. This system uses a rotating laser and networked active beacons connected to a base station that sends position back to the mobile via an RF channel. The main drawback of these systems is the wiring and setup of the beacons. To overcome this issue, a more recent system, similar to the *Imperial College Beacon Navigation System* and *CONAC,* is presented by Zalama *et al.* [89]. It uses an on-board rotating laser and active beacons that send their identifier back to the mobile with some RF coded pulses when they are hit by the rotating laser. The beacons are totally independent and stand-alone (no network, communication cables or base station), which makes the setup easier.

Even if these systems solve the beacon identification problem, there still exist an open

issue: how do such systems behave when multiple robots use the same setup of beacons? A beacon would send its identifier back to all robots even if only one of them has hit that beacon, causing false angle measurements to the other robots. So we guess that these systems are inadequate to be used by multiple robots simultaneously.

2.2.2.2 Static receivers

In general, the 360 *deg* horizontal field of view is covered by a single receiver combined with a rotating system. However, it is possible to cover the whole field of view without mechanical part, as explained hereafter. The first type of static sensor system uses multiple static receivers uniformly distributed on the perimeter of a circle. These systems measure the angles to the beacons by simply "looking" at which receiver receives the signal from a beacon. Since more than one receiver can receive the same signal, an interpolation can be performed to improve the angular position of a beacon, as highlighted by Gutierrez *et al.* [40] and Roberts *et al.* [73]. These systems generally also compute the distance to the beacons. For example, some of these systems [39, 40, 46, 71–73] use the infrared received signal strength to compute the distance to the beacon, in addition to the bearing information. In [41], Hernandez *et al.* compute the distance by using the aperture angle of the received signal (time taken to sweep the receiver). In [27], Durst *et al.* use Nintendo Wii cameras instead of infrared receivers to localize and identify the different beacons. Bergbreiter *et al.* present *PhotoBeacon*, which also consists in a circular array of photodiodes. But, unlike the previous systems, the 256 photodiodes are located on a miniaturized custom CMOS chip. The 360 *deg* horizontal field of view is ensured by a 190 *deg* field of view fisheye lens, placed horizontally on top of the CMOS chip. Lee *et al.* [51], and Arai and Sekiai [4] use infrared light from beacons and measure the incident angle of the infrared light with two fixed photodiodes and a specialized circuit. Another similar idea consists in the use of only one static receiver or laser emitter. The 360 *deg* field of view is obtained by the rotation of the robot itself, which is expected to move to see the beacons [62, 80]. The main drawback is that the position update rate depends on the robot movements and is generally low compared to other systems.

These systems have the benefit of being small, lightweight, and simple (no moving part). Unfortunately, it turns out that these systems are less accurate ($5 \rightarrow 10\,deg$) than rotating sensors ($0.05 \rightarrow 0.5\,deg$), and that the accuracy of the angles depends on the number of receivers. They are often used by swarm of robots for relative positioning and communication, but not for precise absolute positioning.

2.2.2.3 Panoramic cameras for detecting beacons

The second type of static sensor systems uses panoramic cameras to measure angles or distances. A common way to measure angles with a static camera and without moving parts is to transform it into an omni-directional camera via a catadioptric mirror, fisheye lens, or a reflecting ball, as proposed by Betke and Gurvits [13]. With this configuration, a 360 *deg* horizontal field of view of the scene is taken in one image. The angular positions of the beacons are computed through image processing by searching the beacon patterns within a circular region of the image. The authors of [44] base their system on the same principle with only one beacon, but they also compute the distance to that beacon. Yamamoto *et al.* [86] describe a system called *NorthStar*. This system is composed of a fixed projector which creates two distinguishable infrared spots on the ceiling and a camera mounted on the mobile robot. The

position of the robot is computed according to the positions of these spots in the image.

One distinctive feature of panoramic cameras is that angles to beacons are measured at the same time, in one image. This can be an advantage if the positioning algorithm uses a triangulation technique directly. This advantage is useless if the angles are fed into an Extended Kalman Filter (EKF), which can take advantage of one angle at a time. Panoramic cameras also need a more complicated image processing algorithm, and they depend highly on lightning conditions. Finally, like the multiple static receivers, they are less accurate than rotating systems.

2.2.2.4 Most closely related systems

Finally, we present systems that are closely related to ours. One of the oldest systems is described by McGillem and Rappaport in [59]. That system is made up of beacons emitting infrared modulated signals and a rotating infrared detector mounted on a turntable to measure the angles to the beacons. Another recent and similar system is proposed by Brkic *et al.* in [19]. This system relies on infrared beacons and a rotating receiver; a brushless DC motor with rotary transformer overcomes the problem of contact-less power supply, and ensures signal transfer. Unfortunately, no information about motor control, infrared codes, or angle calculation is provided in that paper. Finally, the accuracy of the system is given in terms of distance errors on the moving area, and no information about the accuracy of measured angles is given. Kemppainen *et al.* [47] also describe a system similar to ours for multi-robot spatial coordination, the system being used for inter-robot relative positioning, not absolute positioning. The difference with the previous systems is that the infrared emitting beacons are located onto the robots themselves, instead of being at fixed locations. In addition to the angle measurement, the system estimates the range by the received signal strength. Using the bearing and the range, a robot can compute the relative position of all other robots.

These systems (emitting beacons, and rotating receiver) are able to identify the beacons while using only one communication channel (the beacon signal itself). Due to the nature of this unidirectional channel, multiple receivers (robots) can receive the signals from the same beacons at the same time without disturbing each other (like for the GPS system). But, unlike simple reflective tapes, the beacons have to be powered up.

2.2.3 Summary

There is a large variety of angle measurement systems and the choice of a particular system has to take into account the target application and constraints such as cost, accuracy, available volume, coverage area, usable technologies, and safety (*e.g.* laser class). Some systems do not identify the beacons, and others require more than one communication channel. Some systems cannot position multiple robot simultaneously. If we compare the values found in the literature, it turns out that rotating sensors are more accurate than fixed sensors, but have the disadvantage that information and power have to be transmitted to the sensors, if these are located on the turning part of the system. A fixed sensor can be used, if combined with a mirror and a motor to sweep the horizontal plane and cover a $360\,deg$ field of view. With a mirror, the light rays are redirected to cross the rotation center of the turning system. In general, the mirror is mounted on a hollow gear, which is driven by the motor through a gear or belt, allowing light rays to pass and reach the sensor. This solution tends to make the mechanical part of rotating systems more complicated and cumbersome. It turns out

that the most flexible solutions are rotating lasers with passive bar-coded reflective tapes or active emitting beacons with a rotating receiver. This last solution requires to power-up the beacons. Most of the time, implementation characteristics are missing or incomplete, such as the working distance, power consumption, dimensions, etc. Finally, evaluation criteria such as precision (variance), accuracy (bias) and resolution (number of steps for one turn) are sometimes confused during the performance analysis. Also, some systems are evaluated through a positioning algorithm, and therefore it is a hard task to evaluate the quality of the underlying angle measurements to compare systems. A recap chart of some commercial and home-made systems and their performances is provided later (see Table 5.2 on page 77).

2.3 Hardware description of a new system

While there are many angle measurement systems, none of them was suited for our application, as explained hereafter. Our first motivation for this work was to create a new system for the EUROBOT contest[3], which imposes many constraints. For the positioning part, the most important constraints are: (1) the available volume for the hardware on the robot is limited to $(8\times8\times8)\,cm^3$, (2) home-made laser systems are prohibited except if they are manufactured and kept in their house cases. The EUROBOT contest is a harsh environment for robot position. Firstly, as collisions and shocks are numerous, the knowledge of beacon IDs is an advantage to be robust to the wake-up or kidnapped issues. Secondly, the environment is polluted by many sources of noise including infrared, lasers, radio waves, and ultrasound signals. Also the lightning conditions are very bad and there are lots of shiny or reflective surfaces. Finally, more than one robot per team may evolve on the field.

Considering all these constraints, the system has to identify the beacons, use coded signals, and allow multiple robots. Commercial system are unsuitable because of their sizes, their high prices, and because they cannot identify the beacons. Home-made laser systems are prohibited. Static receivers do not provide the accuracy needed for this contest. Finally we wanted to use a triangulation based positioning to estimate the robot heading precisely (this is important since the heading is highly downgraded by odometry). So, we designed an angle measurement system based on beacons emitting infrared coded signal and a rotating receiver. Note that, to our knowledge, there is only one very similar system, designed by Brkic *et al.* [19], also for the EUROBOT contest. But, according to the authors, their system is not accurate enough to position the robot (see Table 5.2 on page 77).

BeAMS is original but it has the same limitations as any other optical system, as explained in [16, 60, 89]. First, a line of sight between beacons and sensor has to be maintained for the system to work. Also, the reflections of beacon signals on shiny surfaces can lead to false detections. Finally, the sensor could be blinded by direct sunlight (causing the SNR to decrease). This has the effect of reducing the working range in outdoor conditions.

2.3.1 Architecture of BeAMS

The hardware of BeAMS consists of a sensor located on the robot, and several active beacons emitting infrared light in the horizontal plane, located at known positions. This configuration is represented in Figure 2.2. As illustrated in this figure, the aim of the sensor and processing unit is to determine the identifier of each beacon i, as well as its azimuthal angle Φ_i, in

[3] `http://www.eurobot.org`

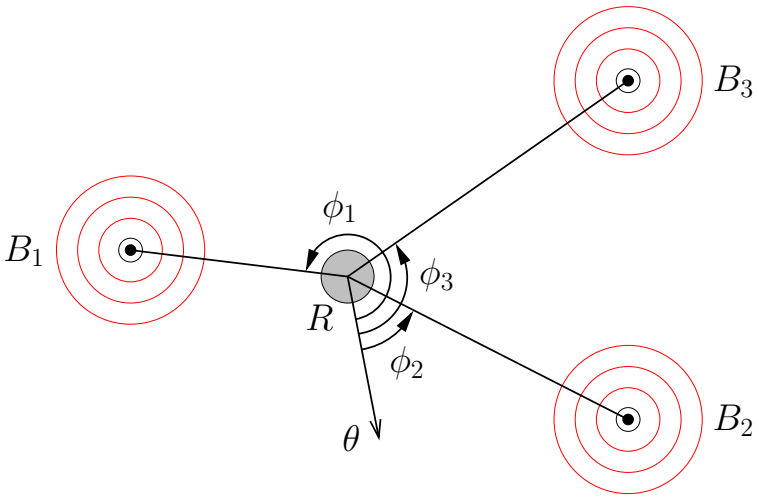

Figure 2.2: Schematic top view representation of BeAMS. The system is composed of: (1) several active beacons B_i emitting infrared light in the horizontal plane, and (2) a sensor located on the robot R. The aim of the sensor is to measure the azimuthal angles ϕ_i of the beacons in the robot reference determined by θ.

the robot reference, whose orientation is given by θ. The sensor is composed of an infrared receiver/demodulator and a motor. The beacons are infrared LEDs whose signal is modulated. To achieve the angle measurements, the infrared receiver is combined with the motor turning at a constant speed. In order to identify beacons and to increase robustness against noise, each beacon sends out a unique binary sequence encoded as an On-Off Keying (OOK) amplitude modulated signal over a 455 kHz carrier frequency. Furthermore, BeAMS only requires one infrared communication channel; there is no synchronization channel between the beacons and the robot, which allows multiple robots to share the same system. Finally, the mechanical part of the system is kept as simple as possible (motor only), with no gear system or belt, thanks to the hollow shaft, and no optical encoder for the motor control or angle measurement is needed. These features make BeAMS a small, low power, flexible, and tractable solution for robot positioning. BeAMS has been continually improved since its inception and has been used successfully in the EUROBOT contest for the last four years. In the next sections, we describe the hardware components of our system.

2.3.2 Description of the beacons

The core of a beacon is composed of IR LEDs (more precisely the SFH485P component); they emit signals in a plane parallel to the moving area and are directed towards the center of the moving area. These LEDs have a large emission beam so that a small number of LEDs per beacon can cover the whole area. A PIC microcontroller generates the appropriate signal to drive the IR LEDs through the power stage. Each beacon continuously emits its own unique IR signal so that the receiver can determine the beacon identifier (ID). This ID is determined by switches connected to the PIC microcontroller, so that the hardware of each beacon is identical. Figure 2.3 represents the schematic architecture of a beacon and Figure 2.4 shows a picture of a beacon. The power consumption is $100\,mA$ at $9\,V$, for a working distance of up to $6\,m$.

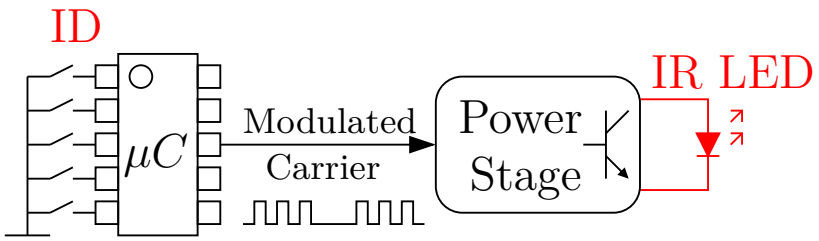

Figure 2.3: Architecture of a beacon. The central element of a beacon is an infrared LED. A PIC microcontroller (μC) generates the appropriate signal to drive the IR LEDs through the power stage. Each beacon emits its own unique IR signal continuously so that the receiver can determine the beacon identifier (ID). This ID is determined by switches connected to the PIC microcontroller, so that the hardware of each beacon is identical.

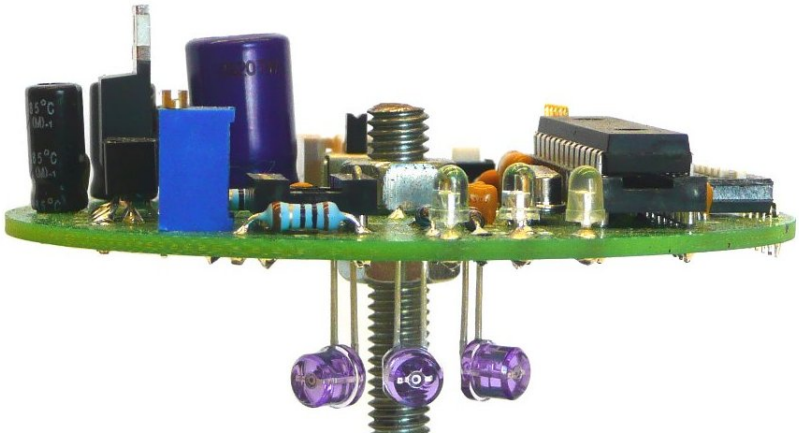

Figure 2.4: Picture of a beacon. The main part of a beacon is made by IR LEDs, which are located under the printed circuit board, parallel to the moving area and directed towards the center of the moving area.

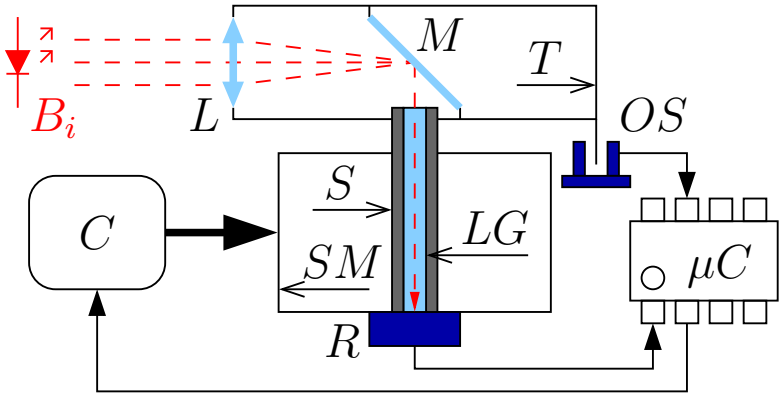

Figure 2.5: Schematic representation of the receiver. B_i is a beacon emitting IR light, L is the lens, M is the mirror, LG is the light guide, R is the receiver, SM is the stepper motor, S is the hollow shaft, T is the turret, C is the motor controller, μC is the PIC microcontroller, and OS is the optical switch.

2.3.3 Description of the sensor

As shown in Figure 2.5, the sensor is composed of a mini stepper motor[4], a $12\,mm$ thin plastic convergent lens, a small front surface mirror with a $45\,deg$ tilt, a polycarbonate light guide placed in the center of the motor shaft (which has been drilled for this purpose), an IR receiver (a TSOP7000 from VISHAY) and an optical switch[5] used to calibrate the zero angle reference θ (see Section 2.4.2). The lens and mirror are placed on a "turret", which is fixed to the motor shaft. The receiver is fixed to the bottom of the motor, just under the light guide. This configuration allows IR signals from a beacon to reach the fixed receiver through the entirely passive "rotating turret" and light guide. As a result, the receiver can virtually turn at the same speed as the turret. By introducing this original disposition of optical elements into our system, the system behaves as if the receiver is turning without the mechanical constraints and inconvenience. Finally, a PIC microcontroller[6] is used to drive the motor through its controller[7] and to decode the output of the receiver. Figure 2.6 shows a picture of the sensor. The entire sensor weights $195\,g$, and the power consumption is $47\,mA$ at $9\,V$. Now that the hardware elements have been presented, we detail some elements of the system: software architecture, stepper motor control, angle measurement principle, and infrared codes.

2.4 Measurement principles of BeAMS

2.4.1 Software description

The software building blocks of BeAMS are drawn in Figure 2.7. The key principle of the software is to use a common timer to drive the stepper motor at a constant speed, and to capture the receiver output edges. The receiver output is connected to a capture module in

[4] MY5602/MY7001 from ASTROSYN.
[5] HOA2001 from HONEYWELL.
[6] PIC18F2620 from MICROCHIP.
[7] A3977 from ALLEGRO.

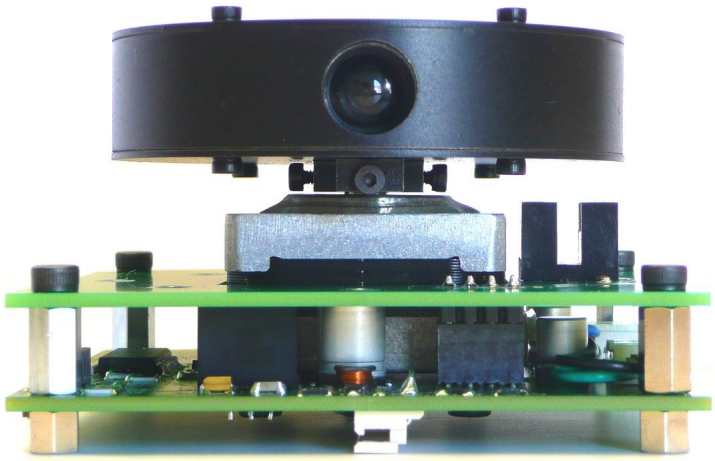

Figure 2.6: Picture of the sensor with the rotating turret in black (top) and the lens, the stepper motor (middle), the optical switch (middle right), and the electronic card (bottom).

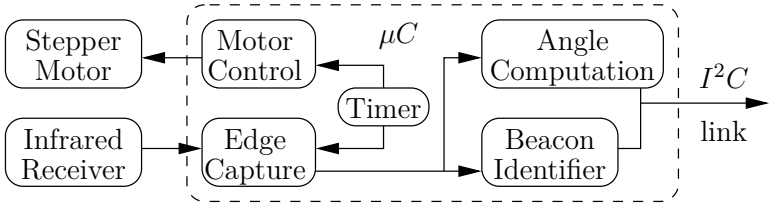

Figure 2.7: Software organization of BeAMS. μC is the PIC microcontroller. A common timer is used to drive the stepper motor at a constant speed and to capture the edges of the receiver output. These captured values are used to compute the angular positions of the beacons as well as the beacon identifiers. An I^2C link is used to communicate the angle measurements to the positioning algorithm.

the microcontroller. On a falling or a rising edge of the receiver output, this module latches (captures) the actual timer value to a register that may be read later by the software. This allows us to associate a time to each incoming event (falling or rising edge). And as the value of the timer is perfectly linked to the motor angular position, the association of an incoming receiver event to an angular position is as accurate as possible. The captured values serve to compute the angular position of the beacons and their IDs. An I^2C link is used to communicate the angle measurements to the positioning algorithm.

2.4.2 Stepper motor control

The stepper motor is driven in an open loop via an input square signal to advance the motor step by step. The stepper motor has 200 real steps and is driven in a half-step mode via its controller, which turns the number of steps into 400. The frequency of the step signal controls the rotation speed of the motor and is derived from the common timer. Since the timer is 156 times faster than the step signal, we achieve a sub-step time resolution so that the number of "virtual" steps is $400 \times 156 = 62400$ exactly. The motor turns at a constant speed ω and the angular position of the turret/receiver ϕ is thus proportional to the value of this timer. Whereas the motor is controlled step by step, the rotation is assumed to be continuous due to the high inertia of the turret compared to the motor dynamics. Since the motor turns at a constant speed, the common timer value can be seen as a linear interpolation of the motor position between two real steps of the motor.

This kind of control in open loop with a stepper motor is possible since the torque is constant and only depends on the turret inertia and motor dynamics, which are known in advance. The advantage of this approach is that we do not need a complicated control loop or expensive rotary encoder in order to detect the position of the turret with precision. Indeed, the common timer acts as a rotary encoder, and the position of the turret can be obtained by reading the value of the timer. As explained earlier, there are 62400 "virtual" steps of the motor. The angular resolution is thus given by $^{360}/_{62400} = 0.00577\,deg$ $(0.1\,mrad)$. The timer clock runs at $625000\,Hz$, to give an angular speed of $^{625000}/_{62400} = 10.016\,^{\text{turn}}/_{\text{s}}$. Since the motor is controlled by an open loop, the motor can start from or stop at any angular position, meaning that there is no angle reference. To overcome this issue, we use a single optical switch (denoted OS in Figure 2.5) to calibrate the zero angle reference θ by reading the timer value when the turret passes through the switch.

2.4.3 Angle measurement mechanism

Let us denote by ϕ the current angular position of the turret/receiver, relatively to θ. As the turret turns at a constant speed ω, the angular position ϕ is directly proportional to time

$$\phi(t) = \omega\, t. \tag{2.1}$$

As a result, we can talk about either time or angular position indifferently. For convenience, we prefer to refer to angles instead of time units.

The TSOP7000 is a miniaturized IR receiver that acts as an OOK demodulator of a 455 kHz carrier frequency. The input is a modulated signal whose carrier wave is multiplied by the "0" or "1" binary message. The receiver outputs a value "1" when it detects the carrier wave and "0" otherwise. The output of the receiver corresponds to the envelope of the modulated carrier wave. No other information is given by the receiver. By design, the

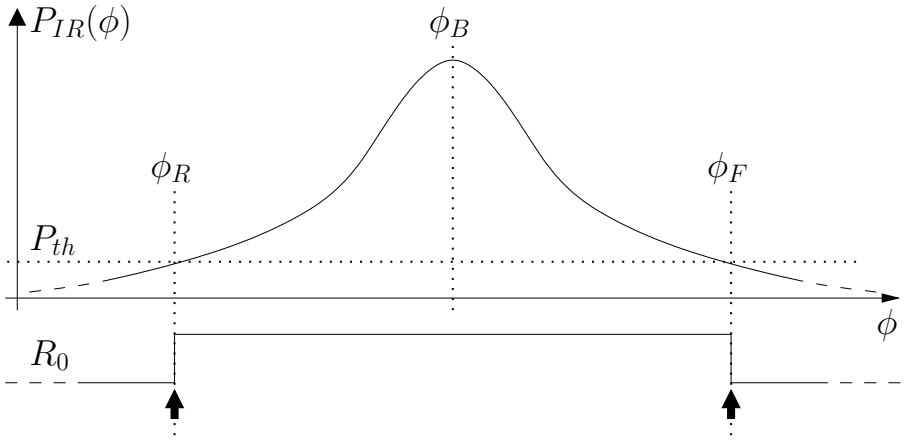

Figure 2.8: The upper curve $P_{IR}(\phi)$ is the expected infrared power collected at the receiver while the turret is turning. R_0 represents the receiver output for a non modulated infrared carrier wave (pure 455 kHz sine wave). The black arrows represent the measured values respectively for ϕ_R to the left (first *Rising* edge) and for ϕ_F to the right (last *Falling* edge).

receiver combined with the optical components has a narrow field of view and, consequently, the amount of infrared power collected at the receiver, denoted by $P_{IR}(\phi)$, depends on the angle. This power also depends on the power emitted by the beacon, and the distance between the beacon and the receiver. Note also that the exact shape of $P_{IR}(\phi)$ depends on the hardware, that is the IR LEDs, the receiver, optical components, and the geometry of the turret. It is impossible to derive the precise power curve from the specifications, because we only have access to the demodulated signal, and no information about power is available. Therefore, we make some basic assumptions regarding the shape of $P_{IR}(\phi)$; the resulting expected curve $P_{IR}(\phi)$ is shown in Figure 2.8. The exact shape of this curve does not have much importance in this study but is assumed to increase from a minimum to a maximum and then to decrease from this maximum to the minimum (ambient noise level). In the following theoretical developments, we make three important assumptions about the curve and the detection process itself:

1. The maximum coincides with an angle, which is the angular position of the beacon, denoted ϕ_B (*i.e.* the angle we want to measure). As a result, for any angle ϕ

$$P_{IR}(\phi) \leq P_{IR}(\phi_B). \tag{2.2}$$

2. The curve is supposed to be symmetric around the maximum since the turret and all optical components are symmetric. This means that

$$P_{IR}(\phi_B - \phi) = P_{IR}(\phi_B + \phi). \tag{2.3}$$

3. Finally, we assume that the receiver reacts to $0 \rightarrow 1$ (raising) and $1 \rightarrow 0$ (falling) transitions at the same infrared power threshold P_{th}, respectively at angles ϕ_R and ϕ_F

$$P_{IR}(\phi_R) = P_{IR}(\phi_F) = P_{th}. \tag{2.4}$$

This hypothesis will be discussed later, in Section 5.7.

From equations (2.3) and (2.4), we derive that $\phi_B - \phi_R = \phi_F - \phi_B$ and that

$$\phi_B = \frac{\phi_R + \phi_F}{2}. \tag{2.5}$$

This equation expresses an important innovation that has two benefits: (1) we derive the angle of the beacon not from the maximum power, but from two angle measurements that take the narrow receiver optical field of view into account, and (2) by measuring an angular window (that is two angles) instead of one angle, it is possible to analyze the temporal evolution of the signal inside this window to determine the code of the beacon (or any other kind of useful information). Note that the angular window, defined as $\phi_F - \phi_R$, depends on the received IR power. It increases if the received power increases, and decreases if the received power decreases (see Figure 2.8).

First, we assume that the beacons send a non modulated IR signal, that is a pure 455 kHz sine wave and explain the measurement principle for one beacon; the principle is the same for any number of beacons. While the turret is turning, the receiver begins to "see" the IR signal from that beacon when the power threshold P_{th} is crossed upwards ($0 \rightarrow 1$ transition). The receiver continues to receive that signal until P_{th} is crossed downwards ($1 \rightarrow 0$ transition). The receiver output is depicted as R_0 in Figure 2.8. At these transitions, the capture module latches values for ϕ_R and ϕ_F. The angular position of the beacon is then computed after equation (2.5).

2.4.4 Beacon identifier and infrared codes

The convenient assumption of continuous IR signals used in the previous section is not realistic because (1) we would not be able to distinguish between the different beacons, and (2) it is essential to establish the beacon ID (especially in a noisy environment like the EUROBOT contest where other IR sources may exist). Moreover, the possibility of identifying beacon IDs solves the wake-up and kidnap issues typical of robot positioning. In other words, we propose to code the IR signal of beacons to solve these issues.

In BeAMS, each beacon emits its own code over the 455 kHz carrier wave; this emission is continuous so as to avoid having any form of synchronization between the beacons and the receiver. As a result, each beacon signal is a periodic signal whose period corresponds to a particular code defining the beacon ID. The design of these codes is subject to several constraints related to (1) the receiver characteristics, (2) the loop emission, (3) the desired precision, (4) the system's immunity against noise, and (5) the number of beacons. We elaborate on these constraints below:

1. Receiver. The TSOP7000 requires that the burst length (presence of carrier wave) be chosen between 22 and 500 μs, the maximum sensitivity being reached with 14 carrier wave periods ($14/455000 = 30.8\,\mu s$). The gap time between two consecutive bursts (lack of carrier wave) should be at least 26 μs.

2. Loop emission. Because of our willingness to avoid a synchronization between beacons and the receiver, we must ensure that the periodic emission of a code does not introduce ambiguities. For example [0101] is equivalent to [1010] when sent in a loop. Thus any rotation of any code on itself must be different from another code.

3. Precision. The lack of synchronization between beacons and the receiver introduces a certain amount of imprecision. Indeed, the first received IR pulse may be preceded by a gap time corresponding to a zero symbol. This affects the estimation of ϕ_R. The same

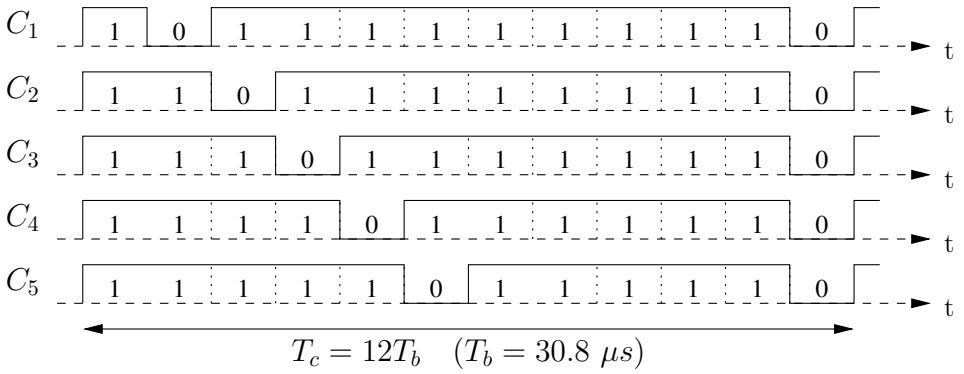

Figure 2.9: Temporal representation of the C_1, C_2, C_3, C_4 and C_5 codes. These codes are repeated continuously and multiply the 455 kHz carrier wave to compose the complete IR signal.

phenomenon occurs for ϕ_F. An intuitive design rule would say that we have to reduce the duration of zeros, as well as their frequency of appearance. Therefore, we forbid two or more consecutive zeros, and the duration of one zero (the gap time) must be reduced as much as possible (see Chapter 3 for further explanations about the error due to the gap time).
4. Immunity. The codes should contain enough redundancy to be robust against noise or irrelevant IR signals.
5. Number of beacons. The codes should be long enough to handle a few beacons, but as short as possible to be seen many times in the angular window associated to a beacon, thus improving the robustness of the decoding.
All these constraints lead us to propose this family of codes:

$$C_i = [1^i\, 0^1\, 1^{2N_c - i}\, 0^1], \qquad i = 1, \ldots, N_c, \tag{2.6}$$

where N_c denotes the number of codes in the system. The duration of a bit is set to $T_b = 30.8\,\mu s$ since this value maximizes the receiver sensitivity, while respecting the minimum gap time ($T_b > 26\,\mu s$). Although not mandatory, the duration of a one symbol has been chosen to be equal to the duration of a zero. This is to simplify the implementation of the beacons and to ease the decoding process. From expression 2.6, one can see that the number of ones in a code is equal to $N_1 = i + (2N_c - i) = 2N_c$, and the number of zeros is $N_0 = 2$, for all the codes. The number of bits in a code is $N_b = N_1 + N_0 = 2N_c + 2$, and the duration of a code is $T_c = N_b T_b$. The gap time is the same for all codes and corresponds to the duration of a unique zero symbol. The second half part of the codes can be seen as a checksum, since it makes the number of ones constant ($2N_c$). In our current implementation, we have $N_c = 5$ codes because this is appropriate for our application. Figure 2.9 shows the temporal representation of the codes for $N_c = 5$. Note that any code meeting the second requirement (differentiable under loop emission) would work to identify the beacons. However, they may not meet the third requirement if the zero symbols appear in random patterns. Indeed, a thorough analysis about the error introduced by the gap time (detailed in Chapter 3) shows that this error increases with the frequency of zero symbols and with the square of the zero duration. Moreover, a simulator (detailed in Chapter 4) has been created to validate this result. This simulator helped us to compare codes with respect to the error they generate. From our experience, the codes presented in expression (2.6) are the best ones that meet all

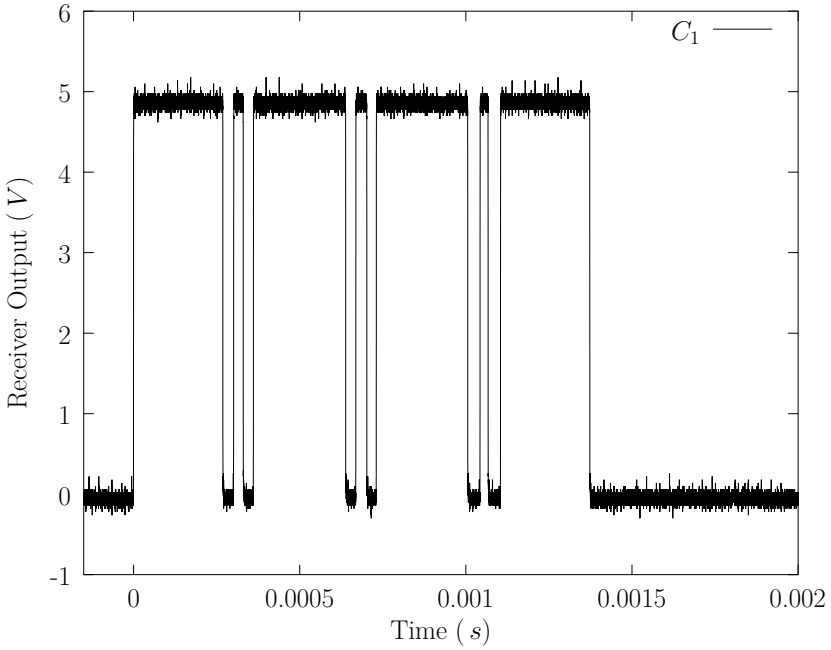

Figure 2.10: Oscilloscope measurement at the receiver output for the code 1.

our requirements, but we have no formal proof of it.

The angle measurement principle still operates exactly as in Section 2.4.3 even if the IR carrier wave is modulated by the codes. Since there are gap times in the IR signal of a beacon, there are more than two edges in the received signal. The intermediate edges are used to determine the beacon ID, by analyzing the durations of burst lengths and gap times. But the first and last edges of the received signal always correspond to our measurements of ϕ_R and ϕ_F. These two edges are isolated from all other edges due to a timeout strategy, which relies on the fact that the separation time (or angle) between two different beacons is much greater than the separation time between consecutive edges in a code. Actually, the separation time is set to four bit durations, which corresponds to a separation angle equal to $0.44\,deg$. To illustrate this, we have performed some oscilloscope measurements at the receiver output. The setup is made of the sensor, and one beacon, in order to ease the synchronization in the oscilloscope. We have taken measurements for the five codes. The results are presented in Figure 2.10 to Figure 2.14. By analyzing the burst and gap times in these graphics, one can recognize the different codes, as defined by expression (2.6). One can notice that the codes appear multiple times in one angular window[8]. Also, one can observe that the first and last bursts may be incomplete, since there is no synchronization between beacons and the receiver.

2.4.5 Multiple beacon detection

Note that, for the proper functioning of the system, it is important that the receiver collects infrared light from one beacon at a time. To do this, some additional optical components are

[8]For this experiment, we have chosen values of the distance and emitted power, such that the angular window has a small value (about $5.4\,deg$). This is to reduce the number of edges, and to ease the interpretation of the graphics.

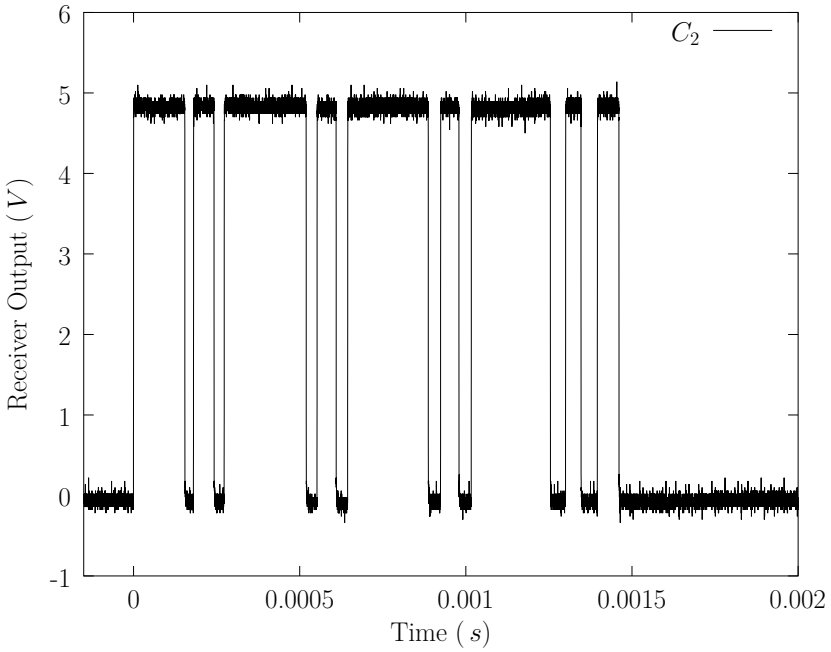

Figure 2.11: Oscilloscope measurement at the receiver output for the code 2.

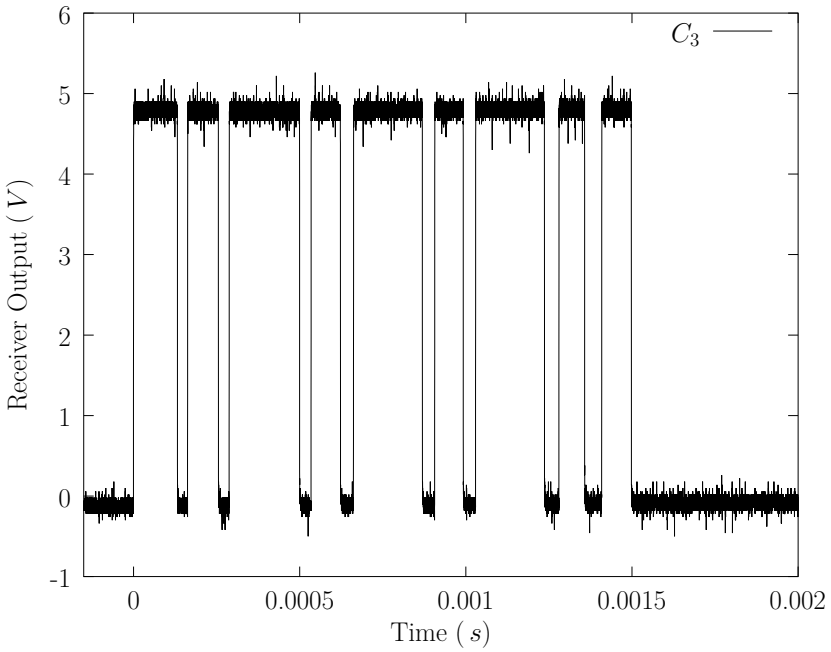

Figure 2.12: Oscilloscope measurement at the receiver output for the code 3.

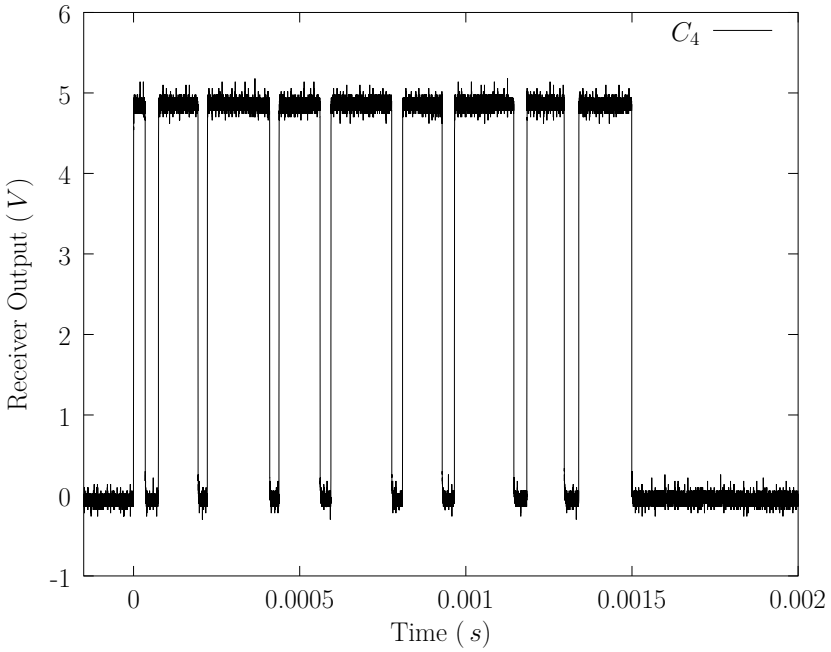

Figure 2.13: Oscilloscope measurement at the receiver output for the code 4.

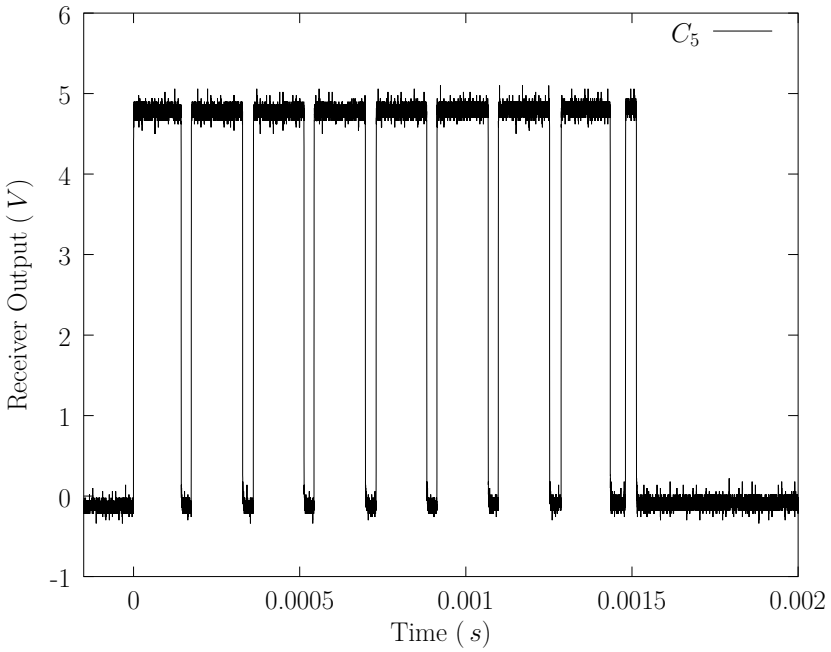

Figure 2.14: Oscilloscope measurement at the receiver output for the code 5.

used to limit the field of view of the receiver to a narrow value (a few degrees). However, despite the narrow field of view, and the timeout strategy to separate beacons, two or more beacons might appear in the same angular window if they are close enough (from an angular point of view). When this situation occurs, the demodulated signal is composed of codes from the different beacons, and they appear in the same order as the turret turns and sees the different beacons. This means that the timeout strategy is not able to cut the different signals. Also, at the transition points, the signal could be composed of burst or gap durations that do not correspond to any code. The decoding algorithm simply adds (in counters) the number of different codes it sees, as well as bad durations. Therefore, the receiver is capable to differentiate between a pure signal from one beacon, or a compound signal from several beacons. The system can then decide to keep or reject a compound signal; this capability to check the consistency of codes is an important advantage of BeAMS. To illustrate this phenomenon, we have also performed some oscilloscope measurements at the receiver output. But this time, we have placed two beacons in the setup, emitting the code 3 and the code 5, respectively. In the first experiment, the separation time between the two beacons is greater than the timeout required. The result is presented in the top plot of Figure 2.15. In this graphic, one can observe both occurrence of consecutive codes, as well as the separation gap between the beacons. In that case, the sensor can separate both signals. In the second experiment, the two beacons are closer than the separation angle required. The result is presented in the bottom plot of Figure 2.15. This time, there is no separation gap and the sensor cannot separate both signals. But, as explained earlier, one can still recognize both codes in the received signal.

2.5 Practical considerations

2.5.1 Parameters and trades-off

The design of BeAMS implies many parameters and some trades-off that need to be explained. First of all, we had to choose a turning speed (or acquisition rate). Lots of commercial or non commercial systems work at $10\,Hz$, which seems sufficient for a robot moving at moderate speed. Also, the accuracy of these systems is more likely to be $0.1\,deg$, to get a reasonable accuracy on the final position. Our system uses an OOK modulation that leads to an error due to the gap times (T_0). Our statistical analysis, as well as our simulator confirm this result. However, it is easy to show that the maximum absolute angular error on the first and last edges is given by $\phi_0 = \omega\, T_0$, since the turret has turned by this angle during a gap time. Therefore, the maximum absolute error on the beacon angle is given by $\phi_0/2$ (according to equation (2.5)). This last equation represents the most important trade-off: for a given receiver, increasing the rotation speed would increase the error on measured angles. We decided to choose a receiver with the smallest T_0, and afterwards the maximum turning speed according to the maximum error accepted. In our case, the TSOP7000 was the only receiver providing the minimum gap time satisfying the pair of parameters ϕ_0 and ω (about $0.1\,deg$ and $10\,{}^{\text{turn}}/_{\text{s}}$, respectively). Then the optical field of view has been tuned with optical components to be narrower, but large enough to receive some bits/codes from one beacon in the angular window, for this turning speed. Typically, we receive a minimum of 20 bits (~ 2 codes) at the maximum range, which corresponds to the smallest angular window. So, for a given receiver, this maximum working distance depends on the emitted power combined with the size of the lens, and the minimum number of bits we need to identify the beacons. In our

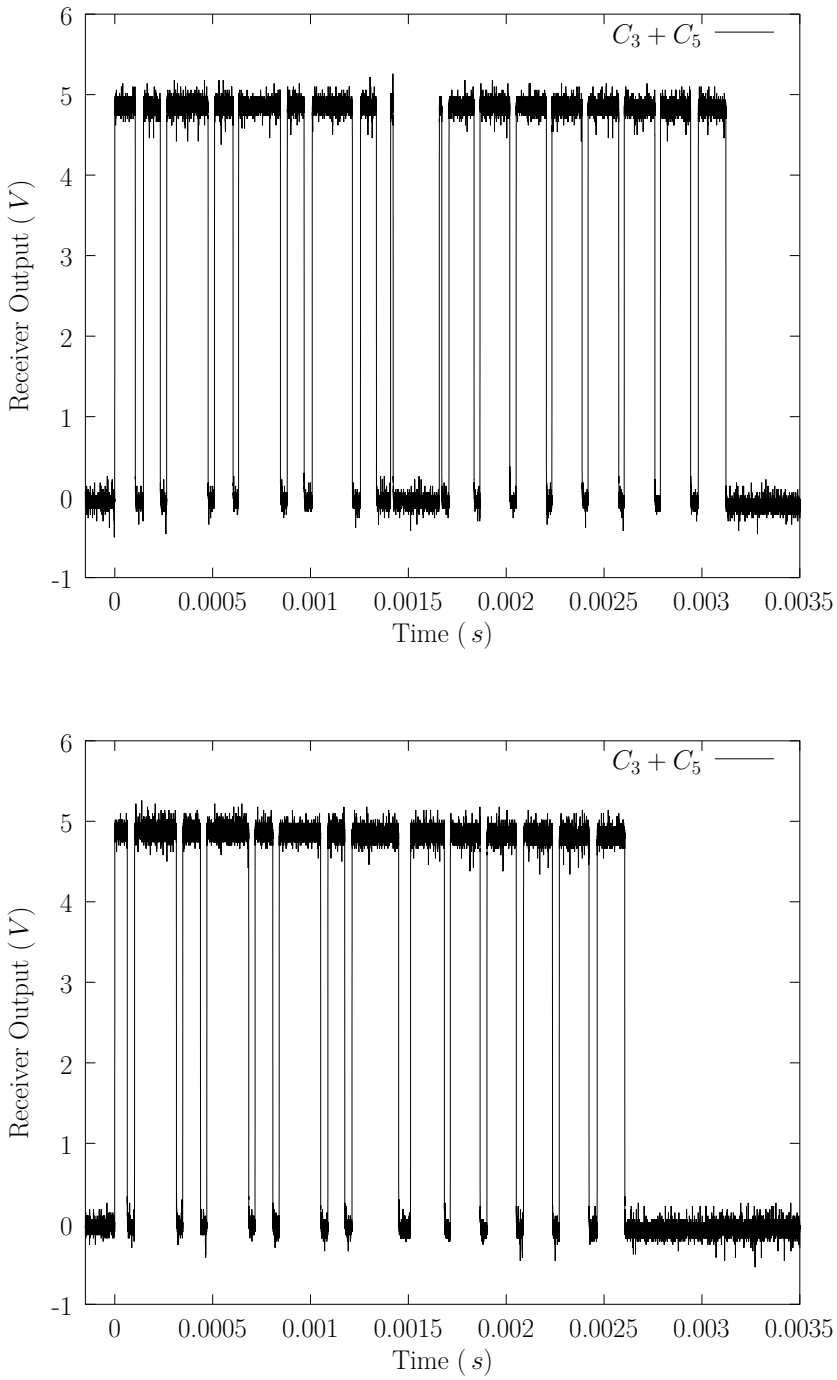

Figure 2.15: Oscilloscope measurement at the receiver output for two beacons emitting the code 3 and the code 5, respectively. Top plot: the separation time is greater than the timeout and the sensor can separate both signals. Bottom plot: the separation time is lower than the timeout and the sensor cannot separate both signals. In both cases, it is possible to distinguish between both codes.

case, these parameters have been chosen to meet the EUROBOT rules. The lens/focal distance has been chosen to hold in the allowed volume. Then the emitted power has been tuned to reach the maximum distance possible on the moving area. Indeed the system works up to $6\,m$, which is greater than required.

2.5.2 System deployment

BeAMS has been designed for the EUROBOT contest. However, the system could be used in any other application involving angle measurements based positioning. Two parameters are important in order to use BeAMS in another context: (1) the covered area, and (2) the number of beacons. Also, we have to consider the multiple beacon detection issue.

Obviously, the covered area is determined by the maximum working distance. The current version of BeAMS reaches $6\,m$ with a small lens ($12\,mm$ diameter) and usual LEDs. This distance can be increased either by increasing the size of the lens, or by rising the emitted power. In our application, the size of the lens is limited since the available volume is limited. The emitted power can be increased either by choosing other IR LEDs or by increasing the number of LEDs. The received power is given by

$$P_r = \frac{S\,P_e}{4\pi d^2}, \tag{2.7}$$

where P_e is the emitted power, d is the distance, and S is the surface of the lens. These parameters can be modified, while the minimum received power is unchanged. For example, multiplying the emitted power P_e by four, and the surface of the lens S by nine ($36\,mm$ diameter) would multiply the working distance d by six, as d is proportional to $\sqrt{SP_e}$ (see equation (2.7)). With our prototype, we would reach a distance of $36\,m$, which is comparable to commercial systems.

Then we have to consider the number of beacons. Although Figure 2.2 represents the system with three beacons, it is important to note that the sensor can measure angles for any number of beacons, three being the minimum number to achieve unequivocal positioning (Chapter 6). We chose a code family that allows 5 beacons because it was sufficient for our application. With the same code family and receiver, we can go up to 9 beacons since we are limited by the maximum burst length permitted by the receiver. However, we can use any other codes respecting the constraints of the receiver, as explained in Section 2.4.4. The number of codes is limited by the minimum number of bits we receive at the maximum working distance. This minimum number of bits received in the time window is fixed by the optical field of view combined with the rotation speed, as expressed by equation (2.1). As explained previously, there is a trade-off. Increasing the rotation speed decreases the time window, and subsequently the number of bits and the number of possible codes. In our application, the minimum number of bits is more or less 20, at the maximum range with a turning speed of $10\,{}^{\text{turn}}/_{\text{s}}$. As explained in Section 2.4.4, we use a checksum and we want to see the code many times in the time window (minimum 2 times in our case). From a practical point of view, 10 of these 20 bits could be used to code the beacons, without jeopardizing the noise robustness. Therefore, this would allow a maximum of 1024 beacons. Note that this is not a hard limit of the system, but this is just an example with the current parameters of BeAMS.

Finally, we have to consider the beacon spacing and multiple beacon detection issue. As explained in Section 2.4.4, the signals from different beacons could appear in the same time window if they are too close (from an angular point of view). In a practical application, it

occurs when the beacons are almost aligned (a few ϕ_0 of angular separation). This has the effect of corrupting the angle measurements of those beacons. Fortunately, the system is able to detect these pathological cases and to reject measurements due to code collisions, unlike laser systems with retro-reflective tapes. However, in the case of BeAMS, there is a way to overcome this issue, as explained hereafter. In general, beacons are placed against walls or in corners, where it is unlikely to find the robot. Also, some papers discuss the issue of beacon placement. Algorithms to find the best place of a minimum number of beacons to meet a given criterion, most often a minimum positioning error, are proposed in [23, 78, 82]. For BeAMS, we could use such an algorithm, with only one additional constraint: anywhere on the moving area, the robot has to "see" at least 3 beacons, not aligned with any other beacon.

2.6 Conclusions

In this chapter, we present a new angle measurement system, named BeAMS, that can be used by an algorithm for mobile robot positioning based on angle measurements. First, we present some of the numerous angle measurement systems developed for robot positioning. It turns out that the most flexible solutions are rotating lasers with passive bar-coded reflective tapes or active emitting beacons with a rotating receiver. Commercial systems use the first solution, but in general, they do not identify beacons. The second solution is generally adopted by home-made systems. It requires to power-up the beacons, but it is easier to identify them. The knowledge of beacon IDs is an advantage to be robust to the wake-up or kidnapped issues, or to feed a data fusion algorithm without requiring a data association step.

Our motivation for this work was to create a new system fitted for the EUROBOT contest, which imposes many constraints. As a consequence, it was impossible to use a commercial sensor, and we decided to create our own system. The hardware of BeAMS consists of a sensor located on the robot, and several active beacons emitting infrared light in the horizontal plane, located at known positions. BeAMS has an acquisition rate of $10\,Hz$, and the entire sensor is contained in a $(8 \times 8 \times 6)\,cm^3$ volume. While the basic ideas are similar to existing systems, BeAMS innovates on many points. The mechanical part of the system is kept as simple as possible (motor only, no gear system or belt) due to the hollow shaft, and it does not need an optical encoder for the motor control or angle measurement. These features tend to reduce considerably the volume of BeAMS. A simple infrared receiver/demodulator is the main sensor for the angle measurements, and the beacons are common infrared LEDs. Furthermore, the system only requires one infrared communication channel, and there is no synchronization channel between the beacons and the robot. Each beacon emits its own code over the 455 kHz carrier wave; this emission is continuous so as to avoid having any form of synchronization between the beacons and the receiver. As a result, each beacon signal is a periodic signal whose period corresponds to a particular code defining the beacon ID. Finally, BeAMS introduces a new mechanism to measure angles: it detects a beacon when it enters and leaves an angular window. This allows the sensor to analyze the temporal evolution of the received signal inside the angular window. In our case, this particularity is used to decode the beacon ID.

Finally, we analyze some practical considerations and trades-off, due to the many parameters implied in the design of BeAMS. We explain how the turning speed, the angular window, the working distance, and the number of beacons are linked together. A simple analysis of the coding scheme of the beacons shows that the OOK modulation of the beacon signals

induces an error in the angle measurement. We show that the maximum absolute error on the beacon angle is given by $\phi_0/2$. An intuitive design rule would say that we have to reduce the duration of zeros, as well as their frequency of appearance. Also, we consider a practical system deployment. Whereas BeAMS has been designed for the EUROBOT contest, it can be used in any other positioning application involving angle measurements. In particular, we explain how to increase the covered area and the number of beacons, by modifying some parameters of the system.

Chapter 3

Error analysis

3.1 Introduction

Now that the hardware part of the system has been presented, we concentrate on the errors that affect angle measurements. We can identify two kinds of noise in BeAMS: (1) the natural noise, and (2) the artificial noise. Like for all other systems, BeAMS is affected by the natural noise, due to the receiver output jitter, rotation jitter, electronic noise, other infrared signals, etc. In addition to the natural noise, BeAMS is affected by another kind of noise, due to the codes and the use of an OOK modulation mechanism. In order to identify the different beacons, we decided that each beacon has to send its own coded signal. Unfortunately, this strategy produces errors when no signal is sent, that is during an OFF period of the sequence. In this chapter, this noise is interpreted as an additional (or artificial) noise due to the OOK modulation mechanism. But, unlike the natural noise, the artificial noise can be controlled, and it is important to evaluate the level of artificial error to guarantee the usability of BeAMS in real conditions. Therefore, we first focus on the artificial noise to evaluate the error made on measured angles resulting from the coding of beacon signals. The natural noise will be discussed in the next chapter. The goal of this chapter is to develop a theoretical model, useful for evaluating the performance of our system. We want to establish an upper bound of the artificial variance that affects the angle measurements, and to understand the impact of the zero symbol durations as well as their frequencies.

This chapter is organized as follows. In Section 3.2, we describe the errors due to the OOK modulation. Then, in Section 3.3, we present some notations, which will be used extensively in the two following chapters. In Section 3.4, we establish the probability density functions of the angular measurements of the first and last edges of the received signal. In Section 3.5, we characterize the estimator used for the beacon angle. In particular, we establish its mean, and an upper bound of its variance. Finally, we conclude the chapter in Section 3.6.

3.2 Description of the errors

As the receiver captures an OOK amplitude modulated signal, it can only detect the presence of the carrier wave (denoted by a 1 or ON period) or the absence of the carrier wave (denoted by a 0 or OFF period). If the carrier were sent continuously (that is, if the signal sent by a beacon was not coded), there would be no OFF periods in the signal captured by the receiver. But the coding of the beacon signals introduces zeros into the emitted sequences.

Let us now examine the influence of the OFF periods on the first rising and last falling edges. Since there are errors in the system, there are no means to access the true values of ϕ_R and ϕ_F. Therefore, we consider random variables instead, denoted by Φ_r and Φ_f. According to equation (2.5), we propose the following estimator Φ_b for the beacon angle ϕ_B

$$\Phi_b = \frac{\Phi_r + \Phi_f}{2}. \tag{3.1}$$

As illustrated in Figure 3.1, if a beacon emits a 1 when it enters into the angular window, there is no error on Φ_r, meaning that the measured value ϕ_R is a correct estimate of Φ_r. However, if a beacon emits a 0 when it enters the angular window, there is an error on Φ_r because the receiver misses the actual $0 \rightarrow 1$ transition. In fact the transition occurs later ($\Phi_r \geq \phi_R$), at the next 1. The same consideration applies to Φ_f, except that the $1 \rightarrow 0$ transition could occur sooner ($\Phi_f \leq \phi_F$). All these specific situations are illustrated in Figure 3.1. We first represent the output of the receiver for a non modulated carrier wave, R_0. In that case, there are no errors in the transition times because the beacon sends out a continuous 1 symbol. The four other cases represent the output of the receiver for four different situations using an arbitrary code (we use here a simpler code than ours for the purpose of illustration, but this does not change the conclusions). The first case, corresponding to the received signal R_1, does not induce any error because P_{th} is crossed upwards and downwards when the beacon emits a 1 symbol. The second case (R_2) generates an error on Φ_r only. The third case (R_3) generates an error on Φ_f only, and the fourth case (R_4) generates an error on both Φ_r and Φ_f. From Figure 3.1, one can see that the receiver output R_i is the logical AND between E_i and R_0. Of course, this hypothesis corresponds to an ideal receiver, and will be discussed later.

Assume now that the OFF periods of a sequence all have the same duration, denoted by T_0 (this is our choice by design). Because the motor rotates at a constant speed, an OFF period is then equivalent to an OFF angle called ϕ_0. The worst case for estimating Φ_r occurs when an OFF period starts at an angle $\phi = \phi_R$, delaying the next transition to an angle $\phi_R + \phi_0$. The same reasoning applies to Φ_f when an OFF period begins at an angle $\phi = \phi_F - \phi_0$. In both cases, the maximum absolute error on Φ_r or Φ_f is equal to ϕ_0. These are the worst cases but there are many combinations of these two errors. In the following sections, we establish the probability density functions (*PDFs*) of the random variables Φ_r and Φ_f, and derive characteristics of the estimator Φ_b.

3.3 Notations

For the following developments, we need to define some notations:

- N_0, N_1 are the number of zeros or ones in a code, respectively. The number of bits in a code is $N_b = N_0 + N_1$.

- p_0, p_1 are the probabilities of obtaining a 0 or a 1 respectively at the IR power threshold (rising or falling edge), that is their frequencies[1]. By definition we have $p_0 = {}^{N_0}/_{N_b}$, $p_1 = {}^{N_1}/_{N_b}$, and $p_0 + p_1 = 1$.

[1]For this, we assume that the power sent by the beacon is higher than the threshold P_{th}. In other words, it means that we are in the working range of the sensor.

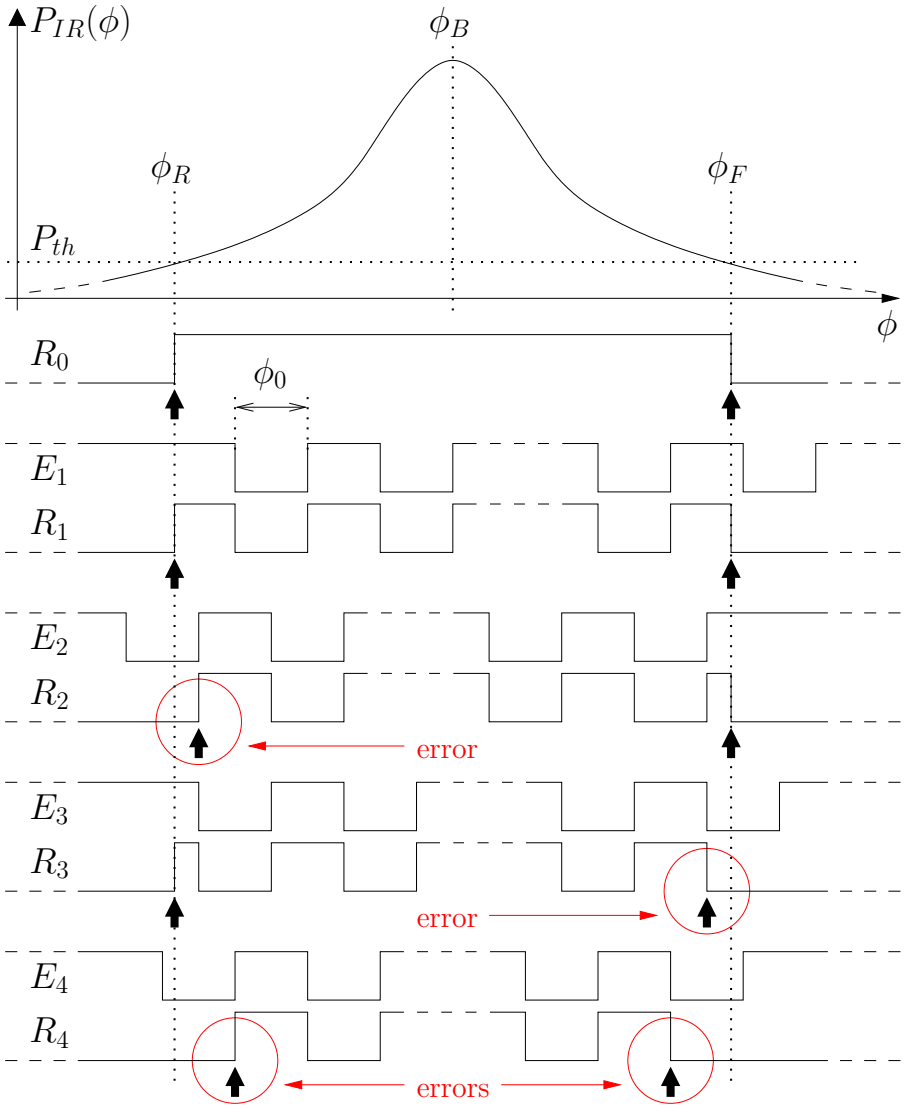

Figure 3.1: The upper curve $P_{IR}(\phi)$ is the infrared power collected at the receiver while the turret is turning. R_0 is the special case corresponding to the non modulated infrared carrier wave (no OFF periods). E_i are examples of emitted signals from the beacons. R_i are the corresponding received signals at the receiver output. The black arrows represent the measured values respectively for Φ_r to the left (first *Rising* edge) and for Φ_f to the right (last *Falling* edge). The encircled arrows emphasize errors made on Φ_r or Φ_f.

- T_0 is the OFF period (duration of a 0) in a code. The only assumption is that the OFF periods of a code must all have the same duration.

- ϕ_0 is the OFF angle. It corresponds to the angular displacement of the turret during the OFF period T_0:

$$\phi_0 = \omega\, T_0. \tag{3.2}$$

  To give an example of real values, in the original setup of our system for mobile robot positioning, we have: $N_1 = 10$, $N_0 = 2$, $p_0 = 1/6$, $\omega = 10.016\,\text{turn/s}$, $T_0 = 30.8\,\mu s$, and $\phi_0 = 0.111\,deg$, for each code.

- The Uniform *PDF* is defined as

$$U_{(a,b)}(x) = \begin{cases} \frac{1}{b-a} & \text{if } a \le x \le b, \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

  Its mean is equal to $\frac{a+b}{2}$ and its variance is $\frac{(b-a)^2}{12}$.

- The symmetric Triangular *PDF* is defined as

$$T_{(a,b)}(x) = \begin{cases} \frac{2}{b-a} - \frac{2|2x-a-b|}{(b-a)^2} & \text{if } a \le x \le b, \\ 0 & \text{otherwise,} \end{cases} \tag{3.4}$$

  where $|x|$ denotes the absolute value of x. Its mean is $\frac{a+b}{2}$ and its variance is $\frac{(b-a)^2}{24}$. Note that with these notations, and if $b - a = d - c$, we have [63, page 137]

$$U_{(a,b)}(x) \otimes U_{(c,d)}(x) = T_{(a+c,\, b+d)}(x), \tag{3.5}$$

  where $\otimes$ denotes the convolution product.

3.4 Probability density functions

BeAMS introduces a new mechanism for measuring angles. To the contrary of systems that look for a maximum to estimate the angle of a beacon, BeAMS detects a beacon when it enters and when it leaves the angular window. Therefore, we have two random variables, Φ_r and Φ_f, corresponding to these events. The estimator of the beacon angle, Φ_b, is the mean of these two variables.

In this section, we establish the probability density functions of Φ_r and Φ_f. Obviously, there are some symmetries for Φ_r and Φ_f; we will use them to shorten some developments.

3.4.1 Probability density function of Φ_r

Errors on Φ_r originate if a beacon emits a 0 symbol while entering the angular window. Assuming time stationarity and as there is no synchronization between the beacons and the receiver, p_1 is the probability of determining the correct angle ϕ_R as the measured value for Φ_r, when the beacon enters the angular window. When the beacon emits a 0, the value measured for Φ_r is not correct; we then assume that its value is uniformly distributed between

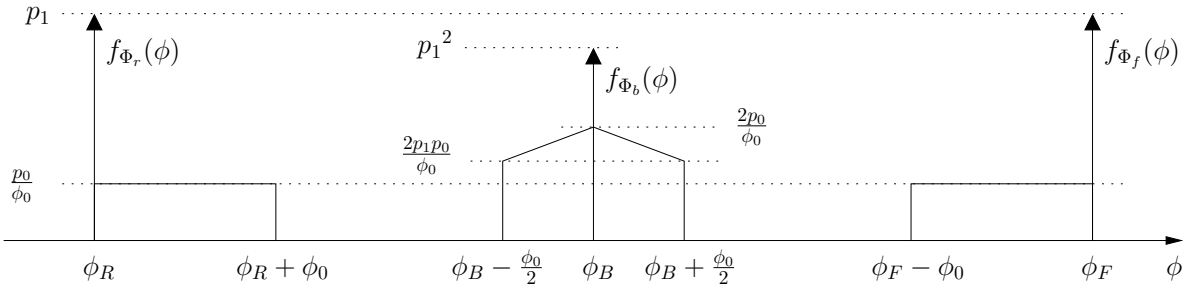

Figure 3.2: Probability density functions of Φ_r (left), Φ_f (right) and Φ_b (center) in the case of independent Φ_r and Φ_f.

ϕ_R and $\phi_R + \phi_0$. Therefore, if we define $\delta(x)$ as the DIRAC delta function, then the *PDF* of Φ_r is given by the following mixture of *PDFs*

$$f_{\Phi_r}(\phi) = p_1 \delta(\phi - \phi_R) + p_0 U_{(\phi_R, \phi_R+\phi_0)}(\phi), \tag{3.6}$$

for $\phi \in [-\pi, \pi)$. The mean and variance of Φ_r are, respectively,

$$\mu_{\Phi_r} = \phi_R + p_0 \frac{\phi_0}{2}, \tag{3.7}$$

$$\sigma^2_{\Phi_r} = p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}. \tag{3.8}$$

The details of the calculus can be found in Section A.2.1.

3.4.2 Probability density function of Φ_f

Because the configuration is symmetric when the beacon exits the angular window, a similar result yields for Φ_f

$$f_{\Phi_f}(\phi) = p_1 \delta(\phi - \phi_F) + p_0 U_{(\phi_F-\phi_0, \phi_F)}(\phi), \tag{3.9}$$

for $\phi \in [-\pi, \pi)$. The mean and variance of Φ_f are

$$\mu_{\Phi_f} = \phi_F - p_0 \frac{\phi_0}{2}, \tag{3.10}$$

$$\sigma^2_{\Phi_f} = p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}. \tag{3.11}$$

The details of the calculus can be found in Section A.2.2.

3.4.3 Characteristics of the estimators Φ_r and Φ_f

The *PDFs* of Φ_r and Φ_f are drawn in Figure 3.2. The expectations of Φ_r and Φ_f have a bias given by $\pm p_0 \frac{\phi_0}{2}$ respectively (see equations (3.7) and (3.10)). The bias is proportional to the OFF angle ϕ_0 and the proportion of zeros in a code p_0. The variances of Φ_r and Φ_f are equal, and they are equal to zero if and only if there is no OFF period in the codes[2].

[2]OFF periods correspond to the absence of any signal. The presence of OFF periods is nevertheless necessary because we need to code the signal to identify beacons. If the identifier is not contained inside the signal (in-band signaling), then we need an additional signal to obtain the beacon ID (out-band signaling). Out-band signaling systems are more complex and have their own problems, not easy to solve for mobile robot positioning. BeAMS uses in-band signaling.

3.5 Characterization of the estimator Φ_b

3.5.1 Mean of Φ_b

The aim of the system being to estimate the beacon angle ϕ_B, we are now interested in finding the mean and variance of Φ_b. Generally the mean and variance of a random variable are calculated with the help of the *PDF*. In the case of Φ_b, it is not necessary to derive them from the *PDF* since the estimator is a function of Φ_r and Φ_f (equation (3.1)), whose *PDFs* are known. Let us first consider the mean of Φ_b. For any random variables X and Y, we have $E\{X+Y\} = E\{X\} + E\{Y\}$ (see for example [63, page 152]). Therefore, the mean of Φ_b is given by

$$\begin{aligned} \mu_{\Phi_b} &= \frac{E\{\Phi_r\} + E\{\Phi_f\}}{2}, \\ &= \frac{\left(\phi_R + p_0\frac{\phi_0}{2}\right) + \left(\phi_F - p_0\frac{\phi_0}{2}\right)}{2}, \\ &= \frac{\phi_R + \phi_F}{2} = \phi_B. \end{aligned} \tag{3.12}$$

As can be seen, the mean of Φ_b is unbiased, despite that both the entering angle Φ_r and leaving angle Φ_f estimators are biased. This justifies the construction of a symmetric receiver and the use of that estimator.

3.5.2 Variance of Φ_b

Let us now derive the variance of Φ_b. The variance of the sum of two random variables can be expanded as [63]

$$\sigma^2_{\Phi_b} = var\left\{\frac{\Phi_r + \Phi_f}{2}\right\} = \frac{var\{\Phi_r + \Phi_f\}}{4} = \frac{\sigma^2_{\Phi_r} + \sigma^2_{\Phi_f} + 2C\{\Phi_r, \Phi_f\}}{4}, \tag{3.13}$$

where $C\{\Phi_r, \Phi_f\}$ denotes the covariance of Φ_r and Φ_f. If Φ_r and Φ_f are uncorrelated, we have that [63, page 155]

$$\sigma^2_{\Phi_b} = \frac{\sigma^2_{\Phi_r} + \sigma^2_{\Phi_f}}{4} = \frac{\sigma^2_{\Phi_r}}{2} = \frac{\sigma^2_{\Phi_f}}{2}, \tag{3.14}$$

since $\sigma^2_{\Phi_r} = \sigma^2_{\Phi_f}$. This could also have been derived from the *PDF* of Φ_b, that is given by, in the case of independent Φ_r and Φ_f,

$$f_{\Phi_b}(\phi) = p_1^2\delta(\phi - \phi_B) + 2p_1p_0\,U_{\left(\phi_B - \frac{\phi_0}{2}, \phi_B + \frac{\phi_0}{2}\right)}(\phi) + p_0^2\,T_{\left(\phi_B - \frac{\phi_0}{2}, \phi_B + \frac{\phi_0}{2}\right)}(\phi). \tag{3.15}$$

This result is obtained by convolving the *PDFs* of Φ_r and Φ_f [63, page 136], using equation (3.5) and rescaling the result by using these properties [63]: 1) if $Y = \alpha X$, then $f_Y(y) = \frac{1}{|\alpha|} f_X\left(\frac{y}{\alpha}\right)$, and 2) $\delta(\alpha x) = \frac{1}{|\alpha|}\delta(x)$. This probability density function is also depicted in Figure 3.2 (center). However, the non correlation or independence of Φ_r and Φ_f are questionable in our case; this is discussed hereafter.

As explained earlier, four situations are possible in one angular window: (1) no error is encountered, (2) an error occurs for Φ_r only, (3) an error occurs for Φ_f only, or (4) an error occurs for both angles. But the codes are deterministic and not random, and the durations

between OFF periods are fixed and known. So, depending on the rotating speed and the code, it is not sure that an error is possible on Φ_r and Φ_f simultaneously. These remarks show that the Φ_r and Φ_f variables are not independent, and that the nature of the relationship depends on the angular window and the coding scheme. To establish this relationship, we should analyze, in full details, the four previous cases in function of the angular window and the different codes. However, the mean of Φ_b is always given by equation (3.12), and despite the relationship between Φ_r and Φ_f, Φ_b remains unbiased. To the contrary, the variance of Φ_b is no longer given by equation (3.14) when Φ_r and Φ_f are correlated.

Fortunately, it is possible to derive an upper bound for $\sigma^2_{\Phi_b}$ for a practical use, as we did in [66]. Indeed, the square of the covariance is upper bounded [63, page 153]

$$C^2\{\Phi_r, \Phi_f\} \leq \sigma^2_{\Phi_r}\sigma^2_{\Phi_f}. \tag{3.16}$$

Given that $\sigma^2_{\Phi_r} = \sigma^2_{\Phi_f}$, we combine equations (3.13) and (3.16) to establish the following limits for $\sigma^2_{\Phi_b}$

$$0 \leq \sigma^2_{\Phi_b} \leq \sigma^2_{\Phi_r}. \tag{3.17}$$

The upper bound of $\sigma^2_{\Phi_b}$ is given by

$$\sigma^2_{\Phi_b} \leq \sigma^2_{\Phi_r} = \sigma^2_{\Phi_f} = p_0\frac{\phi_0^2}{3} - p_0^2\frac{\phi_0^2}{4}. \tag{3.18}$$

The upper bound for $C\{\Phi_r, \Phi_f\}$ is a universal bound because it does not make any guess about a possible relationship between the random variables. This result is confirmed by the simulations but it seems to over estimate the real upper bound of $\sigma^2_{\Phi_b}$. Indeed, this first upper bound does not take into account the particular nature of the committed errors. The purpose of the following section is to provide a more accurate result.

3.5.3 Computation of the upper bound of $\sigma^2_{\Phi_b}$

With respect to the relationship between Φ_r and Φ_f, four important points should be noted:

1. the *PDFs* of Φ_r and Φ_f remain correct, as well as their means and variances.
2. the estimator Φ_b remains unbiased, since the expectation does not depend on the relationship between variables.
3. the upper bound computed in the last section also remains correct, even if it tends to over estimate the variance of Φ_b.
4. the support of the *PDF* of Φ_b is always equal to $\left[\phi_B - \frac{\phi_0}{2}, \phi_B + \frac{\phi_0}{2}\right]$, whatever the relationship between Φ_r and Φ_f.

In order to understand the link between Φ_r and Φ_f, and to find out a more accurate result on the upper bound of $\sigma^2_{\Phi_b}$, we have to analyze the four previous cases more cautiously in function of the angular window and the different codes. From a mathematical point of view, we need to compute the covariance $C\{\Phi_r, \Phi_f\}$ for all possible cases, and put the result back into equation (3.13). Note that the covariance can be expanded as [63, page 152]

$$C\{\Phi_r, \Phi_f\} = E\{\Phi_r, \Phi_f\} - E\{\Phi_r\}E\{\Phi_f\}. \tag{3.19}$$

Since $E\{\Phi_r\}$ and $E\{\Phi_f\}$ are known (see equations (3.7) and (3.10)), we need to compute the joint expectation $E\{\Phi_r, \Phi_f\}$. And to compute the joint expectation, we need to express the joint *PDF* $f_{\Phi_r\Phi_f}(\phi_r, \phi_f)$.

3.5.3.1 Modified random variables

In order to simplify the calculus, we define the modified random variables $\mathcal{E}_r$ and $\mathcal{E}_f$ as follows

$$\mathcal{E}_r = \Phi_r - \phi_R, \tag{3.20}$$
$$\mathcal{E}_f = \Phi_f - \phi_F. \tag{3.21}$$

$\mathcal{E}_r$ and $\mathcal{E}_f$ are shifted versions of Φ_r and Φ_f, by an amount equal to ϕ_R, or ϕ_F respectively, and, as a result, they represent the errors committed on these measurements. The *PDFs* of these new random variables are

$$f_{\mathcal{E}_r}(\epsilon_r) = p_1\delta(\epsilon_r) + p_0\, U_{(0,\,\phi_0)}(\epsilon_r), \tag{3.22}$$

$$f_{\mathcal{E}_f}(\epsilon_f) = p_1\delta(\epsilon_f) + p_0\, U_{(-\phi_0,\,0)}(\epsilon_f), \tag{3.23}$$

and their expectations are

$$\mu_{\mathcal{E}_r} = p_0\frac{\phi_0}{2}, \tag{3.24}$$
$$\mu_{\mathcal{E}_f} = -p_0\frac{\phi_0}{2}. \tag{3.25}$$

The variances are unaltered since

$$\sigma^2_{\mathcal{E}_r} = var\{\mathcal{E}_r\} = var\{\Phi_r - \phi_R\} = var\{\Phi_r\} = p_0\frac{\phi_0^2}{3} - p_0^2\frac{\phi_0^2}{4} = \sigma^2_{\mathcal{E}_f}. \tag{3.26}$$

Finally, we have

$$C\{\mathcal{E}_r, \mathcal{E}_f\} = C\{\Phi_r - \phi_R, \Phi_f - \phi_F\} = C\{\Phi_r, \Phi_f\}. \tag{3.27}$$

The variance of Φ_b, expressed in terms of these new random variables $\mathcal{E}_r$ and $\mathcal{E}_f$, is then given by

$$\begin{aligned}\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\Phi_r} + \sigma^2_{\Phi_f} + 2C\{\Phi_r, \Phi_f\}}{4} \\ &= \frac{\sigma^2_{\mathcal{E}_r} + \sigma^2_{\mathcal{E}_f} + 2C\{\mathcal{E}_r, \mathcal{E}_f\}}{4} \\ &= \frac{2\sigma^2_{\mathcal{E}_r} + 2C\{\mathcal{E}_r, \mathcal{E}_f\}}{4} \\ &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r, \mathcal{E}_f\}}{2}.\end{aligned} \tag{3.28}$$

By definition, the covariance of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by

$$C\{\mathcal{E}_r, \mathcal{E}_f\} = E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\}E\{\mathcal{E}_f\}. \tag{3.29}$$

Therefore, we obtain

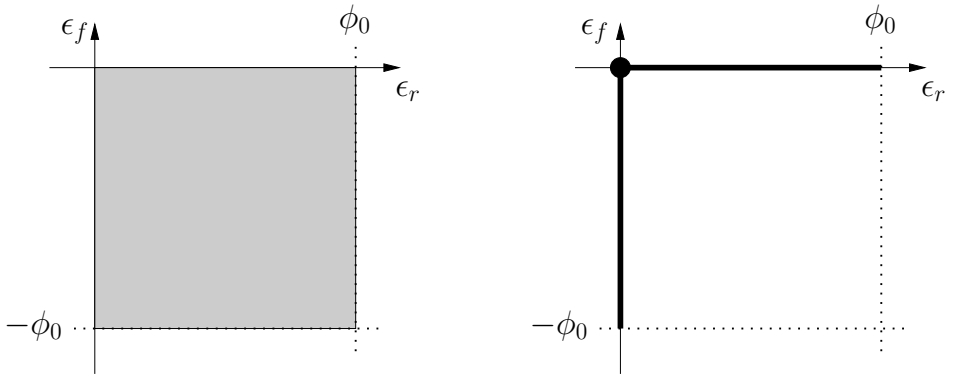

Figure 3.3: Left hand side: possible support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ in the general case. Right hand side: support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ when no error is possible on Φ_r and Φ_f simultaneously.

$$\sigma^2_{\Phi_b} = \frac{\sigma^2_{\mathcal{E}_r} + E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\}E\{\mathcal{E}_f\}}{2} \tag{3.30}$$

$$= \frac{\sigma^2_{\mathcal{E}_r}}{2} + \frac{E\{\mathcal{E}_r, \mathcal{E}_f\}}{2} - \frac{E\{\mathcal{E}_r\}E\{\mathcal{E}_f\}}{2} \tag{3.31}$$

$$= p_0\frac{\phi_0^2}{6} - p_0^2\frac{\phi_0^2}{8} + \frac{E\{\mathcal{E}_r, \mathcal{E}_f\}}{2} + p_0^2\frac{\phi_0^2}{8} \tag{3.32}$$

$$= p_0\frac{\phi_0^2}{6} + \frac{E\{\mathcal{E}_r, \mathcal{E}_f\}}{2}. \tag{3.33}$$

The challenge is to compute the joint expectation $E\{\mathcal{E}_r, \mathcal{E}_f\}$

$$E\{\mathcal{E}_r, \mathcal{E}_f\} = \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\, f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)\, d\epsilon_r\, d\epsilon_f. \tag{3.34}$$

For this, we need to express $f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)$ for the different codes and angular windows.

3.5.3.2 Towards a more realistic upper bound on $\sigma^2_{\Phi_b}$

In order to compute a more realistic upper bound, we have to take into account the particular nature of the errors $\mathcal{E}_r$ and $\mathcal{E}_f$, and, more specifically, their bounds. A closer look onto the $\mathcal{E}_r$ and $\mathcal{E}_f$ definitions, or their *PDFs* shows that we have

$$0 \leq \mathcal{E}_r \leq \phi_0, \tag{3.35}$$

$$-\phi_0 \leq \mathcal{E}_f \leq 0. \tag{3.36}$$

Graphically, it means that the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is not null only inside a square of side ϕ_0, as shown in Figure 3.3 (left). Moreover, their product is always negative or null

$$\mathcal{E}_r\mathcal{E}_f \leq 0, \tag{3.37}$$

and so must be their joint expectation

$$E\{\mathcal{E}_r, \mathcal{E}_f\} \leq 0. \tag{3.38}$$

	first upper bound on σ_{Φ_b}	new upper bound on σ_{Φ_b}
expression	$\sigma_{\Phi_b} \leq \phi_0 \sqrt{\frac{p_0}{3} - \frac{p_0^2}{4}}$	$\sigma_{\Phi_b} \leq \phi_0 \sqrt{\frac{p_0}{6}}$
numerical value	$0.0245\,deg$	$0.0185\,deg$

Table 3.1: Comparison of two bounds on σ_{Φ_b}.

Therefore

$$\max E\{\mathcal{E}_r, \mathcal{E}_f\} = 0, \tag{3.39}$$

regardless of the relationship between $\mathcal{E}_r$ and $\mathcal{E}_f$.

We can therefore derive the following upper bound from expression (3.33), which is reminded hereafter

$$\sigma_{\Phi_b}^2 = p_0 \frac{\phi_0^2}{6} + \frac{E\{\mathcal{E}_r, \mathcal{E}_f\}}{2}. \tag{3.40}$$

Theorem 3.1 (Upper bound of $\sigma_{\Phi_b}^2$)**.** *For all codes, the variance of Φ_b is bounded by $p_0 \frac{\phi_0^2}{6}$, as long as the OFF periods of the codes all have the same duration*

$$\sigma_{\Phi_b}^2 \leq p_0 \frac{\phi_0^2}{6}. \tag{3.41}$$

As expected, the variance is related to the presence of OFF periods in the codes. More precisely, the variance is proportional to the probability of having a zero p_0, and to the square of the OFF angle ϕ_0. It is equal to zero if and only if there is no OFF period in the codes. So, this expression establishes that p_0 and ϕ_0 should be kept as small as possible to minimize the effects of the OOK modulation. Note that this variance is an upper bound, since it represents the worst case (no error on Φ_r and Φ_f simultaneously), and that this upper bound is the same for all codes (since they all have the same p_0 and ϕ_0 by design). Note also that this is a general result, as long as the OFF periods of the codes all have the same duration.

Numerical values. In our case, $p_0 = 1/6$ and $\phi_0 = 0.111\,deg$, for each code. Previously, according to expression (3.18), the standard deviation was lower than 0.0245 deg. This new upper bound implies that $\sigma_{\Phi_b} \leq 0.0185$ deg; this is a decrease of about 25 %. These results are summarized in Table 3.1.

Finally, it is interesting to interpret the condition $E\{\mathcal{E}_r, \mathcal{E}_f\} = 0$, that is the maximum of the joint expectation. This constraint, combined with constraint (3.37) means that either $\mathcal{E}_r$ or $\mathcal{E}_f$ must be null. In other terms, it is impossible to make an error on both Φ_r and Φ_f simultaneously. Intuitively, it is logical that the variance is maximum in that case since an error committed on Φ_r is not balanced by an error on Φ_f, and vice versa. Graphically, it means that the support of the joint *PDF* reduces to the axes in that case. The possible support of the joint *PDF* is represented in Figure 3.3 (right). In the next section, we show that it can be composed of uniform *PDFs* along the axes (the horizontal and vertical line segments) and a two dimensional Dirac *PDF* (the black dot).

3.6 Conclusions

In this chapter, we provide a theoretical framework to analyze the artificial errors on the measured angles, resulting from the use of an On-Off Keying modulation mechanism. The

advantage of having a model is that we can understand and predict the use of other codes in the system. We establish the probability density functions of Φ_r and Φ_f, the estimators used for the entering and leaving angles, respectively. Then, a statistical estimator Φ_b for the angular localization of a beacon is proposed. We demonstrate that this estimator is unbiased and that its variance is upper bounded by $p_0\frac{\phi_0^2}{6}$. This variance represents the power of artificial noise due to the OOK modulation. It increases with the square of the OFF angle ϕ_0 (the angle corresponding to the OFF duration in a code) and with the proportion of zero symbols in a code p_0. This study has also justified some practical choices made in BeAMS, in particular: (1) the use of that particular estimator, (2) the reduction of p_0 versus p_1, and (3) the reduction of T_0 (or ϕ_0). Finally, we show that the upper bound corresponds to a situation in which it is impossible to make an error on both Φ_r and Φ_f simultaneously. This confirms that there is a statistical relationship between these two estimators, which is studied in the next chapter.

Chapter 4

Code statistics

4.1 Introduction

In this chapter, we complement the previous results by going into further details related to the code statistics of modulated signals in general, with an emphasis on BeAMS. In the previous chapter, we have computed the upper bound on the variance of Φ_b. This upper bound is a general result for all codes with the sole assumption *that the OFF periods of all codes must have the same duration.* We have also shown that the Φ_r and Φ_f variables are not independent, and that the nature of the relationship depends on the angular window and the coding scheme.

But, if we can provide the upper bound for all codes, the previous analysis does not tell for which angular windows this upper bound is reached. To determine them, we need to consider both the particular code and angular window. More generally, we need to express $f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)$ for the different codes and angular windows. Note that we did not need to express $f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)$ in order to find the upper bound. The final goal is to compute the real variance of Φ_b, for any code, and for any angular window, in order to show how the variance evolves exactly as a function of the angular window (while remaining below the upper bound).

This chapter is organized as follows. In Section 4.2, we establish the real variance of Φ_b for our codes, and for all angular windows. Then, we present simulation results in Section 4.3, in order to validate our theory. In Section 4.4, we discuss an important hypothesis about our theory, and we conclude the chapter in Section 4.5.

4.2 Evolution of the variance of Φ_b with respect to the angular window

In this section, we establish the evolution of the variance of Φ_b with respect to the angular window. For this, we need to understand the relationship between $\mathcal{E}_r$ and $\mathcal{E}_f$ for the different codes and angular windows. The general mechanism to do this consists in moving a particular code from the left to the right, in front of a fixed angular window, and to report the values of the committed errors on the first and last edges. The situation is depicted in Figure 4.1.

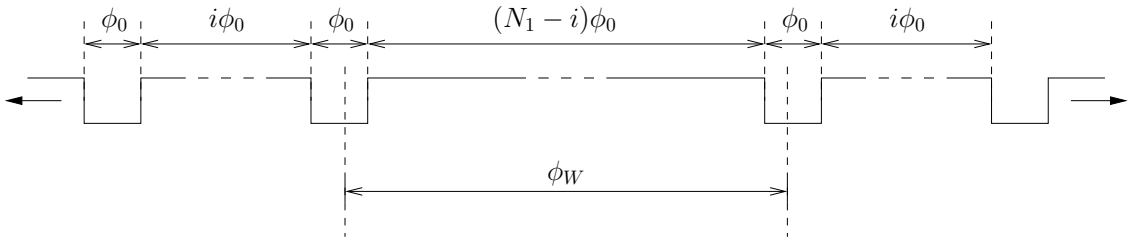

Figure 4.1: Illustration of the mechanism to understand the relationship between $\mathcal{E}_r$ and $\mathcal{E}_f$ for the different codes and angular windows. One has to move a particular code (defined as $C_i = [1^i\, 0^1\, 1^{10-i}\, 0^1]$, for $i = 1, \ldots, 5$) from the left to the right, in front of a fixed angular window, and to report the values of the errors.

4.2.1 Introduction to several situations

When we move a code from the left to the right, as represented in Figure 4.1, it appears that we have to deal with 6 different situations, corresponding to:

Case 0. This case occurs when the angular window ϕ_W is lower than ϕ_0, the OFF angle; that is $\phi_W = \lambda_0\phi_0$, with $\lambda_0 < 1$. It does not depend on any particular code (see Figure 4.1, with $\phi_W < \phi_0$). Errors are therefore bounded by the maximal value of the angular window ($\phi_W = \lambda_0\phi_0$)

$$0 \leq \quad \mathcal{E}_r \quad \leq \lambda_0\phi_0, \tag{4.1}$$

$$-\lambda_0\phi_0 \leq \quad \mathcal{E}_f \quad \leq 0. \tag{4.2}$$

The support of the joint *PDF* is similar to the one represented in Figure 3.3 (right), except that the parts along the axes are smaller. The support of the joint *PDF* is represented in Table 4.1, that regroups the supports of the *PDFs* of all the cases.

Case A. This case corresponds to the situation leading to a variance that is exactly equal to its upper bound, as determined in Section 3.5.3.2. It means that it is impossible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously, when the value of the angular window ϕ_W falls into given intervals, that depend on the particular code (see Figure 4.1, with $\phi_W = (i+3)\,\phi_0$, for example).

Case B. This case is a combination of case A and case D (see below). It appears for some values of the angular window that depend on the code. Since this case is a combination of two cases, its support is formed by the union of two supports.

Case C. This case is a combination of case A and case E (see below). It appears for some values of the angular window that depend on the code. Since this case is a combination of two cases, its support is formed by the union of two supports.

Case D. In that case, it is possible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously, in addition to the three other situations of case 0 and case A (no error, $\mathcal{E}_r \neq 0$ only, $\mathcal{E}_f \neq 0$ only). But, when this situation occurs, the errors are totally dependent. In particular, their difference is constant along a part of the support of the joint *PDF*. It appears for values of ϕ_W, that depend on the particular code (see Figure 4.1 for an example of this situation).

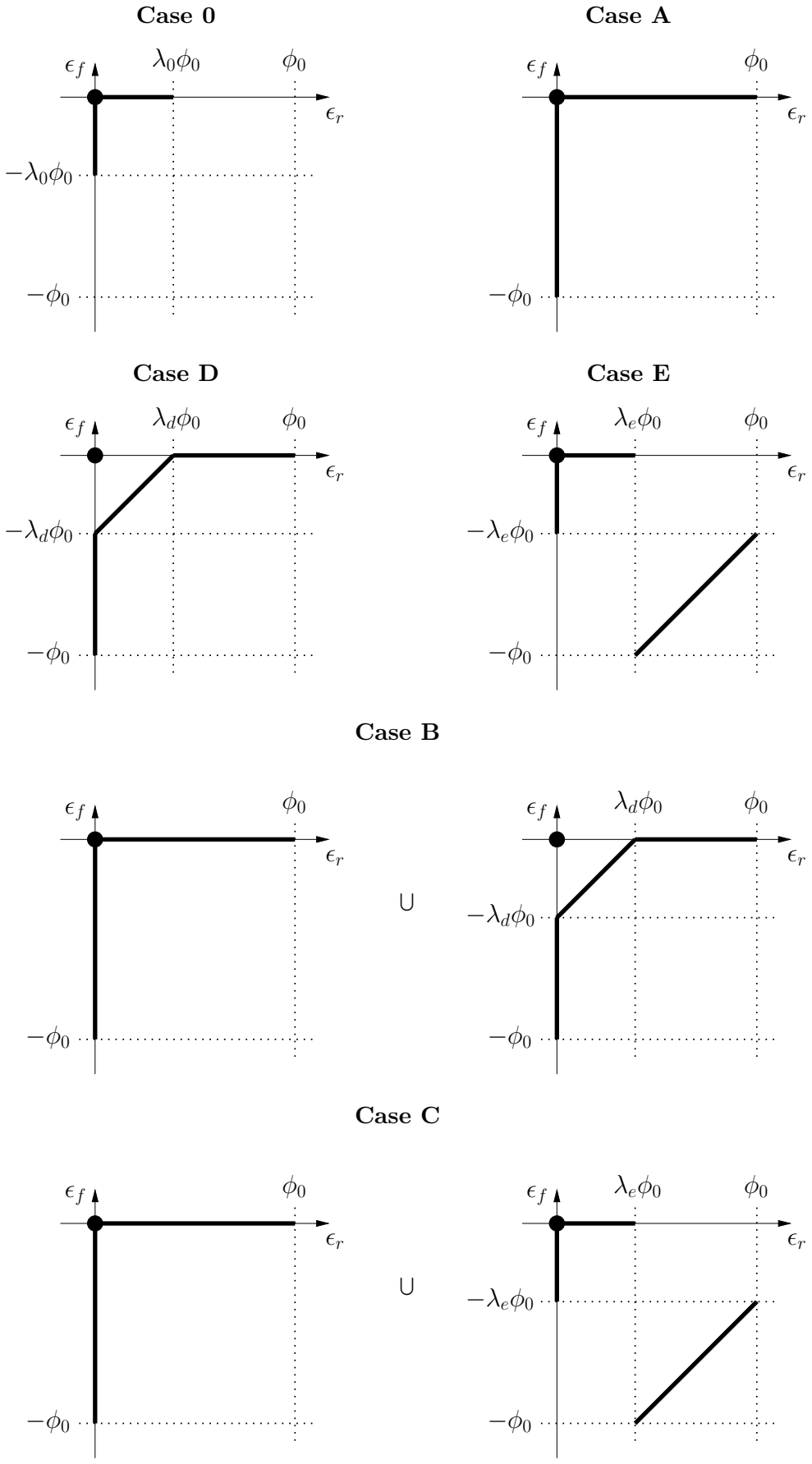

Table 4.1: Supports of the joint *PDFs* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the 6 different Cases.

Case E. This case is similar to case D, meaning that along a part of the support of the joint *PDF*, the errors are totally dependent. But, unlike case D, this situation occurs in another part of the joint *PDF*. It appears for values of ϕ_W, that depend on the particular code.

These situations, named "Cases", have different supports for the joint *PDF* (as established later in this document); these supports are shown in Table 4.1. It is important to understand that, while the absolute error on $\mathcal{E}_r$ or $\mathcal{E}_f$ is ϕ_0, the maximum error on Φ_b is not ϕ_0 but $\phi_0/2$ (due to the algebraic mean). In other words, the errors on $\mathcal{E}_r$ and $\mathcal{E}_f$ do not sum up, but they compensate. This explains the particular shape of the domain of the different *PDFs*. Note that, while the supports of the *PDFs* are continuous from a theoretical perspective, observed values are generally different for each revolution of the turret, and that successive observations move along the supports of the *PDFs*. We elaborate on these aspects in subsection 4.4.

From a practical point of view, BeAMS operates with the best upper bound for Φ_b all over the plane (regardless of the robot position). But this practice is far from being optimal. Indeed, we show that the real variance evolves with the position of the robot in the plane, and therefore the upper bound overestimates the real variance unnecessarily. It would be better to have the exact variance for each position in the plane, and for each beacon (because the variances are related to the code and to the angular window).

It is essential to note that the variance $\sigma^2_{\Phi_b}$ is not a function of the distance to a beacon, but to the seen angular window. This might not be intuitive, but this is coherent and more tractable for *an angular measurement system* that *should not be sensitive to the distance*. The major reason is that the relationship between the physical notion of distance and the measures is dependent on parameters, like power thresholds or propagation laws, that are out of control. The fact that $\sigma^2_{\Phi_b}$ is only dependent on the measured angular window and the beacon code (which is known) is an advantage of BeAMS compared to other angle measurement systems.

In the following, we first establish the probability density functions and variances for all the Cases. Then, we summarize the different Cases in Section 4.2.3. Finally, we present the simulations results in Section 4.3.

4.2.2 Study of the different cases

Before starting the detailed study, one has to remember the general results obtained so far:

1. the upper bound is always valid, and corresponds to case A.
2. the mean of Φ_b remains unbiased, for all cases.
3. the marginal *PDFs* of the joint *PDF* are always given by equations (3.22) and (3.23), for all cases, except for the case 0.

In addition to these remarks, we have noted that:

1. the value of the variance of Φ_b is a function of the particular code, and the angular window ϕ_W. The only constraint on ϕ_W is that it is positive or null by design ($\Phi_f \geq \Phi_r$)

$$\phi_W \geq 0. \tag{4.3}$$

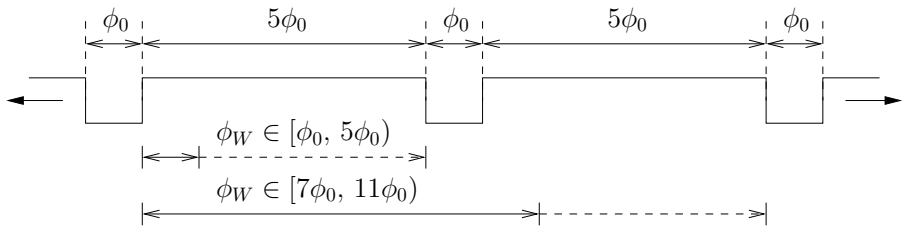

Figure 4.2: Finding the intervals for the Case A, and for the code 5.

2. as the codes are periodic, the evolution of the variance with respect to ϕ_W for each code must be periodic. But, as a consequence of constraint (4.3), the function is not periodic in the strict mathematical sense. Indeed, we will show that this function is partially periodic, for $\phi_W \geq \phi_0$ only

$$\sigma^2_{\Phi_b}(\phi_W) = \sigma^2_{\Phi_b}(\phi_W + P) \qquad \phi_W \geq \phi_0, \tag{4.4}$$

where the period P is related to the length of a code: $P = 12\phi_0$. As a consequence, we introduce a new notation, that will be used extensively in this chapter

$$\phi_W^{mod} = \phi_W \bmod (12\phi_0)\,. \tag{4.5}$$

3. in this study, we think in terms of angles instead of times since the OFF angle perceived by the system ($\phi_0 = \omega\, T_0$) is a combination of the zero duration (T_0) and the rotation speed of the turret (ω). Again, this is suitable for an angle measurement system like BeAMS.

4. the different cases are detailed hereafter in a different order than their logical name. Indeed, we detail them in order of complexity, and the way they are named will become clear later.

4.2.2.1 The case A

This case corresponds to the situation leading to the upper bound of the variance, as determined in Section 3.5.3.2. For each code, it is possible to find values for the angular window for which it is impossible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. Firstly, we determine the values of the angular window for which this situation occurs. We explain this reasoning with the code 5 since it is easier, and we will generalize the result later for the other codes. To find these values, one has to pick up a fixed value for the angular window, and to move virtually the code from the left to the right, and to check if the entering and leaving angles can both fall into the gaps (ϕ_0). Figure 4.2 illustrates the idea. From this analysis, one can see that, if the angular window ϕ_W is comprised between ϕ_0 and $5\phi_0$ or between $7\phi_0$ and $11\phi_0$, it is impossible to commit an error on both the entering and leaving angles. These intervals can be generalized for any code i, as given in Table 4.2. The principle is the same, but we have to replace the first 5 by i, and the second 5 by $10 - i$ (or $N_1 - i$), as a consequence of how to define the different codes

$$C_i = [1^i\, 0^1\, 1^{10-i}\, 0^1], \qquad i = 1, \ldots, 5. \tag{4.6}$$

	Ranges for ϕ_W^{mod}
Code 1	$[3\phi_0,\, 9\phi_0)$
Code 2	$[\phi_0,\, 2\phi_0) \bigcup [4\phi_0,\, 8\phi_0) \bigcup [10\phi_0,\, 11\phi_0)$
Code 3	$[\phi_0,\, 3\phi_0) \bigcup [5\phi_0,\, 7\phi_0) \bigcup [9\phi_0,\, 11\phi_0)$
Code 4	$[\phi_0,\, 4\phi_0) \bigcup [8\phi_0,\, 11\phi_0)$
Code 5	$[\phi_0,\, 5\phi_0) \bigcup [7\phi_0,\, 11\phi_0)$

Table 4.2: Intervals of the angular window for the Case A with respect to the codes ($\bigcup$ denotes the union).

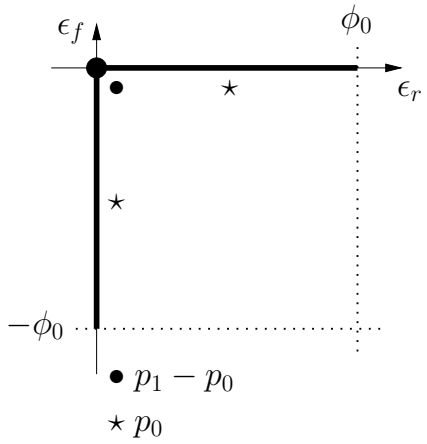

Figure 4.3: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case A.

The general case is depicted in Figure A.1, and further developed in Section A.3.1 of the appendix. We can summarize the intervals for the case A, and for all codes in the following compact form

$$\phi_W^{mod} \in [\phi_0,\, i\phi_0) \bigcup [(i+2)\phi_0,\, (10-i)\phi_0) \bigcup [(12-i)\phi_0,\, 11\phi_0), \qquad i = 1, \ldots, 5, \tag{4.7}$$

where $\bigcup$ denotes the union.

Now that the intervals have been determined, we are interested in finding the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$, in order to compute their joint expectation. We can represent the possible joint values of $\mathcal{E}_r$ and $\mathcal{E}_f$ on a 2D graphic where the abscissa ϵ_r represents a possible value of $\mathcal{E}_r$ ($\epsilon_r \in [0,\, \phi_0]$) and the ordinate ϵ_f represents a possible value of $\mathcal{E}_f$ ($\epsilon_f \in [-\phi_0,\, 0]$). Indeed, this graphic is the support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$. It is represented in Figure 4.3. The support of this joint *PDF* is composed of an horizontal line segment from 0 to ϕ_0 ($\mathcal{E}_r \neq 0$ only), a vertical line segment from $-\phi_0$ to 0 ($\mathcal{E}_f \neq 0$ only), and a black dot at the origin (no errors). The black dot represents a two dimensional Dirac *PDF*, and the line segments are uniform *PDFs*, as explained later. There is no part of the support outside the axes which means that is impossible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. The line segments are denoted by a black star ($\star$), and the origin by a black dot ($\bullet$).

Now, we have to associate probabilities and *PDFs* to these three parts. Assuming time stationarity and as there is no synchronization between the beacons and the receiver, we then assume that the value of $\mathcal{E}_r$ is uniformly distributed between 0 and ϕ_0, when $\mathcal{E}_f = 0$. By symmetry, we assume that the value of $\mathcal{E}_f$ is uniformly distributed between $-\phi_0$ and 0, when

$\mathcal{E}_r = 0$. Therefore, the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f) = p^{\bullet}\delta(\epsilon_r)\,\delta(\epsilon_f) + p^{\star}\delta(\epsilon_r)\; U_{(-\phi_0,0)}(\epsilon_f) + p^{\star}\delta(\epsilon_f)\; U_{(0,\phi_0)}(\epsilon_r), \tag{4.8}$$

where:

- $p^{\bullet}$ is the probability to commit no error, and
- $p^{\star}$ is the probability to have $\mathcal{E}_r \neq 0$ only. By symmetry, we show that the probability to have $\mathcal{E}_f \neq 0$ only is also given by $p^{\star}$, explaining why the notation $p^{\star}$ is the same for both.

The complete explanation about the computation of these probabilities is give in the appendix (Section A.3.1). Their values are given below

$$\begin{cases} p^{\bullet} = p_1 - p_0, \\ p^{\star} = p_0. \end{cases} \tag{4.9}$$

We can check that the probabilities for the case A sum up to 1

$$p^{\bullet} + p^{\star} + p^{\star} = p_1 - p_0 + p_0 + p_0 = p_1 + p_0 = 1. \tag{4.10}$$

As another verification, we can also compute the marginal *PDF* for $\mathcal{E}_r$

$$\begin{aligned} f_{\mathcal{E}_r}(\epsilon_r) &= \int_{-\infty}^{+\infty} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\, \epsilon_f)\; d\epsilon_f \\ &= (p_1 - p_0)\,\delta(\epsilon_r)\int_{-\infty}^{+\infty} \delta(\epsilon_f)\; d\epsilon_f \\ &\quad + p_0\delta(\epsilon_r)\int_{-\infty}^{+\infty} U_{(-\phi_0,0)}(\epsilon_f)\; d\epsilon_f \\ &\quad + p_0\, U_{(0,\phi_0)}(\epsilon_r)\int_{-\infty}^{+\infty} \delta(\epsilon_f)\; d\epsilon_f \\ &= (p_1 - p_0)\,\delta(\epsilon_r) + p_0\delta(\epsilon_r) + p_0\, U_{(0,\phi_0)}(\epsilon_r) \\ &= p_1\delta(\epsilon_r) + p_0\, U_{(0,\phi_0)}(\epsilon_r), \end{aligned} \tag{4.11}$$

which is the same as equation (3.22). Likewise, the marginal *PDF* for the random variable $\mathcal{E}_f$ is identical to equation (3.23). Now that we have the joint *PDF*, we can compute the joint expectation $E\{\mathcal{E}_r,\, \mathcal{E}_f\}$

$$\begin{aligned} E\{\mathcal{E}_r,\, \mathcal{E}_f\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\, f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\, \epsilon_f)\; d\epsilon_r\, d\epsilon_f \\ &= (p_1 - p_0)\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\,\delta(\epsilon_f)\; d\epsilon_r\, d\epsilon_f \\ &\quad + p_0\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\; U_{(-\phi_0,0)}(\epsilon_f)\; d\epsilon_r\, d\epsilon_f \\ &\quad + p_0\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_f)\; U_{(0,\phi_0)}(\epsilon_r)\; d\epsilon_r\, d\epsilon_f \\ &= 0 + 0 + 0 = 0. \end{aligned} \tag{4.12}$$

The joint expectation is null, meaning that random variables $\mathcal{E}_r$ and $\mathcal{E}_f$ are orthogonal for the case A. Then, we can compute $C\{\mathcal{E}_r, \mathcal{E}_f\}$ for the case A

$$\begin{aligned} C\{\mathcal{E}_r, \mathcal{E}_f\} &= E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\} E\{\mathcal{E}_f\} \\ &= 0 - \left(p_0 \frac{\phi_0}{2}\right)\left(-p_0 \frac{\phi_0}{2}\right) = p_0^2 \frac{\phi_0^2}{4}, \end{aligned} \tag{4.13}$$

and, finally, we can compute the variance of Φ_b for the case A

$$\begin{aligned} \sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r, \mathcal{E}_f\}}{2} \\ &= \frac{\left(p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}\right) + \left(p_0^2 \frac{\phi_0^2}{4}\right)}{2} \\ &= p_0 \frac{\phi_0^2}{6}. \end{aligned} \tag{4.14}$$

Note that this value corresponds to the upper bound defined in Section 3.5.3.2. As long as the angular window ϕ_W belongs to admissible ranges for the case A, the variance of Φ_b does not depend on the angular window.

4.2.2.2 The case 0

The case 0 occurs when the angular window ϕ_W is comprised between 0 and ϕ_0, for all codes

$$\phi_W \in [0, \phi_0). \tag{4.15}$$

This case is similar to the case A in that it is impossible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. But, unlike the case A, the intervals for the committed errors range from 0 to ϕ_W in absolute value

$$0 \leq \mathcal{E}_r \leq \phi_W, \tag{4.16}$$

$$-\phi_W \leq \mathcal{E}_f \leq 0. \tag{4.17}$$

Furthermore, there are parts where there is no detection at all, when the angular window is entirely comprised in a zero symbol duration. These comments show that the probabilities depend on the angular window. Since we have $\phi_W \in [0, \phi_0)$ for the case 0, let us introduce a parameter to express ϕ_W in function of ϕ_0

$$\phi_W = \lambda_0 \phi_0, \qquad \lambda_0 \in [0, 1). \tag{4.18}$$

The support of the joint *PDF* is represented in Figure 4.4. The support of this joint *PDF* is composed of an horizontal line segment from 0 to $\lambda_0\phi_0$ ($\mathcal{E}_r \neq 0$ only), a vertical line segment from 0 to $-\lambda_0\phi_0$ ($\mathcal{E}_f \neq 0$ only), and a black dot at the origin (no errors). The black dot represents a two dimensional Dirac *PDF*, and the line segments are uniform *PDFs*, as explained hereafter. There is no part of the support outside the axes which means that is impossible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. The line segments are denoted by an asterisk ($*$), and the origin by a black dot ($\bullet$).

Now, we have to associate probabilities and *PDFs* to these three parts. Assuming time stationarity and, as there is no synchronization between the beacons and the receiver, we

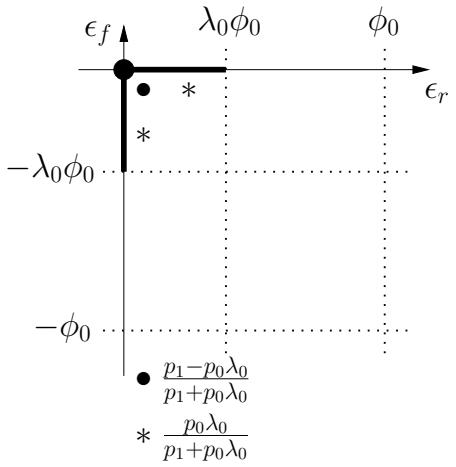

Figure 4.4: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case 0.

then assume that the value of $\mathcal{E}_r$ is uniformly distributed between 0 and $\lambda_0\phi_0$, when $\mathcal{E}_f = 0$. By symmetry, we assume that the value of $\mathcal{E}_f$ is uniformly distributed between $-\lambda_0\phi_0$ and 0, when $\mathcal{E}_r = 0$. Therefore, the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f) = p^\bullet \delta(\epsilon_r)\,\delta(\epsilon_f) + p^* \delta(\epsilon_r)\; U_{(-\lambda_0\phi_0,0)}(\epsilon_f) + p^* \delta(\epsilon_f)\; U_{(0,\lambda_0\phi_0)}(\epsilon_r), \tag{4.19}$$

where:

- $p^\bullet$ is the probability to commit no error, and
- p^* is the probability to have $\mathcal{E}_r \neq 0$ only. By symmetry, we show that the probability to have $\mathcal{E}_f \neq 0$ only is also given by p^*, explaining why the notation p^* is the same for both.

The complete explanation about the computation of these probabilities is given in the appendix (Section A.3.2). Their values are

$$\begin{cases} p^\bullet = \frac{p_1 - p_0\lambda_0}{p_1 + p_0\lambda_0}, \\ p^* = \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}. \end{cases} \tag{4.20}$$

We can check that the probabilities for the case 0 sum up to 1

$$p^\bullet + p^* + p^* = \frac{p_1 - p_0\lambda_0}{p_1 + p_0\lambda_0} + 2\frac{p_0\lambda_0}{p_1 + p_0\lambda_0} = \frac{p_1 + p_0\lambda_0}{p_1 + p_0\lambda_0} = 1. \tag{4.21}$$

Like for the case A, the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$, as well as the support of the joint *PDF* (Figure 4.4), show that $\mathcal{E}_r$ and $\mathcal{E}_f$ are orthogonal (see appendix A.3.2 for the details)

$$E\{\mathcal{E}_r, \mathcal{E}_f\} = \iint_{-\infty}^{+\infty} \epsilon_r\, \epsilon_f\, f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)\, d\epsilon_r\, d\epsilon_f = 0. \tag{4.22}$$

But, unlike the case A (and all other cases), the marginal *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ have changed, as well as their expectations and variances. The marginal *PDF* of $\mathcal{E}_r$ is given by (see appendix A.3.2 for more details)

$$\begin{aligned} f_{\mathcal{E}_r}(\epsilon_r) &= \int_{-\infty}^{+\infty} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)\, d\epsilon_f \\ &= \frac{p_1}{p_1 + p_0\lambda_0}\,\delta(\epsilon_r) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,U_{(0,\lambda_0\phi_0)}(\epsilon_r). \end{aligned} \tag{4.23}$$

The marginal *PDF* of $\mathcal{E}_f$ is given by

$$f_{\mathcal{E}_f}(\epsilon_f) = \frac{p_1}{p_1 + p_0\lambda_0}\,\delta(\epsilon_f) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,U_{(-\lambda_0\phi_0,0)}(\epsilon_f). \tag{4.24}$$

Their expectations are given by, using result A.7,

$$E\{\mathcal{E}_r\} = \frac{p_0\lambda_0}{(p_1 + p_0\lambda_0)}\,\frac{\lambda_0\phi_0}{2} = -E\{\mathcal{E}_f\}, \tag{4.25}$$

and their variances are given by, using result A.9,

$$var\{\mathcal{E}_r\} = var\{\mathcal{E}_f\} = \frac{(\lambda_0\phi_0)^2}{12}\,\frac{p_0\lambda_0\,(4p_1 + p_0\lambda_0)}{(p_1 + p_0\lambda_0)^2}. \tag{4.26}$$

Then, we can compute $C\{\mathcal{E}_r, \mathcal{E}_f\}$ for the case 0

$$\begin{aligned} C\{\mathcal{E}_r, \mathcal{E}_f\} &= E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\}\,E\{\mathcal{E}_f\} \\ &= \left(\frac{p_0\lambda_0}{(p_1 + p_0\lambda_0)}\,\frac{\lambda_0\phi_0}{2}\right)^2 = \frac{(p_0\lambda_0)^2}{(p_1 + p_0\lambda_0)^2}\,\frac{(\lambda_0\phi_0)^2}{4}, \end{aligned} \tag{4.27}$$

and, finally, we can obtain the variance of Φ_b for the case 0 (see appendix A.3.2)

$$\begin{aligned} \sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r, \mathcal{E}_f\}}{2} \\ &= p_0\frac{\phi_0^2}{6}\,\frac{\lambda_0^3}{(p_1 + p_0\lambda_0)} \\ &= p_0\frac{\phi_0^2}{6}P_0(\lambda_0), \end{aligned} \tag{4.28}$$

where the first part is the same as for the case A, and the second part is a function (polynomial ratio) of the parameter λ_0

$$P_0(\lambda_0) = \frac{\lambda_0^3}{p_1 + p_0\lambda_0}. \tag{4.29}$$

Note that $P_0(\lambda_0)$ satisfies

$$0 \le P_0(\lambda_0) < 1, \qquad \text{for } \lambda_0 \in [0,\, 1). \tag{4.30}$$

This function is equal to 0 when $\lambda_0 = 0$ (angular window reduced to 0), monotonically increases with λ_0, and is equal to 1 when $\lambda_0 = 1$. Note that the case 0 represents a particular situation. Indeed, the decoded signal is composed of one rising edge, and one falling edge

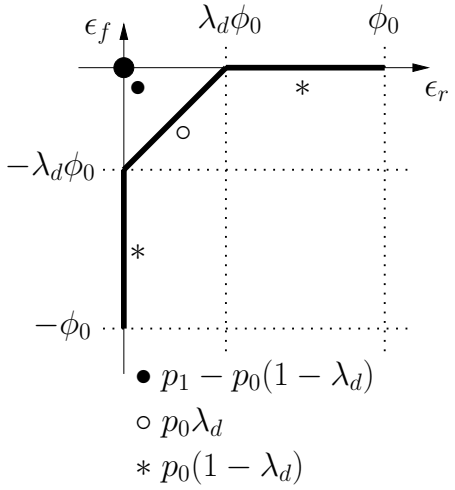

Figure 4.5: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case D.

only. It means that there is not enough information to compute the beacon ID, even if an angle could be computed for this signal. The hardware then discards this information, so that the high level positioning algorithm does not even need to deal with this situation. This case appears when the received power is sufficiently low, or equivalently when the distance is high enough. Therefore, this case could seem uninteresting, but we study this case in order to be comprehensive in our study of the variance with respect to the angular window. Moreover, we could consider sending this angle to the high level positioning algorithm if it can deal with unidentified beacons.

4.2.2.3 The case D

In that case, it is possible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. When this occurs, it can be shown that their difference is constant (see Figure A.3, lines 4 and 8, or red parts, in the appendix A.3.3). This situation occurs for values of the angular window such that

$$\phi_W^{mod} \in [11\phi_0,\, 12\phi_0). \tag{4.31}$$

However, this constant difference depends on the particular value of the angular window ($\epsilon_r - \epsilon_f = \lambda_d\phi_0$, where λ_d is defined hereafter). Therefore, the support of the joint *PDF* depends on the value of the angular window, and as a consequence, we introduce a parameter to express ϕ_W in that interval

$$\phi_W^{mod} = 11\phi_0 + \lambda_d\phi_0, \qquad \lambda_d \in [0,\, 1). \tag{4.32}$$

The support of the joint *PDF* is represented in Figure 4.5. It is composed of an horizontal line segment from $\lambda_d\phi_0$ to ϕ_0 ($\mathcal{E}_r \neq 0$ only), a vertical line segment from $-\phi_0$ to $-\lambda_d\phi_0$ ($\mathcal{E}_f \neq 0$ only), a slanted segment ($\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$) joining the previous ones, and a black dot at the origin (no errors). The black dot represents a two dimensional Dirac *PDF*, and the line segments are uniform *PDFs*, as explained hereafter. The horizontal and vertical line segments are denoted by an asterisk ($*$), the slanted line segment is denoted by a circle ($\circ$) and the origin by a black dot ($\bullet$).

Now, we have to associate probabilities and *PDFs* to these four parts. Again, we assume the time stationarity, and we assume that the pairs of errors are uniformly distributed along the line segments. Therefore, the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$\begin{aligned} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f) &= p^{\bullet}\delta(\epsilon_r)\,\delta(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_r)\; U_{(-\phi_0,\lambda_d\phi_0)}(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_f)\; U_{(\lambda_d\phi_0,\phi_0)}(\epsilon_r) \\ &+ p^{\circ}2\delta((\epsilon_r-\epsilon_f)-\lambda_d\phi_0)\; U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}(\epsilon_r+\epsilon_f), \end{aligned} \tag{4.33}$$

where:

- $p^{\bullet}$ is the probability to commit no error,
- p^{*} is the probability to have $\mathcal{E}_r \neq 0$ only. By symmetry, we show that the probability to have $\mathcal{E}_f \neq 0$ only is also given by p^{*}, explaining why the notation p^{*} is the same for both, and
- p° is the probability to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$.

The complete explanation about the computation of these probabilities is provided in Section A.3.3 of the appendix. Their values are given below

$$\begin{cases} p^{\bullet} = p_1 - p_0(1-\lambda_d), \\ p^{\circ} = p_0\lambda_d, \\ p^{*} = p_0(1-\lambda_d). \end{cases} \tag{4.34}$$

We can check that the probabilities for the case D sum up to 1

$$p^{\bullet} + p^{\circ} + 2p^{*} = p_1 - p_0(1-\lambda_d) + p_0\lambda_d + 2p_0(1-\lambda_d) = p_1 + p_0 = 1. \tag{4.35}$$

The joint expectation is (see appendix A.3.3)

$$E\{\mathcal{E}_r, \mathcal{E}_f\} = -p_0\lambda_d^3\frac{\phi_0^2}{6}. \tag{4.36}$$

Then, we can compute $C\{\mathcal{E}_r, \mathcal{E}_f\}$ for the case D

$$C\{\mathcal{E}_r, \mathcal{E}_f\} = E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\}E\{\mathcal{E}_f\} = -p_0\lambda_d^3\frac{\phi_0^2}{6} + p_0^2\frac{\phi_0^2}{4}, \tag{4.37}$$

and, finally, we can obtain the variance of Φ_b for the case D (see appendix A.3.3)

$$\begin{aligned} \sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r, \mathcal{E}_f\}}{2} \\ &= p_0\frac{\phi_0^2}{6}P_D(\lambda_d), \end{aligned} \tag{4.38}$$

where the first part is the same as for the case A, and the second part is a polynomial function of the parameter λ_d

$$P_D(\lambda_d) = 1 - \frac{\lambda_d^3}{2}. \tag{4.39}$$

Note that, again, $P_D(\lambda_d)$ is inferior to 1 when $\lambda_d \in [0, 1)$.

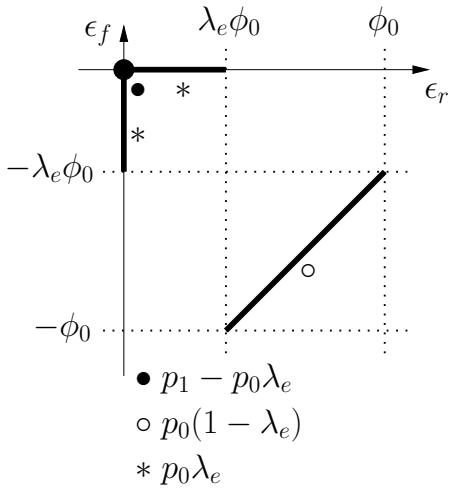

Figure 4.6: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case E.

4.2.2.4 The case E

In that case, it is also possible to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ simultaneously. When it happens, it can be shown that their difference is constant (see Figure A.4, lines 3 and 7, or red parts, in the appendix A.3.4). It occurs for values of the angular window such that

$$\phi_W^{mod} \in [0,\, \phi_0), \qquad \phi_W \geq 12\phi_0. \tag{4.40}$$

Note that, there is a condition on ϕ_W, in order to differentiate this case from the case 0 (compare equations (4.15) and (4.40)). However, this constant difference depends on the particular value of the angular window ($\epsilon_r - \epsilon_f = (1+\lambda_e)\,\phi_0$, where λ_e is defined hereafter). Therefore, the support of the joint *PDF* depends on the value of the angular window, and as a consequence, we introduce a parameter to express ϕ_W in that interval

$$\phi_W^{mod} = 12\phi_0 + \lambda_e\phi_0, \qquad \lambda_e \in [0,\, 1). \tag{4.41}$$

The support of the joint *PDF* is represented in Figure 4.6. It is composed of an horizontal line segment from 0 to $\lambda_e\phi_0$ ($\mathcal{E}_r \neq 0$ only), a vertical line segment from $-\lambda_e\phi_0$ to 0 ($\mathcal{E}_f \neq 0$ only), a slanted segment ($\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$), and a black dot at the origin (no errors). The black dot represents a two dimensional Dirac *PDF*, and the line segments are uniform *PDFs*, as explained hereafter. The horizontal and vertical line segments are denoted by an asterisk ($*$), the slanted line segment is denoted by a circle ($\circ$) and the origin by a black dot ($\bullet$).

Now, we have to associate probabilities and *PDFs* to these four parts. Again, we assume the time stationarity, and we assume that the pairs of errors are uniformly distributed along the line segments. Therefore, the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$\begin{aligned} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f) &= p^{\bullet}\delta(\epsilon_r)\,\delta(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_r)\; U_{(-\lambda_e\phi_0,0)}(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_f)\; U_{(0,\lambda_e\phi_0)}(\epsilon_r) \\ &+ p^{\circ}2\delta\left((\epsilon_r - \epsilon_f) - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(\epsilon_r + \epsilon_f)\,, \end{aligned} \tag{4.42}$$

where:

- $p^\bullet$ is the probability to commit no error,
- p^* is the probability to have $\mathcal{E}_r \neq 0$ only. By symmetry, we show that the probability to have $\mathcal{E}_f \neq 0$ only is also given by p^*, explaining why the notation p^* is the same for both, and
- p° is the probability to have $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$.

The complete explanation about the computation of these probabilities is given in the appendix (Section A.3.4). Their values are

$$\begin{cases} p^\bullet = p_1 - p_0\lambda_e, \\ p^\circ = p_0\left(1-\lambda_e\right), \\ p^* = p_0\lambda_e. \end{cases} \tag{4.43}$$

We can check that the probabilities for the case E sum up to 1

$$p^\bullet + p^\circ + 2p^* = p_1 - p_0\lambda_e + p_0\left(1-\lambda_e\right) + 2p_0\lambda_e = p_1 + p_0 = 1. \tag{4.44}$$

The computation of the joint expectation gives (see appendix A.3.4)

$$E\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\} = -p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right). \tag{4.45}$$

Then, we can compute $C\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\}$ for the case E

$$C\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\} = E\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\} - E\left\{\mathcal{E}_r\right\}E\left\{\mathcal{E}_f\right\} = -p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right) + p_0^2\frac{\phi_0^2}{4}, \tag{4.46}$$

and, finally, we can obtain the variance of Φ_b for the case E (see appendix A.3.4)

$$\begin{aligned} \sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\}}{2} \\ &= p_0\frac{\phi_0^2}{6}P_E\left(\lambda_e\right), \end{aligned} \tag{4.47}$$

where the first part is the same as for the case A, and the second part is a polynomial function of the parameter λ_e

$$P_E\left(\lambda_e\right) = \frac{1-3\lambda_e+3\lambda_e^2+\lambda_e^3}{2}. \tag{4.48}$$

Note that, again, $P_E\left(\lambda_e\right)$ is inferior to 1 when $\lambda_e \in [0,\, 1)$.

4.2.2.5 The case B

When analyzing the possible pairs of values $(\mathcal{E}_r,\, \mathcal{E}_f)$ (see Figure A.5 in the appendix A.3.5), it appears that the case B is a combination of the case A and the case D. This case occurs for values of the angular window such that

$$\phi_W^{mod} \in [i\phi_0,\, (i+1)\,\phi_0) \bigcup [(10-i)\,\phi_0,\, (11-i)\,\phi_0), \qquad i = 1,\, 2,\, 3,\, 4. \tag{4.49}$$

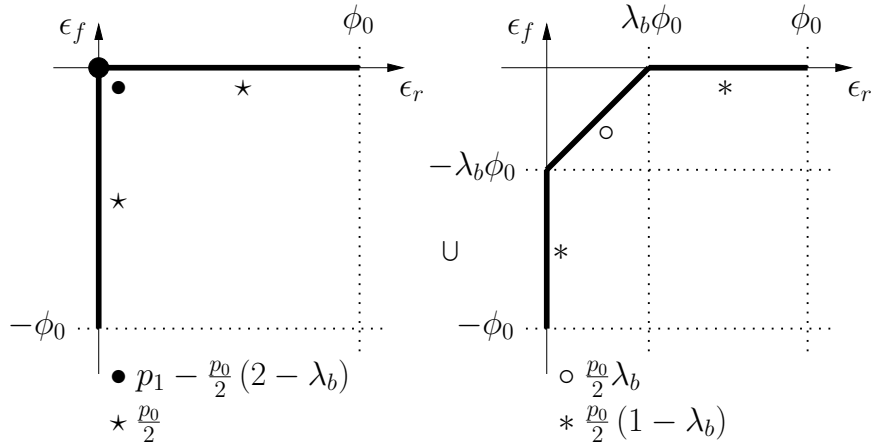

Figure 4.7: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case B. The complete support is obtained by the superimposition of both parts.

Note that this case does not occur for the code 5, as a consequence of its particular shape. Like for the case D, the support of the joint *PDF* depends on the value of the angular window, and as a consequence, we introduce a parameter λ_b to have an analytical expression of ϕ_W

$$\phi_W^{mod} = i\phi_0 + \lambda_b\phi_0, \qquad \lambda_b \in [0,\, 1),\ i = 1,\, 2,\, 3,\, 4, \tag{4.50}$$

in the first part of the interval, or

$$\phi_W^{mod} = (10-i)\,\phi_0 + \lambda_b\phi_0, \qquad \lambda_b \in [0,\, 1),\ i = 1,\, 2,\, 3,\, 4, \tag{4.51}$$

in the second part of the interval. The support of the joint *PDF* is represented in Figure 4.7. Indeed, the support of the joint *PDF* is obtained by the superimposition of both parts. However, we chose to represent this support with separate parts for two reasons: 1) we want that each line segment represents a uniform *PDF*, and 2) we can use all the previous results about the cases that we have already studied. The joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$\begin{aligned} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\epsilon_f) &= p^{\bullet}\delta(\epsilon_r)\,\delta(\epsilon_f) \\ &+ p^{\star}\delta(\epsilon_r)\,U_{(-\phi_0,0)}(\epsilon_f) \\ &+ p^{\star}\delta(\epsilon_f)\,U_{(0,\phi_0)}(\epsilon_r) \\ &+ p^{*}\delta(\epsilon_r)\,U_{(-\phi_0,\lambda_b\phi_0)}(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_f)\,U_{(\lambda_b\phi_0,\phi_0)}(\epsilon_r) \\ &+ p^{\circ}2\delta\left((\epsilon_r-\epsilon_f)-\lambda_b\phi_0\right)U_{(-\lambda_b\phi_0,\lambda_b\phi_0)}(\epsilon_r+\epsilon_f), \end{aligned} \tag{4.52}$$

where the different probabilities have the same meaning as previously defined. The complete explanation about the computation of these probabilities is given in the appendix (Section A.3.5). Their values are

$$\begin{cases} p^{\bullet} = p_1 - \frac{p_0}{2}(2-\lambda_b) \\ p^{\circ} = \frac{p_0}{2}\lambda_b \\ p^{\star} = \frac{p_0}{2} \\ p^{*} = \frac{p_0}{2}(1-\lambda_b). \end{cases} \tag{4.53}$$

We can check that the probabilities for the case B sum up to 1

$$p^{\bullet} + p^{\circ} + 2p^{\star} + 2p^{*} = p_1 - \frac{p_0}{2}(2 - \lambda_b) + \frac{p_0}{2}\lambda_b + 2\frac{p_0}{2} + 2\frac{p_0}{2}(1 - \lambda_b) = p_1 + p_0 = 1. \quad (4.54)$$

The computation of the joint expectation gives (see appendix A.3.5)

$$E\{\mathcal{E}_r, \mathcal{E}_f\} = -p_0\lambda_b^3\frac{\phi_0^2}{12}. \quad (4.55)$$

Then, we can compute $C\{\mathcal{E}_r, \mathcal{E}_f\}$ for the case B

$$C\{\mathcal{E}_r, \mathcal{E}_f\} = E\{\mathcal{E}_r, \mathcal{E}_f\} - E\{\mathcal{E}_r\}E\{\mathcal{E}_f\} = -p_0\lambda_b^3\frac{\phi_0^2}{12} + p_0^2\frac{\phi_0^2}{4}, \quad (4.56)$$

and, finally, we can obtain the variance of Φ_b for the case B (see appendix A.3.5)

$$\begin{aligned}\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r, \mathcal{E}_f\}}{2} \\ &= p_0\frac{\phi_0^2}{6}P_B(\lambda_b), \end{aligned} \quad (4.57)$$

where the first part is the same as for the case A, and the second part is a polynomial function of the parameter λ_b

$$P_B(\lambda_b) = 1 - \frac{\lambda_b^3}{4}. \quad (4.58)$$

Note that, again, $P_B(\lambda_b)$ is inferior to 1 when $\lambda_b \in [0, 1)$.

4.2.2.6 The case C

When analyzing the possible pairs of values for $(\mathcal{E}_r, \mathcal{E}_f)$ (see Figure A.6 in the appendix A.3.6), it appears that the case C is a combination of the case A and the case E. This happens for values of the angular window such that

$$\phi_W^{mod} \in [(i+1)\phi_0, (i+2)\phi_0) \bigcup [(11-i)\phi_0, (12-i)\phi_0), \qquad i = 1, 2, 3, 4. \quad (4.59)$$

Note that this case does not occur for the code 5, as a consequence of its particular shape. Like for the case E, the support of the joint *PDF* depends on the value of the angular window, and as a consequence, we introduce a parameter to express ϕ_W in terms of ϕ_0

$$\phi_W^{mod} = (i+1)\phi_0 + \lambda_c\phi_0, \qquad \lambda_c \in [0, 1),\ i = 1, 2, 3, 4, \quad (4.60)$$

in the first part of the interval, or

$$\phi_W^{mod} = (11-i)\phi_0 + \lambda_c\phi_0, \qquad \lambda_c \in [0, 1),\ i = 1, 2, 3, 4, \quad (4.61)$$

in the second part of the interval. The support of the joint *PDF* is represented in Figure 4.8. Indeed, the support of the joint *PDF* is obtained by the superimposition of both parts.

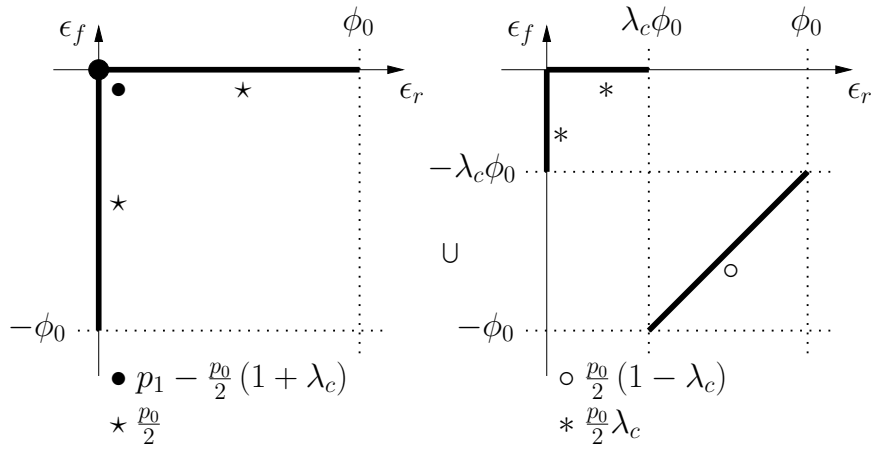

Figure 4.8: Support of the joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the case C. The complete support is obtained by the superimposition of both parts.

However, we chose to represent this support with separate parts for the same reasons as for the case B. The joint *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ is given by the following mixture of joint *PDFs*

$$\begin{aligned} f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\epsilon_f) &= p^{\bullet}\delta(\epsilon_r)\,\delta(\epsilon_f) \\ &+ p^{\star}\delta(\epsilon_r)\,U_{(-\phi_0,0)}(\epsilon_f) \\ &+ p^{\star}\delta(\epsilon_f)\,U_{(0,\phi_0)}(\epsilon_r) \\ &+ p^{*}\delta(\epsilon_r)\,U_{(-\lambda_c\phi_0,0)}(\epsilon_f) \\ &+ p^{*}\delta(\epsilon_f)\,U_{(0,\lambda_c\phi_0)}(\epsilon_r) \\ &+ p^{\circ}2\delta((\epsilon_r-\epsilon_f)-(1+\lambda_c)\,\phi_0)\,U_{(-(1-\lambda_c)\phi_0,(1-\lambda_c)\phi_0)}(\epsilon_r+\epsilon_f), \end{aligned} \tag{4.62}$$

where the different probabilities have the same meaning as previously defined. The complete explanation about the computation of these probabilities is given in the appendix (Section A.3.6). Their values are

$$\begin{cases} p^{\bullet} = p_1 - \frac{p_0}{2}(1+\lambda_c), \\ p^{\circ} = \frac{p_0}{2}(1-\lambda_c), \\ p^{\star} = \frac{p_0}{2}, \\ p^{*} = \frac{p_0}{2}\lambda_c. \end{cases} \tag{4.63}$$

We can check that the probabilities for the case C sum up to 1

$$p^{\bullet} + p^{\circ} + 2p^{\star} + 2p^{*} = p_1 - \frac{p_0}{2}(1+\lambda_c) + \frac{p_0}{2}(1-\lambda_c) + 2\frac{p_0}{2} + 2\frac{p_0}{2}\lambda_c = p_1 + p_0 = 1. \tag{4.64}$$

The computation of the joint expectation gives (see appendix A.3.6)

$$E\{\mathcal{E}_r,\,\mathcal{E}_f\} = -p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)(1+4\lambda_c+\lambda_c^2)}{2}. \tag{4.65}$$

Then, we can compute $C\{\mathcal{E}_r,\,\mathcal{E}_f\}$ for the case C

$$C\{\mathcal{E}_r,\,\mathcal{E}_f\} = E\{\mathcal{E}_r,\,\mathcal{E}_f\} - E\{\mathcal{E}_r\}\,E\{\mathcal{E}_f\} = -p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)(1+4\lambda_c+\lambda_c^2)}{2} + p_0^2\frac{\phi_0^2}{4}, \tag{4.66}$$

and, finally, we can obtain the variance of Φ_b for the case C (see appendix A.3.6)

$$\begin{aligned}\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\left\{\mathcal{E}_r, \mathcal{E}_f\right\}}{2} \\ &= p_0 \frac{\phi_0^2}{6} P_C\left(\lambda_c\right), \end{aligned} \tag{4.67}$$

where the first part is the same as for the case A, and the second part is a polynomial function of the parameter λ_c

$$P_C\left(\lambda_c\right) = \frac{3 - 3\lambda_c + 3\lambda_c^2 + \lambda_c^3}{4}. \tag{4.68}$$

Note that, again, $P_C\left(\lambda_c\right)$ is inferior to 1 when $\lambda_c \in [0,\, 1)$.

4.2.3 Summary of the variance value for all the cases

In the previous section, we have established the value of $\sigma^2_{\Phi_b}$ in function of ϕ_W, for all the cases. It appears that $\sigma^2_{\Phi_b}$ is always equal to a fraction of $p_0 \frac{\phi_0^2}{6}$ (the upper bound). This fraction is a polynomial function of a parameter λ, which represents a fraction of $\phi_0 :\ \lambda\phi_0$, $\lambda \in [0,\, 1)$. If we define an appropriate λ for each case (see the previous section), then we can summarize the polynomial functions as follows

$$P_0\left(\lambda_0\right) = \frac{\lambda_0^3}{p_1 + p_0\lambda_0}, \tag{4.69}$$

$$P_A\left(\lambda_a\right) = 1, \tag{4.70}$$

$$P_B\left(\lambda_b\right) = 1 - \frac{\lambda_b^3}{4}, \tag{4.71}$$

$$P_C\left(\lambda_c\right) = \frac{3 - 3\lambda_c + 3\lambda_c^2 + \lambda_c^3}{4}, \tag{4.72}$$

$$P_D\left(\lambda_d\right) = 1 - \frac{\lambda_d^3}{2}, \tag{4.73}$$

$$P_E\left(\lambda_e\right) = \frac{1 - 3\lambda_e + 3\lambda_e^2 + \lambda_e^3}{2}, \tag{4.74}$$

where each parameter satisfies $\lambda_i \in [0,\, 1)$. For example, for the case D, Φ_b is expressed as the product of the theoretical bound $\left(p_0 \frac{\phi_0^2}{6}\right)$, and $\left(1 - \frac{\lambda_d^3}{2}\right)$, which is smaller than 1, for the admissible values of its parameter.

Then, we can express the evolution of the variance with respect to ϕ_W, and for each code. To do this, we have to find which case is appropriate for a particular value of ϕ_W, when ϕ_W increases from 0 to $13\phi_0$ (in order to observe the case 0, followed by a full period of the variance). This can be done by observing equations 4.7, 4.15, 4.31 4.40 4.49, and 4.59. When the angular window increases, the different cases appear in an order related to their logical names (the situation is slightly different for the code 5, and is treated later). Therefore, the evolution of the variance with respect to ϕ_W is defined as a partially periodic and piecewise

function

$$C_i,\ i=1,\ 2,\ 3,\ 4: \qquad \sigma^2_{\Phi_b}(\phi_W)=\begin{cases} p_0\frac{\phi_0^2}{6}P_0(\lambda_0) & 0\le\phi_W<\phi_0 \\ p_0\frac{\phi_0^2}{6} & \phi_0\le\phi_W^{mod}<i\phi_0 \\ p_0\frac{\phi_0^2}{6}P_B(\lambda_b) & i\phi_0\le\phi_W^{mod}<(i+1)\phi_0 \\ p_0\frac{\phi_0^2}{6}P_C(\lambda_c) & (i+1)\phi_0\le\phi_W^{mod}<(i+2)\phi_0 \\ p_0\frac{\phi_0^2}{6} & (i+2)\phi_0\le\phi_W^{mod}<(10-i)\phi_0 \\ p_0\frac{\phi_0^2}{6}P_B(\lambda_b) & (10-i)\phi_0\le\phi_W^{mod}<(11-i)\phi_0 \\ p_0\frac{\phi_0^2}{6}P_C(\lambda_c) & (11-i)\phi_0\le\phi_W^{mod}<(12-i)\phi_0 \\ p_0\frac{\phi_0^2}{6} & (12-i)\phi_0\le\phi_W^{mod}<11\phi_0 \\ p_0\frac{\phi_0^2}{6}P_D(\lambda_d) & 11\phi_0\le\phi_W^{mod}<12\phi_0 \\ p_0\frac{\phi_0^2}{6}P_E(\lambda_e) & 0\le\phi_W^{mod}<\phi_0,\quad \phi_W\ge 12\phi_0, \end{cases} \tag{4.75}$$

where each parameter satisfies $\lambda_i\in[0,\,1)$, and where a complete period is defined for $\phi_W\in[\phi_0,\,13\phi_0)$. Also, note that the values of these functions at the transition points (from one case to the next) are consistent

$$\begin{aligned} P_0(1)=P_A(0) &= 1, \\ P_A(1)=P_B(0) &= 1, \\ P_B(1)=P_C(0) &= 3/4, \\ P_C(1)=P_A(0) &= 1, \\ P_A(1)=P_D(0) &= 1, \\ P_D(1)=P_E(0) &= 1/2, \\ P_E(1)=P_A(0) &= 1, \end{aligned}$$

where we have introduced $P_A(\lambda_a)$, which is the constant function 1. Therefore, the evolution of the variance with respect to ϕ_W is continuous. Note that some intervals associated to the case A may be reduced to zero (for $i=1$, and $i=4$) in the variance expression, without harming its general form. Because of the particular shape of the code 5, the period of its variance is reduced to $6\phi_0$ instead of $12\phi_0$. Also, it does not contain the cases B and C. Therefore, the evolution of the variance with respect to ϕ_W for the code 5 is expressed in a separate, but similar way

$$C_5: \qquad \sigma^2_{\Phi_b}(\phi_W)=\begin{cases} p_0\frac{\phi_0^2}{6}P_0(\lambda_0) & 0\le\phi_W<\phi_0 \\ p_0\frac{\phi_0^2}{6} & \phi_0\le\phi_W \bmod(6\phi_0)<5\phi_0 \\ p_0\frac{\phi_0^2}{6}P_D(\lambda_d) & 5\phi_0\le\phi_W \bmod(6\phi_0)<6\phi_0 \\ p_0\frac{\phi_0^2}{6}P_E(\lambda_e) & 0\le\phi_W \bmod(6\phi_0)<\phi_0,\quad \phi_W\ge 6\phi_0. \end{cases} \tag{4.76}$$

Note that the evolution of the variance with respect to ϕ_W contains local minima, encountered in $P_C(\lambda_c)$ and $P_E(\lambda_e)$, for values of their arguments equal to (see appendix A.4 for details)

$$\lambda_c=\lambda_e=\sqrt{2}-1. \tag{4.77}$$

The corresponding values of these polynomials are

$$P_C\left(\sqrt{2}-1\right) = 2-\sqrt{2} \simeq 0.5858, \tag{4.78}$$

$$P_E\left(\sqrt{2}-1\right) = 3-2\sqrt{2} \simeq 0.1716. \tag{4.79}$$

Finally, the local minima are encountered when the angular windows are equal to

$$C_5: \qquad \phi_W = 6\phi_0 + \left(\sqrt{2}-1\right)\phi_0 + k\,6\phi_0, \qquad k \in \mathbb{N}, \tag{4.80}$$

for the code 5, and when the angular windows are equal to

$$C_i,\ i=1,\,2,\,3,\,4: \quad \begin{cases} \phi_W = (i+1)\,\phi_0 + \left(\sqrt{2}-1\right)\phi_0 + k\,12\phi_0, & k \in \mathbb{N}, \\ \vee\, \phi_W = (11-i)\,\phi_0 + \left(\sqrt{2}-1\right)\phi_0 + k\,12\phi_0, & k \in \mathbb{N}, \\ \vee\, \phi_W = 12\phi_0 + \left(\sqrt{2}-1\right)\phi_0 + k\,12\phi_0, & k \in \mathbb{N}, \end{cases} \tag{4.81}$$

for the other codes.

4.3 Simulations

In order to validate our theory about the code statistics, we developed a simulator. The four parameters considered by the simulator are:

1. the angular window ϕ_W,
2. the code (symbols and durations),
3. the turret period, and
4. the number of turret turns.

The simulations have been performed for the five codes used by BeAMS, and for values of ϕ_W ranging from 0 to $13\phi_0$, in order to observe the case 0 ($\phi_W \in [0,\, \phi_0)$), followed by a full period of the variance evolution ($\phi_W \in [\phi_0,\, 13\phi_0)$). The values used for the turret period and the number of turret turns are discussed in the next section.

Simulation results are presented in Figure 4.9 to Figure 4.13. From these figures, one can see that the simulations perfectly match the theory. So, in order to distinguish both curves, the theoretical variance is represented by a continuous line, and the simulated variance is represented by small unconnected circles. For convenience, we have plotted the normalized variance ($\sigma^2_{\Phi_b}/p_0\,\frac{\phi_0^2}{6}$) in function of the normalized angular window ($\phi w/\phi_0$). We have also reported the encountered case on the bottom of the graphs. Obviously, for all the codes, the case 0 only appears on the left, when $\phi_W < \phi_0$. For other angular window values, the relevant case is one of the possible A, B, C, D, or E cases.

In complement to the variance graphs, we have reported the pairs of observations of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the different cases of the code 1[1], in order to visualize the simulated supports of the joint *PDFs* of $\mathcal{E}_r$ and $\mathcal{E}_f$. These supports obtained by simulation are presented in Table 4.3

[1]Note that we could have used another code to observe all the different cases, except the code 5, since it does not contain the cases B and C.

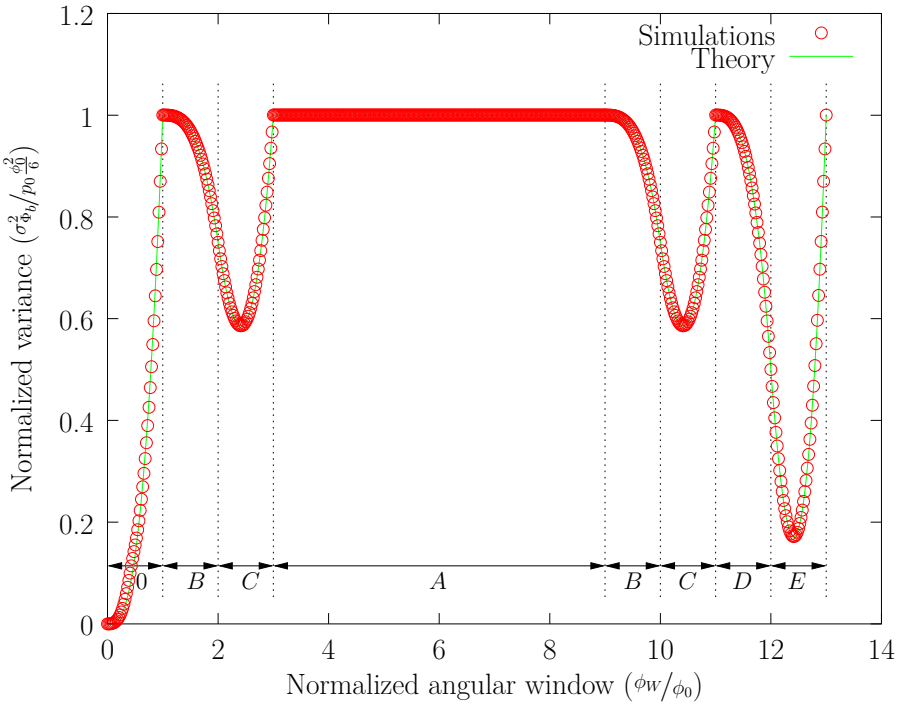

Figure 4.9: Variance of Φ_b in function of ϕ_W for the code 1: simulations versus theory.

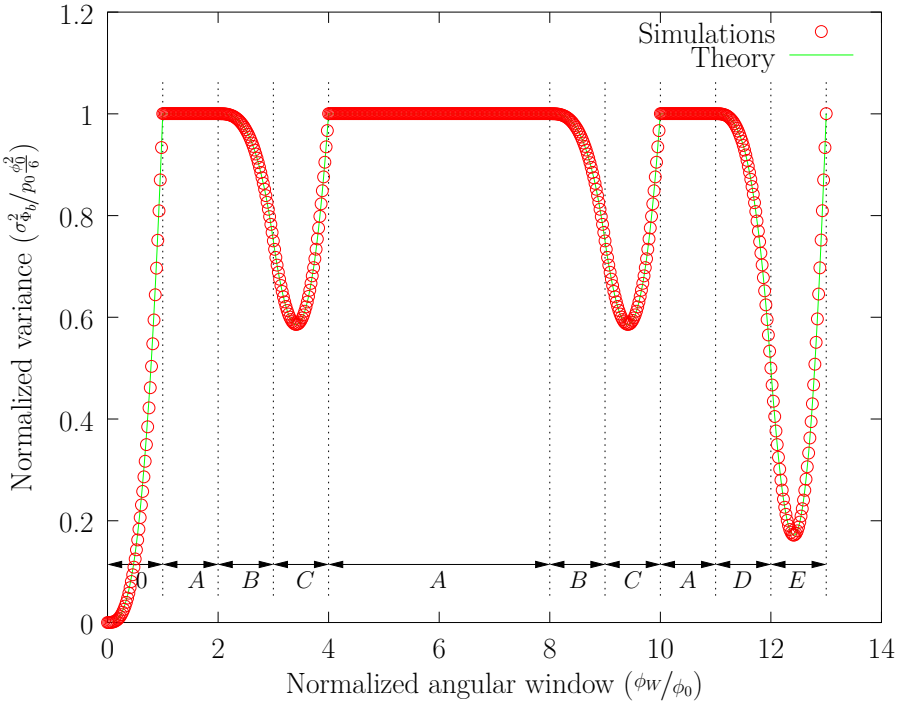

Figure 4.10: Variance of Φ_b in function of ϕ_W for the code 2: simulations versus theory.

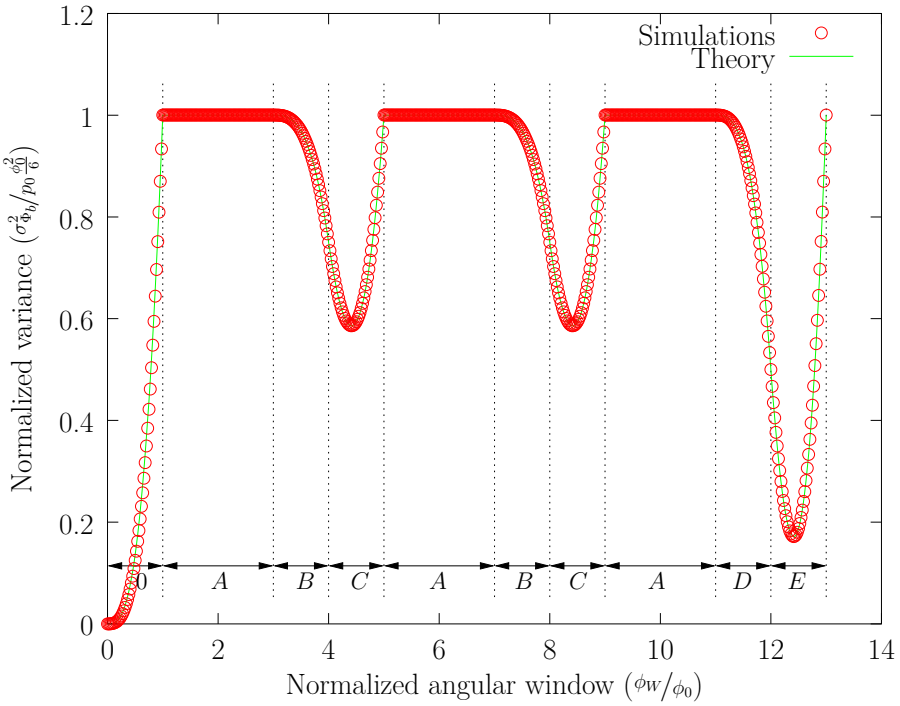

Figure 4.11: Variance of Φ_b in function of ϕ_W for the code 3: simulations versus theory.

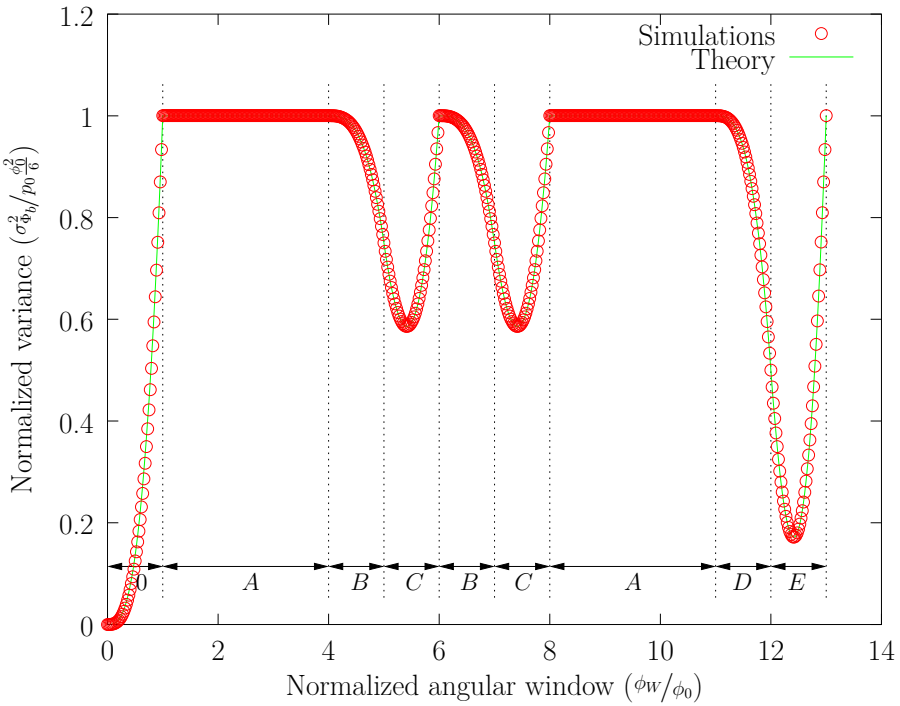

Figure 4.12: Variance of Φ_b in function of ϕ_W for the code 4: simulations versus theory.

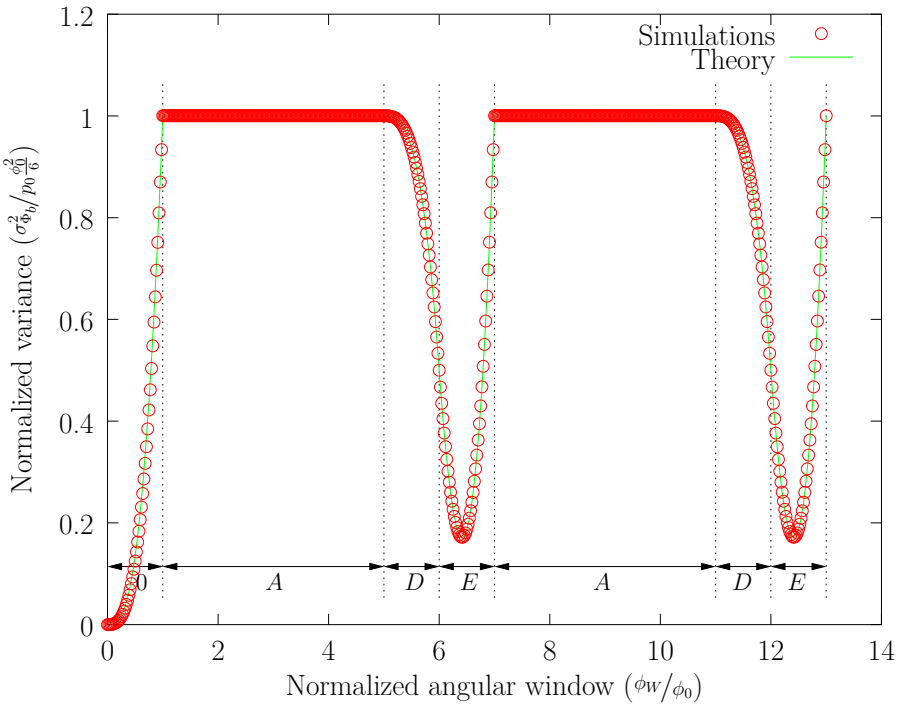

Figure 4.13: Variance of Φ_b in function of ϕ_W for the code 5: simulations versus theory.

(for all the cases, except for the case A, we have chosen values for the angular window, such that $\lambda_0 = \lambda_b = \lambda_c = \lambda_d = \lambda_e = 0.5$). Because the pairs of $\mathcal{E}_r$ and $\mathcal{E}_f$ values are superimposed on these graphs, it is impossible to distinguish the 2D Dirac *PDF* (except for the case D), and the separate parts of the cases B and C, but it appears that simulated supports also match the theory.

4.4 The time stationarity hypothesis

To establish all the theoretical results presented in this chapter, we have assumed the time stationarity. In other words, these theoretical results are consistent if we can observe all the possible values for the pairs $(\mathcal{E}_r, \mathcal{E}_f)$, for all angular windows and codes. This implies that we can observe all the possible shifted versions of a code with respect to the angular window (like we did in the appendix to compute all the probabilities associated to the different cases).

So, in order to produce the previous figures showing the variance evolution with respect to the angular window, and to confirm the adequacy between theory and simulations, we had to "simulate" the time stationarity as explained hereafter. As mentioned, the simulator requires four parameters: the angular window, the code (symbols and durations), the turret period, and the number of turret turns. The two first parameters are variables of the study. However, the values of the turret period and the number of turret turns can be adjusted[2] to "simulate" the time stationarity. To do so, one has to choose a value for the turret period such that the code duration and the turret period are relatively prime (or coprime[3]). As a result, the

[2]It means that we can set these parameters to values that are different from the real system.
[3]Two integers are said to be relatively prime (or coprime) if their greatest common divisor is equal to 1.

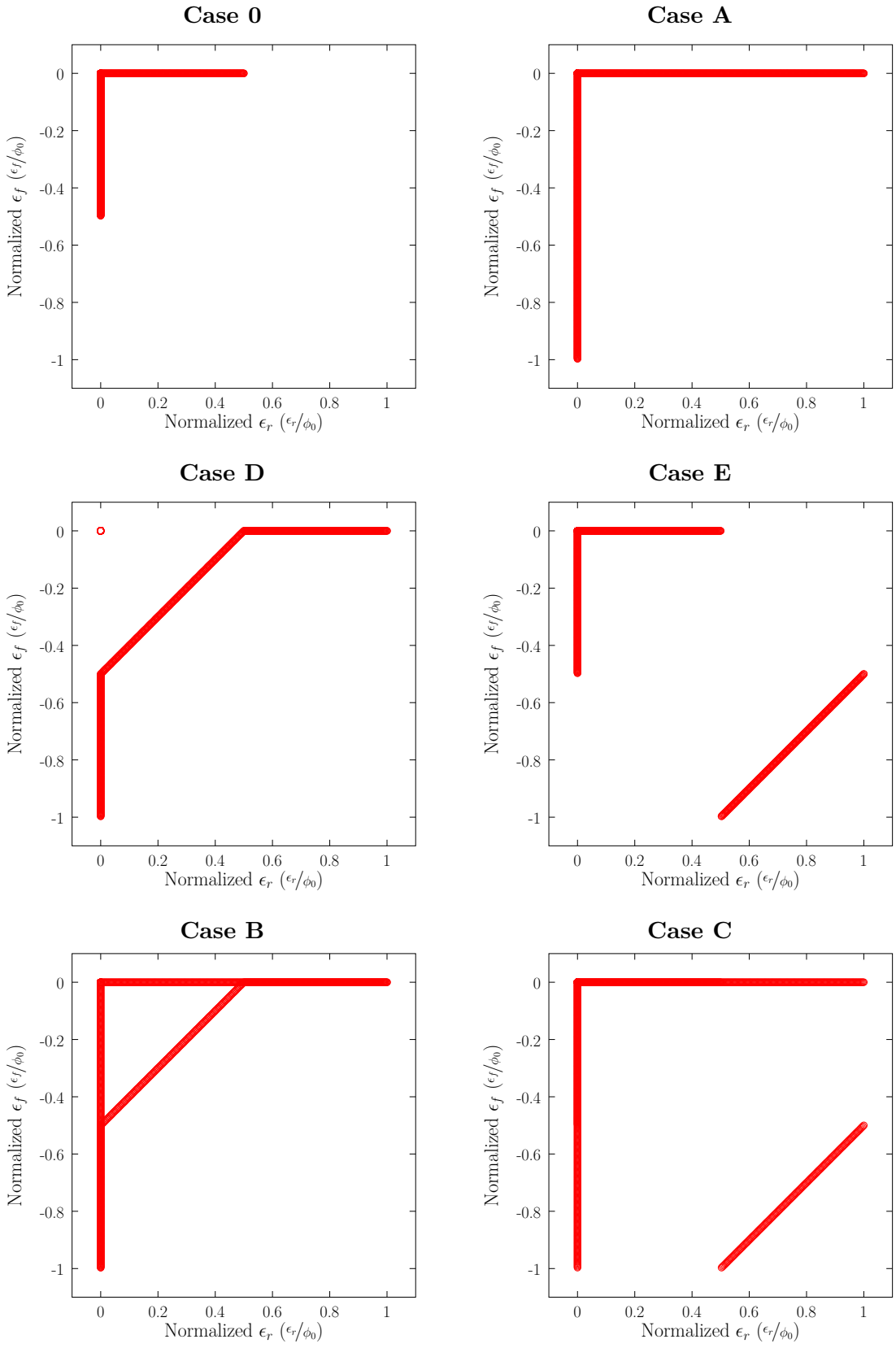

Table 4.3: Simulations for the supports of the joint *PDFs* of $\mathcal{E}_r$ and $\mathcal{E}_f$ for the 6 different Cases ($\lambda_0 = \lambda_b = \lambda_c = \lambda_d = \lambda_e = 0.5$).

code will shift from one unit with respect to the angular window, for each new observation, or turret turn. Finally, if the number of turret turns is equal to the code duration, we will observe all the possible shifted versions of the code, and the time stationarity hypothesis will be met.

In the simulator, as well as in the real system, the bit durations and the turret period are represented by integers, whose one unit is equal to $100\,ns$. Therefore, the bit duration is represented by 308 ($30.8\,\mu s$), and the code duration is equal to $12 \times 308 = 3696$. The turret period is equal to 998400 ($0.09984\,s$). With these values, the code duration and the turret period are not relatively prime, and this would induce some artifacts on the data. So, we changed the value of the turret period to 998401[4], which is relatively prime with 3696, in order to meet the time stationarity hypothesis, and to generate the previous graphics.

However, the turret period of BeAMS does not correspond to the one used to generate the previous graphics. The real turret period and the code duration are not relatively prime, and therefore, the time stationarity hypothesis is not met. It means that, in practice, we do not observe all the possible shifted versions of a code with respect to the angular window. In other terms, the simulator, and the real system, act as a "bad" pseudo random number generator. As a consequence, the pairs of observed values $(\mathcal{E}_r, \mathcal{E}_f)$ are not well balanced, and finally the computed variance of Φ_b can oscillate (slightly) around its theoretical value. To enlighten this phenomenon, we have performed additional simulations, with four different turret periods (998400, 999000, 999200, and 999300 respectively), all not relatively prime with the code duration. The results are presented in Figure 4.14 to Figure 4.17. From these figures, one can observe the oscillations of the variance of Φ_b. As a consequence, the variance may exceed the theoretical bound established in this document. Unfortunately, it is difficult to formalize how the variance can exceed the theoretical bound, since it depends on the code symbols, the code duration, and the turret period. However the simulator can always help in finding all these characteristics. In the case of BeAMS, this variance can exceed the theoretical bound by up to 14 % for the code 2 and 4. Finally, note that the number of turret turns has been chosen such that we cover a complete period of observations of the code through the angular window

$$number\,of\,turret\,turns = \frac{code\,duration}{\gcd(code\,duration,\ turret\,period)}, \tag{4.82}$$

where $\gcd(a, b)$ is the greatest common divisor of a and b. Also, one can observe from these figures, that the oscillations in the simulated variances decrease if the number of turret turns required to cover a complete period of observations increases. The theoretical time stationarity condition is reached when $\gcd(code\,duration, turret\,period)$ is equal to 1, that is, when the code duration and the turret period are relatively prime. To the contrary, it is mandatory to avoid that the turret period be a multiple of the code duration. If it happens, it means that there is one unique observation of the code through the angular window (no code shifting). So, the variance would be null, but the estimator Φ_b would be no longer unbiased.

[4]Note that we chose the closest relatively prime integer, but we could have chosen any other relatively prime integer.

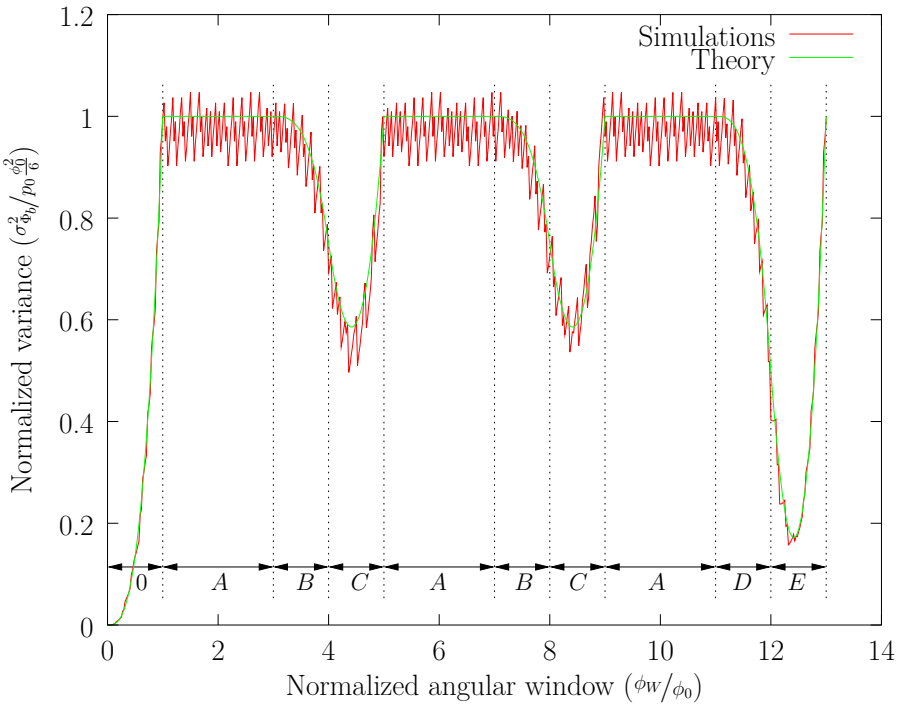

Figure 4.14: Variance of Φ_b in function of ϕ_W for the code 3: simulations versus theory (turret period = 998400, number of turret turns = 77).

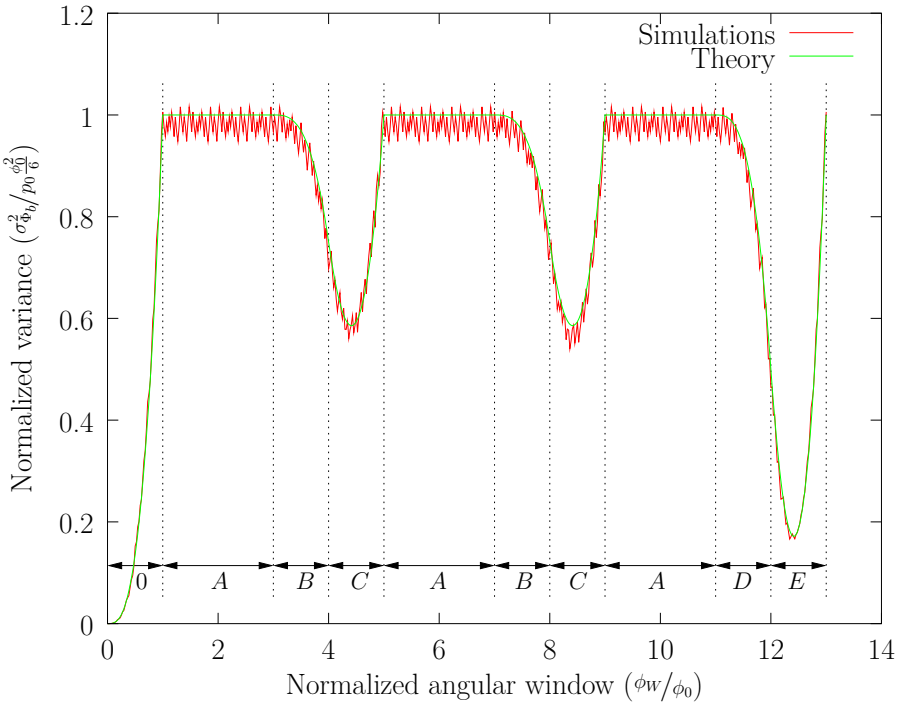

Figure 4.15: Variance of Φ_b in function of ϕ_W for the code 3: simulations versus theory (turret period = 999000, number of turret turns = 154).

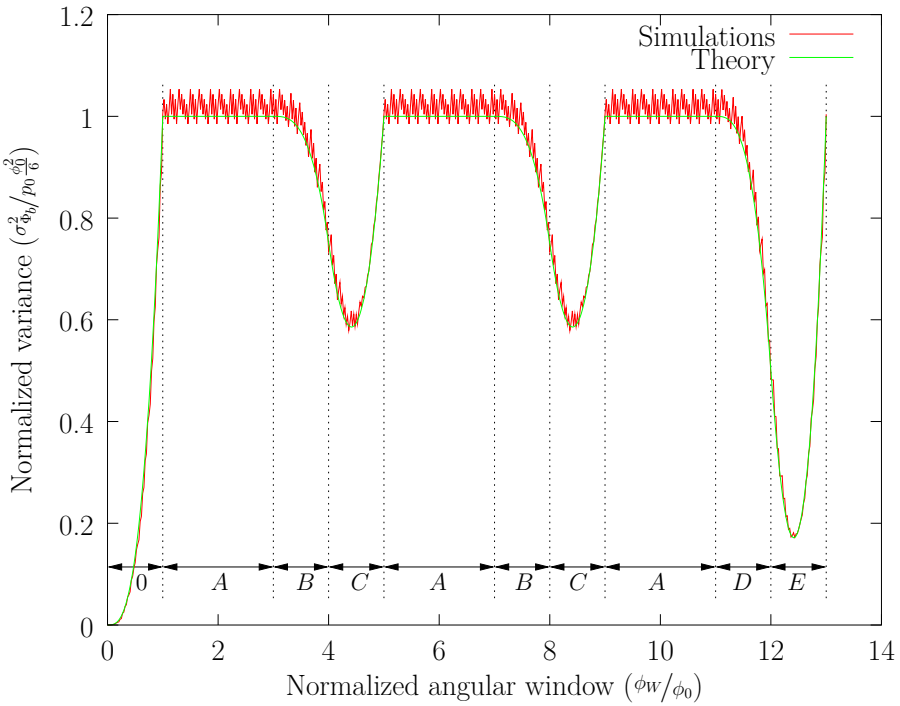

Figure 4.16: Variance of Φ_b in function of ϕ_W for the code 3: simulations versus theory (turret period = 999200, number of turret turns = 231).

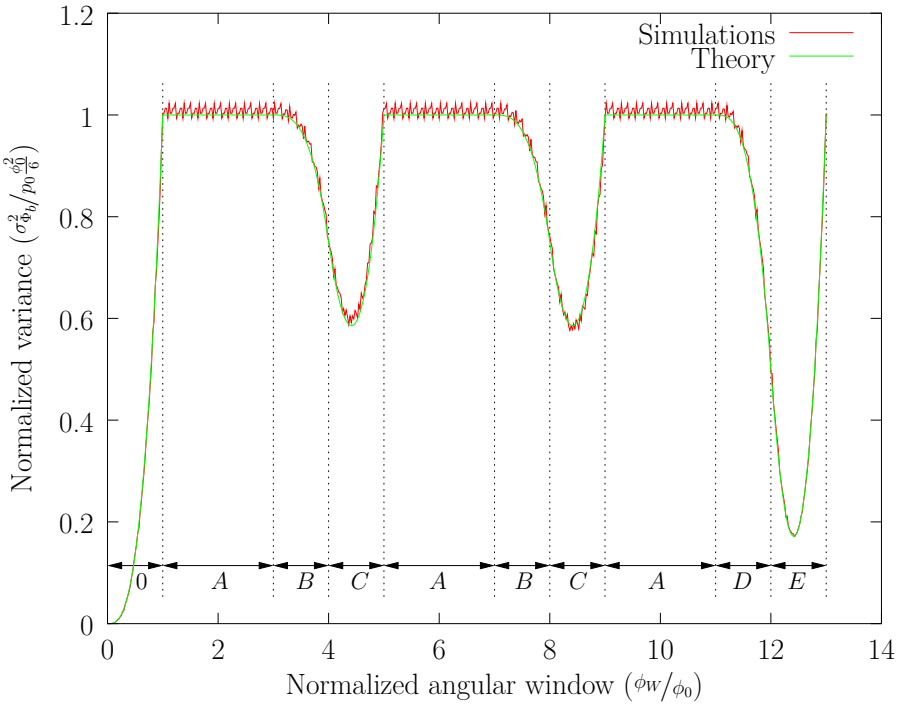

Figure 4.17: Variance of Φ_b in function of ϕ_W for the code 3: simulations versus theory (turret period = 999300, number of turret turns = 308).

4.5 Conclusions

This chapter complements the previous chapter by going into further details related to the code statistics of modulated signals in general, with an emphasis on BeAMS. BeAMS introduces a new mechanism to measure angles: it detects a beacon when it enters and leaves an angular window. One of the major challenges of such a system is to be able to model its behavior from a statistical point of view in order to both have an adequate measure of its efficiency and appropriate values to feed a tracking system (a Kalman filter for instance).

In the previous chapter, we have established the upper bound on the variance of Φ_b. In this chapter, we show how the variance evolves exactly as a function of the angular window (while remaining below the upper bound). The fact that the variance is a function of the angular window is essential for any system that measures angles. Often, authors consider that the variance evolves with the distance. However, our analysis is preferable, because the angular window is measurable by the system even if the robot position is unknown. Our analysis has a direct usage for any practical situation.

In our current implementation of BeAMS for the Eurobot contest (which relies on an Extended Kalman filter), we use the best upper bound of the variance of Φ_b all over the plane (regardless of the robot position). But this practice is far from being optimal. Indeed, we show that the variance evolves with the angular window (or indirectly with the position of the robot in the plane), and therefore the upper bound overestimates the real variance unnecessarily. In order to feed a tracking system, it would be better to have the exact variance for each position in the plane, and for each beacon/code.

Then, we present simulated results, in order to validate our theory about the code statistics. It appears that the simulated results perfectly match the theory. In this chapter, we also discuss an important hypothesis that we use all over our developments, that is, *the time stationarity hypothesis*. We show that this hypothesis is necessary to match the theory perfectly. However, if this hypothesis is not fully met in a practical situation, we show that the variance may exceed the theoretical bound (14 % in the case of BeAMS).

Because the developments given in this document are complete, it is possible to really understand the behavior of the system and to improve the design of BeAMS. It is important to mention that, although this study is carried on to understand and improve BeAMS, the theoretical framework is larger than that of BeAMS. It is applicable to any estimator that is built like BeAMS. In particular, it is applicable to any measurement system that estimates a value by taking the mean of a previous event and a later event, based on the reception of an On-Off Keying modulated signal. For example, the study of the different “Cases” could be generalized and extended to other codes.

Chapter 5

Performance analysis

5.1 Introduction

The goal of this chapter is to provide an error measure for BeAMS. In particular, we want to provide values for the precision (variance) and for the accuracy (bias) of the measured angles. In practice however, angle measurement systems are developed almost exclusively for positioning, as the process of triangulation requires angle measurements. This explains why most authors only evaluate positioning algorithms or present complete systems (hardware and software), and express quality results in meters. As a consequence, it is rare that authors evaluate the performance of the underlying angle measurement system, and there is a lot of confusion about the evaluation criteria (accuracy, precision, resolution, etc). This is unfortunate because the knowledge of these characteristics is useful for the data fusion algorithms to properly take measures into account, with respect to other measurements. They are also useful to compare systems.

It is a well known fact that a positioning process based on angles, regardless of its implementation, depends on the relative configuration of beacons and the robot [21, 28, 32]. So, we believe that an angle measurement system should not be evaluated through a positioning algorithm, unless a common procedure is described and used by everyone (to our knowledge, this work has never been carried out). Moreover, this evaluation procedure is difficult to implement in practice, and due to the setup it adds errors, especially errors on measurements and on the real location of beacons [53]. For that reason, we want to evaluate BeAMS directly, and not through a positioning algorithm.

In Chapter 3, we have provided the upper bound for the additional variance on Φ_b due to the codes, and showed that the estimator Φ_b is unbiased. In the previous chapter, we have computed the exact evolution of the variance of Φ_b, in function of the angular window. But, in a practical situation, we have to take into account the natural (noise) variance of the system by taking real measurements. This noise is inherent to the hardware, even for a non modulated carrier wave (with no OFF periods). This noise originates from the quartz jitter, rotation jitter, etc, and, to a larger extent, from the receiver jitter at the $0 \to 1$ and $1 \to 0$ transitions. From a theoretical point of view, it is acceptable to consider that both noises are independent and, therefore, that the total noise is the sum of the natural noise and $\sigma^2_{\Phi_b}$, the power of additional noise induced by the OOK modulation.

The purpose of this chapter is fourfold: (1) analyze the impact of the code (via the p_0 and ϕ_0 parameters) on the variance of Φ_b, (2) validate the upper bound and the evolution of

the variance of Φ_b, (3) verify if the artificial noise is independent of the natural noise, and (4) provide values for the precision and accuracy. In order to complete these analyses, simulations and measurements are performed with one beacon for several codes and angular windows.

5.2 Adding codes for tests only

We have shown that the upper bound of $\sigma^2_{\Phi_b}$ depends on p_0 and ϕ_0. By design, the codes all have the same p_0 and ϕ_0. This is an important advantage because this implies that the additional variance is not related to any particular code. As a consequence, we have to create other codes to observe the influence of p_0 and ϕ_0. Also, in order to measure the natural variance, we have to use a special code with no OFF period. So, the codes used for testing purposes are the constant code C_k (no OFF period), the real C_5 code (111110111110), and two variations of code C_5 with increasing OFF durations (11111001111100, 1111100011111000). These four codes have a zero symbol probability p_0 respectively equal to 0, $1/6$, $2/7$, and $3/8$, and an OFF angle ϕ_0 respectively equal to 0, 0.111, 0.222, and $0.333\,deg$. These variations have been chosen to emphasize the noise due to the OOK modulation, with increasing p_0 and ϕ_0. In the following section, these four codes are referred to, respectively, as C_k, C_5, C_{5b}, and C_{5c}. Note that C_k, C_{5b}, and C_{5c} are used for experiments only, but we do not use them in practice.

5.3 Modifying the angular window

In the last chapter, we have shown that the variance of Φ_b depends on the angular window. So, we also need to modify the angular window in the experiments. In practice, the value of the angular window $\phi_W = \phi_F - \phi_R$ depends on the received power and the threshold (see Figure 2.8 on page 16). So, for a constant threshold, the angular window decreases if the power curve goes down (less received power), and increases if the power curve goes up (more received power). Then, for a given receiver and optical components, the angular window only depends on the received power. There are two practical ways to modify the received power (or angular window): (1) change the distance between the beacon and the receiver, or (2) change the power emitted by the beacon. But, in any case, the receiver only has access to the angular window via the demodulated signal and it is thus not capable of detecting whether the distance or the emitted power have been modified.

In a practical situation, the emitted power of the beacons is expected to be constant, while the distance can change. So we could measure the variance for all possible distances in the working range. However the experiment would be extremely tedious and time-consuming since the number of distances should be huge to appreciate the variations in the measurements. So, we choose to modify the emitted power, for a fixed working distance of $1\,m$. For each code, a hundred different emitted power values were taken in the $4\,mW$ to $150\,mW$ power range to obtain an approximately linear increase of the angular window. These power values and angular windows are values that correspond to distances ranging from $1\,m$ to $6\,m$. Finally, 1000 angle measurements are taken for each code and power value to compute the mean and variance of Φ_b (to be precise, only Φ_r and Φ_f are recorded and Φ_b is computed after equation (3.1)). As explained earlier, the receiver is not capable of detecting whether the distance or the emitted power have been modified and, as a consequence, all the following graphics are plotted with respect to the angular window.

The angular window has to be estimated from the measurements. To be more precise and formal, the true angular window ϕ_W is defined as $\phi_F - \phi_R$ (see Figure 2.8 on page 16). Therefore, we propose an estimator of the angular window given by

$$\Phi_w = \Phi_f - \Phi_r. \tag{5.1}$$

The mean of Φ_w is equal to

$$\begin{aligned} \mu_{\Phi_w} &= E\{\Phi_f\} - E\{\Phi_r\}, \\ &= \left(\phi_F - p_0\frac{\phi_0}{2}\right) - \left(\phi_R + p_0\frac{\phi_0}{2}\right), \\ &= \phi_W - p_0\phi_0. \end{aligned} \tag{5.2}$$

The mean of Φ_w has a bias given by $-p_0\phi_0$. Fortunately, we know the value of $p_0\phi_0$, in order to remove the bias from the measurements. But, as the bias of C_k is null since $p_0 = 0$, it is also possible to derive ϕ_W by taking μ_{Φ_w} for the constant code C_k.

5.4 Simulator

Again, we use our simulator to validate the upper bound and evolution of the variance of Φ_b, as well as the bias of the mean angular window μ_{Φ_w}. The four parameters considered by the simulator are the angular window, the code (symbols and durations), the turret period, and the number of turns. The codes, the turret period, and the number of turns are known precisely in our experiments. So, in order to compare the simulated results with the measurements, the angular windows used in the simulator have been extracted from the experimental measurements of C_k, so it covers exactly the same range.

The simulated angular windows and variances are presented in Figure 5.1. Simulations confirm our theoretical results as bounds on variances (maximum of the curves) correspond to predicted bounds computed with equation (3.41). Also, the angular windows have a bias corresponding to values predicted by equation (5.2) (the numerical values are given in Table 5.1). The simulator confirms our theoretical results about the variance added by the codes. However, the simulator does not take into account the natural noise of the system. So we have to use the real system to measure this natural noise. The results are presented in the next section.

5.5 Experiments

The angular windows for each code are shown in the top plot of Figure 5.2 and the values of the biases are given in Table 5.1. As expected, the curves are linear with the angular window. But the biases observed for the angular windows are larger (in absolute value) than the theoretical biases. On the other hand, they increase with p_0 and ϕ_0, and the increments between the experimental biases are consistent with the theory.

The variances of the measurements for Φ_b are shown in the bottom plot of Figure 5.2. One can observe that the variance increases with p_0 and ϕ_0, respectively for C_k (the lowest), C_5, C_{5b}, and C_{5c} (the highest), for all angular windows. One also sees large variations, especially for C_{5b} and C_{5c}. This validates the existence of a dependency between Φ_r and Φ_f in function of the angular window. Examining the variances of Φ_r and Φ_f separately helps in this analysis

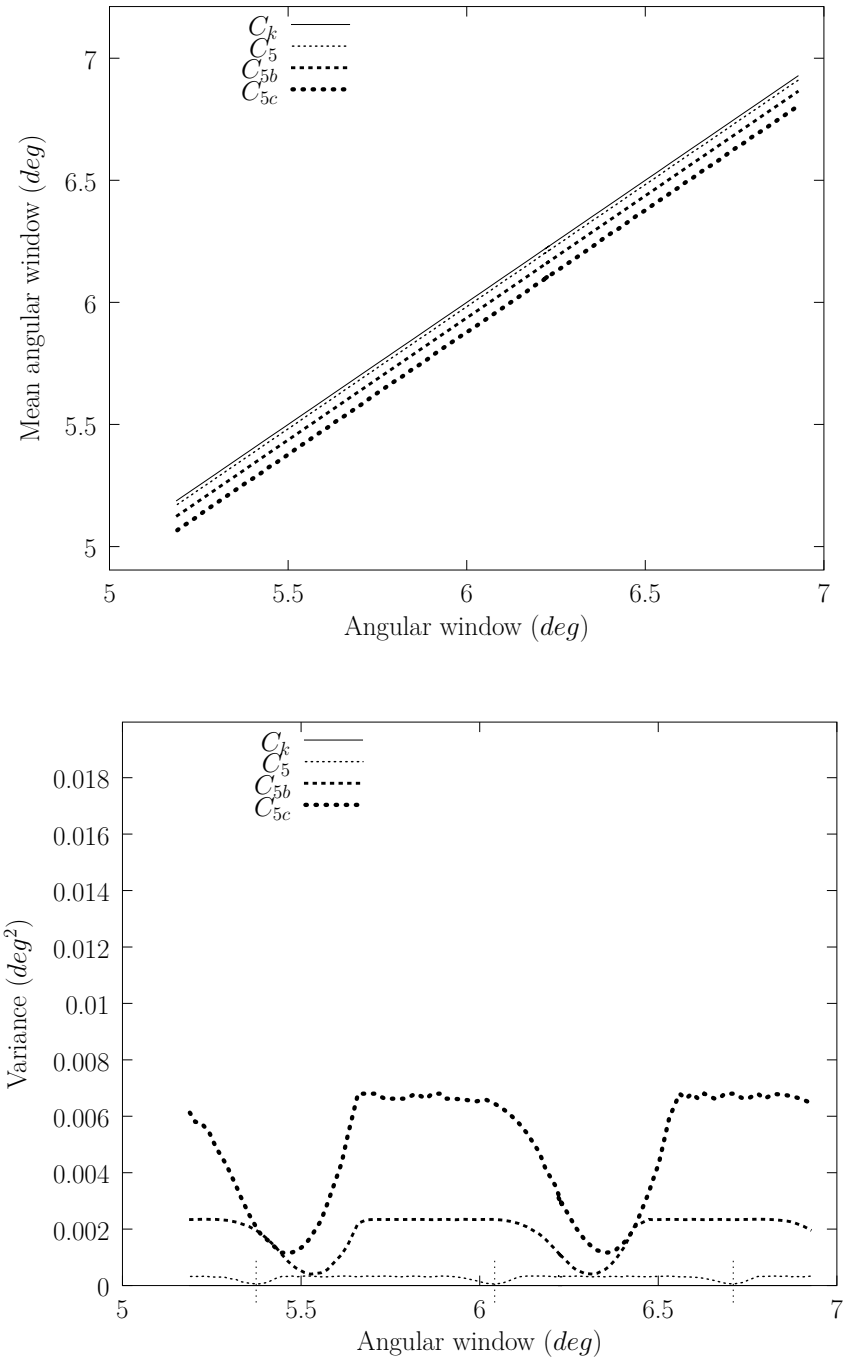

Figure 5.1: Results of simulations: values for the mean of the angular window Φ_w (top), and for the variance of the beacon angular position Φ_b (bottom).

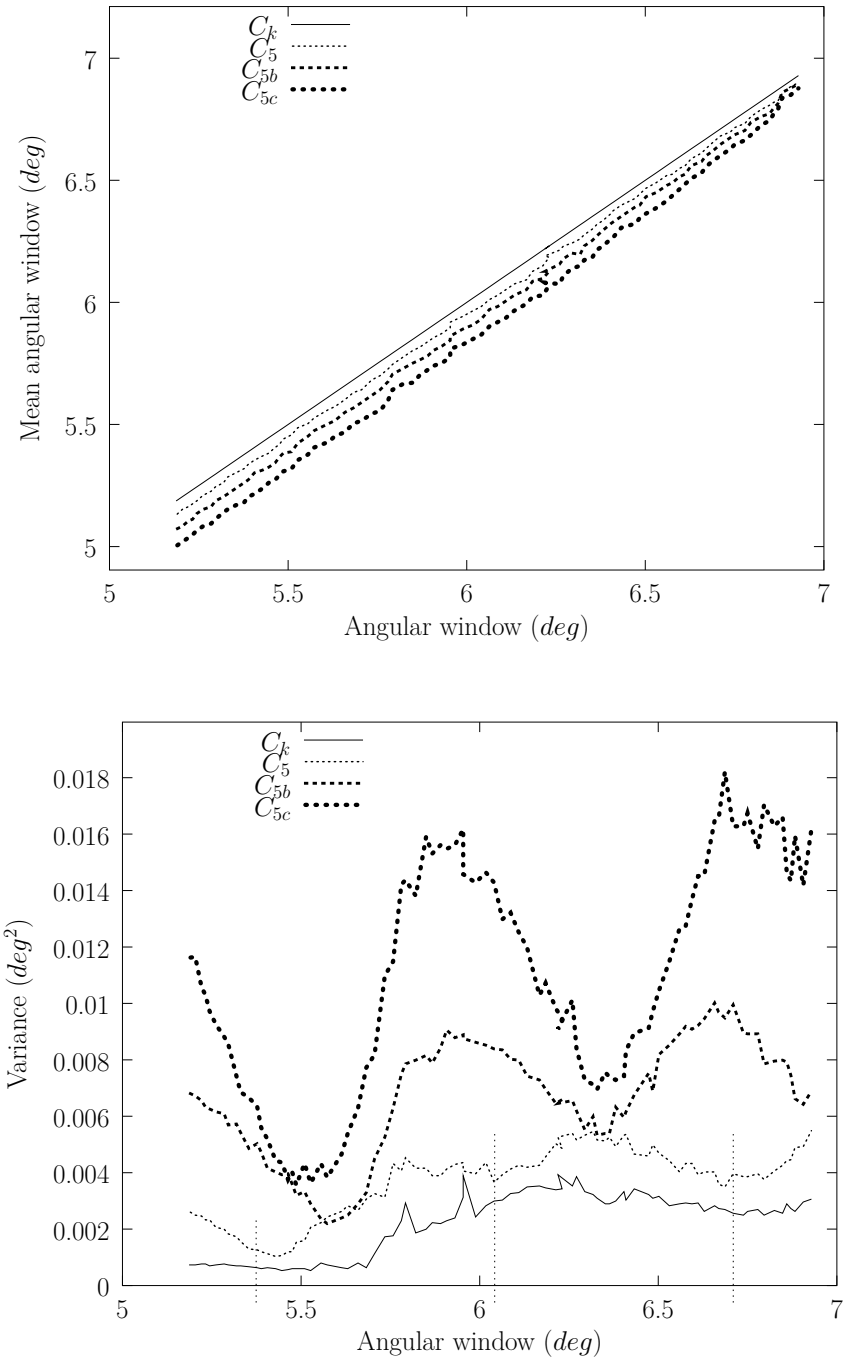

Figure 5.2: Results of Measurements: values for the mean of the angular window Φ_w (top), and for the variance of the beacon angular position Φ_b (bottom).

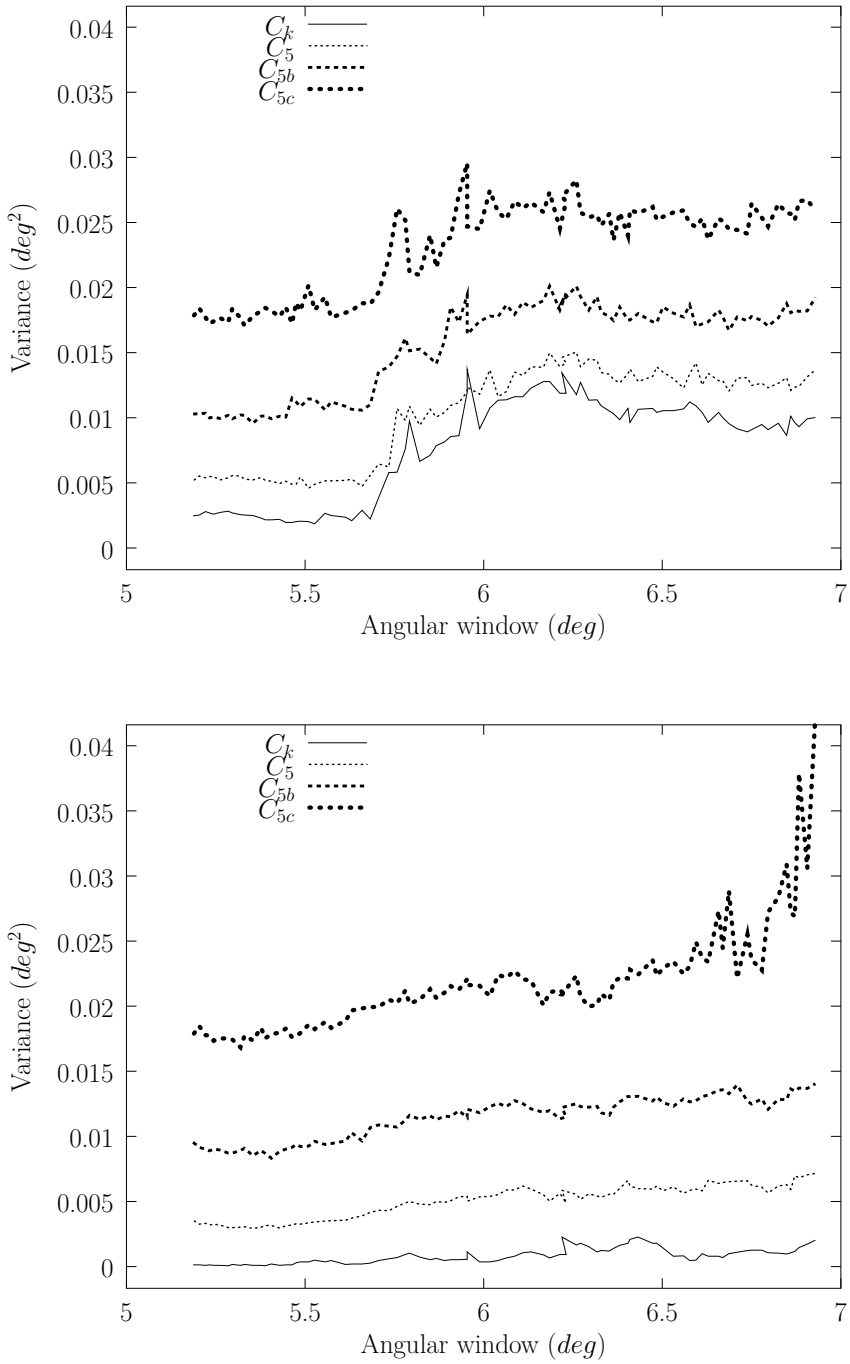

Figure 5.3: Results of Measurements: variance of the beacon angular rising Φ_r (top) and falling Φ_f (bottom) edges.

		Theory	Simulations	Experiments
Φ_w bias (deg)	C_k	0	0	0
	C_5	$-1.85\,10^{-2}$	$-1.78\,10^{-2}$	$-4.85\,10^{-2}$
	C_{5b}	$-6.35\,10^{-2}$	$-6.33\,10^{-2}$	$-9.53\,10^{-2}$
	C_{5c}	$-1.25\,10^{-1}$	$-1.22\,10^{-1}$	$-1.56\,10^{-1}$
Φ_b variance (deg^2)	C_k	0	0	$3.93\,10^{-3}$
	C_5	$3.43\,10^{-4}$	$3.36\,10^{-4}$	$\mathbf{5.49\,10^{-3}}$
	C_{5b}	$2.35\,10^{-3}$	$2.35\,10^{-3}$	$1.00\,10^{-2}$
	C_{5c}	$6.94\,10^{-3}$	$6.80\,10^{-3}$	$1.82\,10^{-2}$

Table 5.1: Comparison of the theoretical, simulated, and experimental values for the biases of Φ_w, and the maximum variances of Φ_b, for the different codes. Note that the experimental variances are higher than the theoretical and simulated variances, since they include the natural noise, inherent to the physical system. The value displayed in bold is the variance of the real system, with the real codes.

(see Figure 5.3). Whereas Φ_r and Φ_f variances are quasi linear with respect to the angular window, Φ_b is not linear despite the fact that the estimator Φ_b is a linear function of the estimators Φ_r and Φ_f. This confirms that a statistical relationship exists between Φ_r and Φ_f.

Of course, there is a difference with the simulations since the measurements include the natural noise of the system and, as a result, the variances of the measurements are higher than the simulations. If the artificial noise due to the codes was independent of the natural noise, the measured variances could be obtained by adding the natural noise (measured with C_k) to the simulated variances. However, this is not the case since the variances obtained with this hypothesis (not shown here) still remain lower (but close) than the real variances. This result indicates that the natural and artificial noises are not independent, and that the real noise is higher than their sum. But, despite this discrepancy, the general shape of the simulated and experimental curves is similar. In particular, the large variations in the curves, the locations of the extrema, as well as their relative distances match our experiments perfectly (compare bottom plots of Figure 5.1 and Figure 5.2). For example, the local minima of the variance associated to C_5 are located at

$$\phi_W = \{5.38\,deg,\ 6.04\,deg,\ 6.71\,deg\}\,, \tag{5.3}$$

in that range. These values are obtained by using equation (4.80), for $k = 7, 8, 9$, and $\phi_0 = 0.111\,deg$. The positions of the minima for C_5 are indicated by vertical dashed lines in the bottom plots of Figure 5.1 and Figure 5.2.

Finally, note that we are interested in finding the variance of the measured angles in BeAMS, in the whole working range. The maximum of the curve measured for C_5 yields a variance equal to $5.49\,10^{-3}\,deg^2$, or equivalently a standard deviation equal to $7.41\,10^{-2}\,deg$.

5.6 Discussions of the experiments

Our simulator provides values for the variances of Φ_b and biases of the angular window that match that of our theoretical model. However, they are some discrepancies between the experimental and simulated results. Amongst these discrepancies, the hypothesis that the natural variance is independent of the variance added by a code, as implemented in

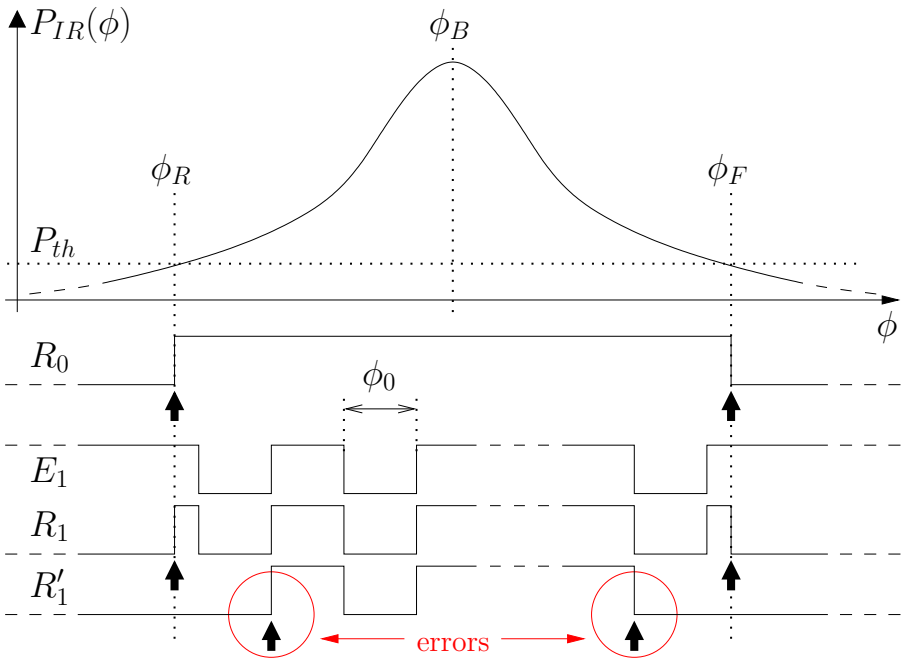

Figure 5.4: Illustration of the "AND hypothesis". The upper curve $P_{IR}(\phi)$ is the infrared power collected at the receiver while the turret is turning. R_0 is the special case corresponding to the non modulated infrared carrier wave (no OFF periods). E_1 is an example of emitted signal from a beacon. R_1 is the corresponding received signal at the receiver output, for an ideal receiver ($R_1 = E_1 \,\text{AND}\, R_0$). R'_1 is the corresponding received signal at the receiver output, for a practical receiver ($R'_1 \neq E_1 \,\text{AND}\, R_0$). The black arrows represent the measured values respectively for Φ_r to the left (first *Rising* edge) and for Φ_f to the right (last *Falling* edge). The encircled arrows emphasize errors made on Φ_r or Φ_f.

the simulator, is most subject to questioning. The reason for this is as follows. A detailed analysis of the receiver hardware shows the presence of an "Automatic Gain Control" (AGC) loop between the input and the demodulator. Typically, the gain is set to a high value when no signal is present for a "long time", resulting in a noisy first transition (Φ_r in our case). This gain then decreases over time, resulting in sharper transitions (especially the last one, Φ_f in our case). This characteristic is clearly identifiable from the variances of Φ_r and Φ_f for a non modulated signal C_k (see Figure 5.3). Indeed, for C_k, the variance of Φ_r stretches from about 0.0018 to 0.0137 deg^2 whereas the variance of Φ_f stretches from about 0.000066 to 0.0023 deg^2. It appears that the gain value depends on the past values of the received signal and the duration of the OFF periods, and this produces a non constant natural variance over time. So, we have to consider this effect in tightening the agreement between theoretical and practical results. But it is no small task to consider this effect because it relates to the hardware.

Also, we have to consider another effect of the receiver hardware. In Section 3.2, we supposed that the received signal R_i could be modeled as the logical AND between the emitted signal E_i and R_0. However, it is not sure that a short leading or tailing burst (shorter than a bit) could trigger the receiver. This phenomenon is depicted in Figure 5.4. R_0 is the special case corresponding to the non modulated infrared carrier wave (no OFF

periods). E_1 is an example of emitted signal from a beacon (we use here a simpler code than ours for the purpose of illustration, but this does not change the conclusions). R_1 is the corresponding received signal at the receiver output, for an ideal receiver (*i.e.* R_1 is the logical AND between E_1 and R_0). R'_1 is the corresponding received signal at the receiver output, for a practical receiver. In that case, the first and last bursts are not detected because they are too short. This has the effect of virtually increasing the OFF period by a quantity equal to the minimum burst duration required to trigger the receiver. In our case, we have determined experimentally that this minimum duration is equal to about 56 % of T_0. In order to validate this hypothesis (called "AND hypothesis"), we have implemented this effect in our simulator. Figure 5.5 presents the new simulation results obtained by taking into account the natural noise (extracted from the measures of C_k), as well as the "AND hypothesis". With these modifications, the simulations are closer to the experiments (compare both graphics in Figure 5.5). From a practical point of view, it means that we have to modify the actual values of ϕ_0 and p_0 in order to use the upper bound (equation (3.41)) adequately (ϕ_0 and p_0 have to be increased by 56 % from their theoretical values).

5.7 Measuring the accuracy

In the previous sections, we have analyzed the variance of the measures of BeAMS. This variance represents the precision of the system. But we also want to evaluate the accuracy, that is the difference between the mean of the measures, and the actual (true) value of the beacon angular position. In contrast with the ease of measuring the precision of the system, the accuracy is difficult to measure in practice (because it would require a precise optical setup). However, there is one way to determine the accuracy for BeAMS. In order to do that, we note that the position computed by any triangulation algorithm depends only on the difference between pairs of angles [32,67], whereas the orientation depends directly on the angles. So, a constant bias in the angles does not affect the position, but only the orientation, which has to be calibrated with the robot heading, anyway. It means that a problem arises if the biases are not the same for all angles. To verify this in BeAMS, we can plot the mean of the angle measurements versus the angular window (see Figure 5.6). This figure clearly shows some fluctuations of the mean angle in function of the angular window (or the received power); this indicates that there is a non constant bias in the measures, in the whole working range. This bias also has its origin by the presence of the AGC in the receiver. Biases in the measurements are due to delayed response times at the receiver. These delays are subject to the natural noise, and we have shown earlier that the natural noise depends on the code. It means that the *PDFs* of the noises at the beginning and at the end of the angular window are different, and so are their expectations, explaining that the bias changes with the angular window (or the received power). We can observe that the fluctuations of the bias occur within a range of 0.23 *deg* (see Figure 5.6), and that this range does not depend on the code. This value is our measure for the accuracy.

To be comprehensive, we present a comparison of different angle measurement systems in Table 5.2. Only the rotating systems similar to BeAMS are presented. As explained earlier, it is difficult to compare systems because the performance criterion and test condition, as well as the available information are different. Some authors mention absolute maximum error values or RMS values, while others provide standard deviation values. Most of the time, authors ignore the notion of the accuracy, and sometimes the angular resolution is expressed

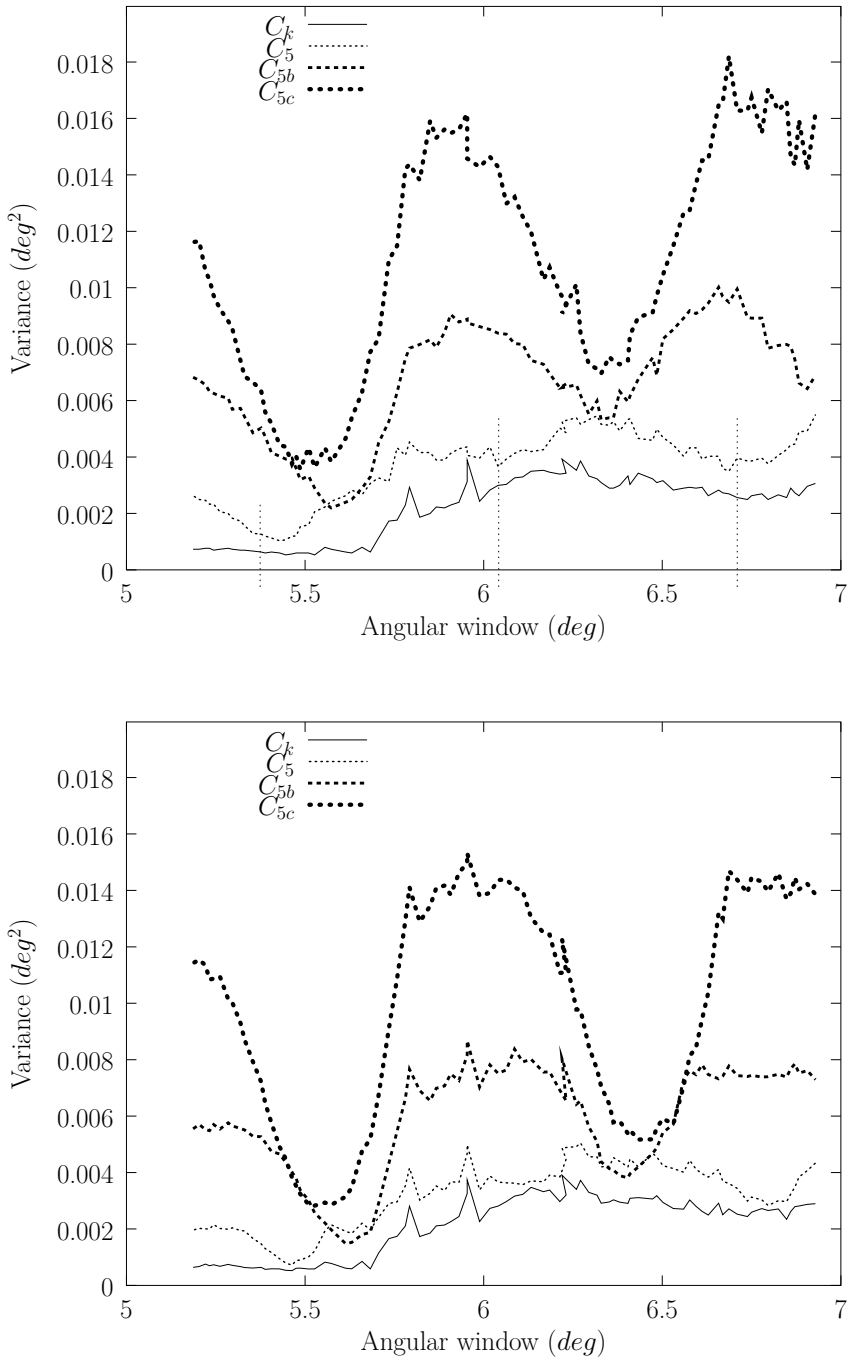

Figure 5.5: Results of simulations: values for the variance of the beacon angular position Φ_b by taking into account the natural noise and the "AND hypothesis" (bottom plot). The top plot is the result of the measurements, reminded here for the comparison (same as Figure 5.2).

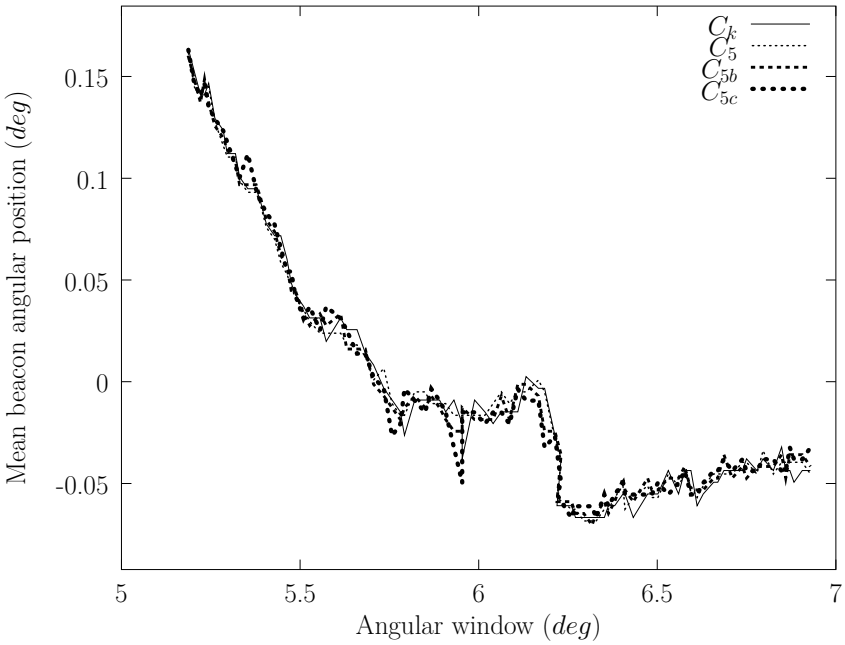

Figure 5.6: Variation in the mean beacon angular positions versus the angular window.

Commercial systems	**performance**	**acquisition rate**
SICK Nav2000	$0.1\,deg$	$8\,Hz$
GCS LS6	a fraction of a degree	$8\,Hz$
NDC LazerWay	$0.1\,deg$	$6\,Hz$
DBIR LaserNav	$0.03\,deg$	$10\,Hz$
TRC Beacon Nav. Sys.	$0.125\,deg$	$1\,Hz$
SSIM RobotSense	$0.17\,deg$	$10-40\,Hz$
Prototypes	**performance**	**acquisition rate**
Zalama *et al.* [89]	$0.6\,deg$ max.	$1-10\,Hz$
Kemppainen *et al.* [47]	$1.5\,deg$	not available
Brkic *et al.* [19]	$5-10\,cm$ $(2\,deg)$	$10\,Hz$
BeAMS	$0.24\,deg$	$10\,Hz$

Table 5.2: Comparison of different angle measurement systems.

in the terms of the precision or of the accuracy. Also, some authors express the performance of the positioning algorithm in meters, and nothing is said about angles (*e.g.* for the system of Brkic *et al.*, we guess an angular precision of $2\,deg$). Finally, some systems have a variable rotating speed, leading to a non constant performance (*e.g.* for the system by Zalama *et al.*, we give the precision at $10\,Hz$ for the comparison). For BeAMS, we have provided values either for the variance and for the accuracy. But, if we have to provide a single error value, we can combine both measures using this general result: $\sqrt{var + bias^2} = 0.24\,deg$. Table 5.2 shows that BeAMS has a better performance than other prototypes and is close to state of the art commercial systems (except for the DBIR LaserNav which is no longer manufactured). It should be noted that the performance of BeAMS are mainly downgraded by the accuracy, despite the low value of the variance. But, the performance is still good and it was sufficient in our application (EUROBOT contest). However, for a large deployment of the system, we would recommend the design and use of a new fitted receiver, in which we can fully control (or have access to) the AGC, in order to cancel the bias and reach the highest performance (limited by the variance only).

Until now, we have assumed that the sensor (robot) does not move during measurement, and of course, this is an unrealistic situation. In a practical situation, there are additional errors due to robot motion, vibrations, shocks, uneven floors, etc. Unlike the last ones (which cannot be predicted), the robot motion is controlled by the robot and can be taken into account by the positioning algorithm. The robot motion can be decomposed into a translation and a rotation, and it appears that the rotation is responsible for the most part of the error [53]. It should be noted that all rotating systems are subjects to this error since the robot motion has the effect of modifying the real turning speed of the sensor, and that the measurement principle relies on a constant speed. Taking into account the robot motion should be done, especially rotations because it can lead to high improvements in the computed position [53]. However, applying the corrections to the angles due to the robot motion is the task of the positioning algorithm. Nevertheless, the characteristics when the robot does not move (precision, accuracy) are still useful for data fusion algorithms, and should be the only ones to compare different systems.

5.8 Conclusions

In this chapter, we provide simulated and experimental results for the variance of the beacon angle estimator Φ_b. It appears that the results of the simulator are coherent with our theoretical results, but not entirely with the experimental results. Experimental results enlighten that the natural variance of the system is not independent on the artificial variance added by the codes because of the Automatic Gain Control (AGC) loop of the receiver, which is responsible for a small mismatch between the experimental and simulated results. But, despite this discrepancy, the general shape of the simulated and experimental curves is similar. Also, we enlighten another particularity of the receiver, called the "AND hypothesis". Indeed, it is not sure that a short leading or tailing burst (shorter than a bit) could trigger the receiver. This has the effect of virtually increasing the OFF period by a quantity equal to the minimum burst duration required to trigger the receiver. In our case, we have determined experimentally that this minimum duration is equal to about $56\,\%$ of T_0. If we take into account the natural noise, as well as the "AND hypothesis", it appears that the simulations are closer to the experiments.

In a practical situation, we would want to limit the variance added by the codes compared to the natural variance of the system. The theoretical bound, as well as the simulator are useful tools to help in this purpose. Note that our system achieves a low variance level on angles. The experimental values encountered in our system for the standard deviation of Φ_b range from 0.023 to 0.063 deg without codes and from 0.032 to 0.074 deg with codes, meaning that the noise added by our codes is small compared to the natural noise.

In addition to the variance, the accuracy is evaluated to 0.23 deg. If we combine the variance and the bias, the final error measure is evaluated to 0.24 deg ($\sqrt{var + bias^2}$). The comparison with other systems shows that BeAMS has a better performance than other prototypes and is comparable to state of the art commercial systems. It appears that the bias, due to the variations of the received power and the AGC, is responsible for the bigger part of the error. We think that the use of a dedicated receiver, in which we have full control of (or have access to) the AGC, would result in a far more precise system, limited mainly by the variance. Note that the error measure provided in this chapter is a first attempt to characterize our hardware system. Further details about the error measure are provided in Chapter 7.

Chapter 6

ToTal: a new triangulation algorithm

6.1 Introduction

In the previous chapters, we have described our new angle measurement sensor, and we have elaborated a mathematical model for the variance of the measured angles. In order to compute the *pose* (position and orientation) of the robot in the 2D plane, we have to combine these angle measurements into a positioning algorithm. The process of determining the robot pose based on angle measurements is generally termed triangulation. The word *triangulation* is a wide concept, which does not specify if the angles are measured from the robot or the beacons, neither imposes the number of angles used. In this work, we are interested in self position determination, meaning that the angles are measured from the robot location. Figure 6.1 illustrates our triangulation setup. Moreover, if only three beacons are used in self position determination, triangulation is also termed *Three Object Triangulation* by Cohen and Koss [21]. Here the general term *object* refers to a 2D point. A positioning algorithm based on more than three angles is termed *multiangulation*.

The fact that self position triangulation requires a minimum of three beacons to compute the pose unequivocally in a 2D plane (except on the circumference defined by the three beacons) explains why *Three Object Triangulation* algorithms take an important place in mobile robot positioning. More than three angles can be used to increase the precision in some pathological geometrical setups or to deal with harsh environments where some beacons might be out of sight of the sensor [13, 53, 75, 89]. Because of the availability of angle measurement systems, triangulation has emerged as a widely used, robust, accurate, and flexible technique [30, 75] for mobile robot positioning. It is important to note that a positioning algorithm based on angles is totally independent of the underlying angle measurement system, and both parts should be analyzed separately.

As explained earlier, our system has been designed for the EUROBOT contest. This contest allows a maximum of three fixed beacons around the moving area. As a consequence, we decided to develop our own *Three Object Triangulation* algorithm, for the following reason. Most of the many triangulation algorithms proposed so far have major limitations. For example, some of them need a particular beacon ordering, have blind spots, or only work within the triangle defined by the three beacons. More reliable methods exist, but they have an increasing complexity or they require to handle certain spatial arrangements separately.

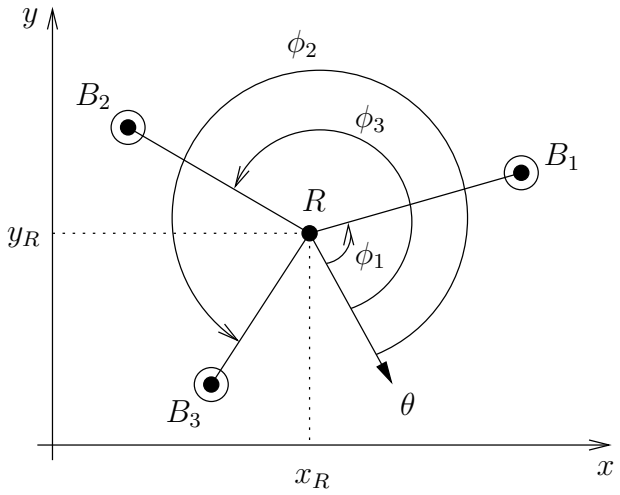

Figure 6.1: Triangulation setup in the 2D plane. R denotes the robot. B_i are the beacons. ϕ_i are the angle measurements for B_i, relative to the robot reference orientation θ. These angles may be used by a triangulation algorithm in order to compute the robot position $\{x_R, y_R\}$ and orientation θ.

In this chapter, we present a simple and new three object triangulation algorithm, named ToTal[1], that natively works in the whole plane, and for any beacon ordering. We also provide a comprehensive comparison with seventeen comparable algorithms, and show that our algorithm is faster and simpler than these algorithms. In addition to its inherent efficiency, our algorithm provides a useful and unique reliability measure, assessable anywhere in the plane, which can be used to identify pathological cases, or as a validation gate in data fusion algorithms.

The chapter is organized as follows. Section 6.2 reviews some of the numerous triangulation algorithms found in the literature. Our triangulation algorithm is described in Section 6.3. Section 6.4 presents simulation results and benchmarks. Then, we conclude the chapter in Section 6.5.

6.2 Related Work

6.2.1 Triangulation algorithms

The principle of triangulation exists for a long time, and many methods have been proposed so far. One of the first comprehensive reviewing work has been carried out by Cohen and Koss [21]. In their paper, they classify the triangulation algorithms into four groups: (1) *Geometric Triangulation*, (2) *Geometric Circle Intersection*, (3) *Iterative methods*, and (4) *Multiple Beacons Triangulation.*

The first group could be named *Trigonometric Triangulation,* because it makes an intensive use of trigonometric functions. Algorithms of the second group determine the parameters (radius and center) of two (of the three) circles passing through the beacons and the robot, then they compute the intersection between these two circles. Methods of the first and second

[1] ToTal: **T**hree **o**bject **T**riangulation **al**gorithm.

groups are typically used to solve the three object triangulation problem. The third group linearizes the trigonometric relations to converge to the robot position after some iterations, from a starting point (usually the last known robot position). In the iterative methods, they also present *Iterative Search*, which consists in searching the robot position through the possible space of orientations, and by using a minimization procedure of a closeness measure. The fourth group addresses the more general problem of finding the robot pose from more than three angle measurements (usually corrupted by errors), which is an overdetermined problem.

Several authors have noticed that the second group (*Geometric Circle Intersection*) is the most popular for solving the three object triangulation problem [34, 59]. The oldest Geometric Circle Intersection algorithm was described by McGillem and Rappaport [58, 59]. Font-Llagunes and Batlle [34] present a similar method, but they first change the reference frame so to relocate beacon 2 at the origin and beacon 3 on the X axis. They compute the robot position in this reference frame and then, they apply a rotation and translation to return to the original reference frame. Zalama *et al.* [88, 89] present a hardware system to measure angles to beacons and a method to compute the robot pose from three angle measurements. A similar hardware system and method based on the work of Zalama *et al.* [88,89] is described by Tsukiyama [84]. Kortenkamp [49] presents a method which turns to be exactly the same as the one described by Cohen and Koss [21]. All these methods compute the intersection of two of the three circles passing through the beacons and the robot. It appears that they are all variations or improvements of older methods of McGillem and Rappaport, or Cohen and Koss. The last one is described by Lukic *et al.* [19, 54], but it is not general, as it only works for a subset of all possible beacon locations.

Some newer variations of Geometric/Trigonometric triangulation algorithms are also described in literature. In [30], Esteves *et al.* extend the algorithm presented earlier by Cohen and Koss [21] to work for any beacon ordering and to work outside the triangle formed by the three beacons. In [31,32], they describe the improved version of their algorithm to handle the remaining special cases (when the robot lies on the line joining two beacons). They also analyze the position and orientation error sensitivity in [32]. Whereas Easton and Cameron [28] concentrate on an error model for the three object triangulation problem, they also briefly present an algorithm belonging to this family. It turns out that their simple method works in the whole plane and for any beacon ordering. The work of Hmam [42] is based on Esteves *et al.*, and Cohen and Koss. He presents a method, valid for any beacon ordering, that divides the whole plane into seven regions and handles two specific configurations of the robot relatively to the beacons. In [55], Madsen and Andersen describe a vision based positioning system. Such an algorithm belongs to the trigonometric triangulation family as the vision system is used to measure angles between beacons.

It should be noted that the "three object triangulation problem" is also known as the "three point resection problem" in the surveying engineering research area. In this field, the beacons are often called *stations*, and the angles (or azimuths) are measured with a goniometer. As it is a basic operation for decades in the surveying field, there are many procedures (more than 500) to solve this problem, numerically as well as graphically [35]. Surprisingly, there is almost no link between these two fields, except the recent work of Font-Llagunes and Batlle [35], and the older one of McGillem and Rappaport [58, 59]. Therefore, it appears that some algorithms from the two fields are similar, but have different names. One of the most famous and oldest procedures is called the Kaestner-Burkhardt method, also known as the Pothonot-Snellius method [20]. It appears that this method is similar to the one described by McGillem and Rappaport [58,59], which is a trigonometric approach. Then,

there is the Collins method [20], which is a trigonometric solution, close to the one described by Esteves *et al.* [32], or Cohen and Koss [21]. Also, there is the Cassini method [20], similar to the method of Easton and Cameron [28], both being a trigonometric approach. Finally, there is the Tienstra method [20,68], which is a completely different approach, as it makes use of the barycentric coordinates in order to express the robot position as a weighted sum of the beacons coordinates. This method is known for a long time, but has been concisely proofed in the recent work of Porta and Thomas [68]. Despite that the three point resection problem is known for a long time and has many solutions, some newer works are still emerging. For example, Font-Llagunes and Batlle [35] have published a new method, which uses straight lines intersection and similar triangle property. To our knowledge, the most recent work has been carried out by Ligas [52]. It turns out that both Ligas's method and ToTal rely on the idea of using the radical axis of two circles. However, Ligas intersects one radical axis and one circle, whereas our algorithm intersects the three radical axis of the three pairs of circles[2]. Likewise, Ligas also uses only two trigonometric functions (like our method ToTal), and as a consequence, it is one of the most efficient methods (with ToTal), as shown is Section 6.4.2.

Some of the Multiple Beacons Triangulation (multiangulation) algorithms are described hereafter. One of the first work in this field was presented by Avis and Imai [7]. In their method, the robot measures k angles from a subset of n indistinguishable landmarks, and therefore they produce a bounded set a valid placements of the robot. Their algorithm runs in $\mathcal{O}(kn^2)$ if the robot has a compass (*i.e.* if the orientation is known) or in $\mathcal{O}(kn^3)$ otherwise. One of the most famous algorithm was introduced by Betke and Gurvits [13]. They use an efficient representation of landmark 2D locations by complex numbers to compute the robot pose. The landmarks are supposed to have an identifier known by the algorithm. The authors show that the complexity of their algorithm is proportional to the number of beacons. They also perform experiments with noisy angle measurements to validate their algorithm. Finally, they explain how the algorithm deals with outliers. A newer interesting approach is proposed by Shimshoni [75]. He presents an efficient SVD based multiangulation algorithm from noisy angle measurements, and explains why transformations have to be applied to the linear system in order to improve the accuracy of the solution. The solution is close to the optimal solution computed with nonlinear optimization techniques, while being more than a hundred times faster. Moreover, his method seems easier to implement, compared to the one of Betke and Gurvits [13]. Briechle and Hanebeck [18] present a new localization approach in the case of relative bearing measurements by reformulating the given localization problem as a nonlinear filtering problem.

Siadat and Vialle [77] describe a multiangulation method based on the Newton-Raphson iterative method to converge to a solution, by minimizing an evaluation criterion. Lee *et al.* [51] present another iterative method similar to Newton-Raphson. Their algorithm was first designed to work with three beacons, but it can be generalized to a higher number of beacons. The initial point of the convergence process is set to the center of the beacons and good results are obtained after only four steps.

Sobreira *et al.* [80] present an hybrid triangulation method working with two beacons only. They use a concept similar to the *running-fix* method introduced by Bais in [9], in which the robot has to move by a known distance to create a virtual beacon measure and to compute the robot pose after it has stopped to take another angle measurement. In [79], Sobreira *et al.* perform an error analysis of their positioning system. In particular, they express the

[2]Note that the paper of Ligas [52] is posterior to ours [67].

position uncertainty originated by errors on measured angles in terms of a surface. Sanchiz *et al.* [74] describe another multiangulation method based on *Iterative Search* and circular correlation. They first compute the robot orientation by maximizing the circular correlation between the expected beacons angles and the measured beacons angles. Then, a method similar to *Iterative Search* is applied to compute the position. Hu and Gu [43] present a multiangulation method based on Kohonen neural network to compute the robot pose and to initialize an extended Kalman filter used for navigation.

6.2.2 Brief discussion

It is difficult to compare all the above mentioned algorithms, because they operate in different conditions and have distinct behaviors. In practice, the choice is dictated by the application requirements and some compromises. For example, if the setup contains three beacons only or if the robot platform has a low computing power, methods of the first and second groups are the best candidates. Methods of the third and fourth groups are appropriate if the application must handle multiple beacons and if it can accommodate to a higher computational cost. The main drawback of the third group is the convergence issue (existence or uniqueness of the solution) [21]. The main drawback of the fourth group is the computational cost [13, 18, 75].

The drawbacks of the first and second group are usually a lack of precision related to the following elements: (1) the beacon ordering needed to get the correct solution, (2) the consistency of the methods when the robot is located outside the triangle defined by the three beacons, (3) the strategy to follow when falling into some particular geometrical cases (*e.g.* when computing trigonometric functions with arguments like 0 or π, division by 0, etc), and (4) the reliability measure of the computed position. Simple methods of the first and second groups usually fail to propose a proper answer to all of these concerns. For example, to work in the entire plane and for any beacon ordering (for instance, Esteves *et al.* [32]), they have to consider a set of special geometrical cases separately, resulting in a lack of clarity. Finally, to our knowledge, none of these algorithms gives a realistic reliability measure of the computed position.

6.2.3 Other aspects of triangulation

For now, we have focused on the description of triangulation algorithms which are used to compute the position and orientation of the robot. Other aspects of triangulation have to be considered as well to achieve an optimal result on the robot pose in a practical situation. These are: (1) the sensitivity analysis of the triangulation algorithm, (2) the optimal placement of the landmarks, (3) the selection of some landmarks among the available ones to compute the robot pose, and (4) the knowledge of the true landmark locations in the world and the true location of the angular sensor on the robot.

Kelly [45] uses the famous Geometric Dilution of Precision (GDOP) concept, used in GPS error analysis, and applies it to range based and angle based positioning techniques. He derives two useful formulas in the case of two beacons. Madsen *et al.* [56] perform a sensitivity analysis of their triangulation algorithm. They use the classical first order propagation of angle measurement errors through covariance matrix and Jacobian to derive the precision of location. Easton and Cameron [28] present the same error analysis as that of Madsen *et al.* but, in addition to the angle uncertainty, they also take into account the beacon location uncertainty. They also present some simulations for various beacons configurations as well as

some metrics to evaluate their model's performance.

Optimal landmark placement has been extensively studied. Sinriech and Shoval [76, 78] define a nonlinear optimization model used to determine the position of the minimum number of beacons required by a shop floor to guarantee an accurate and reliable performance for automated guided vehicles (AGVs). Demaine *et al.* [23] present a polynomial time algorithm to place reflector landmarks such that the robot can always localize itself from any position in the environment, which is represented by a polygonal map. Tedkas and Isler [81, 82] address the problem of computing the minimum number and placement of sensors so that the localization uncertainty at every point in the workspace is less than a given threshold. They use the uncertainty model for angle based positioning derived by Kelly [45].

Optimal landmark selection has been studied by Madsen *et al.* [55, 56]. They propose an algorithm to select the best triplet among several landmarks seen by a camera, yielding to the best position estimate. The algorithm is based on a "goodness" measure derived from an error analysis which depends on landmarks and on the robot relative pose.

Having a good sensor providing 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 the measuring angle range (a complete revolution). Moreover, the beacon locations that are generally measured manually are subject to inaccuracies, affecting directly the positioning algorithm. In their paper, Loevsky and Shimshoni [53] propose a method to calibrate the sensor and a method to correct the measured beacon locations. They show that their procedure is effective and mandatory to achieve a good positioning performance.

In the remainder of this chapter, we concentrate on three object triangulation methods. We present a new three object triangulation algorithm that works in the entire plane (except when the beacons and the robot are concyclic or collinear), and for any beacon ordering. Moreover, it minimizes the number of trigonometric computations, and provides a unique quantitative reliability measure of the computed position.

6.3 Description of a New Three Object Triangulation Algorithm

Our motivation for a new triangulation algorithm is fourfold: (1) we want it to be independent of the beacon ordering, (2) the algorithm must also be independent of the relative positions of the robot and the beacons, (3) the algorithm must be fast and simple to implement in a dedicated processor, and (4) the algorithm has to provide a criterion to qualify the reliability of the computed position.

Our algorithm, named ToTal, belongs to the family of *Geometric Circle Intersection* algorithms (that is, the second group). It first computes the parameters of the three circles passing through the robot and the three pairs of beacons. Then, it computes the intersection of these three circles, by using all the three circles parameters (not only two of them, to the contrary of other methods).

Our algorithm relies on two assumptions: (1) the beacons are distinguishable (a measured angle can be associated to a given beacon), and (2) the angle measurements from the beacons are taken separately, and relatively to some reference angle θ, usually the robot heading (see Figure 1.1). Note that the second hypothesis simply states that angles are given by a rotating angular sensor. Such sensors are common in mobile robot positioning using

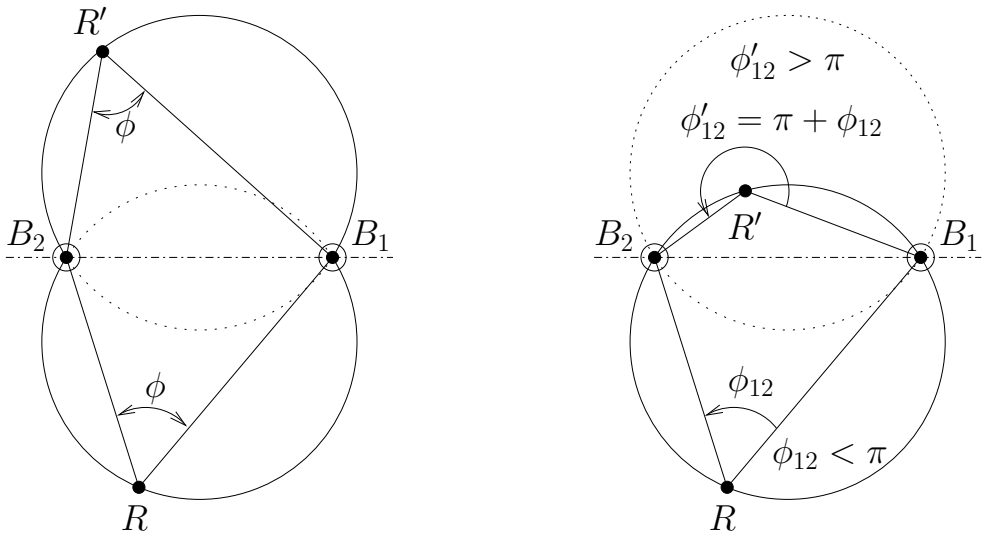

Figure 6.2: Left-hand side drawing: the locus of points R that *see* two fixed points B_1 and B_2 with a constant angle ϕ, in the 2D plane, is formed by two arcs of circle. Right-hand side drawing: the ambiguity is removed by taking the following convention: $\phi_{12} = \phi_2 - \phi_1$.

triangulation [15, 16, 51, 59, 65, 88]. By convention, in the following, we consider that angles are measured counterclockwise (CCW), like angles on the trigonometric circle. Inverting the rotating direction to clockwise (CW) would only require minimal changes of our algorithm.

6.3.1 First part of the algorithm: the circle parameters

In a first step, we want to calculate the locus of the robot positions R, that *see* two fixed beacons, B_1 and B_2, with a constant angle ϕ, in the 2D plane. It is a well-known result that this locus is an arc of the circle passing through B_1 and B_2, whose radius depends on the distance between B_1 and B_2, and ϕ (Proposition 21 of Book III of Euclid's Elements). 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 (see the left-hand side drawing of Figure 6.2).

A robot that measures an angle ϕ between two beacons can stand on either of these two arcs. This case occurs if the beacons are not distinguishable or if the angular sensor is not capable to measure angles larger than π (like a vision system with a limited field of view, as used by Madsen *et al.* [55]). To avoid this ambiguity, we impose that, as shown in the right-hand drawing of Figure 6.2, the measured angle between two beacons B_1 and B_2, denoted ϕ_{12}, is always computed as $\phi_{12} = \phi_2 - \phi_1$ (this choice is natural for a CCW rotating sensor). This is consistent with our measurement considerations and it removes the ambiguity about the locus, but it requires that beacons are indexed and that the robot is capable to establish the index of any beacon. As a result, the locus is a single circle passing through R, B_1, and B_2. 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 $\phi_{12} > \pi$. In the following, we compute the circle parameters.

The circle equation may be derived by using the complex representation of 2D points (Argand diagram), and by expressing angles as the argument of complex numbers. In

particular, the angle of $(B_2 - R)$ is equal to that of $(B_1 - R)$ plus ϕ_{12}. Equivalently,

$$\arg\left\{\frac{B_2 - R}{B_1 - R}\right\} = \phi_{12}, \tag{6.1}$$

$$\Rightarrow \arg\left\{(B_2 - R)\,\overline{(B_1 - R)}\right\} = \phi_{12}. \tag{6.2}$$

Then, if we substitute R, B_1, B_2, respectively by $(x+iy)$, (x_1+iy_1), (x_2+iy_2), we have that

$$\arg\left\{(x_2 + iy_2 - x - iy)\,(x_1 - iy_1 - x + iy)\,e^{-i\phi_{12}}\right\} = 0, \tag{6.3}$$

$$\begin{aligned}\Rightarrow -\sin\phi_{12}\,(x_2 - x)\,(x_1 - x) + \sin\phi_{12}\,(y_2 - y)\,(y - y_1)\\ + \cos\phi_{12}\,(x_2 - x)\,(y - y_1) + \cos\phi_{12}\,(y_2 - y)\,(x_1 - x) = 0,\end{aligned} \tag{6.4}$$

where $i = \sqrt{-1}$. After lengthy simplifications (see appendix B.1 for details), we find the locus

$$(x - x_{12})^2 + (y - y_{12})^2 = R_{12}^2, \tag{6.5}$$

which is a circle whose center $\{x_{12},\, y_{12}\}$ is located at

$$x_{12} = \frac{(x_1 + x_2) + \cot\phi_{12}\,(y_1 - y_2)}{2}, \tag{6.6}$$

$$y_{12} = \frac{(y_1 + y_2) - \cot\phi_{12}\,(x_1 - x_2)}{2}, \tag{6.7}$$

and whose squared radius equals

$$R_{12}^2 = \frac{(x_1 - x_2)^2 + (y_1 - y_2)^2}{4\,\sin^2\phi_{12}}. \tag{6.8}$$

The three last equations may also be found in the work of Font-Llagunes and Batlle [34], in an alternate formulation. The replacement of ϕ_{12} by $\pi + \phi_{12}$ in the above equations yields the same circle parameters (Figure 6.2, right), which is consistent with our measurement considerations. For an angular sensor turning in the CW direction, these equations are identical except that one has to change the sign of cot(.) in equations (6.6) and (6.7).

Hereafter, we use the following notations:

- B_i is the beacon i, whose coordinates are $\{x_i,\, y_i\}$,
- R is the robot position, whose coordinates are $\{x_R,\, y_R\}$,
- ϕ_i is the angle for beacon B_i,
- $\phi_{ij} = \phi_j - \phi_i$ is the bearing angle between beacons B_i and B_j,
- $T_{ij} = \cot(\phi_{ij})$,
- $\mathcal{C}_{ij}$ is the circle passing through B_i, B_j, and R,
- c_{ij} is the center of $\mathcal{C}_{ij}$, whose coordinates are $\{x_{ij},\, y_{ij}\}$

$$x_{ij} = \frac{(x_i + x_j) + T_{ij}\,(y_i - y_j)}{2}, \tag{6.9}$$

$$y_{ij} = \frac{(y_i + y_j) - T_{ij}\,(x_i - x_j)}{2}, \tag{6.10}$$

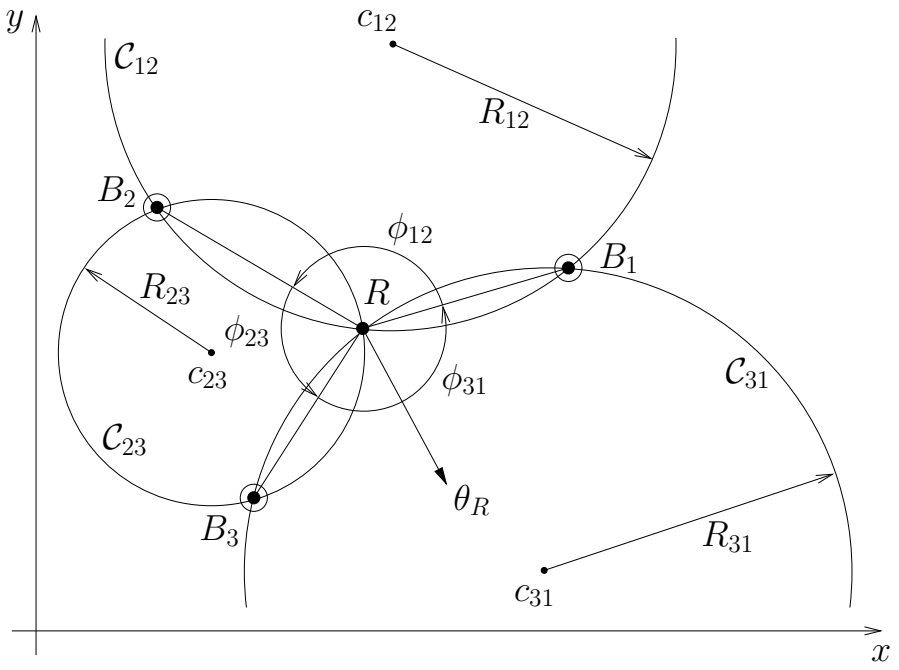

Figure 6.3: Triangulation setup in the 2D plane, using the geometric circle intersection. R is the robot. B_1, B_2, and B_3 are the beacons. ϕ_{ij} are the angles between B_i, R, and B_j. $\mathcal{C}_{ij}$ are the circles passing through B_i, R, and B_j. R_{ij} and c_{ij} are respectively the radii and center coordinates of $\mathcal{C}_{ij}$. θ_R is the robot heading orientation.

- R_{ij} is the radius of $\mathcal{C}_{ij}$, derived from

$$R_{ij}^2 = \frac{(x_i - x_j)^2 + (y_i - y_j)^2}{4 \sin^2 \phi_{ij}}. \tag{6.11}$$

All the previous quantities are valid for $i \neq j$, otherwise the circle does not exist. Also, we have to consider the case $\phi_{ij} = k\pi$, $k \in \mathbb{Z}$. In that case, the sin(.) and cot(.) are equal to zero, and the circle degenerates as the B_iB_j line (infinite radius and center coordinates). In a practical situation, it means that the robot stands on the B_iB_j line, and measures an angle $\phi_{ij} = \pi$ when being between the two beacons, or $\phi_{ij} = 0$ when being outside of the B_iB_j segment. These special cases are discussed later.

6.3.2 Second part of the algorithm: the circles intersection

In the previous section, we showed that each bearing angle ϕ_{ij} between beacons B_i and B_j constraints the robot to be on a circle $\mathcal{C}_{ij}$, passing through B_i, B_j, and R. The situation is illustrated in Figure 6.3. The parameters of the circles are given by equations (6.9), (6.10), and (6.11). Common methods use two of the three circles to compute the intersections (when they exist), one of which is the robot position, the second being the common beacon of the two circles. This requires to solve a quadratic system and to choose the correct solution for the robot position (see Font-Llagunes and Batlle [34], for instance). Moreover the choice of the two circles is arbitrary and usually static, whereas this choice should depend on the measured angles or beacons and robot relative configuration to have a better numerical behavior.

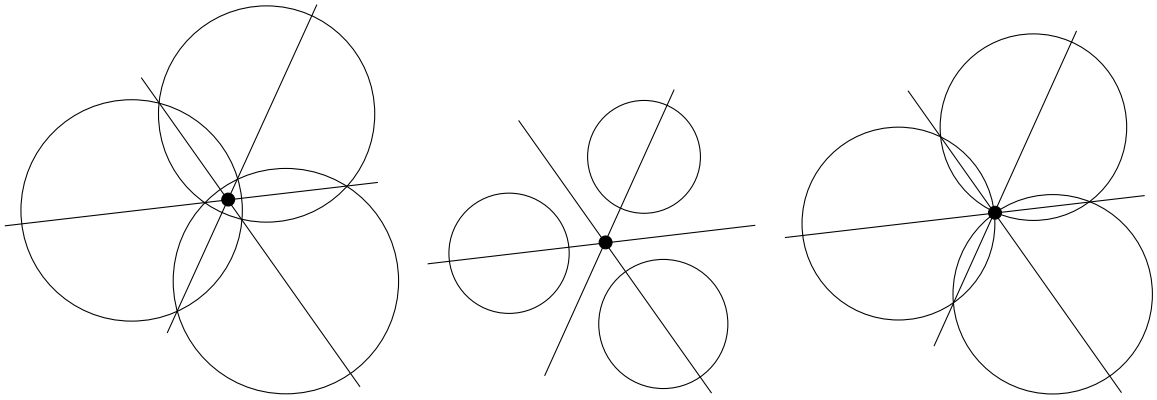

Figure 6.4: The black point is the power center of three circles for various configurations. It is the unique point having the same power with respect to the three circles. The power center is the intersection of the three power lines.

Hereafter, we propose a novel method to compute this intersection, by using all of the three circles, and by reducing the problem to a linear problem[3]. To understand this elegant method, we first introduce the notion of *power center* (or *radical center*) of three circles. The *power center* of three circles is the unique point of equal *power* with respect to these circles [29]. The *power* of a point p relative to a circle $\mathcal{C}$ is defined as

$$\mathcal{P}_{\mathcal{C},p} = (x - x_c)^2 + (y - y_c)^2 - R^2, \tag{6.12}$$

where $\{x, y\}$ are the coordinates of p, $\{x_c, y_c\}$ are the circle center coordinates and R is the circle radius. The power of a point is null onto the circle, negative inside the circle, and positive outside the circle. It defines a sort of relative distance of a point from a given circle. The *power line* (or *radical axis*) of two circles is the locus of points having the same power with respect to both circles [29]; in other terms, it is also the locus of points at which tangents drawn to both circles have the same length. The power line is perpendicular to the line joining the circle centers and passes through the circle intersections, when they exist. Monge demonstrated that, when considering three circles, the three power lines, defined by the three pairs of circles are concurring in the power center [29]. Figure 6.4 shows the power center of three circles for three configurations. The power center is always defined, except when at least two of the three circle centers are equal, or when the circle centers are collinear (parallel power lines).

The third case of Figure 6.4 (right-hand drawing) is remarkable as it perfectly matches our triangulation problem (Figure 6.3). Indeed, the power center of three concurring circles corresponds to their unique intersection. In our case, we are sure that the circles are concurring since we have

$$\phi_{31} = -(\phi_{12} + \phi_{23}), \tag{6.13}$$

[3]The idea of using all the parameters from the three circles is not new, and has been used at least by the authors of the following report: Fuentes, O.; Karlsson, J.; Meira, W.; Rao, R.; Riopka, T.; Rosca, J.; Sarukkai, R.; Van Wie, M.; Zaki, M.; Becker, T.; Frank, R.; Miller, B. and Brown, C. M.; Mobile Robotics 1994, Technical Report 588, The University of Rochester Computer Science Department, Rochester, New York 14627, June 1995 (see `http://www.cs.cmu.edu/afs/.cs.cmu.edu/Web/People/motionplanning/papers/sbp_papers/integrated2/fuentas_mobile_robots.pdf`). However, they did not simplify their algorithm as far as we do in our developments.

by construction (only two of the three bearing angles are independent), even in presence of noisy angle measurements ϕ_1, ϕ_2, and ϕ_3. It has the advantage that this intersection may be computed by intersecting the power lines, which is a linear problem. The power line of two circles is obtained by equating the power of the points relatively to each circle (equation (6.12)). In our problem, the power line of $\mathcal{C}_{12}$ and $\mathcal{C}_{23}$ is given by

$$\begin{aligned}(x-x_{12})^2+(y-y_{12})^2-R_{12}^2 &= (x-x_{23})^2+(y-y_{23})^2-R_{23}^2\\ \Rightarrow x\,(x_{12}-x_{23})+y\,(y_{12}-y_{23}) &= \frac{x_{12}^2+y_{12}^2-R_{12}^2}{2}-\frac{x_{23}^2+y_{23}^2-R_{23}^2}{2}\\ \Rightarrow x\,(x_{12}-x_{23})+y\,(y_{12}-y_{23}) &= k_{12}-k_{23},\end{aligned}$$

where we introduce a new quantity k_{ij} which only depends on $\mathcal{C}_{ij}$ parameters

$$k_{ij}=\frac{x_{ij}^2+y_{ij}^2-R_{ij}^2}{2}. \tag{6.14}$$

In our triangulation problem, we have to intersect the three power lines, that is to solve this linear system

$$\begin{cases} x\,(x_{12}-x_{23})+y\,(y_{12}-y_{23})=k_{12}-k_{23}\\ x\,(x_{23}-x_{31})+y\,(y_{23}-y_{31})=k_{23}-k_{31}\\ x\,(x_{31}-x_{12})+y\,(y_{31}-y_{12})=k_{31}-k_{12}\end{cases} \tag{6.15}$$

As can be seen, any of these equations may be obtained by adding the two others, which is a way to prove that the three power lines concur in a unique point: the power center. The coordinates of the power center, that is the robot position is given by

$$x_R = \frac{\begin{vmatrix} k_{12}-k_{23} & y_{12}-y_{23}\\ k_{23}-k_{31} & y_{23}-y_{31}\end{vmatrix}}{D_\triangle}, \tag{6.16}$$

$$y_R = \frac{\begin{vmatrix} x_{12}-x_{23} & k_{12}-k_{23}\\ x_{23}-x_{31} & k_{23}-k_{31}\end{vmatrix}}{D_\triangle}. \tag{6.17}$$

The denominator $D_\triangle$, common to x_R and y_R, is equal to

$$D_\triangle=\begin{vmatrix} x_{12}-x_{23} & y_{12}-y_{23}\\ x_{23}-x_{31} & y_{23}-y_{31}\end{vmatrix}=\begin{vmatrix} x_{12} & y_{12} & 1\\ x_{23} & y_{23} & 1\\ x_{31} & y_{31} & 1\end{vmatrix}, \tag{6.18}$$

which is the signed area of the triangle defined by the circle centers, multiplied by two. This result shows that the power center exists, if the circle centers are not collinear, that is if $D_\triangle \neq 0$. The special case ($D_\triangle = 0$) is discussed later.

6.3.3 First (naive) version of the algorithm

A first, but naive version of our algorithm consists in applying the previous equations to get the robot position. This method is correct, but it is possible to further simplify the equations. First, note that the squared radii R_{ij}^2 only appear in the parameters k_{ij}. If we replace the

expression of R_{ij}^2 (equation (6.11)) and the expressions of x_{ij} and y_{ij} (equations (6.9) and (6.10)) into the expression of k_{ij} (equation (6.14)), we find, after many simplifications that (see appendix B.2 for details)

$$k_{ij} = \frac{x_i x_j + y_i y_j + T_{ij}(x_j y_i - x_i y_j)}{2}, \tag{6.19}$$

which is much simpler than equations (6.11) and (6.14) (no squared terms anymore). In addition, the 1/2 factor involved in the circle centers coordinates (equations (6.9) and (6.10)) as well as in the parameters k_{ij} (equation (6.14)) cancels in the robot position coordinates (equations (6.16) and (6.17)). This factor can thus be omitted. For now, we use these modified circle center coordinates $\{x'_{ij},\, y'_{ij}\}$

$$x'_{ij} = (x_i + x_j) + T_{ij}\,(y_i - y_j)\,, \tag{6.20}$$
$$y'_{ij} = (y_i + y_j) - T_{ij}\,(x_i - x_j)\,, \tag{6.21}$$

and modified parameters k'_{ij}

$$k'_{ij} = x_i x_j + y_i y_j + T_{ij}(x_j y_i - x_i y_j). \tag{6.22}$$

6.3.4 Final version of the algorithm

The most important simplification consists in translating the world coordinate frame into one of the beacons, that is solving the problem relatively to one beacon and then add the beacon coordinates to the computed robot position (like Font-Llagunes and Batlle [34], without the rotation of the frame). In the following, we arbitrarily choose B_2 as the origin ($B'_2 = \{0,\, 0\}$). The other beacon coordinates become: $B'_1 = \{x_1 - x_2,\, y_1 - y_2\} = \{x'_1,\, y'_1\}$ and $B'_3 = \{x_3 - x_2,\, y_3 - y_2\} = \{x'_3,\, y'_3\}$. Since $x'_2 = 0$ and $y'_2 = 0$, we have $k'_{12} = 0$, $k'_{23} = 0$. Also, we can compute the value of one cot(.) by referring to the two other cot(.) because the three angles are linked (equation (6.13))

$$T_{31} = \frac{1 - T_{12}T_{23}}{T_{12} + T_{23}}. \tag{6.23}$$

The final algorithm is given in Algorithm 6.1.

6.3.5 Discussion

The ToTal algorithm is very simple: computations are limited to basic arithmetic operations and only two cot(.). Furthermore, the number of conditional statements is reduced, which increases its readability and eases its implementation. Among them, we have to take care of the cot(.) infinite values and the division by D, if equal to zero. If a bearing angle ϕ_{ij} between two beacons is equal to 0 or π, that is the robot stands on the B_iB_j line, then $\cot(\phi_{ij})$ is infinite. The corresponding circle degenerates to the B_iB_j line (infinite radius and center coordinates). The robot is then located at the intersection of the remaining power line and the B_iB_j line; it can be shown that the mathematical limit $\lim_{T_{ij}\to\pm\infty}\{x_R,\, y_R\}$ exists and corresponds to this situation.

Like for other algorithms, our algorithm also has to deal with these special cases, but the way to handle them is simple. In practice, we have to avoid *Inf* or *NaN* values in the floating point computations. We propose two ways to manage this situation. The first way consists in limiting the cot(.) value to a minimum or maximum value, corresponding to a small angle

Algorithm 6.1 Final version of the ToTal algorithm.

Given the three beacon positions $\{x_1, y_1\}$, $\{x_2, y_2\}$, $\{x_3, y_3\}$, and the three angles ϕ_1, ϕ_2, ϕ_3:

1. compute the modified beacon coordinates:

$$\begin{aligned} x'_1 &= x_1 - x_2, \\ y'_1 &= y_1 - y_2, \\ x'_3 &= x_3 - x_2, \\ y'_3 &= y_3 - y_2, \end{aligned}$$

2. compute the three cot(.):

$$\begin{aligned} T_{12} &= \cot(\phi_2 - \phi_1), \\ T_{23} &= \cot(\phi_3 - \phi_2), \\ T_{31} &= \frac{1 - T_{12}T_{23}}{T_{12} + T_{23}}, \end{aligned}$$

3. compute the modified circle center coordinates:

$$\begin{aligned} x'_{12} &= x'_1 + T_{12}\, y'_1, \\ y'_{12} &= y'_1 - T_{12}\, x'_1, \\ x'_{23} &= x'_3 - T_{23}\, y'_3, \\ y'_{23} &= y'_3 + T_{23}\, x'_3, \\ x'_{31} &= (x'_3 + x'_1) + T_{31}\, (y'_3 - y'_1), \\ y'_{31} &= (y'_3 + y'_1) - T_{31}\, (x'_3 - x'_1), \end{aligned}$$

4. compute k'_{31}:

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

5. compute D (if $D = 0$, return with an error):

$$D = (x'_{12} - x'_{23})(y'_{23} - y'_{31}) - (y'_{12} - y'_{23})(x'_{23} - x'_{31}),$$

6. compute the robot position $\{x_R, y_R\}$ and return:

$$\begin{aligned} x_R &= x_2 + \frac{k'_{31}(y'_{12} - y'_{23})}{D}, \\ y_R &= y_2 + \frac{k'_{31}(x'_{23} - x'_{12})}{D}. \end{aligned}$$

that is far below the measurement precision. For instance, we limit the value of the cot(.) to $\pm 10^8$, which corresponds to an angle of about $\pm 10^{-8}\,rad$; this is indeed far below the existing angular sensor precisions. With this approximation of the mathematical limit, the algorithm remains unchanged. The second way consists in adapting the algorithm when one bearing angle is equal to 0 or π. This special case is detailed in Algorithm 6.2, in which the indexes $\{i, j, k\}$ have to be replaced by $\{1, 2, 3\}$, $\{3, 1, 2\}$, or $\{2, 3, 1\}$ if $\phi_{31} = 0$, $\phi_{23} = 0$, or $\phi_{12} = 0$ respectively.

The denominator D is equal to 0 when the circle centers are collinear or coincide. For non collinear beacons, this situation occurs when the beacons and the robot are concyclic; they all stand on the same circumference, termed the *critical circumference* by Font-Llagunes and Batlle [34]. In that case, the three circles are equal as well as their centers, which causes $D = 0$ (the area defined by the three circle centers is equal to zero). For collinear beacons, this situation is encountered when the beacons and the robot all stand on this line. For these cases, it is impossible to compute the robot position. This is a *restriction common to all three object triangulation*, regardless of the used algorithm [30, 34, 59].

The value of D, computed in the final algorithm, is the signed area delimited by the circle centers, multiplied by height[4]. $|D|$ decreases to 0 when the robot approaches the critical circumference (almost collinear circle center, almost parallel power lines). Therefore, it is quiet natural to use $|D|$ as a reliability measure of the computed position. In the next section, we show that $1/|D|$ is a good approximation of the position error. In practice, this measure can be used as a validation gate after the triangulation algorithm, or when a data fusion algorithm is used. Finally, it should be noted that the robot orientation θ_R may be determined by using any beacon B_i and its corresponding angle ϕ_i, once the robot position is known:

$$\theta_R = \text{atan2}(y_i - y_R,\, x_i - x_R) - \phi_i, \tag{6.24}$$

where atan2(y, x) denotes the C-like two arguments function, defined as the principal value of the argument of the complex number $(x + iy)$.

6.4 Simulations

6.4.1 Error Analysis

The problem of triangulation given three angle measurements is an exact calculus of the robot pose, even if these angles are affected by noise. This contrasts with multiangulation, which is an overdetermined problem, even with perfect angle measurements. It turns out that the sensitivity of triangulation with respect to the input angles is unique and does not depend on the way the problem is solved, neither on the algorithm. In this chapter, we focus on the presentation of a new three object triangulation algorithm and the comparison with seventeen other algorithms of the same family. So, we do not elaborate on the error analysis for triangulation, as it has been studied in many papers; the same conclusions, as found in [28, 32, 34, 45, 55, 76], also yield for our algorithm. However, in order to validate our algorithm and to discuss the main characteristics of triangulation sensitivity, we have performed some simulations. The simulation setup comprises a square shaped grid area ($4 \times 4\,m^2$), with three non collinear beacons forming a regular triangle ($B_1 = \{0\,m, 1\,m\}$,

[4]Note that the quantity D computed in the final algorithm is different from the quantity $D_\triangle$ defined in Section 6.3.2, since the centers coordinates have been multiplied by two.

Algorithm 6.2 Special case $\phi_{ki} = 0 \vee \phi_{ki} = \pi$.

Given the three beacon positions $\{x_i, y_i\}$, $\{x_j, y_j\}$, $\{x_k, y_k\}$, and the three angles ϕ_i, ϕ_j, ϕ_k:

1. compute the modified beacon coordinates:

$$\begin{aligned} x'_i &= x_i - x_j, \\ y'_i &= y_i - y_j, \\ x'_k &= x_k - x_j, \\ y'_k &= y_k - y_j, \end{aligned}$$

2. compute $T_{ij} = \cot(\phi_j - \phi_i)$,

3. compute the modified circle center coordinates:

$$\begin{aligned} x'_{ij} &= x'_i + T_{ij}\, y'_i, \\ y'_{ij} &= y'_i - T_{ij}\, x'_i, \\ x'_{jk} &= x'_k + T_{ij}\, y'_k, \\ y'_{jk} &= y'_k - T_{ij}\, x'_k, \\ x'_{ki} &= (y'_k - y'_i), \\ y'_{ki} &= (x'_i - x'_k), \end{aligned}$$

4. compute $k'_{ki} = (x'_i y'_k - x'_k y'_i)$,

5. compute D (if $D = 0$, return with an error):

$$D = (x'_{jk} - x'_{ij})(y'_{ki}) + (y'_{ij} - y'_{jk})(x'_{ki}),$$

6. compute the robot position $\{x_R, y_R\}$ and return:

$$\begin{aligned} x_R &= x_j + \frac{k'_{ki}(y'_{ij} - y'_{jk})}{D}, \\ y_R &= y_j + \frac{k'_{ki}(x'_{jk} - x'_{ij})}{D}. \end{aligned}$$

$B_2 = \{-0.866\,m,\ -0.5\,m\}$, and $B_3 = \{0.866\,m,\ -0.5\,m\}$). The distance step in the grid is $2\,cm$ in each direction. For each point in this grid, we compute the exact angles ϕ_i seen by the robot (the robot orientation is arbitrarily set to $0\,deg$). Then we add Gaussian noise to these angles, with zero mean, and with three different standard deviations ($\sigma = 0.01\,deg$, $\sigma = 0.1\,deg$, and $\sigma = 1\,deg$). The noisy angles are then used as inputs of our algorithm to compute the estimated position. The position error Δd_R is the Euclidean distance between the exact and estimated positions:

$$\Delta d_R = \sqrt{(x_{true} - x_R)^2 + (y_{true} - y_R)^2}, \tag{6.25}$$

and the orientation error $\Delta\theta_R$ is the difference between the exact and estimated orientations:

$$\Delta\theta_R = \theta_{true} - \theta_R. \tag{6.26}$$

The experiment is repeated 1000 times for each position to compute the standard deviation of the position error $\sqrt{var\{\Delta d_R\}}$ and of the orientation error $\sqrt{var\{\Delta\theta_R\}}$. The standard deviations of the position and orientation errors are drawn in Figures 6.5 and 6.6. The beacon locations are represented by black and white dot patterns. The first, second, and third column provide the result for $\sigma = 0.01\,deg$, $\sigma = 0.1\,deg$, and $\sigma = 1\,deg$ respectively. The first, second, and third rows show the standard deviation of the position error, the standard deviation of the orientation error, and the mean error measure $1/|D|$ respectively. In Figure 6.5, the graphics are plotted by using a logarithmic scale, and by removing 1 % of the largest values (because the errors are unbounded as we get closer to the critical circumference). In Figure 6.6, the graphics are plotted by using an histogram equalization in order to enhance the visual representation, and to point out the similarities between the position and orientation error, and our new error measure.

Our simulation results are consistent with common three object triangulation algorithms. In particular, we can easily spot the critical circumference where errors are large, the error being minimum at the center of this circumference. Also, one can see that, outside the critical circumference, the error increases with the distance to the beacons. It is also interesting to note that $1/|D|$ has a similar shape than the position or orientation errors. It can be proven (starting from equations (6.16) and (6.17), by a detailed sensitivity analysis of the robot position error with respect to angles, that (see appendix B.3 for the details)

$$\Delta d_R \simeq \frac{1}{|D|}\,|\Delta\phi|\,f(.), \tag{6.27}$$

where $\Delta\phi$ is the angle error (assumed to be the same for the three angles), and $f(.)$ is some function of all the other parameters. This confirms our claim that $1/|D|$ can be used as an approximation of the position error[5]. Furthermore, one can observe from the graphic scales, that the position or orientation errors almost evolve linearly with angle errors, when they are small (look at the scale of the different graphics).

[5]Note that this is a strong analogy with the Geometric Dilution of Precision (GDOP) concept, used in GPS error analysis. It is shown that the GDOP, based on four satellites, is related to the volume of the tetrahedron defined by these four satellites. In our case, the error is related to the area of the triangle defined by the three circle centers.

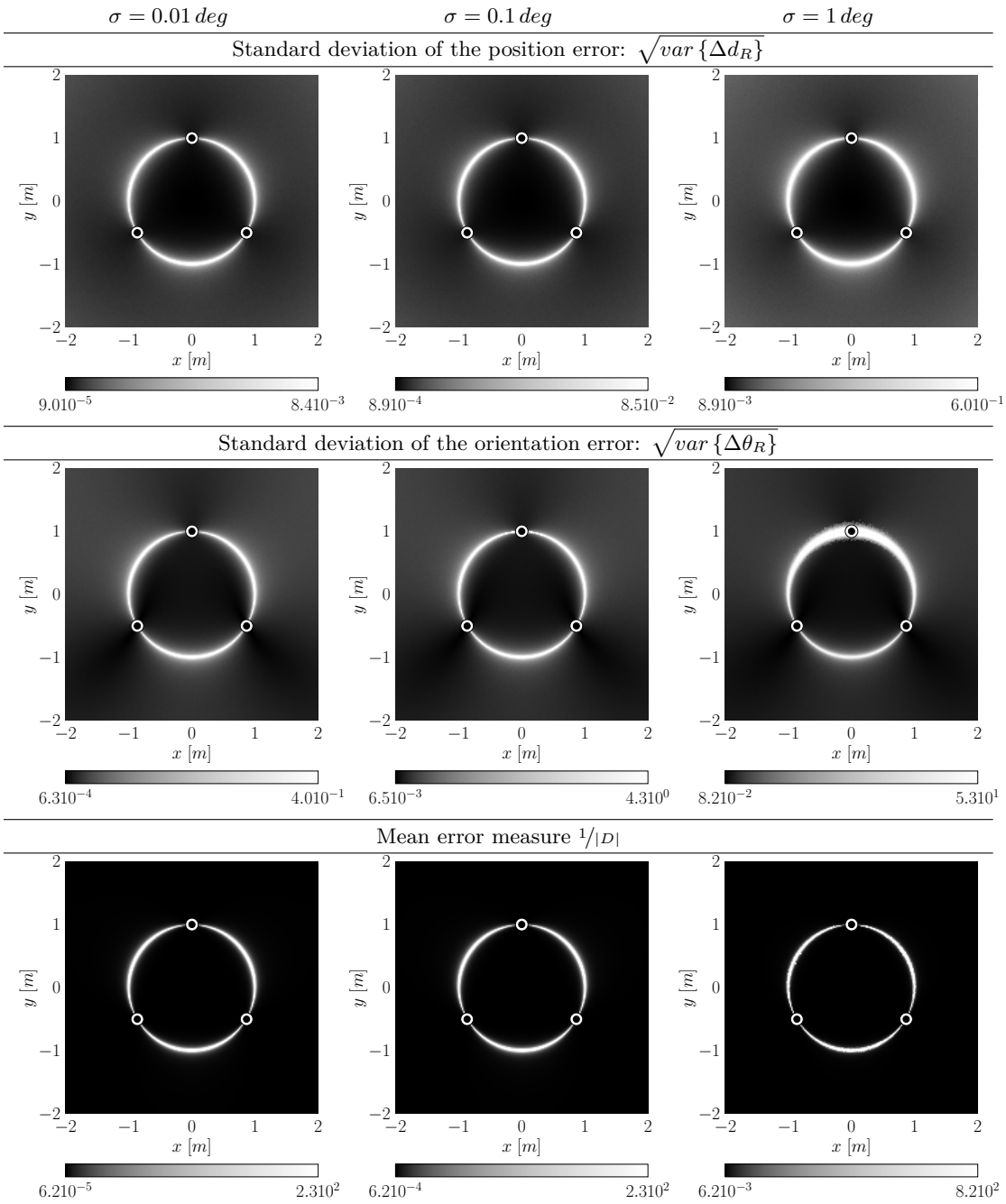

Figure 6.5: Simulation results giving the position and orientation errors for noisy angle measurements. The beacon positions are represented by black and white dot patterns. The first, second, and third columns provide the results for $\sigma = 0.01\,deg$, $\sigma = 0.1\,deg$, and $\sigma = 1\,deg$ respectively. Position errors are expressed in meters, the orientation error is expressed in degrees, and the error measure $1/|D|$ is in $1/m^2$. The graphics are displayed by using a logarithmic scale, and by removing 1 % of the largest values (because the errors are unbounded as we get closer to the critical circumference).

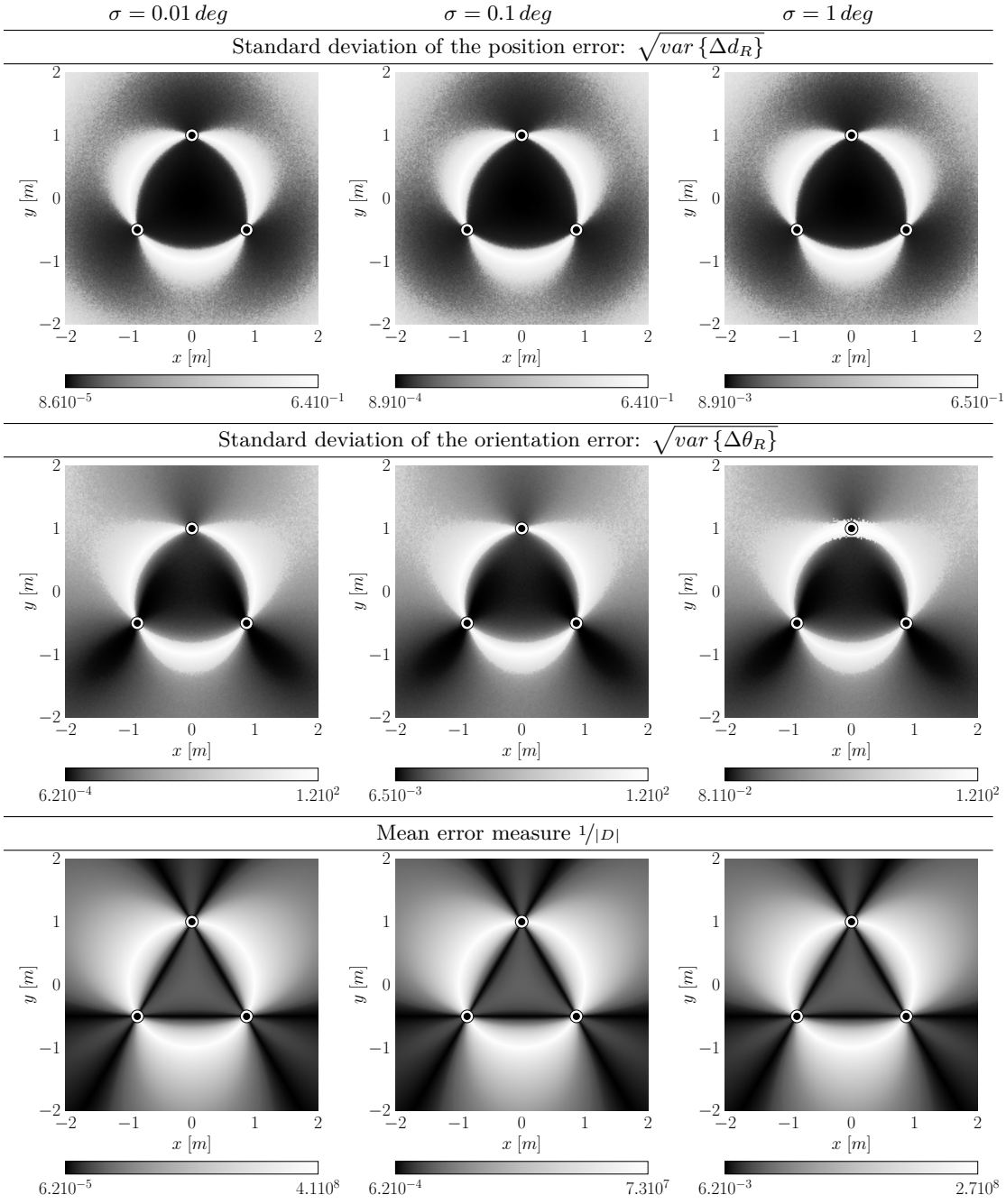

Figure 6.6: Simulation results giving the position and orientation errors for noisy angle measurements. The beacon positions are represented by black and white dot patterns. The first, second, and third columns provide the results for $\sigma = 0.01\,deg$, $\sigma = 0.1\,deg$, and $\sigma = 1\,deg$ respectively. Position errors are expressed in meters, the orientation error is expressed in degrees, and the error measure $1/|D|$ is in $1/m^2$. The graphics are displayed by using an histogram equalization to enhance its visual representation and interpretation.

	Algorithm	+	×	/	$\sqrt{x}$	*trigo*	time (s) †
	ToTal [1]	30	17	2	0	2	0.1633
[52]	Ligas [1]	29	22	2	0	2	0.1707
[34]	Font-Llagunes [1]	23	17	2	0	5	0.2278
[20]	Cassini [2]	19	8	3	0	4	0.2491
[21]	Cohen [1]	37	15	3	2	4	0.2718
[28]	Easton [2]	22	24	1	0	5	0.2976
[59]	McGillem [1]	37	18	5	2	8	0.3403
[42]	Hmam [2]	29	11	3	3	9	0.4277
[21]	Cohen [2]	26	11	3	2	11	0.4374
[32]	Esteves [2]	43	14	2	2	11	0.4705
[20]	Collins [2]	34	10	2	2	11	0.4852
[59]	McGillem [2]	29	9	3	2	11	0.5013
[20]	Kaestner-Burkhardt [2]	28	10	3	2	11	0.5044
[84]	Tsukiyama [1]	52	22	3	5	14	0.5963
[88]	Zalama [1]	52	21	4	5	14	0.6092
[68]	Tienstra [2]	33	18	8	3	9	0.6396
[35]	Font-Llagunes [1]	62	25	6	1	8	0.6475
[55]	Madsen [2]	38	24	5	3	15	0.7072

† For 10^6 executions on an Intel(R) Core(TM) i7 920 @ 2.67GHz.

[1] Geometric circle intersection

[2] Trigonometric solution

Table 6.1: Comparison of various triangulation algorithms to our ToTal algorithm.

6.4.2 Benchmarks

We have also compared the execution time of our algorithm to seventeen other three object triangulation algorithms similar to ours (*i.e.* which work in the whole plane and for any beacon ordering). These algorithms have been introduced in Section 6.2, and have been implemented after the author's guidelines[6]. Each algorithm has been running 10^6 times at random locations of the same square shaped area as that used for the error analysis. The last column of Table 6.1 provides the running times on an *Intel(R) Core(TM) i7 920 @ 2.67GHz* (6GB RAM, Ubuntu 11.04, GCC 4.5.2). We used the C `clock_gettime` function to measure the execution times, in order to yield reliable results under timesharing. It appears that our algorithm is the fastest of all (about 30 % faster than the last best known algorithm of Font-Llagunes and Batlle [34], and 5 % faster than the recent algorithm of Ligas [52]). In addition to the computation times, we have also reported the number of basic arithmetic computations, squared roots, and trigonometric functions used for each algorithm. This may help to choose an algorithm for a particular hardware architecture, which may have a different behavior for basic arithmetic computations, or complex functions such as square roots or trigonometric functions. One can see that our algorithm has the minimum number of trigonometric functions, which is clearly related to the times on a classical computer architecture (see Table 6.1). A fast algorithm

[6]The C source code used for the error analysis and benchmarks is available at `http://www.ulg.ac.be/telecom/triangulation`. The C source codes of all algorithms, including ToTal, are also provided.

is an advantage for error simulations, beacon placement, and beacon position optimization algorithms (see the next chapter). Note that the algorithm of Ligas, as for ToTal, also uses the minimum number of trigonometric functions (two cot(.) computations), explaining why both algorithms are basically similar in terms of efficiency. However, the algorithm of Ligas does not provide a reliability measure, contrarily to our algorithm ToTal.

6.5 Conclusions

Most of the many triangulation algorithms proposed so far have major limitations. In this chapter, we present a new three object triangulation algorithm based on the elegant notion of power center of three circles. Our new triangulation algorithm, named ToTal, natively works in the whole plane (except when the beacons and the robot are concyclic or collinear), and for any beacon ordering. Furthermore, it only uses basic arithmetic computations and two cot(.) computations. Comprehensive benchmarks show that our algorithm is faster than comparable algorithms, and simpler in terms of the number of operations. We also compare the number of basic arithmetic computations, squared roots, and trigonometric functions used for seventeen known triangulation algorithms.

In addition, we propose a unique reliability measure of the triangulation result in the whole plane, and we establish by simulations that $1/|D|$ is a natural and adequate criterion to estimate the error of the positioning. To our knowledge, none of the algorithms of the same family does provide such a measure. This error measure can be used to identify the pathological cases (critical circumference), or as a validation gate in data fusion algorithms based on triangulation.

For all these reasons, ToTal is a fast, flexible, and reliable three object triangulation algorithm. Such an algorithm is an excellent choice for many triangulation issues related to the performance or optimization, such as error simulations, beacon placement or beacon position optimization algorithms. A fast and inexpensive algorithm is also an asset to initialize a more complex positioning algorithm, that internally relies on a Kalman filter for instance.

Chapter 7

System calibration

7.1 Introduction

In this manuscript, we have described our new angle measurement sensor. Then we have elaborated a mathematical model for the variance of the measured angles, and we have evaluated this model through simulations and experiments. Finally, in the previous chapter, we have described our new three object triangulation algorithm. The combination of these two parts, the hardware and the positioning algorithm, are the necessary first steps for building a complete positioning system. One could think that combining these two parts is sufficient to achieve good results. However, even with the most carefully designed hardware and the most accurate positioning algorithm, experiments show that it is not sufficient to achieve the most accurate results. Until now, we have focused mainly on the variance. However, many sources of biases (also denoted errors in this chapter) are unavoidable in a real world situation and a good positioning system should deal with them.

The angles measured by our system BeAMS are corrupted by different sources of errors. We can identify four sources of errors resulting in inaccuracies in the computed positions. The first one is due to a non constant bias in the angle measurements with respect to the received power; in other words, it means that the angle measurements are sensitive to the distance. The second source of errors is caused by the sensor mechanics and optics, leading to a non constant bias affecting the measurements during a complete revolution of the sensor. The third source of errors is caused by an inaccurate knowledge of the beacon positions in the environment, because their coordinates are measured manually. The fourth source of errors is caused by an inaccurate knowledge of the real rotation center of the turret, from which the angles are measured.

In this chapter, we focus on these four situations and explain how to deal with them. The final goal consists in providing a complete calibration procedure in order to improve the accuracy of the computed positions and orientations.

This chapter is organized as follows. Section 7.2 presents the calibration methods found in the literature. The different sources of errors are detailed in Section 7.3.2, Section 7.3.3, and Section 7.3.4. In Section 7.3.5, we detail our complete calibration procedure, and we discuss it in Section 7.3.6. Finally, we conclude the chapter in Section 7.4.

7.2 Related work

In Chapter 2 and Chapter 6, we have shown that there are many works discussing mobile robot positioning. In particular, there are several papers about new angle measurement systems, positioning algorithms including triangulation and multiangulation, error models and error sensitivity analysis, optimal beacon placement and beacon selection algorithms. Surprisingly, there are almost no papers discussing the calibration of such systems.

To our knowledge, there is only one paper discussing the calibration of a complete positioning system similar to ours. This work has been carried out by Loevsky and Shimshoni [53], and we present it in details hereafter. In their experiments, they use a Denning MRV4 mobile robot equipped with the *LaserNav* commercial angle measurement system, which has been presented in Section 2.2.1. It uses a rotating laser, combined with passive reflective beacons. These reflective beacons are bar-coded in order to allow their identification. Moreover they use more than three beacons and, as a consequence, they use a multiangulation algorithm. This algorithm has been designed by Shimshoni, and presented in a previous paper [75]. As explained in [53], a calibration step is necessary to achieve accurate results in a complex measuring system. In their paper, they identify four possible causes leading to inaccurate computed positions. The first one is due to the hardware mechanics, causing biased measurements. The second one is due to inaccurate manual measurements of the beacon coordinates in the reference frame, affecting the multiangulation algorithm directly. The third reason is due to the robot motion itself, combined with the rotation of the sensor. The last reason is caused by misidentified beacons, causing large errors in the computed position, even if their corresponding angles are accurate. These four sources of errors are detailed hereafter.

The first source of errors can be observed when the robot rotates around its own center at a fixed location. It is not said in the paper, but we suppose that the rotation center of the robot corresponds to the one of the sensor. As explained in Section 5.7 and Section 6.3, the position computed by any angle-based positioning algorithm depends on the angle differences, and not on the absolute angles. As a result, a rotation of the sensor should not influence the position, only the orientation. But, with a real sensor, one can observe that the computed positions spread over a large area of the 2D plane, instead of being normally distributed around the rotation center, with a small variance and bias. This suggests that the angles are affected by a bias depending on the sensor orientation. The authors propose to correct these biased angles with a periodic correction function depending on four parameters. The parameters have been tuned by using the Nelder-Mead simplex method[1], in order to minimize an error measure (detailed later). Their method seems to achieve good results as the dispersion of the positions is reduced from about $10\,cm$ to $1\,cm^2$. This method is called "hardware calibration", and is applied to BeAMS in Section 7.3.3.

The second source of errors is caused by inaccurate beacons coordinates, due to manual measurements performed by the user. As these coordinates are direct inputs of the multiangulation algorithm, it directly affects the computed positions, even in presence of noise free angles. Again, Loevsky and Shimshoni [53] propose a method based on the Nelder-Mead simplex, in order to find the optimal set of beacons coordinates that minimizes an error measure.

The third source of errors is due to the robot motion as explained hereafter. The angle

[1] The Nelder-Mead simplex method is an unconstrained nonlinear minimization method.

[2] The absolute values are not important, as they depend on the particular calibration setup. What is important is the reduction ratio $(10:1)$ achieved by their method.

measurements are based on the rotation of a mirror deflecting the outgoing laser beam and incoming reflected beam. But, because of the robot motion, the absolute rotation speed of the mirror (with respect to the reference frame) may change, causing biased measurements (because the angles are deduced from the position of the mirror, which is related to the rotation speed). As the robot motion is known and can be decomposed in a translation and a rotation, the authors propose two methods to compensate these effects. The first one uses the Nelder-Mead simplex, and the second one uses Brent's method[3]. It appears that the correction for rotation is mandatory to achieve a good accuracy during fast rotations, whereas the correction for translation is negligible compared to the increase in complexity.

The last source of errors comes from the misidentification of beacons. Like for the beacons coordinates, it directly affects the computed position, resulting in large errors. Due to the different nature of this noise source, it has to be addressed differently. This problem can be solved in three steps: 1) detect the presence of a misidentified beacon, 2) remove this beacon from the set, and 3) recompute the position with the new set. The authors propose to use the RANSAC algorithm[4], which is suited for this kind of problem.

For each source of errors, the authors propose a dedicated correction method. The two first correction methods are named "calibration of the localization system components" since they are the first necessary steps to achieve a good accuracy in static conditions. The authors propose a "combined calibration procedure" to find the optimal parameters associated to each method (the correction function and the beacons coordinates). This "combined calibration procedure" is detailed further in Section 7.3.5.

Finally, it should be noted that the methods presented by Loevsky and Shimshoni do not require ground truth references (*i.e.* fixed positions and orientations) to work, which is an advantage to avoid time-consuming placements of the robot at given positions. Instead, the minimization methods of each step are based on a mean square error calculation, which is derived from their multiangulation algorithm. Indeed, the computed position based on angle measurements is an exact calculus for three angles [21, 32, 67]. Therefore, methods based on more than three angles are overdetermined problems, and they compute the position according to a particular minimization criterion [13,75]. The computed position is then associated to an error measure, which serves as a basis for the methods presented by Loevsky and Shimshoni. Unfortunately, it is important to note that these methods cannot be applied to systems using only three beacons, as pointed by the authors. In our case, we need to use ground truth references, as explained in the next section.

7.3 The calibration method

Like for any angle-based positioning system, our system BeAMS is subject to the errors described by Loevsky and Shimshoni. So, in order to improve the localization results, we need to calibrate our system. But, despite that BeAMS can measure angles for any number of beacons, it has initially been designed for the Eurobot contest, which limits the number of beacons to three. Since an angle based algorithm requires a minimum of three beacons to compute the position unequivocally (except on the circumference defined by the three beacons), the number of beacons has been set to three. As a consequence, the calibration

[3]Brent's method is a root-finding algorithm.

[4]RANSAC (RANdom SAmple Consensus) is a model parameter estimation algorithm, known to be robust to outliers contained in the data.

	X^B	Y^B
B_1	1.151	0.049
B_2	0.600	0.907
B_3	0.049	0.049

Table 7.1: Coordinates of the beacons, expressed in (m). These coordinates are the supposed prior beacon positions. The expected accuracy of the measurements is about $1\,mm$.

method proposed by Loevsky and Shimshoni cannot be applied directly, as explained earlier. Moreover, our hardware system is different from the one they use and, in a consequence, the proposed solutions need to be adapted.

To our knowledge, no method has been proposed so far to calibrate a system using three beacons. The most important difference of our method compared to that of Loevsky and Shimshoni lies in the minimization criterion. As explained earlier, we cannot base our method on a mean square error calculation derived from a multiangulation algorithm, because triangulation is an exact calculus. As a consequence, we have no choice but to use some ground truth references. It means that we have to place the robot/sensor at known locations, in order to create our own error criterion.

7.3.1 The calibration setup

As explained previously, we need to use some ground truth references, in order to produce an error measure. Since the sensor measures angles that are affected by errors, we could use absolute angles as the ground truths, in order to find the adequate correction functions. However, as explained in Section 5.7, in which we have evaluated the bias, it is difficult to measure absolute angles accurately, compared to the ease of measuring the variance. Moreover, the beacons coordinates are expressed in terms of distances, as well as the result of the triangulation algorithm, which is the final goal of a positioning system. Also, it is easier to place the robot at known positions, as well as measuring distances.

So, in order to provide accurate ground truth positions, and to validate our new calibration procedure, we have designed a test setup, as represented in Figure 7.1. The reference frame is represented by the two orthogonal axis denoted $(x,\, y)$. It is composed of a grid of 15×12 squares with a side length of $8\,cm$, which defines an area of $120\,cm$ by $96\,cm$. These dimensions are about the third of the ones used in the Eurobot contest, in order to ease the manipulations. Each square of the grid has a side of $8\,cm$, which corresponds exactly to the size of the structure of BeAMS. This eases the placement of the sensor on the grid. A picture of the calibration setup is displayed in Figure 7.2.

The set of beacons is denoted $\mathcal{B}$, and contains three elements

$$\mathcal{B} = \{B_1,\, B_2,\, B_3\}\,, \tag{7.1}$$

where B_i denotes the coordinates of the beacon i

$$B_i = \left(X_i^B,\, Y_i^B\right). \tag{7.2}$$

The three beacons (represented by encircled black points in Figure 7.1) have been placed at coordinates given in Table 7.1, at the borders of the grid. The positions of the beacons (*i.e.* the IR LEDs) have been measured manually, and the expected accuracy is about $1\,mm$. Note

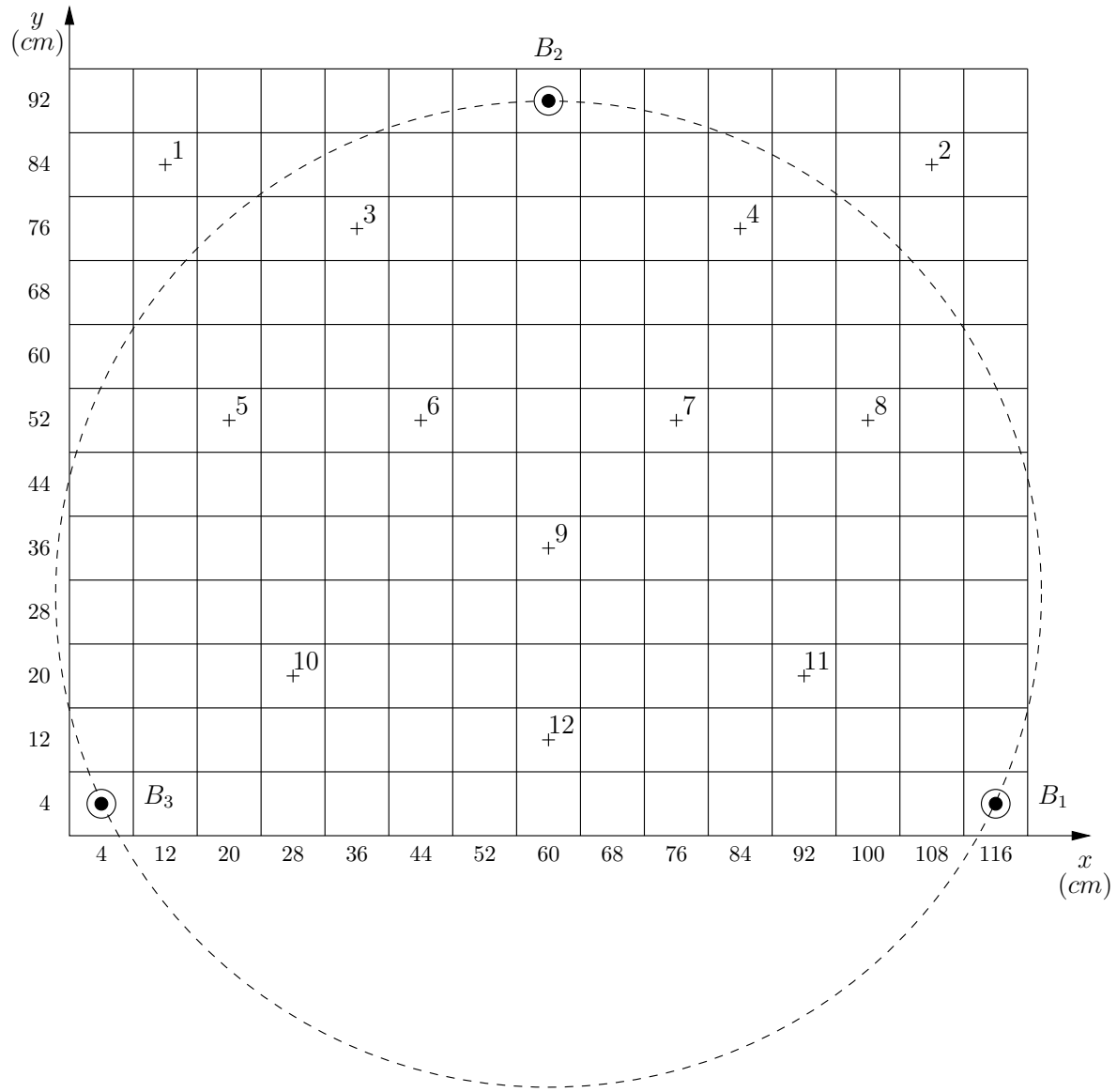

Figure 7.1: Calibration setup used for the experiments. The encircled black points are the beacons. The crosses are the reference positions. The dashed circle is the critical circumference. The grid measures 120 cm by 96 cm, and is composed of 15×12 squares with a side length of 8 cm.

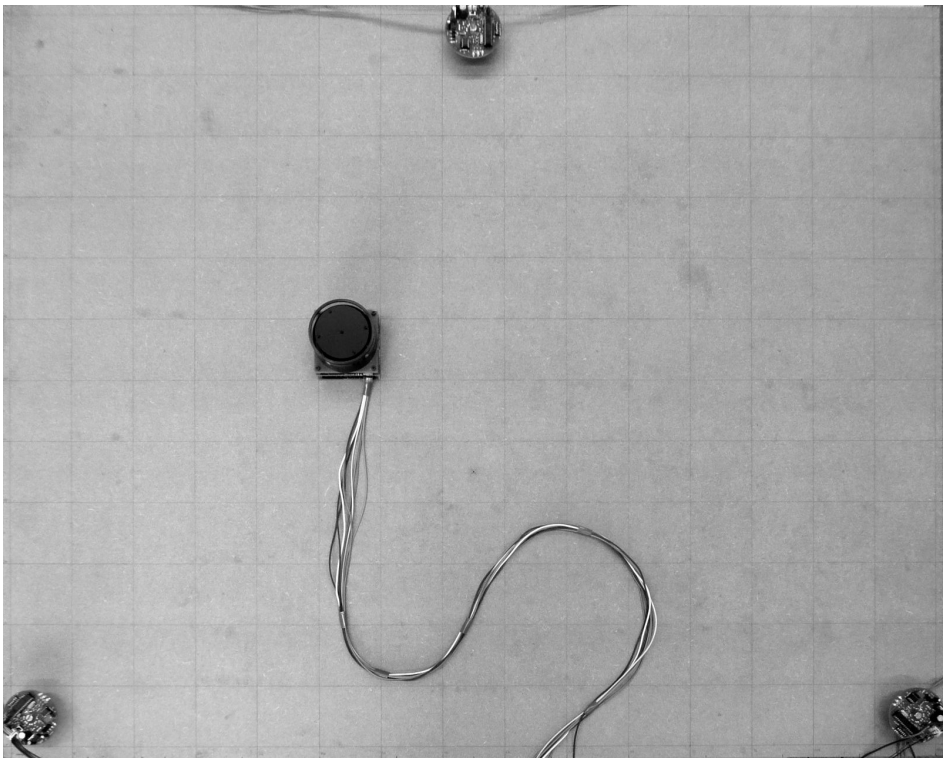

Figure 7.2: Picture of the calibration setup.

that we have tried to estimate the position of the emitting part of the LED, inside its $5\,mm$ body. Then, we have chosen 12 reference positions P_i (represented by crosses in Figure 7.1), whose coordinates are given in Table 7.2. The reference positions are well distributed over the grid. The expected accuracy of the grid and the placement of the sensor at the reference positions is about $2\,mm$. Also, the orientation of the sensor/robot is important, and it has to be calibrated as well. So, we have to define some reference orientations for each reference position. Due to the shape of the grid and the square structure of the sensor, it is easy to align the sensor with the grid axis. As a consequence, we have chosen four reference orientations O_i as follows

$$O_i \in \left\{0,\, \frac{\pi}{2},\, \pi,\, -\frac{\pi}{2}\right\}. \tag{7.3}$$

So, the combination of a particular reference position and orientation defines a set of $n = 48 = 12 \times 4$ reference configurations (poses) of the sensor, denoted $\mathcal{C}$

$$\mathcal{C} = \left\{\overrightarrow{C_1},\, \overrightarrow{C_2}, \ldots,\, \overrightarrow{C_n}\right\}, \tag{7.4}$$

where each element $\overrightarrow{C_k}$ is defined as

$$\overrightarrow{C_k} = \left(X_k^P,\, Y_k^P,\, O_k\right), \tag{7.5}$$

where $\left(X_k^P,\, Y_k^P\right)$ is one of the 12 reference positions, and O_k is one of the 4 reference orientations.

	X^P	Y^P
P_1	0.12	0.84
P_2	1.08	0.84
P_3	0.36	0.76
P_4	0.84	0.76
P_5	0.20	0.52
P_6	0.44	0.52
P_7	0.76	0.52
P_8	1.00	0.52
P_9	0.60	0.36
P_{10}	0.28	0.20
P_{11}	0.92	0.20
P_{12}	0.60	0.12

Table 7.2: Coordinates of the reference positions, expressed in (m).

The first step of the calibration procedure consists in recording the angles to the three beacons, for each configuration. The set of angle measurements, denoted $\mathcal{A}$, contains n elements

$$\mathcal{A} = \left\{\overrightarrow{\phi_1}, \overrightarrow{\phi_2}, \ldots, \overrightarrow{\phi_n}\right\}, \tag{7.6}$$

where each element is a triplet of the measured angles[5] to the beacons

$$\overrightarrow{\phi_k} = \left(\phi_k^1, \phi_k^2, \phi_k^3\right). \tag{7.7}$$

From these angle measurements and the manual measurements of the beacons coordinates, the estimated positions are computed as follows

$$\left(\widehat{X_k}, \widehat{Y_k}\right) = \mathcal{T}(\overrightarrow{\phi_k}, \mathcal{B}), \tag{7.8}$$

where $\mathcal{T}(\overrightarrow{\phi_k}, \mathcal{B})$ denotes a three object triangulation algorithm, which computes the robot position from three angles and the beacon positions. In our experiments, we use our algorithm ToTal. Also, one can compute the sensor/robot orientation by using any beacon i and its corresponding angle, once the robot position is known (see Section 6.3.5)

$$\widehat{O_k} = \text{atan2}(Y_i^B - \widehat{Y_k}, X_i^B - \widehat{X_k}) - \phi_k^i. \tag{7.9}$$

In our experiments, we use the beacon $i = 1$, but this choice is arbitrary (we discuss this choice later). Now that the calibration setup has been presented, we define the different error measures used later in the chapter. The position RMS error is defined as

$$perr_{RMS} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left[\left(X_k^P - \widehat{X_k}\right)^2 + \left(Y_k^P - \widehat{Y_k}\right)^2\right]}, \tag{7.10}$$

[5]Indeed, each angle is computed as the mean of 100 angle measurements, in order to decrease the noise due to the variance, and to concentrate on the biases. In other words, it means that each pose is maintained for a duration of $10\,s$.

and the orientation RMS error is defined as[6]

$$oerr_{RMS} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(O_k - \widehat{O_k}\right)^2}. \tag{7.11}$$

But, as we are interested in improving the results for both the position and the orientation, we need to define an additional error measure, which depends on both errors. The configuration RMS error is defined as

$$err_{RMS} = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left[\left(X_k^P - \widehat{X_k}\right)^2 + \left(Y_k^P - \widehat{Y_k}\right)^2 + \left(\eta\left(O_k - \widehat{O_k}\right)\right)^2\right]}, \tag{7.12}$$

and is the error measure we want to minimize in our calibration method. In order to be consistent with respect to the units, we have introduced a coefficient η. Moreover, this coefficient allows to weight differently the position and orientation errors (this will be discussed later). For now, this coefficient is set to $\eta = 1\,{}^{m}/_{rad}$.

The results obtained with the raw measurements and the a priori beacons positions (*i.e.* without calibration) are presented in Table 7.5 on page 119, line 1. It shows a position RMS error equal to $8.7\,mm$ and an orientation RMS error equal to $0.43\,deg$. In this chapter, we propose a complete calibration procedure in order to improve the localization results. In the next sections, we detail each separate correction and, in Section 7.3.5, we present our final calibration method.

7.3.2 Power bias correction

The first correction we propose follows the conclusions of Section 5.7. We have shown that the measured angles depend on the angular window (or received power). This is illustrated in Figure 5.6 on page 77. This phenomenon has an effect of applying a non constant bias to the angles, causing localization errors. In other words, it means that if we move along a radius starting from a beacon (while keeping the orientation constant), the measured angles slightly change. This is illustrated in Figure 7.3. From a theoretical point of view, along the beacon to the sensor line, the angles are identical

$$\phi_1 = \phi_2, \tag{7.13}$$

whereas, in a practical situation, the sensor measures different angles

$$\widehat{\phi_1} \neq \widehat{\phi_2}. \tag{7.14}$$

A first solution to tackle this problem consists in fitting a curve to the measures of Figure 5.6 on page 77 (that is our first attempt to measure the power bias), and use this curve as a correction function. Unfortunately, this curve does not cover the whole range of angular windows. As explained in Section 5.7, this curve has been obtained by modifying the emitted power, for a constant distance, in order to modify the angular window. But, the current

[6]Note that the difference $\left(O_k - \widehat{O_k}\right)$ between the expected and computed orientations in equations (7.11) and (7.12) has to be normalized to fit in the $[-\pi, \pi)$ range, to avoid huge artificial errors.

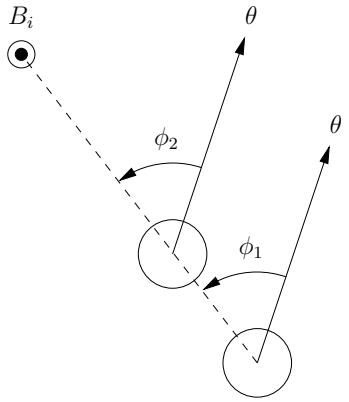

Figure 7.3: Illustration of the "power bias" experiment. B_i is a beacon. The robot is represented by a circle, and θ is the robot orientation. When the robot moves along a radius starting from a beacon (while keeping the orientation constant), we should have $\phi_1 = \phi_2$. But, in a real situation, these angles are slightly different.

hardware of the beacons does not allow to cover the full range of angular windows[7]. So, we have carried on with the same experiment with two additional distances, in order to cover the full range. The results are presented in Figure 7.4. But, as a consequence of the physical displacement of the turret during the experiment, we have lost the angle reference, and the curves are not connected together. For their representation on the graphic, we have centered each curve around zero. Different attempts have been made to merge these curves into one global curve, but the results were not effective (the correction function obtained by this method did not improve the localization results). Anyway, these results are valuable, as they indicate how the bias evolves with the angular window. From the graphic, one can observe that the bias decreases with the angular window in the lower range, then it slightly increases with the angular window in the middle range, and finally, it decreases again with the angular window in the higher range. So, we tried another method, as explained hereafter. Instead of using the results from a static experiment (as in Section 5.7), we chose to use the measures of our calibration setup directly. Note that, in our calibration procedure, we have also recorded the angular windows associated to each angle. The set of angular windows is denoted $\mathcal{W}$, and contains also n elements

$$\mathcal{W} = \{\overrightarrow{w_1}, \overrightarrow{w_2}, \ldots, \overrightarrow{w_n}\}, \tag{7.15}$$

where each element is the triplet of angular windows for each measured angles

$$\overrightarrow{w_k} = \left(w_k^1, w_k^2, w_k^3\right). \tag{7.16}$$

Indeed, the different reference positions are located at various distances from the three beacons, and it appears that the measures contain enough values of the angular window to cover its whole range. So, we propose to use our calibration setup to find the bias correction function that affects the angles. We suppose that an angle ϕ_{real} is affected by a bias, which is a function $f_{corr1}(w)$ of its associated angular window w

[7]Indeed, the hardware has changed over the years since it was in constant evolution. It appears that the current sensor has a better SNR than the one used for the first bias measurements. One of the consequences is that the angular window evolves over a bigger range that the previous versions, as well as the bias.

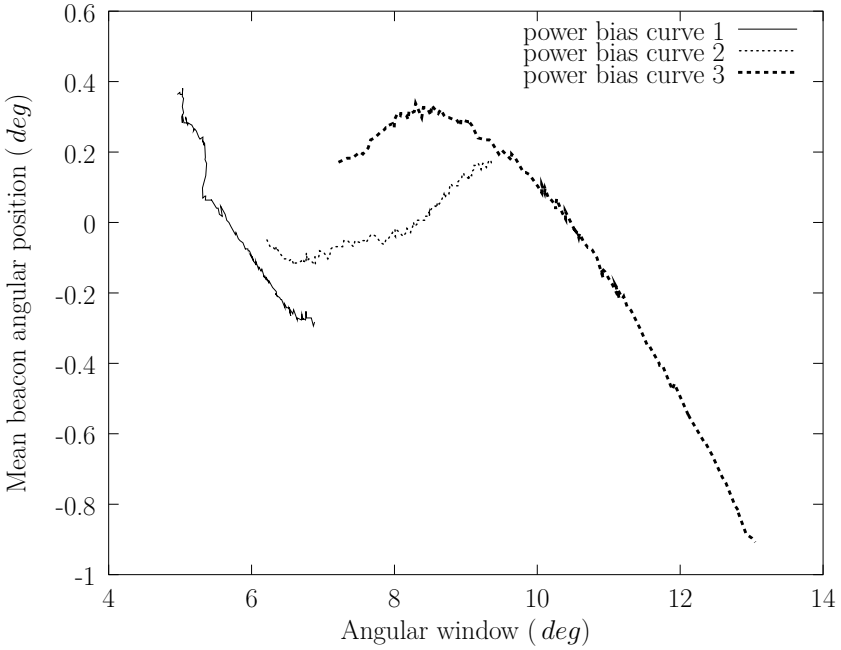

Figure 7.4: Variation of the mean beacon angular position with respect to the angular window, for three different ranges of angular windows. Each curve is obtained by modifying the emitted power of a beacon, for three different distances. To ease the comparison, each curve is centered around zero, as the angle reference has been lost during the physical displacement of the sensor.

$$\phi_{biased} = \phi_{real} + f_{corr1}(w). \tag{7.17}$$

The goal is to find an approximation of this function $\hat{f}_{corr1}(w)$, in order to correct the angle, and to reduce the bias

$$\phi_{corr1} = \phi_{biased} - \hat{f}_{corr1}(w). \tag{7.18}$$

The evolution of the bias in function of the angular window (see Figure 7.4) would suggest to approximate this function by a third order polynomial

$$\hat{f}_{corr1}(w) = a\,w^3 + b\,w^2 + c\,w + d, \tag{7.19}$$

where the coefficients $\{a,\, b,\, c,\, d\}$ are determined by minimizing the configuration RMS error (equation (7.12))

$$\{a,\, b,\, c,\, d\} = \arg\min\left(err_{RMS}\right). \tag{7.20}$$

We used the Nelder-Mead Simplex method for the minimization, with starting coefficients equal to zero. Of course, the angles of equations (7.8) and (7.9) have been corrected according to equations (7.18) and (7.19)[8]. The correction function is represented in Figure 7.5. The

[8]Note that the beacon angles, which are computed as a function of two angles (the algebraic mean), are corrected by a value, which is a function of these same angles (their difference). Indeed, it is difficult to proceed differently in our case, except if we use different datasets. So, we focus on the *resubstitution error*, that is the evaluation of the model in the same dataset. In order to compute the *generalization error*, we need to evaluate the model in another dataset. However, this technique is generally used for more complex systems with lots of variables (like machine learning), to prevent the over fitting of the data. In our case, it is not necessary since the model is quite simple, and the generalization is validated visually by moving the sensor in other poses.

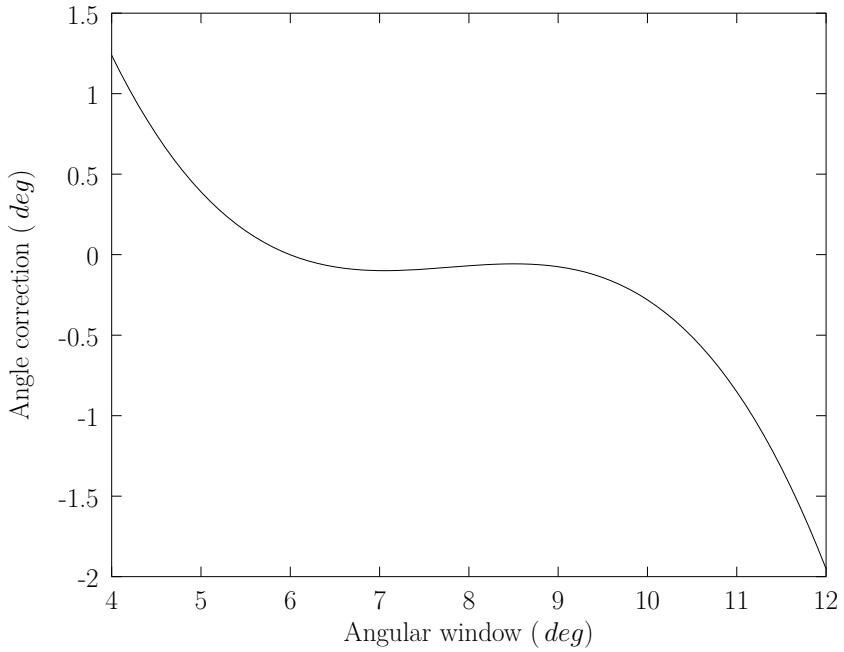

Figure 7.5: Power bias correction function applied to the angles, with respect to the angular window ($a = -89.93$, $b = 36.64$, $c = -4.93$, $d = 0.22$).

global shape of this function seems to corresponds to our observations. The results obtained with this correction function are presented in Table 7.5, line 2. Now, we obtain a position RMS error equal to $6.7\,mm$ (improvement by 23 %) and an orientation RMS error equal to $0.33\,deg$ (improvement by 23 %).

7.3.3 Rotation bias correction

This correction has been suggested by Loevsky and Shimshoni [53], and can be applied to any rotating sensor. They suggest that a measured angle is affected by a non constant bias, which depends on the orientation on the sensor. In other words, it means that if we turn around the rotation center of the sensor, the difference between the measured angles are slightly different than the real rotation angle. This is illustrated in Figure 7.6. From a theoretical point of view, we should have

$$\phi_1 = \phi_2 + (\theta_2 - \theta_1), \tag{7.21}$$

whereas, in a practical situation, we measure angles such that

$$\widehat{\phi_1} \neq \widehat{\phi_2} + (\theta_2 - \theta_1). \tag{7.22}$$

This is due to the sensor mechanics (*e.g.* non uniform distribution of the steps/magnets in the stepper motor[9]), optics (*e.g.* non uniform refraction of the acrylic protection of the turret), etc. This phenomenon can be observed in an experiment where we apply a complete rotation of the sensor/robot at the same reference position. For this particular experiment, we used

[9]The datasheet of the stepper motor (MY5602/MY7001) reports a maximum angular deviation equal to 5 % of one step, that is $0.09\,deg$. This deviation is not negligible as it is almost equal to ϕ_0, the OFF angle.

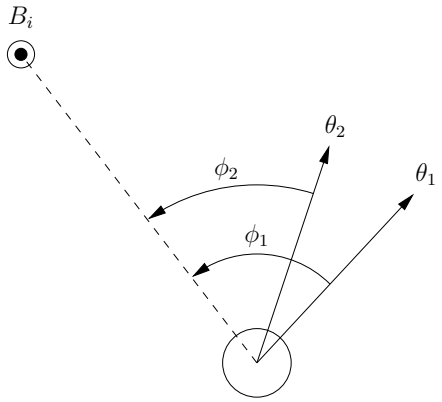

Figure 7.6: Illustration of the "rotation bias" experiment. B_i is a beacon. The robot is represented by a circle, and θ_i is the robot orientation. When the robot rotates around its center, we should have $\phi_1 = \phi_2 + (\theta_2 - \theta_1)$. But, in a real situation, these two values are slightly different.

the reference position P_9 and a modified set of 24 reference orientations

$$O_i \in \left\{ k\,\frac{\pi}{12} \mid k = 0, \ldots, 23 \right\}. \tag{7.23}$$

The results are displayed in Figure 7.7, which is the area around the reference position (represented by a cross). The computed positions are represented by stars, and are scattered around the reference position. The position RMS error is equal to $1.8\,mm$. This experiment suggests that the measured angles are not independent of the sensor rotation. To address this problem, we also suppose that an angle ϕ_{real} is affected by a bias, represented by the function $f_{corr2}(\phi)$. The goal is to find an approximation of this function $\hat{f}_{corr2}$, in order to correct the angle, and to reduce the bias

$$\phi_{corr2} = \phi_{biased} - \hat{f}_{corr2}(\phi_{biased}). \tag{7.24}$$

As this function has to be periodic with respect to the angle, Loevsky and Shimshoni [53] suggest to approximate it by its Fourier coefficients, up to the second order:

$$\hat{f}_{corr2}(\phi) = A\,\cos(\phi) + B\,\sin(\phi) + C\,\cos(2\phi) + D\,\sin(2\phi) + E. \tag{7.25}$$

As the last coefficient E adds a constant value to the angles, we have to set its value to zero: $E = 0$. If not, it would change the orientation by adding a constant value to the angles, and this effect has already been addressed by the coefficient d of equation (7.19). In addition to this correction function proposed by Loevsky and Shimshoni, we add another refinement. A closer look into the sensor structure shows that its real rotation center is not exactly located at the center of the structure. We expect a difference of at most $1\,mm$ in BeAMS, due to the mounting holes and fixing screws. Indeed, the problem is more general and can be applied to any robot equipped with this kind of sensor. From our experience, it is always difficult to measure the real position of the sensor on the robot, and this introduces localization inaccuracies. If the real rotation center of the sensor is different from the rotation center of the structure/robot, the computed positions will evolve over a circumference centered at the

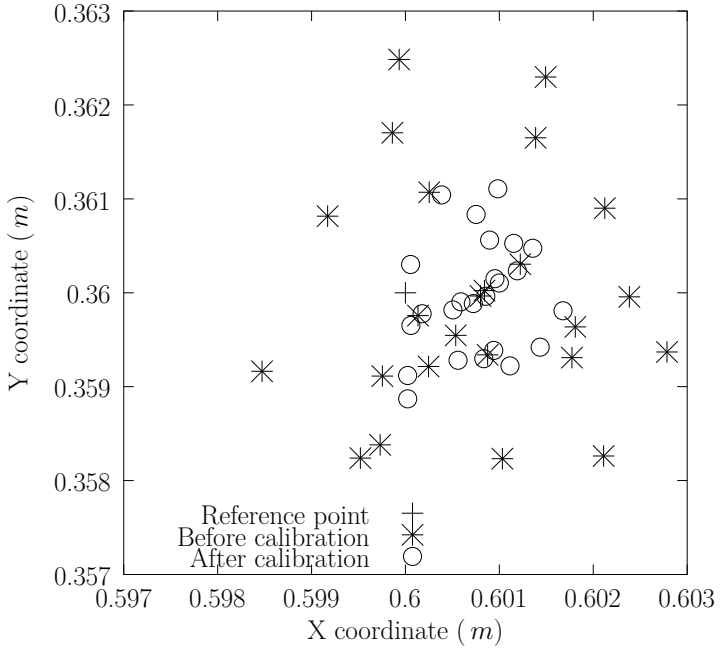

Figure 7.7: Area around the reference position used for the rotation experiment. The graph has a square aspect ratio (the side is equal to $6\,mm$, and one graduation is equal to $1\,mm$). The reference position is represented by a cross and is located at the center of the graph. The positions computed before calibration are represented by stars, and the positions computed after calibration are represented by circles.

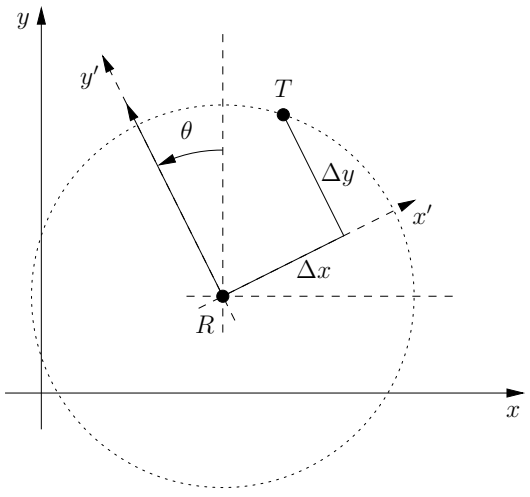

Figure 7.8: Scheme of the real position of the sensor (denoted by T) with respect to the structure/robot (denoted by R). The orientation of the structure/robot is denoted by θ. During a rotation of the structure/robot, the position of the sensor evolves over a circumference.

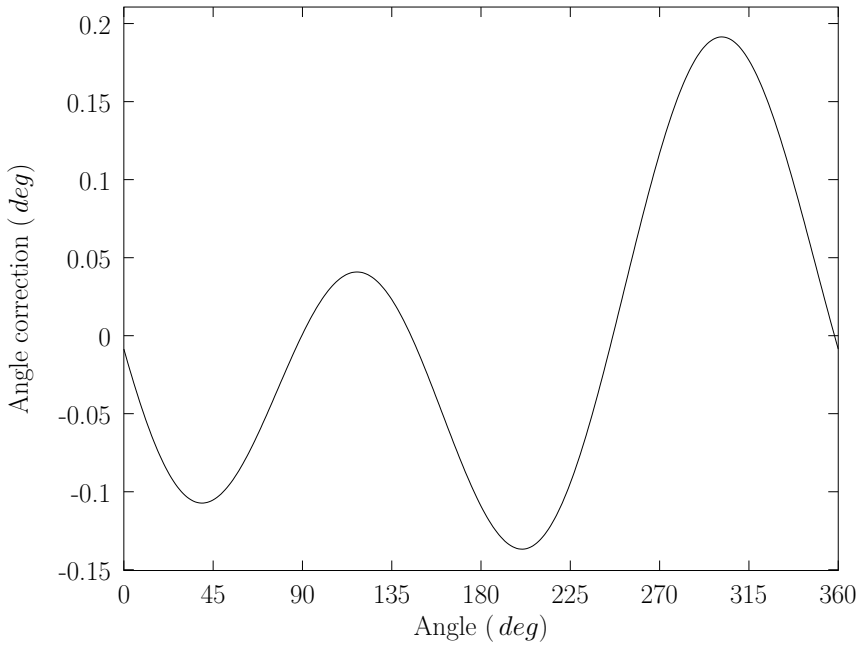

Figure 7.9: Rotation bias correction function applied to the angles ($A = 8.79 \times 10^{-4}$, $B = -1.01 \times 10^{-3}$, $C = -1.03 \times 10^{-3}$, $D = -1.74 \times 10^{-3}$).

rotation center of the structure/robot. The situation is depicted in Figure 7.8. R denotes the structure/robot, T denotes the sensor, θ denotes the orientation of the structure/robot. The local reference frame of the structure/robot is denoted by (x', y'), and (Δx, Δy) is the position of the sensor in this local frame. Therefore, the position of the sensor (X_T, Y_T) in the reference frame (x, y) is given by

$$\begin{pmatrix} X_T \\ Y_T \end{pmatrix} = \begin{pmatrix} X_R \\ Y_R \end{pmatrix} + \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}. \tag{7.26}$$

Again, the coefficients $\{A, B, C, D\}$ and $\{\Delta x, \Delta y\}$ are determined by minimizing the configuration RMS error (7.12)

$$\{A, B, C, D, \Delta x, \Delta y\} = \arg\min\left(err_{RMS}\right). \tag{7.27}$$

We used the Nelder-Mead Simplex method for the minimization, with starting coefficients equal to zero[10]. But this time, the angles of equations (7.8) and (7.9) are corrected according to equation (7.25), and the positions are corrected according to equation (7.26). The correction function is represented in Figure 7.9, and the values found for (Δx, Δy) are equal to (-1, -1.1) mm. The new position RMS error is equal to $1\,mm$ (improvement by 44 %). The positions computed after calibration are represented by circles in Figure 7.7.

7.3.4 Beacon positions calibration

The last source of errors in the "localization system components" is due to the inaccuracies in the beacons coordinates, which are generally manually measured. And, as these coordinates

[10]Whereas it is difficult to give other starting values for $\{A, B, C, D\}$, it is possible to provide manual measurements of (Δx, Δy) as a first guess.

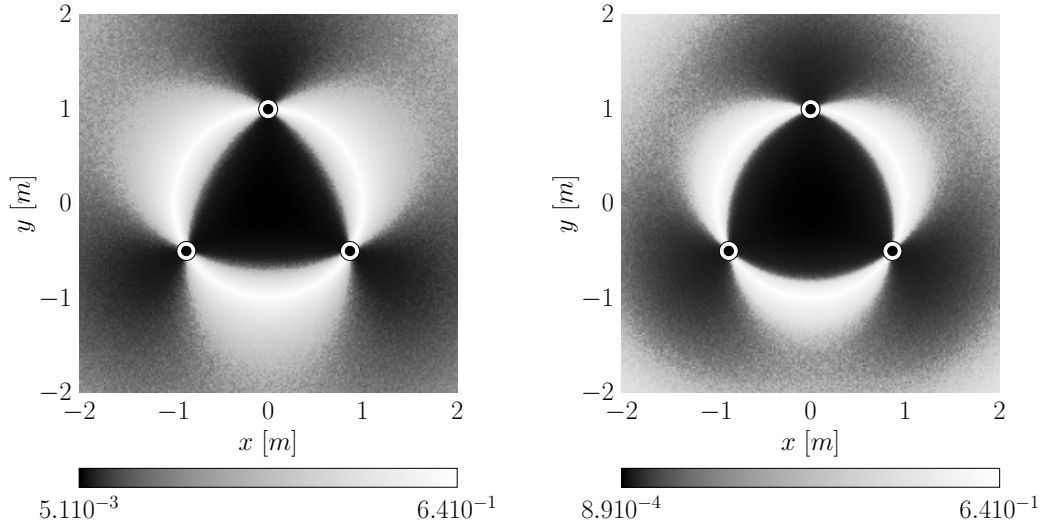

Figure 7.10: Left-hand side: simulation results giving the position errors for noisy beacons coordinates ($\sigma = 1\,cm$). Right-hand side: simulation results giving the position errors for noisy angle measurements ($\sigma = 0.1\,deg$). The beacon positions are represented by black and white dot patterns. Position errors are expressed in meters. Both graphs are displayed by using an histogram equalization in order to enhance the visual representation and to ease the interpretation.

are direct inputs of the triangulation algorithm, like the angles, it generates localization errors. In general, we study how errors on angles affect the position or the orientation (see Section 6.4.1). However, it is also interesting to observe how errors in the beacons coordinates propagate through the triangulation algorithm. To our knowledge, this "beacon sensitivity analysis" has been done only once by Easton and Cameron [28], in addition to the traditional "angle sensitivity analysis". So, in order to complete our "angle sensitivity analysis" of Section 6.4.1, we have performed simulations in which the beacons coordinates are affected by a zero mean Gaussian noise with a standard deviation equal to $1\,cm$. The simulation setup is identical as the one used in Section 6.4.1. The results are presented in Figure 7.10 (left-hand). To ease the comparison with the previous results of Section 6.4.1, we added one "angle sensitivity analysis" (for $\sigma = 0.1\,deg$), in the right-hand side. It is interesting to note that, despite small details, both graphs are similar, especially in the surrounding of the critical circumference. In other words, it means that a constant error in the beacon position is not traduced by a constant error in the computed position. Intuitively, an error in one beacon position can be converted to an equivalent error on the measured angle. The conclusion of that study is that it is also important to calibrate the beacons coordinates. Here, the method is straightforward since we have to find the beacon coordinates that minimize the configuration RMS error. The new set $\mathcal{B}$ of beacon coordinates is determined as

$$\mathcal{B} = \arg\min\left(err_{RMS}\right). \tag{7.28}$$

Again, we use the Nelder-Mead Simplex method in our experiments. The starting values of the method are the manual measurements of the beacon coordinates, which are the best guesses we can provide. The values of the new coordinates are presented in Table 7.3. The

	before calibration		after calibration	
	X^B	Y^B	X^B	Y^B
B_1	1.151	0.049	1.1504	0.0449
B_2	0.600	0.907	0.6045	0.9068
B_3	0.049	0.049	0.0452	0.0515

Table 7.3: Coordinates of the beacons, expressed in (m), before and after calibration.

results obtained with these new coordinates are presented in Table 7.5, line 5. We obtain a position RMS error equal to $6.8\,mm$ (improvement by 22 %) and an orientation RMS error equal to $0.40\,deg$ (improvement by 7 %). Note that some coordinates are corrected up to $4\,mm$, which is a little bit higher than the expected accuracy of our measurements ($1\,mm$). However, the corrections are not excessive, and the results are improved. So, it is difficult to draw conclusions about these corrections.

7.3.5 Global calibration method

Now that each separate improvement has been presented, we explain our global calibration method. Obviously, the idea is to combine the different methods. Each single method seems to improve the accuracy, but this improvement is limited by the remaining sources of errors. So it seems intuitive that combining these methods is the right solution to reach accurate results. But the question is: how do we have to combine these methods?

As explained by Loevsky and Shimshoni [53], it is difficult to elaborate a stable optimization method to a problem with multiple variables, especially if the error function has many local minima around the optimal solution. To tackle this problem, a useful technique consists in separating the problem into sub-problems, apply the optimization procedure iteratively to each problem, and then perform the global optimization with the solutions of the previous step as the starting point. As explained in the previous sections, our problem can be divided into three sub-problems. As a consequence, there are many ways (actually 6) to arrange the sub-problems into the iterative loop. To the contrary of the solution proposed by Loevsky and Shimshoni, we think that it is better to begin with the beacon coordinates calibration, because the errors caused by bad beacons coordinates can be large, as soon as we move away from the circle center. Then, by analyzing the corrections due to the power and rotation bias, it appears that the range of corrections due to the power bias is larger (about ten times) than the one for the rotation bias. So, we chose the power bias calibration as the next calibration method, followed by the rotation bias calibration. The complete calibration procedure is presented in Algorithm 7.1.

In our experiment, we set the convergence threshold to a value $t = 0.1$ to obtain good values for the starting point of the global optimization. Also, the iterative loop is executed one to two times, depending on the starting parameters. The power bias correction function is displayed in Figure 7.11. The rotation bias correction function is displayed in Figure 7.12, and the value found for $(\Delta x, \Delta y)$ is equal to $(-0.98, -0.11)\ mm$. The new beacons coordinates are presented in Table 7.4 (right-hand side). Again, some coordinates (the same as earlier) are corrected up to $4\,mm$, which is consistent with the previous experiment. Since the body of the LED measures $5\,mm$, it is possible that our manual measurements of the actual emitting part were not so accurate after all. The final position RMS error is equal to $2.16\,mm$ and

Algorithm 7.1 Global calibration method.

1. Initialize the beacons coordinates $\mathcal{B}$ with manual measurements, and all the other parameters $\{a, b, c, d, A, B, C, D, \Delta x, \Delta y\}$ to zero, if no guess can be provided.
2. **do**
3. Compute the optimal set of beacons coordinates $\mathcal{B}$ as described in Section 7.3.4, by using the current values of the other parameters $\{a, b, c, d, A, B, C, D, \Delta x, \Delta y\}$. The associated RMS error is denoted e_b.
4. Compute the optimal parameters $\{a, b, c, d\}$ as described in Section 7.3.2, by using the current values of the other parameters $\{\mathcal{B}, A, B, C, D, \Delta x, \Delta y\}$.
5. Compute the optimal parameters $\{A, B, C, D, \Delta x, \Delta y\}$ as described in Section 7.3.3, by using the current values of the other parameters $\{\mathcal{B}, a, b, c, d\}$. The associated RMS error is denoted e_r.
6. **until** $\frac{e_b - e_r}{e_b} < t$
7. Run the global optimization procedure. The starting point is composed of the different parameters computed in the iterative loop

$$\{\mathcal{B}, a, b, c, d, A, B, C, D, \Delta x, \Delta y\} = \arg\min\left(err_{RMS}\right) \tag{7.29}$$

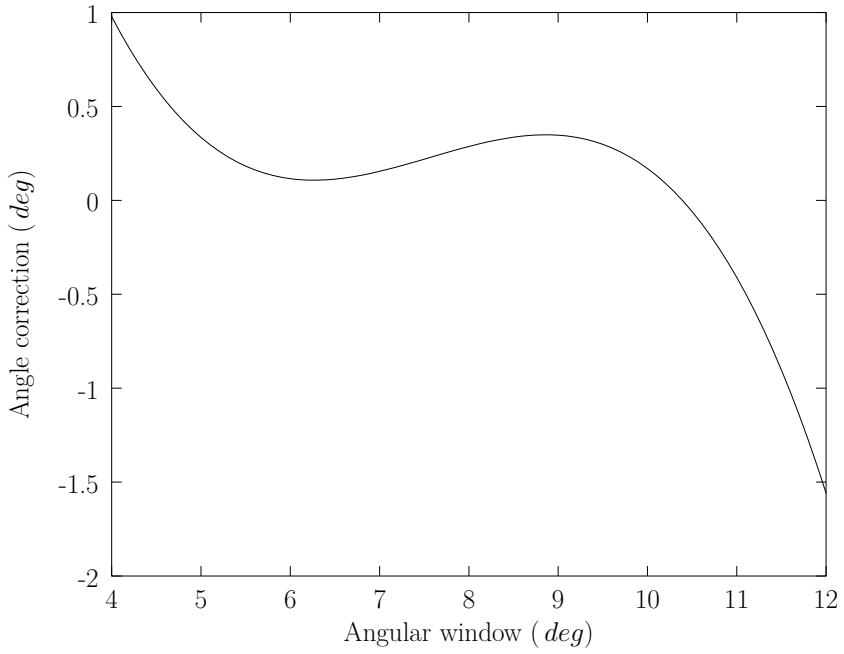

Figure 7.11: Power bias correction function applied to the angles, with respect to the angular window ($a = -90.46$, $b = 35.83$, $c = -4.59$, $d = 0.19$), after the global calibration.

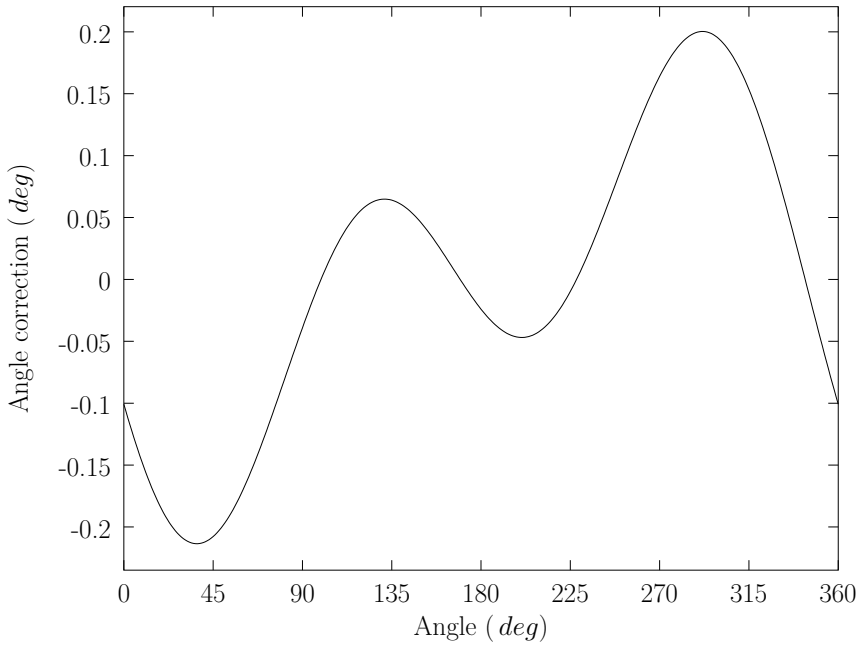

Figure 7.12: Rotation bias correction function applied to the angles ($A = -6.69 \times 10^{-4}$, $B = -1.77 \times 10^{-3}$, $C = -1.09 \times 10^{-3}$, $D = -1.89 \times 10^{-3}$), after the global calibration.

	before calibration		after calibration	
	X^B	Y^B	X^B	Y^B
B_1	1.151	0.049	1.1499	0.0455
B_2	0.600	0.907	0.6048	0.9076
B_3	0.049	0.049	0.0455	0.0516

Table 7.4: Coordinates of the beacons, expressed in (m), before and after the global calibration.

	Calibration method	Position RMS error (mm)	Orientation RMS error (deg)
1	**No calibration**	**8.75**	**0.4297**
2	Power bias only	6.67	0.3316
3	Rotation bias only	6.21	0.3157
4	Rotation bias only[1]	6.27	0.3175
5	Beacons only	6.78	0.3984
6	Power bias + beacons	6.40	0.3345
7	Power bias + rotation bias	2.95	0.1741
8	Rotation bias + beacons	2.79	0.2715
9	Global calibration[1]	2.49	0.1788
10	Global calibration[2]	2.34	0.1756
11	**Global calibration**	**2.16**	**0.1806**

[1] Without turret translation calibration (*i.e.* Δx and Δy are forced to zero).

[2] Without final combined optimization.

Table 7.5: Values of the position and orientation RMS errors for various combinations of calibration methods.

the orientation RMS error is equal to $0.18\,deg$ (see Table 7.5, line 11). This represents an improvement of 75 % and 58 % in the position and orientation RMS error respectively, with respect to the uncalibrated setup. Finally, we have represented the new computed positions in Figures 7.13 and 7.14, which are zooms into the area around each reference position (represented by a cross). The new computed positions are represented by circles, and the positions before calibration are represented by stars. One can see that the new positions are closer to their corresponding reference point, than the uncalibrated positions, which are sometimes outside the graph ($error > 7.5\,mm$).

7.3.6 Discussion

In addition to our global calibration method, we have tried other combinations and arrangements of the different methods. First, we have tried each method separately, as explained in each corresponding section. The results have already been presented in Table 7.5, lines 2 to 5. For the rotation bias experiment, we have also tried to disable the turret translation correction (line 4), by forcing Δx and Δy to zero. As expected, the position RMS error is higher, but the difference is small. We discuss this effect later. Alone, the power bias and rotation bias corrections lead to better improvements than the beacons coordinates calibration. In our case, this can be explained because the manual measurements of the beacons coordinates are quite accurate.

Also, we tried combinations of two of the three calibration methods. It appears that the rotation bias correction has an important effect on the final result, since both combinations including this correction lead to RMS errors close to the optimal solution (lines 7 and 8). From the orientation point of view, the best dual combination is “power and rotation bias correction”, since both methods correct the angles.

Then, we tried other arrangements in the iterative loop of our global calibration method, with all methods activated. In particular we tried the version of Loevsky and Shimshoni, in which the beacon calibration appears after the hardware calibration. It appears that the

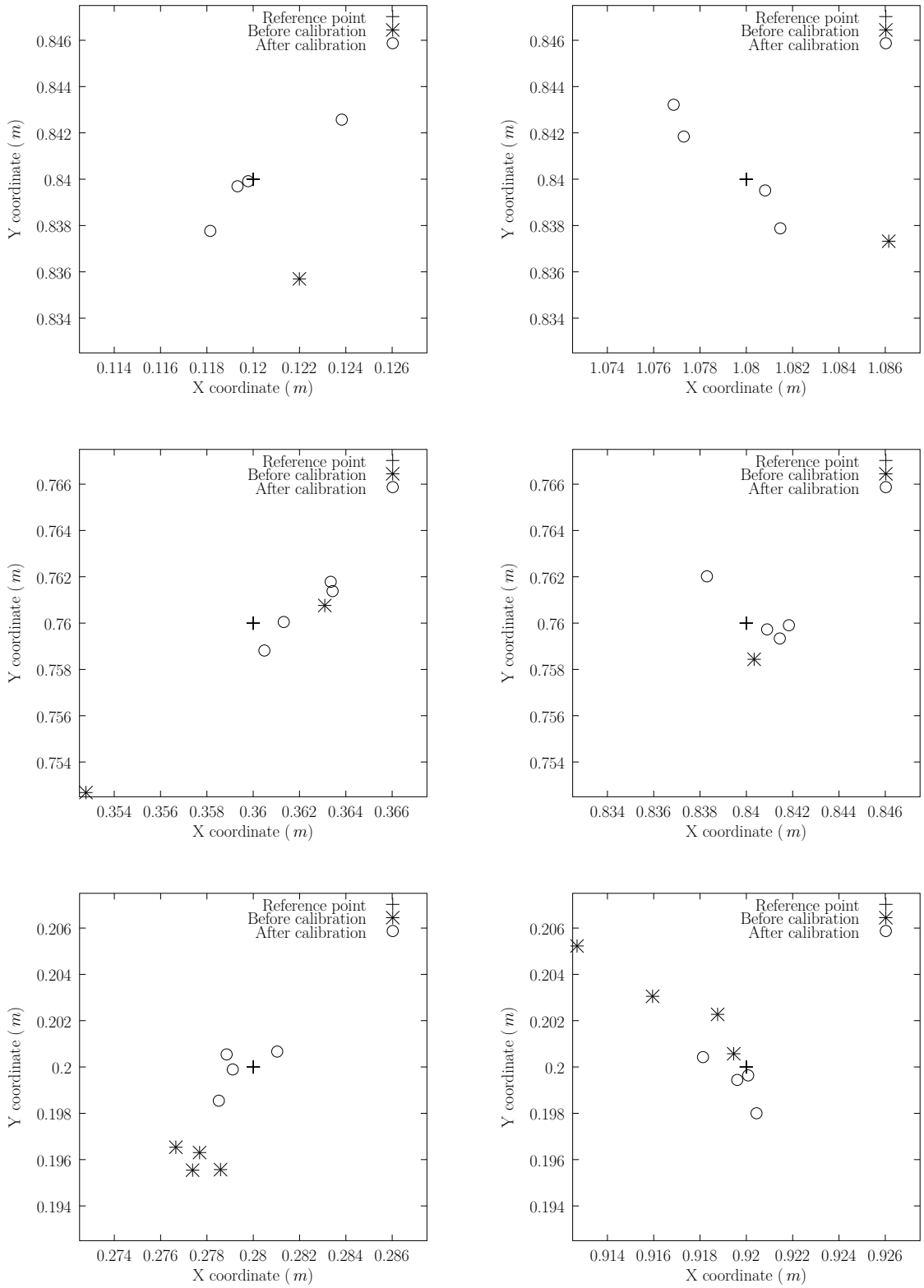

Figure 7.13: Areas around different reference positions of the test grid. All graphs have the same square aspect ratio (each side is equal to $1.5\,cm$, and one graduation is equal to $2\,mm$). Each reference position is represented by a cross located at the center of each graph. The positions computed before calibration are represented by stars, and the positions computed after calibration are represented by circles. Note that some stars (positions computed before calibration) may be outside the visible area ($error > 7.5\,mm$).

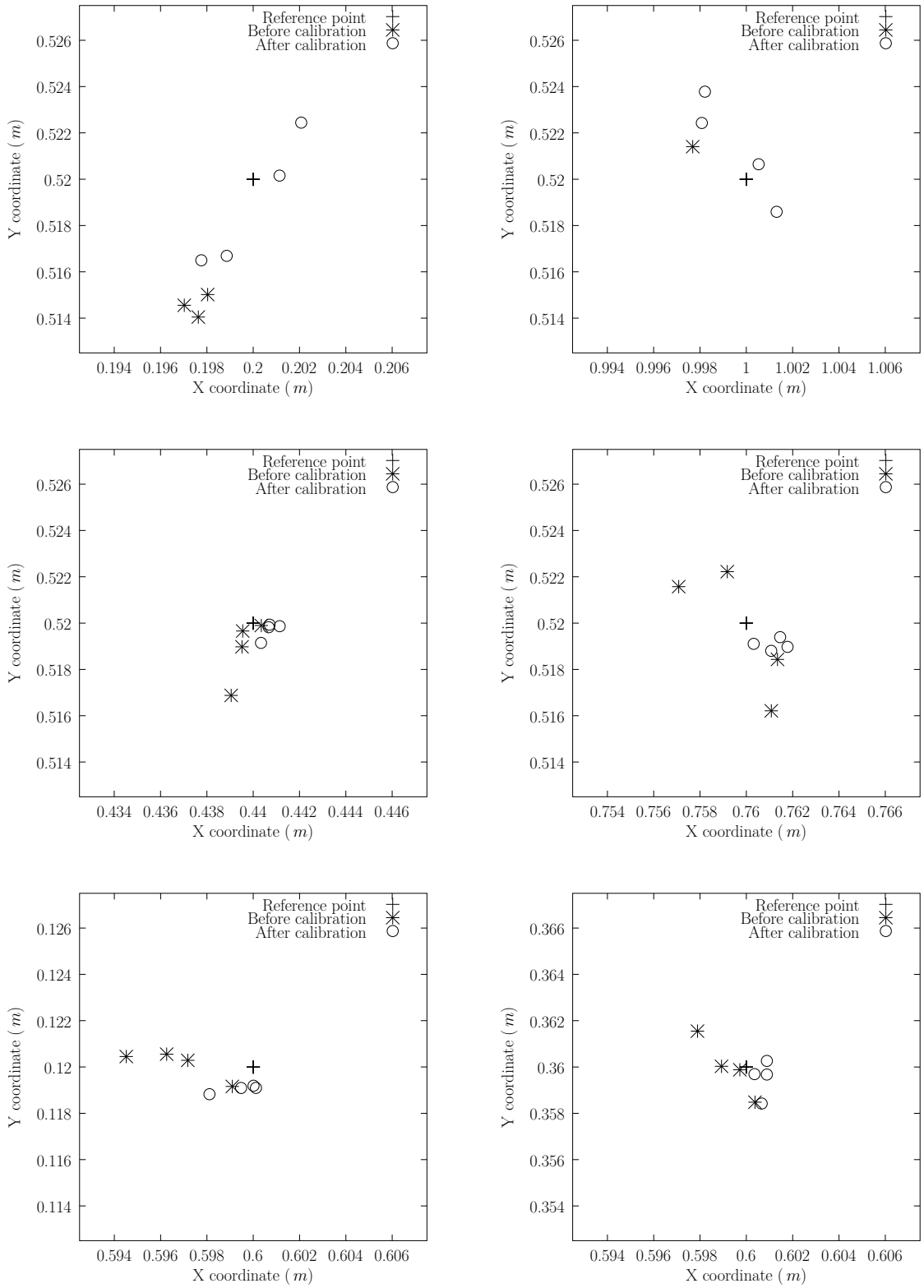

Figure 7.14: Areas around different reference positions of the test grid. All graphs have the same square aspect ratio (each side is equal to $1.5\,cm$, and one graduation is equal to $2\,mm$). Each reference position is represented by a cross located at the center of each graph. The positions computed before calibration are represented by stars, and the positions computed after calibration are represented by circles. Note that some stars (positions computed before calibration) may be outside the visible area ($error > 7.5\,mm$).

method we propose seems more stable with respect to the starting point, especially for the beacons coordinates. This can be explained because we have three beacons only and, as a consequence, the results are more sensitive to small changes in the beacons coordinates. This contrasts with multiangulation algorithms which are more robust to variations in the beacons positions, since the problem is overdetermined.

We have also tried the global calibration method without the final global optimization (line 10). In our experiment, it appears that the global optimization adds a gain of 7.7 % in the position RMS error, while the orientation error remains almost unchanged. Finally, we also tried to disable the turret translation correction in the global calibration method (line 9). Results show that this correction adds a gain of 13.3 % in the position RMS error, while the orientation error remains almost unchanged. The gain added by this correction is far more important in this case than the one observed between line 3 and 4 (0.9 %). Again, this can be explained because, in our experiment, the correction added by the turret translation is negligible against the corrections added by the power bias and the beacons calibration. In other words, it means that all corrections have to be activated in order to observe the impact of the turret translation correction.

Note that we have also tried different values for the coefficient η of the error measure (see equation (7.12)). As explained earlier, this expression mixes the notion of distance and angle in the same error function. This could seem inappropriate for a matter of units, but it is difficult to proceed differently since we want to improve both the position and the orientation. We have tried values ranging from $0.1\,{}^{m}/_{rad}$ to $10\,{}^{m}/_{rad}$. It appears that there are almost no differences in the final results (less than 0.1 %), and therefore, we kept using $\eta = 1\,{}^{m}/_{rad}$. Also, we have tried to use the other beacons to compute the orientation (see equation (7.9)). Again, it appears that the results are totally similar (differences less than 0.1 %). Therefore, we kept using beacon 1 to compute the orientation.

It is important to note that our method is different in many ways to that of Loevsky and Shimshoni. First, the hardware is different. In particular, our hardware allows the measure of an angular window, which is an image of the received power. This particularity allowed us to correct the bias due to the received power. Note that this could be applied to any angle measurement system, which has access to the received power. Then, our method is fitted for three beacons, and as a consequence, the error measures and minimization criteria are different. The disadvantage is that we need ground truth references in order to provide an error measure. Also, we changed the order of the methods in the iterative loop because it was more stable in our case. Finally, we added the refinement of the turret translation correction, which adds non negligible gain in the position RMS error.

In general, this kind of method requires lots of intuition about the underlying problem. This is necessary to establish an adequate model of a complete positioning system. The method then requires the help of an operator in order to check the optimized parameters. For example, the optimization process could fail if there is not enough data (over fitting of the data). So, it is important to check the final parameters. For example, we expect that the new beacon coordinates are close to the manual measurements (a few millimeters in our setup). For the angles, we expect power bias corrections of a few degrees maximum, and the rotation bias a few tenths of degree maximum. A hint about the good use of the global optimization step is that the final parameters should be close to the ones found in each separate step. Therefore, we recommend to try each method separately and compare the individual results to the global one.

Note also that some elements of the hardware may be calibrated separately. For example,

in the EUROBOT contest, the beacons are frequently mounted and unmounted for each game. As a result, it is difficult to associate accurate coordinates to the beacons. Moreover, it is not sure that we have the time to calibrate the beacon positions before each game. However, the other components of the model (*i.e.* the angle correction functions and the position of the turret) are worth to be calibrated, as it still adds a non negligible gain in the position accuracy (see Table 7.5, line 7). Also, note that, except for the beacon coordinates, the other calibration parameters are specific to one sensor, as it relates to its particular hardware. In other words, the power bias and rotation bias corrections, as well as the real turret position are valid for the prototype we used in our experiments.

Finally, some questions remain opened. In particular, the number of reference positions and orientations, and their positions. In their paper, Loevsky and Shimshoni do not answer these questions, and this remains difficult because it depends on many parameters including the moving area, the number of beacons, and their arrangement. It also depends on the calibration method. For example, one could give more importance to the position than the orientation. In our case, with our hardware and number of beacons, and in our calibration setup, we observed these elements. The reference positions and orientations have to be chosen such that they cover the whole range of angular windows (for the power bias correction) and the whole range of absolute angle measurements (for the rotation bias correction). Also, the reference positions have to be uniformly distributed over the moving area, which is consistent with the previous remark.

7.4 Conclusions

Many sources of errors (biases) are unavoidable in a real world situation and a good positioning system should deal with these errors. In this chapter, we propose a complete calibration method, which is mandatory in order to achieve good positioning results. Indeed, the measured angles are corrupted by many sources of errors, and the beacons coordinates may be inaccurate, leading to biased computed positions. We identify four possible causes for the angles to be corrupted. These are: (1) the received power bias, (2) the rotation bias, (3) the real turret position with respect to the robot, and (4) the real beacons coordinates with respect to manual measurements.

Based on these observations, we propose to apply some corrections to the system, that is, the two correction functions of the angles, the real position of the turret, and the real positions of the beacons. A calibration setup is designed, in order to measure the angles required by the calibration procedure, and to validate the correction functions. Then, we describe a complete calibration procedure in order to find the parameters of the correction functions. This procedure is based on the minimization of an RMS error measure.

There is almost no work about the calibration of such positioning systems, except the one of Loevsky and Shimshoni. It appears that their method cannot be applied to our system, because their method is designed for more than three beacons. So, we have extended their work to be applicable to the case of three beacons, with some variations and improvements. Note that the disadvantage of using three beacons is that we need some ground truth references, which can be time consuming. One of the improvements of our method is the "turret translation correction". In our setup, this refinement results in an additional reduction of 13.3 % in the position RMS error.

We explain that this procedure requires to avoid the over fitting of the data, and we

elaborate on it. Finally, we discuss two important choices of the calibration procedure. These are the number of reference positions and their locations on the moving area. From our experience, we would address these concerns as follows: the number of reference positions and their locations should be such that the range of variation of the different variables involved in the model are fully covered. In general, this should be obtained if the reference positions are well distributed over the moving area.

Finally, it should be noted that the improvements obtained for the RMS errors (75 % and 58 %, for the position and orientation, respectively) are valid for this calibration setup, for these reference positions and orientations. In other words, these values should not be understood as the final performance of our calibration method. What is important is not the absolute values of the RMS errors, but the improvements made in the underlying angle measurements, that lead to improved localization results. In our case, the angular RMS error corresponding to the position RMS error for the uncalibrated case ($8.7\,mm$) is equal to about $0.4\,deg$. In the case of the calibrated case ($2.16\,mm$), the angular RMS error is equal to about $0.27\,deg$. This represents an improvement of 33 % in the angular RMS error. These values have been derived during the calibration procedure, by converting the ground truth poses into ground truth angles. Therefore, the final value $0.27\,deg$ should be considered as the current angular RMS error of BeAMS, when the system is calibrated. Note that this value is different, but close to the error measure established in Chapter 5, during the performance evaluation of BeAMS. As explained earlier, the hardware system evolved continuously over the years. Moreover, we did not take into account the rotation bias, neither the real turret position in our first analysis. However, the two values are similar when the system is calibrated, and it does not change the conclusions of Chapter 5. Even for the uncalibrated hardware, BeAMS has a better performance than other prototypes and, when the system is calibrated, BeAMS is close to state of the art commercial systems (see Table 5.2 on page 77).

Chapter 8

Conclusions

Mobile robots are used increasingly in various fields, especially to transport materials in workstations, manufacturing industry, warehouses, harbors, airports, etc. In order to be totally autonomous, navigate, avoid obstacles, and execute their actions correctly, mobile robots need some form of positioning. Therefore, positioning is a crucial issue and an essential component in mobile robot applications. Relative positioning based on odometry is accurate for small offsets, but can lead to an increasing drift resulting from the unbounded accumulation of errors over time (due to the integration step, uncertainty about the wheelbase, wheel slippage, etc). Therefore, an global positioning system is required to recalibrate the position of the robot periodically. Because of the availability of angle measurement systems, triangulation has emerged as a widely used, robust, accurate, and flexible absolute positioning technique for mobile robots.

In this thesis, we present an original angle measurement system, as well as original methods and algorithms, which are parts of an absolute robot positioning system in the 2D plane. These parts are summarized hereafter.

Summary

In Chapter 2, we present our original angle measurement system, named BeAMS, that can be used by any angle-based positioning algorithm. The hardware of BeAMS consists of a sensor located on the robot, and several beacons emitting infrared light in a common horizontal plane. BeAMS has an acquisition rate of $10\,Hz$, and the entire sensor is contained in a $(8 \times 8 \times 6)\,cm^3$ volume. Also, BeAMS innovates on many points. The mechanical part of the system is kept as simple as possible (motor only, no gear system or belt) due to the hollow shaft, and it does not need an optical encoder for the motor control or angle measurement. These features tend to reduce considerably the volume of BeAMS. A simple infrared receiver is the main sensor for the angle measurements, and the beacons are common infrared LEDs emitting an On-Off Keying signal containing the beacon ID. Furthermore, the system does not require an additional synchronization channel between the beacons and the robot. Therefore, the same setup of beacons can be naturally used by multiple robots.

Then, in Chapters 3 and 4, we provide a theoretical framework to analyze the errors on the measured angles, induced by the use of an On-Off Keying modulation mechanism. In particular, we establish the upper bound of the variance affecting the angle measurements. Then, we complement the previous result by going into further details related to the code

statistics of modulated signals in general, with an emphasis on BeAMS, and establish how the variance evolves exactly as a function of the angular window. The advantage of having a model is that we can understand and predict the use of other codes in the system. In order to validate the upper bound of the variance and its evolution with respect to the angular window, we perform simulations, that match the theory.

In Chapter 5, we also provide simulated and experimental results for the variance of the beacon angle estimator. It appears that the simulations are coherent with the theory, but not entirely with the experimental results. Experimental results enlighten that the natural variance of the system is not independent on the variance added by the codes, because of the Automatic Gain Control (AGC) loop of the receiver, which is responsible for a small mismatch between the experimental and simulated results. Also, we enlighten another particularity of the receiver, called the "AND hypothesis". Indeed, it is not sure that a short leading or tailing burst (shorter than a bit) could trigger the receiver. This has the effect of virtually increasing the OFF period by a quantity equal to the minimum burst duration required to trigger the receiver. If we take into account the natural noise, as well as the "AND hypothesis", it appears that the simulations are closer to the experiments.

After the description of BeAMS and its performance analysis, we present our original three object triangulation algorithm in Chapter 6. Our algorithm, named ToTal, is based on the elegant notion of power center of three circles. ToTal natively works in the whole plane (except when the beacons and the robot are concyclic or collinear), and for any beacon ordering. Furthermore, it only uses basic arithmetic computations and two cot(.) computations. Comprehensive benchmarks show that our algorithm is faster than comparable algorithms, and simpler in terms of the number of operations. In addition, we propose a unique reliability measure of the triangulation result in the whole plane, and we establish by simulations that this reliability measure is a natural and adequate criterion to estimate the positioning error. This reliability measure can be used to identify the pathological cases (critical circumference), or as a validation gate in data fusion algorithms based on triangulation. For all these reasons, ToTal is a fast, flexible, and reliable three object triangulation algorithm. Such an algorithm is an asset for many triangulation issues related to the performance or optimization, such as error simulations, beacon placement or beacon position optimization algorithms. A fast and reliable algorithm is also an asset to initialize a more complex positioning algorithm, that internally relies on a Kalman filter for instance.

Even with the most carefully designed hardware, experiments show that it is not sufficient to achieve accurate positioning results. Indeed, many sources of biases are unavoidable in a real situation and a good positioning system should deal with them. In Chapter 7, we identify four possible causes of biases, and we propose a complete calibration procedure in order to improve the positioning results. The biases are: (1) the received power bias, (2) the rotation bias, (3) the real turret position with respect to the robot, and (4) the real beacons coordinates with respect to manual measurements. We develop a new calibration method that is designed for systems using three beacons, and which is based on the minimization of an RMS error, derived from ground truth references.

Contributions

In the following, we enlighten the main contributions of each part:

BeAMS: our angle measurement sensor introduces a new mechanism to measure angles: it

detects a beacon when it enters and leaves an angular window. This contrasts with systems that measure angles, based on the maximum received power. Our mechanism allows the sensor to analyze the temporal evolution of the received signal inside the angular window. In our case, this feature is used to code the beacon ID, but it could be used to code other kinds of information (like the beacon coordinates for example).

Theoretical framework: we provide the upper bound of the variance induced by the use of an On-Off Keying modulation mechanism. This is a general result, as long as the OFF periods of the codes all have the same duration. We also establish how the variance evolves exactly in function of the angular window. Although this study is carried on to understand and improve BeAMS, the theoretical framework is larger than that of BeAMS. It is applicable to any measurement system that estimates a value by taking the mean of a previous event and a later event, based on the reception of an On-Off Keying modulated signal. The advantage of having a model is that we can understand and predict the use of other codes in the system.

ToTal: our three object triangulation algorithm is faster than comparable algorithms, and simpler in terms of the number of operations. We also compare the number of basic arithmetic computations, squared roots, and trigonometric functions used for eleven known triangulation algorithms. In addition, we propose a unique reliability measure of the triangulation result in the whole plane, and we establish by simulations that this reliability measure is a natural and adequate criterion to estimate the positioning error. To our knowledge, none of the algorithms of the same family does provide such a measure. Finally, the C source code of all algorithms, including ToTal, as well as the code for the error analysis and benchmarks, are made available to the scientific community[1].

Calibration method: we extend the work of Loevsky and Shimshoni [53] to systems based on three beacons. To our knowledge, such a calibration method has never been proposed. Also, we add two refinements to their method. The first one is the correction due to the real sensor location on the robot. The second one is the correction of the bias due to the variation of the received power ("power bias"). This correction is possible in the case of BeAMS, thanks to the new mechanism to measure angles based on an angular window, which is related to the received power. However, this correction can be applied to any system that can measure the received power.

System deployment

Our motivation for this work was to create a new system optimized for the Eurobot contest, which imposes many constraints. As a consequence, it was impossible to use a commercial sensor, and we have created our own system. However, BeAMS can be used in any other mobile robot applications based on angle measurements. In particular, we explain how to increase the covered area and the number of codes (*i.e.* beacons), which are the main requirements for a larger system deployment. Moreover, note that BeAMS identifies the beacons, which is an asset to (re)initialize the position unequivocally (wake-up or kidnapped issues),

[1] `http://www.ulg.ac.be/telecom/triangulation`

or to feed a data fusion algorithm without requiring a data association step. Also, multiple robots can use the same setup of beacons, without disturbing each other, and without requiring a modification of the system.

Finally, note that we have concentrated on methods and algorithms based on three beacons, because the EUROBOT contest allows a maximum of three beacons. However, it is important to note that BeAMS can measure angles for any number of beacons, and it is not limited to methods nor algorithms based on three beacons.

Performance analysis

In the following, we discuss the performance analysis of angle measurement systems in general. Then, we present the performance of BeAMS, and explain how we have carried out its evaluation.

All along this manuscript, we point out that an angle measurement system should not be characterized through a positioning algorithm, because it depends on the relative configuration of the robot with respect to the beacons, and the number of beacons. In practice however, angle measurement systems are developed almost exclusively for positioning, as the process of triangulation requires angle measurements. This explains why most authors only evaluate positioning algorithms or present complete systems (hardware and software). As a consequence, it is rare that authors evaluate the performance of the underlying angle measurement system, and there is a lot of confusion about the evaluation criteria. Indeed, the problem is not to use a positioning algorithm (or any other method) to evaluate the performance, the problem is to evaluate an angle measurement system with a distance criterion. Instead, we believe that an angle measurement system should be characterized only by the error affecting the measured angles, whatever the method used for the evaluation is. The advantage is that this measure is not related to any particular setup, and therefore, it eases the comparison between hardware systems. Moreover, this error measure combined with a simulator, is sufficient to predict the performance (in terms of distance) of any hardware system, and for any arrangement of beacons.

In our case, we tried to evaluate the variance and the bias affecting the angle measurements, because we believe that these two values are sufficient to fully characterize the measurements. But, compared to the ease of measuring the variance, it is not a simple task to evaluate the bias. In Chapter 5, we have established a first method to measure the bias, related to the variation of the received power. The combination of the variance and the bias was our first try to measure the performance of BeAMS. However, after a deeper study related to the calibration procedure, it has appeared that the bias related to the received power was not the sole source of errors. But, as it is not easy to characterize the other sources of errors separately, we decided to evaluate them through our calibration setup and triangulation algorithm, which involves distance measurements. Therefore, in order to compute an error measure that does not depend on our particular setup, we derived the underlying angular RMS error corresponding to the position error observed in our setup. The advantage of this method is that it takes into account all the possible causes of errors, even those that we do not identify. In other words, it means that our error model is a simplified version of the real error process. Also, this evaluation procedure introduces an additional error, due to the manual placement of the sensor/robot at the different poses. This tends to slightly overestimate the real angular RMS error. In our case, this angular RMS error has been evaluated

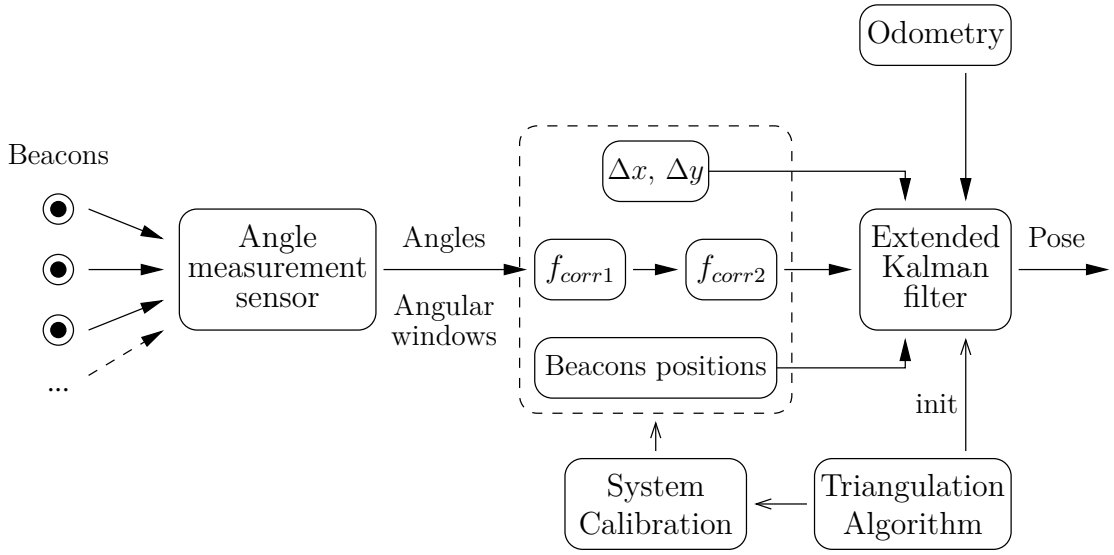

Figure 8.1: Scheme of a complete positioning system in the 2D plane, based on angle measurements and odometry, which are merged into an Extended Kalman filter.

to $0.4\,deg$ without calibration, and to $0.27\,deg$, after the calibration procedure. Even for the uncalibrated hardware, BeAMS has a better performance than other prototypes found in the literature and, when the system is calibrated, BeAMS is close to state of the art commercial systems (performance around $0.1\,deg$).

Involvement of our work in a complete positioning system

In this thesis, we present components of a complete positioning system for mobile robots in the 2D plane. More precisely, we focus on the description of a new angle measurement system, and its performance evaluation in static conditions. Indeed, the simple combination of the angle measurement system with the triangulation algorithm defines a positioning system. However, the lack of odometry, as well as the low acquisition rate of the angular sensor (compared to odometry), make this positioning system inadequate in dynamic conditions. Therefore, in our current implementation of BeAMS in our robots, we use an Extended Kalman filter (EKF), which merges the odometry with the angle measurements. The scheme of the complete positioning system is represented in Figure 8.1. In this thesis, we concentrate on each part of this scheme, except the EKF and the odometry. In other words, our goal is to provide accurate angle measurements to the EKF, as well as accurate data for the beacons positions and for the real sensor position with respect to the robot. In the following, we summarize how each part of our work is involved in the complete positioning system represented in Figure 8.1:

BeAMS: obviously, BeAMS is used as the angle measurement sensor. It can be used in systems based on angle measurements to any number of beacons.

Theoretical framework and simulator: the theory about the code statistics has been used to understand and improve BeAMS. In particular, the upper bound of the variance has been used to build the beacon codes, in conjunction with the simulator. The theoretical framework and the simulator are useful tools to predict the use of other

codes in the system. The exact evolution of the variance with respect to the angular window should be used to feed the EKF with adequate variance values, but it is not used in our current implementation.

ToTal: even if it is not used as the main positioning algorithm, a stand alone triangulation algorithm is useful in many cases. Our triangulation algorithm is used to initialize the EKF. It is also used as a building block of our calibration method, in order to compute the error measure necessary for the minimization process. Finally, we use it to analyze the position error sensitivity in the moving area.

Calibration method: it is used to improve the positioning results. In particular, it computes the parameters of the two correction functions, in order to provide accurate angle measurements to the EKF. Moreover, it computes the real beacons coordinates, as well as the real turret position (Δx, Δy), which are also inputs of the EKF.

Improvements

In our current implementation of BeAMS in the complete positioning system (which relies on an Extended Kalman filter), we use the best upper bound of the variance all over the plane. But this practice is far from being optimal. Indeed, since the variance evolves with the angular window (or indirectly with the position of the robot in the plane), the upper bound unnecessarily overestimates the real variance. Moreover, our theory establishes how the variance evolves exactly with respect to the angular window. In order to feed a tracking system, it would be better to provide the exact variance for each position in the plane, and for each beacon/code.

We have shown that the performance of BeAMS is mainly limited by the biases affecting the measurements (a few tenths of degree). In comparison, the maximum standard deviation encountered in BeAMS has been evaluated to $0.074\,deg$, which is lower than the error measure of most similar systems (see Table 5.2 on page 77). The bigger part of the biases is due to the variation of the received power (power bias), in conjunction with the use of an Automatic Gain Control (AGC) loop in the receiver, which changes the receiver characteristics at the entrance and exit of the angular window. But, despite these limitations, we have shown that good positioning results could be obtained with such a receiver (*i.e.* with an AGC loop), if the system is calibrated. All these features make BeAMS a small, low power, flexible, and tractable solution for robot positioning. BeAMS has now been used successfully during the Eurobot contest for four years. However, we believe that far better results could be obtained with the use of a new dedicated receiver. For example, a read access to the AGC, or a direct control of the gain by the system could help reducing the bias due to the received power. Another solution is to use a cascaded filtering strategy, instead of an AGC, as proposed by Roberts *et al.* [73]. The idea consists in cascading a few fixed gain filters in order to cover the whole range of variation of the received power, and to select the adequate stage for each incoming signal. As a consequence, the characteristics at the entrance and exit of the angular window are similar, which should reduce considerably the power bias.

Appendix A

Theoretical developments related to the code statistics

A.1 Mean and variance of a random variable whose *PDF* is expressed as a weighted sum (mixture) of *PDFs*

Let X be a random variable with a probability density function (PDF) $f_X(x)$

$$X \sim f_X(x). \tag{A.1}$$

Suppose that $f_X(x)$ can be written as a weighted sum (or mixture) of functions $f_i(x)$

$$f_X(x) = \sum_i k_i f_i(x), \tag{A.2}$$

whose coefficients k_i sum up to 1

$$\sum_i k_i = 1, \tag{A.3}$$

and that the area under each $f_i(x)$ is equal to 1

$$\int_{-\infty}^{+\infty} f_i(x)\, dx = 1, \tag{A.4}$$

so that we have

$$\int_{-\infty}^{+\infty} f_X(x)\, dx = \int_{-\infty}^{+\infty} \sum_i k_i f_i(x)\, dx = \sum_i k_i \int_{-\infty}^{+\infty} f_i(x)\, dx = 1. \tag{A.5}$$

The existence theorem (see [63, page 73]) states the following.

Theorem A.1. *If $f_i(x)$ is nonnegative, if its area is equal to 1, if its integral $F_i(x)$ is continuous from the right, and if, as x increases from $-\infty$ to ∞, $F_i(x)$ increases monotonically from 0 to 1, then we can define a random variable X_i whose* PDF *is $f_i(x)$*

$$X_i \sim f_i(x) = f_{X_i}(x). \tag{A.6}$$

Then the expectation of X is given by

$$E\{X\} = \sum_i k_i \, E\{X_i\} \tag{A.7}$$

and the variance of X is given either by

$$var\{X\} = \sum_i k_i \left[var\{X_i\} + E\{X_i\}^2 \right] - (E\{X\})^2 , \tag{A.8}$$

or by

$$var\{X\} = \sum_i k_i \, var\{X_i\} + \sum_{i<j} k_i k_j \, (E\{X_i\} - E\{X_j\})^2 . \tag{A.9}$$

Proof. For the mean, we have

$$\begin{aligned} E\{X\} &= \int_{-\infty}^{+\infty} x f_X(x)\, dx \\ &= \int_{-\infty}^{+\infty} x \sum_i k_i \, f_{X_i}(x)\, dx \\ &= \sum_i k_i \int_{-\infty}^{+\infty} x f_{X_i}(x)\, dx \\ &= \sum_i k_i E\{X_i\} . \end{aligned} \tag{A.10}$$

The first expression of the variance is obtained as follows

$$\begin{aligned} var\{X\} &= \int_{-\infty}^{+\infty} (x - E\{X\})^2 f_X(x)\, dx \\ &= \int_{-\infty}^{+\infty} x^2 f_X(x)\, dx - (E\{X\})^2 \\ &= \int_{-\infty}^{+\infty} x^2 \sum_i k_i \, f_{X_i}(x)\, dx - (E\{X\})^2 \\ &= \sum_i k_i \int_{-\infty}^{+\infty} x^2 f_{X_i}(x)\, dx - (E\{X\})^2 \\ &= \sum_i k_i \left[var\{X_i\} + E\{X_i\}^2 \right] - (E\{X\})^2 , \end{aligned} \tag{A.11}$$

and the second expression of the variance is obtained by starting from the previous result

$$\begin{aligned}
var\{X\} &= \sum_i k_i \left[var\{X_i\} + E\{X_i\}^2\right] - (E\{X\})^2 \\
&= \sum_i k_i \left[var\{X_i\} + E\{X_i\}^2\right] - \left(\sum_i k_i E\{X_i\}\right)^2 \\
&= \sum_i k_i \left[var\{X_i\} + E\{X_i\}^2\right] - \left[\sum_i k_i^2 E\{X_i\}^2 + \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\}\right] \\
&= \sum_i k_i var\{X_i\} + \left[\sum_i k_i E\{X_i\}^2 - \sum_i k_i^2 E\{X_i\}^2\right] - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_i k_i (1-k_i) E\{X_i\}^2 - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_i k_i \sum_{j,\, j\neq i} k_j E\{X_i\}^2 - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_{j\neq i} k_i k_j E\{X_i\}^2 - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_{i<j} k_i k_j E\{X_i\}^2 + \sum_{j<i} k_i k_j E\{X_i\}^2 - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_{i<j} k_i k_j E\{X_i\}^2 + \sum_{i<j} k_j k_i E\{X_j\}^2 - \sum_{i<j} 2k_i k_j E\{X_i\} E\{X_j\} \\
&= \sum_i k_i var\{X_i\} + \sum_{i<j} k_i k_j \left(E\{X_i\}^2 + E\{X_j\}^2 - 2E\{X_i\} E\{X_j\}\right) \\
&= \sum_i k_i var\{X_i\} + \sum_{i<j} k_i k_j (E\{X_i\} - E\{X_j\})^2 . \qquad \text{(A.12)}
\end{aligned}$$

□

A.2 Means and variances of Φ_r and Φ_f

A.2.1 Mean and variance of Φ_r

The *PDF* of Φ_r is given by

$$f_{\Phi_r}(\phi) = p_1 \delta(\phi - \phi_R) + p_0 U_{(\phi_R, \phi_R + \phi_0)}(\phi) . \qquad \text{(A.13)}$$

By using (A.7), we can compute the mean of Φ_r

$$\begin{aligned}
\mu_{\Phi_r} &= p_1 \phi_R + p_0 \left(\phi_R + \frac{\phi_0}{2}\right) \qquad \text{(A.14)} \\
&= \phi_R + p_0 \frac{\phi_0}{2} . \qquad \text{(A.15)}
\end{aligned}$$

The variance of Φ_r is computed after (A.9)

$$\begin{aligned}
\sigma^2_{\Phi_r} &= p_1 0 + p_0 \frac{\phi_0^2}{12} + p_1 p_0 \left(\phi_R - \left(\phi_R + \frac{\phi_0}{2} \right) \right)^2 && \text{(A.16)} \\
&= p_0 \frac{\phi_0^2}{12} + (1 - p_0)\, p_0 \left(\frac{\phi_0}{2} \right)^2 && \text{(A.17)} \\
&= p_0 \frac{\phi_0^2}{12} + p_0 \frac{\phi_0^2}{4} - p_0^2 \frac{\phi_0^2}{4} && \text{(A.18)} \\
&= p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}. && \text{(A.19)}
\end{aligned}$$

A.2.2 Mean and variance of Φ_f

The *PDF* of Φ_f is given by

$$f_{\Phi_f}(\phi) = p_1 \delta(\phi - \phi_F) + p_0\, U_{(\phi_F - \phi_0, \phi_F)}(\phi). \tag{A.20}$$

By using (A.7), we can compute the mean of Φ_f

$$\begin{aligned}
\mu_{\Phi_f} &= p_1 \phi_F + p_0 \left(\phi_F - \frac{\phi_0}{2} \right) && \text{(A.21)} \\
&= \phi_F - p_0 \frac{\phi_0}{2}. && \text{(A.22)}
\end{aligned}$$

The variance of Φ_f is computed after (A.9)

$$\begin{aligned}
\sigma^2_{\Phi_f} &= p_1 0 + p_0 \frac{\phi_0^2}{12} + p_1 p_0 \left(\phi_F - \left(\phi_F - \frac{\phi_0}{2} \right) \right)^2 && \text{(A.23)} \\
&= p_0 \frac{\phi_0^2}{12} + (1 - p_0)\, p_0 \left(\frac{\phi_0}{2} \right)^2 && \text{(A.24)} \\
&= p_0 \frac{\phi_0^2}{12} + p_0 \frac{\phi_0^2}{4} - p_0^2 \frac{\phi_0^2}{4} && \text{(A.25)} \\
&= p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}. && \text{(A.26)}
\end{aligned}$$

A.3 Details for the different cases

A.3.1 Details for the case A

The purpose of this section is to establish the $p^\bullet$ and $p^\star$ probabilities.

To compute these probabilities, we have to measure the total duration for which the different events occur over a code period, and then divide by the code period (we can do this since the codes are periodic). The situation is depicted in Figure A.1. The scheme represents the generalized case, for any code i. We have to shift the code from left to right (for a fixed value of the angular window), and check when the different events occur, until we have analyzed the complete period of the code. We can distinguish between 8 different scenarios ($1 \to 8$) depending on the angular shift:

- 0: initial condition: the left extremity of ϕ_W coincides with the start of a code period (no shift),

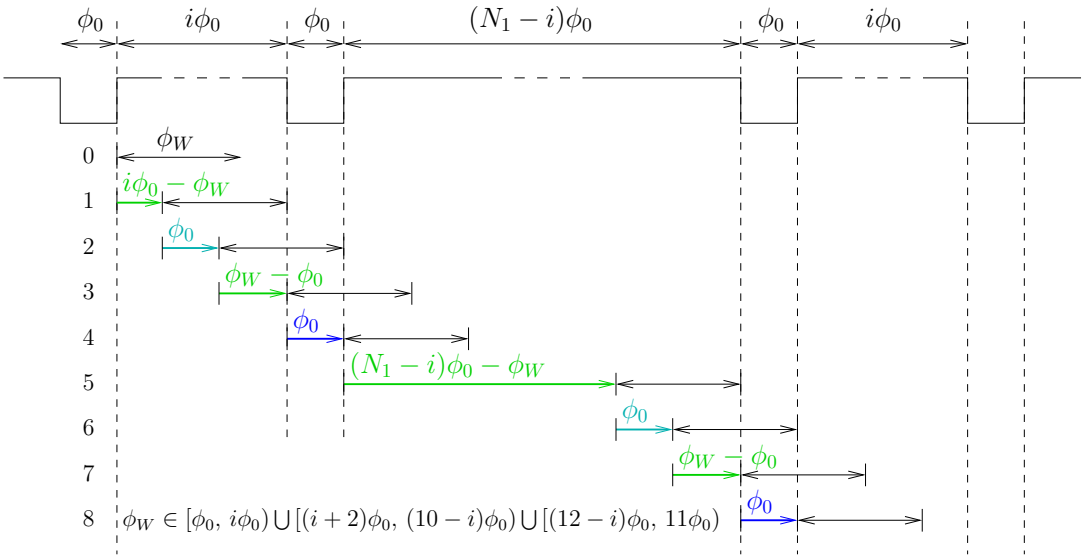

Figure A.1: Computation of the probabilities and intervals associated with the case A, and for all codes (ϕ_W is supposed to be constant).

- 1: we can shift ϕ_W for $i\phi_0 - \phi_W$ with no error,
- 2: for the next ϕ_0 shift, there is an error on Φ_f,
- 3: we can shift ϕ_W for $\phi_W - \phi_0$ with no error,
- 4: for the next ϕ_0 shift, there is an error on Φ_r,
- 5: we can shift ϕ_W for $(N_1 - i)\phi_0 - \phi_W$ with no error,
- 6: for the next ϕ_0 shift, there is an error on Φ_f,
- 7: we can shift ϕ_W for $\phi_W - \phi_0$ with no error,
- 8: for the next ϕ_0 shift, there is an error on Φ_r.

Note that in Figure A.1, as for all other cases (Figures A.2, A.3, A.4, A.5, A.6), we use the following color convention:

- green: no error,
- cyan: $\mathcal{E}_r = 0$ and $\mathcal{E}_f \neq 0$,
- blue: $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f = 0$,
- red: $\mathcal{E}_r \neq 0$ and $\mathcal{E}_f \neq 0$ (for the cases B, C, D, and E only),
- brown: no edge detection at the receiver (for the case 0 only).

Now we add the favorable parts (lines 1, 3, 5, and 7, or green parts of the drawing) to find the duration of the event associated to $p^\bullet$

$$\phi^\bullet = (i\phi_0 - \phi_W) + (\phi_W - \phi_0) + ((N_1 - i)\phi_0 - \phi_W) + (\phi_W - \phi_0) = N_1\phi_0 - N_0\phi_0. \quad \text{(A.27)}$$

To compute $p^{\bullet}$, we have to divide this duration by the period duration of a code $N_b\phi_0 = (N_1 + N_0)\phi_0$

$$p^{\bullet} = \frac{N_1\phi_0 - N_0\phi_0}{N_b\phi_0} = \frac{N_1}{N_b} - \frac{N_0}{N_b} = p_1 - p_0. \tag{A.28}$$

Firstly, we can notice that this probability does not depend on the angular window ϕ_W (if the value belongs to the intervals defined for the case A). Secondly, we notice that this probability does not depend on the code. Of course the case A for the other codes does not occur for the same intervals, but the probabilities, and the *PDFs* are the same.

Now we have to compute $p^{\star}$. Let us consider the errors on Φ_f first. We can use the same reasoning as before and use Figure A.1 to count favorable and unfavorable situations. In this case, the favorable parts are the cyan parts of lines 2 and 6. The counting gives

$$\phi^{\star} = 2\phi_0 = N_0\phi_0, \tag{A.29}$$

and the probability $p^{\star}$ to commit an error on Φ_f is obtained by dividing it by the period duration

$$p^{\star} = \frac{N_0\phi_0}{N_b\phi_0} = \frac{N_0}{N_b} = p_0. \tag{A.30}$$

The same reasoning also applies to Φ_r, but for the blue parts of line 4 and 8. The probability to commit an error on Φ_r is the same as for Φ_f, explaining why the notation $p^{\star}$ is the same for both. We can check that the probabilities for the case A sum up to 1

$$p^{\bullet} + p^{\star} + p^{\star} = p_1 - p_0 + p_0 + p_0 = p_1 + p_0 = 1. \tag{A.31}$$

A.3.2 Details for the case 0

The probabilities $p^{\bullet}$ and p^{*} are computed as follows. To compute these probabilities, we have to measure the total duration for which the different events occur over a code period, and then divide by the code period (we can do this since the codes are periodic). The situation is depicted in Figure A.2.

The scheme represents the generalized case, for any code i. One has to move virtually the code from left to right (for a fixed value of the angular window) and check when the different events occur, until we reach the period of the code. We can distinguish between 8 different scenarios depending on the angular shift:

- 0: initial condition: the left extremity of ϕ_W coincides with the start of a code period (no shift),
- 1: we can shift ϕ_W for $i\phi_0 - \phi_W$ with no error,
- 2: then we can shift ϕ_W for a duration ϕ_W, with an error on Φ_f only,
- 3: for the next $\phi_0 - \phi_W$ move, there is no edge detection at the receiver since there is no 1 symbol between the rising and falling edge,
- 4: then we can shift ϕ_W for a duration ϕ_W, with an error on Φ_r only,
- 5: we can shift ϕ_W for $(N_1 - i)\,\phi_0 - \phi_W$ with no error,
- 6: same as line 2,

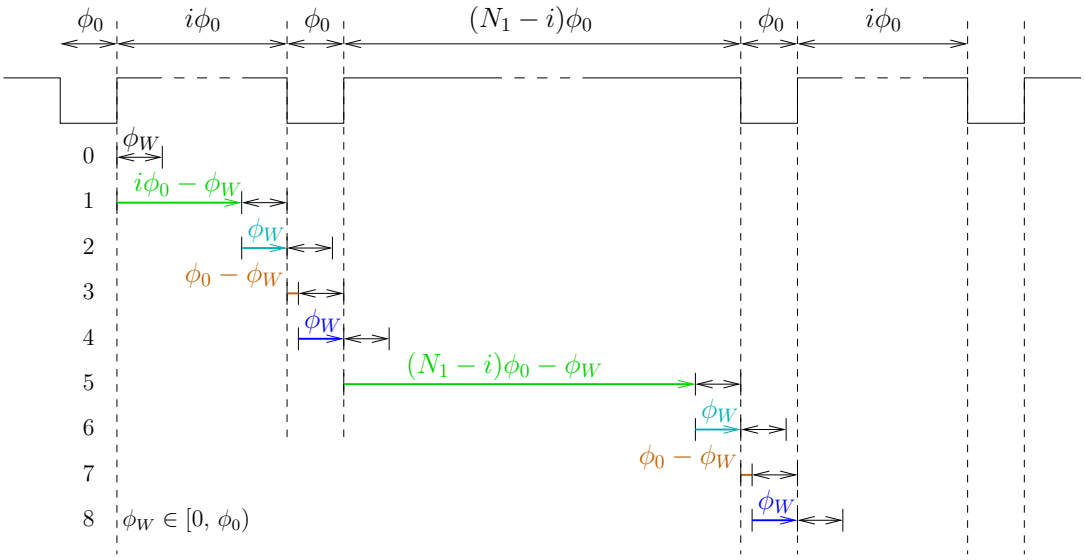

Figure A.2: Computation of the probabilities and intervals associated with the case 0, and for all codes ($\lambda_0 = 0.8$ in the scheme).

- 7: same as line 3,
- 8: same as line 4.

This is similar to the case A, except that there are two parts (lines 3 and 7, or brown parts of the drawing) where there is no edge detection at all. To compute $p^\bullet$, we first add the favorable parts (lines 1, and 5, or green parts of the drawing)

$$\begin{aligned} \phi^\bullet &= (i\phi_0 - \phi_W) + ((N_1 - i)\,\phi_0 - \phi_W) \\ &= N_1\phi_0 - 2\phi_W = N_1\phi_0 - N_0\phi_W = N_1\phi_0 - N_0\phi_0\lambda_0. \end{aligned} \tag{A.32}$$

But unlike the case A (and all other cases), we cannot divide this duration by the code duration. Indeed, we have to remove from the code duration the parts where there is no edge detection (lines 3 and 7, or brown parts of the drawing)

$$\begin{aligned} \phi_{duration-case\,0} &= (N_1 + N_0)\,\phi_0 - 2\,(\phi_0 - \phi_W) \\ &= (N_1 + N_0)\,\phi_0 - N_0\,(\phi_0 - \phi_W) \\ &= (N_1 + N_0)\,\phi_0 - N_0\phi_0\,(1 - \lambda_0) \\ &= N_1\phi_0 + N_0\phi_0\lambda_0. \end{aligned} \tag{A.33}$$

Now we can compute $p^\bullet$

$$p^\bullet = \frac{\phi^\bullet}{\phi_{duration-case\,0}} = \frac{N_1\phi_0 - N_0\phi_0\lambda_0}{N_1\phi_0 + N_0\phi_0\lambda_0} = \frac{p_1 - p_0\lambda_0}{p_1 + p_0\lambda_0}. \tag{A.34}$$

Now , it remains to calculate p^*. Let us consider the errors on Φ_f first. We can use the same reasoning as before and use Figure A.2 to do the counting. But in this case, the favorable parts are the cyan parts of lines 2 and 6. The counting gives

$$\phi^* = 2\phi_W = N_0\phi_W = N_0\phi_0\lambda_0, \tag{A.35}$$

and the probability p^* to commit an error on Φ_f is obtained by dividing ϕ^* by the reduced period duration

$$p^* = \frac{\phi^*}{\phi_{duration-case\,0}} = \frac{N_0\phi_0\lambda_0}{N_1\phi_0 + N_0\phi_0\lambda_0} = \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}. \tag{A.36}$$

The same reasoning also applies to Φ_r, but for the blue parts of line 4 and 8. The probability to commit an error on Φ_r is the same as for Φ_f, explaining why the notation p^* is the same for both. Again, we can check that the probabilities for the case 0 sum up to 1

$$p^\bullet + p^* + p^* = \frac{p_1 - p_0\lambda_0}{p_1 + p_0\lambda_0} + 2\,\frac{p_0\lambda_0}{p_1 + p_0\lambda_0} = \frac{p_1 + p_0\lambda_0}{p_1 + p_0\lambda_0} = 1. \tag{A.37}$$

The computation of the joint expectation is detailed hereafter

$$\begin{aligned}
E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,f_{\mathcal{E}_r\mathcal{E}_f}\left(\epsilon_r,\,\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&= p^\bullet \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\delta\left(\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&+ p^* \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\,U_{(-\phi_0,0)}\left(\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&+ p^* \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_f\right)\,U_{(0,\phi_0)}\left(\epsilon_r\right)\,d\epsilon_r\,d\epsilon_f \\
&= 0 + 0 + 0 = 0.
\end{aligned} \tag{A.38}$$

The marginal *PDF* of $\mathcal{E}_r$ and $\mathcal{E}_f$ have changed (unlike the case A and all other cases), as well as their expectations and variances. The marginal *PDF* of $\mathcal{E}_r$ is given by

$$\begin{aligned}
f_{\mathcal{E}_r}\left(\epsilon_r\right) &= \int_{-\infty}^{+\infty} f_{\mathcal{E}_r\mathcal{E}_f}\left(\epsilon_r,\,\epsilon_f\right)\,d\epsilon_f \\
&= p^\bullet\delta\left(\epsilon_r\right)\int_{-\infty}^{+\infty}\delta\left(\epsilon_f\right)\,d\epsilon_f \\
&+ p^*\delta\left(\epsilon_r\right)\int_{-\infty}^{+\infty} U_{(-\lambda_0\phi_0,0)}\left(\epsilon_f\right)\,d\epsilon_f \\
&+ p^*\,U_{(0,\lambda_0\phi_0)}\left(\epsilon_r\right)\int_{-\infty}^{+\infty}\delta\left(\epsilon_f\right)\,d\epsilon_f \\
&= \frac{p_1 - p_0\lambda_0}{p_1 + p_0\lambda_0}\,\delta\left(\epsilon_r\right) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,\delta\left(\epsilon_r\right) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,U_{(0,\lambda_0\phi_0)}\left(\epsilon_r\right) \\
&= \frac{p_1}{p_1 + p_0\lambda_0}\,\delta\left(\epsilon_r\right) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,U_{(0,\lambda_0\phi_0)}\left(\epsilon_r\right).
\end{aligned} \tag{A.39}$$

Likewise, the marginal *PDF* of $\mathcal{E}_f$ is given by

$$f_{\mathcal{E}_f}\left(\epsilon_f\right) = \frac{p_1}{p_1 + p_0\lambda_0}\,\delta\left(\epsilon_f\right) + \frac{p_0\lambda_0}{p_1 + p_0\lambda_0}\,U_{(-\lambda_0\phi_0,0)}\left(\epsilon_f\right). \tag{A.40}$$

Their expectations are given by (using result A.7)

$$E\left\{\mathcal{E}_r\right\} = \frac{p_1}{(p_1 + p_0\lambda_0)}\,0 + \frac{p_0\lambda_0}{(p_1 + p_0\lambda_0)}\,\frac{\lambda_0\phi_0}{2} = \frac{p_0\lambda_0}{(p_1 + p_0\lambda_0)}\,\frac{\lambda_0\phi_0}{2} = -E\left\{\mathcal{E}_f\right\}, \tag{A.41}$$

and their variances are given by (using result A.9)

$$\begin{aligned} var\left\{\mathcal{E}_r\right\} &= \frac{p_1}{(p_1+p_0\lambda_0)}\,0+\frac{p_0\lambda_0}{(p_1+p_0\lambda_0)}\,\frac{(\lambda_0\phi_0)^2}{12}+\frac{p_1}{(p_1+p_0\lambda_0)}\,\frac{p_0\lambda_0}{(p_1+p_0\lambda_0)}\,\left(0-\frac{\lambda_0\phi_0}{2}\right)^2 \\ &= \frac{p_0\lambda_0}{(p_1+p_0\lambda_0)}\,\frac{(\lambda_0\phi_0)^2}{12}+\frac{p_1p_0\lambda_0}{(p_1+p_0\lambda_0)^2}\,\frac{(\lambda_0\phi_0)^2}{4} \\ &= \frac{(\lambda_0\phi_0)^2}{12}\,\frac{p_0\lambda_0\left(p_1+p_0\lambda_0\right)+3p_1p_0\lambda_0}{(p_1+p_0\lambda_0)^2} \\ &= \frac{(\lambda_0\phi_0)^2}{12}\,\frac{p_0\lambda_0\left(4p_1+p_0\lambda_0\right)}{(p_1+p_0\lambda_0)^2}=var\left\{\mathcal{E}_f\right\}. \end{aligned} \tag{A.42}$$

Then, we can compute $C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\}$ for the case 0

$$C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} = E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\}-E\left\{\mathcal{E}_r\right\}E\left\{\mathcal{E}_f\right\} \tag{A.43}$$

$$= \left(\frac{p_0\lambda_0}{(p_1+p_0\lambda_0)}\,\frac{\lambda_0\phi_0}{2}\right)^2=\frac{(p_0\lambda_0)^2}{(p_1+p_0\lambda_0)^2}\,\frac{(\lambda_0\phi_0)^2}{4}, \tag{A.44}$$

and, finally, we obtain the variance of Φ_b for the case 0

$$\begin{aligned} \sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r}+C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\}}{2} \\ &= \frac{\left(\frac{(\lambda_0\phi_0)^2}{12}\,\frac{p_0\lambda_0(4p_1+p_0\lambda_0)}{(p_1+p_0\lambda_0)^2}\right)+\left(\frac{(p_0\lambda_0)^2}{(p_1+p_0\lambda_0)^2}\,\frac{(\lambda_0\phi_0)^2}{4}\right)}{2} \\ &= \frac{(\lambda_0\phi_0)^2}{24}\left[\frac{p_0\lambda_0\left(4p_1+p_0\lambda_0\right)+3\left(p_0\lambda_0\right)^2}{(p_1+p_0\lambda_0)^2}\right] \\ &= \frac{(\lambda_0\phi_0)^2}{24}\,\frac{4p_0\lambda_0}{(p_1+p_0\lambda_0)} \\ &= p_0\frac{\phi_0^2}{6}\,\frac{\lambda_0^3}{(p_1+p_0\lambda_0)}=p_0\frac{\phi_0^2}{6}P_0\left(\lambda_0\right). \end{aligned} \tag{A.45}$$

A.3.3 Details for the case D

The different probabilities are computed as follows: we have to measure the total duration for which the different events occur over a code period, and then divide by the code period. The situation is depicted in Figure A.3. The computation of these probabilities is detailed hereafter:

$$\begin{aligned} \phi^{\bullet} &= 2\phi_W-2\left(N_1+N_0\right)\phi_0+N_1\phi_0 \\ &= 2\left[\left(N_1+1\right)\phi_0+\lambda_d\phi_0\right]-N_1\phi_0-2N_0\phi_0 \\ &= N_1\phi_0-N_0\phi_0+N_0\lambda_d\phi_0 \\ &= N_1\phi_0-N_0\phi_0\left(1-\lambda_d\right). \end{aligned} \tag{A.46}$$

$$p^{\bullet}=\frac{\phi^{\bullet}}{\phi_{duration}}=\frac{N_1\phi_0-N_0\phi_0\left(1-\lambda_d\right)}{\left(N_1+N_0\right)\phi_0}=p_1-p_0\left(1-\lambda_d\right). \tag{A.47}$$

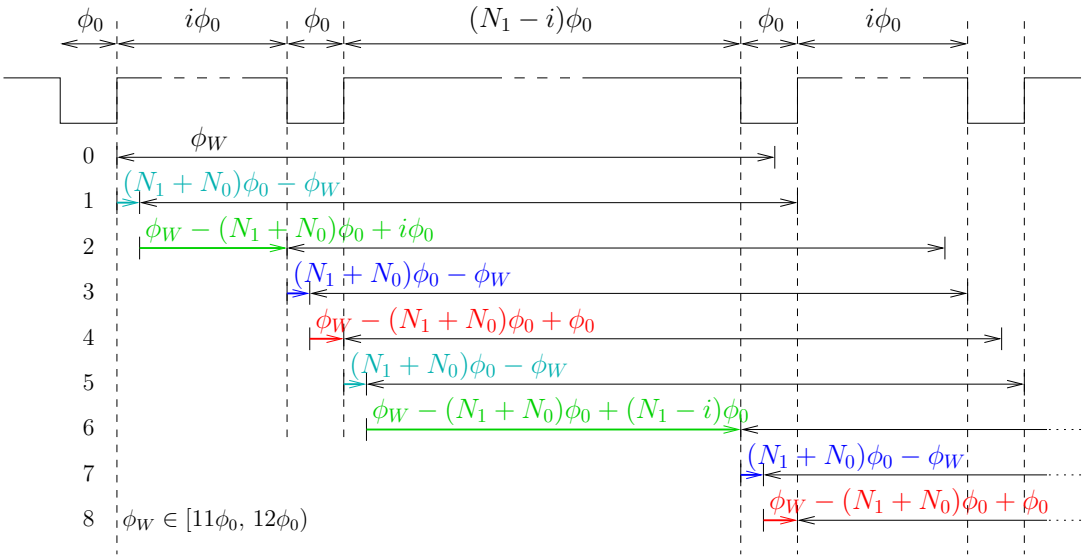

Figure A.3: Computation of the probabilities and intervals associated with the case D, and for all codes ($\lambda_d = 0.6$ in the scheme).

$$\begin{aligned}
\phi^\circ &= 2\phi_W - 2\left(N_1 + N_0\right)\phi_0 + 2\phi_0 \\
&= 2\left[\left(N_1 + 1\right)\phi_0 + \lambda_d\phi_0\right] - 2N_1\phi_0 - N_0\phi_0 \\
&= 2N_1\phi_0 + N_0\phi_0 + N_0\lambda_d\phi_0 - 2N_1\phi_0 - N_0\phi_0 \\
&= N_0\phi_0\lambda_d. \qquad \text{(A.48)}
\end{aligned}$$

$$p^\circ = \frac{\phi^\circ}{\phi_{duration}} = \frac{N_0\phi_0\lambda_d}{\left(N_1 + N_0\right)\phi_0} = p_0\lambda_d. \qquad \text{(A.49)}$$

$$\begin{aligned}
\phi^* &= 2\left(N_1 + N_0\right)\phi_0 - 2\phi_W \\
&= 2\left(N_1 + N_0\right)\phi_0 - 2\left[\left(N_1 + 1\right)\phi_0 + \lambda_d\phi_0\right] \\
&= N_0\phi_0 - N_0\lambda_d\phi_0 \\
&= N_0\phi_0\left(1 - \lambda_d\right). \qquad \text{(A.50)}
\end{aligned}$$

$$p^* = \frac{\phi^*}{\phi_{duration}} = \frac{N_0\phi_0\left(1 - \lambda_d\right)}{\left(N_1 + N_0\right)\phi_0} = p_0\left(1 - \lambda_d\right). \qquad \text{(A.51)}$$

We can check that the probabilities sum up to 1

$$p^\bullet + p^\circ + 2p^* = p_1 - p_0\left(1 - \lambda_d\right) + p_0\lambda_d + 2p_0\left(1 - \lambda_d\right) = p_1 + p_0 = 1. \qquad \text{(A.52)}$$

The joint expectation is derived as follows

$$\begin{aligned}
E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,f_{\mathcal{E}_r\mathcal{E}_f}\left(\epsilon_r,\,\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&= p^{\bullet}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\delta\left(\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\,U_{(-\phi_0,\lambda_d\phi_0)}\left(\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_f\right)\,U_{(\lambda_d\phi_0,\phi_0)}\left(\epsilon_r\right)\,d\epsilon_r\,d\epsilon_f \\
&+ p^{\circ}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,2\,\delta\left(\left(\epsilon_r-\epsilon_f\right)-\lambda_d\phi_0\right)\,U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(\epsilon_r+\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f \\
&= 0+0+0+p^{\circ}\left(-\frac{\left(\lambda_d\phi_0\right)^2}{6}\right) = -p_0\lambda_d\frac{\left(\lambda_d\phi_0\right)^2}{6} = -p_0\lambda_d^3\frac{\phi_0^2}{6}.
\end{aligned} \tag{A.53}$$

The covariance is then given by

$$C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} = E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} - E\left\{\mathcal{E}_r\right\}E\left\{\mathcal{E}_f\right\} = -p_0\lambda_d^3\frac{\phi_0^2}{6} + p_0^2\frac{\phi_0^2}{4}. \tag{A.54}$$

So that we can establish the value of the variance as

$$\begin{aligned}
\sigma_{\Phi_b}^2 &= \frac{\sigma_{\mathcal{E}_r}^2 + C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\}}{2} \\
&= \frac{\left(p_0\frac{\phi_0^2}{3} - p_0^2\frac{\phi_0^2}{4}\right) + \left(-p_0\lambda_d^3\frac{\phi_0^2}{6} + p_0^2\frac{\phi_0^2}{4}\right)}{2} \\
&= \frac{p_0\frac{\phi_0^2}{3} - p_0\lambda_d^3\frac{\phi_0^2}{6}}{2} \\
&= p_0\frac{\phi_0^2}{6}\frac{2-\lambda_d^3}{2} \\
&= p_0\frac{\phi_0^2}{6}\left(1-\frac{\lambda_d^3}{2}\right) = p_0\frac{\phi_0^2}{6}P_D\left(\lambda_d\right).
\end{aligned} \tag{A.55}$$

The presence of the factor 2 in the fourth term of $f_{\mathcal{E}_r\mathcal{E}_f}\left(\epsilon_r,\epsilon_f\right)$ is explained hereafter. This factor is necessary so that the joint *PDF* integrates to 1, as explain below

$$I = \iint_{-\infty}^{+\infty} k\,\delta\left(\left(\epsilon_r-\epsilon_f\right)-\lambda_d\phi_0\right)\,U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(\epsilon_r+\epsilon_f\right)\,d\epsilon_r\,d\epsilon_f = 1. \tag{A.56}$$

To compute this, we apply this change of variables

$$\begin{cases} a = \epsilon_r - \epsilon_f \\ b = \epsilon_r + \epsilon_f \end{cases} \tag{A.57}$$

or, equivalently,

$$\begin{cases} \epsilon_r = \frac{a+b}{2} \\ \epsilon_f = \frac{b-a}{2} \end{cases} \tag{A.58}$$

and the associated Jacobian is equal to

$$J = \begin{vmatrix} \frac{\partial \epsilon_r}{\partial a} & \frac{\partial \epsilon_r}{\partial b} \\ \\ \frac{\partial \epsilon_f}{\partial a} & \frac{\partial \epsilon_f}{\partial b} \end{vmatrix} = \begin{vmatrix} \frac{1}{2} & \frac{1}{2} \\ \\ -\frac{1}{2} & \frac{1}{2} \end{vmatrix} = \frac{1}{4} - \left(-\frac{1}{4}\right) = \frac{1}{2}. \tag{A.59}$$

So, we have

$$\begin{aligned} I &= \iint_{-\infty}^{+\infty} k\,\delta\left(a - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, |J|\, da\, db \\ &= k\, |J| \int_{-\infty}^{+\infty} \delta\left(a - \lambda_d\phi_0\right)\, da \int_{-\infty}^{+\infty} U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, db \\ &= k \times \frac{1}{2} \times 1 \times 1 = \frac{k}{2} \\ &= 1 \Rightarrow k = 2. \end{aligned} \tag{A.60}$$

The computation of the fourth term of $E\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\}$ is detailed hereafter

$$K = \iint_{-\infty}^{+\infty} \epsilon_r\, \epsilon_f\, 2\delta\left(\left(\epsilon_r - \epsilon_f\right) - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(\epsilon_r + \epsilon_f\right)\, d\epsilon_r\, d\epsilon_f. \tag{A.61}$$

By using the same change of variables, we have

$$\begin{aligned} K &= \iint_{-\infty}^{+\infty} \left(\frac{a+b}{2}\right)\left(\frac{b-a}{2}\right) 2\delta\left(a - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, |J|\, da\, db \\ &= \iint_{-\infty}^{+\infty} \left(\frac{b^2 - a^2}{4}\right) \delta\left(a - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, da\, db \\ &= \frac{1}{4} \iint_{-\infty}^{+\infty} b^2 \delta\left(a - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, da\, db \\ &\quad - \frac{1}{4} \iint_{-\infty}^{+\infty} a^2 \delta\left(a - \lambda_d\phi_0\right)\, U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, da\, db \\ &= K_1 - K_2. \end{aligned} \tag{A.62}$$

The first term K_1 is equal to

$$\begin{aligned} K_1 &= \frac{1}{4}\left(\int_{-\infty}^{+\infty} \delta\left(a - \lambda_d\phi_0\right)\, da\right)\left(\int_{-\infty}^{+\infty} b^2 U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, db\right) \\ &= \frac{1}{4}\,(1)\left(\frac{\left(2\lambda_d\phi_0\right)^2}{12}\right) \\ &= \frac{\left(\lambda_d\phi_0\right)^2}{12}. \end{aligned} \tag{A.63}$$

The second term K_2 is equal to

$$\begin{aligned} K_2 &= \frac{1}{4}\left(\int_{-\infty}^{+\infty} a^2 \delta\left(a - \lambda_d\phi_0\right)\, da\right)\left(\int_{-\infty}^{+\infty} U_{(-\lambda_d\phi_0,\lambda_d\phi_0)}\left(b\right)\, db\right) \\ &= \frac{1}{4}\left(\lambda_d\phi_0\right)^2\,(1) \\ &= \frac{\left(\lambda_d\phi_0\right)^2}{4}. \end{aligned} \tag{A.64}$$

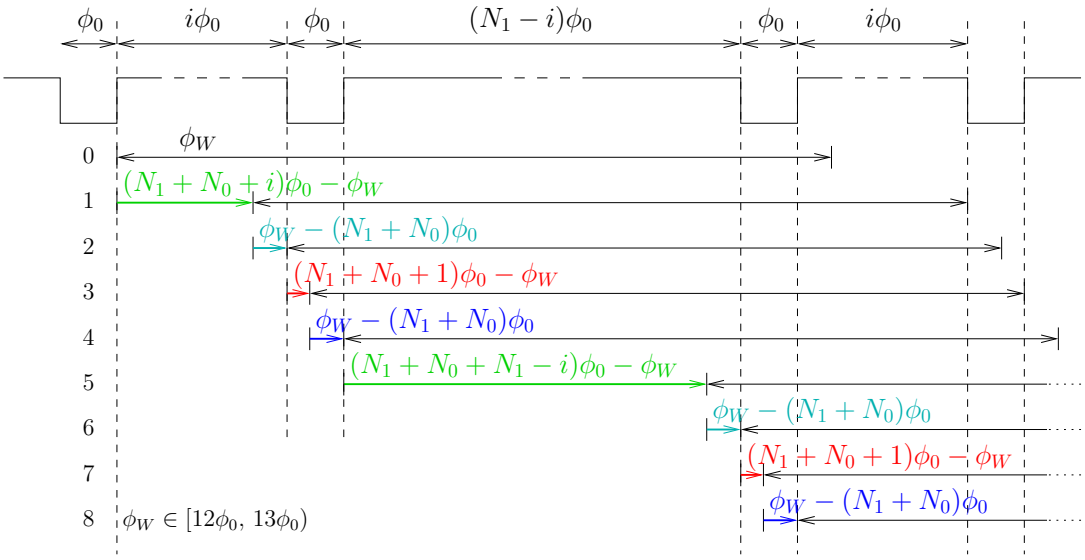

Figure A.4: Computation of the probabilities and intervals associated with the case E, and for all codes ($\lambda_e = 0.6$ in the scheme).

And, finally we have

$$K = K_1 - K_2 = \frac{(\lambda_d \phi_0)^2}{12} - \frac{(\lambda_d \phi_0)^2}{4} = -\frac{(\lambda_d \phi_0)^2}{6} = -\frac{\phi_0^2}{6}\lambda_d^2. \tag{A.65}$$

A.3.4 Details for the case E

The different probabilities are computed as follows: we have to measure the total duration for which the different events occur over a code period, and then divide by the code period. The situation is depicted in Figure A.4. The computation of these probabilities is detailed hereafter:

$$\begin{aligned} \phi^\bullet &= (2N_1 + 2N_0 + N_1)\,\phi_0 - 2\phi_W \\ &= 3N_1\phi_0 + 2N_0\phi_0 - 2\left[(N_1 + N_0)\,\phi_0 + \lambda_e\phi_0\right] \\ &= 3N_1\phi_0 + 2N_0\phi_0 - 2N_1\phi_0 - 2N_0\phi_0 - 2\lambda_e\phi_0 \\ &= N_1\phi_0 - N_0\phi_0\lambda_e. \end{aligned} \tag{A.66}$$

$$p^\bullet = \frac{\phi^\bullet}{\phi_{duration}} = \frac{N_1\phi_0 - N_0\phi_0\lambda_e}{(N_1 + N_0)\,\phi_0} = p_1 - p_0\lambda_e. \tag{A.67}$$

$$\begin{aligned} \phi^\circ &= (2N_1 + 2N_0 + 2)\,\phi_0 - 2\phi_W \\ &= 2N_1\phi_0 + 3N_0\phi_0 - 2\left[(N_1 + N_0)\,\phi_0 + \lambda_e\phi_0\right] \\ &= N_0\phi_0 - N_0\lambda_e\phi_0 \\ &= N_0\phi_0\,(1 - \lambda_e)\,. \end{aligned} \tag{A.68}$$

$$p^\circ = \frac{\phi^\circ}{\phi_{duration}} = \frac{N_0\phi_0\,(1 - \lambda_e)}{(N_1 + N_0)\,\phi_0} = p_0\,(1 - \lambda_e)\,. \tag{A.69}$$

$$\begin{aligned}\phi^* &= 2\phi_W - 2\left(N_1+N_0\right)\phi_0 \\ &= 2\left[\left(N_1+N_0\right)\phi_0 + \lambda_e\phi_0\right] - 2\left(N_1+N_0\right)\phi_0 \\ &= 2\lambda_e\phi_0 \\ &= N_0\phi_0\lambda_e. \end{aligned} \tag{A.70}$$

$$p^* = \frac{\phi^*}{\phi_{duration}} = \frac{N_0\phi_0\lambda_e}{\left(N_1+N_0\right)\phi_0} = p_0\lambda_e. \tag{A.71}$$

We can check that the probabilities sum up to 1

$$p^\bullet + p^\circ + 2p^* = p_1 - p_0\lambda_e + p_0\left(1-\lambda_e\right) + 2p_0\lambda_e = p_1 + p_0 = 1. \tag{A.72}$$

The joint expectation is derived as follows

$$\begin{aligned} E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\, f_{\mathcal{E}_r\mathcal{E}_f}\left(\epsilon_r,\,\epsilon_f\right)\, d\epsilon_r\, d\epsilon_f \\ &= p^\bullet \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\delta\left(\epsilon_f\right)\, d\epsilon_r\, d\epsilon_f \\ &+ p^* \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_r\right)\, U_{(-\lambda_e\phi_0,0)}\left(\epsilon_f\right)\, d\epsilon_r\, d\epsilon_f \\ &+ p^* \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta\left(\epsilon_f\right)\, U_{(0,\lambda_e\phi_0)}\left(\epsilon_r\right)\, d\epsilon_r\, d\epsilon_f \\ &+ p^\circ \iint_{-\infty}^{+\infty} \epsilon_r\epsilon_f 2\delta\left(\left(\epsilon_r-\epsilon_f\right)-\left(1+\lambda_e\right)\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}\left(\epsilon_r+\epsilon_f\right) d\epsilon_r d\epsilon_f \\ &= 0+0+0+p^\circ\left(-\frac{\phi_0^2}{6}\left(1+4\lambda_e+\lambda_e^2\right)\right) \\ &= -p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right). \end{aligned} \tag{A.73}$$

This leads to the expression of the covariance

$$C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} = E\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\} - E\left\{\mathcal{E}_r\right\}E\left\{\mathcal{E}_f\right\} = -p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right) + p_0^2\frac{\phi_0^2}{4}. \tag{A.74}$$

The variance is then given by

$$\begin{aligned}\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\left\{\mathcal{E}_r,\,\mathcal{E}_f\right\}}{2} \\ &= \frac{\left(p_0\frac{\phi_0^2}{3} - p_0^2\frac{\phi_0^2}{4}\right) + \left(-p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right) + p_0^2\frac{\phi_0^2}{4}\right)}{2} \\ &= \frac{p_0\frac{\phi_0^2}{3} - p_0\frac{\phi_0^2}{6}\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right)}{2} \\ &= p_0\frac{\phi_0^2}{6}\left[\frac{2-\left(1-\lambda_e\right)\left(1+4\lambda_e+\lambda_e^2\right)}{2}\right] \\ &= p_0\frac{\phi_0^2}{6}\left(\frac{1-3\lambda_e+3\lambda_e^2+\lambda_e^3}{2}\right) = p_0\frac{\phi_0^2}{6}P_E\left(\lambda_e\right). \end{aligned} \tag{A.75}$$

The explanation about the presence of the factor 2 in the fourth term of $f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r, \epsilon_f)$ is the same as for the case D. The computation of the fourth term of $E\{\mathcal{E}_r, \mathcal{E}_f\}$ gives

$$K = \iint_{-\infty}^{+\infty} \epsilon_r\, \epsilon_f\, 2\delta\left((\epsilon_r - \epsilon_f) - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(\epsilon_r + \epsilon_f)\; d\epsilon_r\, d\epsilon_f. \quad \text{(A.76)}$$

By using the same change of variables (equation (A.58)), we have

$$\begin{aligned} K &= \iint_{-\infty}^{+\infty} \left(\frac{a+b}{2}\right)\left(\frac{b-a}{2}\right) 2\delta\left(a - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; |J|\; da\, db \\ &= \iint_{-\infty}^{+\infty} \left(\frac{b^2-a^2}{4}\right) \delta\left(a - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; da\, db \\ &= \frac{1}{4}\iint_{-\infty}^{+\infty} b^2 \delta\left(a - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; da\, db \\ &\quad - \frac{1}{4}\iint_{-\infty}^{+\infty} a^2 \delta\left(a - (1+\lambda_e)\,\phi_0\right) U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; da\, db \\ &= K_1 - K_2. \end{aligned} \quad \text{(A.77)}$$

The first term K_1 is equal to

$$\begin{aligned} K_1 &= \frac{1}{4}\left(\int_{-\infty}^{+\infty} \delta\left(a - (1+\lambda_e)\,\phi_0\right)\; da\right)\left(\int_{-\infty}^{+\infty} b^2 U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; db\right) \\ &= \frac{1}{4}\;(1)\left(\frac{\left(2\,(1-\lambda_e)\,\phi_0\right)^2}{12}\right) \\ &= \frac{\left((1-\lambda_e)\,\phi_0\right)^2}{12}. \end{aligned} \quad \text{(A.78)}$$

The second term K_2 is equal to

$$\begin{aligned} K_2 &= \frac{1}{4}\left(\int_{-\infty}^{+\infty} a^2 \delta\left(a - (1+\lambda_e)\,\phi_0\right)\; da\right)\left(\int_{-\infty}^{+\infty} U_{(-(1-\lambda_e)\phi_0,(1-\lambda_e)\phi_0)}(b)\; db\right) \\ &= \frac{1}{4}\;\left((1+\lambda_e)\,\phi_0\right)^2\;(1) \\ &= \frac{\left((1+\lambda_e)\,\phi_0\right)^2}{4}. \end{aligned} \quad \text{(A.79)}$$

And, finally, we have

$$\begin{aligned} K = K_1 - K_2 &= \frac{\left((1-\lambda_e)\,\phi_0\right)^2}{12} - \frac{\left((1+\lambda_e)\,\phi_0\right)^2}{4} \\ &= \frac{\phi_0^2}{12}\left[(1-\lambda_e)^2 - 3\,(1+\lambda_e)^2\right] \\ &= \frac{\phi_0^2}{12}\left(-2 - 8\lambda_e - 2\lambda_e^2\right) \\ &= -\frac{\phi_0^2}{6}\left(1 + 4\lambda_e + \lambda_e^2\right). \end{aligned} \quad \text{(A.80)}$$

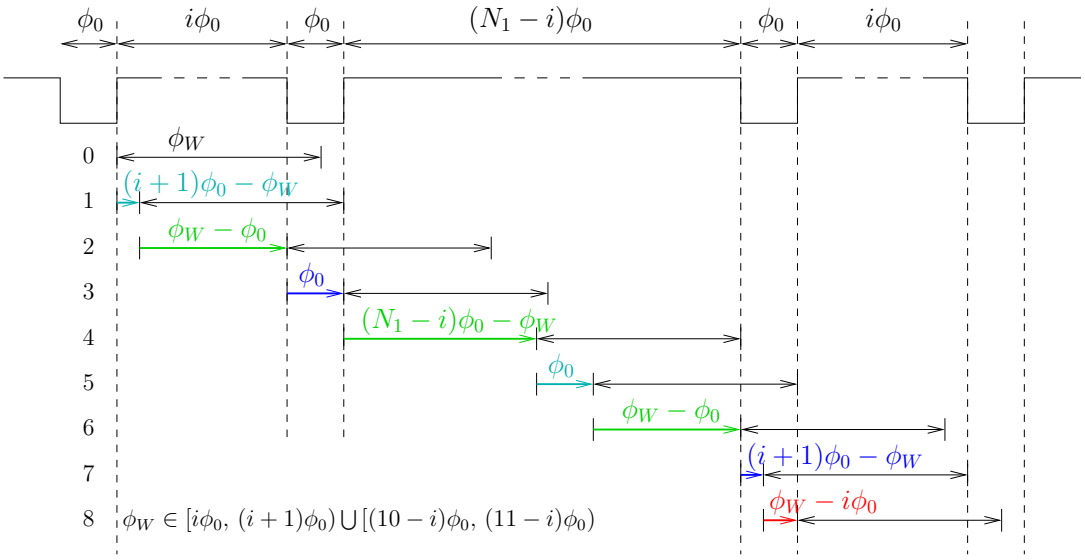

Figure A.5: Computation of the probabilities and intervals associated with the case B, and for all codes, except the code 5 ($\lambda_b = 0.6$ in the scheme).

A.3.5 Details for the case B

The different probabilities are computed as follows: we have to measure the total duration for which the different events occur over a code period, and then divide by the code period. The situation is depicted in Figure A.5. The computation of these probabilities is detailed hereafter:

$$\begin{aligned}\phi^\bullet &= \phi_W - \phi_0 + (N_1 - i)\,\phi_0 - \phi_W + \phi_W - \phi_0 \\ &= \phi_W + (N_1 - i - 2)\,\phi_0 \\ &= (i\phi_0 + \lambda_b\phi_0) + (N_1 - i - N_0)\,\phi_0 \\ &= N_1\phi_0 - \frac{N_0}{2}\phi_0\,(2 - \lambda_b)\,. \end{aligned} \tag{A.81}$$

$$p^\bullet = \frac{\phi^\bullet}{\phi_{duration}} = \frac{N_1\phi_0 - \frac{N_0}{2}\phi_0\,(2-\lambda_b)}{(N_1+N_0)\,\phi_0} = p_1 - \frac{p_0}{2}\,(2-\lambda_b)\,. \tag{A.82}$$

$$\begin{aligned}\phi^\circ &= \phi_W - i\phi_0 \\ &= (i\phi_0 + \lambda_b\phi_0) - i\phi_0 \\ &= \frac{N_0}{2}\phi_0\lambda_b. \end{aligned} \tag{A.83}$$

$$p^\circ = \frac{\phi^\circ}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0\lambda_b}{(N_1+N_0)\,\phi_0} = \frac{p_0}{2}\lambda_b. \tag{A.84}$$

$$\begin{aligned}\phi^\star &= \phi_0 \\ &= \frac{N_0}{2}\phi_0. \end{aligned} \tag{A.85}$$

$$p^{\star} = \frac{\phi^{\star}}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0}{(N_1 + N_0)\,\phi_0} = \frac{p_0}{2}. \tag{A.86}$$

$$\begin{aligned} \phi^{*} &= (i+1)\,\phi_0 - \phi_W \\ &= (i+1)\,\phi_0 - (i\phi_0 + \lambda_b\phi_0) \\ &= \phi_0 - \lambda_b\phi_0 \\ &= \frac{N_0}{2}\phi_0\,(1-\lambda_b)\,. \end{aligned} \tag{A.87}$$

$$p^{*} = \frac{\phi^{*}}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0\,(1-\lambda_b)}{(N_1 + N_0)\,\phi_0} = \frac{p_0}{2}\,(1-\lambda_b)\,. \tag{A.88}$$

We can check that the probabilities sum up to 1

$$p^{\bullet} + p^{\circ} + 2p^{\star} + 2p^{*} = p_1 - \frac{p_0}{2}\,(2-\lambda_b) + \frac{p_0}{2}\lambda_b + 2\frac{p_0}{2} + 2\frac{p_0}{2}\,(1-\lambda_b) = p_1 + p_0 = 1. \tag{A.89}$$

The joint expectation is expressed as

$$\begin{aligned} E\{\mathcal{E}_r,\,\mathcal{E}_f\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\,\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\ &= p^{\bullet}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\,\delta(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\ &+ p^{\star}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\;U_{(-\phi_0,0)}(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\ &+ p^{\star}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_f)\;U_{(0,\phi_0)}(\epsilon_r)\;d\epsilon_r\,d\epsilon_f \\ &+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\;U_{(-\phi_0,\lambda_b\phi_0)}(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\ &+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_f)\;U_{(\lambda_b\phi_0,\phi_0)}(\epsilon_r)\;d\epsilon_r\,d\epsilon_f \\ &+ p^{\circ}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,2\,\delta((\epsilon_r-\epsilon_f)-\lambda_b\phi_0)\;U_{(-\lambda_b\phi_0,\lambda_b\phi_0)}(\epsilon_r+\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\ &= 0+0+0+0+0+p^{\circ}\left(-\frac{(\lambda_b\phi_0)^2}{6}\right) \\ &= -\frac{p_0}{2}\lambda_b\frac{(\lambda_b\phi_0)^2}{6} = -p_0\lambda_b^3\frac{\phi_0^2}{12}. \end{aligned} \tag{A.90}$$

Then we derive the covariance of $\mathcal{E}_r$ and $\mathcal{E}_f$

$$C\{\mathcal{E}_r,\,\mathcal{E}_f\} = E\{\mathcal{E}_r,\,\mathcal{E}_f\} - E\{\mathcal{E}_r\}\,E\{\mathcal{E}_f\} = -p_0\lambda_b^3\frac{\phi_0^2}{12} + p_0^2\frac{\phi_0^2}{4}. \tag{A.91}$$

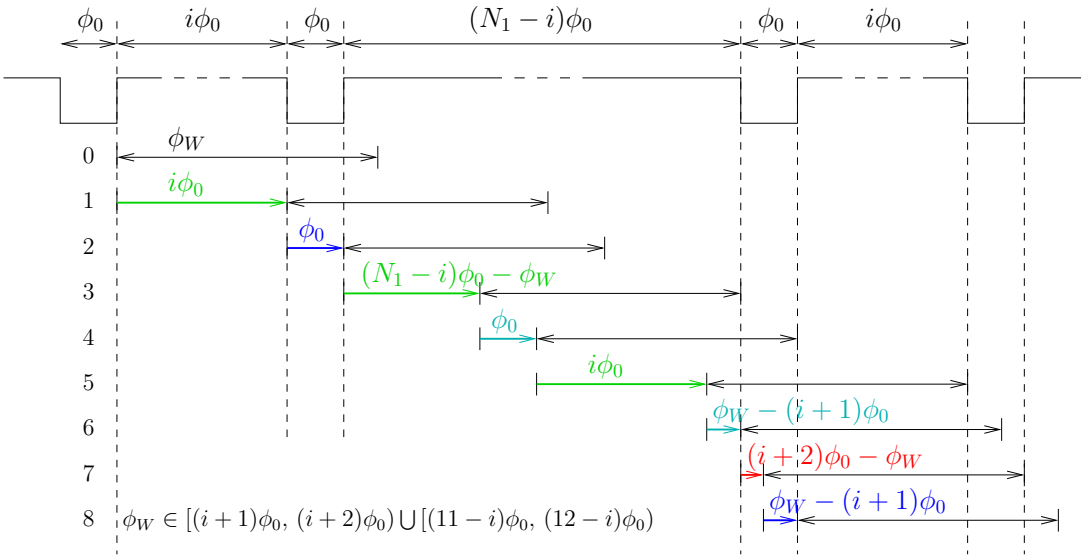

Figure A.6: Computation of the probabilities and intervals associated with the case C, and for all codes, except the code 5 ($\lambda_c = 0.6$ in the scheme).

This leads to the following expression of the variance:

$$\begin{aligned}
\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\left\{\mathcal{E}_r,\, \mathcal{E}_f\right\}}{2} \\
&= \frac{\left(p_0 \frac{\phi_0^2}{3} - p_0^2 \frac{\phi_0^2}{4}\right) + \left(-p_0 \lambda_b^3 \frac{\phi_0^2}{12} + p_0^2 \frac{\phi_0^2}{4}\right)}{2} \\
&= \frac{p_0 \frac{\phi_0^2}{3} - p_0 \lambda_b^3 \frac{\phi_0^2}{12}}{2} \\
&= p_0 \frac{\phi_0^2}{6}\left(1 - \frac{\lambda_b^3}{4}\right) = p_0 \frac{\phi_0^2}{6} P_B\left(\lambda_b\right). \qquad \text{(A.92)}
\end{aligned}$$

A.3.6 Details for the case C

The different probabilities are computed as follows: we have to measure the total duration for which the different events occur over a code period, and then divide by the code period. The situation is depicted in Figure A.6. The computation of these probabilities is detailed hereafter:

$$\begin{aligned}
\phi^\bullet &= i\phi_0 + \left(N_1 - i\right)\phi_0 - \phi_W + i\phi_0 \\
&= N_1\phi_0 + i\phi_0 - \phi_W \\
&= N_1\phi_0 + i\phi_0 - \left(\left(i+1\right)\phi_0 + \lambda_c\phi_0\right) \\
&= N_1\phi_0 - \frac{N_0}{2}\phi_0\left(1 + \lambda_c\right). \qquad \text{(A.93)}
\end{aligned}$$

$$p^\bullet = \frac{\phi^\bullet}{\phi_{duration}} = \frac{N_1\phi_0 - \frac{N_0}{2}\phi_0\left(1+\lambda_c\right)}{\left(N_1 + N_0\right)\phi_0} = p_1 - \frac{p_0}{2}\left(1 + \lambda_c\right). \qquad \text{(A.94)}$$

$$\begin{aligned}\phi^{\circ} &= (i+2)\,\phi_0 - \phi_W \\ &= i\phi_0 + 2\phi_0 - ((i+1)\,\phi_0 + \lambda_c\phi_0) \\ &= \phi_0 - \lambda_c\phi_0 \\ &= \frac{N_0}{2}\phi_0\,(1-\lambda_c)\,. \end{aligned} \tag{A.95}$$

$$p^{\circ} = \frac{\phi^{\circ}}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0\,(1-\lambda_c)}{(N_1+N_0)\,\phi_0} = \frac{p_0}{2}\,(1-\lambda_c)\,. \tag{A.96}$$

$$\begin{aligned}\phi^{\star} &= \phi_0 \\ &= \frac{N_0}{2}\phi_0. \end{aligned} \tag{A.97}$$

$$p^{\star} = \frac{\phi^{\star}}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0}{(N_1+N_0)\,\phi_0} = \frac{p_0}{2}. \tag{A.98}$$

$$\begin{aligned}\phi^{*} &= \phi_W - (i+1)\,\phi_0 \\ &= ((i+1)\,\phi_0 + \lambda_c\phi_0) - (i+1)\,\phi_0 \\ &= \lambda_c\phi_0 \\ &= \frac{N_0}{2}\phi_0\lambda_c. \end{aligned} \tag{A.99}$$

$$p^{*} = \frac{\phi^{*}}{\phi_{duration}} = \frac{\frac{N_0}{2}\phi_0\lambda_c}{(N_1+N_0)\,\phi_0} = \frac{p_0}{2}\lambda_c. \tag{A.100}$$

We can check that the probabilities sum up to 1

$$p^{\bullet} + p^{\circ} + 2p^{\star} + 2p^{*} = p_1 - \frac{p_0}{2}\,(1+\lambda_c) + \frac{p_0}{2}\,(1-\lambda_c) + 2\frac{p_0}{2} + 2\frac{p_0}{2}\lambda_c = p_1 + p_0 = 1. \tag{A.101}$$

The joint expectation is expressed as

$$\begin{aligned}
E\{\mathcal{E}_r,\,\mathcal{E}_f\} &= \iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,f_{\mathcal{E}_r\mathcal{E}_f}(\epsilon_r,\,\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\
&= p^{\bullet}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\,\delta(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\
&+ p^{\star}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\;U_{(-\phi_0,0)}(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\
&+ p^{\star}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_f)\;U_{(0,\phi_0)}(\epsilon_r)\;d\epsilon_r\,d\epsilon_f \\
&+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_r)\;U_{(-\lambda_c\phi_0,0)}(\epsilon_f)\;d\epsilon_r\,d\epsilon_f \\
&+ p^{*}\iint_{-\infty}^{+\infty} \epsilon_r\,\epsilon_f\,\delta(\epsilon_f)\;U_{(0,\lambda_c\phi_0)}(\epsilon_r)\;d\epsilon_r\,d\epsilon_f \\
&+ p^{\circ}\iint_{-\infty}^{+\infty} \epsilon_r\epsilon_f 2\delta\left((\epsilon_r-\epsilon_f)-(1+\lambda_c)\,\phi_0\right)U_{(-(1-\lambda_c)\phi_0,(1-\lambda_c)\phi_0)}(\epsilon_r+\epsilon_f)\,d\epsilon_r d\epsilon_f \\
&= 0+0+0+0+0+p^{\circ}\left(-\frac{\phi_0^2}{6}\left(1+4\lambda_c+\lambda_c^2\right)\right) \\
&= -\frac{p_0}{2}(1-\lambda_c)\,\frac{\phi_0^2}{6}\left(1+4\lambda_c+\lambda_c^2\right) \\
&= -p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)\,(1+4\lambda_c+\lambda_c^2)}{2}. \qquad \text{(A.102)}
\end{aligned}$$

Then we derive the covariance of $\mathcal{E}_r$ and $\mathcal{E}_f$

$$C\{\mathcal{E}_r,\,\mathcal{E}_f\} = E\{\mathcal{E}_r,\,\mathcal{E}_f\} - E\{\mathcal{E}_r\}\,E\{\mathcal{E}_f\} = -p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)\,(1+4\lambda_c+\lambda_c^2)}{2} + p_0^2\frac{\phi_0^2}{4}, \quad \text{(A.103)}$$

and, finally, the variance:

$$\begin{aligned}
\sigma^2_{\Phi_b} &= \frac{\sigma^2_{\mathcal{E}_r} + C\{\mathcal{E}_r,\,\mathcal{E}_f\}}{2} \\
&= \frac{\left(p_0\frac{\phi_0^2}{3} - p_0^2\frac{\phi_0^2}{4}\right) + \left(-p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)\left(1+4\lambda_c+\lambda_c^2\right)}{2} + p_0^2\frac{\phi_0^2}{4}\right)}{2} \\
&= \frac{p_0\frac{\phi_0^2}{3} - p_0\frac{\phi_0^2}{6}\frac{(1-\lambda_c)\left(1+4\lambda_c+\lambda_c^2\right)}{2}}{2} \\
&= p_0\frac{\phi_0^2}{6}\left(\frac{2-\frac{(1-\lambda_c)\left(1+4\lambda_c+\lambda_c^2\right)}{2}}{2}\right) \\
&= p_0\frac{\phi_0^2}{6}\left(\frac{4-1-4\lambda_c-\lambda_c^2+\lambda_c+4\lambda_c^2+\lambda_c^3}{4}\right) \\
&= p_0\frac{\phi_0^2}{6}\left(\frac{3-3\lambda_c+3\lambda_c^2+\lambda_c^3}{4}\right) = p_0\frac{\phi_0^2}{6}P_C(\lambda_c). \qquad \text{(A.104)}
\end{aligned}$$

A.4 Local Minima of the variance of Φ_b

The local minima are encountered in $P_C(\lambda_c)$ and $P_E(\lambda_e)$ (see Figure 4.9 to Figure 4.13). For $P_C(\lambda_c)$, we have

$$\frac{\partial P_C(\lambda_c)}{\partial \lambda_c} = \frac{-3 + 6\lambda_c + 3\lambda_c^2}{4} = 0 \tag{A.105}$$

$$\Rightarrow \lambda_c^2 + 2\lambda_c - 1 = 0 \tag{A.106}$$

$$\Rightarrow \lambda_c = \sqrt{2} - 1. \tag{A.107}$$

The second solution $(-\sqrt{2} - 1)$ is rejected since it is not comprised in the interval $[0, 1)$. The local minimum for $P_C(\lambda_c)$ is equal to

$$P_C\left(\sqrt{2} - 1\right) = 2 - \sqrt{2} \simeq 0.5858. \tag{A.108}$$

For $P_E(\lambda_e)$, we have to note first that

$$P_E(\lambda_e) = 2P_C(\lambda_e) - 1. \tag{A.109}$$

Therefore the local minimum of $P_E(\lambda_e)$ is also encountered for

$$\lambda_e = \sqrt{2} - 1, \tag{A.110}$$

and the corresponding value of $P_E(\lambda_e)$ is equal to

$$P_E\left(\sqrt{2} - 1\right) = 2\left(2 - \sqrt{2}\right) - 1 = 3 - 2\sqrt{2} \simeq 0.1716. \tag{A.111}$$

Appendix B

Theoretical developments related to the triangulation algorithm

B.1 The circle equation

The circle equation is derived entirely hereafter

$$\begin{aligned}
\arg\left\{\frac{B_2-R}{B_1-R}\right\} &= \phi_{12} && \text{(B.1)}\\
\Rightarrow \arg\left\{(B_2-R)\,\overline{(B_1-R)}\right\} &= \phi_{12} && \text{(B.2)}\\
\Rightarrow \arg\left\{(x_2+iy_2-x-iy)\,(x_1-iy_1-x+iy)\,e^{-i\phi_{12}}\right\} &= 0 && \text{(B.3)}\\
\Rightarrow \Im\left\{(x_2+iy_2-x-iy)\,(x_1-iy_1-x+iy)\,(\cos\phi_{12}-i\,\sin\phi_{12})\right\} &= 0 && \text{(B.4)}\\
\Rightarrow -\sin\phi_{12}\,(x_2-x)\,(x_1-x)+\sin\phi_{12}\,(y_2-y)\,(y-y_1) & && \\
+\cos\phi_{12}\,(x_2-x)\,(y-y_1)+\cos\phi_{12}\,(y_2-y)\,(x_1-x) &= 0 && \text{(B.5)}\\
\Rightarrow -(x_2-x)\,(x_1-x)+(y_2-y)\,(y-y_1) & && \\
+\cot\phi_{12}\,(x_2-x)\,(y-y_1)+\cot\phi_{12}\,(y_2-y)\,(x_1-x) &= 0 && \text{(B.6)}\\
\Rightarrow (x-x_1)\,(x-x_2)+(y-y_1)\,(y-y_2) & && \\
+T_{12}\,(x-x_2)\,(y-y_1)-T_{12}\,(x-x_1)\,(y-y_2) &= 0 && \text{(B.7)}\\
\Rightarrow x^2-x\,[x_1+x_2+T_{12}\,(y_1-y_2)] & && \\
+y^2-y\,[y_1+y_2-T_{12}\,(x_1-x_2)] & && \\
+x_1x_2+y_1y_2+T_{12}x_2y_1-T_{12}x_1y_2 &= 0 && \text{(B.8)}\\
\Rightarrow x^2-x\,(2x_{12})+x_{12}^2+y^2-y\,(2y_{12})+y_{12}^2-x_{12}^2-y_{12}^2+2k_{12} &= 0 && \text{(B.9)}\\
\Rightarrow (x-x_{12})^2+(y-y_{12})^2 &= R_{12}^2 && \text{(B.10)}
\end{aligned}$$

Note that we divided by $\sin\phi_{12}$ to obtain expression B.6. Therefore, $\sin\phi_{12}$ has to be different from zero. In other terms, it means that we must have $\phi_{12} \neq k\pi,\ k \in \mathbb{Z}$. Otherwise, the circle degenerates as the B_1B_2 line (infinite radius and center coordinates). Also, we introduced the notation $T_{12} = \cot\phi_{12}$, and a quantity denoted k_{12}

$$k_{12} = \frac{x_1x_2+y_1y_2+T_{12}x_2y_1-T_{12}x_1y_2}{2}. \qquad \text{(B.11)}$$

The circle center coordinates $\{x_{12},\, y_{12}\}$ are

$$x_{12} = \frac{(x_1 + x_2) + T_{12}\,(y_1 - y_2)}{2}, \tag{B.12}$$

$$y_{12} = \frac{(y_1 + y_2) - T_{12}\,(x_1 - x_2)}{2}, \tag{B.13}$$

and the squared radius equals

$$\begin{aligned}
R_{12}^2 &= x_{12}^2 + y_{12}^2 - 2k_{12} && \text{(B.14)}\\
&= x_{12}^2 + y_{12}^2 - (x_1x_2 + y_1y_2 + T_{12}x_2y_1 - T_{12}x_1y_2) && \text{(B.15)}\\
&= \frac{(x_1 + x_2)^2 + T_{12}^2\,(y_1 - y_2)^2 + 2T_{12}\,(x_1 + x_2)\,(y_1 - y_2)}{4} && \text{(B.16)}\\
&+ \frac{(y_1 + y_2)^2 + T_{12}^2\,(x_1 - x_2)^2 - 2T_{12}\,(y_1 + y_2)\,(x_1 - x_2)}{4} && \text{(B.17)}\\
&- \frac{4x_1x_2 + 4y_1y_2 + 4T_{12}x_2y_1 - 4T_{12}x_1y_2}{4} && \text{(B.18)}\\
&= \frac{(x_1 + x_2)^2 - 4x_1x_2 + T_{12}^2\,(y_1 - y_2)^2}{4} && \text{(B.19)}\\
&+ \frac{(y_1 + y_2)^2 - 4y_1y_2 + T_{12}^2\,(x_1 - x_2)^2}{4} && \text{(B.20)}\\
&= \frac{(x_1 - x_2)^2 + T_{12}^2\,(y_1 - y_2)^2}{4} + \frac{(y_1 - y_2)^2 + T_{12}^2\,(x_1 - x_2)^2}{4} && \text{(B.21)}\\
&= \frac{(x_1 - x_2)^2\,(1 + T_{12}^2) + (y_1 - y_2)^2\,(1 + T_{12}^2)}{4} && \text{(B.22)}\\
&= \frac{(x_1 - x_2)^2 + (y_1 - y_2)^2}{4\,\sin^2\phi_{12}}, && \text{(B.23)}
\end{aligned}$$

where we used the following trigonometric result at line B.22

$$1 + T_{12}^2 = 1 + \cot^2\phi_{12} = \frac{1}{\sin^2\phi_{12}}. \tag{B.24}$$

B.2 The parameter k_{ij}

The computation of the parameter k_{ij} works as follows. One has to replace the expressions of x_{ij} and y_{ij} (equations (B.25) and (B.26)), and the expression of R_{ij}^2 (equation (B.27)) into the expression of k_{ij} (equation (B.28))

$$x_{ij} = \frac{(x_i + x_j) + T_{ij}\,(y_i - y_j)}{2}, \tag{B.25}$$

$$y_{ij} = \frac{(y_i + y_j) - T_{ij}\,(x_i - x_j)}{2}, \tag{B.26}$$

$$R_{ij}^2 = \frac{(x_i - x_j)^2 + (y_i - y_j)^2}{4\,\sin^2\phi_{ij}}, \tag{B.27}$$

$$k_{ij} = \frac{x_{ij}^2 + y_{ij}^2 - R_{ij}^2}{2}. \tag{B.28}$$

However, the computation is straightforward if we reformulate equation (B.14), and if we note that it is equivalent to equation (B.28)

$$\begin{aligned} R_{12}^2 &= x_{12}^2 + y_{12}^2 - 2k_{12} && \text{(B.29)} \\ \Rightarrow k_{12} &= \frac{x_{12}^2 + y_{12}^2 - R_{12}^2}{2} && \text{(B.30)} \\ &= \frac{x_1x_2 + y_1y_2 + T_{12}x_2y_1 - T_{12}x_1y_2}{2} && \text{(B.31)} \\ &= \frac{x_1x_2 + y_1y_2 + T_{12}\left(x_2y_1 - x_1y_2\right)}{2}, && \text{(B.32)} \end{aligned}$$

which has already been introduced in expression B.11, when computing the circle parameters. Finally, we can generalize the previous result as

$$k_{ij} = \frac{x_ix_j + y_iy_j + T_{ij}(x_jy_i - x_iy_j)}{2}. \tag{B.33}$$

B.3 Position sensitivity analysis

In this section, we detail the sensitivity analysis of the computed position. The coordinates of the robot position are reminded hereafter

$$\begin{aligned} x_R &= x_2 + \frac{k'_{31}(y'_{12} - y'_{23})}{D}, && \text{(B.34)} \\ y_R &= y_2 + \frac{k'_{31}(x'_{23} - x'_{12})}{D}. && \text{(B.35)} \end{aligned}$$

We start by computing the derivative of x_R with respect to the first angle ϕ_1

$$\begin{aligned} \frac{\partial x_R}{\partial \phi_1} &= \frac{\partial k'_{31}}{\partial \phi_1}\frac{(y'_{12} - y'_{23})}{D} + \frac{\partial (y'_{12} - y'_{23})}{\partial \phi_1}\frac{k'_{31}}{D} + \frac{\partial\left(\frac{1}{D}\right)}{\partial \phi_1}k'_{31}(y'_{12} - y'_{23}) && \text{(B.36)} \\ &= \frac{\partial k'_{31}}{\partial \phi_1}\frac{(y'_{12} - y'_{23})}{D} + \frac{\partial (y'_{12} - y'_{23})}{\partial \phi_1}\frac{k'_{31}}{D} - \frac{1}{D^2}\frac{\partial (D)}{\partial \phi_1}k'_{31}(y'_{12} - y'_{23}) && \text{(B.37)} \\ &= \frac{1}{D}\,g_1(.), && \text{(B.38)} \end{aligned}$$

where $g_1(.)$ is some function of all the other parameters. Similar results yield for the derivative of x_R with respect to the second and third angles, ϕ_2 and ϕ_3 respectively

$$\frac{\partial x_R}{\partial \phi_2} = \frac{1}{D}\,g_2(.), \tag{B.39}$$

$$\frac{\partial x_R}{\partial \phi_3} = \frac{1}{D}\,g_3(.), \tag{B.40}$$

where $g_2(.)$ and $g_3(.)$ are some functions of all the other parameters. The total differential of x_R with respect to ϕ_1, ϕ_2, and ϕ_3 is given by

$$\begin{aligned} \Delta x_R &= \frac{\partial x_R}{\partial \phi_1}\Delta\phi_1 + \frac{\partial x_R}{\partial \phi_2}\Delta\phi_2 + \frac{\partial x_R}{\partial \phi_3}\Delta\phi_3 && \text{(B.41)} \\ &= \frac{1}{D}\,\Delta\phi\,\left(g_1(.) + g_2(.) + g_3(.)\right) && \text{(B.42)} \\ &= \frac{1}{D}\,\Delta\phi\,g(.), && \text{(B.43)} \end{aligned}$$

where we assumed that the three infinitesimal increments are equal $\Delta\phi = \Delta\phi_1 = \Delta\phi_2 = \Delta\phi_3$. A similar result yields for the total differential of y_R

$$\Delta y_R = \frac{1}{D}\,\Delta\phi\, h(.) \tag{B.44}$$

where $h(.)$ is some function of all the other parameters. Finally, we can compute Δd_R as follows

$$\begin{aligned} \Delta d_R &= \sqrt{(\Delta x_R)^2 + (\Delta y_R)^2} && \text{(B.45)} \\ &= \frac{1}{|D|}\,|\Delta\phi|\,\sqrt{(g(.))^2 + (h(.))^2} && \text{(B.46)} \\ &= \frac{1}{|D|}\,|\Delta\phi|\,f(.) && \text{(B.47)} \end{aligned}$$

where $f(.)$ is some function of all the other parameters.

List of publications

Articles in journals

1. V. Pierlot and M. Van Droogenbroeck. **BeAMS: a Beacon based Angle Measurement Sensor for mobile robot positioning.** Submitted for publication in IEEE Transactions on Robotics.

2. V. Pierlot and M. Van Droogenbroeck. **A New Three Object Triangulation Algorithm for Mobile Robot Positioning.** Submitted for publication in IEEE Transactions on Robotics.

Conference articles

1. V. Pierlot and M. Van Droogenbroeck. **Analysis of a robot positioning system based on a rotating receiver, beacons, and coded signals.** In Proceedings of the European Signal Processing Conference (EUSIPCO), Barcelona, Spain, pages 1766-1770, August 2011.

2. V. Pierlot, M. Van Droogenbroeck, and M. Urbin-Choffray. **A new three object triangulation algorithm based on the power center of three circles.** In Research and Education in Robotics (EUROBOT), volume 161 of Communications in Computer and Information Science, pages 248-262, 2011. Springer.

3. V. Pierlot, P. Tassin, and M. Van Droogenbroeck. **A new precise ultrasonic range sensor based on the emission of two coded FSK signals combined to a ping-pong strategy.** In Conference on Research in Information and communication Technology, Veldhoven, The Netherlands, pages 36-39, November 2010.

4. V. Pierlot and M. Van Droogenbroeck. **A simple and low cost angle measurement system for mobile robot positioning.** In Workshop on Circuits, Systems and Signal Processing (ProRISC), Veldhoven, The Netherlands, pages 251-254, November 2009.

5. S. Piérard, V. Pierlot, A. Lejeune, and M. Van Droogenbroeck. **I-see-3D! An Interactive and Immersive System that dynamically adapts 2D projections to the location of a user's eyes.** In International Conference on 3D Imaging (IC3D), Liège, Belgium, December 2012.

6. S. Piérard, V. Pierlot, O. Barnich, and M. Van Droogenbroeck, and J. Verly. **A platform for the fast interpretation of movements and localization of users in 3D applications driven by a range camera.** In 3DTV Conference, Tampere, Finland, June 2010.

Internal reports

1. V. Pierlot and M. Van Droogenbroeck. **Statistical analysis of modulated codes for robot positioning - Application to BeAMS.** Technical report, University of Liège, Belgium, May 2013.

Bibliography

[1] E. Aitenbichler and M. Mühlhäuser. An IR local positioning system for smart items and devices. In *IEEE International Conference on Distributed Computing Systems Workshops*, pages 334–339, Providence, RI, USA, May 2003. [cited on page 6]

[2] T.A. Alhmiedat and S.-H. Yang. A survey: Localization and tracking mobile targets through wireless sensors network. In *Postgraduate Symposium on The Convergence of Telecommunications, Networking and Broadcasting*, Liverpool, United Kingdom, June 2007. [cited on page 6]

[3] I. Amundson and X.D. Koutsoukos. A survey on localization for mobile wireless sensor networks. In *International conference on Mobile entity localization and tracking in GPS-less environments*, pages 235–254, Orlando, FL, USA, 2009. Springer-Verlag. [cited on page 6]

[4] Y. Arai and M. Sekiai. Absolute position measurement system for mobile robot based on incident angle detection of infrared light. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 986–991, Las Vegas, NV, USA, October 2003. [cited on page 8]

[5] A. Argyros, K. Bekris, and S. Orphanoudakis. Robot homing based on corner tracking in a sequence of panoramic images. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3–10, Hawaii, USA, December 2001. [cited on page 2]

[6] L. Armesto and J. Tornero. Robust and efficient mobile robot self-localization using laser scanner and geometrical maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3080–3085, Beijing, China, October 2006. [cited on page 2]

[7] D. Avis and H. Imai. Locating a robot with angle measurements. *Journal of Symbolic Computation*, 10:311–326, August 1990. [cited on page 84]

[8] P. Bahl and V.N. Padmanabhan. RADAR: an in-building RF-based user location and tracking system. In *Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 775–784, Tel Aviv, Israel, March 2000. [cited on page 6]

[9] A. Bais, R. Sablatnig, and J. Gu. Single landmark based self-localization of mobile robots. In *Third Canadian Conference on Computer and Robot Vision (CRV)*. IEEE Computer Society, June 2006. [cited on page 84]

[10] G. Bauzil, M. Briot, and P. Ribes. A navigation subsystem using ultrasonic sensors for the mobile robot HILARE. In *First International Conference on Robot Vision and Sensory Controls*, pages 47–58 & 681–698, Stratford-upon-Avon, U.K, April 1981. [cited on page 7]

[11] K.E. Bekris, A.A. Argyros, and L.E. Kavraki. Angle-based methods for mobile robot navigation: Reaching the entire plane. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2373–2378, New Orleans, LA, USA, April 2004. [cited on page 6]

[12] Y.J. Beliveau, J.E. Fithian, and M.P. Deisenroth. Autonomous vehicle navigation with real-time 3D laser based positioning for construction. *Automation in Construction*, 5(4):261–272, October 1996. [cited on page 7]

[13] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13(2):251–263, April 1997. [cited on pages 1, 2, 6, 8, 81, 84, 85, and 103]

[14] J. Bisson, F. Michaud, and D. Letourneau. Relative positioning of mobile robots using ultrasounds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1783–1788, Las Vegas, NV, USA, October 2003. [cited on page 6]

[15] J. Borenstein, H. Everett, and L. Feng. Where am I? Systems and methods for mobile robot positioning. Technical report, University of Michigan, March 1996. [cited on pages 1, 2, 6, and 87]

[16] J. Borenstein, H. Everett, L. Feng, and D. Wehe. Mobile robot positioning - sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, April 1997. [cited on pages 1, 2, 6, 10, and 87]

[17] J. Borenstein and L. Feng. UMBmark: A benchmark test for measuring odometry errors in mobile robots. In *Society of Photo-Instrumentation Engineers*, volume 1001, pages 113–124, Philadelphia, Pennsylvania, USA, October 1995. [cited on pages 2 and 3]

[18] K. Briechle and U. Hanebeck. Localization of a mobile robot using relative bearing measurements. *IEEE Transactions on Robotics and Automation*, 20(1):36–44, February 2004. [cited on pages 1, 84, and 85]

[19] M. Brkić, M. Lukić, J. Bajić, B. Dakić, and M. Vukadinović. Hardware realization of autonomous robot localization system. In *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 146–150, Opatija, Croatia, May 2012. [cited on pages 9, 10, 77, and 83]

[20] R. Burtch. *Three point resection problem*, chapter 8, pages 175–201. Surveying Engineering Department, Ferris State University, surveying computations course notes 2005/2006 edition, 2005. [cited on pages 83, 84, and 99]

[21] C. Cohen and F. Koss. A comprehensive study of three object triangulation. In *Mobile Robots VII*, volume 1831, pages 95–106, Boston, MA, USA, November 1992. Society of Photo-Instrumentation Engineers. [cited on pages 6, 67, 81, 82, 83, 84, 85, 99, and 103]

[22] J.L. Crowley. Mathematical foundations of navigation and perception for an autonomous mobile robot. Technical report, I.N.P. Grenoble, November 1995. [cited on page 2]

[23] E. Demaine, A. López-Ortiz, and J. Munro. Robot localization without depth perception. In *Proceedings of Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 2368 of *Lecture Notes in Computer Science*, pages 177–194. Springer, July 2002. [cited on pages 25 and 86]

[24] G.A. Demetriou. A survey of sensors for localization of unmanned ground vehicles (UGVs). In *International Conference on Artificial Intelligence (ICAI)*, volume 2, pages 659–668, Las Vegas, NV, USA, June 2006. CSREA Press. [cited on page 1]

[25] R. D'Errico, M. Bottazzi, F. Natali, E. Savioli, S. Bartoletti, A. Conti, D. Dardari, N. Decarli, F. Guidi, F. Dehmas, L. Ouvry, U. Alvarado, N. Hadaschik, C. Franke, Z. Mhanna, M. Sacko, Y. Wei, and A. Sibille. An UWB-UHF semi-passive RFID system for localization and tracking applications. In *IEEE International Conference on RFID-Technologies and Applications (RFID-TA)*, pages 18–23, Nice, France, November 2012. [cited on page 6]

[26] O. Devise and C. Doncarli. Dynamic localization of a mobile robot: a data fusion approach for a statistical filtering problem. *Traitement du signal*, 10(4):311–318, 1993. [cited on page 2]

[27] V. Durst, D. Hagel, J. Vander, M. Blaich, and O. Bittel. Designing an omni-directional infrared sensor and beacon system for the eurobot competition. In *Research and Education in Robotics - EUROBOT 2011*, volume 161, pages 102–113, Prague, Czech Republic, June 2011. Springer. [cited on page 8]

[28] A. Easton and S. Cameron. A gaussian error model for triangulation-based pose estimation using noisy landmarks. In *IEEE Conference on Robotics, Automation and Mechatronics*, pages 1–6, Bangkok, Thailand, June 2006. [cited on pages 6, 67, 83, 84, 85, 94, 99, and 115]

[29] J.-D. Eiden. *Géometrie analytique classique*. Calvage & Mounet, 2009. [cited on page 90]

[30] J. Esteves, A. Carvalho, and C. Couto. Generalized geometric triangulation algorithm for mobile robot absolute self-localization. In *International Symposium on Industrial Electronics (ISIE)*, volume 1, pages 346–351, Rio de Janeiro, Brazil, June 2003. [cited on pages 3, 81, 83, and 94]

[31] J. Esteves, A. Carvalho, and C. Couto. An improved version of the generalized geometric triangulation algorithm. In *European-Latin-American Workshop on Engineering Systems*, Porto, Portugal, July 2006. [cited on page 83]

[32] J. Esteves, A. Carvalho, and C. Couto. Position and orientation errors in mobile robot absolute self-localization using an improved version of the generalized geometric triangulation algorithm. In *IEEE International Conference on Industrial Technology (ICIT)*, pages 830–835, Mumbai, India, December 2006. [cited on pages 1, 3, 6, 67, 75, 83, 84, 85, 94, 99, and 103]

[33] J.M. Font-Llagunes and J.A. Batlle. Localization of a mobile robot with omnidirectional wheels using angular kalman filtering and triangulation. In *ASME Conference on Engineering Systems Design and Analysis (ESDA)*, volume 1, pages 713–720, Torino, Italy, July 2006. [cited on page 2]

[34] J.M. Font-Llagunes and J.A. Batlle. Consistent triangulation for mobile robot localization using discontinuous angular measurements. *Robotics and Autonomous Systems*, 57(9):931–942, September 2009. [cited on pages 1, 2, 3, 6, 83, 88, 89, 92, 94, and 99]

[35] J.M. Font-Llagunes and J.A. Batlle. New method that solves the three-point resection problem using straight lines intersection. *Journal of Surveying Engineering*, 135(2):39–45, May 2009. [cited on pages 83, 84, and 99]

[36] S.S. Ghidary, T. Tani, T. Takamori, and M. Hattori. A new home robot positioning system (HRPS) using IR switched multi ultrasonic sensors. In *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, volume 4, pages 737–741, Tokyo, Japan, October 1999. [cited on page 6]

[37] P. Goel, S.I. Roumeliotis, and G.S. Sukhatme. Robust localization using relative and absolute position estimates. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1134–1140, October 1999. [cited on page 2]

[38] Y. Gu, A. Lo, and I. Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys & Tutorials*, 11(1):13–32, First Quarter 2009. [cited on pages 1 and 6]

[39] Á. Gutiérrez, A. Campo, M. Dorigo, D. Amor, L. Magdalena, and F. Monasterio-Huelin. An open localization and local communication embodied sensor. *Sensors*, 8(11):7545–7563, November 2008. [cited on page 8]

[40] Á. Gutiérrez, A. Campo, M. Dorigo, J. Donate, F. Monasterio-Huelin, and L. Magdalena. Open e-puck range & bearing miniaturized board for local communication in swarm robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3111–3116, Kobe, Japan, May 2009. [cited on page 8]

[41] S. Hernández, J. Torres, C. Morales, and L. Acosta. A new low cost system for autonomous robot heading and position localization in a closed area. *Autonomous Robots*, 15(2):99–110, September 2003. [cited on page 8]

[42] H. Hmam. Mobile platform self-localization. In *Information, Decision and Control*, pages 242–247, Adelaide, Qld, Australia, February 2007. [cited on pages 83 and 99]

[43] H. Hu and D. Gu. Landmark-based navigation of industrial mobile robots. *International Journal of Industry Robot*, 27(6):458–467, 2000. [cited on pages 1, 7, and 85]

[44] G. Jang, S. Kim, J. Kim, and I. Kweon. Metric localization using a single artificial landmark for indoor mobile robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3407–3412, August 2005. [cited on pages 2, 6, and 8]

[45] A. Kelly. Precision dilution in triangulation based mobile robot position estimation. *Intelligent Autonomous Systems*, 8:1046–1053, 2003. [cited on pages 85, 86, and 94]

[46] I. Kelly and A. Martinoli. A scalable, on-board localisation and communication system for indoor multi-robot experiments. *Sensor Review*, 24(2):167–180, January 2004. [cited on page 8]

[47] A. Kemppainen, J. Haverinen, and J. Röning. An infrared location system for relative pose estimation of robots. In *Symposium of Robot Design, Dynamics, and Control*, pages 379–386, Warsaw, Poland, June 2006. [cited on pages 9 and 77]

[48] L. Kleeman. Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead-reckoning. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2582–2587, May 1992. [cited on page 2]

[49] D. Kortenkamp. Perception for mobile robot navigation: A survey of the state of the art. Technical Report 19960022619, NASA, May 1994. [cited on page 83]

[50] M. Kushwaha, K. Molnar, J. Sallai, P. Volgyesi, M. Maroti, and A. Ledeczi. Sensor node localization using mobile acoustic beacons. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference (MASS)*, pages 491–499, Washington, DC, USA, November 2005. [cited on page 6]

[51] C. Lee, Y. Chang, G. Park, J. Ryu, S.-G. Jeong, S. Park, J.W. Park, H.C. Lee, K.-S. Hong, and M.H. Lee. Indoor positioning system based on incident angles of infrared emitters. In *Conference of the IEEE Industrial Electronics Society (IECON)*, volume 3, pages 2218–2222, Busan, South Korea, November 2004. [cited on pages 8, 84, and 87]

[52] M. Ligas. Simple solution to the three point resection problem. *Journal of Surveying Engineering*, 139(3):120–125, August 2013. [cited on pages 84 and 99]

[53] I. Loevsky and I. Shimshoni. Reliable and efficient landmark-based localization for mobile robots. *Robotics and Autonomous Systems*, 58(5):520–528, May 2010. [cited on pages 6, 7, 67, 78, 81, 86, 102, 111, 112, 116, and 127]

[54] M. Lukić, B. Miodrag, and J. Bajić. An autonomous robot localization system based on coded infrared beacons. In *Research and Education in Robotics - EUROBOT 2011*, volume 161, pages 202–209, Prague, Czech Republic, June 2011. Springer. [cited on page 83]

[55] C.B. Madsen and C.S. Andersen. Optimal landmark selection for triangulation of robot position. *Robotics and Autonomous Systems*, 23(4):277–292, July 1998. [cited on pages 6, 83, 86, 87, 94, and 99]

[56] C.B. Madsen, C.S. Andersen, and J.S. Sorensen. A robustness analysis of triangulation-based robot self-positioning. In Christensen, Henrik I. (ed.), editor, *International Symposium on Intelligent Robotic Systems*, pages 195–204, Stockholm, Sweden, July 1997. [cited on pages 85 and 86]

[57] J. Maneesilp, C. Wang, H. Wu, and N.-F. Tzeng. RFID support for accurate 3D localization. *IEEE Transactions on Computers*, 62(7):1447–1459, July 2013. [cited on page 6]

[58] C.D. McGillem and T.S. Rappaport. Infra-red location system for navigation of autonomous vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1236–1238, Philadelphia, PA, USA, April 1988. [cited on page 83]

[59] C.D. McGillem and T.S. Rappaport. A beacon navigation method for autonomous vehicles. *IEEE Transactions on Vehicular Technology*, 38(3):132–139, August 1989. [cited on pages 6, 9, 83, 87, 94, and 99]

[60] K. Muthukrishnan, M. Lijding, and P. Havinga. Towards smart surroundings: Enabling techniques and technologies for localization. In T. Strang and C. Linnhoff-Popien, editors, *International Workshop on Location- and Context-Awareness*, volume 3479 of *Lecture Notes in Computer Science*, pages 350–362, Oberpfaffenhofen, Germany, May 2005. [cited on pages 1, 6, and 10]

[61] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AoA. In *INFOCOM Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1734–1743, San Franciso, CA, USA, March 2003. [cited on page 3]

[62] T. Nishizawa, A. Ohya, and S. Yuta. An implementation of on-board position estimation for a mobile robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 395–400, Nagoya, Japan, May 1995. [cited on page 8]

[63] A. Papoulis. *Probability, random variables, and stochastic processes.* McGraw-Hill, February 1991. [cited on pages 30, 32, 33, and 131]

[64] T. Pfeifer and D. Elias. Commercial hybrid IR/RF local positioning system. In *Kommunikation in Verteilten Systemen*, pages 119–127, Leipzig, Germany, February 2003. [cited on page 6]

[65] V. Pierlot and M. Van Droogenbroeck. A simple and low cost angle measurement system for mobile robot positioning. In *Workshop on Circuits, Systems and Signal Processing (ProRISC)*, pages 251–254, Veldhoven, The Netherlands, November 2009. [cited on pages 4 and 87]

[66] V. Pierlot and M. Van Droogenbroeck. Analysis of a robot positioning system based on a rotating receiver, beacons, and coded signals. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 1766–1770, Barcelona, Spain, August 2011. [cited on pages 4 and 33]

[67] V. Pierlot, M. Van Droogenbroeck, and M. Urbin-Choffray. A new three object triangulation algorithm based on the power center of three circles. In *Research and Education in Robotics (EUROBOT)*, volume 161 of *Communications in Computer and Information Science*, pages 248–262. Springer, 2011. [cited on pages 3, 4, 6, 75, 84, and 103]

[68] J.M. Porta and F. Thomas. Simple solution to the three point resection problem. *Journal of Surveying Engineering*, 135(4):170–172, November 2009. [cited on pages 84 and 99]

[69] K.S. Premi and C.B. Besant. A review of various vehicle guidance techiques that can be used by mobile robots or AGVs. In *2nd International Conference on Automated Guided Vehicle Systems*, Stuttgart, Germany, June 1983. [cited on pages 6 and 7]

[70] N.B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *International conference on Mobile computing and networking*, pages 32–43, Boston, MA, United States, 2000. ACM. [cited on page 6]

[71] J. Pugh and A. Martinoli. Relative localization and communication module for small-scale multi-robot systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 188–193, May 2006. [cited on page 8]

[72] J. Pugh, X. Raemy, C. Favre, R. Falconi, and A. Martinoli. A fast onboard relative positioning module for multirobot systems. *IEEE/ASME Transactions on Mechatronics*, 14(2):151–162, April 2009. [cited on page 8]

[73] J.F. Roberts, T.S. Stirling, J.-C. Zufferey, and D. Floreano. 2.5D infrared range and bearing system for collective robotics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3659–3664, St. Louis, MO, USA, October 2009. [cited on pages 8 and 130]

[74] J.M. Sanchiz, J. Badenas, and F. Pla. Control system and laser-based sensor design of an automonous vehicle for industrial environments. *Society of Photo-Instrumentation Engineers*, 5422(1):608–615, 2004. [cited on pages 2 and 85]

[75] I. Shimshoni. On mobile robot localization from landmark bearings. *IEEE Transactions on Robotics and Automation*, 18(6):971–976, December 2002. [cited on pages 1, 3, 6, 81, 84, 85, 102, and 103]

[76] S. Shoval and D. Sinriech. Analysis of landmark configuration for absolute positioning of autonomous vehicles. *Journal of Manufacturing Systems*, 20(1):44–54, 2001. [cited on pages 86 and 94]

[77] A. Siadat and S. Vialle. Robot localization, using p-similar landmarks, optimized triangulation and parallel programming. In *IEEE International Symposium on Signal Processing and Information Technology*, Marrakesh, Morocco, December 2002. [cited on page 84]

[78] D. Sinriech and S. Shoval. Landmark configuration for absolute positioning of autonomous vehicles. *IIE Transactions*, 32(7):613–624, July 2000. [cited on pages 25 and 86]

[79] H. Sobreira, A.P. Moreira, and J. Esteves. Characterization of position and orientation measurement uncertainties in a low-cost mobile platform. In *Portuguese Conference on Automatic Control (CONTROLO)*, pages 635–640, Coimbra, Portugal, September 2010. [cited on page 84]

[80] H. Sobreira, A.P. Moreira, and J. Esteves. Low cost self-localization system with two beacons. In *International Conference on Mobile Robots and Competitions (ROBOTICA)*, pages 73–77, Leiria, Portugal, March 2010. [cited on pages 8 and 84]

[81] O. Tekdas and V. Isler. Sensor placement algorithms for triangulation based localization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4448–4453, Roma, Italy, April 2007. [cited on page 86]

[82] O. Tekdas and V. Isler. Sensor placement for triangulation-based localization. *IEEE Transactions on Automation Science and Engineering*, 7(3):681–685, July 2010. [cited on pages 25 and 86]

[83] F. Thomas and L. Ros. Revisiting trilateration for robot localization. *IEEE Transactions on Robotics*, 21(1):93–101, February 2005. [cited on page 3]

[84] T. Tsukiyama. Mobile robot localization from landmark bearings. In *World Congress on Fundamental and Applied Metrology*, pages 2109–2112, Lisbon, Portugal, September 2009. [cited on pages 83 and 99]

[85] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, January 1992. [cited on page 6]

[86] Y. Yamamoto, P. Pirjanian, J. Brown, M. Munich, E. DiBernardo, L. Goncalves, J. Ostrowski, and N. Karlsson. Optical sensing for robot perception and localization. In *IEEE Workshop on Advanced Robotics and its Social Impacts*, pages 14–17, June 2005. [cited on page 8]

[87] J. Yun, S. Kim, and J. Lee. Robust positioning a mobile robot with active beacon sensors. In *Knowledge-Based Intelligent Information and Engineering Systems*, volume 4251 of *Lecture Notes in Computer Science*, pages 890–897. Springer, 2006. [cited on page 2]

[88] E. Zalama, S. Dominguez, J. Gómez, and J.R. Perán. A new beacon-based system for the localization of moving objects. In *IEEE International Conference on Mechatronics and Machine Vision in Practice*, Chiang Mai, Tailand, September 2002. [cited on pages 83, 87, and 99]

[89] E. Zalama, S. Dominguez, J. Gómez, and J.R. Perán. Microcontroller based system for 2D localisation. *Mechatronics*, 15(9):1109–1126, November 2005. [cited on pages 6, 7, 10, 77, 81, and 83]