UNIVERSITÉ LIBRE DE BRUXELLES

Brussels Faculty of Engineering

Vrije Universiteit Brussel

High-accuracy localization of robotic platforms with ultra-wideband wireless transceivers

Quentin Fesler

Master thesis submitted under the supervision of
Prof. Dr. Ir. Francois Quitin

The co-supervision of
Dr. Ir. Michel Osée

In order to be awarded the Master's Degree in
Electromechanical Engineering

Academic year
2017-2018

High-accuracy localization of robotic platforms with ultra-wideband wireless transceivers

Quentin Fesler
Master of Science in Electromechanical Engineering
2017-2018

This paper shows the development and implementation of an indoor Real-time Locating System (RTLS) using ultra-wideband (UWB) transceivers. This system is planned to be installed on an autonomous robotic platform. The RTLS is based on distance measurement between a tag and beacons using the Symmetric Double-sided Two-way Ranging (sds-twr) technique. The trilateration of those measurements gives the location of the equipped robot. In order to evaluate the chosen algorithm, its performances are compared with the Cramer-Rao Lower Bound (CRLB) obtained from the error characterisation of the sds-twr. Finally, a Kalman filter is applied to the system in order to increase even more its accuracy. The final system shows a very good accuracy when moving on a straight line but shows a loss of precision when the robot is changing direction. Trying to implement an Extended Kalman filter showed that the system needed additional information such as the robot inputs or state measurements to lower this precision loss.

Keyword Real-time locating system, Ultra-wideband, Kalman filter

Contents

List of Figures

Acknowledgement

I would like to thank prof. Francois Quitin as my supervisor and Michel Osée as my co-supervisor for giving me the opportunity to make my thesis on this subject.

I want to express my gratitude to the members of the *bULBot* robotics club at the ULB for letting me participate in their activities and be a member of this great family.

Thanks to my fellow students for all these years spent together.

And, finally, thanks to my family for everything they did for me. My parents for giving me all the opportunities and for the moral support, my brothers for being there when I had to make a break and do something else and a special big thank to my girlfriend for everything she did during the past years.

Quentin Fesler
May 2018

Motivation

Nowadays, the development of autonomous robots and artificial intelligence is in a real expansion. This comes with a lot of different challenges and problematics. The main challenge discussed in this work consist of the robot capability to locate himself in the environment. The attract of this technology is that, not only it can be used in the scope of an autonomous robot situation, but there is a lot of other fields where this could be useful such as, by example, the tracking of assets in a wide area. As this technology is relatively new, it does not currently exist a lot of commercially available solutions and they are generally overpriced.

With this work, I want, not only to contribute to the subject, but I also want to prove that the development of a cheap but accurate localisation system is possible with all the new technologies available.

Goal

The work of this thesis focuses on the design of a localisation system using ultra-wideband communication. This system is intended to be implemented on an autonomous robot in order to ease its displacements. The main goal of this thesis is to assess the accuracy of such a system and to increase it using filtering algorithms.

To give a context on this work, the system is aimed to be used during the *Eurobot* robotics cup as the localisation system. Optionally, this system could also be mounted on the opponent's robot during this competition to allow tracking and collisions avoidance.

Outline

This work starts in chapter 1 with a brief subject introduction. The chapter 2 discusses the characteristics of a real-time location system. Then the different options available when designing such system are presented. The first phase of the development is detailed in chapter 3. This phase focuses on simple one to one measurements and on accuracy optimisation. The error characterisation on these measurements is then used in chapter 4 to compute the theoretical best accuracy for the system. The location algorithm is then presented and its performances are compared with the best theoretical accuracy. Finally, system accuracy is even more improved with the use of a filtering algorithm in chapter 5. Chapter 6 brings conclusions to this work along with ideas for future improvements.

Chapter 1

Introduction

In the early 40', when Isaac Asimov began to write science fiction novels about autonomous robots, everything seemed unrealistic. However, nowadays, more and more studies are conducted toward the development of such technology. Creating technological beings which will be able to almost live by themselves as presented in Asimov's writing comes with a lot of different problematics to solve. One of these problematics, and it will be the subject of this work, is the capability for the robot to locate himself in his environment.

Imagine you want to know where you are. You can look around you, trying to see landmarks or other clues which can help you know your location. You can even take your smartphone and find, with the GPS, your -almost- exact location on earth. More difficult now, imagine you need to go to buy some groceries at the supermarket. How do you get there? That is still very easy you say. Now imagine the same but you are blind, deaf and you don't feel anything. Almost impossible? Yet, this is the situation of most actual autonomous robots.

Self-localisation in space is something which needs a lot of data acquisition through a lot of different sensors. And with each sensor comes a cost in money, in energy or in computational capacity. This is one of the reasons why development of cheap but accurate locating systems has been one of the main concerns for a lot of scientists in the past few years.

This work will focus on the implementation of what is called a "Real Time Locating System" or RTLS on an autonomous robot. This RTLS should allow the robot to know at any time where it is located in a defined area with the best possible accuracy. This implementation can be decomposed in several distinct steps. For each of these steps, this work will detail the technique applied along with an evaluation of the system performance.

Thanks to the cooperation of the bULBot robotics club from ULB, a real situation testing will be realised during the participation of their robot to the *Eurobot* robotics cup. This cup gathers several robots from different university to compete in 1v1 timed match. During each match, the robot is asked to fulfil a defined number of tasks on the arena. Each robot is evaluated based on its task completion. The RTLS of this work will be used here to help the robot locate himself in the arena and to help him go to the location needed for task completion.

As the *bULBot* robot is already equipped with its own locating system, it's results will be able to be compared with this work's system allowing a comparative performance evaluation.

Chapter 2

State of the art : Real Time Locating Systems (RTLS) nowadays

2.1 Definition of RTLS

Real-time locating systems (RTLS) are defined as systems allowing the tracking and localisation of objects and/or persons in real or near-real-time[10][8]. These systems use wireless communication between devices at known locations (named *beacons* in this work) and a *tag* associated with the object or the person to locate.

This localisation can be divided into two major phases. At first, each beacon communicates individually with the tag. Based on this communication and its physical properties, a "relative" position such as a distance or an angle is measured between the tag and the beacon.

During the second phase, the measurements from each beacon are gathered and the "absolute" location of the tag is computed. This second phase occurs only at software level in contrast to the first phases which relies mostly on the physical layer of communication.

The most obvious example of RTLS in our daily life is the GPS system used in cars and phones where the satellites are the beacons and the car (or rather the GPS antenna in the car) is the tag.

Depending on the application and the wanted result, the absolute position computation can either be done by the tag or by a software connected to the beacons. When the tag does the computation, as it is the case with GPS, the user use it to know its location. On the other side, beacon level computation is often used to locate goods in big storage areas.

It exists several ways to either get the relative location between tags and beacons or to compute these data into an absolute position. The next sections present a non-exhaustive list of ways to realise this.

2.2 Different types of RTLS - Relative position

Nowadays, there exists a lot of different techniques allowing to measure distances between two communicating objects. And each of these techniques can be realised using a different type of beacon-tag communication having their own pros and cons. These can be achieved using sound, Bluetooth, light, Wi-Fi, GPS, Ultra Wideband (UWB) and a lot more[11, 17].

Of course, not all of these techniques are equivalent. They will be selected depending on the situation, the accuracy needed, the power consumption and so on.

2.2.1 Line-of-sight/proximity detection

One of the simplest ways to locate approximately a device, without implying any kind of calculation is the line-of-sight detection. The idea is to see if the tag is in range (in line of sight) of a beacon. By knowing in range of which beacon the tag is located, the location of the tag can be approximated. This can be a quite costly way to locate an object as the precision of the localisation depends on the area coverage and the amount of the beacons.

As an example of this, the worldwide coverage with mobile phone antenna can be considered as such a system. If every antenna has a defined ID, every phone user can know it's approximate position on the world depending on which antenna it is connected to.

Another more frequent way to use this type of locating system is to put the beacons at choke points; some locations the tag is forced to go through while moving. By knowing if the tag has passed at defined choke points, it's approximate position can be deduced.

This system is actually developed to be installed in hospitals to track the medical equipment and even patients allowing a gain of efficiency[10]. By putting beacons on choke points only, there is a lesser of them to cover the area. However, this system needs to memorise all the previous choke points traversed by the tag to deduce it's approximate position. This is then more a near-real-time locating system rather than a real-time one.

To illustrate the choke point system, on can think of the way luggage travels from one airport to another. At each crucial step, they are scanned so in the case of a loss, it is easier to find it by looking the last choke point it was registered.

The main advantage of this technology is its low energy price. As the tags can be passive ones, there are no batteries needed inside. Moreover, the price of the tags themselves is really low, making this technique very attractive when the user wants to track a lot of items.

2.2.2 Received signal strength detection

Very similar to the previous technology, received signal strength detection implies a few more calculation to work. The idea is to measure the distance between the tag and the beacon by knowing at which rate the signal strength decrease with the distance. In theory, this technology can give very good results with low technological cost[20]. However, it is not well suited for indoor localisation since reflections and presences of walls create interference leading to the signal strength not being directly proportional to the distance between the tag and the beacon. To counteract this, a calibration phase can be realised: the signal strength is "mapped" according to the floor plan on the area where the localisation occurs[9].

The advantage of such a system is that it can use communications already installed within most of the buildings around the world. Indeed, by using the Wi-Fi coverage of such building and mapping it during a calibration phase, one can locate every device connected to this network without the need to invest in a whole dedicated hardware installation.

2.2.3 Angle of arrival detection

The angle of arrival detection is a relative position measurement technology which focuses on detecting the propagation direction of a wave signal[14]. In most cases, this is done by analysing the difference of signal's time of arrival on an array of antenna[14]. Despite its great precision on open fields, this technology is very sensitive to multipath issues and shadowing - with the presence of obstacles between the beacon and the tag for example and it makes this technology not widely used for indoor positioning. Moreover, although this technology needs only two beacons to compute the absolute position of the tag (see section 2.3.2) these devices are more complex and then more costly than the ones used with other technologies.

2.2.4 Time of flight detection

The last way to compute the relative location of the tag is to rely on the time of flight of a message between tag and beacon. Knowing the time the message was sent and the time it is received, a distance can be measured by multiplying this time difference with the signal propagation speed. Ultimately, this technology can be used in several ways to compute a relative tag's location. Multiple methods of relative location do exist. They differ on the implementation. Some of them require the synchronisation of the clocks on both the tag and the beacon. Some are more complex and avoid such synchronisation needs.

Time of arrival

The "easiest" way to know the travel time of the message is to compute it's time of arrival relatively to it's sending time. It is called time of arrival detection and it is what is used for GPS localisation. This system is not well suited for very precise localisation because this method requires clock synchronisation between every tag and beacons to get a precise time measurement. The more precise the synchronisation, the more precise the localisation. Such requirement is hard to achieve because not only should they be accurately set at the beginning, the clock drift needs also to be reduced as much as possible.

Time difference of arrival

To bypass some of the flows of the time of arrival measurement, the idea with time difference of arrival (TDOF) is to compute only the difference of arrival times and avoid absolute time computation. This is done by emitting a message simultaneously with several beacons. when the tag receives the messages, it measures their the time difference of arrival and is then able to compute a relative position. With this technique, only the beacons clock needs to be synchronised in order to send their message at the exact same time.

Two-way ranging

In order to completely get rid of clock synchronisation, the two-way ranging technique (also called Single-sided Two-way ranging) can be used. With this method, two messages are exchanged between the beacons and the tag. This is represented on figure 2.1. As each device compute the difference of time between emission and reception ($T_2 - T_1$ for one and $T_3 - T_0$ for the other), the time of flight - and by the way the distance - can be computed using the following equation :

$$2 * TOF = (T_3 - T_0) - (T_2 - T_1) \tag{2.1}$$

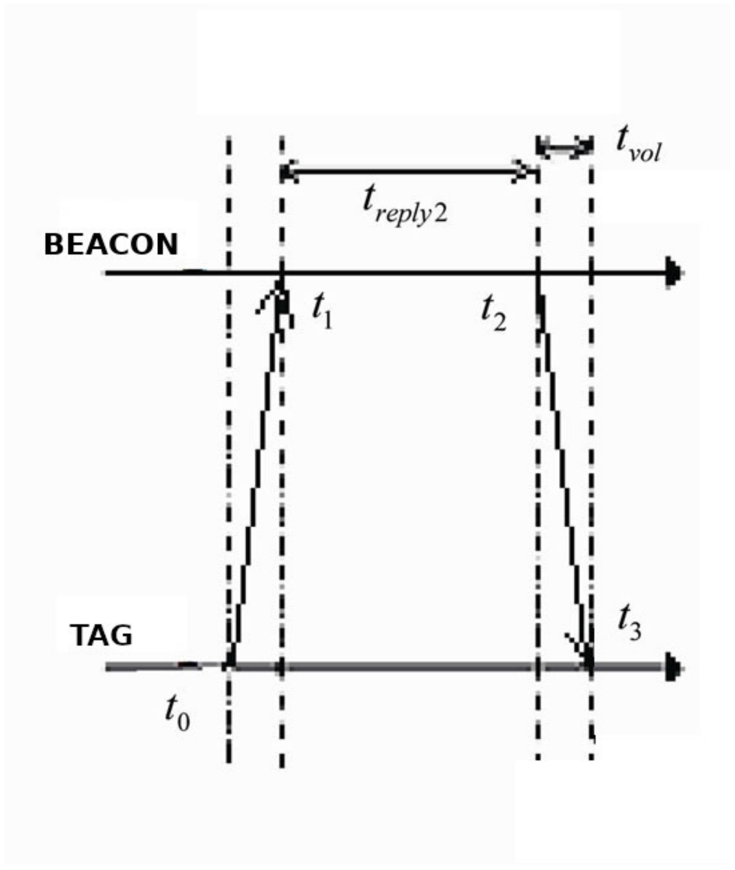


Figure 2.1: Two-way ranging[16]

However, this system still have a flaw : as good as two devices can be made, their internal clock will never "tick" at the same frequency creating what is called a *clock drift*. There is then still potential errors when using this technique. It can be proved (as shown in appendix A) that the error in measurement is :

$$error = TOF\ e_A + \frac{1}{2}T_{reply}(e_A - e_B) \tag{2.2}$$

Where e_A an e_B are the errors on clock frequency (clock drift).

Symmetric double sided two-way ranging

In the previous error formula, the term $(T_2 - T_1)$ also called T_{reply} is significantly greater than the clock drift[1]. This error can then be reduced by avoiding the dependencies with T_{reply}. This is done by "improving" the two-way ranging with symmetric double sided two-way ranging (see figure 2.2).

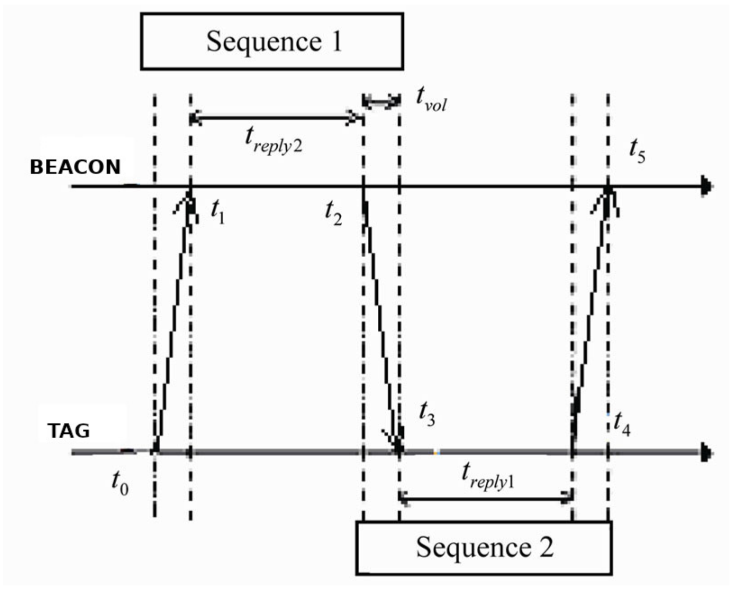


Figure 2.2: Symmetric double sided Two-way ranging[16]

In this configuration, the two-way ranging (TWR) is done from one device and then repeated from the other device. The last message of the first TWR and the first message of the second (TWR) can be merged to spare resources. With this method, the time of flight is computed as shown in equation 2.3 and the error in measurement becomes 2.4 (see appendix B for the details).

$$4 * TOF = [(t_3 - t_0) - t_{reply1}] + [(t_5 - t_2) - t_{reply2}] \tag{2.3}$$

$$error = \frac{1}{2} TOF(e_A + e_B) + \frac{1}{4} \Delta T_{reply}(e_A - e_B) \tag{2.4}$$

Compared to the previous one, this technique can be used on slower system without accuracy loss as the error is no longer directly proportional to the processing time.

[1]With the chip used in this work - the DW1000 - this clock drift is rated a 2ppm with a frequency of 38.4 MHz (7.68 nanosecond) while T_{reply} is in the order of 1 millisecond

2.2.5 Summary

In this section, several techniques allowing to compute a relative position between tag and beacons were shown. It is worth noting that, despite the amount of techniques available, most of them give the same type of relative measurement : either a distance, a direction or a difference of distance. The main criteria when selecting the relative measurement technique is then the cost either in cash, in technology or in computer resources. Ultimately, the relative location presented in this paper are summarised as follows :

- Distance measurement - TOF (time of arrival, twr, sds-twr) and received signal strength.
- Angle measurement - angle of arrival.
- Distance difference measurement - Time difference of arrival (TDOA). Also available with all the distance measurement techniques.

2.3 Different types of RTLS - Absolute position

The measure of an absolute position is done by gathering the results of all the relative measurements and apply a software computation to them. An algorithm is then needed to complete this phase.

It has been shown in the previous section that the output of the relative position measurement could be of three different possible types. This defines the absolute measurement algorithm which will be used.

2.3.1 Trilateration

Trilateration allows to compute an absolute position from the combination of three (in 2D; four in 3D) or more relative distance measurements. As shown on figure 2.3, each distance measurement give a circle of possible position. With each additional relative measurement, the amount of possible absolute locations decreases as the searched solution is the intersection of all the circles. With three beacons, we can draw three circles - in 3D; we would need 5 beacons and therefore 4 circles - and, in an ideal situation, the three circles will have 1 point of common intersection defining the exact position of the tag. Finding an absolute location is then done by solving the following equation.

$$(p_i - p_o)^T(p_i - p_o) = r_i^2 \qquad \begin{array}{l}(i = 1 \rightarrow 3 \ in \ 2D) \\ (i = 1 \rightarrow 4 \ in \ 3D)\end{array} \tag{2.5}$$

Where p_i are the coordinates of the ith beacon, p_o are the coordinates of the tag and r_i is the distance between p_o and p_i.

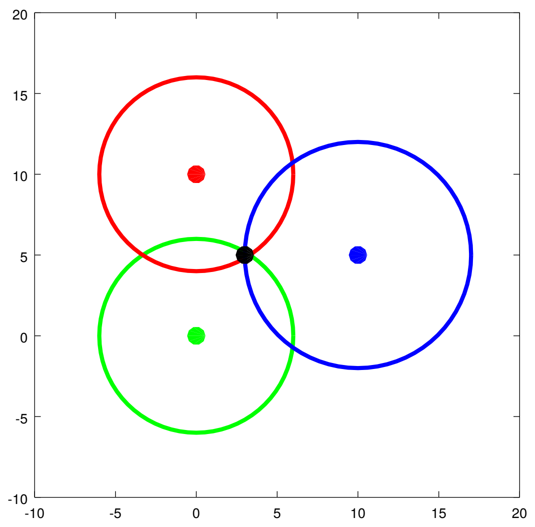


Figure 2.3: Trilateration

One of the limits of this algorithm however is that the position of the beacons influence the localisation precision. It can be shown that ideally, the beacons should be positioned following an equilateral triangle. In the case where the beacons are aligned, the equation system 2.5 would give 0 or 2 solutions (see figure 2.4).

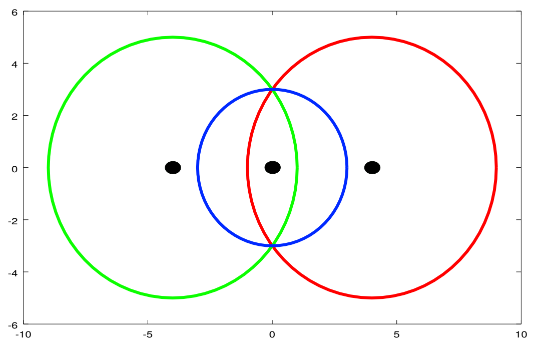


Figure 2.4: Trilateration - problematic case

2.3.2 Triangulation

Triangulation is used with angle of arrival measurements. By intersecting the direction of propagation measured by at least two beacons, an absolute position can be find (see figure 2.5). Triangulation is more advantageous than trilateration in most of the cases because it only needs 2 beacons to work. However, as it has been said earlier, those beacons are more costly and this technology is very sensitive to multipath situation. However, it must be noted that with only 2 beacons, there will be a blind area where the tag could not be located. This is along the line linking the two beacons. The addition of a third unaligned beacon will be needed to remove this blind area.

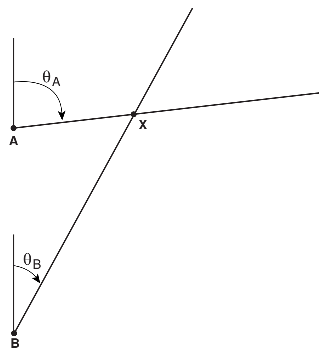


Figure 2.5: Triangulation[2]

2.3.3 Multilateration

The last relative measurement discussed in the previous section is the distance difference measurement using time difference of arrival. Computing a difference of distances instead of a straight distance leads to an hyperbole of possible positions instead of a circle (see figure 2.6).

Like trilateration, this methods needs 3 beacons to work (4 in 3D) and it is, in general, more precise than the former because of the hyperbolic system. However, it needs permanent communication between the different beacons (to compare their time of arrival) or must the tag be able to receive several transmissions in a very short time or even sometimes simultaneously.

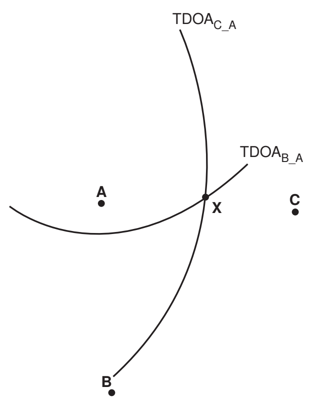


Figure 2.6: Multilateration[2]

2.3.4 Summary

In this section, the three main absolute positioning algorithm where presented. The trilateration is the most commonly used amongst the three because it theoretically the easiest in therm of calculations. Triangulation despite the fact it needs only two beacons to work is less used because of the cost of implementation. Multilateration tends to be more and more used as it shows better results than trilateration without high cost increase like triangulation.

2.4 Ultra-Wideband (UWB) system : Overview

2.4.1 UWB technology

Ultra-Wideband (UWB) is a communication technology which uses, as stated by the name, a very high bandwidth; at least 500MHz or 25% of the central frequency. The Federal Communications Commission (FCC) allows the use of the frequency bands between 3.1 and 10.6 GHz for this technology with a limited power emission of 75nW/MHz[13].

UWB was formerly known as "pulse-radio". The system transmits very short timed impulse-like waveform, shorter than a nanosecond long (see figure 2.7). Due to these unique characteristics, UWB comes with a lot of advantages[4, 19, 12].

- Great robustness against multipath situation

  The very short duration of the pulses allows to easily filter the signal reflections (figure 2.8).

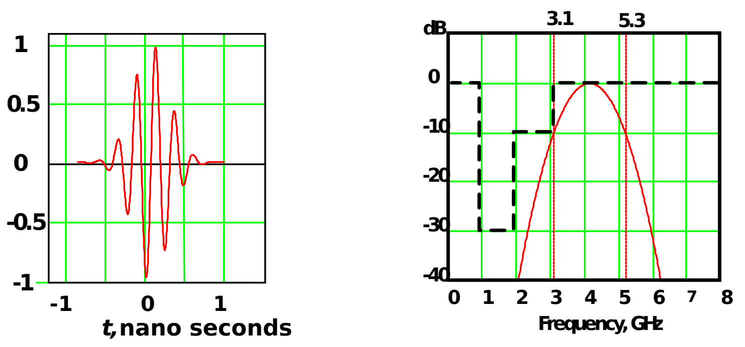


Figure 2.7: UWB waveform in time and frequency domain[19]

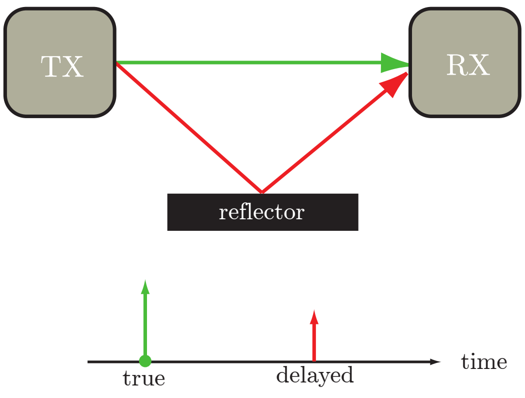


Figure 2.8: UWB signal in multipath situation[15]

- Energy efficiency

  Compared to other wireless technologies such as Wi-Fi or Bluetooth, UWB has a very low power consumption.

- Low detection probability/Coexistence with other systems

  The low power spectral density of the UWB technology make it hard to detect for unaware systems as it is not very higher than typical electromagnetic noises (figure 2.9).

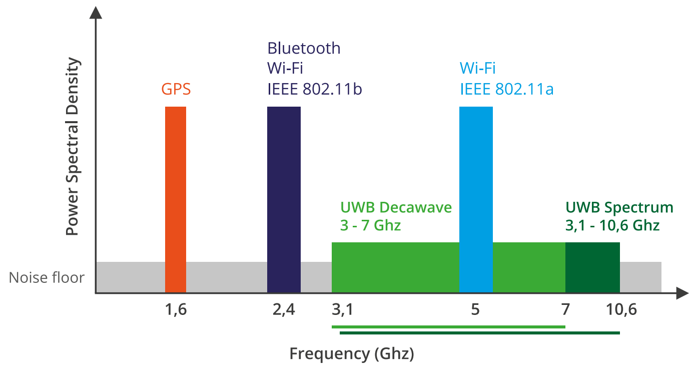


Figure 2.9: UWB compared to other radio communication systems

- Low signal to noise ratio

  Intuitively, one can already say that a pulse signal is less sensitive to noise (figure 2.10). This can be further demonstrated using the Shannon-Hartley theorem (2.6) with the bandwith properties of UWB.

  $$C = B * log_2(1 + S/N) \tag{2.6}$$

  - C: Channel capacity (bit/s)
  - B: Bandwith (Hz)
  - S/N: Signal to noise ratio

- High bit rate

  The short pulse duration and the low noise sensitivity allows a high bit rate within communications using UWB. However, due to the behaviour of the signal, this high bit rate is only possible at close range. If the communication distance increase, the bit rate should decrease in order to ensure good message transmission. As an example, the DWM1000 used for this thesis allows a bit rate up to 6.8Mb/s at 10m range while the bit rate can decrease to 110kb/s while working at higher range.

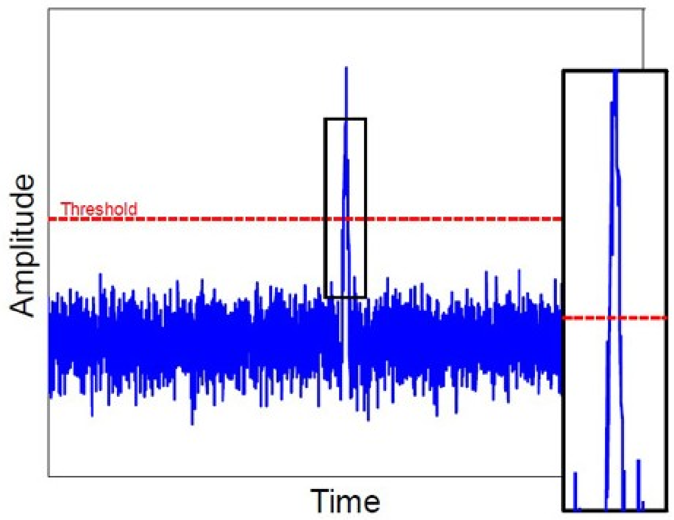


Figure 2.10: Ultra WideBand with noise[5]

2.5 Conclusion

In this chapter, the way real time locating systems works was decomposed in two section : relative position measurement and absolute position measurement. For each of these sections, the different techniques available were presented. For the first part, the techniques were numerous but leaded to only few different results. The selection of one of them should be based essentially on the budget available and the desired accuracy. The techniques of the second section where only three and the selection amongst them is done based on the choice made for the first section. In this work, the RTLS will be based on the sds-twr ranging technique (distance measurement) because it is the more precise one when working with time of flight measurements. The used algorithm will then be a trilateration algorithm.

Chapter 3

Range estimation

3.1 Experimental setup : DWM1000

The experimental setup for this work consist of four ultra wideband tranceivers; each connected to a master board. Three of these devices will play the role of beacons while the last will be the tag. As there is only 3 beacons, this work will focus on two-dimensional localisation.

The tranceivers are DWM1000 modules developed by the company DecaWave (see figure 3.1). These modules allows an easy implementation of UWB communication with high performances for a relative low cost[1].

Figure 3.1: DWM1000 module[1]

Communication between the DWM1000 and the master board are made through an SPI interface.

The board used to control the DWM1000 in this work is a STM32 Discovery board (figure 3.2). It allows SPI communication interface along with an UART interface to easily communicate with a computer during the prototyping phase. Moreover, this board allows a 100mA power output which is needed to supply the DecaWave chip. This allows a faster prototyping since no external power board will be needed.

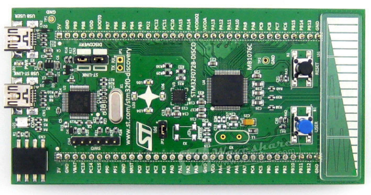


Figure 3.2: STM32-F0 Discovery board

3.2 Distance measurement with the DWM1000

As explained earlier, the first step when designing a real-time locating system is to focus on the relative location measurement. In this case, the relative location is a distance measurement realised with the symmetric double sided two-way ranging technique (sds-twr) (see chapter 2.2.4).

The distance measurement between the tag and a beacon is made following the communication scheme of figure 3.3. The communication is initiated by the tag which is located on the robot. A fourth message is added to the sds-twr ranging technique to send the beacon's times to the tag which will then make the distance measurement computation.

The software of the two devices is a state machine as represented on figures 3.4. The tag software is self explanatory and follows the baselines of the ranging technique. If it detects an error during the communication, the ranging is aborted and another full cycle is restarted. The beacon protocol is a little different. To avoid synchronisation problems, due to a message loss for example, it will check the received message to know what will be the next step of the ranging. For this, a different ID has been given to every exchanged messages allowing a simple identification of the ranging progression.

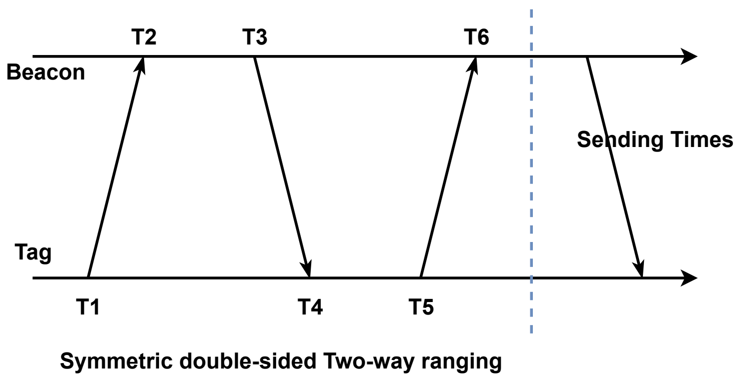


Figure 3.3: Distance measurement protocol with DecaWave devices

The choice to abort and restart a full cycle could seem to be an extreme solution. This choice is based on the first experiment which showed that this message lost was a quite frequent situation. There was a need to conceive a system resilient enough to sustain this. The logic of the beacon software result directly from this solution.

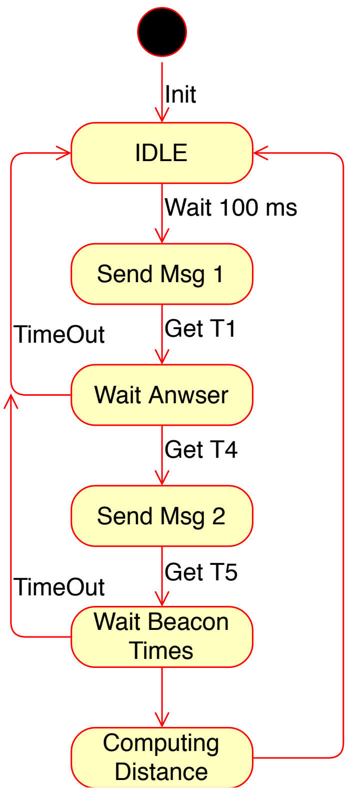
3.3 Error Characterization

The error in the measurement is characterised by taking 1000 measures each 10cm from 0.5 to 6.5m. During this experiment, two issues appeared (see figure 3.5). First, with a distance lower than 70cm, a lot of error appears leading to bad measurement. These errors where the result of very low time of flight which leaded to unstable computation in the software (such as dividing by a very low number).

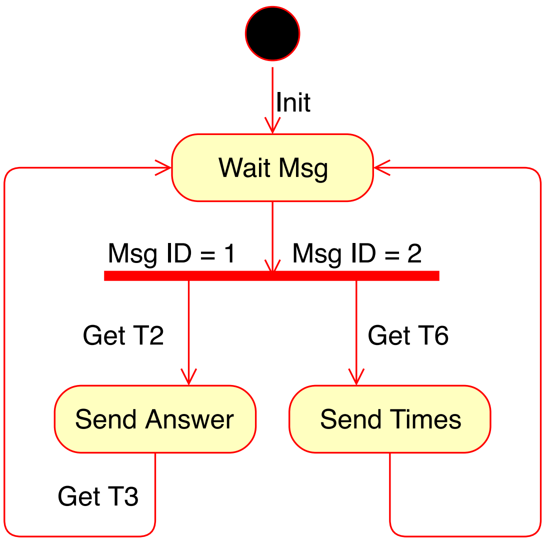
Second, and it can be clearly seen on the figure 3.5, there was an increasing static error the smaller the measured distance was. It is supposed that this phenomenon is due to the calibration phase which focuses on longer range measurement.

To solve those two problems, the device has been calibrated to over evaluate the time of flight by reducing the antenna delay of the DWM1000. This way, the measurements stays in the quasi linear range of the graph 3.5. Finally, a corrective fitting polynomial computed with *Matlab* allowed to compensate the increasing static error and get correct results. The measurements obtained after this are shown on figure 3.6.

With this setup, the error on the measurements is minimal. In fact, it can be measured that the error on the distance follows a Gaussian curve with a standard deviation of around 3.2cm (mean std for all the sets of 1000 measurements).



(a) Tag software



(b) Beacon software

Figure 3.4: Representation of the devices' software

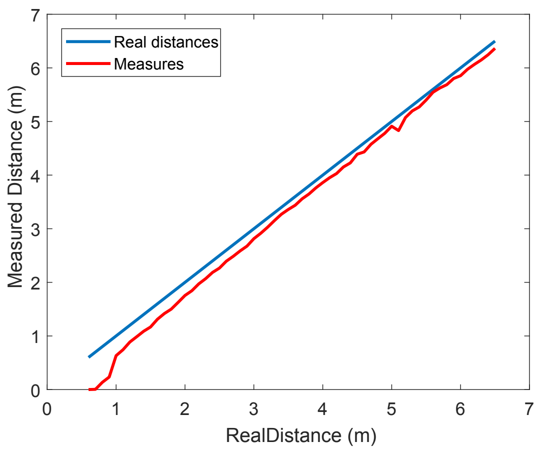


Figure 3.5: Result of the first measurement campaign

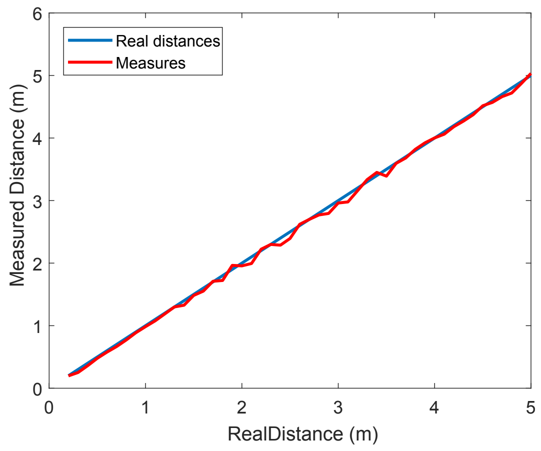


Figure 3.6: Measurement campaign with polynomial correction

3.4 Distance measurement with multiple DWM1000

The previous section described the distance measurement between the tag and one beacon. In a localisation, problem, an additional layer of complexity must be added since locating needs simultaneous measurements with three different beacons.

As a real simultaneous measurement of the distance with three beacons will be very difficult to implement, it has been decided in this work to make the three different distance measurements in quick succession. As one measurement take approximately 2ms (17 seconds for 1000 measurements), making three successive measurements is considered fast enough for the error resulting from the non-simultaneity staying in acceptable range. As the base speed of the robot is 0.4m/s, the maximum distance it can travel between two measurements is 1.6mm. This is a tenth of the standard deviation on one measurement.

On another side, communicating with three different devices is not an easy task. A way must be find to avoid simultaneous message emissions which would create interference. To solve this, a specific ID has bee given to every message which will contain the "name" of the next device allowed to speak. With this, the tag always know the beacons it is communicating with and every beacon can filter the received messages to send their answer and times only when required. Moreover, this can allows for an external listener to know at every time the evolution of the measurement. This asset was very valuable during the debugging phase.

3.5 Conclusion

This chapter was dedicated to the implementation of a symmetric double-sided two-way ranging distance measurement using the *Decawave* DWM1000 module. At this stage, we mainly focused on the distance measurement between the tag and a beacon. The first trials allowed us to identify several source of measurements perturbation - reflection, antenna position, environment influence, antenna delay. Some of those perturbations could be limited through an adequate positioning of the antenna. To fix intrinsic errors, we went through a calibration phase which gave a polynomial correction function of the distance.

Chapter 4

Target localization

4.1 Cramer-Rao Lower Bound / Error theory

As shown in the previous chapter, the distance measurement between a tag and beacon is subject to noise. The tag absolute position will be a function of those measured distances and will, then, be impacted by their noisy behaviour. The vector defined by the estimate of the three distances could be considered as a noisy estimator. To measure the effectiveness of the positioning algorithm, we can compare the noticed variance of the position measurements with its theoretical minimal variance.

The theoretical minimal variance for an estimator of a variable parameter is known as the Cramer-Rao lower bound (CRLB). This value gives an indication of the quality of the used estimator. In this case, the CRLB is the minimum variance obtainable on the measured position knowing the variance on the relative distance measurements between tag and beacons. By comparing this value with the one obtained with the trilateration algorithm, the quality of this algorithm could be assessed. On the other hand, this CRLB also will also give us the information on the best positioning of the beacons to maximise the accuracy of the position measurement.

To compute the CRLB, the first step is to define the estimator. In this case, the absolute position of the robot is estimated from the relative distance measurements on the beacons :

$$z = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix} + \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \tag{4.1}$$

$$d_i = \sqrt{(x - x_i)^2 + (y - y_i)^2} \tag{4.2}$$

Where z is the position parameter, d_i is the distance relative to the position of the beacon $i : (x_i, y_i)$ and w_i are the noises on the measurements.

From the results of the previous experiments, w_i is supposed to follow a normal distribution $w_i \sim N(0, \sigma^2)$. As the noises are considered independent, the noise matrix follows a multivariate normal distribution of mean 0 and standard deviation σ :

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \sim MVR\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & \sigma^2 & 0 \\ 0 & 0 & \sigma^2 \end{bmatrix} \right) \tag{4.3}$$

Since d_i is a constant for a given position, from the equation 4.1, it can be shown that z also follows a multivariate distribution :

$$\begin{aligned} z &\sim MVR\left(\begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}, \begin{bmatrix} \sigma^2 & 0 & 0 \\ 0 & \sigma^2 & 0 \\ 0 & 0 & \sigma^2 \end{bmatrix} \right) \\ &\sim MVR\big(f(x,y), R\big) \end{aligned} \tag{4.4}$$

The computation of the CRLB needs the introduction of the Fisher information matrix. This matrix represents the "amount of information given by the estimator about the observed parameter". Mathematically, the CRLB is obtained by inverting it. In the case of a multivariate normal distribution, the elements of the Fisher matrix are computed as follows.

$$I_{\theta_i,\theta_j} = \frac{\partial f(x,y)}{\partial \theta_i}^T R^{-1} \frac{\partial f(x,y)}{\partial \theta_j} + \frac{1}{2} tr\left(R^{-1} \frac{\partial R}{\partial \theta_i} R^{-1} \frac{\partial R}{\partial \theta_j} \right) \tag{4.5}$$

Where θ_i and θ_j are the dimensions of the function f; x and y in our case.

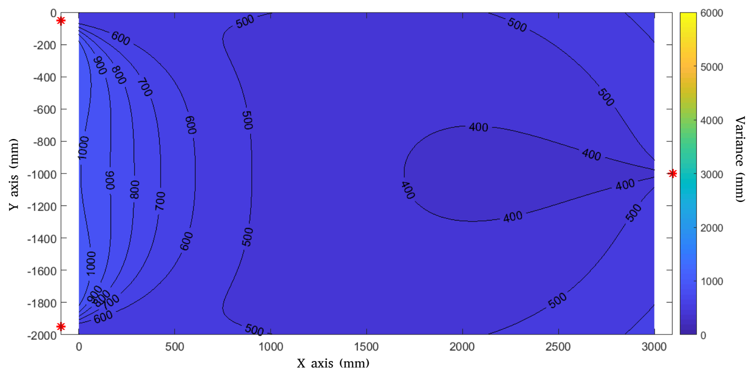
As the covariance of the measurements in our system has been measured constant, matrix R is also constant. This makes the last term of the previous equation become zero.

$$I_{\theta_i,\theta_j} = \frac{\partial f(x,y)}{\partial \theta_i}^T R^{-1} \frac{\partial f(x,y)}{\partial \theta_j} \tag{4.6}$$

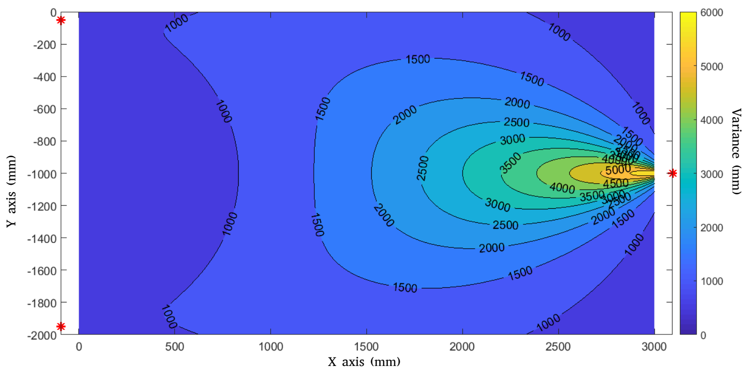
Using the value of $f(x,y)$ and R expressed on equation 4.4 and expressing the d_i with the equation 4.2, the elements of the Fisher information matrix are computed as follows.

$$
\begin{aligned}
I_{x,x} &= \sum_{i=1,2,3} \frac{-(x-x_i)^2}{((x-x_i)^2+(y-y_i)^2)\sigma^2} \\
I_{x,y} = I_{y,x} &= \sum_{i=1,2,3} \frac{-(x-x_i)(y-y_i)}{((x-x_i)^2+(y-y_i)^2)\sigma^2} \\
I_{y,y} &= \sum_{i=1,2,3} \frac{-(y-y_i)^2}{((x-x_i)^2+(y-y_i)^2)\sigma^2}
\end{aligned} \tag{4.7}
$$

I being a 2x2 matrix and the CRLB being it's inverse, it will also be a 2x2 matrix. The components of this matrix are the lower bound for the variance decomposed along the x and y axis.



(a) $CRLB_{xx}$



(b) $CRLB_{yy}$

Figure 4.1: Values of the CRLB matrix representing the minimal variance on the position measurement. The beacons are positioned according to the *Eurobot* rules

$$CRLB = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 \end{bmatrix} \tag{4.8}$$

The result obtained for σ_{xx}^2 an σ_{yy}^2 are presented on figure 4.1 for the beacon disposed as required on the $Eurobot$ rules[6].

In order to ease the analysis of these data and to avoid axis dependencies, a "general" CRLB is defined as the trace of the previously computed matrix.

$$\begin{aligned} CRLB_{general} &= CRLB_{xx} + CRLB_{yy} \\ &= \sigma_{xx}^2 + \sigma_{yy}^2 \end{aligned} \tag{4.9}$$

This new CRLB is then the sum of the two presented on figures 4.1a and 4.1b. This general bound is shown on figure 4.2.

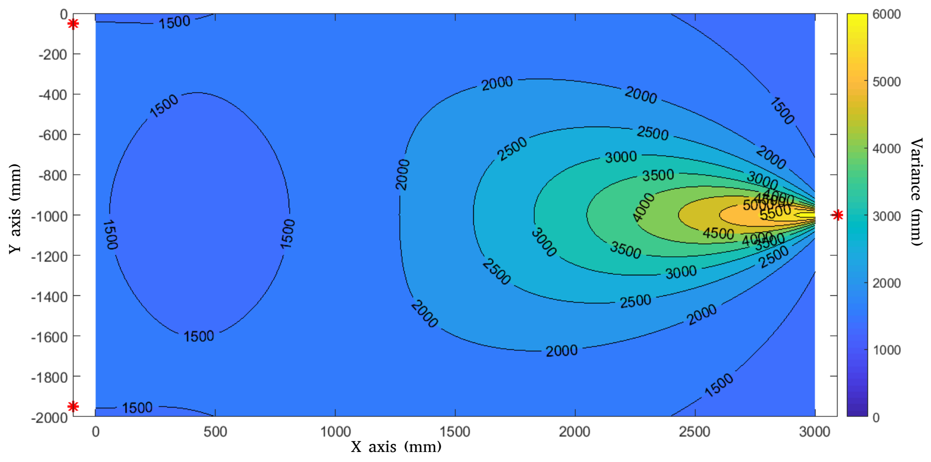


Figure 4.2: "General" bound on the variance with the beacons positioned according to the *Eurobot* rules

The optimal positioning is obtained when the beacons are placed following a regular shape[18] (here, an equilateral triangle). As a comparison and to evaluate the beacon positioning of the *Eurobot*, the CRLB of the optimal case is shown on figure 4.3.

It can be seen that the $Eurobot$ disposition is clearly worse than the optimal positioning. The CRLB reaching a value for the variance up to 6000 (standard deviation of 77 millimetres) near the far right beacon. On the other side, the worst variance with an optimal disposition reaches only 2500 (50mm std) when getting close to a beacon.

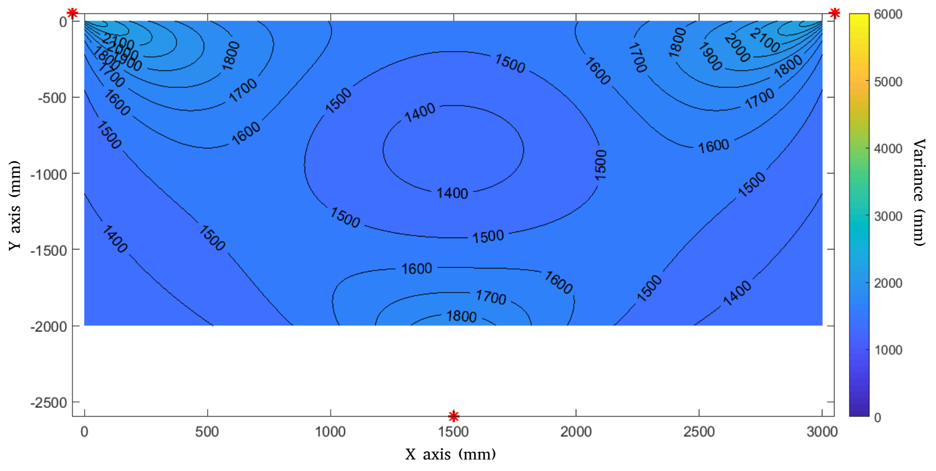


Figure 4.3: "General" bound on the variance with the beacons at optimal position (regular shape)

4.2 Trilateration algorithm

Now that we have identified the best obtainable accuracy, we will choose a trilateration algorithm and evaluate its performances based on the CRLB.

As presented earlier, trilateration locate an object by solving the system of equations :

$$(p_i - p_o)^T(p_i - p_o) = r_i^2 \tag{4.10}$$

However, in reality, errors and inaccuracies on the measurements of r_i makes this system unsolvable in most cases. As shown on figure 4.4, the three circles do not systematically cross on one unique point. It is then necessary to find a way to compute a best possible position in the cases the system of equations 4.10 is unsolvable.

For this thesis, the trilateration algorithm developed by Zhou[22] will be used. The low computational complexity and robustness of this algorithm are important factors in its choice since it allows to spare computational resources and allocate them to other tasks.

This algorithm is based on a least-square error formulation of the trilateration which can be measured as of equation 4.11.

$$S(p_0) = \sum_{i=1}^{N} \left[(p_i - p_0)^T(p_i - p_0) - r_i^2\right]^2 \tag{4.11}$$

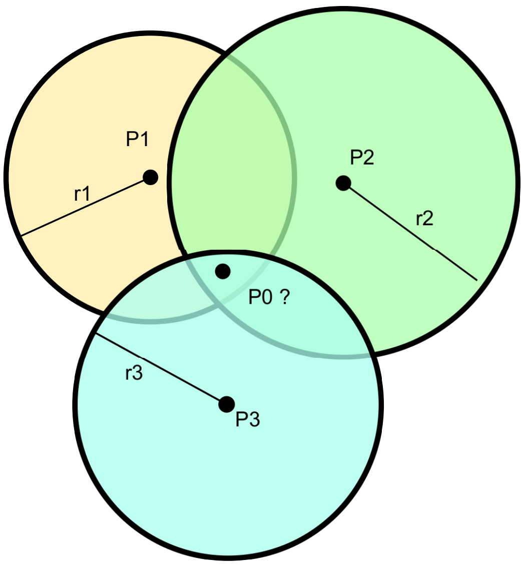


Figure 4.4: Trilateration with measurement errors

The principle of this algorithm is to find the position p_0 that will minimise the square of the error: $S(p_0)$. In this equation, p_i represents the position of the ith beacon, r_i is the distance measured by the beacon and N is the amount of beacon. The details of the resolution will not be presented here. They are explained in the work of Zhou[22]. The computation sequence, however, can be found in appendix C.

After measuring the distances between the tag and each beacon and putting the results in the algorithm, the position of the robot is known. To ensure the efficiency of the algorithm, the variance of its results will be compared with the CRLB previously presented as this is by definition the best possible variance for any trilateration algorithm.

4.3 Target localisation accuracy and errors

In order to measure the accuracy of the chosen trilateration algorithm, a full mapping of the play area has been realised with the beacons placed according to the *Eurobot* rules. By comparing the results obtained with the theoretical CRLB, the quality of the algorithm can be assessed. The mapping is realised by taking 1000 measurements on each point of a ten millimetres grid. The variance of these measures is then computed as shown on figure 4.5. Due to the limited table size, a full mapping of the area could not be realised as some points near the edge could not be attained.

Comparing the figures 4.5 and 4.2, we can see some recurring patterns with, for example a better accuracy on the left side of the table compared to the right side. Broadly speaking, the variances values stays in the range of the CRLB values. This comfort the choice of this algorithm. The main differences between the theory and the practice are the error peaks appearing on the corners of the play area. This could be explained by a higher occurrence of reflection and multipath transmission in these areas due to the presence of obstacles.

In general, the accuracy of the system can be considered pretty good with a standard deviation of 3.1cm on the left side of the table. Luckily, thanks to how the *Eurobot* matches are made, it is the side where the robot will spend the most of it's time. This standard deviation reach around 5.4cm on the other side of the table where it reaches 7.7cm. These performances should be compared with the ratings from the DecaWave company which is 20cm for the tags localisation precision[1].

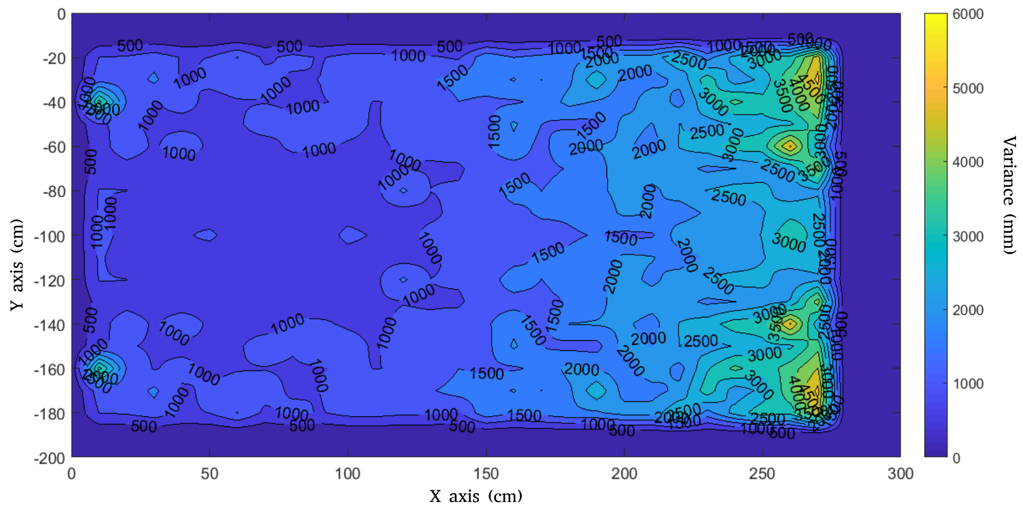


Figure 4.5: Measured variance on the play area

Unfortunately, as the *bULBot* robot participating to the *Eurobot* cup was not ready on time, tests during the cup could not be realised. However, in order to test the algorithm reliability, in a real situation, the device has been installed on an *Elegoo smart robot kit* programmed to follow a line on the ground. As the traced path coordinates are known, a comparison between the measured location and the exact location can be done and the accuracy of the algorithm can be measured. For this test session, the beacons were placed as close as possible *Eurobot* rules prerequisites. The results obtained after this are shown on the figure 4.6.

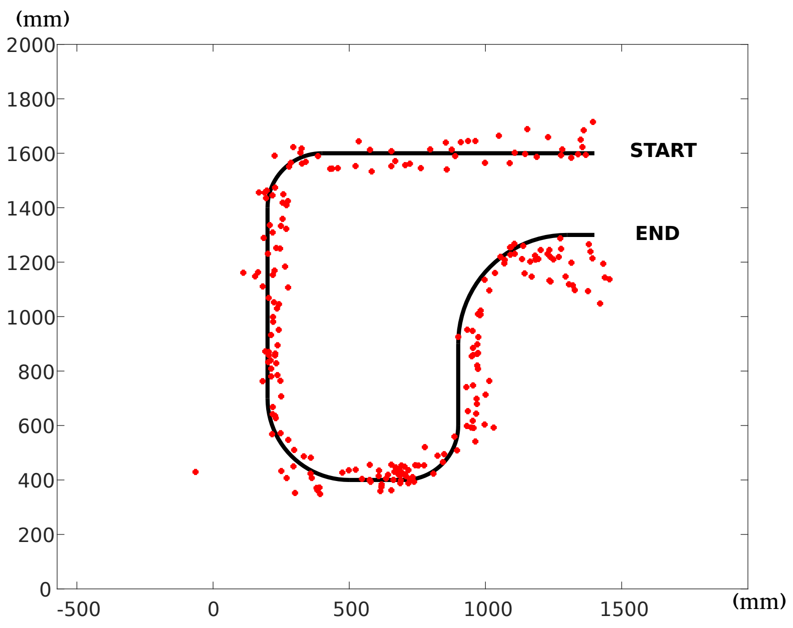


Figure 4.6: Result of the trilateration measurement

Although the results are quite accurate, we can still see a certain degree of noise in the system with an error reaching 15cm in the worst case.

4.4 Conclusion

In this chapter, the trilateration algorithm used to localise the robot has been presented. It has been compared with the theoretical lower bound for the accuracy. The small difference between this lower bound and the algorithm accuracy has assessed it's effectiveness. However, the testing session revealed that the system was still pretty noisy. The next chapter will focus on a way to smooth those measurements and increase their accuracy in doing so.

Chapter 5

Target tracking

To increase the accuracy of the localisation, a way to filter the noise in the system must be found. In most systems, noise reduction is done by implementing a low-pass filter or a moving average. However these techniques have major flaws in the way that the filtered result comes several cycles after the real situation. It is then difficult to correct a trajectory based on these because the correction will always act with a time offset.

Some more advanced algorithms reduce the noise by trying to predict the next state of the system based on all or part of the previous states, hence the name tracking. The Kalman filter which will be presented in this chapter is one of these algorithms.

5.1 Kalman Filter

In short, a Kalman filter is a recursive algorithm which tries to estimate the current state of a noisy system based on measurements related to the states of this system, but perturbed by noise[7]. At each iteration, the Kalman filter tries to predict the future state of the system based on the previous measurements. Then, it updates itself based on the difference between the actual measurements and the prediction.

5.1.1 Implementation

State space model

Every system can be represented like in the figure 5.1. A series of inputs U_k influence some system variables X_k which produce themselves some measured outputs Y_K.

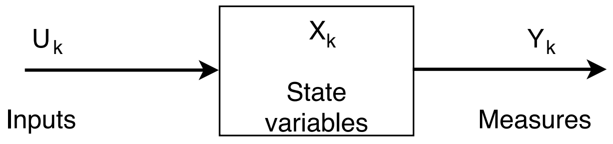


Figure 5.1: System representation

In order to implement a Kalman filter to such system and by extension, to our robot system, a linear state space model must be found. This linear state space model describe the whole system evolution using the equation 5.1.

$$\begin{aligned} X_{k+1} &= AX_k + BU_k \\ Y_k &= CX_k \end{aligned} \tag{5.1}$$

In this equation, X_k is a vector containing the system variables, U_k is a vector containing the input and Y_k contains the measurements. A,B and C are matrices of operations applied to those vectors while the system is working. To apply this to our robot, we must first seek equations describing the behaviour of the robot while working. If we decide that the full state of the robot is described by its position and speed along both axes and that the inputs of the system control the speed, the equations are as follows.

$$\begin{aligned} x_{k+1} &= x_k + \Delta t\ \dot{x}_k \\ y_{k+1} &= y_k + \Delta t\ \dot{y}_k \\ \dot{x}_{k+1} &= \dot{x}_k + f(input) \\ \dot{y}_{k+1} &= \dot{y}_k + f(input) \end{aligned} \tag{5.2}$$

$$\begin{aligned} x_{measured} &= x_k \\ y_{measured} &= y_k \end{aligned} \tag{5.3}$$

To describe this system as a state space mode, the equations 5.2 and 5.3 must be expressed according to the equation 5.1.

With the actual system installation, the localisation part does not have access to the wheels controls. The inputs and, by extension, U_k can not be measured and included in the system representation. As a first estimation, is as be chosen that they will be supposed equal to zero as if the robot was moving along a straight line. The influence of the inputs will then be treated as noise in the filter.

The definition of X_k and Y_k is pretty straightforward as those vectors are both on the left side of the equations 5.2 and 5.3.

$$X_k = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \tag{5.4}$$

$$Y_k = \begin{bmatrix} x \\ y \end{bmatrix} \tag{5.5}$$

From this, expressing the values of A and C is relatively easy.

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5.6}$$

To implement a Kalman filter on this system, three additional matrices must be defined :

- P : The error covariance matrix. It represents the errors in the estimations realised by the filter.
- Q : The process noise covariance matrix. It represents the noise in the inputs (U_k) of the system.
- R : The observation noise covariance matrix which represents the noise in the measurements (Y_k)

Finding the precise values for these matrices in not an easy task and this would go beyond the scope of this work. The method used here was to chose the identity matrix as a first value for them. Then, by modifying the values of these matrices during a trial and error phase, the Kalman filter has been tuned to give the best possible result (see section 5.1.2).

Prediction phase

Now that every variable has been defined, the implementation of the Kalman filter strictly speaking can begin.

As said earlier, at each iteration, the filter predicts the future state of the system. This prediction phase consists of the following equations[21].

$$\begin{aligned} \hat{X}_{k+1} &= AX_k \\ \hat{P}_{k+1} &= APA^T + Q \end{aligned} \tag{5.7}$$

During this phase, the filter makes an estimate of the next system state $(\hat{X}_{k+1})$. It also predicts the next value for the error covariance matrix $(\hat{P}_{k+1})$ representing the possible error in this estimate.

Innovation phase

The filter now compares the prediction with the measurements and will compute what is called the "Kalman gain" K.

$$\begin{aligned} Z &= Y_{k+1} - C\hat{X}_{k+1} \\ S &= C\hat{P}_{k+1}C^T + R \\ K &= \hat{P}_{k+1}C^T S^{-1} \end{aligned} \tag{5.8}$$

On one side, the filter will measure the error between the actual measurements and the prediction (Z) while simultaneously computing the "Kalman gain" (K) representing how much Z will impact the filtered result.

Update phase

Based on the Kalman gain, the filter give a final result for X_{k+1} and update its error matrix.

$$\begin{aligned} X_{k+1} &= \hat{X}_{k+1} + KZ \\ P_{k+1} &= (I - KC)\hat{P}_{k+1} \end{aligned} \tag{5.9}$$

5.1.2 Results

When this Kalman filter is applied to the trilateration results presented earlier, the trajectory from figure 5.2 is obtained.

These results shows a good noise reduction during the first straightforward part of the trajectory. However, because of the filter having no access to the commands made to the robot, the inputs of the system, it can not predict the turns. This causes what can be seen as some kind of inertia when the robot is actually turning. The accuracy of the filtered trajectory is then decreased as turns are realised in a more and more rapid succession.

It is possible to reduce this inertia phenomenon by further modification of the values the matrices P, Q and R. But, this reduces the "smoothing" effect of the filter (see figure 5.3). A trade-off must then be found. For this work, a better following of the real trajectory will be prefered over a smooth trajectory.

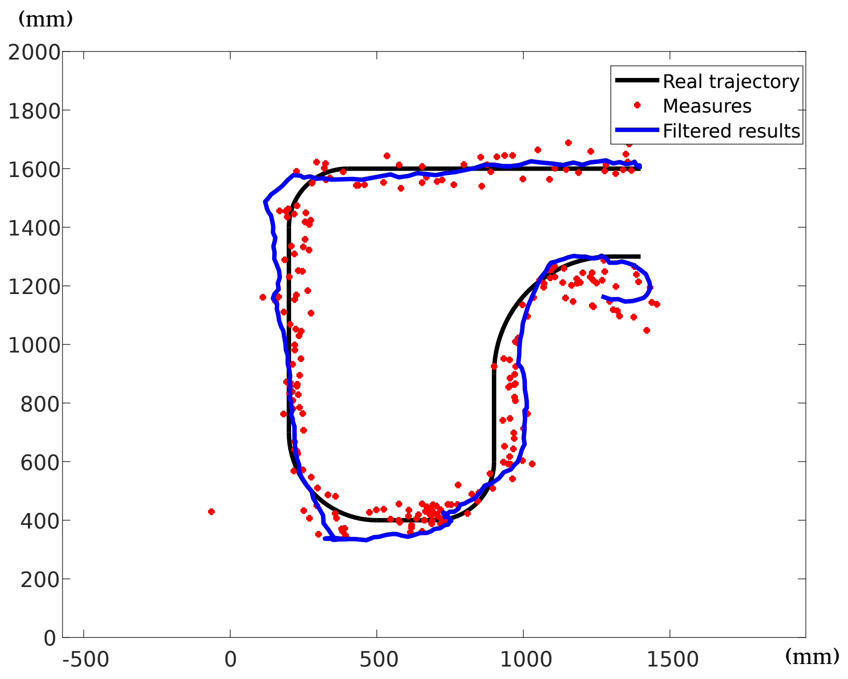


Figure 5.2: Results of the Kalman filter

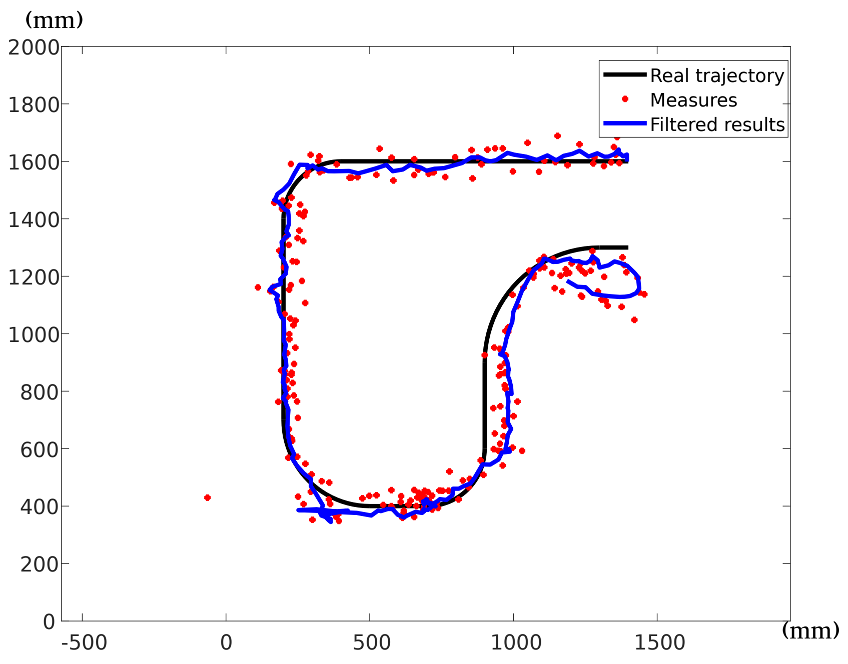


Figure 5.3: Results of the Kalman filter with reduced inertia

5.2 Extended Kalman filter

The main weakness of the Kalman filter is that it can only be used for linear systems. Yet, most real life systems are not linear. This is why the "Extended Kalman Filter" has been developed for. The concept behind this is to linearize the system around the working state so the equation of the standard Kalman filter can be used.

5.2.1 EKF : orientation

To illustrate the extended Kalman filter implementation, the orientation of the robot is added as a state variable in the system.

$$X_k = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \\ \theta \end{bmatrix} \tag{5.10}$$

Introducing this new variable makes the new system non-linear. The state evolution can not be expressed with the equation 5.1 anymore. Instead, the state evolution equation becomes :

$$X_{k+1} = \begin{bmatrix} x_k + \Delta t \cdot \dot{x}_k \\ y_k + \Delta t \cdot \dot{y}_k \\ V cos\theta_k \\ V sin\theta_k \\ \theta_k \end{bmatrix} \tag{5.11}$$

In order to use the Kalman filter equations on this new system it must be linearised and the different state space matrices must be redefined. The linearisation of the system is done by taking the derivative of its state evolution equation at each iteration. This is done by taking the jacobian of the matrix in equation 5.11 and using it in the Kalman filter as the new A Matrix. In this case, the obtained matrix is :

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & -V sin\theta \\ 0 & 0 & 0 & 0 & V cos\theta \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.12}$$

The other (Y_k) part of the system remains unchanged, as there is no additional measured variable. So, the other matrices stay the same as in the previous section.

Applying these new values in what is now called the Extended Kalman filter, the results presented on figure 5.4 are obtained.

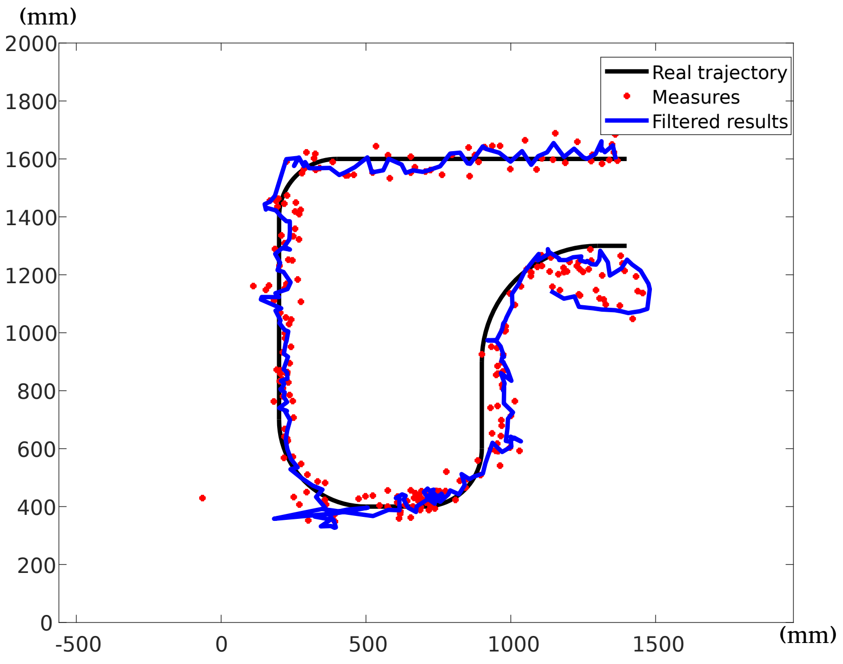


Figure 5.4: Results of the Extended Kalman filter

Comparing this figure with the results of the standard Kalman filter from figure 5.3, it can be seen that the EKF does not give better results. It is explained by the fact that the nonlinear part of the system added with the addition of the orientation as a space variable bring additional noise in the system. As this noise is not compensated with any additional measured variable there is a loss of accuracy in the filtered results. Going from a linear system to an nonlinear one here bring clearly no benefits. However, it is supposed that if the orientation of the robot could be directly measured, it's addition as measurement of the system could make the EKF a better solution than the standard KF.

5.2.2 EKF : trilateration

Although it has been shown that, the implementation of an Extended Kalman filter didn't came with any advantage in this case, it is still a very powerful tool. For example, it can be used to get rid of the trilateration which is often considered as one of the more heavy computing part in RTLS. This can be done because of the philosophy behind the Kalman filter : it predicts the future state and the measurements matching this prediction, then compares predicted measurements with real measurements and find a compromise between them.

To implement this in our system, the same philosophy as previously is used. The variables of X_k stays the same than in the previous section but the observations Y_k are now the distances between the tag and the beacons. The system behaviour is now described by the following equations.

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \dot{x}_{k+1} \\ \dot{y}_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \Delta t \cdot \dot{x}_k \\ y_k + \Delta t \cdot \dot{y}_k \\ V cos\theta_k \\ V sin\theta_k \\ \theta_k \end{bmatrix} \tag{5.13}$$

$$Y_k = \begin{bmatrix} d1 \\ d2 \\ d3 \end{bmatrix} = \begin{bmatrix} \sqrt{(x_k - x_A)^2 + (y_k - y_A)^2} \\ \sqrt{(x_k - x_B)^2 + (y_k - y_B)^2} \\ \sqrt{(x_k - x_C)^2 + (y_k - y_C)^2} \end{bmatrix} \tag{5.14}$$

With $x_{A,B,C}$ and $y_{A,B,C}$ being the coordinates of the beacons.

Like for the matrix A in the previous example, the jacobian of those expressions must be used in order to linearize the system. The jacobian of the expression 5.13, which will be used as the matrix A in the Kalman filter equations, has already be computed in the previous section (see equation 5.12). The matrix C will be defined by the jacobian of the expression 5.14. It is expressed as follow.

$$C = \begin{bmatrix} \frac{x_k - x_A}{\sqrt{(x_k-x_A)^2+(y_k-y_A)^2}} & \frac{y_k - y_A}{\sqrt{(x_k-x_A)^2+(y_k-y_A)^2}} & 0 & 0 & 0 \\ \frac{x_k - x_B}{\sqrt{(x_k-x_B)^2+(y_k-y_B)^2}} & \frac{y_k - y_B}{\sqrt{(x_k-x_B)^2+(y_k-y_B)^2}} & 0 & 0 & 0 \\ \frac{x_k - x_C}{\sqrt{(x_k-x_C)^2+(y_k-y_C)^2}} & \frac{y_k - y_C}{\sqrt{(x_k-x_C)^2+(y_k-y_C)^2}} & 0 & 0 & 0 \end{bmatrix} \tag{5.15}$$

The filtered trajectory obtained with this filter is shown on figure 5.5. Unfortunately, this result is even worse than the previous one. However, it shows that it is possible, in order to spare computational resources, to remove the trilateration algorithm from the software.

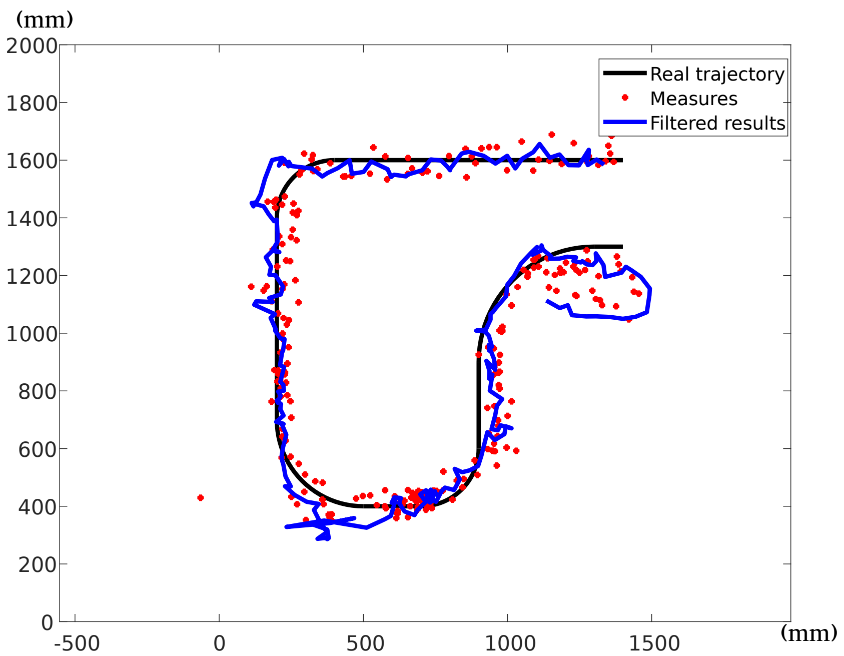


Figure 5.5: Results of the Extended Kalman filter with trilateration

5.3 Conclusion

It has been shown in this chapter that, for the system as it is, a standard Kalman filter is a very good solution to reduce the noise and increase the accuracy of the localisation. Some preparatory work has been realised concerning the implementation of an Extended Kalman filter but it will need more measurable variables - like the orientation for example - to become a better alternative.

Chapter 6

Conclusion and future work

This work has shown the development process to implement a real-time locating system on an autonomous robot. This was decomposed into three major phases. The first one was the relative measurement phase which focused on the direct beacon-tag distance measurements. This has been done using the Symmetric Double-Sided Two-Way Ranging technique. This phase mainly focused on identifying sources of error and defining the system prerequisites in order to avoid those errors as much as possible. The devices were programmed with a robust logic to allow proper operation even when a lot of communication errors occur. The remaining error in the system is software corrected with a polynomial correction function.

The second phase studied the computation of the absolute localisation of the robot which is obtained through the use of a trilateration algorithm. This algorithm uses the combination of range measurements from different beacons to compute a position. In order to check the quality of this algorithm, data obtained during experimental measurements where compared with the theoretical minimal obtainable variance represented by the Cramer-Rao Lower Bound.

Finally, the third phase consisted of analysing the possibility of noise reduction through the use of a Kalman filter. This allowed noise reduction and smoothing of the measured trajectory of the robot while it was following a line. Further accuracy improvement solutions where investigated with the usage of Extended Kalman filter. Unfortunately, they did not brought any improvement when applied to the system as it is.

The system obtained at the end of this work shows a pretty decent accuracy compared to the rating of other similar systems commercially available[17]. Additionally, it is strongly believed that the accuracy of this system can still be improved with the addition of a few features.

- Increasing the dimensions of the state model should allow a better tracking with the Kalman filter as it will provide more information about the system. This can be done either by adding the measurement of variables to the system - like the orientation for example -, either by including the system inputs U_k to the model and by extension to the Kalman filter.
- In a more global point of view, location accuracy in a more wide area and/or with other beacon positions should be studied and compared with actual results to see if this system does not lose its advantages in those situations.
- The robotics cup could be a great opportunity to evaluate the system in a real situation. Moreover, this devices could be installed on both robots competing during the match allowing detection and tracking of the opponents. This should ease the collision avoidance during competition.

Bibliography

[1] Decawave website. https://www.decawave.com/products/dwm1000-module. Accessed: 2018-04-21.

[2] Cisco. *Wi-Fi Location-Based Services 4.1 Design Guide*, 2014.

[3] DecaWave. *Sources of error in DW1000 based two-way ranging (TWR) schemes*, 2014. Ver. 1.0.

[4] Jense Defraye. *Determining the position of sporters using ultra-wideband indoor localization*. Master's thesis, Universiteit Gent, 2017.

[5] Electronic Design. *What's The Difference Between Measuring Location By UWB, Wi-Fi, and Bluetooth?*, 2015.

[6] Eurobot. *Robot cities; Rules 2018; 25th edition of robotic contests*, 2018.

[7] Mohinder Grewal and Angus Andrews. Kalman filtering: theory and practice using matlab. 14, 01 2001.

[8] Information technology – Real-time locating systems (RTLS) – Part 1: Application programming interface (API). Standard, International Organization for Standardization, Geneva, CH, February 2014.

[9] Maissa Ben JamÃ¢a, Anis KoubÃ¢a, and Yasir Kayani. Easyloc: Rss-based localization made easy. *Procedia Computer Science*, 10:1127 – 1133, 2012. ANT 2012 and MobiWIS 2012.

[10] Maged N. Kamel Boulos and Geoff Berry. Real-time locating systems (rtls) in healthcare: a condensed primer. *International Journal of Health Geographics*, 11(1):25, Jun 2012.

[11] A. Malik. *RTLS For Dummies*. Wiley, 2009.

[12] Faranak Nekoogar. *Ultra-wideband Communications: Fundamentals and Applications*. Prentice Hall Press, Upper Saddle River, NJ, USA, first edition, 2005.

[13] Office of Engineering and Technology Federal Communications Commission. *Understanding the FCC regulations for low-power, non-licensed transmitters*, 1993.

[14] R. P and M. L. Sichitiu. Angle of arrival localization for wireless sensor networks. In *2006 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks*, volume 1, pages 374–382, Sept 2006.

[15] Amanda Stella Markowska Prorok. *Models and Algorithms for Ultra-Wideband Localization in Single- and Multi-Robot Systems*. Master's thesis, Ecole Polytechnique Fédérale de Lausanne, 2013.

[16] T. Val R. Dalce and A. Bossche. Comparison of indoor localization systems based on wireless communications. *Wireless Engineering and Technology*, 2(4):240–256, 2011.

[17] Brian Ray. Real-time location systems: An overview of 5 technologies, January 2017. https://www.airfinder.com/blog/rtls-technologies/real-time-location-systems-overview-technologies.

[18] Juan Carlos Rodríguez Silva. *Theoretical Study of an RF Positioning System Using Phase Synchronized Anchor Nodes*. Master's thesis, Universitat Politècnica de Catalunya, 2011.

[19] Time Domain. *PulsON Ranging & Communications Part Two : UWB Definition and Advantages*, 2012.

[20] G. Wang, H. Chen, Y. Li, and M. Jin. On received-signal-strength based localization with unknown transmit power and path loss exponent. *IEEE Wireless Communications Letters*, 1(5):536–539, October 2012.

[21] Greg Welch and Gary Bishop. An introduction to the kalman filter, 1995.

[22] Y. Zhou. An efficient least-squares trilateration algorithm for mobile robot localization. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3474–3479, Oct 2009.

Appendix A

Errors in two-way ranging

For two-way ranging systems, the error in the measurement of the time of flight can be demonstrated as follows[3].

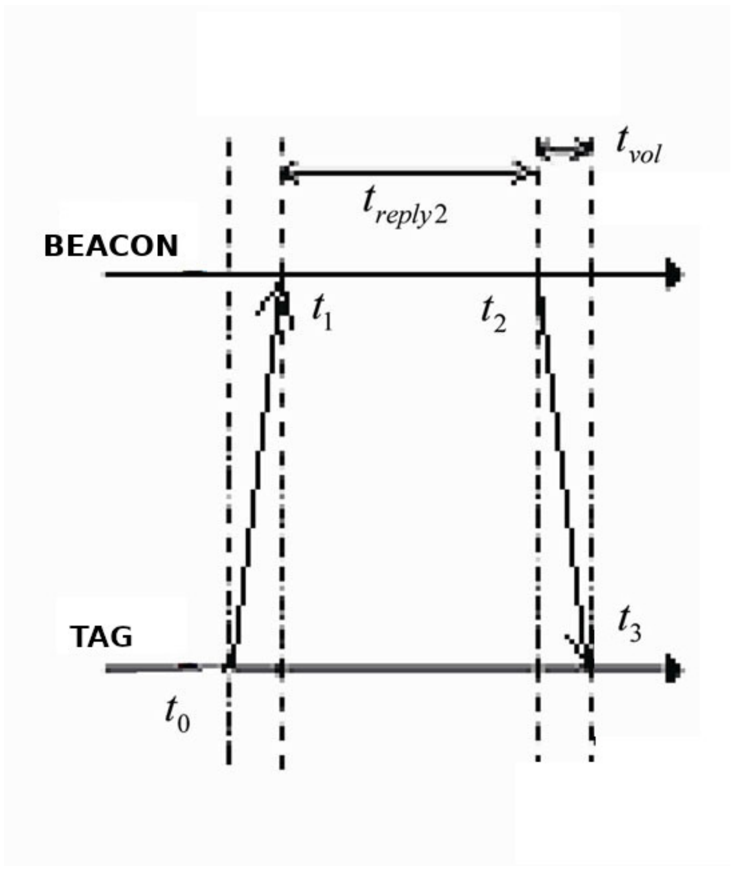


Figure A.1: Two-way ranging[16]

Let's first define T_{round} and T_{reply} as $T_3 - T_0$ and $T_2 - T_1$ respectively. The time of flight is computed as follows :

$$\begin{aligned} T_{round} &= 2 * TOF + T_{reply} \\ 2 * TOF &= T_{round} - T_{reply} \end{aligned} \tag{A.1}$$

If the clock drifts are included in this equation, it becomes :

$$2 * T\hat{O}F = T_{round}(1 + e_A) - T_{reply}(1 + e_B) \tag{A.2}$$

Where e_A and e_B are the errors on time measurement due to clock drift on each device. The error is the difference between the estimated $T\hat{O}F$ and the true TOF.

$$\begin{aligned} 2 * error &= 2 * TOF - 2 * T\hat{O}F \\ &= T_{round}(1 + e_A) - T_{reply}(1 + e_B) - (T_{round} - T_{reply}) \end{aligned} \tag{A.3}$$

Simplifying this, it becomes :

$$2 * error = T_{round}\, e_A - T_{reply}\, e_B \tag{A.4}$$

Finally, substituting T_{round} from equation A.1 yields the final error :

$$\begin{aligned} 2 * error &= 2 * TOF\, e_A + T_{reply}\, e_A - T_{reply}\, e_B \\ error &= TOF\, e_A + \frac{1}{2} T_{reply}(e_A - e_B) \end{aligned} \tag{A.5}$$

Appendix B

Errors in symmetrical double-sided two-way ranging

The error in time of flight measurement in sds-twr systems is computed as follows[3].

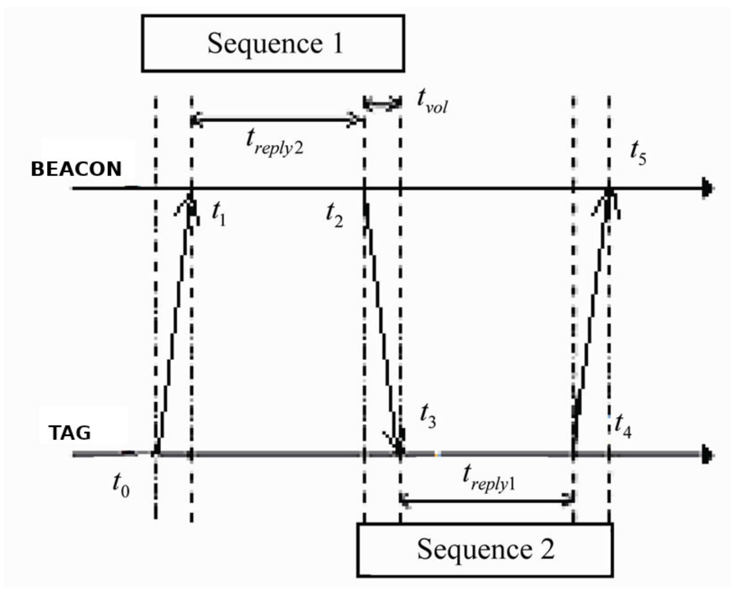


Figure B.1: Symmetric double sided Two-way ranging[16]

As already said, sds-twr can be seen as a two-way ranging occurring on each device. This leads to the following equation :

$$\begin{aligned} T_{roundA} &= 2 * TOF + T_{replyB} \\ T_{roundB} &= 2 * TOF + T_{replyA} \end{aligned} \tag{B.1}$$

From this, the time of flight is computed

$$4 * TOF = T_{roundA} - T_{replyB} + T_{roundB} - T_{replyA} \tag{B.2}$$

Introducing clock drift, it becomes

$$4 * T\hat{O}F = (T_{roundA} - T_{replyA})(1 - e_A) + (T_{roundB} - T_{replyB})(1 - e_B) \tag{B.3}$$

The difference between the estimated TOF and the true TOF gives the error

$$4 * error = (T_{roundA} - T_{replyA})e_A + (T_{roundB} - T_{replyB})e_B \tag{B.4}$$

Lets assume than

$$\begin{aligned} T_{replyA} &= T_{reply} \\ T_{replyB} &= T_{reply} + \Delta T_{reply} \end{aligned} \tag{B.5}$$

Then equation B.1 becomes

$$\begin{aligned} T_{roundA} &= 2 * TOF + T_{reply} + \Delta T_{reply} \\ T_{roundB} &= 2 * TOF + T_{reply} \end{aligned} \tag{B.6}$$

Substituting the two previous results in equation B.4, the final expression of the error is obtained.

$$\begin{aligned} 4 * error &= (2 * TOF + \Delta T_{reply})e_A + (2 * TOF - \Delta T_{reply})e_B \\ error &= \frac{1}{2}TOF(e_A + e_B) + \frac{1}{4}\Delta T_{reply}(e_A - e_B) \end{aligned} \tag{B.7}$$

Appendix C

Trilateration algorithm

This trilateration algorithm focusses on solving the equation 4.11 for a value of $S(p_0)$ which is minimum.

$$S(p_0) = \sum_{i=1}^{N} \left[(p_i - p_o)^T (p_i - p_o) - r_i^2 \right]^2 \tag{C.1}$$

Where p_i represents the position of the ith beacon, r_i is the distance measured by the beacon and N is the amount of beacon.

Finding this minimum value is equivalent to solve

$$\frac{\partial S(p_0)}{\partial p_0} = a + Bp_0 + [2p_0 p_0^T + (p_0^T p_0)I]c - p_0 p_0^T p_0 = 0 \tag{C.2}$$

Where

$$\begin{aligned} a &= \frac{1}{N} \sum_{i=1}^{N} (p_i p_i^T p_i - r_i^2 p_i) \\ B &= \frac{1}{N} \sum_{i=1}^{N} [-2p_i p_i^T - (p_i^T p_i)I + r_i^2 I] \\ c &= \frac{1}{N} \sum_{i=1}^{N} p_i \end{aligned} \tag{C.3}$$

This resolution is done with the following steps

$$f = a + Bc + 2cc^T c \tag{C.4}$$

$$H = -\frac{2}{N}\sum_{i=1}^{N} p_i p_i^T + 2cc^T \tag{C.5}$$

$$q = -H^{-1} f \tag{C.6}$$

And finally

$$p_0 = q + c \tag{C.7}$$