

ULTRA WIDEBAND LOCATION IN SCENARIOS WITHOUT CLEAR LINE OF SIGHT: A PRACTICAL APPROACH

Valentín Barral Vales

Doctoral Thesis UDC / 2019

Advisor: Carlos J. Escudero Cascón

PhD Program in Information Technology and Mobile Network Communication



LOCALIZACIÓN CON ULTRA WIDEBAND EN ESCENARIOS SEN LIÑA DE VISIÓN DIRECTA: UN ENFOQUE PRÁCTICO

Valentín Barral Vales

Tese de Doutoramento UDC / 2019

Director: Carlos J. Escudero Cascón

Programa de Doutoramento en Tecnoloxías da Información e das Comunicaci3ns en
Redes M3viles

LOCALIZACIÓN CON ULTRA WIDEBAND EN ESCENARIOS SIN LÍNEA DE VISIÓN DIRECTA: UN ENFOQUE PRÁCTICO

Valentín Barral Vales

Tesis Doctoral UDC / 2019

Director: Carlos J. Escudero Cascón

Programa de Doctorado en Tecnologías de la Información y de las Comunicaciones en
Redes Móviles

Valentín Barral Vales

CERTIFICA / CERTIFICA / CERTIFIES


Que a presente memoria é o resultado do meu propio traballo de investigación e que o traballo doutros autores está citado axeitadamente.

Que la presente memoria es el resultado de mi propio trabajo de investigación y que el trabajo de otros autores está citado apropiadamente.

That the present report is the result of my own research work and that the work done by other authors is appropriately cited.

A Coruña, outubro de 2019 / octubre de 2019 / October 2019

Valentín Barral Vales

Código Seguro De Verificación	/wvmolbfz0KhDqqXpb5iSA==	Estado	Data e hora	
Asinado Por	Valentín Barral Vales	Asinado	30/10/2019 11:19:27	
Observacións		Páxina	1/1	
Url De Verificación	https://sede.udc.gal/services/validation//wvmolbfz0KhDqqXpb5iSA==			

Carlos J. Escudero Cascón

CERTIFICA / CERTIFICA / CERTIFY

Que a presente tese titulada “Localización con Ultra Wideband en escenarios sen liña de visión directa: un enfoque práctico” foi realizada por Valentín Barral Vales baixo a miña dirección no Departamento de Enxeñaría de Computadores da Universidade da Coruña e preséntase para obter o grao de Doutor.

Que la presente tesis titulada “Localización con Ultra Wideband en escenarios sin línea de visión directa: un enfoque práctico” fue realizada por Valentín Barral Vales bajo mi dirección en el Departamento de Ingeniería de Computadores de la Universidad de A Coruña y se presenta para obtener el grado de Doctor.

That the present thesis titled “Ultra Wideband location in scenarios without clear line of sight: a practical approach” was done by Valentín Barral Vales under my supervision in the Department of Computer Engineering at the University of A Coruña and it is submitted to obtain the Ph.D. degree.

O director da tese / *El director de la tesis / The Ph.D. supervisor.*
A Coruña, outubro de 2019 / *octubre de 2019 / October 2019.*

Dr. Carlos J. Escudero Cascón
Profesor titular
Dpto. de Enxeñaría de Computadores
Universidade da Coruña
*Associate Professor
Dept. of Computer Engineering
University of A Coruña*

Código Seguro De Verificación	o1N//m2jyoRBAAYJPLSHUQ==	Estado	Data e hora
Asinado Por	Carlos José Escudero Cascón	Asinado	30/10/2019 11:22:04
Observacións		Páxina	1/1
Url De Verificación	https://sede.udc.gal/services/validation/o1N//m2jyoRBAAYJPLSHUQ==		



Tese de Doutoramento / *Tesis Doctoral / Doctoral Thesis*

Título:	Localización con Ultra Wideband en escenarios sen liña de visión directa: un enfoque práctico
Título:	<i>Localización con Ultra Wideband en escenarios sin línea de visión directa: un enfoque práctico</i>
Title:	<i>Ultra Wideband location in scenarios without clear line of sight: a practical approach</i>
Autor / Autor / Author:	Valentín Barral Vales
Directores / Directores / Supervisors:	Carlos J. Escudero Cascón
Data / Fecha / Date:	outubro de 2019 / <i>octubre de 2019 / October 2019</i>

Tribunal / Tribunal / Evaluation Committee

Presidente / Presidente / President:	José Ramiro Martínez de Dios
Vogal / Vocal / Member:	Leyre Azpilicueta Fernández
Secretaria / Secretaria / Secretary:	Luis Castedo Rivas

To my family and friends

Acknowledgements

The first one I want to thank for his work and support is Carlos, my thesis advisor. Without his support and dedication this work would not have been possible. I also want to thank you for the opportunities you have given me over the years to work on many different projects from which I have drawn an invaluable experience for my career.

I would also like to thank Jose Antonio for all his efforts in helping me prepare this thesis. Thank you for your suggestions and new ideas and for helping me to put them into practice in the best possible way.

Another important piece that has helped me to get several of the results presented in this platform is Pedro. Thank you for your advice, your work and your ability to make me understand those structures of various dimensions that caused me so many headaches.

I must include in the acknowledgments to many of the members of the GTEC who during these last years have helped me to finish this work. Thank you Xosé for your always explicit support, thank you Tomás for your help with the darkest parts of Matlab™, thank you Javi for your advice and for always being ready to lend a hand, thank you Dario for your help in bringing to fruition the last experiments of this thesis.

I want to extend my thanks also to all the people of the GTEC who have accompanied me all these years, starting with Luis Castedo, without whose management of the group this thesis would not have taken place either. Thanks also to Cris for helping me these years with the bureaucracy and the eternal paperwork of the university. And thanks also to the rest of the GTEC members who have passed through here and who I have had the pleasure of meeting (sorry if I forget someone): Adriana, Adriano, Aida, Ángel, Belén, Dani, Darian, Diego, Fran, Fran Laport, Hector, Ismael, Iván, Jose, Jose Francisco, Josmary, Julio, Miguel, Manuel, Marc, Nestor, Oscar Blanco, Oscar Fresnedo, Paula Castro, Paula Fraga, Roberto, Santi and Tiago.

Finally, I want to remember my family, my friends, and Silvia, who have accompanied me these years and without whom I would not have gotten that far. And I also want to leave a final memory for my little friend Nana, who will never be forgotten.

Resumo

A localización en interiores sufriu dun importante empuxe nos últimos anos. Os servizos baseados en localización (location based services (LBS)) que ata fai pouco estaban restrinxidos a escenarios exteriores e a o uso do GPS, foron estendéndose tamén cara o interior dos edificios. Dende grandes estruturas públicas como aeroportos ou hospitais ata multitude de escenarios industriais, os LBS están cada vez mais presentes baixo teito.

De entre as diversas tecnoloxías que poden ser utilizadas para acadar esta localización en interiores, as baseadas en sinais ultra-wideband (UWB) convertéronse nunha das mais demandadas debido fundamentalmente á súa precisión na estimación de posición. Adicionalmente, a aparición no mercado de cada vez mais fabricantes e mais produtos fixo diminuír os prezos destes dispositivos ata niveis que permiten pensar no seu uso para grandes despregues con un presuposto contido.

Pola súa natureza, os sinais UWB son moi resistentes ao fenómeno de multi-traxecto, polo que nunha situación de boa visión directa entre os dous dispositivos (LOS) a tecnoloxía é capaz de proporcionar estimación de distancia moi precisas. Isto non ocorre así cando esta liña de de visión está total ou parcialmente bloqueada (NLOS). Neste caso os posibles rebotes do sinal poden ser erroneamente confundidos co sinal orixinal e derivar nunha estimación de tempo, e por tanto distancia, moi alonxada do seu valor real.

O obxectivo desta tese é o de ofrecer una solución completa ao problema da localización en interiores con UWB dende un enfoque práctico. Isto quere dicir que se aborda o problema dende todos os ángulos que se deberían ter en conta nun caso de uso real: a investigación básica en busca dunha mellora a nivel técnico dos valores de UWB, a simulación previa para garantir un bo despregue e rendemento e a integración final da solución dentro dunha operativa externa que permita a LBS ou clientes poder facer uso dos datos de localización.

A primeira parte desta proposta presentase nos primeiros capítulos da tese, e consiste en utilizar algoritmos de aprendizaxe automática (machine learning (ML)) para detectar e mitigar as posibles medidas NLOS xeradas por dispositivos UWB de baixo custo. Para esta tarefa móstranse as diferentes campañas de medida con dispositivos reais que se realizaron para obter os datos de adestramento dos algoritmos, e como estes se empregaron en diferentes experimentos para medir ata que punto a solución proposta era válida nun entorno real.

A segunda parte da tese presenta pola súa parte os datos dun simulador de dispositivos UWB orientado á localización creado para poder realizar probas e detectar posibles problemas antes de efectuar un despregue con dispositivos reais. Entre os datos presentados durante esta segunda parte móstranse diferentes comparativas entre escenarios reais e simulados para comprobar como de fidedigna é a simulación. Ademais, preséntase o simulador integrado dentro dun caso de uso habitual nos sistema de localización en interiores: a localización de vehículos nun entorno industrial utilizando múltiples sensores.

Por último, a terceira e derradeira parte desta tese describe unha plataforma de localización en tempo real (real-time location system (RTLS)) multi-tecnoloxía capaz de dar servizo aos diferentes actores habituais nun sistema de localización real. A plataforma serve de punto de unión entre os propios sensores de localización, os LBS e clientes finais, os investigadores encargados da algorítmica de localización e os xestores de servizos derivados como alertas. Mediante a plataforma RTLS proposta nesta terceira parte da memoria lograse desacoplar a funcionalidade destas entidades de forma que todas elas poden traballar sen depender das demais.

Ademais da descrición da plataforma, inclúese tamén un resumo dos proxectos reais de localización nos que esta foi utilizada, así como un listado das diversas modificación e melloras que a plataforma sufriu nos últimos tempos grazas precisamente as experiencias obtidas tras estes traballos sobre o terreo con proxectos reais.

Resumen

La localización en interiores ha sufrido un importante empuje en los últimos años. Los servicios basados en localización (location based services (LBS)) que hasta hace poco estaban restringidos a escenarios exteriores y al uso del GPS, han ido extendiéndose también hacia el interior de los edificios. Desde grandes estructuras públicas como aeropuertos o hospitales hasta multitud de escenarios industriales, los LBS han pasado a estar cada vez más presentes en escenarios bajo techo.

De entre las diversas tecnologías que pueden utilizarse para conseguir esta localización en interiores, las basadas en señales ultra-wideband (UWB) se han convertido en una de las más demandadas debido fundamentalmente a su precisión en la estimación de posición. Adicionalmente, la aparición en el mercado de cada vez más fabricantes y más productos ha hecho disminuir los precios de estos dispositivos hasta niveles que permiten pensar en su uso para grandes despliegues con un presupuesto contenido.

Por su naturaleza, las señales UWB son muy resistentes al fenómeno de multi trayecto, por lo que en una situación de buena visión directa entre dos dispositivos (line-of-sight (LOS)) la tecnología es capaz de proporcionar estimaciones de distancia muy precisas. Esto no ocurre así cuando esta línea de visión está total o parcialmente bloqueada (non-line-of-sight (NLOS)). En este caso los posibles rebotes de la señal pueden ser erróneamente confundidos con la señal original y derivar en una estimación de tiempo, y por tanto distancia, muy lejana al valor real.

El objetivo de esta tesis es ofrecer una solución completa al problema de la localización en interiores con UWB desde un enfoque práctico. Esto quiere decir que se aborda el problema desde todos los ángulos que se deberían tener en cuenta en un caso de uso real: la investigación básica en busca de una mejora a nivel técnico de los valores UWB, la simulación previa para garantizar un buen despliegue y rendimiento y la integración final de la solución dentro de una operativa externa que permita a LBS o clientes poder hacer uso de los datos de localización.

La primera parte de esta propuesta se presenta en esta tesis en los primeros capítulos, y consiste en utilizar algoritmos de aprendizaje automático (machine learning (ML)) para detectar y mitigar las posibles medidas NLOS generadas por dispositivos UWB de bajo coste. Para esta tarea se muestran las diferentes campañas de medida con hardware real que se realizaron para obtener los datos de entrenamiento de los algoritmos, y cómo estos se utilizaron en diferentes experimentos para medir hasta qué punto la solución planteada era válida en un entorno real.

La segunda parte de la tesis presenta por su parte los datos de un simulador de dispositivos UWB orientado a la localización creado para poder realizar pruebas y detectar posibles problemas antes de efectuar un despliegue con hardware real. Entre los datos presentados durante esta segunda parte se muestran diferentes comparativas entre escenarios reales y simulados para comprobar cómo de fidedigna es la simulación. Además, se presenta el simulador integrado dentro de un caso de uso habitual en los sistemas de localización en interiores: la localización de vehículos en un entorno industrial utilizando múltiples sensores.

Por último, la tercera y última parte de esta tesis describe una plataforma de localización en tiempo real (real-time location system (RTLS)) multi-tecnología capaz de dar servicio a los diferentes actores habituales en un sistema de localización real. La plataforma sirve de nexo de unión entre los propios sensores de localización, los LBS y clientes finales, los investigadores encargados de la algorítmica de localización y los gestores de servicios derivados como alertas. Mediante la plataforma RTLS propuesta en esta tercera parte de la memoria se logra desacoplar la funcionalidad de estas entidades de forma que todas ellas puedan trabajar sin dependencias con las demás.

Además de la descripción de la plataforma, se incluye también un resumen de los proyectos reales de localización en los que esta ha sido utilizada, así como un listado de las diversas modificaciones y mejoras que la plataforma ha sufrido en los últimos tiempos gracias precisamente a las experiencias obtenidas tras esos trabajos sobre el terreno con proyectos reales.

Abstract

Indoor location has experienced a major boost in recent years. Location based services (LBS), which until recently were restricted to outdoor scenarios and the use of GPS, have also been extended into buildings. From large public structures such as airports or hospitals to a multitude of industrial scenarios, LBS has become increasingly present in indoor scenarios.

Of the various technologies that can be used to achieve this indoor location, the ones based on ultra-wideband (UWB) signals have become ones of the most demanded due primarily to their accuracy in position estimation. Additionally, the appearance in the market of more and more manufacturers and products has lowered the prices of these devices to levels that allow to think about their use for large deployments with a contained budget.

By their nature, UWB signals are very resistant to the multi-path phenomenon, so in a situation of good direct vision between two devices (line-of-sight (LOS)) the technology is able to provide very accurate distance estimates. This is not the case when this line of sight is totally or partially blocked (non-line-of-sight (NLOS)). In this case the possible rebounds of the signal can be wrongly confused with the original emitted signal and result in an estimate of time, and therefore distance, very far from the actual value.

The aim of this thesis is to offer a complete solution to the problem of indoor location with UWB from a practical approach. This means that the problem is approached from all the angles that should be taken into account in a real use case: basic research in search of an improvement at technical level of the UWB values, prior simulation to ensure a good deployment and performance and the final integration of the solution within an external operational system that allows LBS or clients to make use of location data.

The first part of this proposal is presented in this thesis in the first chapters, and consists of using machine learning (ML) techniques to detect and mitigate the possible NLOS measurements generated by low-cost UWB devices. For this task the memory describes the different measurement campaigns with real hardware that were carried out to obtain the training data of the algorithms, and how these were used in different experiments to measure to what extent the proposed solution was valid in a real environment.

The second part of the thesis presents data from a location-oriented UWB devices simulator created to perform tests and detect possible problems before deploying real hardware. Among the data presented during this second part, different comparisons between real and simulated scenarios are shown in order to check how reliable the simulation is. In addition, the simulator is presented as part of a case commonly used in indoor location systems: the location of vehicles in an industrial environment using multiple sensors.

Finally, the third and final part of this thesis describes a multi-technology real-time location system (RTLS) capable of serving the different actors in a real location system. The platform serves as a link between the location sensors themselves, the LBS and end clients, the researchers in charge of the

location algorithms and the managers of derived services such as alerts. By means of the proposed RTLS platform it is possible to decouple the functionality of these entities so that all of them can work without dependencies with the others.

In addition to the description of the platform, this memory also includes a summary of the real localization projects in which it has been used, as well as a list of the various modifications and improvements that the platform has undergone in recent times thanks exactly to the experiences obtained after those works on site with real projects.

Table of contents

1	Introduction	1
1.1	Ultra Wideband	3
1.2	DW1000 sources of error	4
1.3	NLOS detection and mitigation: Machine Learning	11
1.4	A practical approach: simulation	14
1.5	A practical approach: Accessing location information	14
1.6	Contributions related with the chapter	15
2	Machine learning features selection	17
2.1	Hardware used	18
2.2	Scenario description	19
2.3	Machine Learning	21
2.4	Results	22
2.5	Conclusions	24
2.6	Contributions related with the chapter	25
3	Impact of Non-Line of Sight in localization	27
3.1	Measurement campaign	28
3.2	Simulator	29
3.3	Location Algorithms	31
3.3.1	Linear Least Squares	31
3.3.2	Nonlinear Least Squares	32
3.3.3	Iterative Extended Kalman Filter	33
3.4	Results	34
3.5	Conclusions	37
3.6	Contributions related with the chapter	38
4	Non-Line of Sight detection and mitigation	39
4.1	Measure campaign	40
4.2	Machine Learning	41
4.2.1	Implemented algorithms	42

4.2.2	Features used	43
4.2.3	Bayesian Optimization	44
4.2.4	Discrete Measuring Points	44
4.3	Results	45
4.3.1	Results Classification	45
4.3.2	Mitigation Results	47
4.4	Conclusions	49
4.5	Contributions related with the chapter	50
5	Cross validation of the location system	51
5.1	Measurement Campaigns	52
5.1.1	Hardware used	52
5.1.2	Measurement and test scenarios	53
5.2	Classification and mitigation algorithms	54
5.2.1	Multi-layer perceptron	54
5.3	Location algorithms	54
5.4	Description of experiments	55
5.5	Results	57
5.6	Conclusions	61
5.7	Contributions related with the chapter	62
6	Simulation of Ultra Wideband based location devices	63
6.1	Problem statement: Locating pallets	64
6.1.1	The proposed solution	65
6.2	The chosen sensors	66
6.3	Simulation platform	66
6.3.1	Simulated Scenarios	67
6.3.2	Simulated Vehicle and Sensors	68
6.4	UWB simulation	69
6.4.1	Simulation accuracy	71
6.5	Location Algorithm	72
6.6	Software Implementation	76
6.6.1	ROS Nodes	76
6.6.2	ROS Custom Messages	78
6.7	Results	78
6.8	Simulation vs real world	83
6.9	Conclusions and future lines	87
6.10	Contributions related with the chapter	88

7	Multi-technology Real Time Location System	91
7.1	Multi-technology RTLS platform	93
7.1.1	Platform objectives	93
7.1.2	Main actors	95
7.1.3	Platform Architecture	96
7.1.4	Location Server	97
7.1.5	Protection and Security	99
7.2	The platform in real projects	99
7.3	Platform Upgrades	101
7.3.1	Internationalization	101
7.3.2	Target device	101
7.3.3	Roles mechanism	102
7.3.4	MQTT	103
7.3.5	Subscription Services	104
7.3.6	New map model	105
7.3.7	Semantic positions	106
7.3.8	Alerts	107
7.4	Conclusions	108
7.5	Contributions related with the chapter	109
8	Conclusions and future work	111
8.1	Conclusions	111
8.1.1	NLOS Classification and mitigation	112
8.1.2	UWB simulation	113
8.1.3	RTLS platform	114
8.2	Future Work	114
8.3	Contributions	115
8.3.1	Projects	116
8.3.2	Journal papers	117
8.3.3	Conference papers	117
8.3.4	Public datasets	118
8.3.5	Technological-base company foundation	118
	Appendices	120
A	Ultra Wideband and IEEE 802.15.4-2011	121
A.1	The IEEE 802.15.4-2011 standard	122
A.1.1	UWB Impulse Radio	123
A.1.2	802.15.4-2011 UWB Physical layer	125
A.1.3	802.15.4-2011 UWB MAC layer	128

A.2	UWB regulation in United States and European Union	129
B	DW1000 transceiver	133
B.1	DW1000 implementation of IEEE 802.15.4-2011 UWB Physical Layer	133
B.2	DW1000 implementation of IEEE 802.15.4-2011 MAC Layer	135
C	Resumen de la tesis	137
C.1	Clasificación y mitigación de medidas UWB usando hardware de bajo coste y algoritmos de machine learning	139
C.1.1	Capítulo 2: Selección de características de interés para Machine Learning	140
C.1.2	Capítulo 3: Análisis del impacto de NLOS en los resultados de localización	141
C.1.3	Capítulo 4: Detección y mitigación NLOS	141
C.1.4	Capítulo 5: Validación cruzada	142
C.2	Simulación de dispositivos UWB para localización	143
C.3	Plataforma RTLS multi-tecnología: diseño y evolución	144
D	List of Acronyms	145
	References	149

List of figures

1.1	First setup: raw ranging error over the total receive power considering multiple orientations for the tag.	6
1.2	First setup: mean raw ranging error versus the total receive power for different tag orientations, grouped by measurement points.	6
1.3	Second setup: raw ranging error over the total receive power considering a single tag orientation.	7
1.4	Second setup: mean raw and corrected ranging error versus distance between the tag and the anchor.	8
1.5	Theoretical two-ray propagation model applied to the measurement scenario.	8
1.6	Raw ranging error with respect to the difference between the total receive power and that of the signals corresponding to the first-path.	9
1.7	channel impulse response (CIR) Power of a channel in a LOS situation	10
1.8	CIR Power of a channel in a NLOS situation	11
1.9	Machine learning approach to the NLOS classification-mitigation problem.	13
2.1	Experiment Configuration	20
2.2	Measurements obtained in the three scenarios	21
2.3	Success rate LOS vs NLOS	23
2.4	Success rate NLOS-S vs NLOS-H	24
2.5	Success index with sets of features	25
3.1	Measurement scenario	28
3.2	Tripod with measurement tags.	28
3.3	Anchor shifting	30
3.4	Mean error in rectangular path	36
3.5	Mean error in random trajectory	36
3.6	Mean error using IEKF and NLOS error estimation	37
4.1	Block diagram of a location system based on ranging measurements	40
4.2	Estimate vs. actual distance.	41
4.3	RSS vs real distance. Raw measurements.	42
4.4	F_1 -score with jump factor $j = 1$	46

4.5	F_1 -score vs. jump factor, j	46
4.6	Mitigation mean absolute error (MAE) with 3 classes.	48
4.7	Mitigation MAE with 2 classes.	49
5.1	Block diagram of a location system based on ranging measurements	51
5.2	Test Scenario.	53
5.3	Classification results using neural networks (NN).	56
5.4	empirical cumulative distribution function (ECDF) of the location error using iterative extended Kalman filter (IEKF).	58
5.5	Localization MAE using IEKF.	59
5.6	ECDF of the location error using nonlinear least squares (NLS)	60
5.7	Localization MAE using NLS.	61
6.1	The two simulated scenarios.	67
6.2	The color of the anchors indicates their state: green - LOS, yellow - NLOS <i>Soft</i> , blue - NLOS <i>Hard</i> , and red - out of range.	71
6.3	Measurement campaign scenario —described in Chapter 4— and its replica in Gazebo.	72
6.4	Simulated ranging values corresponding to the measurement scenario and comparison with the measured ranging values.	72
6.5	Simulated RSS values corresponding to the measurement scenario and comparison with the measured RSS values.	73
6.6	Mean and standard deviation (mean $\pm\sigma$) of the measured and simulated ranging values shown in Fig. 6.4 for the three different situations considered.	74
6.7	Sensor placement on the forklift.	75
6.8	Sensor placement in detail. A: UWB tag, B: PX4 Flow.	76
6.9	Trajectories using different sensors in Scenario A.	80
6.10	MAE of position estimation in Scenario A.	80
6.11	ECDF of position error in Scenario A.	81
6.12	Trajectories using different sensors in Scenario B.	81
6.13	MAE of position estimates in Scenario B.	82
6.14	ECDF of position error in Scenario B.	82
6.15	Grid simulation scenario	83
6.16	ECDF of the location error using IEKF.	84
6.17	Localization MAE using IEKF.	84
6.18	ECDF of the location error using NLS	85
6.19	Localization MAE using NLS.	85
7.1	UML class diagram of the system nodes, sensors, networks, <i>etc.</i>	94
7.2	Proposed architecture for Hybrid Real-Time Location Systems.	97

7.3	New target device entity.	102
7.4	Upgrades in the RTLS platform architecture. Modifications are colored in green.	104
7.5	Different scenarios with heterogeneous floor configurations.	105
8.1	Author contributions during the work on the thesis	116
A.1	UWB PHY Frame	126
A.2	Two way ranging	128
A.3	Power mask US	130
A.4	Power mask EU	130
B.1	Two way ranging in DW1000	134

List of tables

2.1	Feature Sets	24
6.1	Plug-in <i>gtec_uwb_plugin</i> . The <i>id</i> in the topic route corresponds to the tag identifier.	77
6.2	Plug-in <i>gtec_tag_pos_publisher</i>	77
6.3	Node <i>gazebo2ros</i>	77
6.4	Node <i>kfpos</i> . The <i>id</i> in the UWB range topic route corresponds to the tag identifier.	77
6.5	Real vs Simulated MAE comparison using IEKF.	86
6.6	Real vs Simulated MAE comparison using NLS.	86
8.1	Impact factor of journals.	117
A.1	UWB physical layer (PHY) Channel definitions in IEEE 802.15.4-2011.	126
A.2	Length 31 preamble codes.	127
A.3	LDC limits	131
B.1	DW1000 Allowed Channels	134
B.2	TX per second at 18% air-utilization	136

Chapter I

Introduction

The field of indoor location has always been a place of debate and proposals. The heterogeneous conditions of this type of environment means that there is no single solution capable of efficient and robust performance in all scenarios. A localization process needs three different entities: a target whose position is unknown, a system of coordinates of reference on which to provide this position and a more or less complex localization algorithm that, based on the measure of certain physical parameters of the environment or from the target, is able to estimate where it is. The physical parameters from which a location algorithm can infer something about the position of a target are many and of a very different nature. Values of magnetic field, luminosity, acceleration, time of propagation of sound waves or radio signals, angles of incidence of a signal on an array of antennas, changes in the chemistry of the air, intensity of an RF signal to reach its destination, temperature, atmospheric pressure, changes in the image captured by a camera ... The possibilities are almost endless.

From an organizational point of view, indoor location systems could be categorized into two types: those based on sensor networks and those based on autonomous devices. The first ones require a structure of devices deployed by the scenario in fixed and known positions and a device (usually known as *tag*) that must be carried by the object or person to be located. The fixed elements in the scenario are usually defined as *beacons* or *anchors*. Normally in this configuration some radio-frequency (RF) signal is used to measure some kind of physical parameter between the beacons and the *tag*. Typical parameters include the time of arrival (TOA), time difference of arrival (TDOA) or angle of arrival (AOA). The first of these would be the time it takes for a signal to arrive from each beacon to the *tag*, the second would be the difference in arrival times to two beacons of the same signal emitted from a *tag* and the third would be the angle of incidence of the signal emitted by a *tag* to reach a beacon or vice versa.

A lot of RF technologies have been used to feed location systems like that even when they were not initially designed with this goal in mind. Technologies such as RFID [SN10; XU+17; HAS+15; SJ18; BER+18], WiFi [YS15; GQ16; MEN+18], ZigBee [LAR+10; SGD08; HC11] or Bluetooth [MA+17; RID+15; TER+17] can be mentioned in this group. Thought to be used in for communication tasks, their cost and wide deployment in many buildings had them

used (and continue to use) as the basis for many positioning systems. Of course, because their original goal was other than the location, is not easy to get a good accuracy with them. Even with sophisticated algorithms or a deep scenario characterization (i.e. signal fingerprinting) the estimates achieved with these technologies have always several meters of error.

Another methods of positioning do not use a sensors network deployed on the scenario, but they are based in data captured by sensors placed on the *tag*. Thus, these methods use data like values of magnetic field [CAR+15; BAR+16c; AM18] or atmospheric pressure [XIA+15; RAN+18] (although this last type of technology is mainly used to estimate height). In these cases the measurements are very dependent on the specific scenario where the positioning is carried out, so their mission is usually more that of supporting another technology of greater accuracy and precision.

When these accuracy and precision becomes more important, and its need to get errors below the meter or even below a few centimeters, it is necessary to resort to other types of technologies already designed for positioning tasks.

- LIDAR devices employ laser to measure the time between pulses or the changes in the phase of the signal [KUM+17; HUA+17]. This technology is widely used to track vehicles in indoor areas, and has a good accuracy when combined with SLAM [LI+14; BAM17; WAN+18].
- Cameras can also be used for location purposes. They can be used as sources of visual odometry tracking points or lines between frames [GBG16; PUM+17](when embarked on the object to be located) or also be used from the infrastructure with artificial vision techniques [KC15; KUM+16; CRR18]. They can achieve centimeter accuracy [KUM+16; KUM+17], although of course they always need a clear line of sight to work.
- Inertial sensors allow to obtain a precise result, although their behavior degrades over time if they are not periodically recalibrated [TIA+15; HTJ16; ZHA+18; GU+18].
- Infrared systems can be used in different ways to obtain a location with an accuracy of few centimeters. Some virtual reality systems use this technology to locate the player in an small area but with a great accuracy (around 2 cm of mean error). They use a set of infrared (IR) receivers on the headset and two or more sources of IR that emit the signal with different patterns and angles [ZFB10; NLL17]. IR systems can be used also to estimate positions based on the thermal heat, with sensors of this type that can measure the angle of incidence of a heat source [HAU+10].

All these technologies as a whole can solve a specific problem of indoor positioning, provided that the environment is limited and the accuracy requirements do not go beyond the theoretical limit of each of them. Thus, a laser will not work in a glazed area, a WiFi fingerprint will suffer great degradation if the position or power of several APs is modified or a camera location system will fail completely if an object is placed just in front of the lens. In short, indoor positioning is not resolved and each technology has its own limits in terms of accuracy, robustness and characteristics of the environment in which it can operate with guarantees.

This is why the task of locating objects or people in an indoor setting becomes even more complex when the morphology of the environment includes many distorting elements: metal walls, heavy machinery, large flows of people, low ceilings, very wide or very thin walls, etc. All these elements have a clear negative effect on most location technologies based on RF signals, but they also have a negative effect on others based on another physical principle (although in a different way). This is especially important in industrial environments, where the amount and intensity of these distortion focuses is multiplied. In these situations, the solution comes not from a particular technology, but from a combination of them. Using different technologies at the same time on the same targets multiplies costs, but also generates a more robust and reliable system. The impact of the distortions is thus diluted on the different sensors used, since normally the effect is not the same on all of them.

1.1 Ultra Wideband

Among the different location systems based on RF signals, one of the most prominent in the last years is the based on ultra-wideband (UWB) signals. These signals are characterized by their high bandwidth as opposed to narrow-band signals.

Although its signals have been known for decades, its general use for localization tasks began with the approval of the IEEE 802.15.4a standard [KAR+10] in March 2007. This was the starting gun for manufacturers in their race to commercialize the first UWB-based transceiver oriented to ranging tasks. For these positioning tasks, the UWB signals used are built using Impulse Radio. This method is based on the transmission of very short pulses in time, which causes a large bandwidth in frequency but has a very beneficial effect for localization tasks: the temporal granularity is so high that very short time intervals can be quantized. Thus, it is possible to build a positioning system based on the time of flight (TOF) of the signals from the moment they are emitted until they reach their destination. A resolution of few picoseconds is enough to reach a centimeter accuracy. A long and detailed explanation of UWB, impulse radio and the last IEEE 802.15.4-2011 standard (continuation of the IEEE 802.15.4a) can be found in Appendix A.

Today there are many companies whose business model is based on the use of UWB signals for localization. Among the most important are Localino, Sewio, Eliko or Airtls. However, one of the biggest leaps in the popularization of UWB came with Decawave and its DW1000 [DEC19c] module, the first commercial UWB transceiver on a single chip that also had a very competitive price. With its arrival the access to this technology became much simpler, in such a way that nowadays the majority of commercial devices of localization in UWB take this chip or some of its more recent variants.

The DW1000 follows the IEEE802.15.4-2011 standard and supports 6 different RF bands between 3.5 GHz and 6.5 GHz. Among its different operating modes is data transmission with data rates of 110 kbit/s, 850 kbit/s and 6.8 Mbit/s. The module has many configuration

parameters, so it can be adapted to multiple types of problems depending on the needs of each. More information about this chip can be found in Appendix B. The versatility of the DW1000, along with its price, makes it the transceiver chosen in devices and solutions from many manufacturers such as Pozyx.

Pozyx devices [POZ19b] are a low-cost hardware that integrates the DW1000 transceiver plus several additional inertial sensors. Because of their price and flexibility (they can be used easily with Arduino or standalone with a computer running the control scripts), they have become very popular in the indoor-localization field. These devices have been used extensively during the experiments detailed throughout this memory.

1.2 DW1000 sources of error

The performance of devices based on the DW1000 has proven to be very good, at least when the environmental conditions are favorable. When there is a good direct line of sight between the transmitter and receiver and the distance between the two is reduced the average error in the distance estimates is around 0.1 m to 0.2 m.

However, as some works relate [JS16], even in a good line-of-sight (LOS) situation the error variance in the estimates can variate with the distance. This is partly due to a bias error in the estimation of ranging that these chips have, and that is given by the value of received power detected in the receiver. Although at first the ranging and the power estimation of the received signal should be independent entities (since the ranging is calculated by TOA and not by received signal strength (RSS)), in reality the ranging values provided by the DW1000 have a certain error that depends on the received power. In [DEC14] it can be seen as different power values and depending on the configuration mode chosen this error value is different. Thus, after performing a measure process k , the ranging estimate r_k can be modeled as

$$r_k = d_k + B(p_k) + n_k, \quad (1.1)$$

where d_k is the actual distance between the anchor and the tag, $B(\cdot)$ is the bias modeling function, p_k is the total receive power, and n_k is a noise component.

The DW1000 chip itself provides a corrected version of the ranging that tries to compensate for the bias $B(p_k)$, but taking only the raw ranging values into account, regardless of the total receive power. The idea behind this approach is that the total receive power is related to the distance through a path loss model, i.e. $p_k = P_R(d_k)$, with $P_R(\cdot)$ the considered model of the received power as a function of only the actual distance. In particular, DW1000 driver uses a free-space path loss model [DEC14].

$$P_R(d) = P_T + G + 20 \log_{10}(c) - 20 \log_{10}(4\pi f_c d), \quad (1.2)$$

where P_T is the total transmit power, G the antenna gain, c the speed of light, and f_c is the carrier frequency. Consequently, the bias model in Eq. (1.1) changes to $B(P_R(d_k))$, and thus the bias can be assumed to be a function of the distance.

However, in a real-world environment propagation conditions diverge from those expected in a LOS scenario –e.g. obstacles that partially absorb the transmit power, reflections on nearby objects– and corrections based solely on the values of the raw ranging are distorted by the typical features of the signal propagation in multipath indoor environments.

In order to verify the effects that this biased error had on the measurements provided by the DW1000, a series of experiments were performed using actual data from two measurement campaigns in which two EVB1000 devices were used (two development boards including the DW1000 and an external antenna [DEC19a]). The two measurement campaigns took place in an indoor sports hall of 25 m by 45 m with no obstacles between the transmitter and the receiver. Both the tag and the anchor were placed on two tripods at a height of 1.5 m. In order to model the bias, two different setups were used, both with the same UWB anchor placed in a fixed position, whereas a single tag was located at different distances with respect to the anchor: 1) In the first setup we considered the following 4 orientations to analyze the diagram of radiation of the devices with respect to the vertical axis: 0° , 90° , 180° , and 270° ; whereas the anchor was kept fixed at 0° . Ranging measurements were recorded at four spots with a separation between the anchor and the tag of 1, 3, 6, and 12 m. 2) In the second setup both the anchor and the tag were kept fixed at 0° . Ranging measurements were recorded at several spots with a separation between the anchor and the tag varying from 1.5 m to 39 m and a spacing of 1.5 m between consecutive spots. Both measurement setups considered the profile corresponding to a carrier frequency of 3.9993 GHz, a data rate of 110 kbit/s, a preamble length of 1024 symbols, and a pulse repetition frequency (PRF) of 16 MHz. The following magnitudes were recorded for each measurement campaign: the so-called raw ranging values without any post-processing, the estimations for the total receive power and the power of the signals corresponding to the first path, and finally, the ranging values corrected by the DW1000 firmware.

Fig. 1.1 shows the estimates of the raw ranging error, obtained as $\hat{b}_k = r_k - d_k$, with respect to the estimated total receive power and considering the first setup. A correlation between the estimated total power and the error is appreciated. However, the trend is different for high and low total receive power values. For high receive power values, small variations in the receive power lead to large variations in the error. The documentation provided by the manufacturer states that the DW1000 integrated circuit (IC) underestimates the received power in this regime [DEC14], and this could account for the steep variations of the error when compared to the estimated received power.

Fig. 1.2 shows the mean raw ranging error over the estimated total receive power, grouped by the different tag orientations and the measurement points at 1, 3, 6, and 12 m. We can see how a simple change in the tag orientation impacts on the total received power, yielding a variation in the raw ranging bias as well. We can see also how this effect seems to grow according to the distance. These effects related with the distance between devices are also addressed in other works like [RG17].

Fig. 1.3 shows the results obtained for the second setup, where the ranging error is

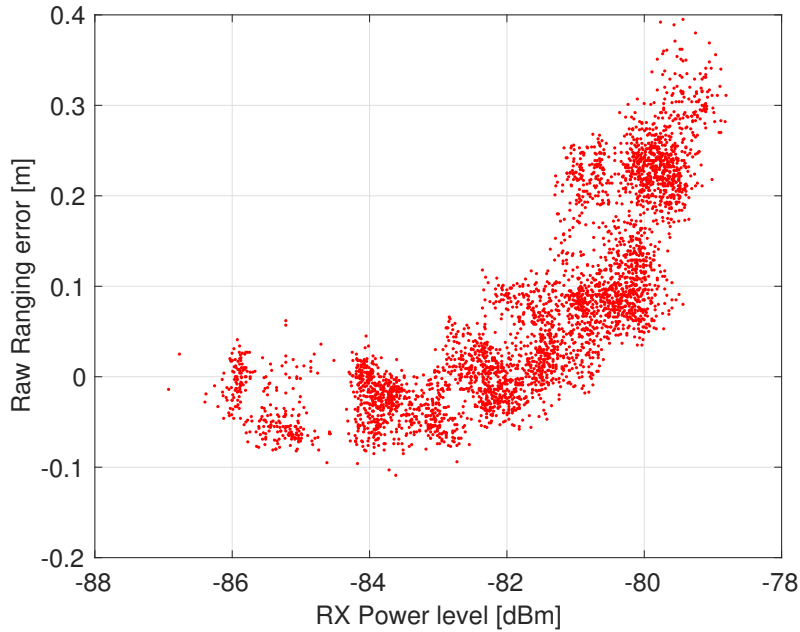


Figure 1.1: First setup: raw ranging error over the total receive power considering multiple orientations for the tag.

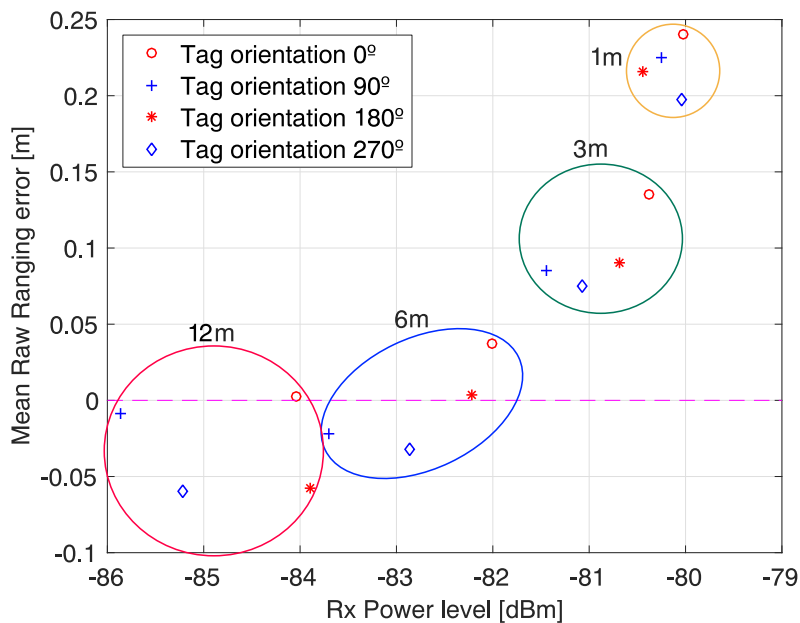


Figure 1.2: First setup: mean raw ranging error versus the total receive power for different tag orientations, grouped by measurement points.

represented against the estimated total receive power. Again, it can be appreciated some correlation between the estimated total receive power and the error. However, in this setup there are some ranging measurements that seem to be out of the trend when compared to Fig. 1.1. This effect can be seen in Fig. 1.4, where the mean raw ranging error is plotted against the actual distance between the anchor and the tag. For some distances abrupt variations in the error that do not follow the bias trend with the distance are appreciated. The mean error corresponding to

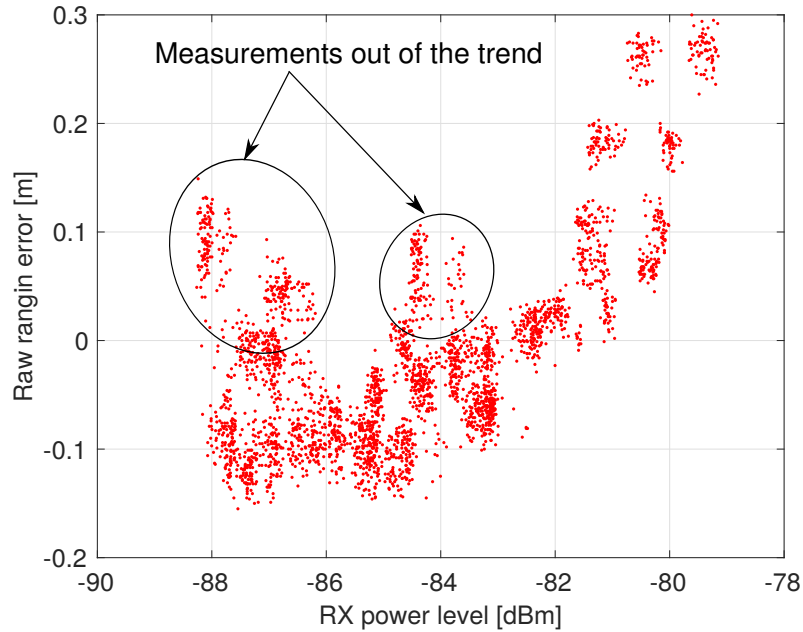


Figure 1.3: Second setup: raw ranging error over the total receive power considering a single tag orientation.

the corrected ranging provided by the DW1000 firmware is also plotted in Fig. 1.4. Although the maximum mean error was around ± 0.05 m for the aforementioned distance values the error raised up to ± 0.2 m because at these points the ranging bias could not be correctly predicted from the raw ranging values.

Although this measurements were carried out in a semi-open environment (a sports hall) without obstacles and the walls were far enough to have a negligible effect, there was a remaining source for multi-path: the floor. In this situation, the two-ray reflection model [GOL05; RAP+96] could be used, where it is considered the direct path and a reflected path coming from the reflection on the floor. In the aforementioned measurements the tag and the anchor were placed at a height of 1.5 m hence, considering the UWB carrier frequency, the two-ray reflection model yields the plot shown in Fig. 1.5. This figure shows the normalized power of the LOS path and the combination of both paths with respect to the distance between the tag and the anchor.

The figure shows deep fades that were produced by the destructive combination of both rays (due to opposite phases). These fades appear around the distances: 12, 15, 20, and 30 m. Therefore, considering the discrete resolution of the measurements (in steps of 1.5 m), it can be seen how this fades matched almost perfectly with the abrupt changes in the raw ranging error shown in Fig. 1.4.

Although the two-ray model could explain the source of distortion in the measurements, it did not explain the variations of the raw ranging bias shown in Figs. 1.3 and 1.4, where it can be seen how the raw ranging error changed not only with the estimated total received power, but also with a another factor.

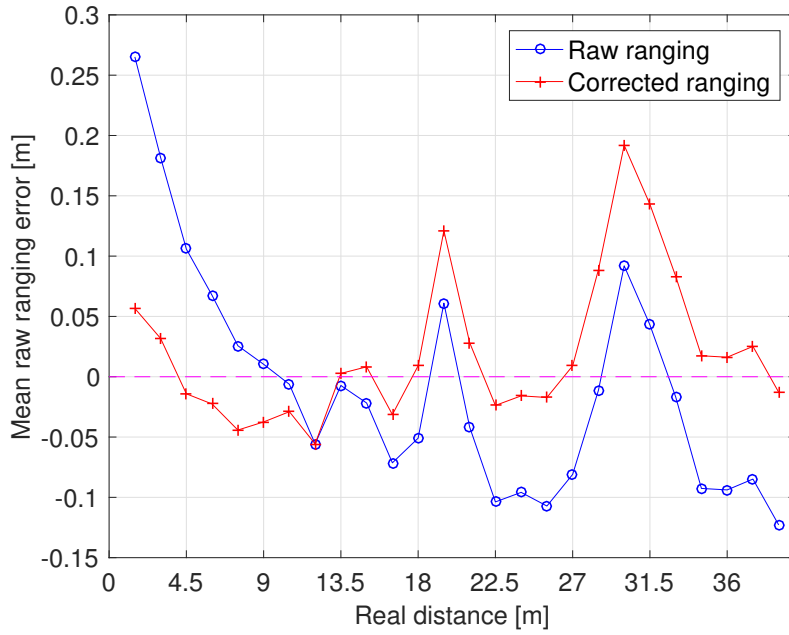


Figure 1.4: Second setup: mean raw and corrected ranging error versus distance between the tag and the anchor.

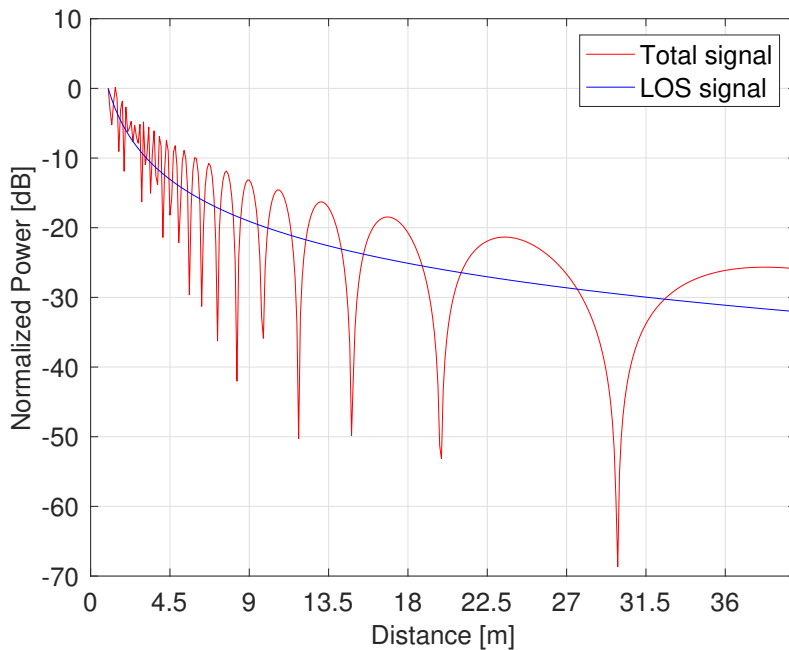


Figure 1.5: Theoretical two-ray propagation model applied to the measurement scenario.

Fig. 1.6 shows the raw ranging error against the difference between the total estimated receive power and the estimated received power of the signals corresponding to the first path (that is, the one which is used to estimate the ranging) in the anchor. When the difference between these two magnitudes was larger than ≈ 6 dB, a correlation with the ranging error appeared. These measurements corresponded with those taken at the points where the bias trend presented an abnormal behavior in Fig. 1.4. There was a relationship between the bias

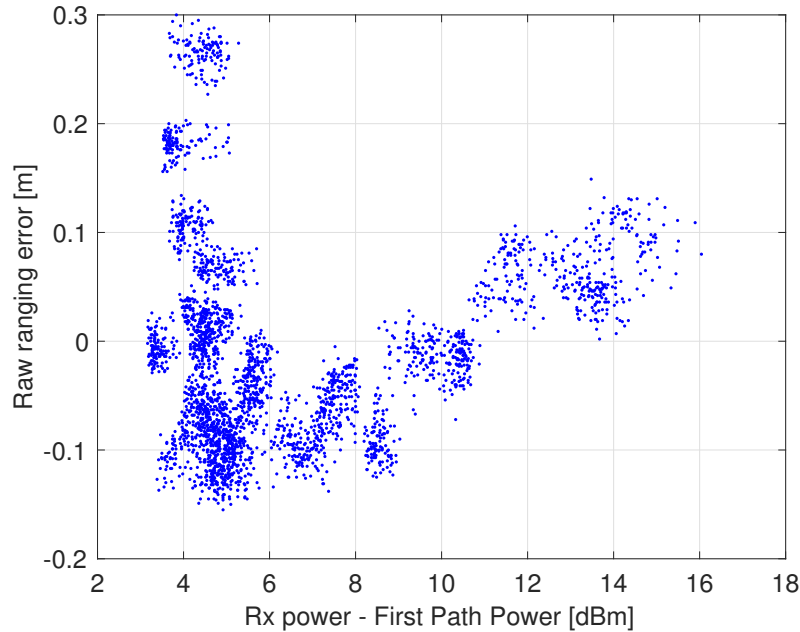


Figure 1.6: Raw ranging error with respect to the difference between the total receive power and that of the signals corresponding to the first-path.

and the difference of the aforementioned power values. Hence suggesting that another bias model should be also considered with this difference.

If these data are already a good example of the problems that this type of devices can have when the relation between received power and ranging does not correspond with the expected one, this behavior is accentuated in situations of non-line-of-sight (NLOS). Works like [RG17; JS16; GUR+17] show the effect of this configuration over the range values and over the location algorithms that use these measurements as source of information to make new position estimations.

To understand how the NLOS conditions affect the performance of the UWB devices, is necessary a brief explanation about how this hardware is able to estimate the specific instant when the signal arrives to its destination. The DW1000 divides the reception phase in two parts. In the first one, the hardware looks for the start of frame delimiter (SFD), a part of the transmitted frame that marks the end of the preamble and the beginning of the data part of the frame. To locate the SFD, the hardware uses the preamble present in each frame. This preamble is a series of symbols that contains more or less elements depending on the configuration. Each of these symbols is divided into approximately 500 *chip* time intervals, and in each of these *chip* intervals a negative or positive pulse is transmitted, or no pulse. This preamble sequence has a property of perfect periodic autocorrelation, so it is possible to find the exact channel impulse response (CIR) of the channel between transmitter and receiver [WOL92]. Once the SFD is found, the hardware performs an additional step to determine the exact TOA of the signal. This second steps uses a the estimation of the CIR and a leading edge detection algorithm based on a threshold to detect the exact instant where the first path of the signal arrived. The first path

corresponds to the real distance, while the replicas of the signal caused by the multi-path effect are always delayed (and consequently they appear later in the CIR set of samples). Once this instant is detected, its value is used to correct the final estimate of distance outputted. More detailed information about the operating mode of the DW1000 can be found in Appendix B, and a detailed description of UWB impulse radio can also be read in Appendix A.

Once the mode of operation of the DW1000 has been explained, how does an NLOS situation affect this module? Fig. 1.7 shows the CIR Power observed after performing a ranging operation between two DW1000 modules in a clear LOS situation. As it can be seen, there are a number of values with much greater energy than the rest. In addition, the first of these clearly highlighted values corresponds to the maximum of the entire set of samples. In a clear LOS scenario the energy of the first path is very large compared to the rest of the replicas, since it is not affected by any important source of attenuation. In this case, threshold detection algorithm is able to locate very precisely the point at which that increase in energy was received, so it can effectively correct the estimated value.

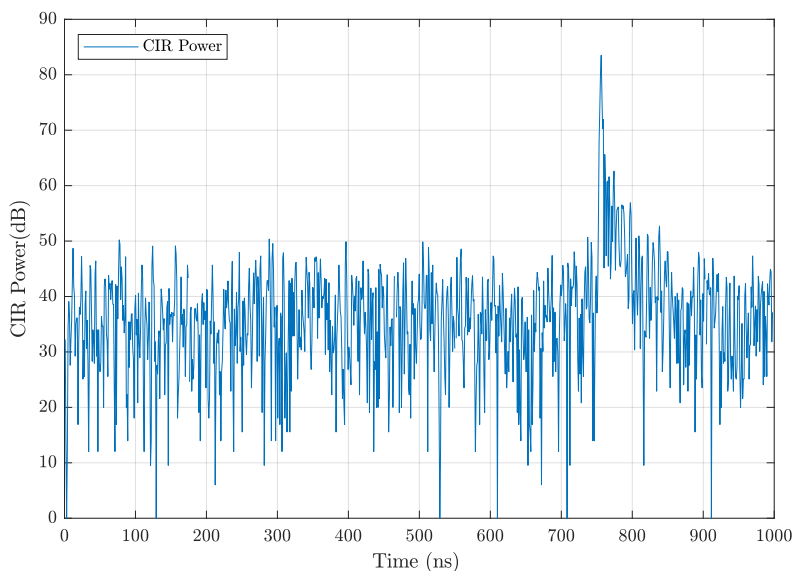


Figure 1.7: CIR Power of a channel in a LOS situation

In Fig. 1.8 it can be seen what happens, however, when there is no clear line of sight between the devices. In this case there is no a clear value of energy above the rest. As a matter of fact, there are a lot of samples with a power value very similar. In this type of scenarios, the leading edge detector algorithm has more problems to work efficiently. Could happen that the first path, too attenuated to surpass the threshold, is ignored by the algorithm and a replica is chosen instead. This would lead to erroneously correcting the estimation of ranging, causing a large increase in the error. In the worst case, the first value to exceed the threshold could be one that is far behind the first path (perhaps after bouncing off several unattenuating surfaces), which could (and does) lead to an error even in the range of meters.

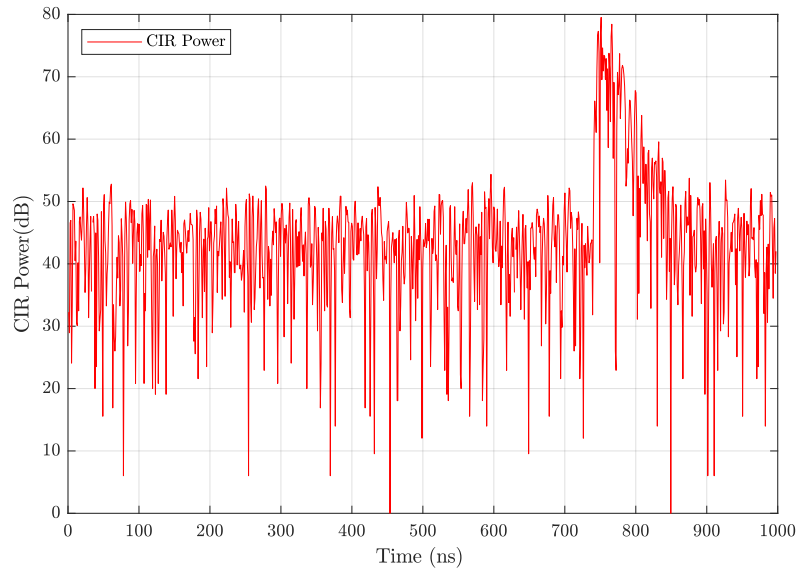


Figure 1.8: CIR Power of a channel in a NLOS situation

1.3 NLOS detection and mitigation: Machine Learning

In view of these results, it is clear that the use of devices UWB, such as those based on the DW1000 (which are the majority in the market), can bring many problems if they lack a clear line of sight between them. Thus, the detection of NLOS conditions and the mitigation of their effects seems to be a promising way to improve the overall performance of UWB-based systems [DAR+09],[DEC+10],[JDW08]. There are several techniques and approaches that follow this idea. Statistical analysis of the resulting range measurements is one of these methods [BHM98; VB07], whereas another different set of algorithms are based on the study of the CIR [MAA+09; AC05; GÜV+07; MAR+10]. The main idea behind these solutions is that the energy of the first path is noticeably greater than the energy of the delayed paths in LOS conditions, whereas this difference tends to shorten in a NLOS scenario. In addition to these methods, NLOS detection can be performed also at a higher logic level, that is, in the location algorithm that uses the range estimation [WH07]. In this case, some form of additional information, such as a map of the scenario, is needed. Although with very good results, the cited works have one thing in common: they make use of a CIR estimation to try to identify if a measure has been generated in a LOS or NLOS scenario. The DW1000 has a method to extract the samples of the CIR used internally to detect the first path. However, there are some practical details about this operation:

- The CIR must be extracted from the chip sequentially in chunks of data through the serial port, which requires to download 4064 bytes per CIR estimate [POZ19a]. That is, the latency introduced to obtain this parameter is no less than 300 ms for each CIR estimate (although in our real tests this values was near 900 ms due the limitations in the buffer size of the *I2C* port). Other works like [GUR+17] claim even a longer time to extract the

complete CIR, about 2 s to 3 s.

- CIR measurements correspond to the wireless channel between the emitter and the receiver just after completing a transmission of data. However, these two devices could be different from the ones involved in the ranging process depending on the mode of operation considered. For example, Pozyx devices (that include a DW1000 IC) allow for a so-called *remote* mode in which a node A (typically connected to a computer to receive the measurements) can command a remote node B to perform a ranging operation between the node B itself and a third node C . In this case, the CIR available at A corresponds to the channel between A and B , instead of the channel between the two nodes involved in the ranging process, i.e., B and C . Thus, NLOS detection would be possible only between nodes A and B , rather than between nodes B and C , which are the ones that really participate in the ranging process.

Another important factor about the CIR analysis approach to detect and mitigate the NLOS effects is that this information must be available after each ranging process. DW1000 IC has a mechanism to do it but other hardware may not have this option, especially when it comes to commercial devices for end-users rather than for researchers.

With all this in mind, and looking for a problem approach that would allow a real-time implementation in which the data used would be available in any UWB transceiver, it was decided to try a solution based on other information other than the CIR. The idea would be to take range and RSS measurements (the basic values that a transceiver of UWB signals for location should provide) in different scenarios with LOS and NLOS and use those measurements to train the classification and mitigation machine learning (ML) based algorithms. Once trained, the execution of these models could be done in real-time and implemented even within the small microcontroller unit (MCU) included on devices like those of Pozyx or the Decawave DWM1001C (a module that includes the DW1000 to handle the UWB part but also a Nordic MCU with bluetooth low energy (BLE) [NOR19]). Fig. 1.9 shows a block diagram with the different steps of this approach. First, some features of interest are extracted from the UWB measurements (*range* and RSS, and later they are used to pass through a classifier that classifies each one as LOS or NLOS. The mitigation phase tries to reduce the error of the estimation, and finally the location algorithm uses resulted values to calculate a new position estimation. The classifier and the mitigator, in this approach, are built using machine learning techniques.

This approach has already been used previously in problems also related to localization and NLOS but using other technologies. For example, the work described in [XIA+14] shows a similar idea but employing the ML techniques to detect the NLOS in WiFi deployment. On this occasion they use different statistics based on the RSS for the training, as this is the only parameter available with this technology.

This thesis dedicates several chapters to describe a solution to try to detect and mitigate the effects that NLOS causes in distance measurements generated by devices that use UWB signals

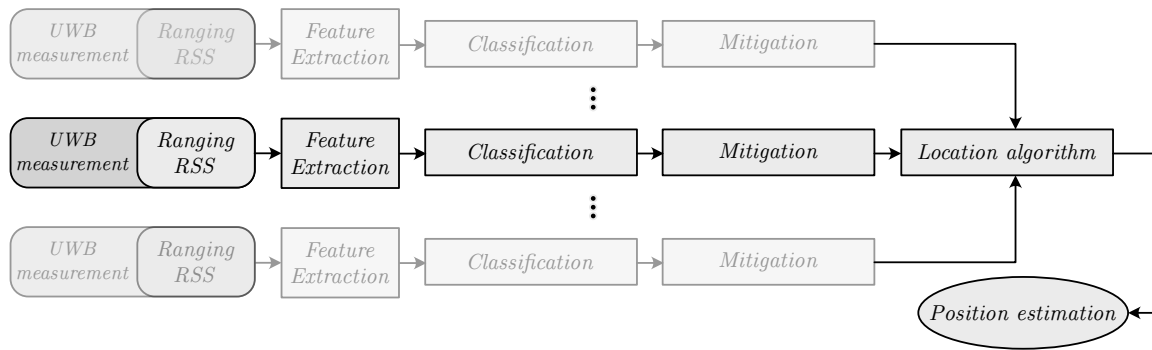


Figure 1.9: Machine learning approach to the NLOS classification-mitigation problem.

for this purpose. The work has been divided into different chapters for better reading:

- Chapter 2 describes the study carried out to compare different first-order statistics on the parameters of range and RSS in order to check which of them behaved better when serving as a training set for a classification algorithm based on ML. For this purpose, different measurement campaigns were carried out in real environments, so that part of the samples obtained were used to train the algorithms and another part to validate the results and check which statistics were the best in terms of success rate. The results obtained after the experiments described in this chapter are then used as the basis for subsequent experiments.

- Chapter 3 details a series of simulations (based on real measures) that served to verify to what extent the elimination and/or mitigation of NLOS measures could affect the final performance of a positioning algorithm based on measures UWB. To do this, a model was created capable of generating LOS or NLOS measurements between a tag and different beacons placed on the scenario, as well as a mechanism to virtually move a tag over a predefined route. All this, together with an implementation of several localization algorithms, served to quantify the weight of the NLOS on the final positioning error.

- Chapter 4 shows the results of a study in which different ML algorithms were compared with real measures both to detect measures coming from a NLOS scenario and to try to mitigate its error. These results allowed to establish the maximum level of improvement that could be obtained, since when training and testing the algorithms with the measures coming from the same scenario the situation was the best possible for the automatic learning algorithms.

- Finally, Chapter 5 describes a complete system in which several ML algorithms are trained with the measurements coming from a measurement campaign in a controlled indoor scenario and later they are used on samples from a totally different one. It is then analyzed the impact it has on the final positioning errors considering different approaches: using all the measurements, using only those classified as LOS, performing or not mitigation, etc. The different combinations were tested to see whether an ML-based detection and mitigation solution is good enough to be applied in different indoor scenarios.

1.4 A practical approach: simulation

Although testing in a real environment is always the best test of any location system, this is not always easy from several points of view: economic cost, difficulty of deployment, availability of a suitable environment, etc. This is why it is often necessary to have a simulation tool that allows predicting to a certain degree the performance of a system before it is finally deployed in production.

There are not many UWB simulators publicly available, and all of them are focused on replicating the radio signal at low level [VWW05]. Although this approach is perfectly valid for other areas of research more focused on signal theory and processing, it is not very useful when it comes to implementing a real location system. A simulator with enough temporal precision in the signals and capable of generating the diverse effects that the multi-path has on them is presented as a certainly complex task and perhaps impossible to implement in complex scenarios with home computers.

Chapter 6 describes the implementation of a location-oriented UWB simulator based on measurements from real devices. The simulator was used to simulate one practical case for a project in indoor location: the location of pallets in an industrial environment. More specifically, this chapter describes an approach to the problem in which the elements to be located were the forklifts. The simulator was implemented on Gazebo [GAZ19b], a physics simulation platform, using Robot Operating System (ROS) [ROS19c] as a means to manage the different simulations. ROS worked as a backend to run the simulations and record the data for the different simulated sensors. At the end of this chapter we detail the results of various location simulations in different scenarios and using different combinations of sensors in addition to those based on UWB.

1.5 A practical approach: Accessing location information

In a real deployment, any real-time location system (RTLS) must be integrated into the organization for which it was created. Although for researchers the main objective is always the improvement of accuracy, this factor is diluted among others when talking about implementing and deploying a RTLS in a real environment, such as a hospital, a sports center or a factory. In this situation it is possible to point to other factors such as the most important ones:

- **Robustness:** a RTLS must be robust in the presence of possible disruptive elements of the environment, whether temporary or permanent. Changes in furniture, movement of people or modifications in work equipment should not critically affect the positioning system.
- **Scalability:** a good RTLS must be prepared to be easily scalable if the scope of operation is to be extended or reduced. The system must be flexible enough so that a possible change in the coverage area does not mean having to start from scratch again.

- **Reliability:** a RTLS is only useful if it is reliable. If there is no absolute certainty that the data provided is correct, then it is unfeasible to implement any kind of action or automation based on it. Areas such as security, logistics, etc. can in no way fall on an unreliable localization system.

Chapter 7 shows the description of a multi-technology localization platform designed to meet these requirements. This platform contemplates two aspects: one oriented to its use as a productive mechanism within the operation of a company, and a second one focused on didactic aspects for its use by the teaching and researching community. The platform has been validated in several real deployments in different projects at national and European level.

1.6 Contributions related with the chapter

Parts of this chapter are based on the following contribution:

- Valentín Barral, Pedro Suárez-Casal, Carlos Escudero, and José A. García-Naya. “**Assessment of UWB ranging bias in multipath environments**”. *Proc. of Proc. of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Alcalá de Henares, Spain, 2016

Chapter II

Machine learning features selection

As part of the strategy of employing automatic learning techniques to detect measurements from a scenario non-line-of-sight (NLOS), the first step was to choose the set of features that would make up the respective training and test sets. This chapter presents in detail the work carried out in order to make this selection of the best features from those available in the low-cost devices analyzed.

One of the first tasks that must be performed when using automatic learning systems is to select the features to be used in the training of the algorithms. These features must contain the greatest amount of available information related to the final objective to be solved. In the case presented in this work, this objective was to try to classify the measurements coming from a device based on ultra-wideband (UWB) as coming from a line-of-sight (LOS) or a NLOS scenario.

The first approach to this problem was to try to replicate an experiment [XIA+14] that approached this task from a practical point of view, using measurements obtained with commercial hardware. Although this work was based on WiFi technology, the idea was easily exportable to UWB. This approach consisted on carrying out a measurement campaign at different points in space and using the data obtained to check the performance of a classic automatic learning classification algorithm (support vector machine (SVM) was the selected one) using different groups of features. These features were essentially different statistics applied to the raw data: the mean (μ), the standard deviation (σ), the Skewness (γ) or the Kurtosis (κ).

So, during this work the following tasks were executed:

- The UWB devices to be used were chosen. In our case, and following Chapter 1, several devices from Pozyx company were used. A description of these devices can be found in Section 2.1.
- A series of scenarios were defined according to the type of spatial relationship between transmitter and receiver. This led to the definition of the cases LOS, NLOS-*Soft* and NLOS-*Hard* that can be read in Section 2.2.
- A indoor scenario to get the measurements was selected. A place inside the building of

the Scientific Area, on the campus of the University of A Coruña, was chosen. Once the measurement area was delimited, several Pozyx devices were deployed according to an experimental planning, so measurements from scenarios LOS and NLOS were recorded and tagged. Details of this process can be seen in Section 2.2.

- Once raw data was obtained, different statistics were generated using different window sizes. These new data were incorporated as training, validation and test sets for an implementation of the algorithm SVM. Information related to this task can be seen in Section 2.3.
- Several executions of the algorithm were generated using different combinations of statistics, in order to find the combination that obtained the highest success rate when classifying the test measurements. The results obtained are listed in Section 2.4.
- Finally, in view of the results the relevant conclusions were drawn. These conclusions can be read in the last section of this chapter, Section 2.5.

2.1 Hardware used

Various Pozyx [POZ19b] devices were used to carry out the measurement campaign. As mentioned before, they are built around the DW1000 chip, a UWB transceiver manufactured by Decawave [DEC19b], and offer the following advantages: 1) reduced cost; 2) flexibility to work with different configurations, acting as anchors and tags, with different radio parameters, and connected either to an Arduino or directly to a computer by serial port; 3) availability of additional sensors, such as accelerometer, gyroscope, compass, and pressure sensor, useful to support localization tasks; and 4) ease of use and deployment. These advantages have converted the Pozyx solution into a great candidate to deploy UWB technology in almost any indoor scenario.

Pozyx devices have different working modes. They can be programmed to initiate a *ranging* process against other device and get the estimation of distance and received signal strength (RSS), but they can also operate in a so called *remote mode*. In this mode a Pozyx module can initiate the ranging process between other two different devices, and get reported the distance between them. This is very useful when a centralized deployment is selected.

A Pozyx module has a size of 6 cm × 5.3 cm, and weights 12 g. Internally, it contains the next components:

- **STM32F4 microcontroller** The core of the system, is in charge of providing an application programming interface (API) to manage the location functionalities. This microcontroller unit (MCU) also hides some of the internal characteristics of the DW1000 module, so some of data present in the UWB chip is not accessible in Pozyx.
- **DW1000** The UWB transceiver manufactured by Decawave. A big description of this module can be found in Appendix B.
- **BNO055** An inertial measurement unit (IMU) manufactured by Bosh [BOS19]. It is a

very sophisticated unit that includes:

- Triaxial 14-bit accelerometer.
 - Triaxial 16-bit gyroscope.
 - Triaxial geomagnetic sensor.
 - 32-bits cortex M0+ MCU. Used to run a fusion software that improves the inertial values.
- **MPL3115A2** A barometric sensor manufactured by NXP [NXP19].
 - **Other components** Pozyx devices include an Arduino Uno R3, Mega and Nano pin header, 4 general purpose LEDs, 4 optional GPIO pins and can be powered from battery, USB or from an attached Arduino.

2.2 Scenario description

In this work, three different classes were defined according to the behavior of the first *path* of the decoded signal in the receiver UWB. Thus, the class defined as LOS modeled the situation where the receiver was able to correctly decode the first *path* of the signal, so that the time of flight (TOF) of the signal was approximately equal to the corresponding one with the shortest distance between both devices. This would be the usual case when there are no obstacles present between transmitter and receiver.

As for the case NLOS, it was decided to divide this situation into two subclasses, as well as other existing proposals in the literature [VB07; YAN+06]. It was considered that a measurement belonged to the class NLOS-*Soft* when it came from a signal where, despite the existence of an obstacle that prevented direct vision between transmitter and receiver, was able to successfully decode the first *path*. Thus, although the received power would be significantly lower due to the attenuation caused by the obstacle, the TOF of the signal would be very similar to the LOS case. So the final estimate of *ranging* would be also similar.

In contrast to this situation, it was defined a second NLOS case named NLOS-*Hard*. Here, the obstacle between transmitter and receiver prevented direct vision between them, but in such a way that the first *path* of the signal could not be detected. However, a rebound of the signal somewhere in the scenario could be decoded. In this case, of course, the TOF of the signal would be greater than the distance between the devices. This would lead to an estimation of *ranging* which could be much larger than the actual distance.

In Fig. 2.1 it can be seen the positioning scheme of the different devices in order to be able to collect representative measurements of the three classes, LOS, NLOS-*Soft* and NLOS-*Hard*. A total of five Pozyx devices were used:

- One called *tag* (marked with the letter *T* in the figure) that was moved through various points of the stage.
- Three devices acting as anchors that were placed in fixed positions (*A*, *B* and *C* in the figure).

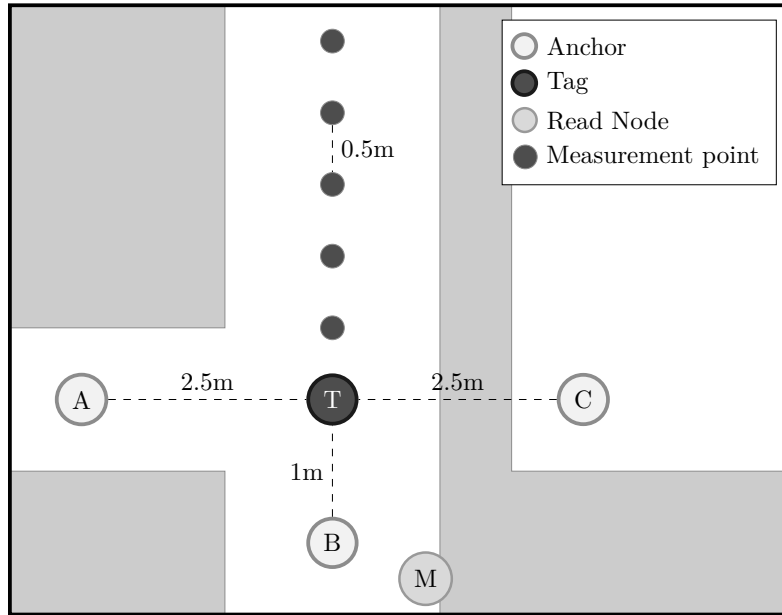


Figure 2.1: Experiment Configuration.

- One for *monitoring* (letter *M* in the diagram) whose mission was to receive the measurements of *ranging* remotely and pass them through a serial port to a computer responsible for storing the data.

As can be seen from Fig. 2.1, the nodes *A*, *B* and *C* were placed with respect to the *tag* so that its line of sight corresponded to the cases previously contemplated. Thus, between the *A* and *T* devices there was a condition of *NLOS-Hard*, between the *B* and *T* a condition of *LOS* and between the *C* (which was placed behind a door) and *T* had a condition *NLOS-Soft*.

The measurement capture process was managed from the device *M*, which was in charge of initiating a remote *ranging* request between the *tag* and each of the other static nodes. The data was then received on a computer with an instance of Robot Operating System (ROS) and a *log* application to store the records. This process was repeated over several measuring points with a separation of 0.5 m between positions. At each position, measurements were made for 90 s at a rate of approximately 20 measurements per second. Once all the measurements were collected, they were processed to make the data available for analysis within Matlab™. The measurements captured in this session are publicly available online at [BAR19e].

In Fig. 2.2 it can be seen the data obtained after the measurement campaign, colored according to the class to which each sample belonged. In this graph it can be seen how the identification between *LOS* and *NLOS* is more feasible than the distinction between the two subtypes of *NLOS*.

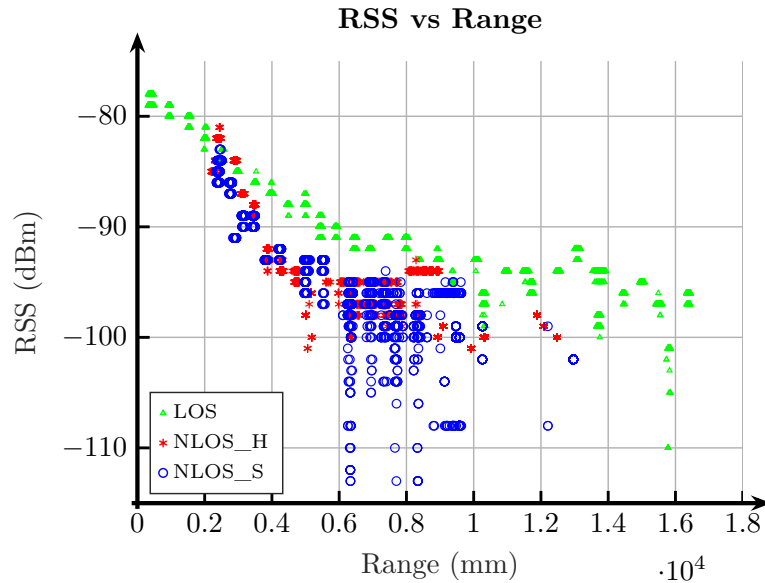


Figure 2.2: Measurements obtained in the three scenarios.

2.3 Machine Learning

Machine learning techniques are widely used for classification tasks [KZP07]. In particular, for this work was selected the classic SVM algorithm. SVM is a classic supervised machine learning (ML) algorithm originally described in [VAP95] and used for both classification and regression problems. The main idea behind this algorithm consists in finding the hyperplane that maximizes the distance between the classes or values of interest. To perform its work, SVM relies on the concept of *kernel* function. A *kernel* is a function that can transform a low-dimensional space into a higher-dimensional one, hence non-separable problems can be converted into separable ones. There are many kernel functions (linear, Gaussian, Polynomial, etc.) and another set of hyper-parameters that must be selected for each instance of an SVM algorithm. As with other ML algorithms, this can be achieved using Bayesian optimization.

SVM can be used for binary classification, even though it can be also considered for multi-class classification problems when using the so-called “one versus all” strategy [PCS00]. In addition, it can be applied to regression problems with slight modifications [DRU+97].

As any machine-learning based algorithm, work with the SVM algorithm typically follows the next steps:

1. A set from the original measurements is selected as the training data, whereas the remaining data is used as the test set. Notice that different measurements at the same position can be included in both sets. Although, as it will be shown in Chapter 4, this condition might be seem unrealistic, it does not impact on the parameters comparative.
2. The training set is used to train the SVM model using a cross-validation schema. This means that, in an iterative process, the training set is split into several sub-sets in order to compare different realizations of the SVM model depending on some configurations of

the algorithm. The most successful configuration is the one selected to be used in the test stage.

3. Once the training stage has finished, the SVM model is applied over the test set to get an estimation of its real performance.

To create the training and test sets, all the data obtained from the measurement campaign was shuffled in order to obtain the sets with a mixture of NLOS-*Hard*, NLOS-*Soft* and LOS conditions.

The characteristics to be compared were chosen based on the data provided by the devices used and how easy was to generate them in real time. In addition, the reference [XIA+14] was followed. Thus, the following characteristics were considered in the comparison:

- μ_{rss}, μ_{ran} : Average of RSS and average of *ranging*, respectively.
- $\sigma_{rss}, \sigma_{ran}$: Standard deviation of RSS and mean of *ranging*, respectively.
- $\gamma_{rss}, \gamma_{ran}$: Skewness of RSS and half of *ranging*, respectively.
- $\kappa_{rss}, \kappa_{ran}$: Kurtosis of RSS and half of *ranging*, respectively.

Note that to calculate the above statistics it is necessary to establish a sample window. In this way, different values of these parameters have also been taken into account in the comparison.

As mentioned before, an important step before starting with the training and test process, was the search for the best hyper-parameters of the SVM algorithm. For this task a technique known as Bayesian optimization was used. Bayesian optimization is an optimization technique considered for the so-called *black-box* functions [BET91]. It is a search strategy that tries to find the optimal values of a function in situations where the evaluation of the function is very expensive, especially in terms of time. Bayesian optimization provides a more efficient search strategy than grid or random search. It is based on the use of a surrogate model of the objective function f . Typical surrogate models for Bayesian optimization are Gaussian processes. The model uses prior knowledge and previous observations of f to generate a posterior estimation of the objective function. Finally, an acquisition function is used on this estimation to propose a new sampling point. This is an iterative method. In this work, the Bayesian optimization approach was employed to find the best suitable hyper-parameters of the considered SVM algorithm.

Once the best values for the hyper-parameters were obtained, the final model was constructed by training the algorithm with these parameters and the training set.

2.4 Results

The first of the performed experiments used each of the features individually as material to train and test the classification algorithm. In Fig. 2.3 it can be seen the success rate when classifying samples as either LOS or NLOS using each of these features. In this case, the measurements of type NLOS included all those of the subclasses NLOS-*Hard* and NLOS-*Soft*. In the graph can also be seen the differences in the success rate when the size of the window used to calculate the

different statistics was enlarged. It can be seen how for this classification task, the best results were obtained for the features that used the mean as statistic: μ_{ran} and μ_{rss} got the best and second best results respectively. With much more error were the features that use the standard deviation, σ_{rss} and σ_{ran} .

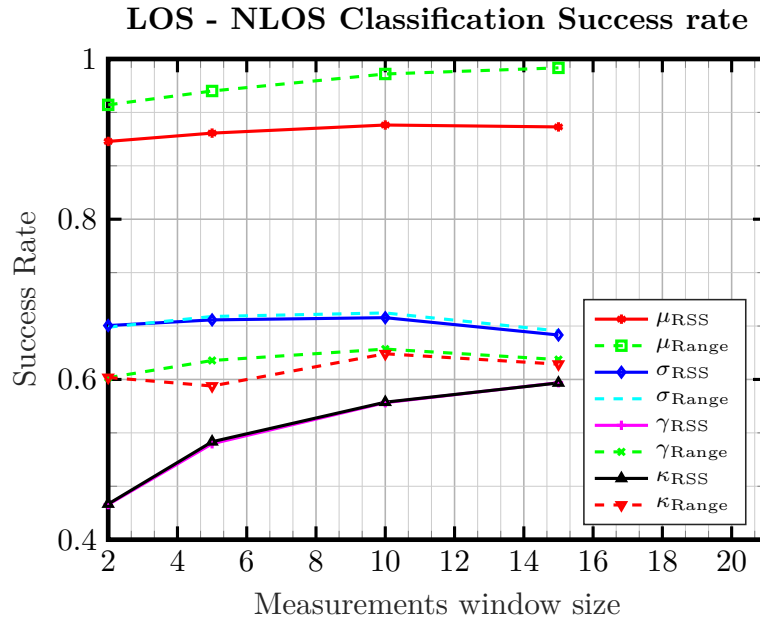


Figure 2.3: Success rate classification LOS vs NLOS.

In Fig. 2.4 it can be seen a similar comparison, in this case trying to classify samples as either NLOS-*Hard* or NLOS-*Soft*. In this case, samples of the class LOS were not used for the training or test sets. As could be guessed in Fig. 2.2, in this case the classification became more complicated due to the smaller differences between the two classes. In spite of this, the features the best results were again those that use the average as a statistic: μ_{ran} and μ_{rss} .

After analyzing the behavior using the features individually, a new experiment was designed in which the training and test sets were formed not by individual statistics, but by sets of them. For this task, the most successful features of the previous tests (μ_{ran} , μ_{rss} , σ_{rss} and σ_{ran}) were chosen and a series of sets with different combinations of them was defined. These sets can be seen in Table 2.1.

Once these test sets were defined, the corresponding trainings/tests were performed and the obtained results can be seen in Fig. 2.5. In this case, the classification results corresponded to a more complex situation than in the previous experiments, since on this occasion the classifiers had to discriminate the samples between the three classes at the same time, LOS, NLOS-*Hard* and NLOS-*Soft*. It can be seen in the graph how, in spite on this, the sets $S1$ and $S5$ offered much higher success rates than the cases where the characteristics used for the training were the individual statistics. These two sets were those that included μ_{rss} and μ_{ran} on one side and $\mu_{rss}, \mu_{ran}, \sigma_{rss}$ and σ_{ran} on the other. Based on these results, it seems clear that the best classifier was obtained by using as features of the training set the statistics μ_{rss} and μ_{ran} .

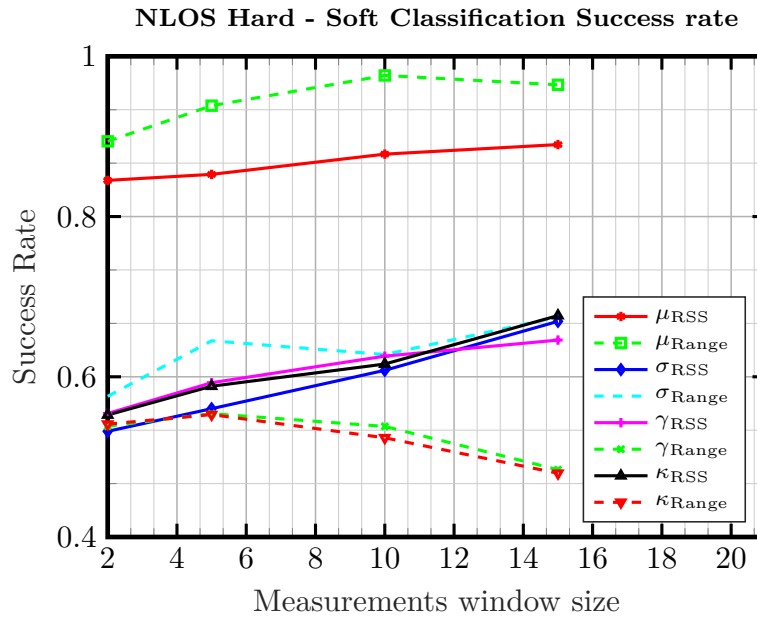


Figure 2.4: Success rate classification NLOS-S vs NLOS-H.

Table 2.1: Features Sets

Characteristic	Set 1	Set 2	Set 3	Set 4	Set 5
μ_{rss}	✓		✓		✓
μ_{ran}	✓			✓	✓
σ_{rss}		✓	✓		✓
σ_{ran}		✓		✓	✓

Finally, regarding the size of the window used to calculate the statistics, a value of 5 samples seems to be sufficient to obtain a yield close to the maximum success rate.

2.5 Conclusions

This chapter has detailed the process of searching for the best features available in our UWB devices in order to identify the measurements in three classes: LOS, NLOS-*Hard* and NLOS-*Soft*. For this purpose, a classifier was built following the algorithm SVM and the actual measurements obtained in a measurement campaign carried out in a building on the campus of the University of A Coruña were used. The features were chosen from a series of basic statistics applied to the *ranging* and RSS data captured during the measurement campaign. Several experiments were carried out, first with each statistic individually and then grouping several of them into different configurations. The performance difference was also analyzed when

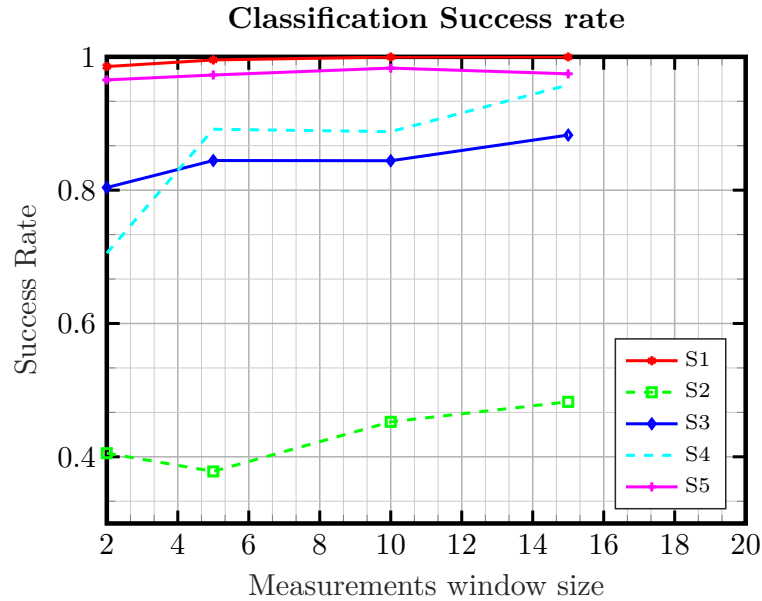


Figure 2.5: Success index with sets of features.

classifying between LOS and NLOS, between NLOS-*Hard* and NLOS-*Soft* and finally with respect to classifying between the three classes LOS, NLOS-*Hard* and NLOS-*Soft*. The number of samples used to calculate each statistic was also taken into account in the experiments.

As a result, a series of features and a window size were obtained that would be used later to continue constructing the classification and mitigation model proposed in this work. In addition, the measurement campaign generated a database that was published online for the research community with access completely free.

In Chapter 3 will be analyzed the impact that a classifier like the presented in this chapter can have in the final estimates of position of a location algorithm. This will define the base of the performance improvement that our solution can reach.

2.6 Contributions related with the chapter

The content of this chapter is based mainly on the following contribution:

- Valentín Barral, Carlos J. Escudero, and José A. García-Naya. “**Nlos classification based on rss and ranging statistics obtained from low-cost uwb devices**”. *Proc. of 27th European Signal Processing Conference (EUSIPCO)*. A Coruña, Spain, 2019

Chapter III

Impact of Non-Line of Sight in localization

This chapter details the results obtained after studying how the identification of ultra-wideband (UWB) *ranging* measurements from a non-line-of-sight (NLOS) scenario improves the performance of the location algorithms that use them.

Continuing with the original idea of implementing a system capable of detecting and mitigating errors arising from the absence of direct line of sight in low-cost location devices using UWB technology, this chapter analyzes the impact in the estimates of position in case of carrying out such implementation.

For this purpose, samples obtained during a new indoor measurement campaign were used to build a simulator of virtual location scenarios (details of this campaign can be seen in Section 3.1). The idea of this simulator was to be able to generate real *ranging* estimates for any point within a virtual scenario, so that these measurements could be passed to a positioning algorithm that would perform the final position estimation. These *ranging* measurements would be of either class line-of-sight (LOS) or NLOS depending on a probability parameter set for each simulation execution. Thus, this simulator could be configured with the following parameters:

- Number of UWB anchors (reference devices with known positions) present in the scenario.
- Probability of each device to generate a sample corresponding to the scenario NLOS.
- Points of a trajectory or *waypoints* that the simulated mobile device would follow during the test.

All details of this simulator can be seen in Section 3.2. Once configured, the simulator provided a list of estimates of *ranging* and received signal strength (RSS) for each point in the trajectory for each reference beacon, as well as the corresponding position of the mobile device at each instant. With this data, and using a series of location algorithms of a different nature that were implemented for this work (details of which can be seen in Section 3.3), a series of position estimates were obtained after applying some actions on the measurements NLOS. The results of comparing these positions with the original trajectories can be seen in Section 3.4. Finally, the conclusions drawn after the experiment can be read in Section 3.5.

3.1 Measurement campaign

In order to be able to feed the simulator with real measurements, a measurement campaign was carried out in the building of the Scientific Area of the Elviña campus, at the University of A Coruña. As can be seen in Fig. 3.1, two Pozyx anchors based on the DW1000 microcontroller unit (MCU) were placed in two fixed positions while an array was built with five other devices on a tripod that was moved at different distances. The special features of the hardware used can be seen in Section 2.1 and Appendix B.



Figure 3.1: Measurement scenario



Figure 3.2: Tripod with measurement tags.

Due to the position of the two static anchors, the measurements corresponding to one of them had a clear line of sight between transmitter and receiver, so the samples were labeled as LOS. The other beacon, concealed behind two walls, provided the measurements considered NLOS.

On the mobile tripod, 5 Pozyx tags were placed, separated from each other 0.2 m. A picture of this setup can be seen in Fig. 3.2. This configuration was made with the sole purpose of accelerating the measurement process, so that for each position of the tripod samples could be obtained at five different distances. In this way it was possible to collect a bank of measurements UWB with measurements from 3 m to 16 m, spaced 0.2 m before commented. All these measurements were made available to the research community in [BAR19a].

3.2 Simulator

In order to measure the impact that the NLOS measurements had on the final positioning results, one of the tasks of this work was to implement a mechanism for generating virtual scenarios in which to obtain measures of *ranging* between a point of the scenario and a set of UWB anchors placed around the scenario. However, in order to make the simulation more realistic, it was decided to extract the estimates from the previously captured measurement repository. These measurements could be some of those captured in the scenario LOS or in the scenario NLOS, according to a probability value designated at the start of each execution of the simulation.

Thus, the first task of the simulator was to generate the scenario itself with a given number of UWB anchors placed inside it. For this purpose, defined dimensions were used (in the case of this experiment, a cube of $9\text{ m} \times 9\text{ m} \times 9\text{ m}$ was chosen) and the beacons were placed around the limits so that each one of them was placed at a different height and the coordinates x, y were uniformly distributed in those axes.

Once the scenario and the reference beacons were generated, the next task implemented in the simulator was to generate a mechanism to define the routes that the mobile should follow when moving. To do this we used the toolbox of Matlab™ *Sensor Fusion and Tracking*, which could generate routes based on a series of *waypoints* temporarily marked. In the experiment carried out in this work, two routes were created using this formula: a rectangular route and a totally random one.

Thirdly, the simulator needed to have a mechanism by which, for a given mobile position and a beacon, an estimate of *ranging* was obtained from those available in the measurement bank. However, this posed a problem due to the discrete nature of the measuring points used in the campaign. That is, within the measurement bank there were records only for a certain number of specific distances, and not for any arbitrary value.

The solution was to choose the closest distance to the needed, and to provide a random measurement from all those available for that distance in the measurement bank. However, this approach introduced a problem, as it created an incoherence between the position of the

3. Impact of Non-Line of Sight in localization

tag, of the anchor and the distance between both. For example, suppose the *tag* was placed in the position (P_x, P_y, P_z) and its distance from a $A1$ beacon placed in $(A1_x, A1_y, A1_z)$ was 3.16 m. Since in the measurement bank there are only values for jumping distances of 0.2 m (that is, for cases 3 m, 3.2 m, 3.4 m, \dots), the original distance between the *tag* and $A1$ should be approximated by the nearest available, which in this case would be 3.2 m. But in doing this, we would find a discrepancy of 0.04 m between the actual distance between *tag* and $A1$ and the value provided by the simulator. To eliminate this error factor, we chose to move the anchor position on the same line that connects the *tag* with the original position of the anchor. Therefore the distance between the *tag* and this new position after the displacement corresponded exactly with the value returned by the simulator (in the previous example, 3.2 m). In Fig. 3.3 you can see graphically how this setting works. Obviously, this adjustment had to be made for each beacon in each position of the trajectory, since when the *tag* was moving its distance with the anchors changed and consequently also the approach chosen in the measurement bank.

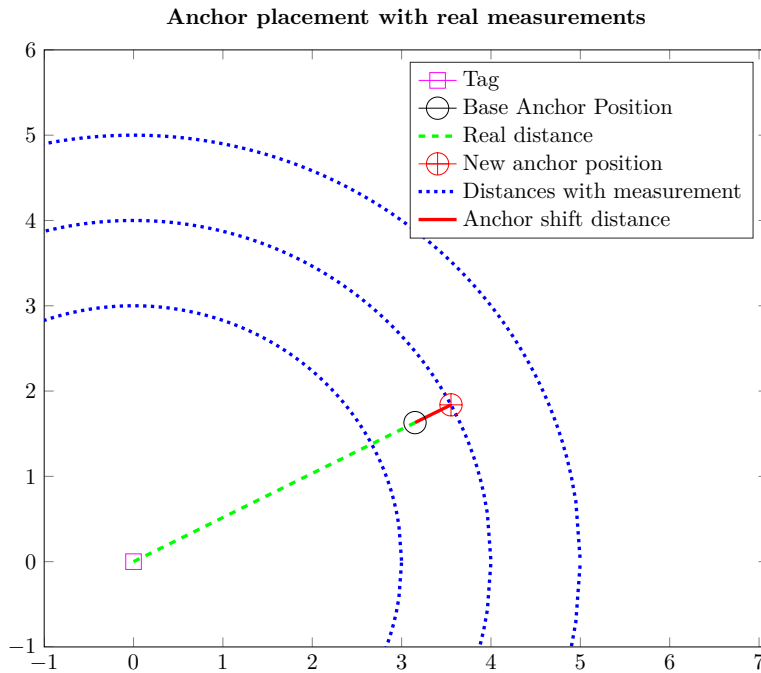


Figure 3.3: Anchor shifting to fit distances with a measurement value associated to a distance available in the repository.

Another important part of the simulator was to select whether for a given position and a particular beacon, the value of returned *ranging* corresponded to a scenario LOS or NLOS. For this task an algorithm was implemented that selected the type of measurement depending on a given probability of outputting a NLOS value. This was implemented using a random process modeled with a given probability distribution.

3.3 Location Algorithms

Pozyx devices are able to obtain an estimation of the distance between a tag and an anchor based on the round-trip time of arrival (TOA) of the signal traveling from the tag to the anchor. When there are multiple anchors in fixed positions and one tag in an unknown location, the ranging estimations can be used to estimate the coordinates of this tag. We have the expression

$$r_{\text{TOA},l} = d_l + n_{\text{TOA},l} \quad l = 1, 2, \dots, L, \quad (3.1)$$

where $r_{\text{TOA},l}$ are the ranging measurements between the tag and the l -th anchor, d_l is the actual distance between them, $n_{\text{TOA},l}$ is an error component associated to this measurement and modeled as AWGN, and L corresponds to the number of anchors. Using the euclidean distance we have

$$r_{\text{TOA},l} = \sqrt{(x - x_l)^2 + (y - y_l)^2 + (z - z_l)^2} + n_{\text{TOA},l}, \quad (3.2)$$

$$l = 1, 2, \dots, L$$

where (x, y, z) are the actual coordinates of the *tag* and (x_l, y_l, z_l) are the coordinates of l -th anchor. If several ranging measurements are available, the previous equation can be used to estimate the location of the tag.

Different location algorithms were chosen to test the effects of considering or not the prior knowledge of the LOS / NLOS condition between a tag and several anchors. The algorithms were selected among the many of them available in the literature taking into account their nature. All of them used the ranging estimations between the tag and the anchors as an information source for their operations.

Thus, the first considered approach is the linear least squares (LLS), by using a gradient descent technique to perform the error minimization. The second approach is the nonlinear least squares (NLS), which considers a Gauss-Newton iterative procedure to minimize the objective function. Finally, the iterative extended Kalman filter (IEKF) was also used in the comparison. Sections 3.3.1 to 3.3.3 respectively describe the aforementioned algorithms.

3.3.1 Linear Least Squares

The LLS algorithm performs the location task in two steps: first, Eq. (3.2) is approximated by means of a linearization and, second, a least squares method is used to find the position that provides the minimum error. There are several methods to approximate the nonlinear equation in Eq. (3.2) such as those described in [NOR12; WAN15; MUR07]. Considering [WAN15], we have the following final approximation

$$\mathbf{A}\boldsymbol{\theta} = \mathbf{b}, \quad (3.3)$$

where

$$\mathbf{A} = \begin{bmatrix} -2x_1 & -2y_1 & -2z_1 & 1 \\ -2x_2 & -2y_2 & -2z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -2x_L & -2y_L & -2z_L & 1 \end{bmatrix}, \quad (3.4)$$

$$\boldsymbol{\theta} = \begin{bmatrix} x \\ y \\ z \\ (x^2 + y^2 + z^2) \end{bmatrix}, \quad (3.5)$$

and

$$\mathbf{b} = \begin{bmatrix} r_{\text{TOA},1}^2 - x_1^2 - y_1^2 - z_1^2 \\ r_{\text{TOA},2}^2 - x_2^2 - y_2^2 - z_2^2 \\ \vdots \\ r_{\text{TOA},L}^2 - x_L^2 - y_L^2 - z_L^2 \end{bmatrix}. \quad (3.6)$$

Thus, following the least squares method, the estimation of the position $\hat{\boldsymbol{\theta}}$ is the result of

$$\hat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta}} (\mathbf{A}\boldsymbol{\theta} - \mathbf{b})^T (\mathbf{A}\boldsymbol{\theta} - \mathbf{b}). \quad (3.7)$$

We can use an iterative gradient descent technique to approximate the solution. Therefore, we define the cost function as

$$J(\boldsymbol{\theta}) = (\mathbf{A}\boldsymbol{\theta} - \mathbf{b})^T (\mathbf{A}\boldsymbol{\theta} - \mathbf{b}) \quad (3.8)$$

and, consequently, the gradient step is

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) \quad (3.9)$$

for each j component of $\boldsymbol{\theta}$, with α being the learning rate. Finally, using Eq. (3.8) and Eq. (3.9), we have the formulations to calculate the coordinates of the unknown position:

$$x := x - \alpha \sum_{i=1}^L \left(1 - \frac{r_{\text{TOA},i}}{d_i} \right) (x - x_i), \quad (3.10)$$

$$y := y - \alpha \sum_{i=1}^L \left(1 - \frac{r_{\text{TOA},i}}{d_i} \right) (y - y_i), \text{ and} \quad (3.11)$$

$$z := z - \alpha \sum_{i=1}^L \left(1 - \frac{r_{\text{TOA},i}}{d_i} \right) (z - z_i). \quad (3.12)$$

3.3.2 Nonlinear Least Squares

The NLS is an approach to solve the problem starting from Eq. (3.2) without performing a first linear approximation [MAR63]. In this case, we can rewrite Eq. (3.2) as:

$$\mathbf{r}_{\text{TOA}} = \mathbf{f}_{\text{TOA}}(\mathbf{x}) + \mathbf{n}_{\text{TOA}}, \quad (3.13)$$

where $\mathbf{f}_{\text{TOA}}(\mathbf{x})$ is a nonlinear function and \mathbf{x} is the position to be estimated. Therefore, we can define our cost function as

$$\begin{aligned} J_{\text{NLS,TOA}}(\tilde{\mathbf{x}}) &= \sum_{l=1}^L \left(r_{\text{TOA},l} - \sqrt{(\tilde{x} - x_l)^2 + (\tilde{y} - y_l)^2 + (\tilde{z} - z_l)^2} \right)^2 \\ &= (\mathbf{r}_{\text{TOA}} - \mathbf{f}_{\text{TOA}}(\tilde{\mathbf{x}}))^T (\mathbf{r}_{\text{TOA}} - \mathbf{f}_{\text{TOA}}(\tilde{\mathbf{x}})). \end{aligned} \quad (3.14)$$

where $\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} & \tilde{y} & \tilde{z} \end{bmatrix}$ is the optimization variable for \mathbf{x} . Using a least squares method, we have that our best estimation is

$$\hat{\mathbf{x}} = \arg \min_{\tilde{\mathbf{x}}} J_{\text{NLS,TOA}}(\tilde{\mathbf{x}}). \quad (3.15)$$

Finding this minimum point is not a trivial task, and there are many different techniques to achieve it [RJ15]. In this work, we chose to use the Gauss-Newton method [HAR61]. This is an iterative method that, starting from some given initial point, approximates the solution in each iteration. The equation that defines the evolution of the estimation is

$$\begin{aligned} \hat{\mathbf{x}}^{m+1} &= \hat{\mathbf{x}}^m \\ &+ (\mathbf{G}^T (\mathbf{f}_{\text{TOA}}(\hat{\mathbf{x}}^m)))^{-1} \mathbf{G}^T (\mathbf{f}_{\text{TOA}}(\hat{\mathbf{x}}^m)) \\ &(\mathbf{r}_{\text{TOA}} - \mathbf{f}_{\text{TOA}}(\hat{\mathbf{x}}^m)), \end{aligned} \quad (3.16)$$

where \mathbf{G} is the Jacobian matrix of $\mathbf{f}_{\text{TOA}}(\hat{\mathbf{x}}^m)$ calculated at $\hat{\mathbf{x}}^m$.

3.3.3 Iterative Extended Kalman Filter

The Kalman filter is a well-known algorithm to estimate the hidden state of a system given some observation variables and is widely applied to positioning problems. The original Kalman algorithm provides an exact solution for this estimation problem in systems where the observations are linear on the state together with Gaussian-distributed noise sources. However, when some of these assumptions do not hold, numerous variations were proposed to overcome these limitations, such as the Extended Kalman filter [GEL74], the Unscented Kalman filter [JU97], and particle filters [TBF05].

In this work, the ranging observations are nonlinear with respect to the tag and anchor positions, hence we have implemented an IEKF, which linearly approximates the actual observation functions and solves the associated maximum a posteriori probability problem using as prior information a prediction on the state when a new observation arrives. In particular, the state $\mathbf{x} \in \mathbb{R}^6$ contains the position and velocity on the three axes, with a prediction model

$$\mathbf{x}_{t|t-1} = \mathbf{G}\mathbf{x}_{t-1} + \mathbf{m}_t, \quad (3.17)$$

$$\mathbf{P}_{t|t-1} = \mathbf{G}\mathbf{P}_t\mathbf{G}^H + \mathbf{C}_m, \quad (3.18)$$

where $\mathbf{G} = [\mathbf{I}, \Delta t \mathbf{I}; \mathbf{0}, \mathbf{I}]$, $\mathbf{m}_t \sim \mathcal{N}(0, \mathbf{C}_m)$ is a noise component caused by the unpredicted acceleration, with $\mathbf{C}_m = [\Delta t^4/4\mathbf{I}, \Delta t^3/2\mathbf{I}; \Delta t^3/2\mathbf{I}, \Delta t^2\mathbf{I}]$ and Δt is the time delay since the last received observation. The observation model is

$$\mathbf{y}_t = r(\mathbf{x}_t) + \mathbf{n}_t, \quad (3.19)$$

where $r(\cdot)$ is the ranging function for all the anchors and $\mathbf{n}_t \sim \mathcal{N}(0, \mathbf{C}_n)$ is the observation error component, with \mathbf{C}_n being a diagonal matrix with the estimated error variance of each anchor on its main diagonal. The IEKF iteratively searches for the solution on the state for this observation model with prior information Eq. (3.17) and Eq. (3.18). The function $r(\cdot)$ is linearized using vectors

$$\mathbf{r}_l = [(x - x_l)/d_l, (y - y_l)/d_l, (z - z_l)/d_l, \mathbf{0}]^T, \quad (3.20)$$

obtaining the Jacobian matrix $\mathbf{R}(\mathbf{x}) = \frac{\partial r(\mathbf{x})}{\partial \mathbf{x}} = [\mathbf{r}_1, \dots, \mathbf{r}_L]^T$, where x, y, z are the state variables corresponding to the position of the tag. Hence, state estimations are iteratively updated for each group of received rangings at the time instant t as

$$\begin{aligned} \mathbf{x}_i &= \mathbf{x}_{t|t-1} + \mathbf{P}_{t|t-1} \mathbf{R}_i^T (\mathbf{R}_i \mathbf{P}_{t|t-1} \mathbf{R}_i^T + \mathbf{C}_n)^{-1} \\ &\times (\mathbf{y}_t - r(\mathbf{x}_{i-1}) - \mathbf{R}_i(\mathbf{x}_{t|t-1} - \mathbf{x}_{i-1})), i = 1, \dots, I, \end{aligned} \quad (3.21)$$

where $\mathbf{x}_0 = \mathbf{x}_{t|t-1}$ and matrix $\mathbf{R}_i = \mathbf{R}(\mathbf{x}_{i-1})$ is updated after each iteration. More details can be found in [GEL74].

3.4 Results

Once the positioning algorithms and the virtual scenario simulator with real measurements were implemented, the following experiments were carried out:

1. The different positioning algorithms were executed to estimate the positions of a mobile *tag* that followed the two proposed trajectories (rectangular and random) within a virtual scenario with real measurements. The experiment was repeated, varying the probability of obtaining a NLOS measurement of an anchor from 0 to 1.0 (that is, from a 100% LOS to a 100% NLOS scenario). In this first part of the experiment, the positioning algorithms made their estimates with all available measurements, regardless they were LOS or NLOS.
2. This second experiment included the same steps and configurations as the previous one, with the only exception that on this occasion all NLOS measurements were discarded before being entered into the positioning algorithms.
3. The third experiment was similar to the previous ones, but in this case only the positioning algorithm IEKF was used. The idea of this experiment was to use all the measurements in the algorithm but considering the estimation of the error of each measurement according to whether it was LOS or NLOS.

In Fig. 3.4 you can see the results obtained by the three positioning algorithms after following the *tag* a rectangular path in $2D$ at constant speed. In the virtual scenario a total of 8 beacons were simulated, each one capable of generating a measurement of the type NLOS with a specific probability according to the values of the abscissa axis. In this figure you can see the two types of configuration explained above: on the one hand the values after using all the measures in the positioning algorithms and on the other using only the measures classified as LOS. We can see how in the first case (solid lines) the three algorithms generate position estimates whose error with respect to the real position grows rapidly as the probability that a measure NLOS is generated by the beacons increases. It must be said that in this configuration the algorithm IEKF does not know which measures are LOS or NLOS, so the covariance error that is passed to it is always the one corresponding to the case LOS. This makes the performance of this algorithm severely penalized. On the other hand, we can also see in the figure how the behavior of the three algorithms is similar, even though there are slight differences between the error values obtained at each point. In any case, the algorithm IEKF was the one that obtained the best result.

The other set of curves that we can see in Fig. 3.4 (dashed lines) corresponds to the results obtained by the different positioning algorithms by using only the measurements LOS and ignoring the NLOS. In this case we see how the results in terms of average error are clearly better for this configuration. Between the three algorithms, we see a clear difference between IEKF and the others once the probability of getting measurements NLOS is higher. This is because, as the algorithms LLS and NLS need at least 4 values of *ranging* in order to estimate a new position. When the probability of getting a measure is high and ignoring these, there comes a time when that 4 minimum of measurements is not reached and the algorithms can't generate a new estimate (it returns the value of the previous position). In the graph it can be seen, on the right ordinate axis, a curve that marks precisely the probability of obtaining at least 4 values of type LOS in each point. It can be seen how when the value falls approximately below 0.1 (which corresponds to a probability around 0.7 that an anchor generates a value NLOS) the algorithms LLS and NLS begin to be unable to generate new position estimations and consequently the average error increases exponentially.

In Fig. 3.5 it can be seen the results for the random path. In this case it can be seen a behavior very similar to that of the rectangular path, with the only difference of the algorithm IEKF. It can be seen how its performance decreases with this configuration. This is because in this implementation of the algorithm, the only data used are those from the UWB measurements, without any other sensor. This causes the IEKF to be forced to estimate the speed of the *tag* only on the basis of the positions generated by the values of *range*. In the rectangular route the only direction changes occur in the corners of the rectangle, so most of the way the algorithm is able to make a fairly accurate tracking of the position and direction of movement of the *tag*. However, in the random path the direction changes are almost constant, so the algorithm is not able to adjust quickly enough to the new value and ends up making a bigger mistake.

3. Impact of Non-Line of Sight in localization

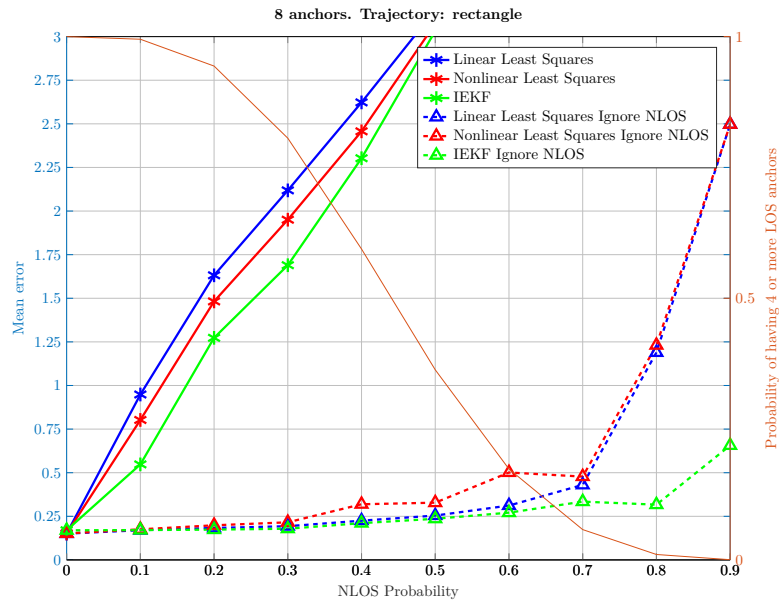


Figure 3.4: Mean error in rectangular path according to NLOS probability. Right ordinate axis: probability of obtaining at least 4 LOS measurements at a point.

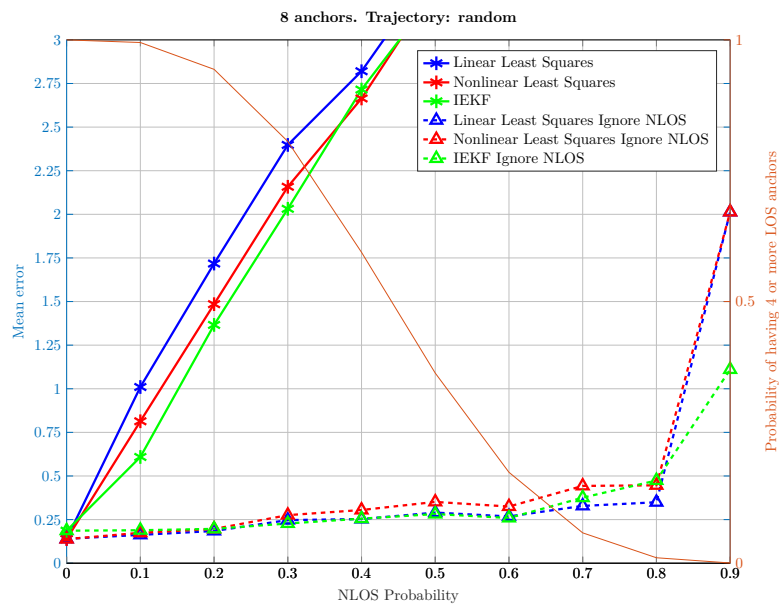


Figure 3.5: Mean error in random trajectory according to NLOS probability. Right ordinate axis: probability of obtaining at least 4 LOS measurements in a point.

Finally, in Fig. 3.6 we can see a comparison between the performance of the IEKF in two different configurations. On the one hand when the NLOS measurements are ignored (green line, triangles) and on the other hand when instead of ignoring the measurements are incorporated into the algorithm but adjusting the covariance error associated with them (black line, squares). We can see how a slight improvement was obtained in the second case, that is, when all measurements were used by the algorithm. This is because, by its nature, the algorithm

IEKF is able to extract information from the NLOS measurements even if they are affected by some kind of noise as long as the error is well characterized.

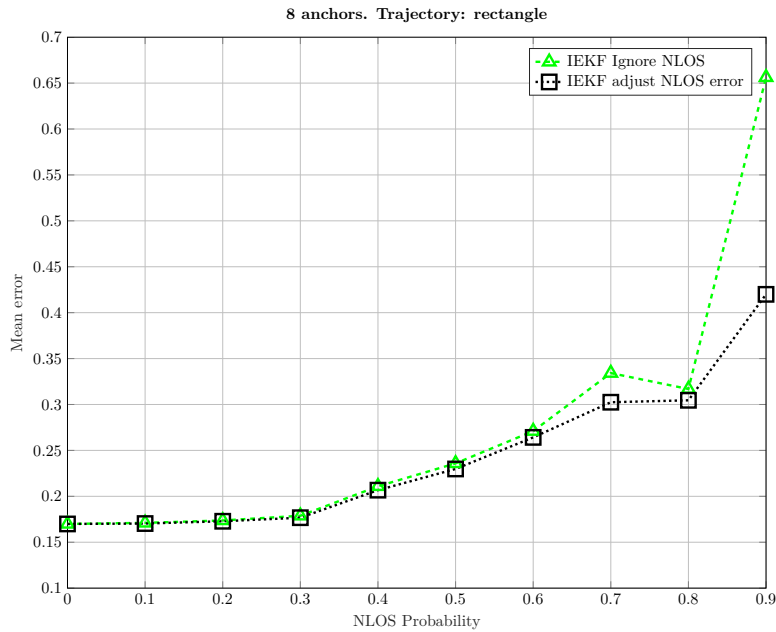


Figure 3.6: Mean error in rectangular path using IEKF according to NLOS probability.

3.5 Conclusions

Throughout this chapter we have compiled the work done in order to confirm the benefits that a UWB measurement classifier in the NLOS situation could have within the final idea of improving positioning results. To this end, a measurement campaign was carried out to feed a virtual scenario simulator capable of generating, for any point of the scenario and a position of a beacon, a ranging value appropriate to the distance between the two. This value could belong to the class LOS or NLOS depending on a probability chosen beforehand.

Using this simulator it was possible to generate range estimations between a series of several anchors and a *tag* that moved along two different trajectories: one rectangular and one random. The data obtained was used to feed different positioning algorithms in three different ways: using all measurements, using only the measurements LOS and using all but adjusting the error covariance of the measurements according to their class. The algorithms chosen were three from different families: one based on LLS, one based on NLS and finally one IEKF. All these algorithms were tested with and without ignoring the measurements.

The results obtained confirmed that the use of a priori information on the nature of some measures can have a significant impact on the final positioning result. Whether ignoring measurements of this type or adjusting the covariance of the error, this paper demonstrates that the final benefit can be substantial in terms of average positioning error.

The process to complete a full analysis of a machine learning based NLOS classifier and mitigator will continue in the next Chapter 4, where a full implementation of both classifiers and mitigators will be performed. For this task, several different machine learning algorithms will be presented, as such as comparison of their results when classifying and mitigating several sets of real measurements.

3.6 Contributions related with the chapter

The content of this chapter is mainly based on the following contribution:

- Valentín Barral, Carlos J. Escudero, Pedro Suárez-Casal, and José A. García-Naya. **“Impact of nlos identification on uwb-based localization systems”**. *Proc. of 10th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Pisa, Italy, 2019

Chapter IV

Non-Line of Sight detection and mitigation

This chapter details the work done to implement and test mechanisms for classifying and mitigating the effects of ultra-wideband (UWB) measurements, made with low-cost hardware, when line-of-sight (LOS) and non-line-of-sight (NLOS) are present.

After the results described in Chapters 2 and 3, which confirmed possible improvements in location estimates by ignoring or mitigating UWB measurements from a NLOS environment, the next step was to build a classifying and mitigating system based on machine learning (ML) techniques and test it with actual measurements.

In Fig. 4.1 it can be seen a block diagram with the pipeline describing the steps needed to generate a position estimation from the values of some UWB measurements including estimates of *ranging* and received signal strength (RSS). The diagram shows the different operations that would be performed on the data from the time the beacon generates the measurement until the localization algorithm processes them to estimate a new position.

The first step is to extract the features of interest from the values of *ranging* and RSS provided by the UWB device, following what is described in Chapter 2. These features of interest are then passed through a classifier that attempts to categorize each sample according to the propagation environment in which it was captured, LOS or some NLOS variant. After the classification phase, an error mitigation process is applied on each of the samples according to an specific model for each class. After this, the measurements are finally introduced into the corresponding positioning algorithm to be processed. This last step will be covered in detail in the next Chapter 5.

The present chapter is organized as following: Section 4.1 details the measurement campaign performed to obtain the base data for training the ML algorithms and checking the obtained results. Section 4.2 explains the different machine learning techniques considered, both for classification and mitigation processes. Section 4.3 describes the experiments performed and their results and, finally, Section 4.4 presents the conclusions obtained. The last Section 4.5 shows the contributions related with this chapter.

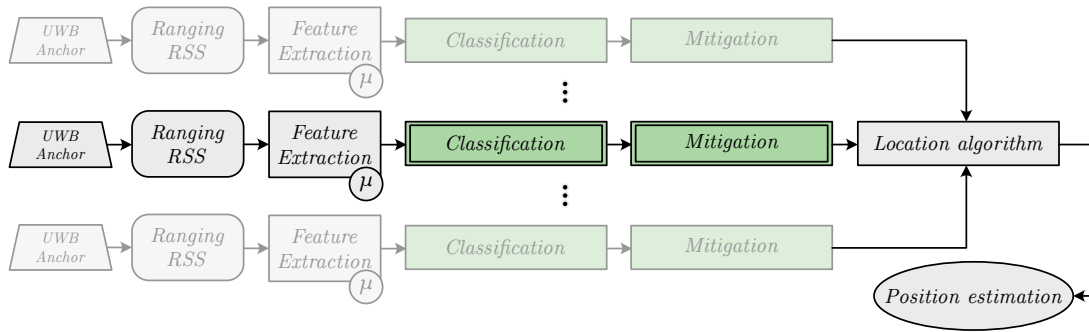


Figure 4.1: Block diagram of a location system based on ranging measurements.

4.1 Measure campaign

For this work a new campaign of measurements was carried out in order to obtain the training and test data of the different classification and mitigation algorithms. As in previous chapters, the hardware used to capture the data consisted of a series of UWB devices of the Pozyx company. Details of these devices were already described in Section 2.1. As mentioned before, Pozyx modules are based on the DW1000 UWB transceiver. A detailed discussion about this chip can be found in Appendix A.

The measurement campaign was carried out in the building of the Scientific Area, in the Campus of Elviña of the University of A Coruña. The chosen place was the same than in the measurement campaign described in Section 2.2, and the fixed anchors were placed also in the same positions. This setup allowed the capture of measurements in the three scenarios described in the previous chapters: LOS, NLOS-*Hard* and NLOS-*Soft*.

The main difference between both campaigns was the number of measurements captured and the number of measurements points were the *tag* was placed. Thus, for this last measurement campaign a tripod with 5 *tags* was used, with a separation among them of 0.2 m. This element was the same used in the works related in the previous chapter and described in Section 3.1. A picture of it can be seen in Fig. 3.2. The measurements were therefore taken at intervals of 0.2 m, and this time the range of measure started at 3 m to 15.8 m. At each position, measurements were taken for 5 minutes.

Although the measurements were recorded in the range of distances mentioned above (from 3 m to 15.8 m), finally not all were included in the study. Fig. 4.2 shows the values of *ranging* provided by the anchors. It can be seen how from 9.5 m, the Pozyx devices produced erroneous values of *ranging*. Specifically, the devices began to repeat the same exact values of *ranging* and RSS, without these being related to the actual distance between transmitter and receiver. This is a *bug* that was reported to the company responsible for manufacturing the Pozyx devices, confirming the problem and a possible solution in an upcoming revision of the *firmware*. For this reason, values beyond the 9.5 m obtained in the measurement campaign were discarded in

this work.

In Fig. 4.2 it can be seen also the notable difference between the cases LOS (green color) and NLOS *Hard* (red color). It can be seen as the slope that the values followed was clearly different, and while the first ones followed approximately the slope of the actual values, the NLOS *Hard* values seemed to follow a completely different slope as well as a greater variance. In the case of NLOS *Soft*, on the contrary, it can be seen how the behavior was very similar to the LOS case, at least up to a distance of 8m. After this distance the estimates began to differentiate and gave more random values due to the very low values of RSS that were detected and that caused a greater variance in the estimates of the device. Obviously, this 8 m distance was linked to the specific physical configuration of the measurement scenario, in a different scenario this value would be different.

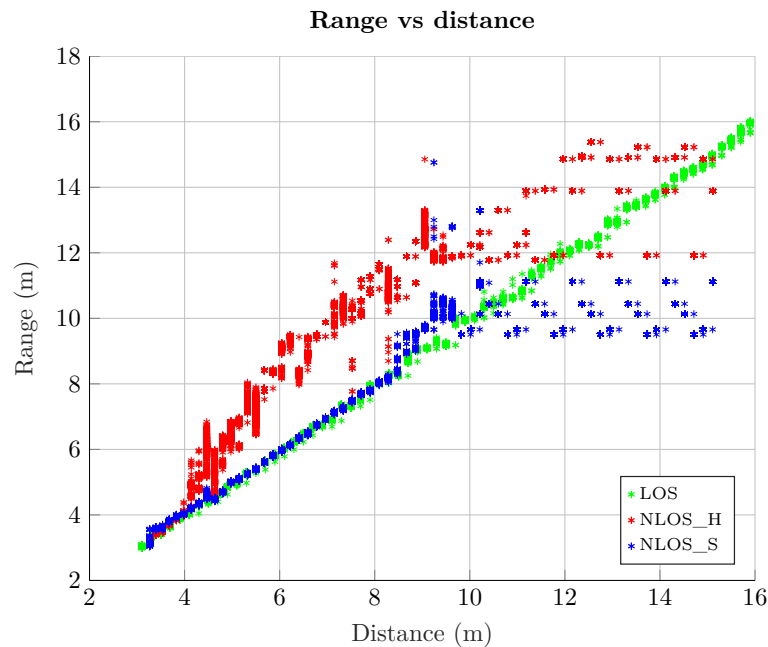


Figure 4.2: Raw measurements. Estimation vs. real distance.

Fig. 4.3 shows the values of RSS for each of the classes depending on the actual distance between the devices. It can be seen how the values of the case LOS were greater than those of both NLOS cases. For the NLOS *Hard* values, this greater attenuation was due to the fact that the receiver was always receiving a bounced signal, which had travelled so much further than the direct *path*. For its part, in the NLOS *Soft* case there were obstacle between transmitter and receiver that attenuated the received signal, but the first *path* was still received.

4.2 Machine Learning

This section presents all aspects related to the machine learning techniques used for this work. In Section 4.2.1 the different classification and mitigation algorithms implemented are

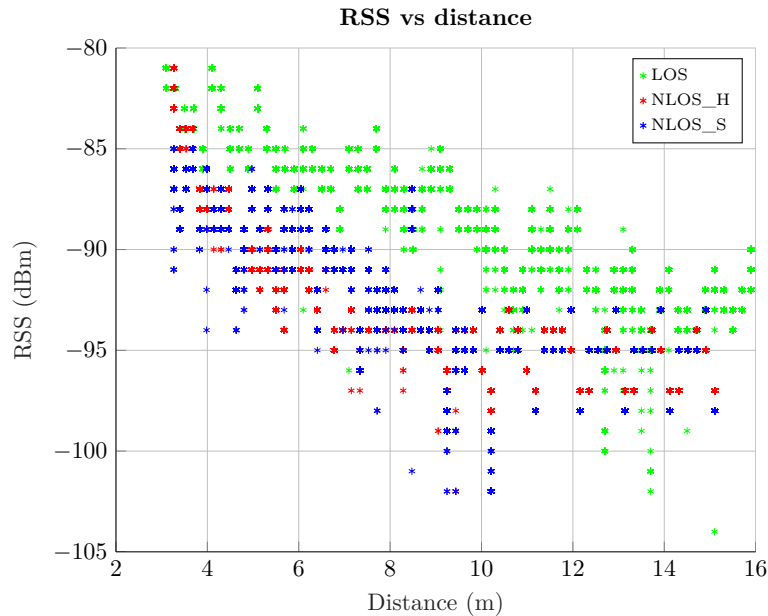


Figure 4.3: RSS vs real distance. Raw measurements.

presented. Note that the same algorithms can be used for classification and mitigation but their implementation, configuration and use are independent and different for each case.

Section 4.2.3 describes the method used to find the best hyper-parameters for each algorithm. On the other hand, Section 4.2.2 summarizes the characteristics chosen to train and test the algorithms. Finally, the Section 4.2.4 proposes a discussion on the problems derived from having a training set formed by measurements in discrete points of space and how to try to minimize the impact of that circumstance in the final results.

4.2.1 Implemented algorithms

This section presents the classical machine learning algorithms selected as a sample in this paper to perform the classification and mitigation tasks. The implementation of these algorithms was done using multiple functions included in different MatlabTM *toolboxes*. In the particular case of the classifier and mitigator based on Gaussian process (GP), an external library available in [RN10] was used.

Due to the computational cost of the training and testing for the large number of execution combinations analyzed, the implemented algorithms were compiled and packaged for use on the servers of the Fundación Pública Galega Centro Tecnolóxico de Supercomputación de Galicia (CESGA) [CES19], a super-computing center located in Santiago de Compostela.

4.2.1.1 Binary decision tree

The binary decision tree (*binary decision tree*) [BRE+84] is a simple algorithm that tries to map different input variables over other targets. Its operation is based on breaking down the input

values into different branches, until you get to the leaves where the different target values are placed. Binary decision trees can be used for both classification and regression tasks. In the first case the target variables take discrete values while in the second they can take any value in a certain interval. The decision trees are easy to use and interpret, but they have a certain tendency to make an *overfit* on the training data, that is, not generalizing well. In this work, the binary decision trees were used both in their version as classifiers and in their version for regression.

4.2.1.2 Support Vector Machine

The classic support vector machine (SVM) was implemented, for both classification and mitigations tasks. Details about this algorithm can be read in the previous Section 2.3.

4.2.1.3 k-NN

k-nearest neighbors (k-NN) is an algorithm employed in classification and in regression problems [MS00]. It is based on grouping features according to a given distance metric. There are different metrics that can be used in a k-NN algorithm: Euclidean, Mahalanobis, City block, Minkowski, cosine, etc. Besides the metric considered, another important configuration parameter is the number of neighbors taken into consideration.

4.2.1.4 Gaussian Processes

A GP is a generalization of the Gaussian probability distribution in which the distribution does not describe a scalar random variable, but the properties of a function. Based on this idea, it is possible to build regression and classification models with high accuracy and performance [RAS03].

4.2.1.5 Generalized linear models

The generalized linear model (GLM) [NW72] are a special variant of nonlinear models that use approximations with linear methods. Among the different configuration parameters, one of the most important is the response distribution type. In the case of a linear model this distribution is assumed to be normal with a mean μ . But a GLM it can follow other functions such as binomial, Poisson, Gamma, or inverse Gaussian.

4.2.2 Features used

Following the results obtained in Chapter 2, the average value of *ranging* and RSS (μ_{ran}, μ_{rss}) were chosen as the features for training and testing the classification algorithms.

4.2.3 Bayesian Optimization

Bayesian optimization is an optimization technique that can be used to find the extreme values in *black box* functions. More details about this technique were already described in Section 2.3.

In this work, Bayesian optimization has been used as a method to find the best configuration parameters (hyperparameters) of each algorithm classifier or mitigator implemented.

4.2.4 Discrete Measuring Points

Due to the discrete nature of the measurement points (the positions in which the *tag* was placed to measure its distance to the beacons), the classification and mitigation algorithms could generate very different outputs depending on which of these measurement points were part of the training and test sets. For example, by selecting measurements from the same measurement points to train and test the algorithms, one could arrive at a *overfitting* situation at those particular positions. At the other extreme, if one set of measurement points were selected to obtain the training values and a different set of measurement points to obtain the test measures, the classification algorithms would suffer a significant penalty for not having enough information during the training phase to model the behaviour of the *ranging* and the RSS in the vicinity of the measurement points.

In order to try to weigh these effects, it was decided to define a strategy for the construction of the training and test sets based on a parameter called *jumping* (j). The idea was first to separate the entire available set of measurements into two sets called \mathcal{A} and \mathcal{B} . To create these disjointed sets the j parameter was used so that:

$$d_n = 3 + 0.2(j + 1)n, \quad n = 0, 1, 2, \dots, \quad (4.1)$$

where d_n is the distance (in meters) separating the n -th measuring point (n) and the beacon used to measure (depending on the configuration LOS, NLOS-*Soft* or NLOS-*Hard*). Note that 3 is the minimum distance at which samples were captured and 0.2 m is the distance between measuring points. So, using Eq. (4.1) and different values of the j parameter the \mathcal{A} and \mathcal{B} sets can be built. For example, choosing a factor $j = 1$, the set \mathcal{A} would include the measurements captured at the measurement points $d_n = 3, 3.4, 3.8, 4.2, \dots$, while the set \mathcal{B} would include the measurements captured at the other measurement points. Obviously, for the case with $j = 0$ both sets would take measurements of all available measurement points. The impact on the final performance of the algorithms depending on this parameter j is another of the results analyzed in this work.

Once the sets \mathcal{A} and \mathcal{B} were created using Eq. (4.1) and a value of $j > 0$, it was chosen a percentage of the measurements contained in \mathcal{A} for the training set of the ML algorithms. For the test set, the remaining \mathcal{A} samples were selected and a proportional number of samples from the \mathcal{B} set were added. Finally, it is important to note that although the samples incorporated

into the training set were chosen randomly, it was ensured that there was the same proportion of measurements from all available measurement points in \mathcal{A} .

4.3 Results

This section details the results obtained after carrying out the different experiments. Specifically, Section 4.3.1 summarizes the results related to the classification process, while Section 4.3.2 summarizes the results related to mitigation.

4.3.1 Results Classification

In Fig. 4.4 it can be seen the results obtained with the different classification algorithms with two different configurations. On the one hand, when the classification considered only two classes, LOS and NLOS, and, on the other hand, when the classification was made using the three classes raised in Section 4.1, that is, LOS, NLOS-*Soft*, and NLOS-*Hard*. In this particular experiment, a jump factor of $j = 1$ was chosen.

The figure shows the value F_1 -score for each of the algorithms analyzed in the two configurations described above (two classes and three classes). This value is an indicator widely used in literature to measure performance in a classification process. Its usefulness lies in the fact that it agglutinates in a single figure the impact of the *precision* (the number of false positives) and the *recall* (the number of false negatives). The value is then calculated using the expression:

$$F_1 = 2 \cdot \frac{\textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}}. \quad (4.2)$$

When there are more than two classes to classify, there are different ways to calculate this F_1 -score [SL09]. In this work the macro-average technique was selected to do it.

In Fig. 4.4 it can be seen how the best results are obtained for the binary classifier, with values of F_1 -score above 0.9 for all algorithms. The situation is different for the case with three classes, where the similarities between the two configurations NLOS (NLOS-*Soft* and NLOS-*Hard*) cause an obvious degradation in the classification results. Thus, in this case the value of F_1 -score remains below 0.85 for all algorithms.

With respect to the differences between the different algorithms, it can be seen in Fig. 4.4 that with three classes the best results were obtained with the classifiers k-NN and the one based on GP. On the other hand, the worst results were obtained with the GLM. The results obtained are consistent with those of other similar works such [MUS+], where similar values of $F_1 \approx 0.9$ were obtained for the binary classifier based on binary trees.

The Fig. 4.5 shows the values of F_1 -score for each algorithm and each type of classification (binary or ternary) according to the value of the parameter *jumping* j defined in Eq. (4.1). It should be remembered that increasing the value of the j parameter also increases the distance

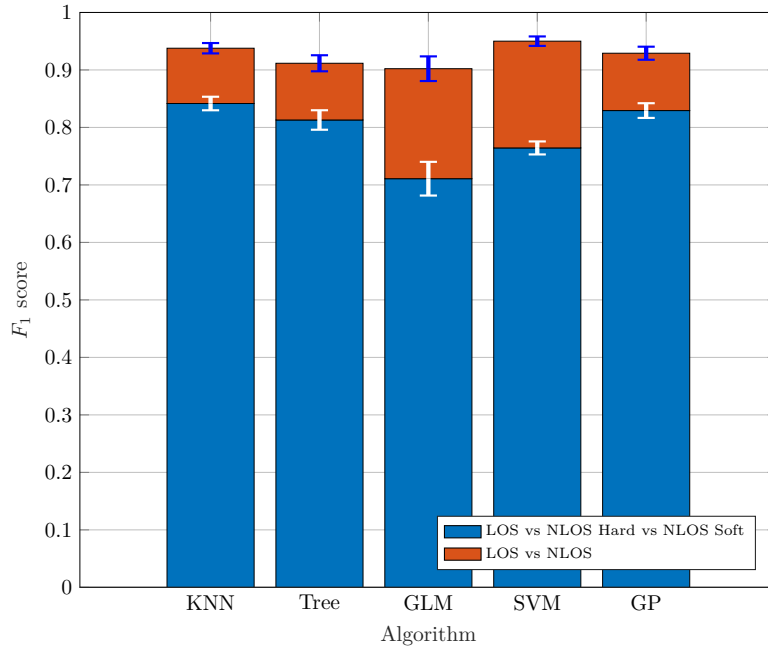


Figure 4.4: F_1 -score with jump factor $j = 1$.

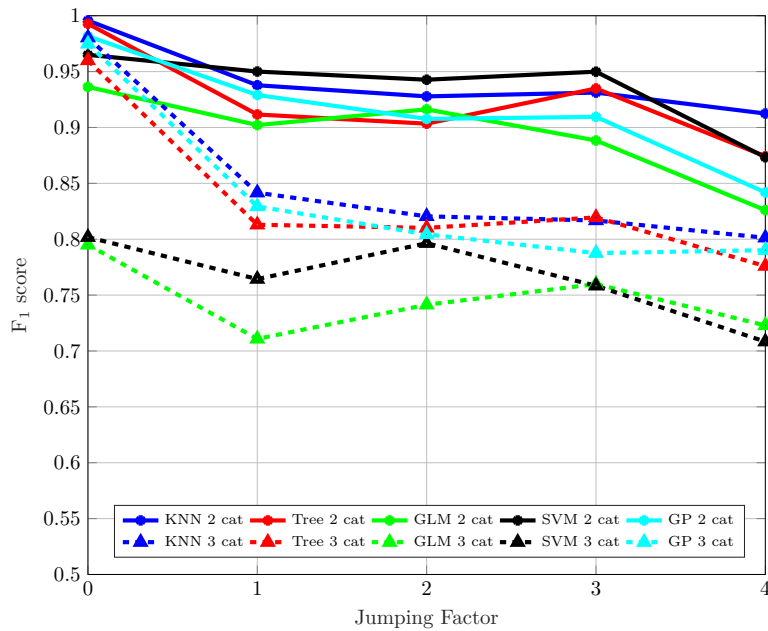


Figure 4.5: F_1 -score vs. jump factor, j .

between the measurement points used to extract the measurements from the training set, so the algorithms have a greater difficulty in constructing a model capable of later classifying the samples from the test set. Obviously, in this case $j = 0$ all the measurement points are used to extract the measurements from the training set, so this is the most favorable situation for the classification algorithms. In the figure it can be seen how the values of the binary classifier (continuous line) always obtain better results than for the ternary case (discontinuous line).

Specifically, for values of $j > 1$ better results are obtained for all the algorithms in the binary case than for the best case of the ternary variant. As for the algorithms themselves, the best results are obtained with the k-NN, the binary tree and the classifier based on GP, especially for the tertiary case where all three achieve stable values of F_1 -score without being significantly influenced by the value of the j parameter. The worst results were obtained with the GLM, both in the binary and tertiary case, while the SVM multi-class was one of the worst for the case with three classes but not so for the case with only two, where it obtained very positive results for reduced values of j .

4.3.2 Mitigation Results

In Fig. 4.6 it can be seen the mean absolute error (MAE), in meters, after the mitigation process and according to the jump factor j , for the algorithms described in Section 4.2.1. The figure also shows the confidence intervals at 95% around each of the values. The Fig. 4.6 shows the performance of the various mitigation algorithms in correcting the ranging values. Three different graphs are shown for each of the three cases considered: LOS (Fig. 4.6a), NLOS-*Soft* (Fig. 4.6b) and NLOS-*Hard* (Fig. 4.6c). In all of them, a discontinuous red line (labeled as *Raw*) is shown as a reference, which indicates the MAE of the original ranging measurements with respect to the real distance.

In Fig. 4.6a it can be seen the results related to the case LOS. You can see how in this case the original error of the measurements was already very low, around $0.1m$ of MAE. Thus, the only mitigation algorithms that manage to slightly reduce the error are GLM and GP, which are able to do so consistently for any value of j . On the other hand, the binary tree based algorithm and the SVM fail to reduce the error when $j > 0$, and actually worsen the original values.

In Fig. 4.6b it can be seen the results related to the case NLOS-*Soft*. Again the original value of MAE was low, although a bit higher than for the LOS case. In this scenario the algorithm based on GP is the one that gets a result clearly superior to the others, reducing the error value to $0.05m$ consistently for any value of j . In this case the rest of the mitigation algorithms are barely able to match the base case, while the binary tree based one clearly worsens the original error level.

Finally, in Fig. 4.6c it can be seen the results for the NLOS-*Hard* case. Here a big difference can be seen between the original error value (about $1.9m$) and the results obtained by the mitigators, all close to $0.2m$ and in some cases even below this value, as when using the algorithm based on GP. To understand why for the configuration NLOS-*Hard* such a high base error is achieved, is necessary to look at the nature of the ranging values received in this situation. When the direct line of sight between transmitter and receiver is completely blocked, the value of ranging that is decoded is the one corresponding not to the first *path* of the signal, but to a rebound. This makes the flight time obtained, and therefore the distance estimation, always longer than the actual distance. In some cases, these discrepancies between the actual

4. Non-Line of Sight detection and mitigation

distance and distance obtained by the UWB devices can be very large, resulting in the very high value of MAE.

By means of mitigation algorithms it is easy to eliminate the error caused by the rebound, as the training has been carried out with measurements whose real distance is known. However, it must be noticed that this mitigation is very dependent on the specific scenario in which the training measurements were taken. In other words, a different scenario would surely lead to the receipt of totally different NLOS-*Hard* measurements than those obtained in the scenario presented in this work. This would make it very likely that the performance of mitigators trained with measurements from one scenario would not achieve the same performance if applied to samples from another scenario. This will be demonstrated empirically with the results presented in the next Chapter 5.

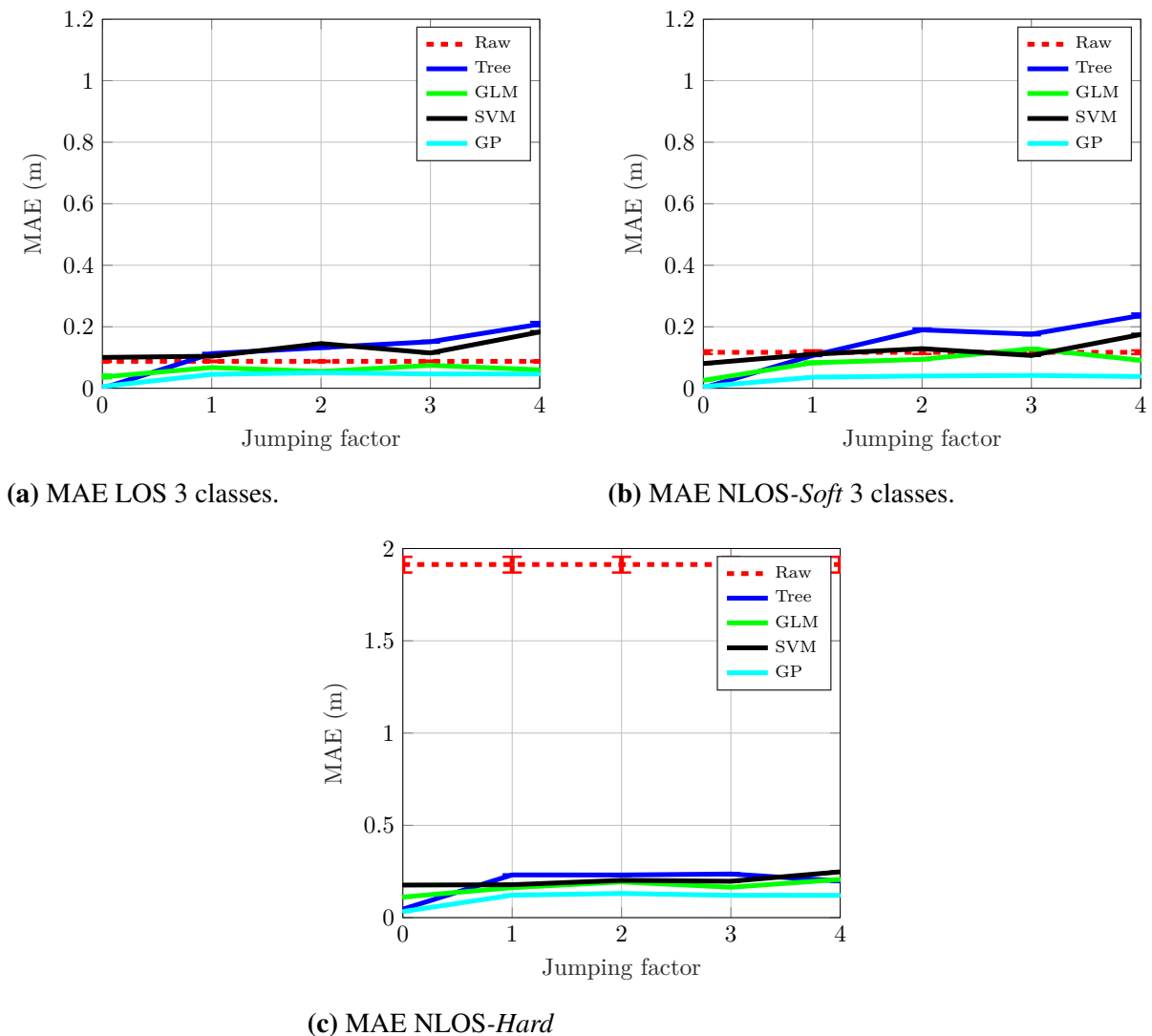
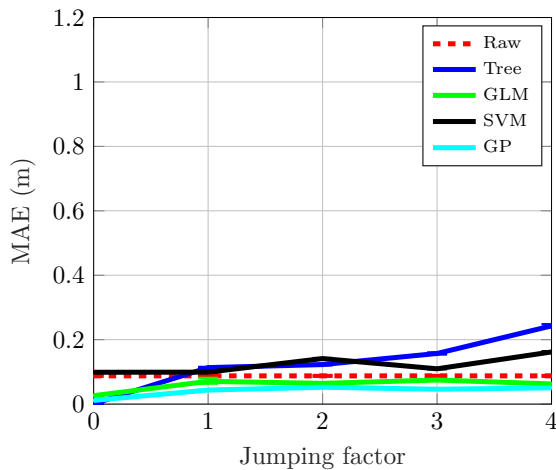


Figure 4.6: Mitigation MAE with 3 classes.

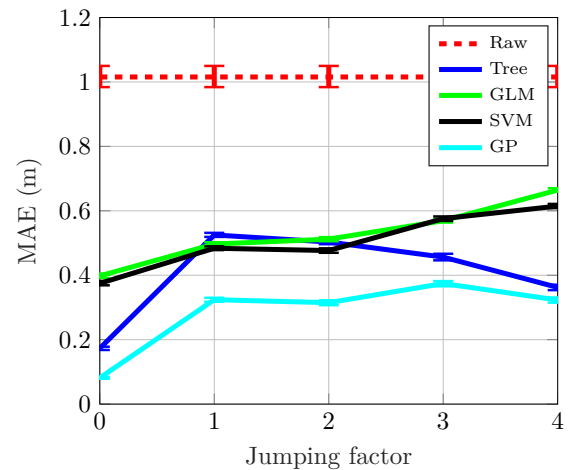
Fig. 4.7 shows the mitigation results when only two classes are contemplated: LOS and NLOS. In this case, the class NLOS includes the measurements for the scenarios NLOS-*Soft*

and NLOS-*Hard*. Obviously, the results for the case LOS (Fig. 4.7a) match those previously obtained for the same case with three classes (Fig. 4.6a), as both situations are based on the same set of measures. Again it is observed how GLM and GP get a slight improvement on the base error, which is already very small. Things are different for the NLOS case (Fig. 4.7b), where it can be seen how the MAE without mitigation is around $1m$. This is due to the fact that the average error of the measurements NLOS-*Soft* and NLOS-*Hard* is now being calculated, which as previously commented in Figs. 4.6b and 4.6c, had a MAE of $0.15m$ and $1.9m$ respectively. For the results obtained after mitigation, it can be seen that the algorithms analyzed are not capable of correcting the error as efficiently as in the cases NLOS-*Soft* and NLOS-*Hard* separately. This is because both sets present a very heterogeneous set of data, which means that mitigators must integrate information that clearly has different characteristics in each case. Among all the algorithms, the one based on GP is the one that gets the best results, reducing the original error to below $0.4m$ for any value of j , but very far from the values obtained when treating the cases NLOS-*Soft* and NLOS-*Hard* independently.

As in the ternary case, this mitigation in the NLOS scenario is very dependent on the measurement scenario, so the results are very little comparably to a different scenario. It is not the same with mitigation in the case of LOS, which will be good enough to be used in scenarios other than training with similar performance (although always slightly worse than in this situation).



(a) MAE LOS 2 classes.



(b) MAE NLOS 2 classes.

Figure 4.7: Mitigation MAE with 2 classes.

4.4 Conclusions

This chapter showed an analysis of performance of different ML algorithms applied for the classification and mitigation of real measurements of UWB obtained from a set of devices based

on this technology. Situations LOS, NLOS-*Soft* and NLOS-*Hard* have been taking into account for the experiments. Performance data was obtained from the classifiers by trying to classify the samples into two classes (LOS and NLOS) and three classes (LOS, NLOS-*Soft* and NLOS-*Hard*). The same scheme was also applied to create different mitigators in these situations and their results were compared both between the different algorithms implemented and between the different situations raised.

In the next Chapter 5 the process to analyze the proposed machine-learning based NLOS classifier and mitigator will continue. In that chapter a full system will be described, including the classification/mitigation steps as such as the final position estimation phase. Additionally, to test how general is the presented approach, the machine learning models will be trained using a set of measurements captured in a real scenario whereas the test will be performed in a totally different one.

4.5 Contributions related with the chapter

The content of this chapter is mainly based on the following contribution:

- Valentín Barral, Carlos J. Escudero, José A. García-Naya, and Roberto Maneiro-Catoira. “**Nlos identification and mitigation using low-cost uwb devices**”. *Sensors*, vol. 19, no. 16, 2019. ISSN: 1424-8220.
DOI: 10.3390/s19163464. Online access: <https://www.mdpi.com/1424-8220/19/16/3464>

The dataset with the measurements captured during this work are publicly available in the following reference:

- Valentin Barral. **NLOS classification based on RSS and ranging statistics obtained from low-cost UWB devices - dataset**. 2019.
DOI: 10.21227/swz9-y281. Online access: <http://dx.doi.org/10.21227/swz9-y281>

Chapter V

Cross validation of the location system

This chapter presents a study on how the application of classification and mitigation models based on machine learning techniques can be applied in different indoor scenarios. The aim of the work presented in this chapter was to obtain an analysis of the performance of these systems when their training set had been captured in a completely different scenario than the one used to apply the models.

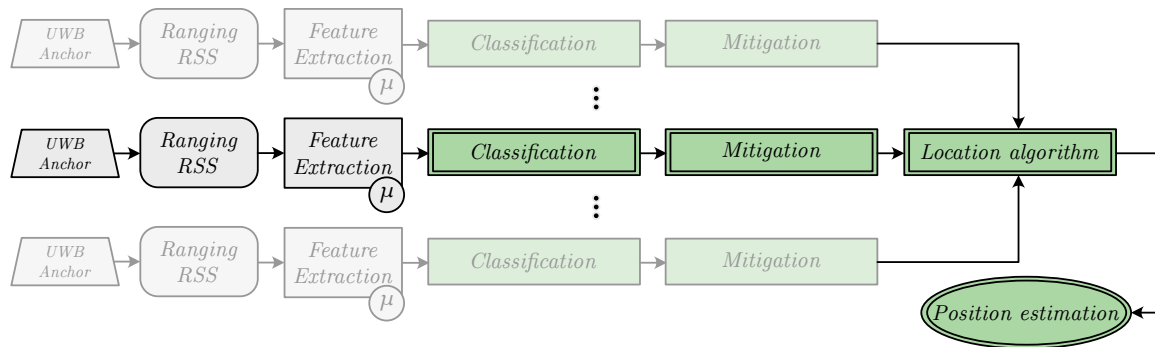


Figure 5.1: Block diagram of a location system based on ranging measurements.

As it was described throughout Chapter 3, the detection (and possible elimination or mitigation) of ultra-wideband (UWB) measurements from a non-line-of-sight (NLOS) situation results in a large reduction in the final positioning error made by the location algorithms that use those measurements. During Chapter 4 it was shown how it was possible to use machine learning techniques and a series of basic features (analyzed in Chapter 2) to implement the models capable of carrying out this classification and mitigation.

This chapter therefore deals with the last phase of the process, which is to validate the proposal in a real environment. Fig. 5.1 shows the block diagram of the proposed solution, but this time it can be seen how the blocks of *Location Algorithm* and *Position estimation* are also marked with a double line and a green color, indicating that this chapter is going to cover the full process: from the UWB measurements to the final position estimates including also the

NLOS classification and mitigation phase. There are several works that have previously used this approach (models based on machine learning and its impact on localization) such as those presented in [MAR+10; LZZ13]. All of them, however, suffer from a considerable limitation, and it is that in these works the models and the successive localization tests are always obtained on the same scenario. Although this approach is valid to obtain the limits of the solution, it leaves an open question about how general is the solution. That is, if the models trained in one scenario can be applied without modification to operate over measurement captured from another different scenario.

This chapter details the experiments carried out to analyze this situation, that is a typical situation in a real implementation. The work was developed in the following phases:

- **Performance of two measurement campaigns.** To obtain the training and test sets, measurements were taken in two different indoor scenarios. One for training the models and another to testing them. The details of these scenarios are described in Section 5.1.
- **Model training.** After the measurements were obtained, various classification and mitigation algorithms were configured and trained, including the multilayer Perceptron (MLP). Section 5.2 describes the implemented algorithms, the features used and the hyper-parameters chosen for each configuration.
- **Position estimation.** Using the classifiers and mitigators, previously trained, with the measurements captured in the test scenario, their results were used to feed some location algorithms and generate position estimates. Implementation details of these algorithms can be found in Section 5.3. The different configurations used for each experiment are described in Section 5.4.
- **Results analysis.** Finally, the estimated positions were compared with the actual positions at each instant, in order to analyze the error obtained for each of the configurations contemplated. This analysis can be read in Section 5.5.

5.1 Measurement Campaigns

In order to collect the necessary measurements to train the classifiers and mitigators as well as to carry out the final performance tests, two rooms of the Scientific Area building were chosen, in the University of A Coruña [UDC19]. For this purpose, different UWB devices were placed in different positions according to the scenario, and measurements were taken during several minutes in each one of them.

5.1.1 Hardware used

As in the previous chapters, the hardware used consisted of a series of Pozyx devices, low-cost UWB devices built around Decawave's DW1000 chip (one of the most used worldwide in commercial systems). Detailed information about this hardware can be found in Section 2.1

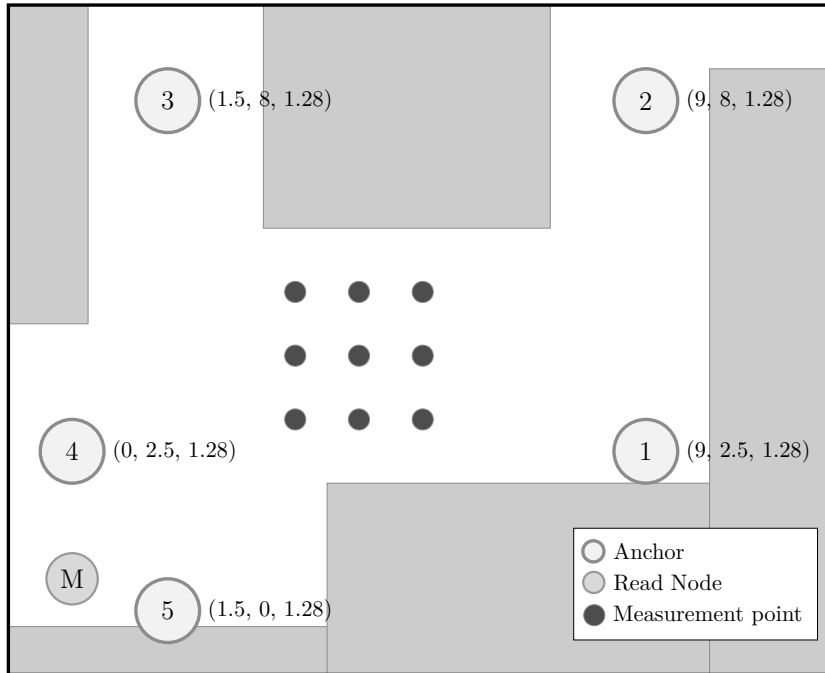


Figure 5.2: Test Scenario.

and Appendix B.

5.1.2 Measurement and test scenarios

The measurements were taken in two different zones of the Scientific Area building, located on the Elviña Campus of the University of A Coruña. In the first scenario (known as *training scenario*) measurements were obtained to train the classification and mitigation algorithms, while in the second (known as *test scenario*) measurements were taken to check the performance of the location systems in different configurations (with/without classification, with/without mitigation, etc.).

As test measurements, the results of the measurement campaign performed during the works described in Chapter 4 were selected. In that occasion the selected scenario was a corridor of the Area Científica building. Details about this area, the positions of anchors and the measure process were related in Section 4.1. Note that for the work described in this chapter and since this configuration was not presented in the test scenario, measurements from the NLOS-Soft situation were not considered for training.

The test scenario can be seen in Fig. 5.2. In this case the scenario corresponded to an open area of the building, at a confluence of several corridors and with the presence of an elevator. The measurement capture protocol was similar to that of the first scenario, although on this occasion and with the aim of demonstrating the weaknesses of the technology, the reference beacons were placed in situations that were premeditatedly not very advantageous. Five fixed beacons were used as a reference, while the measurement points in this case were marked on a

grid near the centre of the scenario. The grid had a size of 3×3 points with a separation between them of 1 m. In the test scenario the measurements were captured for 60 s at each point, for a total of 1500 measured for each position.

All the measurements captured in these measurement campaigns, as well as all the other data captured during any other work described in this thesis are made available to the scientific community in a public and totally free form.

5.2 Classification and mitigation algorithms

This section presents the automatic learning algorithms used to classify and mitigate the effects of NLOS. A comparison of different algorithms for both classification and mitigation was presented in Chapter 4. In this comparison, the algorithms that showed the best overall performance were the k-nearest neighbors (k-NN) in the classification part and the algorithms based on Gaussian process (GP) for mitigation. Information about these algorithms can be seen in the Sections 4.2.1.3 and 4.2.1.4.

In addition, and in order to complete the comparison, this time was added also the implementation of two solutions for classification and mitigation based on MLP. The Section 5.2.1 shows the details of this solution.

It should be noted that the features used in all the algorithms were those detected in the Chapter 2. Specifically, the moving average was chosen for both the *ranging* (μ_{ran}) and the received signal strength (RSS) (μ_{RSS}).

5.2.1 Multi-layer perceptron

The multi-layer perceptron is a classical neural network consisting of an input layer, an output layer and an arbitrary number of hidden layers. The problem of defining the specific number of layers and neurons per layer is usually a task that requires a process of testing and comparing the performance of each configuration. Among the different strategies that can be used for this process, Bayesian optimization is one of the most promising [SLA12]. This option was chosen to search for the parameters of the MLP created in this work. Specifically, 3 different networks were created: one to classify the UWB measurements as line-of-sight (LOS) or NLOS, another to try to mitigate the *ranging* error in LOS situations and a last one to do the same in NLOS situations. Using Bayesian optimization, the best settings for each network were [46, 14, 24], [21, 11] and [65, 69], where the size of the array indicates the number of hidden layers and each individual value the number of neurons in that layer.

5.3 Location algorithms

As in the Chapter 3, two location algorithms of a different nature were implemented to test the impact of classification and mitigation of NLOS measurements. The implemented algorithms were the one based on nonlinear least squares (NLS) using the Gauss-Newton approximation and the iterative extended Kalman filter (IEKF). Details of operation and implementation of these algorithms were described in detail in Section 3.3.

5.4 Description of experiments

This section details the design of the different experiments carried out. All experiments made use of the classification and training algorithms trained with measurements from the training scenario. These models were applied to the test scenario measurements (see Fig. 5.2), and the resulting values were then used to feed the location algorithms and check their results.

The idea behind these experiments was to test the generalization capability of the machine learning-based approach described throughout Chapters 2 to 4. In Fig. 5.3 it can be seen an overlap of the training measurement (from the first scenario) with the measurements after being classified by the classifier based on neural networks (NN) on the test scenario. It can be seen how the distribution of measurements presents slight differences between both scenarios. For this reason, tests were carried out with different configurations in order to analyze the extent to which classification and mitigation can be generalized.

As a basis for the experiments, the following combinations of algorithms were used:

- The k-NN algorithm as the classifier and the GP regression model as the mitigator.
- The NN both as classifier and mitigator.
- The IEKF as the positioning algorithm.
- The NLS with Gauss-Newton as the positioning algorithm.

These combinations were used to create different configurations in order to be able to account for the effect on the final positioning error of each of the different parts. The configurations that were taken into account were:

1. **No ignoring.** This is the configuration that serves as basis for the error. In this case the localization algorithms used the measurements from the test scenario without performing any action on them, i.e. no classification or mitigation process. The errors made in this case would be the ones expected in a situation where the UWB measurements were used directly in the positioning algorithms
2. **Ignoring NLOS without mitigation.** In this case, the trained algorithms of classification were applied to the measurements coming from the test scenario. The measurements classified as NLOS were removed from the set of values used by the positioning algorithms to calculate their estimates. In this case no mitigation was applied.
3. **Ignoring NLOS without mitigation with at least 4 beacons.** This configuration applied the same procedure as in the previous configuration, but in this case it was always guaranteed that for any iteration of the positioning algorithm at least 4 different

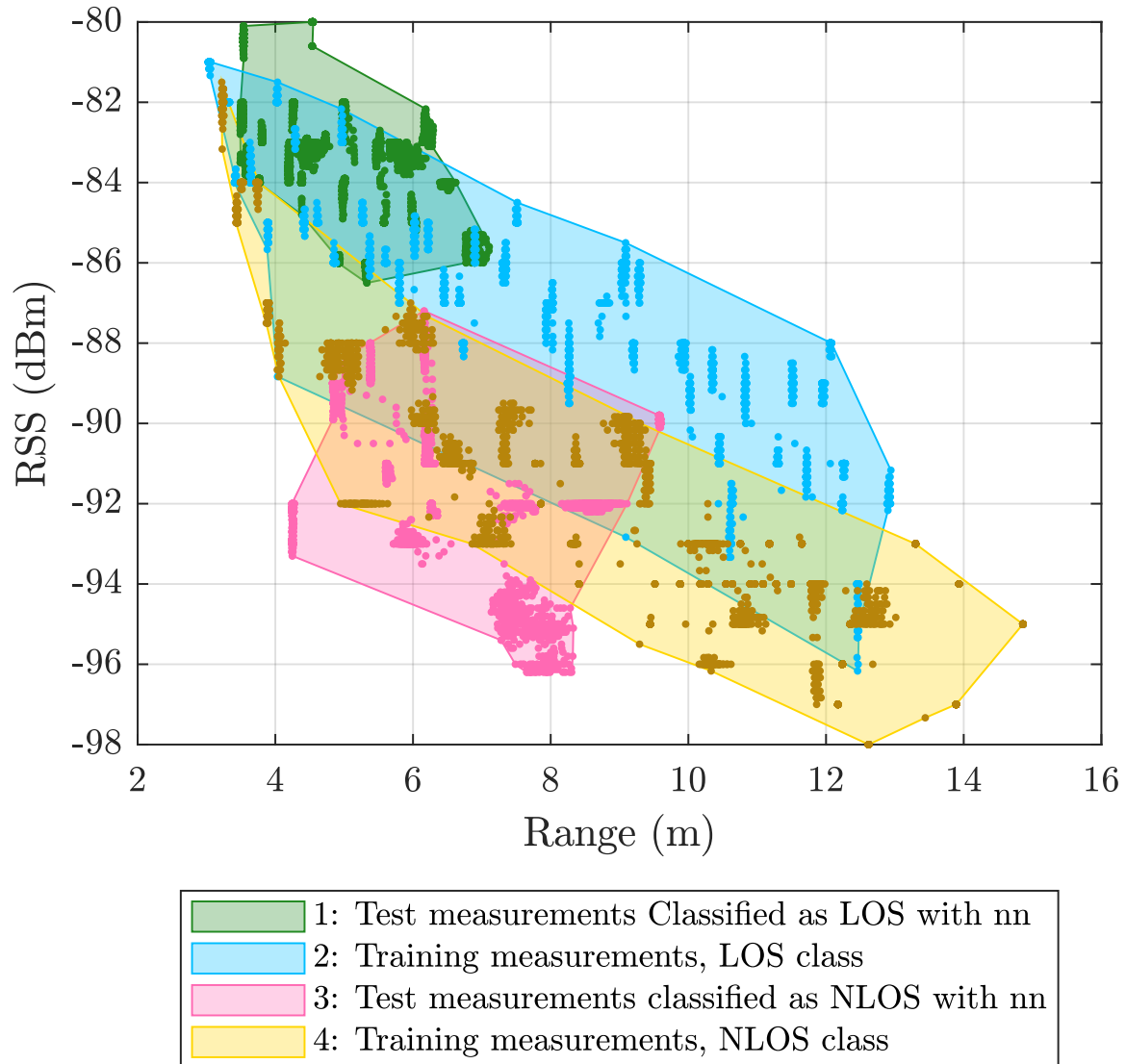


Figure 5.3: Classification results using NN.

measurements of *ranging* were available. This allowed algorithms such as the one based on NLS to work with a minimum number of values in order to generate a position estimation. If for an iteration the classifier selected less than 4 LOS measurements, the necessary samples of type NLOS were added until reaching that number. To do this, those values that were classified as NLOS with a lower score were chosen, that is, those for which there were more doubts that they really belonged to this class. This configuration was not tested with the algorithm IEKF, since it did not need that minimum of 4 values to work.

4. **Ignoring NLOS with mitigation.** In this case again the classifiers were applied and then all those samples classified as NLOS were removed. Then, on the remaining samples (classified as LOS), the mitigator corresponding to that class was applied. Once mitigated, the samples were used by the positioning algorithms to generate their estimates.

5. **Ignoring NLOS with mitigation with at least 4 beacons.** This is a combination of configurations 3 and 4: NLOS ranging values were ignored but the ones more likely to be classified as LOS were considered to ensure four ranging values in each iteration. Before running the positioning algorithm, each measurement was passed through the LOS mitigator. Again, this configuration was not tested with the IEKF, as this algorithm did not need to have at least four values to estimate a position.
6. **No ignoring with mitigation.** In this configuration the classifier is applied over the original values but no measurement is excluded in the positioning algorithms. However, before these algorithms process them, the samples of each class are modified by the mitigators corresponding to each of them.

5.5 Results

In Fig. 5.4 are shown the values of empirical cumulative distribution function (ECDF) of the localization error that were obtained for each of the analyzed configurations. In Fig. 5.4a the data obtained by using the k-NN algorithm as a classifier and the algorithm based on GP to mitigate are shown. In Fig. 5.4b the data are shown using classifier and mitigators based on NN.

The results observed in Fig. 5.4a show that, using all the original measurements (configuration 1), 90% of the positioning errors were below 0.83 m. In this case, with a relatively small number of reference beacons and with many of them in a NLOS situation in many of the positions, the error obtained was much greater than usual when working with a good UWB deployment and a clear LOS. Obviously, a real deployment of this technology would always seek to maximize the situation of LOS between tags and beacons by placing the latter at those points free of possible obstacles. However, in order to carry out the experiments described in this work, it was deliberately decided to use a scenario in which this placement was not optimal. Firstly because in many scenarios the task of placing beacons in positions that are going to be LOS all the time can be practically impossible, and secondly because performing the experiments in a complex situation such as the one described here makes easier to observe the capacity for improvement that the approach proposed in this work can provide.

The Fig. 5.4a shows how applying the mitigation (magenta line tagged with the number 4) the error did not decrease, but increased. This is because the relationship between the values of *ranging* and RSS was not exactly the same in the two scenarios, training and test, as can be seen in Fig. 5.3. Thus, the mitigators tried to correct some measurements that showed a relationship between their features that was not present in the training set that was used to train them. This caused the application of a correction factor that in many cases did not correspond to that value, causing the overall localization error to increase. In addition to this factor, another reason for this worsening in the position values was that the classifiers were not perfect, so a certain percentage of LOS measurements could be classified as NLOS and vice versa. This caused the

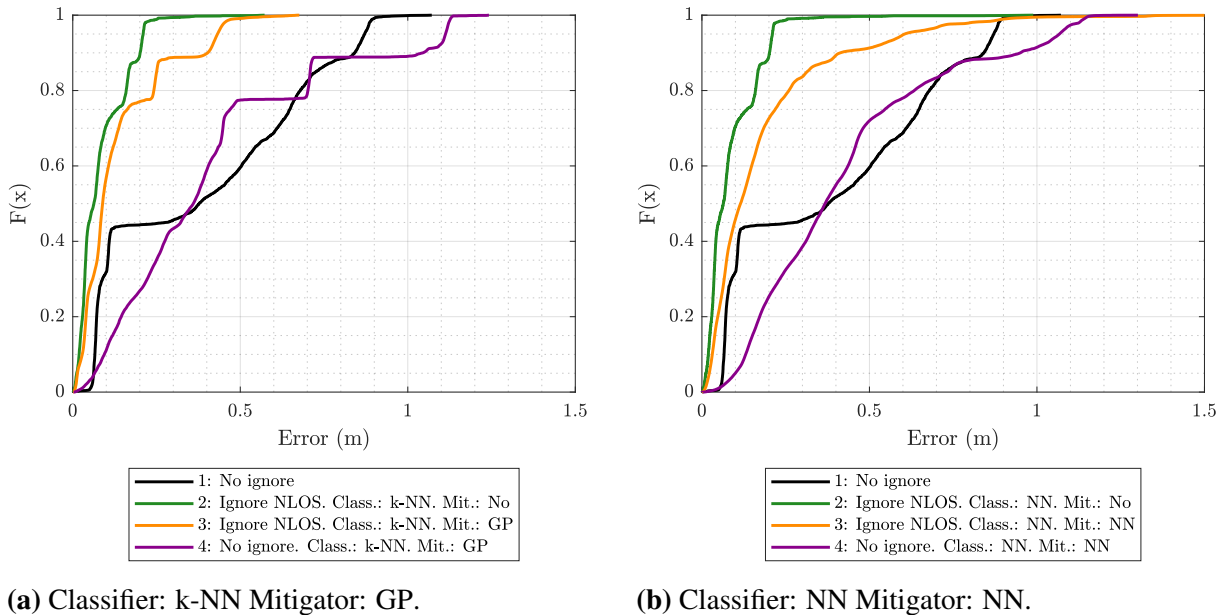


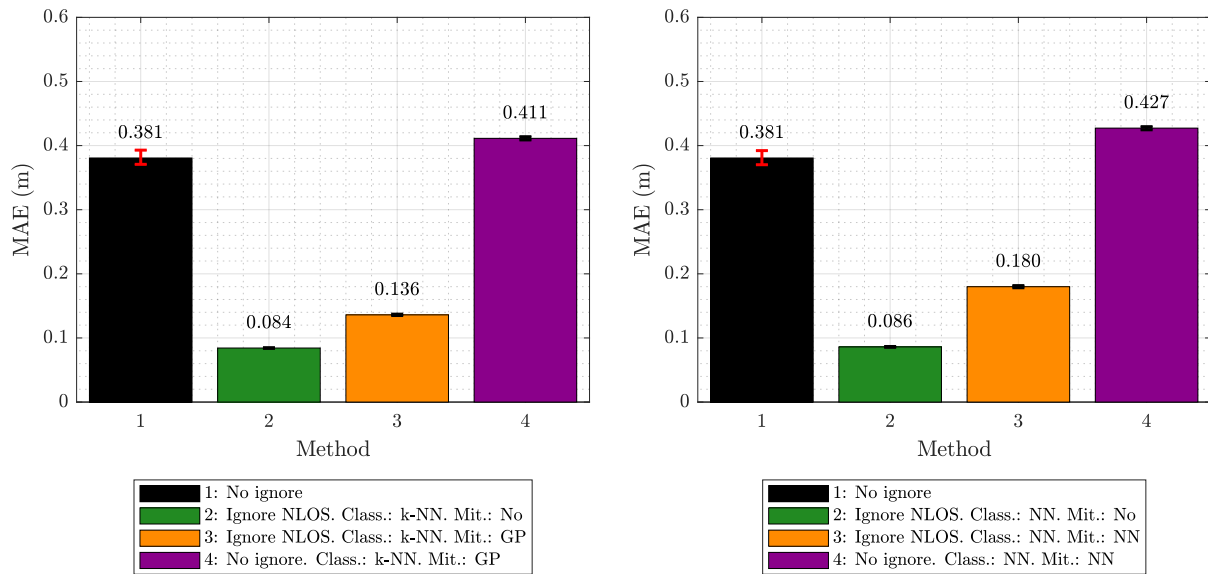
Figure 5.4: ECDF of the location error using IEKF.

mitigation factor of a different class to be applied to these samples, which contributed to an even higher level of error.

The remaining curves in Fig. 5.4a show the results after discarding the measurements NLOS (line with label number 2) and applying mitigation once discarded (line with label number 3). It can be seen how both configurations managed to reduce the original error factor in an important way, especially the configuration without mitigation that managed to go down from 0.83 m of error for 90% of the samples to only 0.25 m. This confirmed that the classifier trained with training scenario measurements was generic enough to perform its work in a different scenario without the need for retraining. In the case of applying the mitigator after discarding the measurements (line 3) a slight degradation could be observed with respect to the case without mitigation (line 2). Again, the slight differences in the relationship between *ranging* and RSS present between the samples of both scenarios, together with the classification errors, would explain this slight worsening.

Fig. 5.4b shows the ECDF using IEKF, but this time employing the NN for both classification and mitigation. Again, the best result was also achieved when the ranging values were classified and the NLOS ones discarded (configuration 2). The result in this configuration was almost the same than with the k-NN (see Fig. 5.4a), reaching 0.19 m of error for 90 % of the estimates, compared to 0.21 m obtained when the k-NN is used (see Fig. 5.4a). When the mitigation was added (orange curve, tagged as 3), again there was some performance degradation, yielding values lower than those produced with k-NN and GP mitigators. This is because, although NN obtained better results in mitigation during training, it was more sensitive to changes in the environment during the test for certain overfitting associated with it.

Fig. 5.5a shows the error MAE values for each configuration, using IEKF as the location



(a) Classifier: k-NN Mitigator: GP.

(b) Classifier: NN Mitigator: NN.

Figure 5.5: Localization mean absolute error (MAE) using IEKF.

algorithm and NN and GP as the classifier and mitigator respectively. It is observed that the base value (without applying any processing to the original measurements) was 0.381 m, while when applying the classification and ignoring the NLOS measurements (configuration 2, green bar) this value was changed to 0.084 m. The configuration in which the mitigation was applied after the discard of the NLOS measurements (configuration 3, orange bar) also showed an improvement over the base case, although in this case smaller than that achieved without mitigation (0.136 m). Finally, the configuration where no measurement was discarded but the mitigation was applied obtained the worst results, going up to an MAE of 0.411 m. Should be noticed how these results are consistent with the values of ECDF observed in Fig. 5.4a.

The Fig. 5.5b shows the location MAE for the last configuration, that is, using IEKF as the positioning algorithm and classifiers and mitigators based on NN. Again the results are similar to the case with k-NN and GP, with the best-performing configuration being the one that discarded the measurements classified as NLOS (green bar, tag 2). With this configuration the error was reduced from the original 0.381 m to only 0.086 m. The version that added mitigation after discarding the NLOS measurements also managed to reduce the error, but in this case was not able to match the results with k-NN and GP and the value remained at 0.180 m. This improvement could be explained by the fact that the version based on NN had a greater degree of adjustment (*fitting*) with the distribution of the training measurements, so that when applied to the slightly different measurements of the second scenario its performance was worse than the version GP. Finally, again the configuration that obtained the worst results was the number 4, where the mitigation was applied on all the samples without making any discard. In this case the MAE went up to 0.427 m.

In order to contrast the results, all previous experiments were repeated using a different

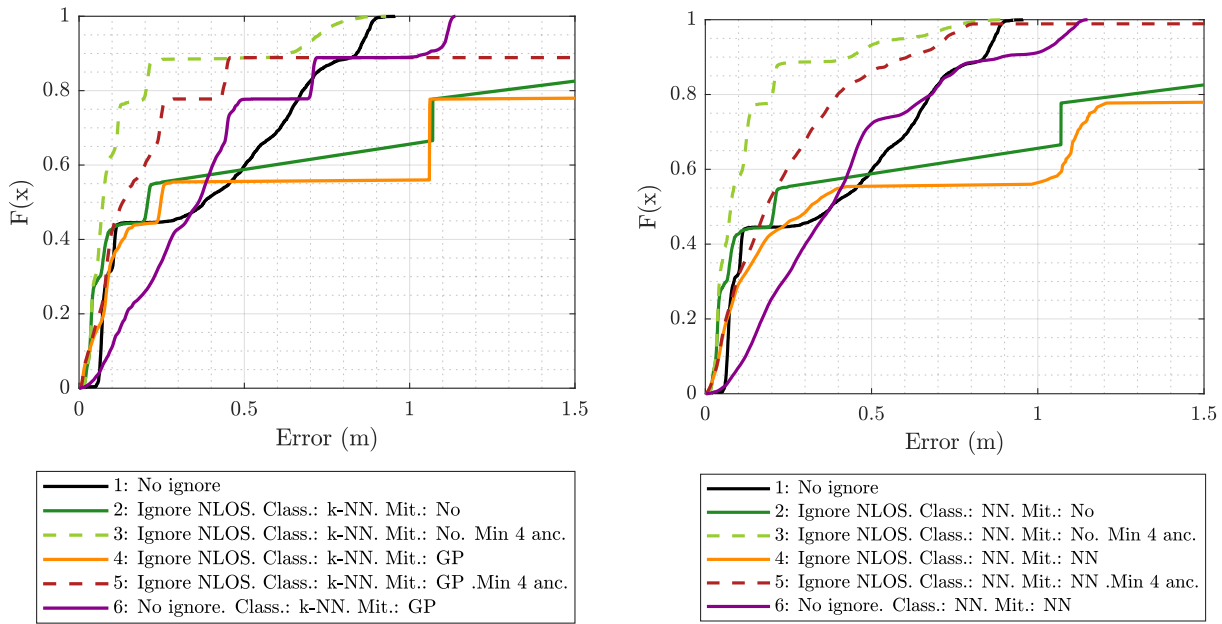


Figure 5.6: ECDF of the location error using NLS

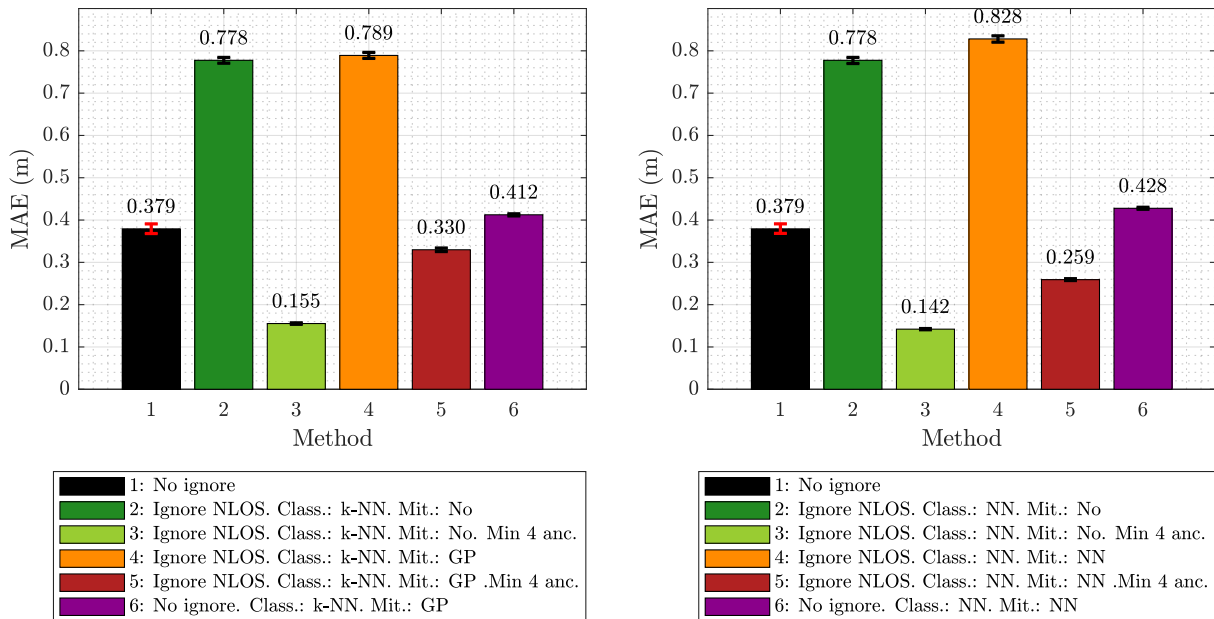
location algorithm, in this case the NLS. The Fig. 5.6 shows the graphs of ECDF for each configuration. Fig. 5.6a shows the values for the case with k-NN and GP as classifier and mitigator, whereas Fig. 5.6b shows the same values when using classifiers and mitigators based on NN.

The main differences between these values and those observed with the IEKF correspond to the 2 and 4 configurations, which are the one that discards the NLOS measurements after classification and the one that also applies mitigation after this classification.

It can be seen how in these cases the error grew exaggeratedly, surpassing 20% of the positions 1.5 m of error. This was because, when using the algorithm NLS, it is needed a minimum of 4 distance values to solve the equations and get a $3D$ position estimate. When the classification was applied and the measurements were ignored, it happened that for several iterations of the algorithm that this minimum number of measurements was not available. In that case, the NLS could not generate a new position so the last calculated position was kept (which could be very far from the actual new position of the tag). That is why these error values changed completely in the 3 and 5 configurations of Figs. 5.6a and 5.6b. These configurations were the same as 2 and 4, but in this case a restriction was applied to the process of discarding NLOS measurements. This restriction consisted in that, in case for an iteration of the algorithm the discard process had eliminated many of the measurement and there were not at least 4 values LOS, some of the measurements classified as NLOS were added to the set until that minimum was completed. To perform this process, those measurements that the classifier had classified as NLOS with the lowest probability were selected. Considering this restriction, it can be seen that the results significantly improved the base case again. In addition, similar values were

observed for the two cases, both with k-NN and GP and with NN, although the configuration with mitigation after the discard of NLOS (line 5) had lower error values for most positions using GP for mitigation, although this improvement was diluted for a small number of *outliers* where the error peaked several meters. In the version with NN that number of anomalous values was much lower and practically 99% of them was below the meter of error.

In relation to the MAE, Fig. 5.7a shows the results for the case with the k-NN and GP classifiers and mitigators respectively. It is observed that for the cases where the restriction of having at least 4 measured was not met (bars 2 and 4), the error was triggered up to a MAE above 0.7 m. On the contrary, by applying that restriction along with the discarding of NLOS measurements (bar number 3) or the discard and mitigation (bar number 5), again the errors were reduced as in the case with IEKF. Similarly, when applying classification and mitigation based on NN (Fig. 5.7b) the behavior was very similar. The main difference was obtained with respect to the final error values obtained with the best configurations (discard NLOS and discard NLOS with mitigation), which in the case with NN were slightly better than in the case of k-NN-GP. Specifically, with NN the average error was reduced to 0.142 m and 0.259 m in these two cases while with k-NN and GP it was only possible to reduce to 0.155 m and 0.330 m.



(a) Classifier: k-NN Mitigator: GP.

(b) Classifier: NN Mitigator: NN.

Figure 5.7: Localization MAE using NLS.

5.6 Conclusions

In this chapter was analyzed the performance of the complete low-cost UWB positioning system considering machine learning (ML) algorithms capable of detecting and mitigating UWB ranging measurements from an NLOS environment.

In the presented approach, unlike other similar works, the ML algorithms were tested using measurements from a scenario whereas the tests were carried out in a different one. This was useful to know to what extent the proposed solution could be generalized to be used in different scenarios without the need to carry out a new campaign of measurements in each one of them.

For this purpose several experiments were performed and described in this chapter. The results showed that, although the classification task seemed to work well even in a different scenario, the mitigation had more problems due the slightly different relation between *ranging* and RSS in both areas. The final Chapter 8 of conclusion and future work will show some ideas to improve that situation.

In the next Chapter 6 will be related the details of a high-level UWB software simulator based on real measurements and ML focused on location purposes. This simulator was used, among other tests, to replicate the experiments related in this current chapter. The results showing the differences between the real world and the software simulation are presented in that next chapter.

5.7 Contributions related with the chapter

The content of this chapter is mainly based on the following contribution:

- Valentín Barral, Carlos Escudero, José García-Naya, and Pedro Suárez-Casal. **“Environmental cross-validation of nlos machine learning classification/mitigation with low-cost uwb positioning systems”**. *Sensors*, 2019. Under second revision.

The dataset with the measurements captured during this work in the test scenario are publicly available in the following reference:

- Valentin Barral. **Environment cross validation of NLOS machine learning classification/mitigation in low-cost UWB positioning systems - dataset**. 2019. DOI: 10.21227/rhhs-fw33. Online access: <http://dx.doi.org/10.21227/rhhs-fw33>

Chapter VI

Simulation of Ultra Wideband based location devices

This chapter presents the details of designing and implementing a simulation platform for indoor location systems. The objective of this solution was to provide a realistic mechanism as possible to test the location algorithms before taking them to the real world. Although the objective at the time was to use this platform for industry 4.0, to develop and test a forklift positioning system, the models and systems developed were general enough to be used in any other similar task.

As seen in previous chapters, indoor localization is still an open challenge due to its multiple sources of problems. The effects on radio signals of the various obstacles inside buildings, magnified in other environments such as industrial ones, sometimes make it a challenge to achieve an accurate position estimate all the time. This is why in many cases, especially when it comes to such complex indoor environments, it is necessary to use redundancy to achieve the best result. That is to say, to use different sensors of different nature to obtain a position estimate as accurate and precise as possible. However, it is not always easy to predict in advance what will be the result of a given combination of sensors for a particular scenario. It is in these conditions when a realistic computer simulation can be of great help as a first step to analyze the problem and find the best solution.

Throughout this chapter we will describe the necessary elements to carry out this simulation, applying them on one of the classic examples in indoor location: the positioning of a forklift inside an industrial building. Thus, Section 6.1 shows an introduction to the problem and how a multi-sensor approach can be the solution. Section 6.2 describes the sensors used in the simulation, while Section 6.4 shows the details of a high-level ultra-wideband (UWB) sensor simulator. This simulator was created using the data obtained from the measurement campaigns described in previous chapters, so its values correspond to real measurements. Section 6.5 describes the positioning algorithm chosen for the simulation, in this case an iterative extended Kalman filter (IEKF). That section details both the implementation and the movement model considered to represent the forklift. Section 6.6 shows the implementation details of all the software created for the simulation. Section 6.7 shows the results of the simulations, in which

the accuracy of the location obtained is compared with different combinations of sensors and in different indoor scenarios. Finally, Section 6.8 shows a final experiment comparing the positioning results obtained in the test scenario described in Chapter 5 with the results obtained after replicate that scenario in the simulator.

6.1 Problem statement: Locating pallets

The typical work in a factory, warehouse or, in general industrial environment needs the movement of objects from one point to another. Depending on the type of object (size, weight, dangerousness, etc.) it can be transported in different ways. When dealing with medium sized loads, one of the most common methods of carrying out this transport is through the use of pallets and forklifts. Thus, the usual workflow in this case is to place the load on a pallet, use the forklift to lift the pallet along with its load and move both things to the end point. The forklift have great mobility within a factory, which makes them ideal for tasks that require great flexibility. That is why they do not usually stick to a fixed route, but travel the area based on the specific transport needs at any given time. This dynamic capability makes it possible to pick up and unload pallets at any point, which in principle is an advantage over other systems that use a more rigid movement scheme. This can however lead to a logistical problem if control over the position of the pallets is lost and you need to waste time searching for a particular one. That is why the task of locating this type of elements becomes of vital importance.

As it could not be otherwise, in the market there are already different commercial alternatives that seek to solve this problem. Companies such as Vero Solutions [VER19] or Tech Solutions [TEC19] rely on WiFi, RFID or video cameras to track pallets. The operation includes the use of an RFID reader on each forklift truck and a series of fixed loading/unloading points where other readers are responsible for controlling what enters and leaves the factory or warehouse. Other alternatives such as Touchpath [TOU19] use a similar philosophy, but in this case placing a multitude of RFID readers on the ceiling.

All these systems suffer, however, from some problems. Many of them base their tracking on the fact that the loading and unloading of pallets takes place in certain areas, which can limit the operation. In other cases, the technology used (such as long distance RFID readers) can be enormously expensive depending on the number of elements to be positioned (since normally one or more tags will be needed for each element).

In small or medium sized factories the number of pallets can already be very large, so trying to locate each of them individually can be a very complicated task. Using electronic equipment for each one of them can have an excessive cost, while other techniques such as the use of cameras and artificial vision can lead to privacy problems (in addition to the need to add a considerable number of them to cover the entire work area).

In general, any system that aims to perform a localization task such as locating pallets in an industrial environment would have to deal with the following problems.

- The multi-path. It is common that in a factory there are numerous metallic elements. The effect of these elements on RF signals is a very important source of problems.
- Obstacles. In a factory it is very common the presence of obstacles (machinery or people) throughout the area of work. This can also have adverse effects on certain radio technologies or based on artificial vision.
- Operational continuity of the localization process. In an industrial environment the positioning system should not affect or condition the usual workflow of the factory.

With these restrictions in mind, a solution based on a single technology does not seem to be sufficient. On the contrary, the simultaneous use of sensors of different natures seems to have a number of advantages:

- Strength: Since each type of technology often has its own weaknesses, a combination of several can distribute this weakness in different parts, since not all technologies will fail in the same situations.
- Flexibility. New requirements or changes in operations can add new sensors that improve the system.
- Granularity. Normally, the same degree of precision is not required in all areas of a factory/warehouse. A multi-sensor approach allows you to focus efforts on certain areas of interest while maintaining a less accurate solution in other less important areas.
- Cost. Different technologies mean different hardware alternatives, so it's easier to find an option that fits every budget.

6.1.1 The proposed solution

The solution proposed in this paper uses a different approach. Instead of locating each individual pallet at all times, the idea is to locate only the forklift trucks in real time and identify the pallets only at the time they are loaded or unloaded. This solution therefore has two important requirements: on the one hand it must be possible to track the position and orientation of the forklift trucks and on the other hand a method must be available to identify which pallet is being loaded or unloaded.

This approach provides a number of benefits over solutions that attempt to locate all pallets at all times:

- The number of trucks will always be smaller than the number of pallets, so the hardware deployment will also be smaller.
- The cost of maintenance is also much lower, as there is no need to maintain a huge set of devices.
- Different sensors can be used to locate the trucks, depending on the specific requirements of each installation.
- Forklift positioning data can be used to improve the company's workflow, optimize processes or increase safety.

6.2 The chosen sensors

In order to be able to simulate the solution described above, it was decided to select a series of real sensors and then try to represent them as faithfully as possible within the simulation. Thus, the UWB technology was chosen as the basis for the location system, due to its precision in position estimate. As hardware based on this technology, the well-known Pozyx devices were chosen, already used in the works described in the previous chapters. However, as it was shown before, in adverse circumstances (no direct line of sight) the performance of such devices can decline greatly. That is why, in addition to the Pozyx devices used in the previous chapters, it was decided to use additional sensors to make the system more robust. First, it was decided to use the inertial sensors already present in these devices. Specifically, the Pozyx include the Bosh BNO-055 chip, which integrates a triaxial 14-bit accelerometer, a triaxial 16-bit gyroscope with a range of ± 2000 degrees per second and a triaxial geomagnetic sensor. Pozyx hardware was already described in detail in Section 2.1.

Another additional sensor named PX4 Flow was chosen, which is an optical-flow smart camera capable of operating in indoor and outdoor scenarios [HON+13]. The PX4 Flow sensor exploits the differences found between consecutive camera frames together with the data obtained from inertial sensors to estimate the velocity along both X and Y axes. An additional acoustic sensor provides height estimates, hence the relative displacement values can be translated into real units such as meters per second (m/s). Therefore, the PX4 Flow sensor provides linear motion estimates along both X and Y axes, which can be easily translated to an orientation angle with respect to the Z axis. The PX4 Flow also provides a quality value for each measurement based on the quality of the processed image. The light in the environment, the ground type and texture, and the speed of the target can degrade the image quality significantly. Thus, poor light conditions and a uniform floor can make the sensor fail, whereas a scenario well illuminated with a *noisy* floor (with irregularities, stains, etc.) is the best environment to get the sensor working at its maximum potential.

6.3 Simulation platform

To perform the simulation of the environment and the proposed solution, the Gazebo physics simulator [GAZ19b] was the chosen platform. Gazebo [GAZ19b] is a multi-platform software consisting of several components and focused on the virtual simulation of real physical environments. Gazebo offers the following capabilities:

- A set of physics engines. Gazebo supports different physics engines such as ODE, Bullet, Simbody, or DART. They allow for simulating different physical processes such as movement, collisions, and falls.
- The OGRE 3D render engine, allowing for creating realistic 3D object models and worlds.
- Several sensor models, such as optical or motion ones, supporting noise addition to the

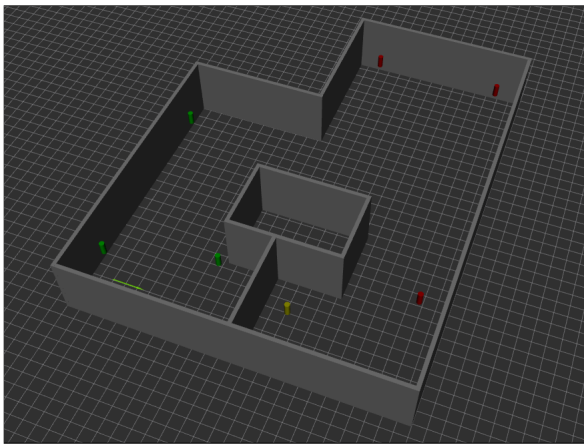
sensor data following different models.

- A plug-in system allowing for creating new functionality and extend its base behavior, including the creation of new sensor types or new environment characteristics.
- There is a large community of developer contributing with sensors models for this simulation. One of them, the PX4 Flow is used in the work presented in this chapter.

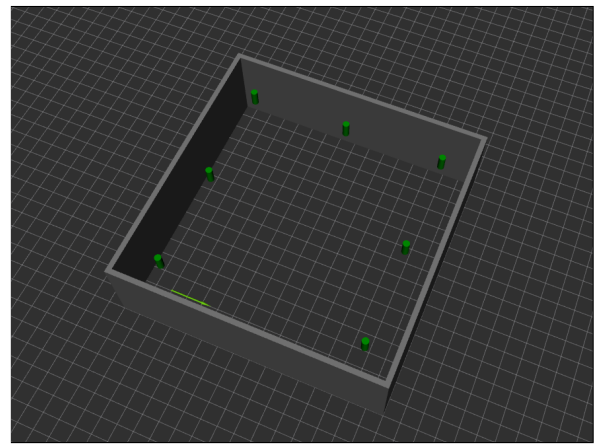
Gazebo allows to create a world an run simulation inside it. But to interact with this world and extract the data, is necessary a extra layer of software. In this work this layer was built with Robot Operating System (ROS). This software offers several methods to operate over Gazebo and extract the results of the simulations. ROS is described in detail in Section 6.6.

6.3.1 Simulated Scenarios

Two different buildings were modeled for this work: Scenario A, shown in Fig. 6.1a, and Scenario B, plotted in Fig. 6.1b. In Scenario A different interior walls were placed in order to serve as obstacles between the UWB tag and the anchors in certain positions with the aim of obtaining ranging values, including values with large errors produced when some of the anchors are under non-line-of-sight (NLOS) hard situations. Contrarily, Scenario B was designed without objects inside the movement area of the forklift with the intention of checking the performance of the positioning when a strong LOS with all UWB anchors is guaranteed.



(a) Scenario A.



(b) Scenario B.

Figure 6.1: The two simulated scenarios.

Both scenarios were designed with a simple ground plane of $20\text{ m} \times 20\text{ m}$ with the standard gravitational acceleration (9.8 m/s^2). The ground was painted with a texture to provide a visual reference for the PX4 Flow simulated sensor. This texture had to be adjusted in size and pattern draw terms to make the sensor work. Depending on the vehicle speed and the sensor height above the ground, the texture is selected in a way that the simulated PX4 Flow is able to track the movement. A texture too simple or out of focus caused the sensor to fail, as well as a texture

with a pattern too regular. However, in an industrial environment the floor is not so perfect, and therefore this condition is no to restrictive.

6.3.2 Simulated Vehicle and Sensors

In order to simulate the forklift within Gazebo, the 3D mesh and sensor configuration of a conveniently scaled *turtlebot3 burger* robot ([ROS19e]) were used. It was added an arm that protruded 1 m by one of the sides of the vehicle to place a camera and simulated the PX4 Flow sensor. A vertical pole was also placed in the center of the vehicle at a height of 1.5 m to place a cube on it simulating the UWB tag. Finally, a inertial measurement unit (IMU) sensor was also added to the model at the same position of the UWB tag, trying to imitate the configuration of a Pozyx device. The noise parameters of the IMU are specified below. These are the default parameters for the IMU sensor in Gazebo:

- Angular type: Gaussian
- Angular $\mu = 0.0$
- Angular $\sigma = 2e - 4$
- Angular bias $\mu = 7.5e^{-6}$
- Angular bias $\sigma = 8e^{-7}$
- Linear type: Gaussian
- Linear $\mu = 0.0$
- Linear $\sigma = 1.7e - 2$
- Linear bias $\mu = 0.1$
- Linear bias $\sigma = 0.01$

The noise parameters of the camera associated to the PX4 Flow sensor are specified below. As in the case of the IMU, these parameters are the values by default in the plugin that models the behavior of the sensor:

- Type: Gaussian
- $\mu = 0.0$
- $\sigma = 1e^{-4}$

To simulate these sensors (UWB tag, IMU and PX4 Flow) inside Gazebo, several plug-ins were used. The *Flow_Plugin* plug-in included in the PX4 Autopilot Firmware [PX419] was employed to create the PX4 Flow, while the internal inertial plug-in included in Gazebo was chosen to simulate the IMU. In order to simulate the UWB tag, it was necessary to implement a suitable plug-in from scratch, as there was nothing similar published. The details of this development are described with detail in Section 6.4.

6.4 UWB simulation

To simulate the behavior of a set UWB devices inside Gazebo, an UWB simulator based on real UWB measurements was implemented. This simulator consisted of two components: a Gazebo [GAZ19b] plug-in to simulate an UWB tag and a set of models representing the UWB anchors. To create the simulation scenarios, the building editor available in Gazebo ([GAZ19a]) was used. With this editor it is possible to create walls, windows and doors, as well as to define buildings with a different number of floors. The size and position of the walls is fully editable. As mentioned in Section 6.3.1, several scenarios were defined to test the simulator under different configurations.

The main component of the UWB simulator was the Gazebo UWB plug-in. This plug-in was created to generate ranging estimates between the tag and each anchor in the scenario. To make this estimate more realistic, a set of UWB measurements was used to model its behavior. Such measurements were obtained from the measurement bank publicly available in [BAR19e], which was created from the experiments described in Chapter 4. This repository contained measurements of the three situations identified in that chapter. These situations were:

- *line-of-sight (LOS)*: In this situation there were no obstacles between the tag and the anchor. With this configuration, the distance estimate provided by the devices was very close to the actual distance between them.
- *NLOS Soft*: In this situation there was an obstacle between the tag and the anchor. Despite this, the devices were able to decode the correct path, hence the distance estimate was also close to the real value. However, as a result of the attenuation caused by the obstacle, the received power level was significantly lower than that in the LOS case.
- *NLOS Hard*: In this situation there was an obstacle that totally blocked the direct line of sight between the tag and the anchor. Unlike in the NLOS Soft case, the first direct path of the signal was totally blocked, hence one of the delayed paths (corresponding to signal rebounds) was decoded instead. In this case, and because the time of flight (TOF) of the signal was considerably greater than the minimum distance, the estimate provided by the UWB devices was always greater than the actual distance. In this situation, measurements were obtained with several centimeters of error, exceeding the meter in some cases.

To develop the simulator, the measurements from these three scenarios were used to train different neural networks (NNs). The final number of layers, neurons and activation functions of each NN were automatically selected by a Bayesian optimization process [SLA12].

The first implemented NN had as goal to estimate the offset observed between the actual distances and the estimates collected by the Pozyx devices for the LOS and NLOS *Soft* cases. This offset is a feature of the DW1000 chip [DEC19c] used in the Pozyx devices, and depends both on the distance and the received energy [BAR+16a] (more about that topic can be read in Appendices A and B). In the LOS and NLOS *Soft* situations, when the actual distance between the devices was known, it was possible to train an NN so that an estimate of the error offset could

be obtained. This approach, however, could not be extended to the NLOS *Hard* case in which a distance estimate from a signal rebound causes that the actual distance traveled by the signal be unknown. That is, the value estimated by the devices in this case corresponds to the sum of all the signal rebounds plus the offset caused by the DW1000 used in Pozyx devices. However, the proportion corresponding to each of these components is unknown. For the simulator, and taking into account that the NLOS *Hard* case is very dependent on the morphology of the scenario, was decided to not use any method to estimate such an offset.

Three other NNs were developed to estimate the received power value, its variance, and the variance of the own ranging estimate for each of the three configurations contemplated. In this way, three models capable of providing the mentioned values from the distance value existing between the tag and the anchor simulator were obtained.

In addition to these models, power limits were estimated from which the devices began to exhibit an anomalous behavior, either failing to generate the samples, or generating repeated values regardless the actual distance (as was described in Section 4.1). These limits were later used in the simulator to avoid generating values in case the power estimates do not reach these minimum values.

All of the above models and values were used to develop the Gazebo plug-in publicly available in [BAR19d]. The mode of operation of this plug-in is as follows:

1. The tag modeled with the plug-in searches the stage for the elements marked as UWB anchors.
2. For each anchor, the plug-in uses ray tracing to check if there is a direct line of sight with the tag. Then, if the power estimate is high enough, the range is considered as belonging to the LOS scenario, hence a measurement corresponding to the distance between them is generated (after applying the offset correction factor).
3. If there is an obstacle between the tag and the anchor, the thickness of the element (i.e. obstacle attenuation) is checked. If its value is below a threshold (configurable in the plug-in), then it is considered that the first path of the signal is able to pass through it. Therefore, the NLOS *Soft* configuration is applied and the ranging value is generated. Again, if the estimated power is below the threshold, then the ranging value would not be generated.
4. If there is an obstacle between the tag and the anchor, but its thickness exceeds the level established to be considered NLOS *Soft*, we proceed to start a search for possible bounces. In this case, and in order to keep the computational cost reduced, ray tracing is employed to look for bounces in obstacles located at the same height as the tag (i.e., walls) and in the floor. In addition, the search is limited to a single bounce before reaching the target. In case that after this rebound there is a path free of obstacles between the tag and the anchor, it is established that the scenario corresponds to NLOS *Hard* and both ranging and received power estimates are generated. As in the previous cases, if the received power estimate is below the threshold, the acquired information is discarded, including

the ranging estimates.

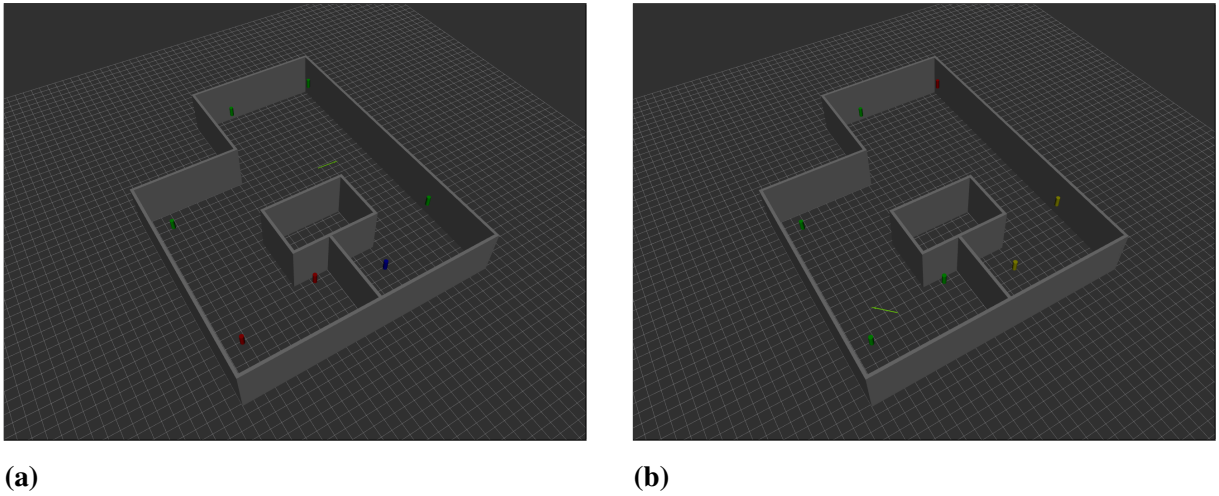


Figure 6.2: The color of the anchors indicates their state: green - LOS, yellow - NLOS *Soft*, blue - NLOS *Hard*, and red - out of range.

The developed plug-in also publishes the current state of the anchors, marking each of them with a different color depending on whether their way to the tag is LOS, NLOS *Soft* or NLOS *Hard*. Fig. 6.2 shows, in the same scenario, how the anchors change their state depending on the position of the tag.

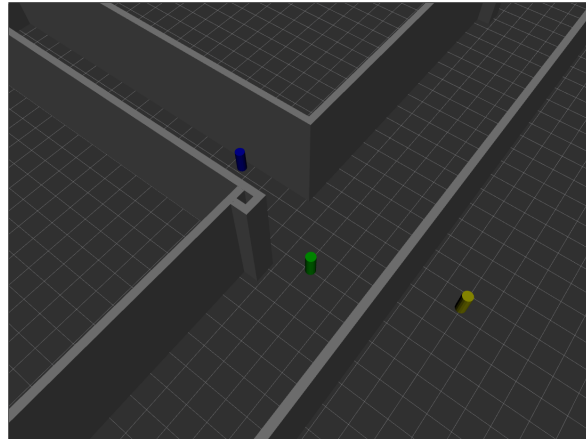
6.4.1 Simulation accuracy

To show the validity of the new plug-in, the original scenario was replicated in Gazebo where the UWB measurements were carried out. To do this, Gazebo’s building editor was used and the walls were created according to the distances described in Chapter 4. Fig. 6.3 shows an image of the original scenario together with its 3D simulated equivalent. Then the three UWB anchors were placed as it was arranged during the measurement campaign, and the simulated tag (with its plug-in) was placed at the same positions.

The ranging and received received signal strength (RSS) values from the measurements and the UWB simulator are shown in Figs. 6.4 and 6.5 respectively. It can be seen that the simulated values approached the measured ones, especially for the LOS and NLOS *Soft* cases, where there are no rebounds and the offset is mitigated. For the NLOS *Hard* case the differences become larger, since in this case the bounced signal takes different paths in both scenarios. While in the simulation was approached with a single rebound, in the actual measurement scenario the situation was much more complex and difficult to predict. However, the important key about these simulated values was that their behavior was similar —at a high level— to the measured values, since clearly the values in the NLOS *Hard* scenario were far from the real distance values. This is what later made it possible to detect the weaknesses of a location system based only on UWB for scenarios in which LOS is not guaranteed in all points. Finally, Fig. 6.6 plots



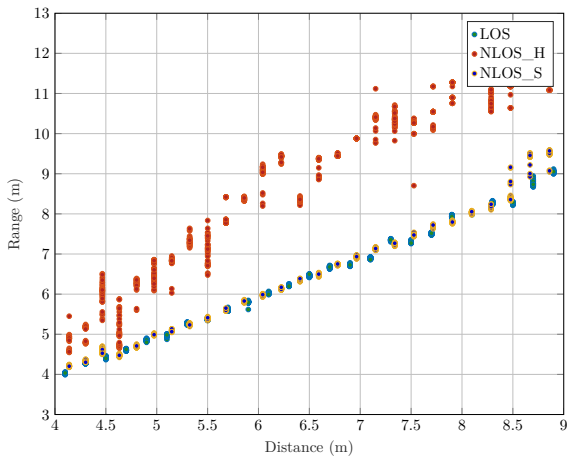
(a) Picture of the corridors showing two of the three anchors and the tag employed in the UWB measurement campaign.



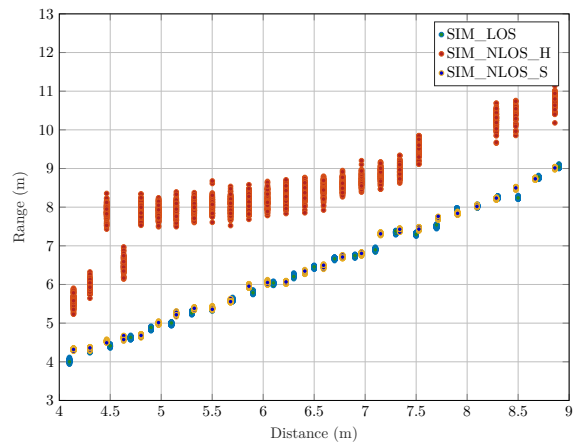
(b) Simulated scenario in Gazebo.

Figure 6.3: Measurement campaign scenario —described in Chapter 4— and its replica in Gazebo.

the mean and standard deviation of the measured and simulated ranging values, showing that the simulated LOS and NLOS *Soft* ranging values were very close to the measured ones.



(a) Measured ranging values.



(b) Simulated ranging values.

Figure 6.4: Simulated ranging values corresponding to the measurement scenario and comparison with the measured ranging values.

6.5 Location Algorithm

The location algorithm implemented to estimate the positions using the values coming from the simulated sensors was a IEKF.

In this specific implementation, the forklift motion model was based on the setup that can be seen in Fig. 6.7, which shows the selected sensors placed on the roof of a forklift together with

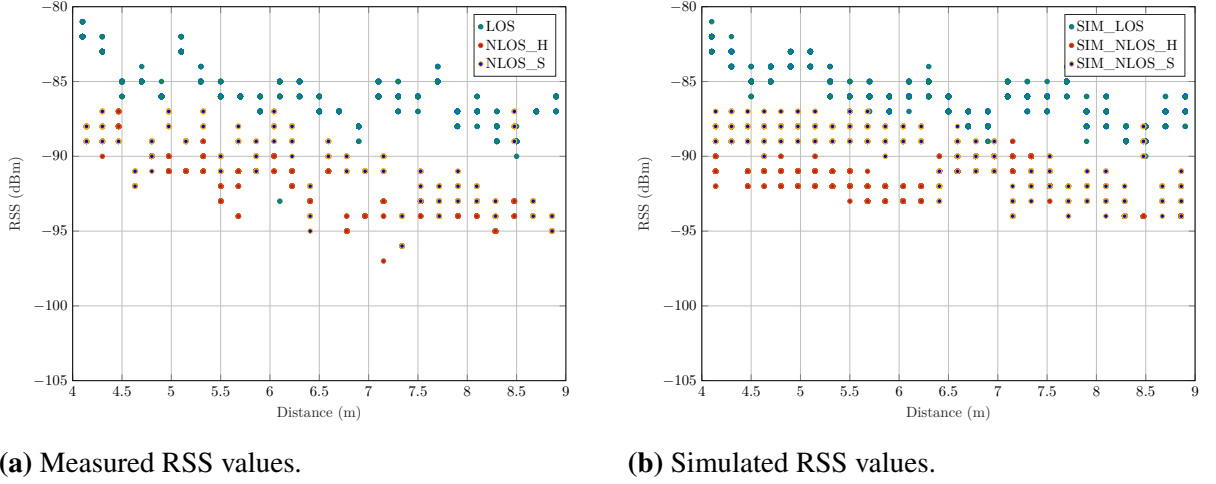


Figure 6.5: Simulated RSS values corresponding to the measurement scenario and comparison with the measured RSS values.

an external battery to power all the devices. The Pozyx tag was placed at some extra height over the forklift roof with a twofold objective: to avoid electromagnetic interferences from the rest of the equipment and also to improve the visibility with the anchors. The PX4 Flow was placed on an arm on the side of the vehicle, hence the camera had full view of the ground without obstacles.

The forklift movement model was defined taken into account that the forklift is allowed to:

- move only in two dimensions (on a plane over the ground).
- move forward and backward.
- move tracing a curve or in a straight line.
- spin around its Z axis without performing a displacement in X and Y axes.

Based on this model, the different elements of the IEKF were defined. Thus, as only the two-dimensional location of the forklift was relevant, the *state* of the filter was defined as

$$\mathbf{x}_k = (x, y, v_x, v_y, a_x, a_y, \theta, \omega)^T, \quad (6.1)$$

where x and y denoted the forklift position; v_x, v_y its velocity; a_x, a_y its acceleration; θ was its heading angle, and ω corresponded to its angular velocity.

For its part, the *transition equation* was

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{n}_k, \quad (6.2)$$

where \mathbf{x}_k was the estimation of the forklift state, $\mathbf{n}_k \sim \mathcal{N}(0, \mathbf{C}_n)$ was a random variable modeling the prediction error, and \mathbf{F} was the *transition matrix* defining the relationships between the state variables.

To define the *observation equation* the diagram shown in Fig. 6.8 was used as reference. In this diagram it can be seen how the Pozyx tag (A) was placed in the center of the forklift roof,

6. Simulation of Ultra Wideband based location devices

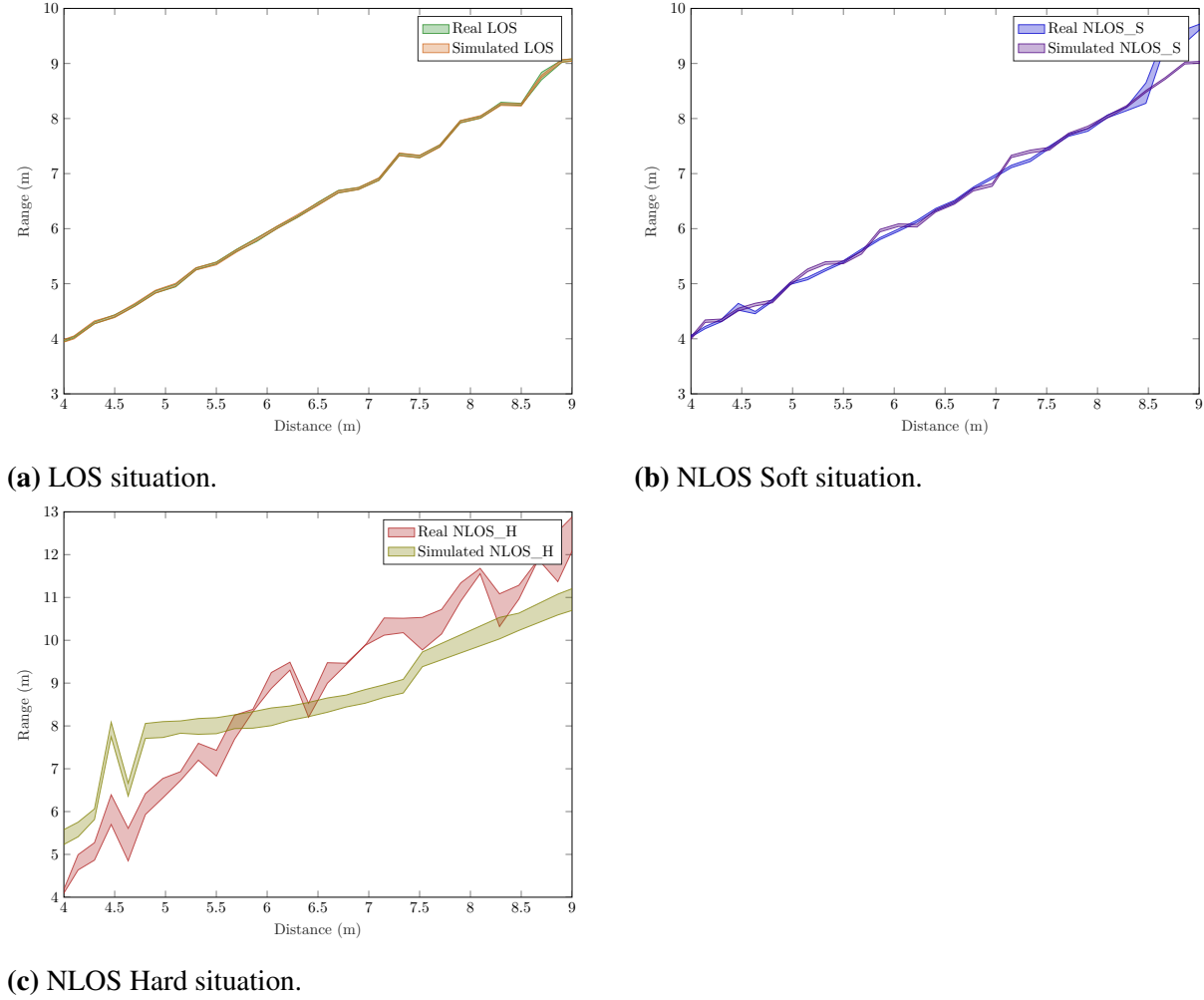


Figure 6.6: Mean and standard deviation (mean $\pm\sigma$) of the measured and simulated ranging values shown in Fig. 6.4 for the three different situations considered.

whereas the PX4 Flow was located on the forklift side (B), on a sort of arm that put the sensor outside the forklift. Taking this into account, the *observation equation* was defined as:

$$\mathbf{y}_{k,j} = H_j(\mathbf{x}_k) + \mathbf{m}_{k,j}, \quad (6.3)$$

where $H_j \in \{H_R, H_F, H_E\}$ represented the observation function for the ranging, flow, and accelerometers, respectively; $\mathbf{m}_{k,j} \in \{m_{k,R}, m_{k,F}, m_{k,E}\}$ corresponded to the observation noise modeled as Gaussian-distributed variables with a variance specific for each sensor type, i.e., $\mathbf{m}_{k,R} \sim \mathcal{N}(0, \mathbf{C}_{m,R})$, $\mathbf{m}_{k,F} \sim \mathcal{N}(0, \mathbf{C}_{m,F})$, and $\mathbf{m}_{k,E} \sim \mathcal{N}(0, \mathbf{C}_{m,E})$; and $\mathbf{y}_{k,j} \in \{\mathbf{y}_{k,R}, \mathbf{y}_{k,F}, \mathbf{y}_{k,E}\}$ were the corresponding observations.

Therefore, the corresponding observation matrices were defined as follows:

$$H_R(\mathbf{p}) = (\|\mathbf{p} - \mathbf{b}_1\|, \dots, \|\mathbf{p} - \mathbf{b}_L\|)^T \quad (6.4)$$

was the observation matrix for the Pozyx sensor where $\mathbf{p} = (x, y, \bar{z})$ was the location of the vehicle, with \bar{z} being the height of the tag, and \mathbf{b}_l being a vector with the coordinates of the l -th

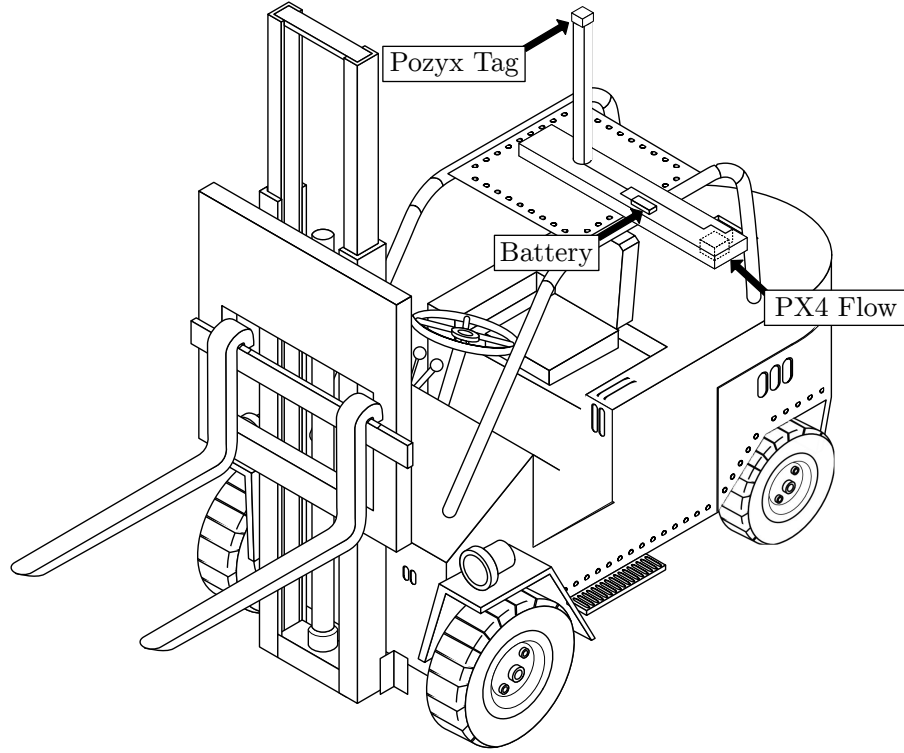


Figure 6.7: Sensor placement on the forklift.

anchor.

$$H_F(\mathbf{v}) = \begin{pmatrix} R(\theta)\mathbf{v} + \frac{1}{\Delta t}(\mathbf{I} - R(\theta))\mathbf{d} \\ \omega \end{pmatrix} \quad (6.5)$$

was the observation matrix for the PX4 Flow sensor where $\mathbf{v} = (v_x, v_y)$ was the velocity vector, \mathbf{d} was the vector with the location of the flow sensor with respect to the center of the forklift (see Fig. 6.8), ω was the angular velocity, and $R(\theta)$ was a standard rotation matrix defined as

$$R(\theta) = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}. \quad (6.6)$$

Finally, the observation matrix for the accelerometer sensor was

$$H_E(\mathbf{a}) = \begin{pmatrix} R(\theta)\mathbf{a} \\ \theta \\ \omega \end{pmatrix}, \quad (6.7)$$

where $\mathbf{a} = (a_x, a_y)$ was the acceleration vector, θ was the heading angle, ω was the angular velocity, and $R(\theta)$ was the standard rotation matrix defined in Eq. (6.6).

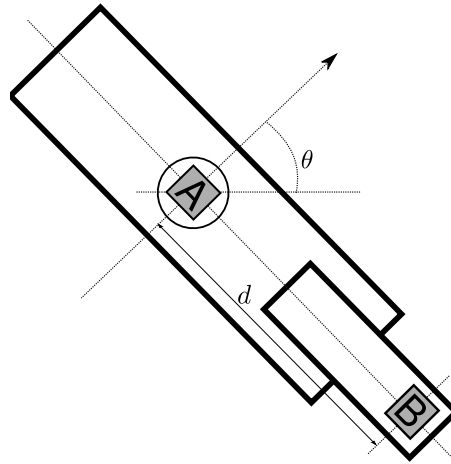


Figure 6.8: Sensor placement in detail. A: UWB tag, B: PX4 Flow.

6.6 Software Implementation

The location algorithm detailed in Section 6.5 as well as other software pieces needed to capture and analyze the data, were implemented in ROS ([ROS19c]), which is a widespread software environment employed in robotics-related projects due to its functionalities and helpers. It is specifically designed to interconnect different processes using a message passing mechanism, thus each process can operate in an isolated way using publishers and subscribers to communicate with others.

Nodes and messages are the basic ROS elements. Nodes are software pieces that perform some processing and, in most of the cases, publish some result of interest, which are packed in the form of messages. A message is an unsorted set of fields with different identifiers and data types. A ROS node can define custom messages or use one of the generic ones offered by the ROS core. Once the result is modeled with some message type, the node publishes it under a topic identifier. Hence, several nodes can publish the same message type in different topics with distinct contents.

Besides publishing results, a ROS node can subscribe to a topic from another node and receive its data. ROS has some tools to show all the published topics in the system, their data type, and publisher settings, thus a node can find and subscribe to the desired sources. Additionally, there are several mechanisms inside ROS to record and replay the set of messages produced by different publishers, which is very useful to test and improve algorithms.

The ROS nodes and messages developed for this work are publicly available in [BAR19j; BAR19i; BAR19h; BAR19g; BAR19f] and they are detailed below.

6.6.1 ROS Nodes

The following ROS nodes were developed:

- The UWB plug-in (*gtec_uwb_plugin*, see Table 6.1) publishes in the ROS ecosystem the

Table 6.1: Plug-in *gtec_uwb_plugin*. The *id* in the topic route corresponds to the tag identifier.

Topic	Message type	Topic type
/gtec/gazebo/uwb/ranging/id	gtec_msgs::Ranging	Published
/gtec/gazebo/uwb/anchors/id	visualization_msgs::MarkerArray	Published

Table 6.2: Plug-in *gtec_tag_pos_publisher*.

Topic	Message type	Topic type
/gtec/gazebo/pos	geometry_msgs::PoseWithCovarianceStamped	Published

Table 6.3: Node *gazebo2ros*.

Topic	Message type	Topic type
/gtec/gazebo/imu	sensor_msgs::Imu	Published
/gtec/gazebo/PX4 Flow	mavros_msgs::OpticalFlowRad	Published

Table 6.4: Node *kfpos*. The *id* in the UWB range topic route corresponds to the tag identifier.

Topic	Message type	Topic type
/gtec/gazebo/uwb/ranging/id	gtec_msgs:: Ranging	Subscription
/gtec/gazebo/PX4 Flow	mavros_msgs::OpticalFlowRad	Subscription
/gtec/gazebo/imu	sensor_msgs:: Imu	Subscription
/gtec/kfpos	geometry_msgs:: PoseWithCovarianceStamped	Published

messages corresponding to the position of the anchors in the scenario and the ranging values with respect to the tag associated to the plug-in.

- The *gtec_tag_pos_publisher_plugin* plug-in (see Table 6.2) publishes the true position of the object to which it is attached. Such position values consists of the x , y , and z coordinates together with the quaternion and they are used to plot the actual trajectories followed by the object.
- The *gazebo2ros* node (see Table 6.3) acts as relay to capture the measurements of different sensors in Gazebo and publish them through ROS. More specifically, in this work it is used to publish the data from the IMU and the PX4 Flow sensors.
- The *kfpos* node (see Table 6.4) contains the implementation of the positioning algorithm described in Section 6.5, it has subscriptions to several topics (to receive data from the different sensors), and publishes the position estimation on another topic. In this specific implementation of the IEKF, the following parameters were used:
 - Maximum number of iterations: 50.
 - Minimum cost goal: 1×10^{-4} .
 - Maximum Jolt: 0.1 m/s^3 .

Additionally, the following external ROS nodes are also used for different purposes:

- The *gazebo2rviz* node [ROS19a] plots in the application *RVIZ* the position of the static elements (e.g., walls) placed in the Gazebo simulation.
- The *turtlebot3_gazebo* node [ROS19e] is used as a base to model the forklift vehicle.
- The *teleop_twist_joy* node [ROS19d] modifies the angular and linear speed of the vehicle through a USB joystick to easily move the simulated forklift across the different scenarios. To reproduce the routes, the messages of this node are recorded in a log that can be replayed later on.
- The *mavros_extras* node [ROS19b], which includes the definition of the message type *OpticalFlowRad* used to represent the output of the PX4 Flow sensor.

All the software developed for the simulations is publicly available in [BAR19d; BAR19c; BAR19b].

6.6.2 ROS Custom Messages

Several message types are used to model the information. Some of them are provided by the ROS core, others by the Mavros ROS package, and finally, some of them were developed ad-hoc for this project. All the custom messages were created inside a package named *gtec_msgs*.

The message *gtec_msgs::Ranging* models a generic range value between a tag and an anchor and has the following fields:

- **anchorId**: anchor identifier.
- **tagId**: tag identifier.
- **range**: distance value expressed in millimeters.
- **seq**: sequence number. Several ranging measurements can have the same sequence number if they were created at the same time.
- **errorEstimation**: error estimate within the provided range value.
- **rss**: received power estimate.

6.7 Results

The simulation process began with the recording of two routes corresponding to both simulated scenarios. The *teleop_twist_joy* node [ROS19d] and a USB joystick were used, so that by moving the sticks and pressing the buttons the forklift could be guided between two points performing different movements. All the angular and linear speed change commands were recorded in a log, allowing for repeating the routes when needed.

Two additional pieces of software were developed to achieve the most accurate results: a script to re-start Gazebo simulations and record the sensor's outputs, and another different script to play a recorded log while a new instance of the location algorithm is executed. Thus, the simulation phase was split into two steps. The first step consists in replaying the routes 100

times and record the sensor data.

After this first step, several ROS logs were recorded for each route. The second step used these logs to feed the location algorithm and to obtain the associated location estimates. The script launched each time three different configurations of the location algorithm: one using only the UWB ranging values, another using the UWB ranging values and the IMU values, and one last configuration considering all the sensors: UWB, IMU and PX4 Flow. Finally, the estimates outputted for each configuration were averaged and stored for later analysis.

Fig. 6.9 shows different lines highlighting the path followed by the forklift in the Scenario A for each of the configurations considered for the location algorithm: using only data from the UWB sensor (dark red line), including UWB and the IMU (yellow line), and taking all the three sensors into account (purple line). The true position of the vehicle corresponds to the green line. Fig. 6.9 shows that, when only UWB is employed, there are areas in the scenario where the error is large due to the obstacles and morphology of Scenario A (see Fig. 6.1a). Such obstacles, depending on the tag position, yield one of the following three possibilities:

1. NLOS *Soft* situation: the direct path can be decoded and hence the positioning algorithm provides estimates close to the true values.
2. There is no link between the anchor and the tag, neither the direct path nor a rebound. In this situation no ranging value is generated from that anchor and therefore has no influence on the algorithm.
3. NLOS-*Hard* situation: the direct path is blocked, but a signal rebound enables a wireless link between the anchor and the tag, although severely degrading the provided ranging estimate, hence strongly penalizing the localization algorithm.

Notice that, in the experiments carried out in this work, no strategy was adopted to detect and mitigate NLOS-*Hard* situations.

Fig. 6.10 shows the mean absolute error (MAE) corresponding to each of the three considered configurations of the positioning algorithm together with its 95 % confidence interval. When only UWB data were considered, the average error exceeded 1 m and its variance was very large. Such high error values were due to the ranging values estimated under a NLOS *Hard* situation. Adding the inertial sensor reduced the error considerably, although its average was still above 0.8 m. Finally, adding the PX4 flow led to the best strategy to reduce the error, which fell below 0.2 m. Fig. 6.11 shows the empirical cumulative distribution function (ECDF) of the position error in the first scenario. Clearly, when the ranging data included values from a NLOS *Hard* situation (and those values were not filtered out or mitigated), adding more sensors contributed to improve the behavior of the positioning algorithm. Especially notable was the improvement using the optical sensor, whose values were robust enough to keep the estimate very close to the actual position. It should be remembered that, as mentioned above, the behavior of this sensor is very dependent on the type of floor existing in the scenario, and that in the experiments carried out during this work a high quality texture was used so that its results were optimal.

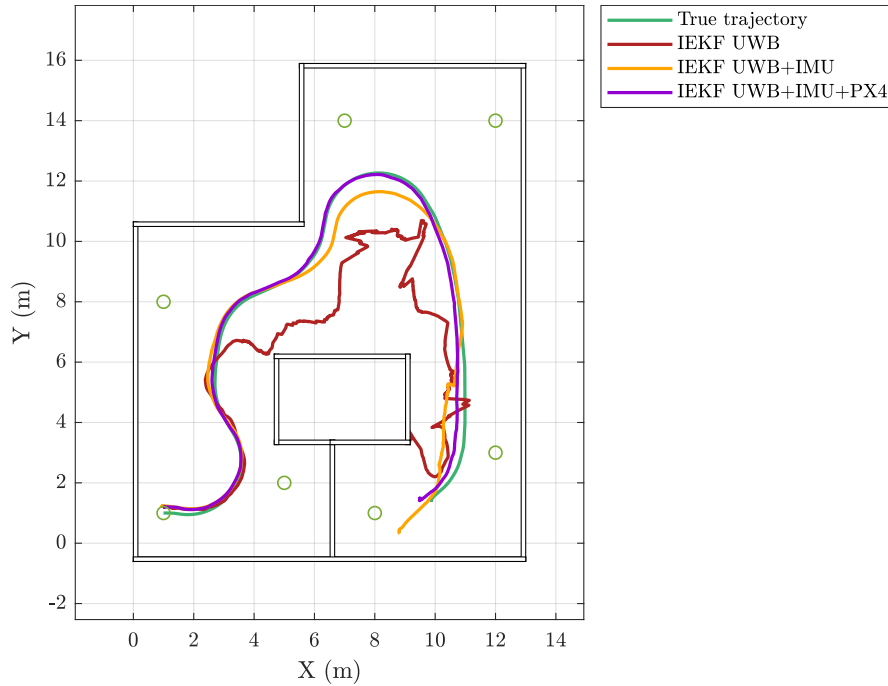


Figure 6.9: Trajectories using different sensors in Scenario A.

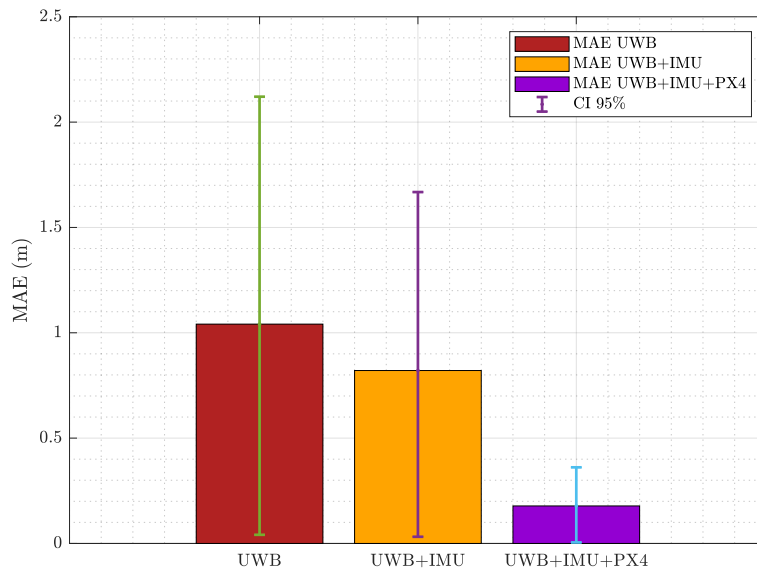


Figure 6.10: MAE of position estimation in Scenario A.

Fig. 6.12 shows the vehicle trajectories in scenario B. In this case, it can be seen how the three configurations of the positioning algorithm adopt a similar behavior. This is because in this deliberately chosen scenario, all UWB anchors were always in a LOS situation with the tag. In this way, all the ranging values generated by the simulator were very close to the real values. Consequently, the positioning algorithm worked throughout the route with information consistent with the physical reality of the vehicle and the status of its sensors.

Finally, Fig. 6.13 shows the MAE of the different configurations in Scenario B. It can be seen

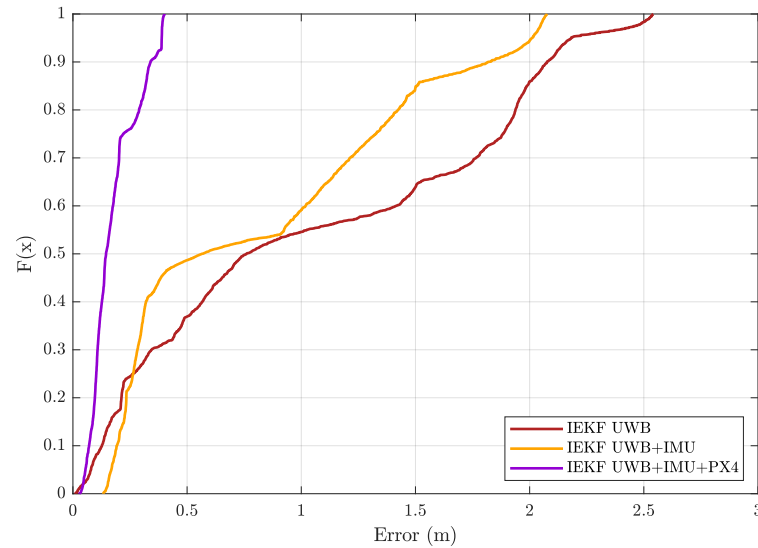


Figure 6.11: ECDF of position error in Scenario A.

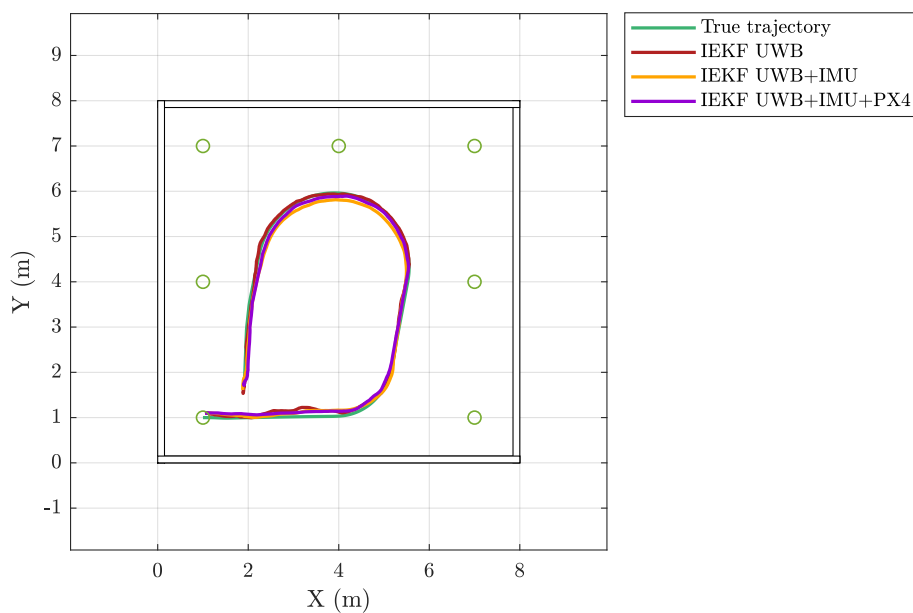


Figure 6.12: Trajectories using different sensors in Scenario B.

how the error values were very low, below 0.1 m even when using only the UWB sensor. Such a value corresponds exactly to the margin offered by Decawave, the manufacturer of the DW1000 base chip of the Pozyx devices, which writes in its product description *"This chip enables you to develop cost-effective RTLS solutions with precise indoor and outdoor positioning to within 10 cm."* [DEC19c]. Another important aspect of the data presented in Fig. 6.13 is that all the configurations of the positioning algorithm performed similarly (less than 0.02 m difference between them) regardless of the number of sensors used. This indicates that probably, with the noise levels modelled on these sensors, it is not possible to significantly reduce the (already

6. Simulation of Ultra Wideband based location devices

very small) error observed when using UWB measurements. Fig. 6.14 shows the ECDF of the position error in the same scenario. In this figure it can be seen how, although the three configurations gave a similar value of MAE, the configuration with three sensors managed to reduce the maximum error up to 13 cm, slightly below the 22 cm obtained using only UWB. It must be said, however, that with the proposed model it was not possible to infer the orientation of the vehicle using only the ranging values from a single UWB tag. For this purpose, it would be necessary to include one of the other two sensors described, either the inertial sensor or the PX4 Flow.

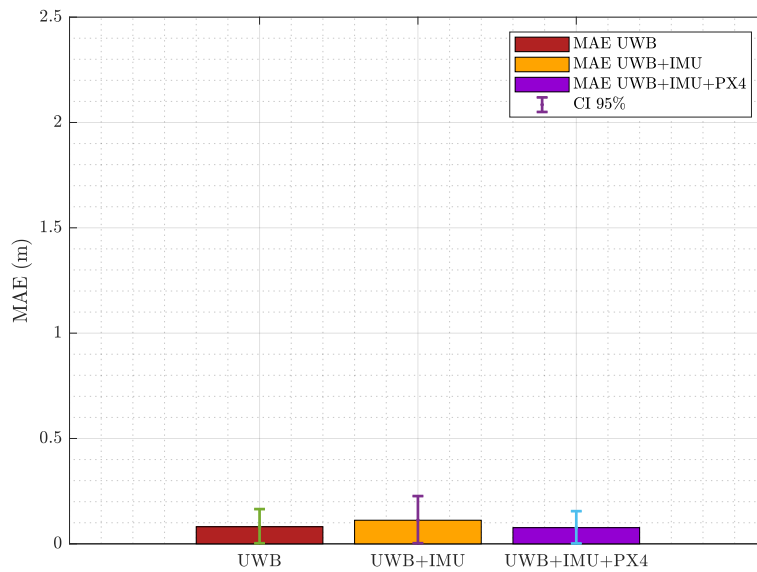


Figure 6.13: MAE of position estimates in Scenario B.

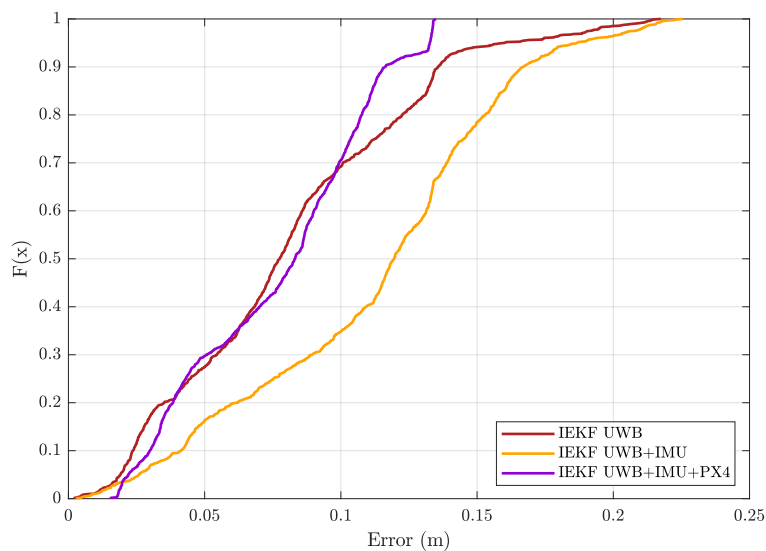


Figure 6.14: ECDF of position error in Scenario B.

6.8 Simulation vs real world

In order to verify if the values provided by the UWB simulator were closer enough to the real measurements in a real case, the experiments described in Section 5.4 were repeated but this time in a simulated environment.

For this purpose, the measurements of the walls and distances between them in the test scenario described in Section 5.1.2 were taken, and with these data a 3D model of the scenario with the same characteristics was built in Gazebo (although without a roof, as the UWB simulator does not take it into account for its calculations). Five UWB anchors were placed in the same positions as in the real case described in Section 5.1.2 and that can be observed in Fig. 5.2. The final model of the scenario can be seen in Fig. 6.15.

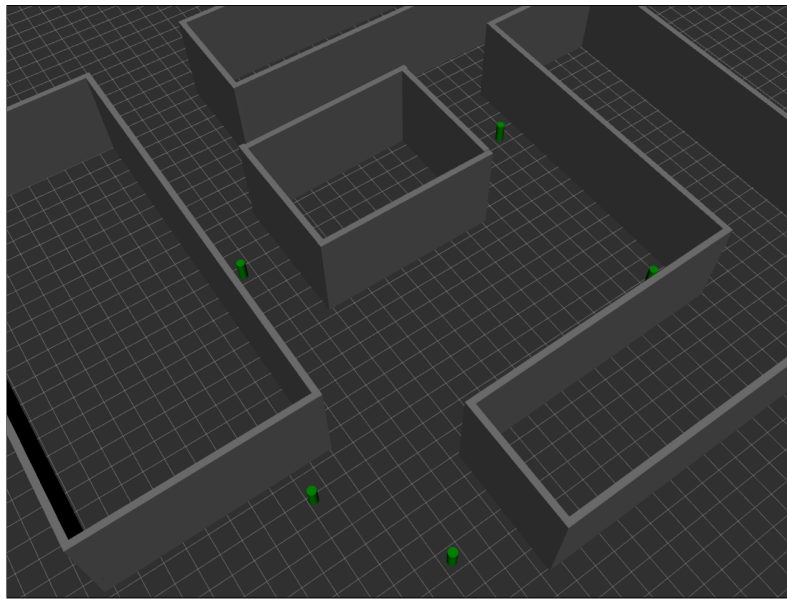


Figure 6.15: Grid simulation scenario

Once the simulation scenario was built, measurements were taken at the same points where they had been taken during the experiment in the real environment. Thus, the simulated UWB tag was placed on the points of a 3×3 grid and measurements were captured during 60 s at each point, the same capture time as in the real case.

After obtaining these simulation measurements, the experiment followed the same process as in the case with real measurements. The same NLOS classifiers and mitigators as in that occasion (based on k-nearest neighbors (k-NN), Gaussian process (GP) and NN) were used to filter the simulated measurements, and finally the results fed the previously implemented positioning algorithms: the one based on nonlinear least squares (NLS) and the IEKF.

Figs. 6.16 and 6.17 show the error values obtained for each of the analyzed configurations (the same that can be seen in Figs. 5.4 and 5.5 for the real case) when using the IEKF. Figs. 6.18 and 6.19 shows the results when using the NLS based algorithm, and the values can be compared with the one previously obtained for the real case in Figs. 5.6 and 5.7.

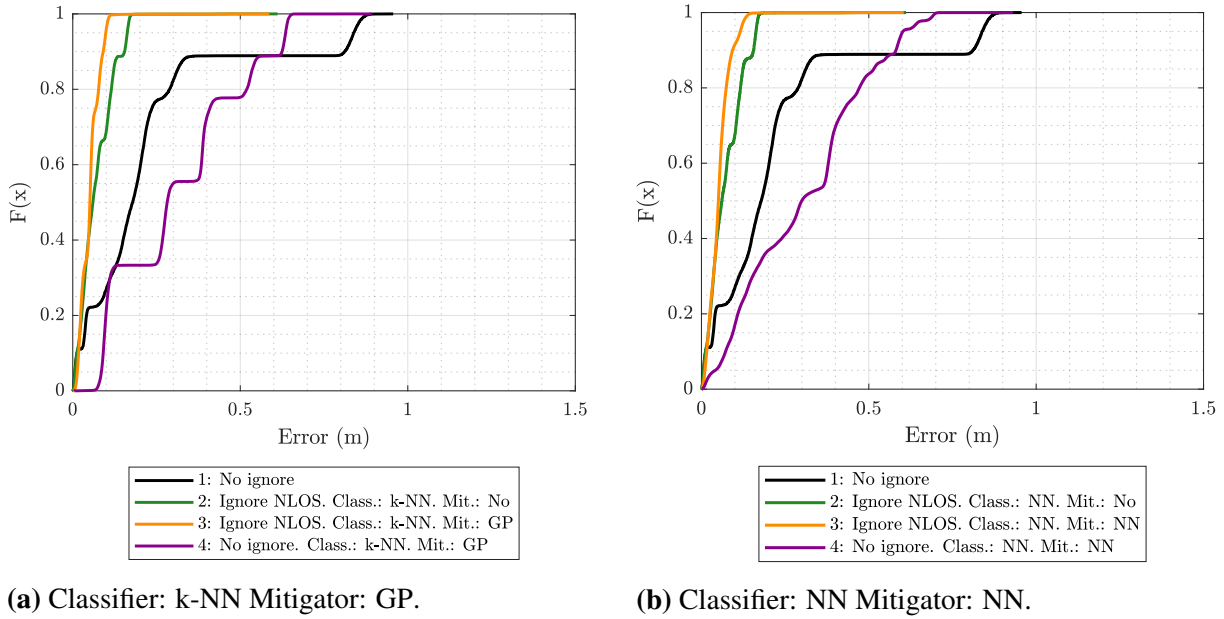


Figure 6.16: ECDF of the location error using IEKF.

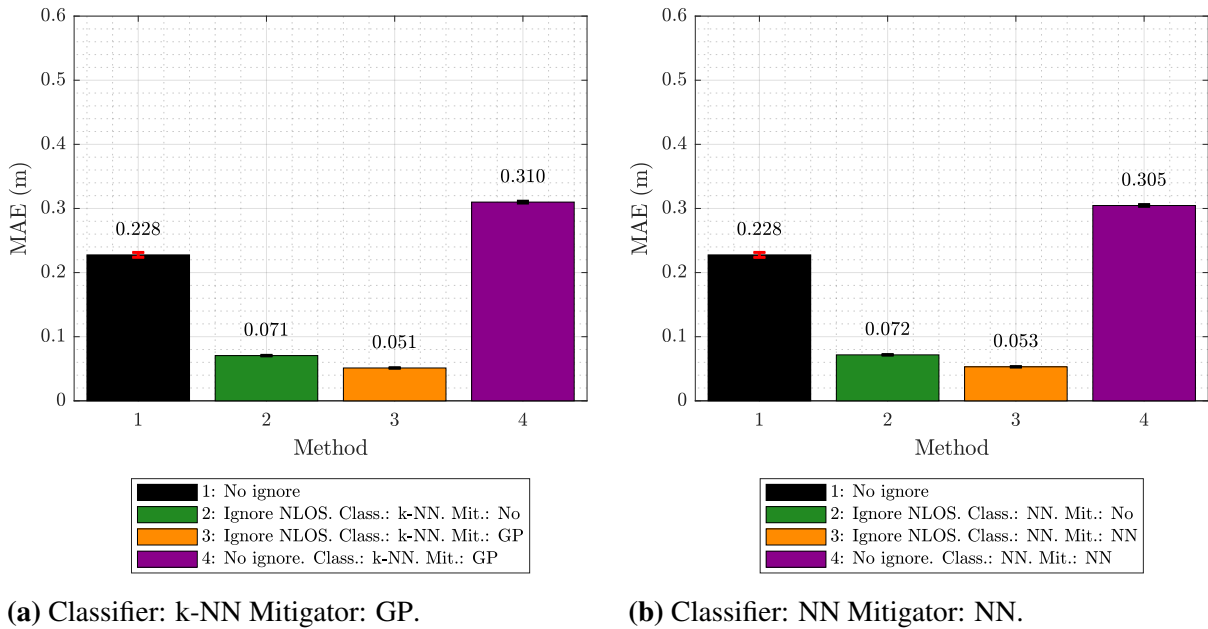
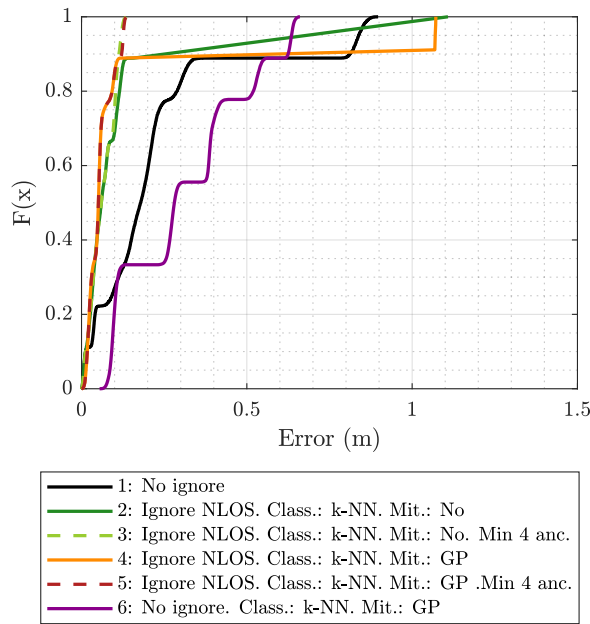
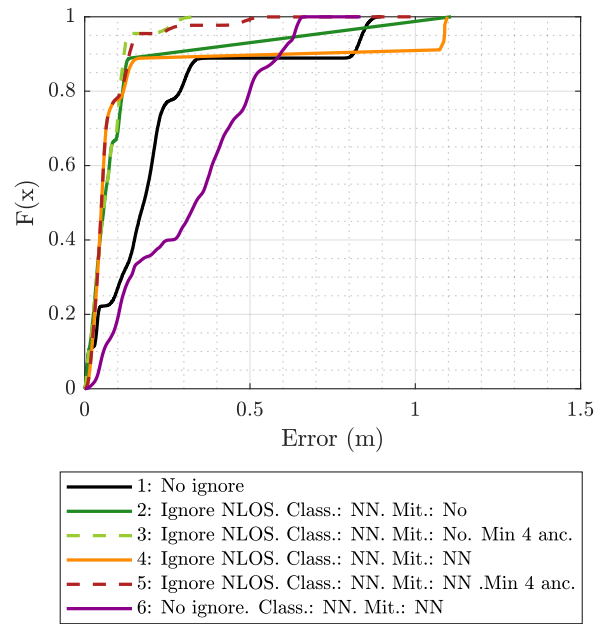


Figure 6.17: Localization MAE using IEKF.

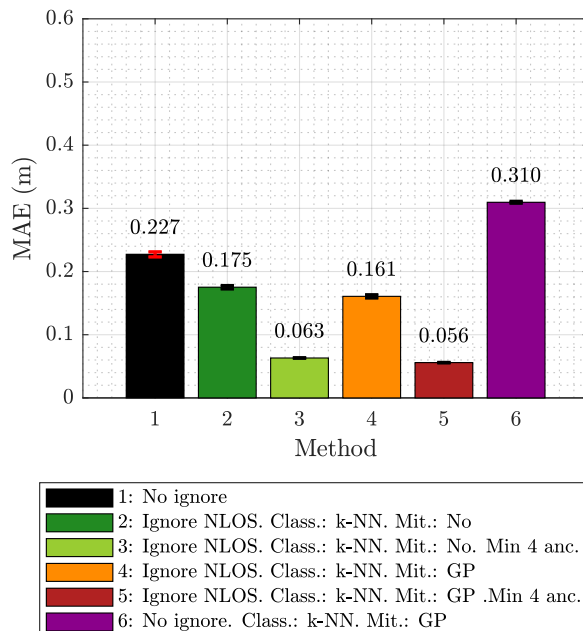


(a) Classifier: k-NN Mitigator: GP.

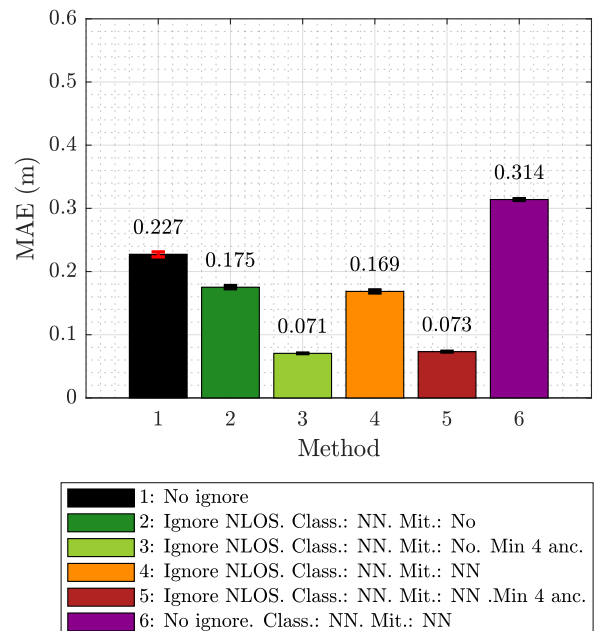


(b) Classifier: NN Mitigator: NN.

Figure 6.18: ECDF of the location error using NLS



(a) Classifier: k-NN Mitigator: GP.



(b) Classifier: NN Mitigator: NN.

Figure 6.19: Localization MAE using NLS.

6. Simulation of Ultra Wideband based location devices

IEKF	Real	Simulated	Diff
No Ignore	0.381	0.228	0.153
Ignore NLOS. Class.: k-NN. Mit.: No.	0.084	0.071	0.013
Ignore NLOS. Class.: k-NN. Mit.: GP.	0.136	0.051	0.085
No Ignore. Class.: k-NN. Mit.: GP.	0.411	0.310	0.101
Ignore NLOS. Class.: NN. Mit.: No.	0.086	0.072	0.014
Ignore NLOS. Class.: NN. Mit.: NN.	0.180	0.053	0.127
No Ignore. Class.: NN. Mit.: NN.	0.427	0.305	0.122

Table 6.5: Real vs Simulated MAE comparison using IEKF.

NLS	Real	Simulated	Diff
No Ignore	0.379	0.227	0.152
Ignore NLOS. Class.: k-NN Mit.: No.	0.778	0.175	0.603
Ignore NLOS. Class.: k-NN Mit.: No. Min 4.	0.155	0.063	0.092
Ignore NLOS. Class.: k-NN Mit.: GP.	0.789	0.161	0.628
Ignore NLOS. Class.: k-NN Mit.: GP. Min 4.	0.330	0.056	0.274
No Ignore. Class.: k-NN Mit.: GP.	0.412	0.310	0.102
Ignore NLOS. Class.: NN Mit.: No.	0.778	0.175	0.603
Ignore NLOS. Class.: NN Mit.: No. Min 4.	0.142	0.071	0.071
Ignore NLOS. Class.: NN Mit.: NN.	0.828	0.169	0.659
Ignore NLOS. Class.: NN Mit.: NN. Min 4.	0.259	0.073	0.186
No Ignore. Class.: NN Mit.: NN.	0.428	0.314	0.114

Table 6.6: Real vs Simulated MAE comparison using NLS.

To easier compare the results obtained in both situations (with real and simulated measurements), Table 6.5 and Table 6.6 show a summary of the MAE for each case.

A number of conclusions can be drawn from these data. It can be observed how in general the values obtained by the simulator provided lower error values than the real scenario. A particularly important case is the configuration *No ignore*, since it analyzes the positioning error without using any technique to try to eliminate the effect of NLOS. It can be seen how for both location algorithms, IEKF and NLS, the deviation with respect to the real error was practically the same, about 15 cm.

It is also important to check the data in the other cases, as they give us a hint of how similar was the NLOS set of measurements generated in both sets. Thus, for the IEKF case it can be seen how the differences in the MAE with respect to the data obtained with the real measurements were quite small, always being below 13 cm and reaching less than 2 cm for the configuration in which the measurements classified as NLOS were ignored (both with k-NN and with NN).

An important difference with respect to the absolute values obtained is that for the simulated case the configuration that applied mitigation after ignoring the NLOS measurements obtained a better result than the configuration in which only the NLOS measurements were eliminated and no mitigation was carried out. As already explained in the analysis of the real case results (Section 5.5), this behavior should be expected if the training set covered the entire sample space of the test scenario. In the experiment with real measurements there were certain measurements in the test scenario that were not represented in the training set (as they were obtained in the other different scenario) and hence the mitigation did not improve the results. With the simulation this was different, since the simulated values came from a model based on the *ranging* and RSS data obtained in the training scenario. Therefore, in the simulation, both the classifiers and the mitigators were trained with similar measurements to those used to create the simulator, so their result should be much better (as in fact can be seen in the MAE data).

With respect to the MAE data obtained using the NLS algorithm (Table 6.6), the major differences were founded in the configurations where the NLOS measurements were eliminated and mitigation was applied or not, but without reserving a minimum number of 4 anchors in each iteration of the algorithm. In the real case, these configurations had given a very high error level, above 75 cm of error, mainly because too many samples were eliminated and the algorithm could not generate new positions. In the simulation it can be seen how, although the same relation was maintained with respect to the version that reserves a minimum of 4 anchors (the latter reduce the error), the absolute value of the error committed in the simulation was much lower (standing around 17 cm). This could be explained in part by the greater efficiency of the classifier in the simulation, which would lead to eliminating fewer erroneous samples and would cause the algorithm to remain in fewer occasions without the minimum number of values required to perform the calculations.

6.9 Conclusions and future lines

Simulation is an essential part of any real localization solution. The cost of deployments and pilots means that in many cases it is not possible to make adjustments or tests in a real environment. In this case, having a realistic simulation environment provides a clear benefit in order to improve the planning or construction of the system.

In this chapter it was described the simulation of a common problem in the industry, such as the location of load pallets. In the proposed approach, it was chosen to locate instead the forklifts that transport them, so that the costs would be lower.

For the proposed model, different sensors were chosen to be added to the vehicle, including UWB ones. However, since there was no specific simulation software for this type of technology, at least at a high level and oriented to localization, a custom software was developed from scratch. This simulator, integrated within the Gazebo simulation platform, was built based on a set of measurements taken from real UWB sensors. These measurements were used to train

and a series of NN models that modeled each of the situations contemplated: LOS, NLOS *Soft*, and NLOS *Hard*. The simulator contemplated the number of walls and their thickness between the tag and the anchor to estimate if one model or another should be applied.

The results of the simulations carried out revealed that the accuracy in the location using only UWB measurements is acceptable only when there is a strong LOS between the tag and each of the anchors. If this is not met and the tag receives signal bounces instead of the direct path between the two devices, then the accuracy in the location declines dramatically. The results also showed how the use of additional sensors considerably reduce the error in situations where UWB data alone are not good enough.

Finally, it has also been shown how some tests carried out in a real environment have a great similarity with the results obtained from the simulator with the UWB plugin in the same scenario created in a synthetic way.

6.10 Contributions related with the chapter

The content of most of this chapter can be found in the following journal article:

- Valentín Barral, Pedro Suárez-Casal, Carlos J. Escudero, and José A. García-Naya. “**Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments**”. *Electronics*, vol. 8, no. 10, 2019.
DOI: 10.3390/electronics8101152. Online access: <https://www.mdpi.com/2079-9292/8/10/1152>

The dataset with the measurements captured during this work are publicly available in the following reference:

- Valentín Barral. **Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments - dataset**. 2019.
DOI: 10.21227/b9tf-fv74. Online access: <http://dx.doi.org/10.21227/b9tf-fv74>

All the ROS nodes and the UWB plugin, as well as the configurations files, worlds definitions and building models were published as open source software:

- Valentin Barral. **Gazebo sensors plugins source code repository**. Available online: <https://github.com/valentinbarral/gazebosensorplugins>. Accessed Oct. 2019

- Valentin Barral. **Gazebo forklift simulation source code repository**. Available online: <https://github.com/valentinbarral/gazeboforkliftsimulation>. Accessed Oct. 2019
- Valentin Barral. **Gazebo 2 ros source code repository**. Available online: <https://github.com/valentinbarral/gazebo2ros>. Accessed Oct. 2019
- Valentin Barral. **Ros uwb reader source code repository**. Available online: <https://github.com/valentinbarral/rosuwbreader>. Accessed Oct. 2019
- Valentin Barral. **Ros toa source code repository**. Available online: <https://github.com/valentinbarral/rostoa>. Accessed Oct. 2019
- Valentin Barral. **Ros msgs source code repository**. Available online: <https://github.com/valentinbarral/rosmsgs>. Accessed Oct. 2019
- Valentin Barral. **Ros logs scripts source code repository**. Available online: <https://github.com/valentinbarral/roslogscripts>. Accessed Oct. 2019

Chapter VII

Multi-technology Real Time Location System

An real-time location system (RTLS) is a mechanism that encapsulates all the logic of the localization process and abstracts it so that it can be used by other elements within an organization or application. These elements are the location based services (LBS). A LBS is a transformation mechanism that is capable of providing information of some kind based on available location data. Its definition arises at the same time as the first positioning estimates, and its evolution goes hand in hand with these. Nowadays, with the presence of more and more precise indoor positioning technologies, LBS have gained in popularity and are now entities of habitual use both by people in their daily lives and within the operations of companies. Examples of LBS based on indoor positioning data could be named for example:

- Tracking: Tracking services of people, vehicle or machinery are common in many companies. Both for route optimization and inventory work, this type of service is one of the most implemented in real applications.
- Navigation: Navigation services are very common today for both home users and professionals. Simply by using a smartphone you can access different navigation LBS that allow you to obtain routes and plan trips. Indoor use is becoming increasingly important, especially in large structures such as factories or airports where there is some kind of location technology capable of providing data with sufficient accuracy.
- Access control: one of the most obvious services, automated access control allows users to dynamically define the use they can make of the available space.
- Occupational safety: increasingly indispensable in large factories, LBS aimed at detecting potential situations of risk for workers are one of the most attention attracting. They are one of the pillars of the so-called industry 4.0 [JAR+17].
- Data analysis: the analysis of location data is one of the major sources of information available to many companies today, especially those whose business model is based on user behavior. Shopping malls, hospitals or any other building with a large model of people on the move is likely to provide information of great interest that a data analysis

LBS may be able to process and transform into action policies determined.

- Climate control: One of the pillars of so-called intelligent buildings, climate control based on the movement patterns of users and their relationship with energy suppliers is one of the most important types of LBS.

Of course, there are multiple scenarios where particular LBS can be defined. LBS focused on sport, industry, health, city management, traffic management, etc. The proposals are countless. In many occasions the definition of LBS is tied to a specific RTLS platform. Platforms such as Localino [LOC19], IndoorAtlas [IND19a], Anyplace [ANY19], Quuppa [QUU19] or Google Maps [IND19b] offer solutions where users can use certain LBS or build others on a specific location based technology

Localino offers an RTLS platform with a series of LBS defined on hardware they themselves provide. The hardware used is based on the use of tags that combine WiFi and UWB technology. Its software platform allows managing the different devices, both from the point of view of location (placement of anchor nodes) and maintenance (battery level). In addition to this, the platform also offers an interface on which to define events that are triggered in case certain conditions are met in the location data. Other functionalities of the platform are also those of being able to visualize the location data in real time and the one of being able to analyze the obtained data. Localino is aimed at companies, retailers and sports.

IndoorAtlas is a platform that operates with indoor location data from different technologies such as WiFi, BLE, presence detectors or inertial sensors. It offers an API through which new LBS can be defined or access to a series of them previously defined, such as navigation or search for nearby points of interest. IndoorAtlas solution is oriented to health care, retail and transportation scenarios.

Anyplace is an open source platform that uses WiFi fingerprint and inertial sensors as the basis for location. It has a REST API for developers, and some LBS for navigation and log. It also has a web viewer that allows to visualize buildings and points of interest on a mobile, in addition to an Android application with the same purpose.

Quuppa is another RTLS platform that incorporates its own hardware, in this case based on BLE. At software level, Quuppa offers a REST API for developers so they can access to the location data and also perform log tasks and replay trajectories.

Indoor Google Maps is a solution that allows users to upload their own indoor maps for integration into Google Maps. Once the plans are reviewed by Google, this offers different LBS on them, such as creating routes, searching for points of interest, etc.. Indoor location is achieved in this case by WiFi fingerprint.

All the previous solutions, however, have an important limitation: they are very dependent on a specific technology or subset of them. This strong link between the logic of the platform and the hardware causes that they have an important lack of flexibility and scalability. Authors in [HBB02] were one of the first groups who introduced the idea of a layered architecture for LBS. The seven layers tried to provide a stack for any kind of location-based service or

application.

In a more general way [SV04] described the LBS by means of a three layer model: positioning, middleware and application layer. The application layer determines the data usage for specific location applications, based on the data extracted from the middleware. The middleware provides the interfaces for the application layer, hiding technical details from the positioning layer. Finally, the positioning layer deals with the deployment, configuration and calibration of wireless location infrastructure; gathers raw data (radio signal strength, time of arrival, angle of arrival...); and calculates location data using location algorithms such as proximity, ranging, triangulation, or signal strength maps. RTLS platforms, like the one introduced in the paper, cover the positioning and middleware layers of an LBS.

Once it is clear that an architecture model for RTLS was necessary, the International Organization for Standardization (ISO) defined two standards: ISO24730-1:2006 [ISO19b] and ISO19762-5:2008 [ISO19a]. The first defines an application programming interface (API) describing an RTLS service and its access methods, to enable client applications to interface with the RTLS, whilst the second provides a harmonized vocabulary with terms and definitions unique to locating systems in the area of automatic identification and data capture techniques. This glossary of terms enables communication between non-specialist users and specialists in locating systems through a common understanding of basic and advanced concepts.

In order to create a flexible system, capable of operating with any location technology and open to support any type of LBS, an RTLS platform was designed and implemented. The most important details of this RTLS platform are detailed in Section 7.1, where its main features, architecture and actors are described.

The RTLS platform presented here has been used in different national projects in recent years. The experiences in each of these deployments can be read in Section 7.2. Parallel to these deployments and sometimes thanks to them, a series of possible improvements were detected to give the RTLS platform greater versatility. A summary of the improvements that were added can be seen in Section 7.3.

Finally, Section 7.4 shows the conclusions drawn after the different experiences with the platform in real implementations.

7.1 Multi-technology RTLS platform

7.1.1 Platform objectives

Trying to overcome the limitations observed in other systems, the proposed RTLS platform emerged with the following objectives:

- **Technology Independent:** it must support inputs from any kind of measurement (RSS, TOA, TDOA, Angle-of-Arrival (AOA), raw distance, etc.), generic data from any sensor (*i.e.*, inertial sensor such as accelerometer, gyroscope, digital compass, motion detector,

image frames from a digital camera, *etc.*).

- **Multi-Technology:** a target to be located could be comprised of several technologies at the same time. In this way, the requirement of supporting several technologies can be used to merge heterogeneous raw data to provide reliable and precise positions, thanks to the diversity obtained.

Figure Fig. 7.1 shows the UML class diagram of the possible components that are defined in the proposed architecture, needed for considering the previous requirement. As we can see, we have *Nodes* of two types: *Anchor* (with fixed and known positions) or *Mobile* (to be located). The *Networks* are groups of several *Anchor* nodes (at least one) to form a common WSN. Finally, the *Target* devices are virtual devices with an identifier that aggregate several *Mobile* nodes (*i.e.*, different technologies) and/or generic *Sensors* which can be jointly located with only one system query. As we can see, target nodes can consider multiple technologies, providing diversity that can be exploited by the algorithms.

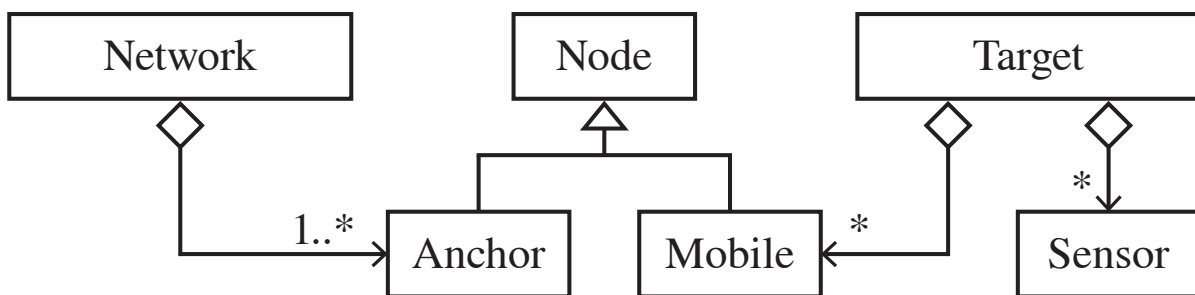


Figure 7.1: UML class diagram of the system nodes, sensors, networks, *etc.*

- **Multiple coordinate system and map-aware application support:** to give support to the map-aware applications in indoor scenarios, the architecture must be able to work with elements and positions with coordinates relative to different maps, which possibly have different local coordinate systems. The local coordinate systems can be defined by an origin point, a scale, and optionally an associated blueprint bitmap. Moreover, the architecture should support a global coordinate system such as the one used by satellite location systems, and should be able to combine both kinds of coordinate system transparently for users.
- **Data Fusion:** the system must provide mechanisms for data fusion coming from different raw measurements (technologies) and/or positions obtained from different algorithms. On one hand, low-level raw measurement fusion could be considered inside the location algorithms (*i.e.*, Kalman or Particle Filters), since an algorithm can obtain data coming from several technologies (sensors) associated to the same target. And, on the other hand, optional high-level fusion of final estimated positions must be provided directly by the platform. In this way, depending on the LBS considered, client applications could switch between, or combine, different algorithm position estimations.

- **Protection and Security:** the system should support at least one mechanism for secure authentication of external users who want to interact with the system, and optional protection of the data interchanged with the system, for instance by some means of data encryption.
- **API:** the platform must define an API describing the access methods, to enable client applications to interface with the RTLS. This API will define a boundary with syntax and semantics that specifies the seamless interface between external modules and the platform. Since the cited standards ISO24730-1:2006 [ISO19b] and ISO19762-5:2008 [ISO19a] do not cover all the requirements of the proposed platform, it is not possible to follow them. However, the API platform is based on them using standardized names and abbreviations for the different actors, commands, modules, parameters, *etc.*
- **Off-line data:** it must be possible to get measurements and position estimations on-line in real time and also off-line. Therefore, it should be possible to get the latest measurements or positions available or those from specific instants of time (off-line or historical data). Off-line data makes it possible to test and compare algorithms and client applications with a common set of data, and obtain repeatable results which are critical for research purposes.
- **Easy-to-Use:** users without special skills should be able to access the system to perform any of the following tasks:
 - Registration of WSNs, building blueprints, anchor networks, mobile nodes, generic sensors, and insertion of raw measurements into the system. This information will provide an abstraction of the hardware, offering users without hardware skills the possibility to obtain real measurements and sensor data without effort.
 - Obtaining measurements and inserting position estimation information by users who design location and tracking algorithms. They provide positioning data to end-user client applications which do not need special skills in how the location algorithms are implemented, optimized or obtain measurements.
 - Obtaining position information by end-user client applications in real time, for several mobile nodes, sensors, or groups of them, which can be implemented by using different WSN hardware technologies. These users do not need to know how and from which WSNs measurements were taken, nor from which location or tracking algorithms were these positions generated.

7.1.2 Main actors

One of the most important goals of the platform was to try to decouple as much as possible the different tasks usually involved in an RTLS system. The benefit of this approach is evident: the people in charge of making a hardware deployment do not have to be the same people who, for example, design the localization algorithm, in the same way that a final client or LBS does not

normally need to know with which technology their position was calculated.

Based on these different tasks, the next typical processes could be defined. Each of them could be performed without the need of having any knowlegment about the actions performed by the others actors.

- **Network Administration and Measuring Processes:** a network installer has to physically deploy a WSN (of anchor nodes) at an indoor scenario, from which to take physical measurements. Therefore, this user registers the *network* elements (*anchors*), *mobiles* and *generic sensors*, all of them with an associated identifier, and the anchors with their real and fixed positions relative to a *map* coordinate system. Once the devices are deployed and configured, they can start providing new measurements to the platform.
- **Target Device Administration:** the same network installer or another user can register virtual *target* devices, which are associated to several *mobile* nodes and/or *generic sensors* used for localization. These targets, and consequently the several elements associated to them, can be queried by other users (in the following steps) with a single system call.
- **Position Estimation using Location Algorithms:** a researcher or location algorithm developer can continuously get measurements from the system, without taking care about how the WSN networks were deployed, or how was the way to extract the measurements from the hardware. These users can implement several variants of a location algorithm which are fed with real measurements, from the same or from different WSNs. Specialized versions of *Location Algorithms* can perform low-level data fusion by retrieving measurements from several *networks* and several *sensors* at a time. After that, they can perform a tracking step, and finally return a position estimation to the *Location Server*. As always, the measurements and positions can be easily obtained from the system or inserted into the system by using standardized commands.
- **Localization Clients and LBS:** finally, several end-user client applications can access the system in real time to obtain position estimations from several technologies. LBS can also get the location information they need to work.

This approach has an additional advantage, which is that it allows the platform to be used both for use in a business-oriented environments and for use at the teaching or research level. For example, different research groups could deploy an instance of the platform in the cloud and connect to it the sensors that each group may have installed in their respective headquarters. Through the platform, everyone could use each others data to test new positioning algorithms in a transparent way.

7.1.3 Platform Architecture

Figure Fig. 7.2 shows the components that make up the proposed client-server architecture as well as the connections among the components in terms of a logic view:

- **Location Server:** the main component of the hybrid location system, which is composed

of several modules for storing measurements and positioning information obtained from external WSNs, sensors, and location algorithms. It allows external actors to insert and retrieve in a flexible way all manner of location information. As a server, it offers an input/output interface to allow remote connections.

- **WSNs and Sensors:** remote client applications that can be deployed in distributed hosts, and which provide all kinds of measurements and generic sensing data (*i.e.*, motion detection, acceleration, turning, inclination, *etc.*) to other actors such as *Location Algorithms*.
- **Location Algorithms:** client applications responsible for obtaining available measurements and sensing information from the *Location Server* to estimate new positions. Then, they can store the new positioning information inside the system. *Low-Level Fusion Algorithms* (*i.e.*, Kalman and Particle filters) are supported as specialized *Location Algorithms* which would retrieve several kinds of measurements from different WSNs (instead of from only one), accessing the *Location server*.
- **Localization Clients:** client applications that consume positioning information previously generated and filtered for one or more location algorithms, with the option of high-level position data fusion. They can also query for measurements if they need them for a specific purpose. A LBS is a particular implementation of a localization client.

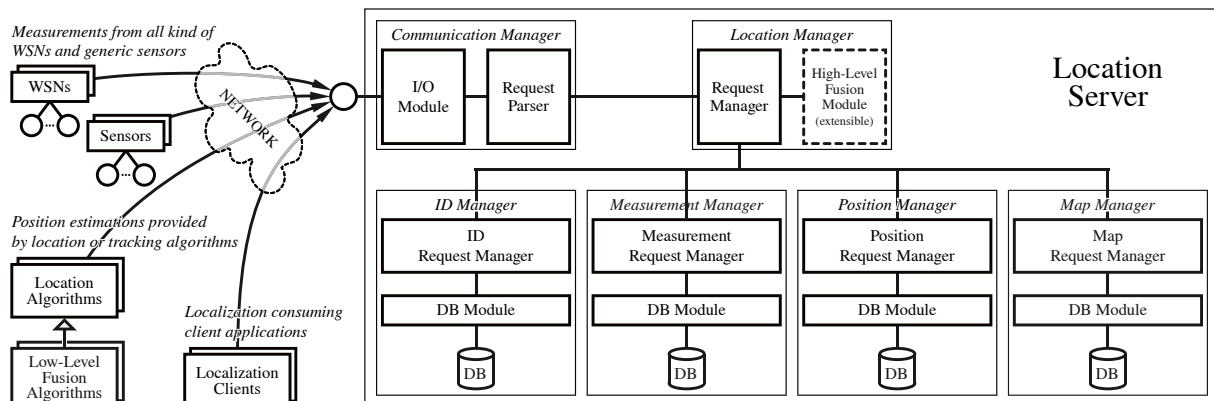


Figure 7.2: Proposed architecture for Hybrid Real-Time Location Systems.

7.1.4 Location Server

In addition, Fig. 7.2 shows the components of the *Location Server* architecture. It is composed of the following subsystems:

- **Communication Manager:** responsible for communications with the server to attend external clients, parsing the received requests and generating appropriate responses. Internally, it is made up of an *Input-Output Module* in charge of the data transmission from or to an external network, and a *Request Parser* module in charge of processing query messages and wrapping output messages. As will be shown in the following

sections, these messages are well defined, based on the javascript object notation (JSON) standard. JSON is a lightweight text-based open standard designed for human-readable data interchange, much more compact and readable than other data formats such as XML [XML19] and YAML [YAM19]. As application layer, HTTPS over TCP/IP was chosen, due its ability to be easily managed by common network devices and because it is one of the most common protocols.

- **Location Manager:** its mission is to analyze all the queries that come into the *Location Server* to find a suitable answer in the server by delegating in the *Position Manager*, *Sensor Manager* or *ID Manager* subsystems. The next element is the *High-Level Fusion Module*. It is responsible for creating fusion data based on the available position estimations from any technology that have been generated by any *Location Algorithm*. This module is easily extensible by a programmer by registering a new fusion module implementation in the system. Adding a fusion method key to an appropriate configuration file and the implementation class should be enough to add a new high-level fusion method to the architecture.
- **ID Manager:** its mission is to store the existing relationships among the *target* devices to be located and the *mobile* nodes and *sensors* assigned to it (see Figure Fig. 7.2 for more details about the relationships allowed in the system). This subsystem manages the associations and disassociations among the target devices, mobile nodes and generic sensors, and a log of these associations for future use.
- **Measurements Manager:** receives and stores measurements from WSNs and sensors, and attends queries from external *Location Algorithms* and *Localization Clients*.
- **Position Manager:** its mission is to receive and provide positioning data. Several *Location Algorithms* can store estimated positions in the system using this module, while external *Localization Clients* can query these data at any time. The stored positioning data is associated with an identifier of the *target* device or single *mobile* node or *sensor* from where it was measured, and an estimation of its accuracy, which is calculated by means of different aspects such as the kind of technology, the number of sensors available, the number of measurements per time, *etc.* The associated identifier can be a single mobile device, although more sophisticated *target* devices are managed by the *ID Manager*.
- **Map Manager:** receives and stores coordinate system information associated to several maps. Map-aware elements such as the anchor nodes or the position estimations generated by the *Location Algorithms* (or a specific *Low-Level Fusion Algorithm*) are relative to a particular map registered in the system. The different clients can register new maps, given an origin coordinate, a map scale and optionally a map bitmap (usually the blueprints of a floor of a building for an indoor scenario).

As shown in Figure Fig. 7.2, some subsystems are connected to different databases, which can be implemented in the same or in several physical databases. Also note that the *ID Manager*, *Measurement Manager*, *Position Manager* and *Map Manager* use the following two modules:

a *Request Manager* to parse the queries and encapsulate the result data, and a *DB Module* to obtain and store information from or into the appropriate database, abstracting and isolating access to it.

When isolating each part within the *Location Server* as we have done, using different modules and layers, we are able to correctly maintain and extend each part in the future. Therefore, we can give a proper support to future application requirements, or adapt the system to specific project constraints with little effort.

7.1.5 Protection and Security

For authentication purposes, OpenID [OPE19] was chosen as the most convenient manner of authenticating the final users with the *Location Server*. It is a highly mature and safe solution, which can be extended in the future. OpenID authentication used and provided by several large websites, such as AOL, BBC, Google, Yahoo!, IBM, MySpace, LiveJournal, Facebook, Vkontakte, Steam, Orange, PayPal and VeriSign. For sensors and low level devices, an alternative lightweight token-based system was implemented.

7.2 The platform in real projects

Since its conception and first version, the RTLS platform described in this work has been used in different projects in real environments. Some of them are detailed below.

- **Smartport Coruña.** 2015-2016: The aim of this project was to convert part of the port of A Coruña (Spain) into an *smart* port. For this purpose, different actions were implemented from different points of view, such as access control, route optimization, control of polluting emissions or both exterior and interior sensing.

An important part of the project fell on several LBS that therefore needed a source of location information to work. The RTLS platform was chosen to manage these positioning data.

One of the localization tasks of this project was to tracking the different vehicles that entered and left the port. For this purpose, more than 50 bluetooth low energy (BLE) readers were installed at different points in the area. These readers were able to detect a series of BLE beacons that were placed on the dashboard of all vehicles and sent the signal level results to the platform via WiFi.

In order to provide the location of the vehicles, a positioning algorithm based on trilateration with the received received signal strength (RSS) values was implemented. This algorithm received each new measurement from the deployed beacons and inserted a new position estimate. The position values were finally consumed by a positioning interface and a data analyzer created by another member of the consortium in charge of the project.

In addition to receiving and processing data from BLE readers, the RTLS platform was also used to detect the entry and exit of vehicles through the entrance door. Upon entering, the vehicles had to collect the BLE tag and place it on the dashboard, while upon leaving, they had to deposit it on a machine prepared for that purpose. Inside this machine was a BLE reader that detected at every moment which tags were inside. The RTLS platform was also used to monitor these data, so that the BLE detection data was received in it and an external LBS continuously monitored them in order to register a new target in the platform and start receiving their positions. This LBS also took into account data from a camera with a character recognition application capable of reading the license plate of the vehicle located just behind the entry barrier. With these two data, the license plate and the BLE identifier collected, the LBS was able to register the new target in the system unequivocally.

The experience with the RTLS platform in the Smartport project can be labeled as successful, as the location-related tasks could be completed without problems.

- **GEMA.** 2019-2020: The objective of this project is to monitor workers both inside and outside their workplace and apply an analysis on the data that allows to optimize processes and reduce costs. One of the legs of this project focuses on monitoring both patients and health care personnel inside a nursing home in A Coruña (Spain). For this purpose, the RTLS platform has been chosen as a mechanism to integrate location data from different BLE readers placed in the rooms and other common spaces of the residence. For their part, both workers and the elderly will wear a BLE bracelet to be able to be located at all times. This project is currently under development and the first results are expected in 2020.

After the requirements analysis, the RTLS platform was chosen as the system for handling location data for two main reasons: its capacity to work with any different technology (although BLE was chosen, it is not ruled out that new technologies can be added in certain spaces) and its possibility of working locally without the data leaving the building at any time.

- **LOCREA.** 2019-2020: The objective of this project is to provide a high-precision virtual reality mechanism for operators in complex scenarios. In this case, precision needs led to the choice of ultra-wideband (UWB) as the location technology to position the virtual reality device as accurately as possible.

In order to be able to trace the different readers and so that they can obtain the position estimates from UWB, the RTLS platform was also chosen as the system responsible for managing the location data.

In this case the communication of the UWB measurements is transmitted directly through this technology to a receiver node in charge of overturning the received values towards the platform. In parallel, work is being done on a positioning algorithm capable of obtaining the best possible position using the techniques of classification and mitigation of non-

line-of-sight (NLOS) measurements described throughout this work.

Additionally, the RTLS platform has been used as base of a *Doctoral* thesis [ROD15] and also, thanks to a collaboration with the department of Signal Theory & Communications, Universidad Carlos III de Madrid (UC3M), the platform was evaluated in several developments and measurement campaigns. Different articles were published as a result of this collaboration [ACH+09; ACH+10; ACH+11; ACH+12].

7.3 Platform Upgrades

After the different experiences of use with the platform in real environments, and following the natural process of any software solution, the platform evolved with new functionalities. Not only at the implementation level, with the emergence of new technologies and development frameworks with new potentials, but also at the level of structure and data model. The following sections show a summary of some of these improvements.

7.3.1 Internationalization

One of the first updates to the platform was to provide it with an internationalization mechanism for those attributes with different values depending on the chosen language. In general, the scheme used was to replace those attributes with *string* type with an object of *array* type. Each one of the elements of this array would be a new object with the form "*language*": "*value*", where *language* is one of the language codes defined by ISO 639-1 and *value* is the corresponding translation.

7.3.2 Target device

One of the cases detected when using the platform in real projects that was not contemplated in the first version was the situation when location devices affected several *targets* at the same time. The simplest example would be when a series of people, who up to that moment were being monitored with another different technology, get on a vehicle equipped with some other localization technology. In order to maintain a log of the positions of these people, it is necessary to assign as theirs the positions of the vehicle during the trip.

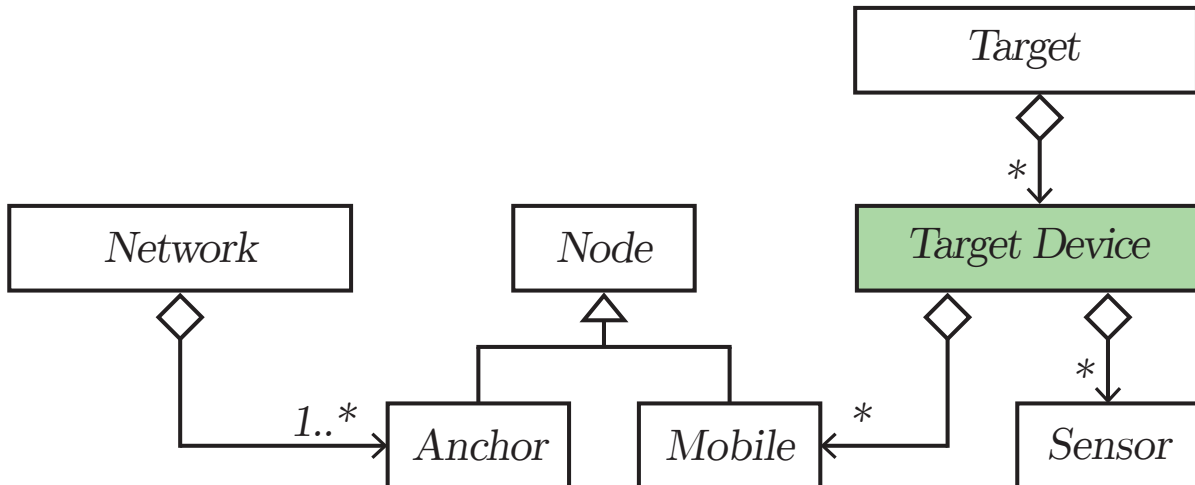


Figure 7.3: New target device entity.

This situation led to an extension of the platform in which a new entity was included: the so called *Target Device*. This entity models any device that includes one or more sensors or mobiles capable of providing location data. After this change, the position estimates become associated with each *Target Device*, instead of being associated with a specific *Target*. This means that in order to be able to locate the *Targets* (which are really the entities whose position it is necessary to know) it is mandatory to associate them at every moment with one or more *Target Devices*.

By means of these dynamic associations, cases such as the one mentioned above are supported. In that case the people would be associated to a *Target Device* while they were out of the vehicle, and once they get on it they would be temporarily associated to the *Target Device* that represents that vehicle. Thus it is possible to keep a continuous register of positions for these people.

In Fig. 7.3 the original class diagram is shown once modified to add the new *Target Device* entity. It can be seen how after this new addition the *Target* entity has no longer a direct relation with the *Sensors* or the *Mobiles*. This does not implies any lack of functionality with respect to the previous version. In fact, the platform is able to perform the same operations than before the upgrade, but now it support more complex relations between the physical location devices and the *Targets*.

7.3.3 Roles mechanism

In order to improve security and limit access to data, a role mechanism was added to the platform. This mechanism allows each user of the platform, whether end client or sensor, to be assigned a role or set of roles. These roles restrict the type of data that can be accessed, which can be inserted into the platform or whether the user can add new targets or sensors to the platform. There are administration roles that allow one or several users to manage the rest

of the other users roles. The roles are cumulative, so they can be added or removed in real time if necessary.

This is the list with the roles currently supported:

- **admin:** Full access to the platform.
- **add_measurement:** With this role a client can send new measurements to the platform. It is the typical role for *Sensor*, *Mobile* or *Anchor*.
- **add_position:** With this role a client can send new estimations of position to the platform.
- **add_alert:** With this role a client can send new alerts to the platform.
- **create_def:** With this role a client can create new definitions of technologies or alerts.
- **create_ent:** With this role a client can create new entities, like new *Targets*, *Target Devices*, *Sensors*, etc.
- **manage:** With this role a client can manage the relations between *Targets* and *Target Devices* or *Target Devices* and *Sensors* or *Mobiles*.

7.3.4 MQTT

One of the new functionalities added to the platform was the integration of a MQ telemetry transport (MQTT) broker [HTS08]. MQTT is a very light messaging protocol oriented to machine to machine (M2M) communications that uses mainly transmission control protocol (TCP) as transport layer (although there are variations that can work with user datagram protocol (UDP) as MQTT-SN [ST13]). The MQTT protocol is standardized by OASIS in its versions 3.1.1 [BG14] and 5.0 [BAN+19], the first from 2014 and the second from 2017. The main differences between the first and second versions are a series of improvements in scalability management, an improvement in error reporting, the addition of a series of message extension mechanisms such as user properties, payload format or content type, and in general a whole set of improvements related with the performance.

Among the main features of the MQTT protocol we can highlight its efficient use of bandwidth (the message headers are very small) and its looking for low energy consumption. The MQTT architecture is based on the publisher/subscriber philosophy, so that clients can publish their data associated with a topic (a unique identifier that may have different nesting levels) and other clients can subscribe to those topics to receive a copy of the data each time they are published.

The entity in charge of managing subscriptions and publications, and redirecting messages, is known as the MQTT broker. In addition to these tasks, other functions of the broker are to check security and control access or store messages for later sending in case of a network failure. There are many implementations of MQTT brokers and clients.

MQTT supports three levels of quality of service (QOS) [YAS+17], which are applied on the link between the client and the broker. The lowest level would be the QOS 0 level, known as *only once at most*. With this level the messages are sent only once from the client to the broker,

and there is no confirmation from the broker of the reception. The next level would be QOS 1 or *at least once*, and only guarantees that the message will be delivered, but this sent could be duplicated. Finally, the maximum level of QOS would be 2, or *exactly once*. With this scheme it is guaranteed that the message is processed in the broker and that this process is carried out only once.

The reason for adding MQTT support to the RTLS platform was to provide the system with a low throughput mechanism to handle many messages simultaneously. Deployments of very large location systems can be made up of hundreds or thousands of devices, so the volume of data transmitted per second can be very high. A protocol such as MQTT allows this flow of information to be absorbed efficiently [THA+14].

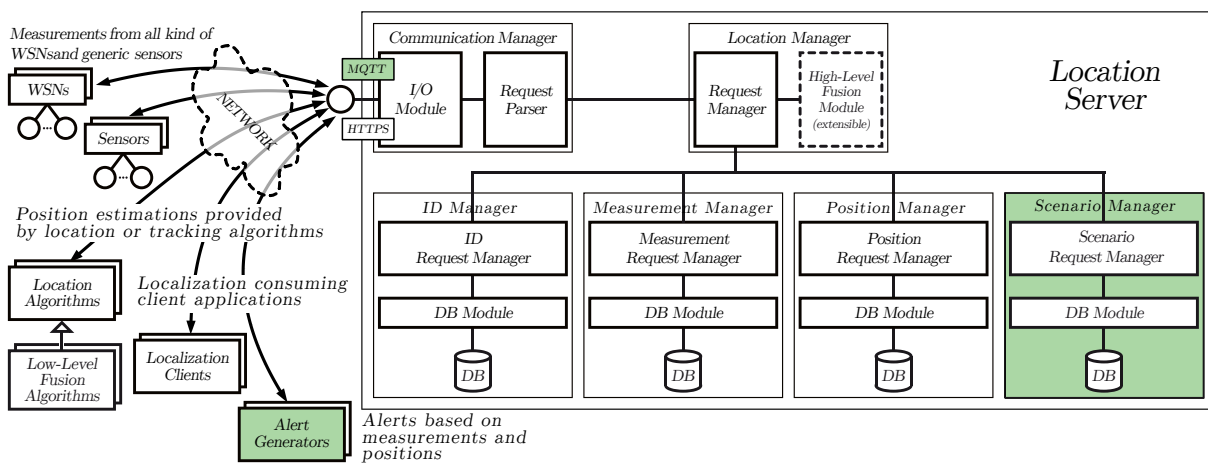


Figure 7.4: Upgrades in the RTLS platform architecture. Modifications are colored in green.

In Fig. 7.4 it can be seen how in the communications interface part of the platform this support for M2M messages has been added in parallel to the old hypertext transfer protocol (HTTP) interface. This interface is maintained both to receive information from sensors with lower throughput and bandwidth needs and to manage the high level requests of the different actors that can operate with the platform. The connection between the MQTT and HTTP worlds is made through a MQTT client that redirects messages from the HTTP part. In this way, the RTLS platform is independent of the MQTT broker you want to use.

7.3.5 Subscription Services

One of the shortcomings detected in the original version of the RTLS platform was its lack of a subscription mechanism similar to that of MQTT. To solve this problem, a subscription service was implemented through which external clients can register an end-point HTTP and receive *POST* messages with the updates produced in the platform.

The subscription mechanism allows a client to specify the type of data that it wants to receive (new positions, new measurements or new alerts) and the frequency of the notifications (in real time or at regular intervals). Besides, one of the subscription parameters is an optional filter that

allows the client to specify some restrictions over the data it wants to receive. For example, a subscription to receive new positions can include a filter to only receive those that correspond to a specific *target* or set of them, those that were created with a specific positioning algorithm or those resulting from using a sensor with a specific type of technology. The filter can also include temporary data, so data are only received at certain time intervals.

To receive the values, subscriptions must include a uniform resource locator (URL) to receive the data and, optionally, an object including the authentication parameters if necessary.

7.3.6 New map model

Sometimes the shape of a building can be difficult to adjust to a rigid scheme of vertical floors. Fig. 7.5 shows some of these cases where it is difficult to name some of the existing spaces. In order to withstand this problem, the RTLS platform was extended. The entity *map* was deprecated and the following new ones were created instead: *Scenario*, *Level*, *Blueprint* and *Level Connector*. Each of these entities models the following aspects:



(a) By Andreas Praefcke, CC BY 3.0.



(b) By the wub, CC BY-SA 3.0.

Figure 7.5: Different scenarios with heterogeneous floor configurations.

- **Scenario:** models a building or group of buildings that share targets or location devices. Examples of scenarios would be an hospital, a factory, a school, etc.
- **Level:** models a space of a building in which the ground level is at the same height. Typically corresponds to a floor, but is not necessary. For its definition it is necessary to specify the global height at which the ground is, and the coordinates that define its perimeter. Examples of *Level* would be *Floor 5* or *Basement A*.
- **Blueprint:** models the graphical information needed for some LBS. Each *Blueprint* entity corresponds to a *Level*. This entity includes the graphic files that represent the *Level* but can also include information about which part of the surface is open space and which part corresponds to non-accessible spaces.

- **Level Connector:** models a connection between levels. It can be used by LBS looking for routes between plants or certain nearby points of interest.

7.3.7 Semantic positions

One of the most important changes of the platform is the one related to the modeling of the location measurements. Depending on the type of LBS, the needs for retrieving location information may be different. Some of these services for example need to access position estimates at a very low level, at a *geometric* level. Services of this type could be, for example, the ones in charge of performing autonomous driving of vehicles or monitoring of emergency equipment. In these cases an estimation of the *3D* position within a *Level* is required, as well as an estimation of its orientation. In addition, in this type of services it is required an estimation of the error committed in each of the dimensions considered, which is usually modeled as a matrix of covariances.

There are another types of LBS, however, that do not need geometric data to work. In fact, one of the main problems when implementing an indoor positioning system at the enterprise level is the belief by customers that the ultimate goal of such a system should be to obtain an estimate in coordinates (x, y) or (x, y, z) . After years of work in this field, experience seems to indicate that this is not true in general. In many occasions a simple estimation of position is unusable directly, so it is necessary to create some kind of superior process logic in order to extract from it some kind of information of interest for the client. For example, within the operation of an hospital it is of no importance to know that a doctor is at the point $(3.5, 10.5, 1.8)$ at a given time. On the other hand, knowing that he or she is in *Operating room 3* can be much more useful.

This modeling of positioning information could be defined as *semantic* or *symbolic* location. These concepts are not new in the literature [HL04], and although their definition is not exactly the same in all of related works, there are some common characteristics:

- A semantic position must be unequivocally identifiable.
- A semantic position can have relationships of belonging or closeness to other semantic positions.
- A semantic position must be easily readable by machines or humans.
- A semantic position must be able to be defined dynamically.

Although the possibilities of implementing a semantic localization system are countless, the approach followed in the RTLS platform was one centered on the concept of area of interest (AOI). This is the entity that allows to link the geometric world with the semantic one. On the one hand, an AOI is defined in a semantic way with its name and high level attributes. On the other hand, the geometric space it occupies within the scenario is defined and a central point is also defined as a representative of the AOI. The AOIs also have a hierarchical structure, so that an AOI can include one or several other AOI.

In view of the previous points, the location platform was modified to accommodate two new types of positions: geometric positions and semantic positions. In the RTLS platform both types of position were implemented in a transparent way, so that any other entity that used them (such as the position of an anchor sensor) could use one or the other type without restrictions.

The geometric position is defined by the next attributes:

- **type**: Defines the type of position. For geometric positions, this value is set to *3D+*.
- **scenarioId**: Identifier of the scenario.
- **levelId**: Identifier of the level.
- **pose**: A object that contains the next attributes:
 - **position**: An array of 3 numbers corresponding with the (x, y, z) coordinates.
 - **quaternion**: An array of 4 numbers corresponding with the (x, y, z, w) values of the quaternion.
- **covariance**: An array of 6×6 numbers with the covariance matrix that models the error estimation.
- **localTimestamp**: An optional timestamp defined by the creator of the position. This value is useful for inserting experimental positions that do not occur in real time. If omitted, the timestamp of data reception on the platform is used instead.

In a similar way, the semantic position can be defined with the next attributes:

- **type**: Defines the type of position. For geometric positions, this value is set to *semantic*.
- **scenarioId**: Identifier of the scenario.
- **levelId**: Identifier of the level.
- **aoiIds**: An array of AOI. Each one is defined by the next attributes:
 - **aoiId**: The AOI identifier.
 - **p**: A numeric value greater than 0 and less than or equal to 1, where 1 would be total certainty that the *target* is at that AOI and 0 total certainty that it is not at that AOI.
- **localTimestamp**: As in the geometric position, this is an optional timestamp defined by the creator of the position.

The previous both positions models were added to the other two already present in the platform, the global coordinates and the *2D* location in local coordinates without orientation.

7.3.8 Alerts

One of the original limitations of the location platform was its lack of mechanisms for incorporating events generated from location data. Many LBS do not need to receive constant position updates from a *target*, they just need to receive certain discrete events that activate their functionality.

In order to support this functionality, the platform was extended in a similar way to how it had been done with the rest of the actors identified in the first version. Thus, if in that first version a series of external actors had been identified, such as sensorization devices,

location algorithms and final clients, now a new one is added that would correspond to the alert generators. The function of an alert generator could be defined in three stages:

1. In a first stage, the alert generator registers the definition of a new type of alert on the RTLS platform. This definition includes the parameters that will be published when the alert is produced together with its type (*floats, strings, etc.*). Additionally, semantic information can be included with the description of each field and of the alert itself (trigger conditions, periodicity, etc.). To model these definitions, a new entity was created within the platform called *AlertDefinition*.
2. In the second stage, the alert generator subscribes to those types of data necessary for its operation. New positions or new measurements allow it to infer whether or not a new alert should be generated.
3. Finally, in the final stage, the alert generator sends a new alert when the alert conditions are satisfied. In order to model this situation, a new type of entity called *Alert* was added to the platform. This entity simply includes an attribute *alertData* which details the values of the parameters previously defined in the object *AlertDefinition* which models this type of alert.

By means of this implementation, all the alert generation logic is placed outside the RTLS platform, just as it had been done before with the implementation of the location algorithms or the sensors. Fig. 7.4 shows how in the updated architecture of the platform this new actor is outside but has a direct communication to get measurements and positions and to insert new alerts. In this way, the flexibility of the platform and its capacity to be used in multiple different problems is maintained. As a final note, it can be said that the subscription mechanism implemented in the platform allows a client to subscribe to one or several different alert definitions, so that each time an alert is generated with any of these definitions the client receives the corresponding notification in real time.

7.4 Conclusions

This chapter presents a multi-technology RTLS platform. The platform allows to separate the different entities that are normally involved in this type of solutions: the different location devices, the positioning algorithms or the final clients and LBS.

The architecture provides a complete API with several methods for interfacing with external modules. Hardware providers can set up a network in the platform based on their own technology in a transparent way, for example, to the algorithm developers. Algorithms can use raw data from one or more available technologies to provide a location estimation. The location estimations are available in real time (or off-line, with access to historical data) for location applications.

The advantage of the proposed architecture, as opposed to monolithic solutions, is that it allows a generalization and retargeting for new LBS and applications, reducing costs of further

development. Other kind of purpose built systems can be expensive and hard to implement in practice.

The platform has been used in several real localization projects with different requirements and technologies during the last years. The experiences obtained in these projects served to possible improvements that helped to expand the functionality of the platform. Thus, several mechanism were added, like the internationalization of texts or the support of MQTT messages, as well as new entities and data models like the *Target Devices* or the semantic positions.

7.5 Contributions related with the chapter

The details of the RTLS platform were presented in the following contribution:

- Javier Rodas, Valentín Barral, and Carlos Escudero. “**Architecture for multi-technology real-time location systems**”. *Sensors*, vol. 13, no. 2, 2013, pp. 2220–2253

Chapter VIII

Conclusions and future work

This chapter presents the conclusions drawn from the work carried out and the future lines. Also a summary of the contributions is presented. Section 8.1 shows the conclusions, Section 8.2 shows the future lines, and Section 8.3 summarizes the contributions during this thesis.

8.1 Conclusions

The work described in this memory can be divided into three distinct parts. The first of them focused on proposing a solution to the problems observed with ultra-wideband (UWB) technology when there is no clear line-of-sight (LOS) between the devices. In order to do this, an approach based on machine learning (ML) was proposed, in such a way that a series of classification and mitigation algorithms would be trained with LOS and non-line-of-sight (NLOS) measurements to be later applied in new scenarios.

The second part of the memory focused on simulation. For this, a physical simulation platform was presented on which a location-oriented UWB device simulator was built. This simulator was used to simulate a case of industry use, such as the positioning of forklifts in a factory or warehouse. In addition, the simulator was also used to obtain *range* and position data that allowed comparing the similarity between the results obtained in a real scenario with the simulated ones.

Finally, the third part of the memory focused on describing a multi-technology real-time location system (RTLS) used in different real location projects. Its characteristics and the evolutions suffered since its first version were described.

The conclusions drawn from each of these parts are detailed below. Section 8.1.1 summarizes the conclusions of the first part of the work (NLOS classification and mitigation), Section 8.1.2 details the conclusion of the second part (UWB simulation, and finally Section 8.1.3 shows a summary of the conclusions of the third part (RTLS platform).

8.1.1 NLOS Classification and mitigation

The first part of this memory focused on one of the most important technologies for indoor location: UWB. By means of this technology very precise distance estimates can be obtained, and due to the proliferation of companies dedicated to the manufacture of UWB devices the prizes have fallen considerably. So nowadays exist in the market even low-cost UWB devices such as the ones from Pozyx company, modules that include the DW1000 transceiver (see Appendix A). This is one of the most used UWB transceivers in both the industry and the research community, as it was the first commercial device to implement the standard IEEE 802.15.4-2011 (info on this standard can be seen in Appendix A).

However, as it was shown in Chapter 1, the accuracy in positioning obtained by systems of this type are only excellent when an ideal scenario is considered. Thus, already in this first chapter it was described how in situations of NLOS the accuracy and precision are degraded in some cases in a very important factor.

The first part of the work consisted on a study of the possibilities of applying ML techniques to detect and mitigate the effect of the measurements obtained in a UWB system in an NLOS scenario. This study was developed along Chapters 2 to 5, being the first of them the one that tried to look for the best features for the ML models when the channel impulse response (CIR) is not available or easy to acquire. This work gave as a result that with simple statistics like the average of *ranging* and received signal strength (RSS) it was enough to obtain very good classification results with classic ML algorithm such as support vector machine (SVM).

In the next chapter it was analyzed to what extent detecting and mitigating NLOS measurements could help with the final performance obtained by a positioning system. As expected, it was confirmed that the higher the number of NLOS measurements the worse was the error made by the location algorithms. Therefore ignoring NLOS values could reduce the error of the position estimation in many centimeters.

In the third chapter the classification and mitigation techniques were analyzed in greater detail, testing different algorithms based on ML. The tests served to corroborate the improvement in ranging values when using these techniques in a specific scenario.

In the last chapter dedicated to this solution, it was possible to extend the previous results to another different scenario, so that it was proved that a classifier trained with measurements coming from one scenario could be used on the measurements of a totally different scenario and also reduce the final positioning error. Another conclusion of this last chapter was that mitigation did not appear to be general enough to be used in the same way, as in this case the error values were not reduced.

As a final conclusion after this first part of the work, it seems clear that there is an opportunity to improve the results obtained with UWB devices when these are used in environments without direct line of sight. In these cases the precision that is achieved is very far from the few centimeters promised, since the multi-path together with the attenuation of the first path can cause very big ranging errors. With the approach raised in this work, and that could be

implemented even in microcontroller unit (MCU) not too powerful as the included in low-cost devices such as Pozyx or DWM1001C of Decawave, a UWB based system could be improved with greater robustness when deployed in a real environment where there is no guarantee of always having a good LOS.

8.1.2 UWB simulation

In the second part of this work, corresponding to chapter Chapter 6, a software solution to simulate UWB measurement was presented. This application was integrated into the Gazebo simulation engine and the Robot Operating System (ROS) operating system, both oriented to robot simulation. The simulator presented, and built using ML techniques applied to measurements captured with real devices, was implemented and used to simulate a common problem of indoor location: the location of pallets in an industrial warehouse. In this case, the approach to the problem was oriented towards the location of the forklifts instead the pallets themselves.

During the simulation, different indoor scenarios were built where various UWB beacons were placed. In addition, a vehicle was created to simulate the forklift and it was equipped with a UWB tag, inertial sensors and an optical sensor: the PX4 Flow. The vehicle moved through the scenarios while a location algorithm (an iterative extended Kalman filter (IEKF)) used the different measurements from the sensors to try to estimate its position. The results were compared by using one, two or all three sensors simultaneously, and it was verified how the positions estimates improved when more information was available.

In order to check the reliability of the UWB simulator, measurements were made in a simulated environment that precisely tried to imitate the environment from which the measurements used to create the software had been extracted. It was verified that the estimates were close to reality for the LOS and NLOS-*Soft* cases, while for the NLOS-*Hard* case they seemed not to be so similar.

In order to extend this verification of the quality of the measurements generated by the simulator, a new simulation was carried out, in which the room of the Scientific Area Building in A Coruña that had been used in Chapter 5 to check the generalization capacity of the classification and mitigation solution based on ML was replicated inside Gazebo. Once the 3D model of that area of the building was created, the beacons were placed in the same positions as in that situation and measurements were taken with a simulated tag placed in the same points of a 3×3 grid that had been used in the real case.

Finally, the positioning algorithm was executed with the measurements obtained in the same configurations that had been tested in the real case: with and without ignoring measurement, with and without mitigation and with and without respecting the minimum number of 4 ranging values for each iteration of the algorithm. The final positioning results showed a great closeness between both situations, with values of a few centimeters of difference of mean absolute

error (MAE) in many configurations. The biggest difference observed between the simulation and the real world was the improvement that the mitigation caused in the simulation and that had not been seen in the real case.

The most important conclusion after this second part of the work is that now there is a high-level UWB simulator oriented to positioning that can be very useful for testing algorithms before making a real deployment of technology and, additionally, can also be very useful precisely to plan that deployment and find the areas with a greater likelihood of suffering NLOS and therefore obtain a less accurate location.

8.1.3 RTLS platform

In the third and last part of this work, a multi-technology RTLS platform was presented, capable of managing different actors that are normally involved in a localization system. Thus, the platform serves as a link between location based services (LBS) and final location clients, algorithm developers, physical sensors or alert managers. The platform has been validated in several real localization projects, using different technologies as a base (since the platform is not tied to any of them in particular).

Precisely because of these real projects, the platform has evolved in recent years. Chapter 7 details all the new functionalities and evolutions that the platform has adopted since its first conception.

The final conclusion of this part of the work would be that any location system, to be really useful within another higher organization, must have a flexible mechanism both to access the positioning data and to manage the physical devices that create them. A closed, technology-centric solution can work well while it does, but rapid advances in technology can make the solution obsolete in the medium to long term. With the RTLS platform presented in this work, the final localization technology is transparent for the system, so that adding new technologies or extending existing ones only has the effect of improving the reliability, accuracy and robustness of the system.

8.2 Future Work

With regard to the first part of this work, focused on the construction of a NLOS situation detector and mitigator system with UWB devices, the future lines could be summarized in the following points:

- **Applicability of the solution to new technologies:** with Bluetooth 5.1 technology (capable of providing angle of arrival (AOA) and/or angle of departure (AOD)) around the corner and the IEEE 802.11mc standard under development (including support for time of arrival (TOA) with Wi-Fi Round-Trip-Time), it seems clear that UWB will have competition in different scenarios. One of the future lines to consider will therefore be to

check to what extent the approach used in this work with ML algorithms to detect NLOS is applicable to these other technologies.

- **Extension of the analysis to different devices and configurations:** another important future line will be to extend the experiments proposed in this work considering other different UWB devices and configurations (frequency band, preamble length, data rates, antenna orientations, etc.).
- **Study of the feasibility of real implementation in MCUs:** a future line should include the study about the computational feasibility of implementing the proposed solution within the small MCU included in the hardware of many UWB solutions. This would allow the possible improvement in the ranging values independently of any other superior entity (typically a location algorithm) using such data.

As far as UWB simulation is concerned, future lines include:

- **Expansion of the ray model:** a future line includes improving the modeling of rays so that they can bounce more times and in more areas until now not contemplated, as the ceiling.
- **Expansion of the variability of the training set:** in order to improve the performance of the simulator, a future line should include the creation of a more heterogeneous set of training measurements, with measures in more scenarios, with different hardware and in different physical configurations (rotations, type of obstacles, etc.).

Finally, the future lines related with the RTLS platform would be the following ones:

- **Research on logic distribution mechanisms:** so far the platform is centralized, so that all information is concentrated in a single point. A future line must include the search for techniques that allow the distribution of logic in local units so that they can work autonomously at least for a certain interval of time. This will give the platform even more flexibility when it comes to managing its implementation within an organization.

8.3 Contributions

During the work in this thesis, the author contributed to the research community in different ways. Fig. 8.1 summarizes these contributions, that are analyzed in the next sections. Thus, Section 8.3.1 describes some of the projects were some result from this thesis was used. Section 8.3.2 shows the journal publications co-authored by the author of this thesis, whereas Section 8.3.3 does the same with the conference papers. Section 8.3.4 summarizes the measurement dataset published and, finally, Section 8.3.5 describes a technological-base company co-founded by the author during this period.

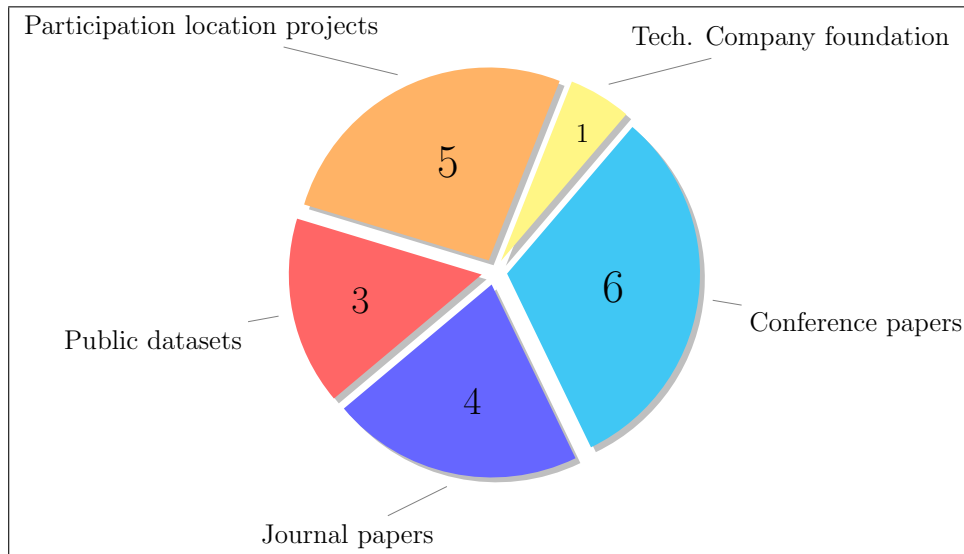


Figure 8.1: Author contributions during the work on the thesis

8.3.1 Projects

During the work presented in this thesis, some of the results have been used in different projects

1. **Smartport Coruña** (2015-2016): During this project a part of the port of A Coruña, Spain was converted into a *smart* one. One of the parts of the project included the real-time location of vehicles and staff circulating around the port. In this case the technology chosen was bluetooth low energy (BLE), so different readers were deployed in the port. The communication between these devices and their coordination with the rest of the systems of the port was carried out by means of the RTLS platform developed during this thesis.
2. **LOCYU** (2015): In this project, an indoor location system was implemented in a shopping centre in Santiago de Compostela, Spain. The objective of the system was to store the parking spot and guide the user to it (and to other points of interest within the mall). More than 200 devices based on BLE technology were deployed, and some modules of the RTLS system described in Chapter 7 were used to model the information.
3. **Drone service** (2015 - 2017): This project was focused on implementing an indoor positioning system based on UWB to guide drones. Some findings regarding the limitations of UWB technology in its implementation in the DW1000 were used for this project.
4. **GEMA** (2019 - 2020): The main objective of this project is to manage the mobility of workers, both those who work outside their usual place of work and those who work inside. One part of the project seeks to locate nurses and residents of a nursing home in A Coruña, Spain. In this case the technology chosen was BLE, with readers installed at different points in the building. The management of all the location data and its integration with the rest of the software of the nursing home is carried out with the RTLS

platform described above.

5. **LOCREA** (2019 - 2021): This project combines augmented reality and high-precision location in an indoor environment. It uses UWB devices as base technology. The RTLS platform developed is used in this project, together with an implementation of NLOS detection techniques based on machine learning described in this work.

8.3.2 Journal papers

The work done during this thesis resulted in the publication of the following co-authored journal articles.

1. Javier Rodas, Valentín Barral, and Carlos Escudero. “**Architecture for multi-technology real-time location systems**”. *Sensors*, vol. 13, no. 2, 2013, pp. 2220–2253
2. Valentín Barral, Carlos J. Escudero, José A. García-Naya, and Roberto Maneiro-Catoira. “**Nlos identification and mitigation using low-cost uwb devices**”. *Sensors*, vol. 19, no. 16, 2019. ISSN: 1424-8220.
DOI: 10.3390/s19163464. Online access: <https://www.mdpi.com/1424-8220/19/16/3464>
3. Valentín Barral, Pedro Suárez-Casal, Carlos J. Escudero, and José A. García-Naya. “**Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments**”. *Electronics*, vol. 8, no. 10, 2019.
DOI: 10.3390/electronics8101152. Online access: <https://www.mdpi.com/2079-9292/8/10/1152>
4. Valentín Barral, Carlos Escudero, José García-Naya, and Pedro Suárez-Casal. “**Environmental cross-validation of nlos machine learning classification/mitigation with low-cost uwb positioning systems**”. *Sensors*, 2019. Under second revision.

Note that all the journals appear in Journal Citation Reports (JCR). The impact factor (IF) is summarized in Table 8.1.

Journal	Year	IF	Category	Position	Q.
Sensors	2013	2.457	INSTRUMENTS & INSTRUMENTATION	10/57	1
Sensors	2018	3.031	INSTRUMENTS & INSTRUMENTATION	15/61	1
Electronics	2018	1.764	ENGINEERING, ELECTRICAL & ELECTRONIC	154/265	3

Table 8.1: Impact factor of journals.

8.3.3 Conference papers

The work done during this thesis resulted in the publication of the following co-authored conference articles:

1. Héctor José Pérez Iglesias, Valentín Barral, and Carlos J Escudero. “**Indoor person localization system through rssi bluetooth fingerprinting**”. *Proc. of 2012 19th*

- International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 40–43
2. Valentín Barral, Javier Rodas, José A García-Naya, and Carlos J Escudero. “**A novel, scalable and distributed software architecture for software-defined radio with remote interaction**”. *Proc. of 2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 80–83
 3. Javier Rodas, Valentín Barral, Carlos J Escudero, and Robert Langwieser. “**A solution for optimizing costs and improving diversity of rfid readers**”. *Proc. of 2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 84–88
 4. Valentín Barral, Pedro Suárez-Casal, Carlos Escudero, and José A. García-Naya. “**Assessment of UWB ranging bias in multipath environments**”. *Proc. of Proc. of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Alcalá de Henares, Spain, 2016
 5. Valentín Barral, Carlos J. Escudero, and José A. García-Naya. “**Nlos classification based on rss and ranging statistics obtained from low-cost uwb devices**”. *Proc. of 27th European Signal Processing Conference (EUSIPCO)*. A Coruña, Spain, 2019
 6. Valentín Barral, Carlos J. Escudero, Pedro Suárez-Casal, and José A. García-Naya. “**Impact of nlos identification on uwb-based localization systems**”. *Proc. of 10th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Pisa, Italy, 2019

8.3.4 Public datasets

The measurements captured to perform the different experiments were uploaded a published without restrictions to be used by others members of the research community:

1. Valentín Barral. **NLOS classification based on RSS and ranging statistics obtained from low-cost UWB devices - dataset**. 2019.
DOI: 10.21227/swz9-y281. Online access: <http://dx.doi.org/10.21227/swz9-y281>
2. Valentín Barral. **Environment cross validation of NLOS machine learning classification/mitigation in low-cost UWB positioning systems - dataset**. 2019.
DOI: 10.21227/rhhs-fw33. Online access: <http://dx.doi.org/10.21227/rhhs-fw33>
3. Valentín Barral. **Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments - dataset**. 2019.
DOI: 10.21227/b9tf-fv74. Online access: <http://dx.doi.org/10.21227/b9tf-fv74>

8.3.5 Technological-base company foundation

As part of the effort in the transference of knowledge, the author of this thesis was founding partner of a technological-base company named Sevenix Ingeniería S.L. [SEV19], in 2014.

The company received the title of *Technology-based Business Initiative* by Xunta de Galicia (regional government of Galicia, Spain) in 2015. The company is dedicated to the development of custom hardware, especially oriented to Internet of Things and low-power sensors. The company has different machines for prototyping and electromagnetic testing and has experience with many of today's technologies such as UWB, LORA, or Sigfox, in addition to the classic WiFi, BLE, or RFID. Sevenix has participated throughout these years in different projects, both nationally and internationally, in collaboration with companies from different sectors, such as automotive, shopping centers or wood processing companies.

Appendix I

Ultra Wideband and IEEE 802.15.4-2011

ultra-wideband (UWB) is a radio technology that is characterized mainly by using low power for transmission occupying a large bandwidth in frequency. Normally this technology is based on the transmission in time of pulses of extremely short duration, which are transformed in frequency in a great occupation of the spectrum.

Historically, UWB has been used since the 1970's as a base technology in radar applications (mainly in the military field) and for communications. Especially successful was its use of radar devices, due to the high accuracy that could be obtained and the ability of the technology to penetrate objects [TAY94].

In the mid-1970s, the first US patent was obtained for a UWB-based communications system. Especially again in the military field, various UWB communications solutions were developed due to their difficulty in being intercepted.

However, it was not until around the year 2000 that the first commercial UWB-based communications systems for the general public emerged (also in the US). The first regulations on UWB from the Federal Communications Commission (FCC) in the US would arrive in 2002, where three fields of application were contemplated: imaging systems (ground and wall radars, medical systems), vehicular radar systems and communications and measurement systems. Within this last group, the regulation made a distinction between indoor and outdoor devices. It is precisely the UWB regulation for use in indoor communications systems that is still in force in the US for location systems based on this technology.

Because one of the main uses of UWB technology nowadays is precisely the indoor location. The high temporal resolution of these systems allows to measure time with great accuracy, which is very useful to calculate the propagation time of a signal from an emitter to a receiver. The big push to this UWB version would come with IEEE approval of standard 802.15.4a, a branch of the previous 802.15.4, in 2005. From there would appear the first commercial positioning systems based on UWB that have evolved to the range of products we have today.

Among the benefits of UWB, the next could be the most important ones:

- Communication and ranging at the same time. Each communication transmission can be used to extract ranging information, so an application based in this technology could use

location and message transmissions simultaneously.

- Multi-path immunity. Due to the extremely short duration of the pulses, it is easier for a receiver to detect the reflected pulses than in a traditional narrow-band system.
- Low equipment cost. When using the carrier-less version of UWB, the pulses are emitted without a carrier signal so it is no need the typical radio-frequency (RF) structure common in other RF systems based on sine-waves.
- High data rate. Because their high bandwidth, these systems can reach a high data rate.

This appendix is organized as follows: Section A.1 details the main aspects of the IEEE 802.15.4-2011 standard, including a description of UWB impulse radio in Section A.1.1 and a summary of both physical and medium access control (MAC) layers in Section A.1.2 and Section A.1.3. Finally, Section A.2 describes the current regulation of UWB in indoor scenarios for the United States and Europe.

A.1 The IEEE 802.15.4-2011 standard

The IEEE 802.15.4a [KAR+10] was the first international standard that defined a wireless physical layer to enable precision ranging with UWB. The works related with the definition of this standard started in 2004, when the first version of the IEEE 802.15.4 came out. This standard was focus on defining the operation of low-rate wireless personal area networks, including the physical layer (PHY) and MAC. Once this standard was completed and released publicly, it was evident that the number of applications based on this kind of networks could be much bigger if it were possible to measure the distance between the different devices with great accuracy. This functionality had been excluded of the IEEE 802.15.4 due the limited signal bandwidth, but in 2004 the IEEE 802.15 Low Rate Alternative PHY Task Group was created to define a new PHY and MAC capable of supporting this type of *ranging* functionality. A year later, in 2005, a first proposal was approved by the group. This first proposal included two different optional PHYs, one based in UWB impulse radio and other based in chirp spread spectrum (CSS).

Although the goal of both proposals was to complete the previous standard with support for higher data rates, longer ranges, lower power consumption and adding the *ranging* capabilities, the differences between the two versions is noticeable. While the solution based on UWB impulse radio was created to support applications that demanded large range and a quite high data rate, the solution based on CSS was more oriented to applications that did not need a so high throughput but needed long ranges at high speeds (i.e. intra-vehicle communications).

The differences at definition level is also evident, since while the UWB impulse radio version was defined to operate on three different frequency bands (250 MHz to 750 MHz, 3244 MHz to 4742 MHz and 5944 MHz to 10 234 MHz), the CSS version was defined to operate only on the ISM band (2400 MHz to 2483.5 MHz). The different data rates specified for each definition were also different, defining the UWB impulse radio version four modes (110 kbit/s,

851 kbit/s, 6.81 Mbit/s and 27.24 Mbit/s while the CSS only contemplated two (1 Mbit/s and 250 kbit/s).

From the point of view of this thesis, however, the main difference between both proposals is the lack of support for *ranging* in the CSS version. That is why this appendix will be focus only on the UWB impulse radio version.

The two PHY proposals along with one for the MAC layer were finally completed in 2007, this being the year in which the IEEE 802.15.4a standard was finally published.

Some years later, in 2011, the original 802.15.4 standard was updated to incorporate the 802.15.4a definitions, thus emerging the standard known as IEEE 802.15.4-2011.

A.1.1 UWB Impulse Radio

UWB Impulse Radio uses small temporal RF pulses to send the information. These pulses have a very short duration, typically in the order of nanoseconds, causing a wide bandwidth in frequency (Ultra WideBand) [GMK07; OHI05].

There are two alternatives for producing a UWB impulse radio signal. The first one does not need Local oscillator (LO) or mixers for the RF stage, since the pulses are directly transmitted in baseband. This version would correspond to the UWB impulse radio carrier-less version. With this variant, pulses with certain characteristics are used so that the shape of the power spectral density (PSD) falls within the bands defined for UWB. In this sense, the family of Gaussian pulses is ideal for this type of transmission. Normally the derivatives of the higher order of the pulse are used, since the higher the order the higher the central frequency is placed in the spectrum occupied by the signal.

A different alternative for generating UWB impulse radio signals is called carrier-based. In this variant a LO is used to create a high frequency carrier signal that is then shaped with an envelope so that the DC energy is moved to the desired UWB band. The envelope can be of various types, rectangular, triangular, root-raised cosine or Gaussian [YE+11]. Thus, carrier-based transmitters can transmit the pulses in two different ways. Either by moving them to the desired center frequency through the signal generated by the LO and a given envelope, or by directly switching the LO.

In UWB IR there are several basic modulation methods that can be used to codify the information. Some of them could be: pulse position modulation (PPM), burst position modulation (BPM), pulse amplitude modulation (PAM) or on-off keying (OOK).

PPM is one of the most common methods. In this case, the pulses are sent in advance or with some delay depending on the value to transmit. The next equation models this behavior, where s_i are the used pulses, $p(t)$ is the selected pulse shape and τ_i is the delay parameter used to create the pulse s_i :

$$s_i = p(t - \tau_i) \tag{A.1}$$

PPM can be used to create a $M - ary$ system, simply using M different values of τ_i . This method is very simple, but requires an extremely fine time resolution to modulate the pulses (so is necessary a sub-nanosecond accuracy).

BPM is another common method, but in this case the pulses are sent at the same rate but their shape changes depending of the value transmitted: a pulse and an inversion of the same pulse are used. The next equation models this behavior, where s_i are the used pulses and $p(t)$ is the selected pulse shape:

$$s_i = \sigma_i p(t), \quad \sigma_i = 1, -1 \quad (\text{A.2})$$

With this method only a binary system can be built.

In the PAM modulation, the amplitude of the pulse is used to codify the information. Thus, the pulses s_i are defined as:

$$s_i = \sigma_i p(t), \quad \sigma_i > 0 \quad (\text{A.3})$$

with σ_i a positive number. As in PPM, this method allows using an arbitrary number of different pulses, simply using distinct values of σ .

OOK can be seen as a particular case of PAM where $\sigma_0 = 0$ and $\sigma_1 = 1$. That means that only s_1 is transmitted, whereas each 0 bit is modeled as an *empty*. This modulation method only allows a binary set of pulses:

$$s_i = \sigma_i p(t), \quad \sigma_i = 0, 1 \quad (\text{A.4})$$

To transmit much larger volumes of information than a simple set of pulses normally a pulse train is used. A pulse train can be defined as:

$$s(t) = \sum_{n=-\infty}^{\infty} p(t - nT) \quad (\text{A.5})$$

where T is the period and $p(n)$ the basis pulse shape.

A problem that this type of structure has is that when the pulses are sent always at the same interval then the corresponding spectrum contains peaks at some specific frequencies. This is a undesirable effect that limits the total transmitted power. A technique to mitigate this effect is the use of a pseudo noise (PN) code that adds some pseudo-random time shift to the transmission time of each pulse. Additionally, applying a PN code to a pulse train has a collateral effect than can be useful in a multi user environment. Thus, selecting different codes to apply to the pulses of each users makes their communications almost invisible for the others.

One of the most common transmission technique in UWB impulse radio uses several of the concepts described previously. It is the so called time-hopping PPM [WS00]. This technique

transmits a pulse train that is defined as:

$$s(t) = \sum_{n=-\infty}^{\infty} p(t - nT_f) \quad (\text{A.6})$$

where $p(n)$ is the chosen pulse shape and T_f is the so called frame time. A single pulse is transmitted on each frame, so it can be defined the pulse repetition frequency pulse repetition frequency (PRF) as:

$$\text{PRF} = \frac{1}{T_f} \quad (\text{A.7})$$

The first step in the time-hopping PPM is adding a small shift to the pulse time position, following the basis of the PPM modulation. Thus, the pulse train would look as:

$$s(t) = \sum_{n=-\infty}^{\infty} p(t - nT_f - T_{p_n}) \quad (\text{A.8})$$

with T_{p_n} the time shift due to the PPM modulation. Finally, another time shift is added based on a time-hopping code T_{c_n} that is repeated periodically. This values could be created using a PN code, as showed before. Thus, the final pulse train would be the next:

$$s(t) = \sum_{n=-\infty}^{\infty} p(t - nT_f - T_{p_n} - T_{c_n}) \quad (\text{A.9})$$

In the reception side, the common procedure includes the next steps: detection, pulse integration and tracking. The first one tries to detected the transmitted pulses, and for this task it is commonly used a *correlator*. The idea is to multiply the received RF signal by a *template pulse shape* and integrate the result to obtain a peak in the correct position. This process must be extremely fast to find the right instant in systems such as UWB impulse radio, were the duration of the pulses is so short.

The next step, the pulse integration, adds several correlator samples corresponding to different pulses to maximize the confidence in the detection of the signal.

Finally, the tracking is the process that corrects the time drift of the pulses, due to the differences that can appear between the clocks of the emitter and the receiver during the transmission duration.

A.1.2 802.15.4-2011 UWB Physical layer

In this section are presented the specific characteristics of the UWB PHY defined in the standard IEEE 802.15.4-2011. The waveform is based on the UWB impulse radio scheme described in the previous section.

The standard defines three different frequency bands of operation: the sub-gigahertz band, the low band and the high band. The first one occupies the range 250 MHz to 750 MHz. The

Channel number	Center frequency, f_c , (MHz)	Band width (MHz)	Mandatory/Optional
0	499.2	499.2	Mandatory in sub-gigahertz band
1	3494.4	499.2	Optional
2	3993.6	499.2	Optional
3	4492.8	499.2	Mandatory in low band
4	3993.6	1331.2	Optional
5	6489.6	499.2	Optional
6	6988.8	499.2	Optional
7	6489.6	1081.6	Optional
8	7488.0	499.2	Optional
9	7987.2	499.2	Mandatory in high band
10	8486.4	499.2	Optional
11	7987.2	1331.2	Optional
12	8985.6	499.2	Optional
13	9484.8	499.2	Optional
14	9984.0	499.2	Optional
15	9484.8	1354.97	Optional

Table A.1: UWB PHY Channel definitions in IEEE 802.15.4-2011.

second one is placed in 3244 MHz to 4742 MHz and finally the last one occupies the spectrum from 5944 MHz to 10 234 MHz. Each of this bands are divided in 16 channels, that can be seen in Table A.1.

The standard defines that a compliant device should implement at least one of the mandatory channels, that is, channel 0 in sub-gigahertz band, channel 3 in low band or channel 9 in high band. As can be seen in Table A.1, all the channels occupy a 500 MHz bandwidth except the channels 4, 7, 11 and 15. These are optional channels thought to provide a longer communication range and an improvement on the multi-path resistance and the ranging accuracy. In the other hand, the power requirements are also higher.

The IEEE 802.15.4-2011 communications are base in the definition of frame showed in Fig. A.1. The frame has three parts: a synchronization header (SHR), a physical layer header (PHR) and the data part.

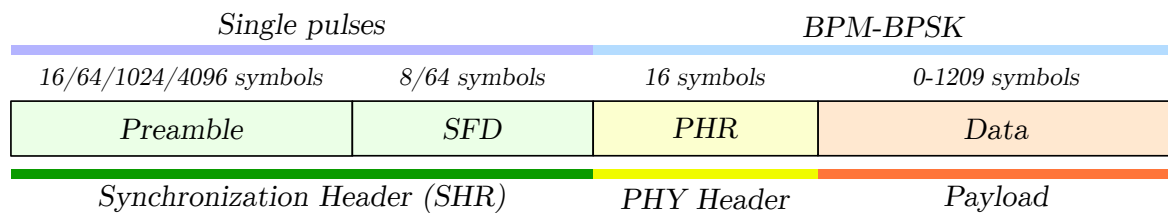


Figure A.1: UWB PHY Frame

The SHR mission is to serve as a mechanism to detect the frame in reception and noting

Index	Code sequence	Channel number
1	-0000+0-0+++0+-000+---00-+0-00	0, 1, 8, 12
2	0+0+-0+0+000-++0-+—00+00++000	0, 1, 8, 12
3	-+0++000-+-++00++0+00-0000-0+0-	2, 5, 9, 13
4	0000+-00-00-++++0+-+000+0-0++0-	2, 5, 9, 13
5	-0+-00+++++000-+0+++0-0+0000-00	3, 6, 10, 14
6	++00+00—+-0+++000+0+0-+0+0000	3, 6, 10, 14
7	+0000+-0+0+00+000+0++—0-+00-+	4, 7, 11, 15
8	0+00-0-0++0000—+00-+0++-++0+00	4, 7, 11, 15

Table A.2: Length 31 preamble codes.

its arriving timestamp. Can be sub-divided in two parts: the preamble and the start-of-frame delimiter (SFD). This last part is the one that marks the end of the preamble and the beginning of the PHY header. Its importance for the *ranging* process is critical, because it is used to set the precise frame reception timestamp. The standard defines two possible start of frame delimiter (SFD) sizes, one mandatory of 8 symbols and other optional made of 64 symbols for the configuration with the low data rate of 110 kbit/s.

The preamble in the SHR part of the frame is used for detection and synchronization purposes. The length of the preamble is variable, and depending of the final application one or other length should be used. The standard defines four options: 16, 64, 1024 and 4096 symbols. A device that complies the standard must support at least one of the previous preamble lengths.

Both in the preamble and in the SFD, the symbols transmitted are composed by a series of pulses that are transmitted at a fixed time interval following a preamble code. The preamble code is a sequence of values from the ternary alphabet $-1, 0, 1$, where -1 means sending a negative pulse, 1 means sending a positive one a 0 means that no pulse is transmitted in that time interval. These preamble codes are created to have a perfect autocorrelation property. That means that when making the correlation with the same sequence en reception, only the perfect align should give values different from zero. The standard defines 8 different length-31 codes, of which at least 2 must be supported by any device that complies with the standard. Table A.2 shows the list of this codes. Additionally, a optional set of codes with length-127 is also defined.

After the synchronization header, the UWB PHY frame appends the PHR. The mission of this fragment is to give the necessary information for the receiver to decode the data part of the frame. Specifically, this field includes information about the data rate used to send the payload and its length. The PHR has a fixed length of 16 symbols.

The payload can include a variable number of symbols, between 0-1209. The specific length is include inside the PHR, as well as the data rate used. As previously mentioned, the standard defines four options: 110 kbit/s, 851 kbit/s, 6.81 Mbit/s and 27.24 Mbit/s.

Unlike the synchronization header, both the payload and the PHR are encoded using the a combination of BPM and binary phase shift keying (BPSK) modulations. With this method

each symbol is split in four quarters, where the 2 and 4 quarters are used as guard intervals and the 1 and 3 are used to transmit the burst of pulses depending of the value of the bit to send (with 0 the burst is sent in the first quarter and with 1 is sent in the third). Additionally, another bit is used to change the phase of the pulses, following the BPSK modulation.

As far as ranging is concerned, 802.11.4-2011 defines a mandatory implementation protocol for any device that wants to comply with the standard. This is the so call two-way ranging (TWR). A visual representation of this protocol is showed in Fig. A.2.

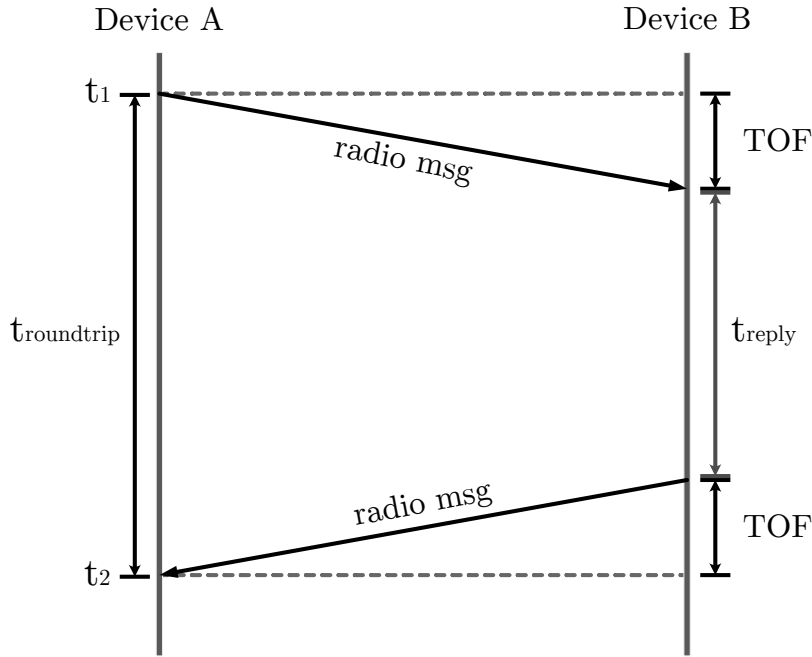


Figure A.2: Two way ranging.

The protocol starts with a first message sent by device A to device B . The emission timestamp of this message t_1 is stored in A . Once B receives the message, it waits a time t_{reply} (this time is also known by A) and the sends another message to A . Once A receives this message it already has everything it needs to calculate the time of flight (TOF):

$$TOF = \frac{(t_{roundtrip} - t_{reply})}{2} = \frac{(t_2 - t_1 - t_{reply})}{2} \quad (A.10)$$

A.1.3 802.15.4-2011 UWB MAC layer

The 802.15.4-2011 standard uses the same MAC definitions as in the original 802.15.4 version, i.e. it supports all the topologies included in it. The 802.15.4 standard defined two classes of devices depending on their MAC computing capabilities. Thus, simple devices (such as sensors) with limited capabilities are modeled as reduced functional devices (RFD)s, while other devices with larger network capabilities are modeled as full functional devices (FFD)s.

These two types of devices are organized at topological level in a personal area network (PAN), where each individual network is called a *piconet*. There are different network topologies defined in the 802.15.4 standard, such as the star topology where a single central node is responsible for distributing messages or the *peer-to-peer* topology where FFD devices can communicate with other nearby FFDs but RFDs can only do so with the network coordinator. The standard does not define the mandatory use of any specific network scheme, and is left to the final application designer.

As for the mechanism for accessing the medium, however, there is an important difference between standard 802.15.4 and 802.15.4-2011. While 802.15.4-2011 also describes the methods defined in 802.15.4 based on carrier sense multiple access with collision avoidance (CSMA/CA), 802.15.4-2011 (and before 802.15.4a) includes the Aloha protocol for the first time.

Its inclusion was motivated by the fact that, due to the robustness offered by UWB impulse radio to communication collisions (pulses have a very short duration), it was thought that a protocol such as Aloha could work in cases where the density of nodes in the network were medium or light. With the Aloha protocol the network nodes simply initiate a transmission assuming that the medium is free, and in case of collision they wait a random time to try it again. With a scheme like this, if the network usage time exceeds 18% then the risk of collision is very high and the performance of the system degrades rapidly. Below this percentage the chances of having a collision-free communication are above 95%.

A.2 UWB regulation in United States and European Union

UWB regulation in the United States is mainly in charge of the FCC, whereas in the European Union is the European Telecommunications Standards Institute (ETSI) the organization in charge of defining its limits. European legislation has always lagged a little behind that of the United States, where UWB technology has had a greater take-off.

The FCC regulates the use of UWB in different environments, being the applied to devices that operate in indoor environments that affects the location devices. The regulation is found specifically in the Code of Federal Regulations (CFR) Section 47 Part 15.

The FCC a UWB system as that with an absolute bandwidth larger than 500 MHz and f_c (central frequency) larger than 2.5 GHz, or have a $B_{frac} = \frac{B}{f_c}$ larger than 0.2 (where B is the bandwidth of the system) [BRE05]. It also defines a permitted range of frequencies of 3.1 GHz to 10.6 GHz.

The main restriction related with the transmission power is that the isotropic radiated power (EIRP) can not exceed the -41.3 dBm/MHz. Fig. A.3 shows the regulatory mask, where this limited is observed.

The situation in the European Union is a little different. The restrictions on UWB can be read in document ETSI EN 302 065[ETS14], UWB defines the requirements to be met by any

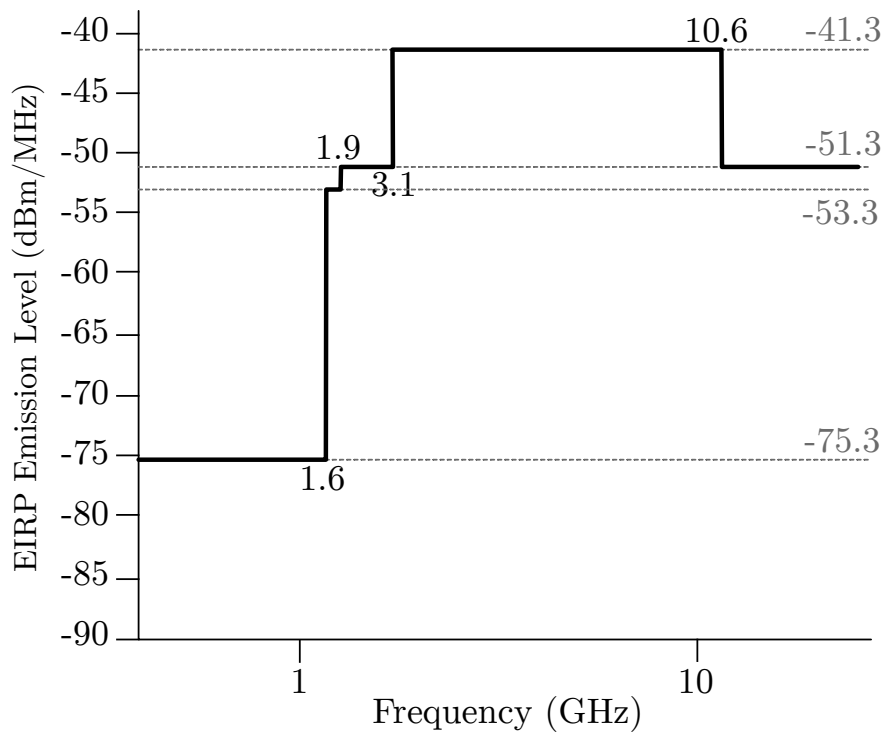


Figure A.3: Power Mask in US.

generic application based on UWB.

In this case the spectrum is divided into several bands, each with a different EIRP limitation. Fig. A.4 shows a summary of these bands and their power limits.

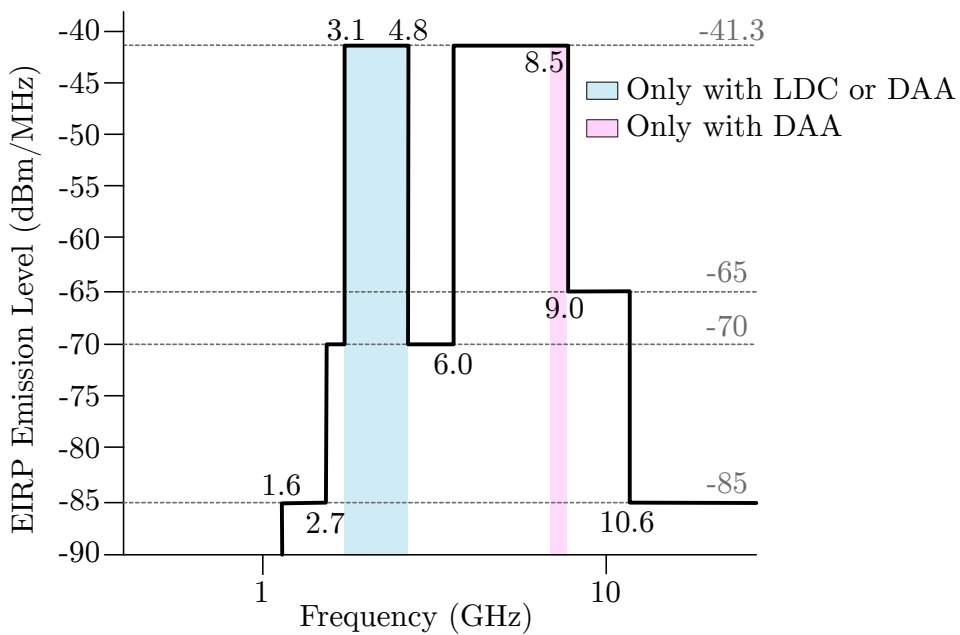


Figure A.4: Power Mask in UE.

One difference from regulation in the United States is the frequency bands 3.1 GHz to 4.8 GHz and 8.5 GHz to 9 GHz. In the first of these bands, a UWB device can only use that EIRP value (-41.3 dBm/MHz) if it implements one of the following two mitigation techniques: low duty cycle (LDC) or detect and avoid (DAA). The second band between 8.5 GHz to 9 GHz can only do the same if it implements DAA. If these interference mitigation techniques are not implemented, the maximum value of EIRP is -70.0 dBm/MHz and -65.0 dBm/MHz respectively.

LDC is a technique that limits the effective fraction of time the device can use to transmit. Table A.3 shows a summary of ETSI limits.

Parameter	Symbol	Limit
Max transmitter on time	T_{on} max	5 ms
Mean transmitter off time	T_{off} mean	≥ 38 ms (averaged over 1 s)
Sum transmitter off time	$\sum T_{off}$	> 950 ms per second
Sum transmitter on time	$\sum T_{on}$	< 18 s per hour

Table A.3: LDC limits.

DAA is a different technique where each device is required to implement a mechanism to listen if there is any other transmission in progress prior to making their own.

Appendix II

DW1000 transceiver

The DW1000 [DEC19c] was the first UWB based commercial transceiver encapsulated in a single chip complying with the 802.15.4-2011 standard. Its use in recent years has become mainstream in the area of indoor location, and is the basis of many commercial positioning systems. In the research world has also had a big impact, and there are already hundreds of articles in which this chip is present in one way or another.

Following are details of the specific implementation of the 802.15.4-2011 standard made by Decawave in its DW1000. Section B.1 describes aspects related to the implementation of the physical layer UWB PHY while Section B.2 shows information about the implementation of the MAC layer.

B.1 DW1000 implementation of IEEE 802.15.4-2011 UWB Physical Layer

DW1000 complies the IEEE 802.15.4-2011 specification of the UWB PHY in all its mandatory requirements, but implements only a subset of the optional parts. Thus, Table B.1 shows the list of frequency bands supported by the module. There are 7 channels among which is channel 3, which is one of what the standard defines as mandatory. The DW1000 complies also with the mandatory data rates, implementing the options of 110 kbit/s, 851 kbit/s and 6.81 Mbit/s.

This DW1000 implements the mandatory TWR protocol [DEC04] to estimate the distance between the emitter and the receiver. However, the version implemented in by Decawave is different to the one shown in Fig. A.2. The problem is that with real hardware (real oscillators) the clocks of both devices can suffer some time drift, causing a big error in the estimation of the TOF. The DW1000 employs a different version of TWR that tries to overcome this problem. This version can be seen in Fig. B.1. In this version of the algorithm it is used an extra message to overcome the clock drift issues.

In the typical working mode, a DW1000 based tag is emitting periodically a ping signal and waiting for some response from some of the anchors placed on the scenario. In one of

UWB Channel Number	Centre Frequency (MHz)	Band (MHz)	Bandwidth (MHz)
1	3494.4	3244.8 – 3744	499.2
2	3993.6	3774 – 4243.2	499.2
3	4492.8	4243.2 – 4742.4	499.2
4	3993.6	3328 – 4659.2	1331.2
5	6489.6	6240 – 6739.2	499.2
7	6489.6	5980.3 – 6998.9	1081.6

Table B.1: DW1000 Allowed Channels

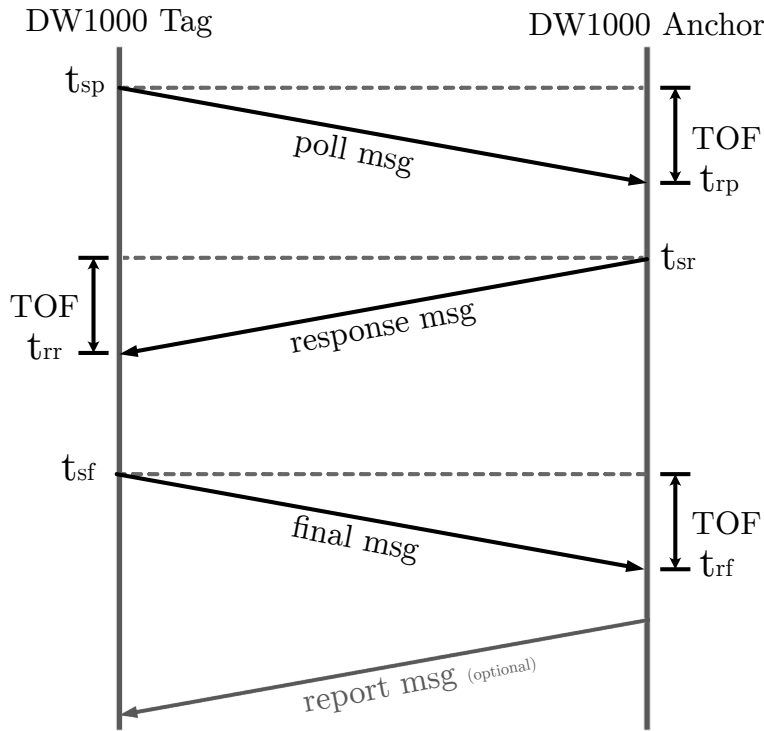


Figure B.1: TWR in DW1000.

these anchors responds, then the TWR protocol starts. Firstly, the tag sends a poll signal to the anchor and records the timestamp of this transmission t_{sp} . The anchor receives the signal and records the timestamp at the very first moment it detects the incoming signal t_{rp} . After that, the anchor sends a response message to the tag and stores the timestamp of transmission t_{sr} . The tag receives this response message, marks the reception timestamp t_{rr} and sends a final message to the anchor. This final message includes all the previous timestamp plus the last timestamp t_{sf} with the moment when this final transmission took place. After the reception of this last message, the anchor has a set of timestamps that allows it to calculate the TOF following the next expression:

$$TOF = \frac{(t_{rr} - t_{sp}) - (t_{sr} - t_{rp}) + (t_{rf} - t_{sr}) - (t_{sf} - t_{rr})}{4} \quad (\text{B.1})$$

To detect the main path (i.e., the one with shortest distance between the emitter and the receiver), the Decawave DW1000 uses a leading edge detection (LED) technique that relies in a predefined threshold to detect this first path [DAS+11]. The quality and accuracy of this detection can be affected by the physical propagation characteristics of the scenario and the actual location of the devices. Thus, in non-line-of-sight (NLOS) conditions, a secondary delayed path can be wrongly selected instead of the main one when its energy does not exceed the corresponding threshold value. Additionally, there are some other problems that can affect the overall accuracy of the ranging estimations, such as noise or bias [BAR+16b] depending on the distance between the devices.

B.2 DW1000 implementation of IEEE 802.15.4-2011 MAC Layer

The DW1000 does not have a implementation of the IEEE 802.15.4-2011 MAC Layer, due that this functionality is intended to be programmed by the host microcontroller unit (MCU) that controls the DW1000.

Thus, the only thing that the DW1000 includes in this respect are a series of features to make it easier from an external host to program the MAC layer of an end application.

These features are the following:

- **Cyclic redundancy check (CRC):** The DW1000 provides a mechanism for automatically calculating the CRC of a frame and for checking it when a frame is received. The result of the check is stored in one of the chip registers after reception, so a host implementing a MAC layer could query this value and decide what action perform (predictably, discarding the received frame in case the CRC was erroneous).
- **Frame filtering:** The DW1000 offers a frame filtering mechanism that allows to ignore those that do not comply with a series of rules. This mechanism allows that, although several frames are received over time, the host is only notified of the arrival of one if it complies with the rules set in the filter. The implementation of the DW1000 offers a large number of filtering rules, many related to MAC layer concepts defined in 802.15.4-2011.
- **Automatic acknowledgement (ACK):** Another mechanism implemented by the DW1000 is the automatic ACK. This functionality allows to configure the chip in such a way that, after receiving a correct ACK request (this is a request defined in the standard 802.15.4-2011), it responds automatically with the requested ACK without the need of being commanded from the host.
- **Wait for response after transmission:** The DW1000 has another functionality through which the chip can be configured to wait for a response just after a transmission. This automatically activates the receiver of the module after transmission (a small delay can also be configured between the end of the transmission and the moment when the receiver

is activated).

Although the DW1000 does not implement the MAC layer defined in 802.15.14-2011, it does implement one of the media access protocols defined in it, the Aloha protocol.

As it was commented during the description of the standard, the devices that follow this protocol simply start their transmission assuming that the channel is free, and only in case of collision they wait a random time to try it again.

The degree of occupation of the channel marks whether this protocol is sufficient to handle the access to the medium. Specifically, below 18% occupancy the Aloha protocol guarantees a collision level below 3%.

Bearing this in mind, the number of transmissions that can be made per second in a network of DW1000 nodes depends on their configuration. Low data rates and long preambles need more time to transmit, so the channel is occupied longer. On the other hand, high data rates and short preambles create messages that need less time to be sent and therefore the use of the channel is lower. Table B.2 shows a comparison among different configuration modes and the number of transmission that can be made per second while maintaining the percentage of the channel use under the 18%.

Data rate	Preamble length	Payload	Transmission Time	TX per second at 18% air-utilization
110 kbit/s	2048 symbols	12 B	3.042 ms	59.2
850 kbit/s	256 symbols	12 B	380.3 μ s	473.4
6.8 Mbit/s	64 symbols	12 B	103.3 μ s	1742

Table B.2: TX per second at 18% air-utilization

In this table it can be seen how the highest rate with the minimum preamble length allows the maximum number of transmission under the 18% air-utilization. An important question, however, is that because the operating range is affected also by the data rate and preamble length, a system with the previous configuration would have a shorter range. In the other hand, a configuration as the one showed in the first row of Table B.2 would have a longer operating range but could handle a lower number of devices.

Appendix III

Resumen de la tesis

Los sistemas de localización en interiores son múltiples y variados. Diferentes tecnologías y aproximaciones son utilizadas en diferentes soluciones para obtener estimaciones de posición de objetos y personas. Debido a las complejas situaciones que se dan en interiores (múltiples obstáculos, condiciones ambientales cambiantes, interferencias electromagnéticas, etc.) no existe una única situación capaz de dar respuesta a todos los problemas en todos los escenarios.

Desde el punto de vista organizativo, los sistemas de localización en interiores podrían categorizarse en dos tipos: los basados en redes de sensores y los autónomos. Los primeros necesitan de una estructura de dispositivos desplegados por el escenario en posiciones fijas y conocidas y un dispositivo (normalmente conocido como *tag* o *etiqueta*) que debe portar el objeto o persona a localizar. Los elementos fijos en el escenario se suelen definir como *balizas* o nodos *ancla*. Normalmente en esta configuración se utiliza algún tipo de señal de RF para medir algún tipo de parámetro físico entre las balizas y el *tag*. Entre los parámetros más habituales se encuentran el time of arrival (TOA), el time difference of arrival (TDOA), el angle of arrival (AOA), o el angle of departure (AOD). El primero de ellos sería el tiempo que tarda una señal en llegar desde cada baliza al *tag*, el segundo sería la diferencia de tiempos de llegada a dos balizas de una misma señal emitida desde un *tag*, y el tercero y cuarto serían el ángulo de incidencia de la señal emitida por un *tag* al llegar a una baliza y viceversa respectivamente. Para obtener estos parámetros se han utilizado y se siguen utilizando numerosas tecnologías RF diferentes, muchas de ellas sin un origen específico para este tipo de labores: Bluetooth, Wi-Fi, Zigbee, RFID, cámaras de video, señales de infrarrojos, ultrasonidos, láser... Con todas ellas se puede obtener una localización mas o menos precisa, mas o menos robusta.

La otra aproximación habitual para la localización en interiores es la autónoma, en la que el propio *tag* es el encargado de localizarse a si mismo en base a los datos obtenidos por diferentes sensores equipados en el mismo. Sensores como los inerciales o magnéticos pueden ser utilizado en este modo, pero también algunas tecnologías como las nombradas anteriormente (WiFi, Bluetooth, etc.) pueden ser utilizadas. En este caso las balizas no están en puntos conocidos pero el *tag* puede detectar la intensidad de las señales a su alrededor y compararlas con un mapa pre-existente de valores (esta técnica se conoce como *fingerprint*). Por supuesto

tanto la versión basada en balizas como la autónoma se puede mezclar para generar sistemas de localización híbridos.

Una de las tecnologías más importantes para la localización en interiores de los últimos años es la tecnología UWB. UWB es una tecnología radio que se caracteriza por usar un gran ancho de banda en frecuencia. Se basa en el envío de pequeños pulsos temporales de muy pequeña duración (del orden de nanosegundos) que se pueden enviar directamente en banda base o usando una señal portadora. Debido a este uso de pulsos de duración tan corta es posible disponer una resolución temporal muy alta a la hora de medir el tiempo. Esto permite medir el TOF de una señal desde que se emite desde un dispositivo hasta que se recibe en otro. Mediante esta medida precisa de tiempo es posible obtener una estimación de distancia entre dispositivos. Esto, extendido a un sistema de sensores con un *tag* UWB y balizas colocadas en posiciones conocidas, permite mediante trilateración conocer la posición del *tag*.

Por sus características físicas las señales UWB son muy resistentes al multi-trayecto, ya que con unos pulsos tan extremadamente cortos y los intervalos de guarda entre pulso y pulso hacen que sea muy difícil interferir la señal directa (el primer *path*) con alguno de los rebotes. Es por eso que en situaciones de buena línea de visión directa (line-of-sight (LOS)) un sistema de localización basado en nodos UWB puede obtener errores medios cercanos a tan solo 10 cm.

El problema aparece cuando esta línea de visión directa no está tan clara. Cuando existe algún obstáculo que impide la visión directa entre el *tag* y una baliza decimos que ambos se encuentran en una situación de falta de visión directa (NLOS). En esta situación las balizas UWB pueden interpretar un rebote de la señal como el primer *path* de la misma, de forma que su estimación de tiempo será errónea y siempre superior a la duración real. Esto hace que en estos escenarios el error de un sistema de posicionamiento basado en UWB pueda dispararse.

En un despliegue real sin embargo, mantener una situación continua de LOS puede ser imposible. Mas allá de pequeños despliegues en laboratorio, en la mayor parte de escenarios habituales para un sistema de localización (fábricas, hospitales, centros comerciales, edificios públicos, etc.) es muy complicado colocar las balizas de forma que nunca se produzca una situación de NLOS, aunque esta sea temporal.

El principal objetivo del trabajo que se presenta en esta memoria es el de intentar aportar algún tipo de solución para poder trabajar con un sistema de localización basado en UWB en situaciones en las que no existe una situación de LOS. Para ello se propone una solución basada en algoritmos de aprendizaje automático con los que intentar detectar y mitigar una medida UWB proveniente de un escenario NLOS.

Todo este trabajo ha sido realizado en todo momento siguiendo un enfoque práctico. Esto quiere decir que siempre se han utilizado dispositivos reales en escenarios reales. En concreto, para todos los experimentos se han utilizado unos dispositivos de localización UWB de bajo coste denominados Pozyx. La idea detrás de esto fue la de intentar mejorar la precisión y robustez de UWB utilizando los dispositivos que previsiblemente podrían ser utilizados en un despliegue real.

Además del análisis sobre la posible mejora de los datos de localización usando técnicas de aprendizaje automático, el presente trabajo incluye el desarrollo de un simulador software capaz de simular un sistema de localización basado en UWB. Dicho sistema es aplicado sobre diversos escenarios virtuales y sus resultados comparados con los obtenidos en los mismos escenarios del mundo real.

Por último, este trabajo presenta también la última de las patas necesaria para integrar los datos de posicionamiento dentro de la operativa de trabajo de cualquier entidad o empresa: una plataforma de localización en tiempo real (real-time location system (RTLS)) con soporte para cualquier tecnología de posicionamiento. Esta plataforma, validada en diversos proyectos de localización reales, se muestra tanto en su primera concepción como en su evolución tras la sucesivas mejoras que ha sido sufriendo en los últimos años.

C.1 Clasificación y mitigación de medidas UWB usando hardware de bajo coste y algoritmos de machine learning

La parte principal de este trabajo se centra en describir una propuesta para intentar mejorar el rendimiento de sistema de localización basados en UWB cuando estos se encuentran en situaciones de NLOS. Durante este trabajo, se han definido tres escenarios posibles en relación al LOS:

- Escenario LOS. En este caso, no existen obstáculos entre el emisor y el receptor y la distancia que los separa está dentro de los márgenes que garantiza una buena comunicación entre ambos.
- Escenario NLOS-*Soft* En este caso existe un obstáculo entre el emisor y el receptor, pero este no es suficiente para bloquear el primer *path* de la señal. El valor de la potencia recibida (received signal strength (RSS)) sí se ve afectada por el obstáculo, con respecto al escenario LOS.
- Escenario NLOS-*Hard* Aquí de nuevo existe un obstáculo entre el emisor y el receptor, pero en esta ocasión dicho obstáculo sí que es capaz de bloquear por completo o atenuar en tal grado el primer *path* de la señal que este no puede ser decodificado en el recepción. De esta forma, el receptor solo es capaz de decodificar un camino secundario de la señal, que siempre va a conllevar un TOF superior al original.

La solución propuesta para intentar mejorar el rendimiento se basa en los siguientes puntos:

1. Realizar campañas de medidas con hardware real para capturar valores en los tres escenarios descritos previamente.
2. Extraer una serie de características de interés de los datos sobre las que poder entrenar una serie de algoritmos de clasificación y mitigación.
3. Utilizar los algoritmos generados en la etapa anterior sobre las muestras procedentes de un despliegue de dispositivos UWB para intentar eliminar aquellas que los clasificadores

hayan clasificado como procedentes de un escenario NLOS.

4. Utilizar los mitigadores creados anteriormente para intentar corregir el error de las muestras resultantes de la fase de cribado con los clasificadores.
5. Utilizar las medidas para alimentar la los sistemas de posicionamiento finales.

Durante este trabajo se han realizado diversas campañas de medidas. En todas ellas el procedimiento fue similar. Se colocaron en diversos puntos del escenario una serie de balizas fijas y se usó un trípode con varios *tags* colocados sobre él para tomar medidas a diferentes intervalos de distancia. Las balizas se colocaron de tal forma que los datos capturados por cada una de ellas se correspondían a alguna de las situaciones analizadas: LOS, NLOS *Soft* y NLOS *Hard*.

En todos estas campaña se utilizaron los módulos Pozyx. Estos módulos son unos dispositivos de bajo coste que incluyen el transceptor de UWB DW1000. Este chip fue uno de los primeros en implementar el estándar 802.15.4-2011.

Esta aproximación se describe a lo largo de la memoria en cuatro capítulos. A continuación se presenta un resumen de cada uno de ellos.

C.1.1 Capítulo 2: Selección de características de interés para Machine Learning

Una de las primeras tareas que se deben abordar a la hora de emplear sistemas de aprendizaje automático es la de seleccionar las características a emplear en el entrenamiento de los algoritmos. Dichas características deben contener la mayor cantidad de información disponible relacionada con el objetivo final que se pretende resolver. En este capítulo se presentan en detalle los trabajos realizados de cara a realizar esta selección de características de entre las disponibles en los dispositivos low-cost analizado. Un resumen de estos trabajo puede verse a continuación:

- Se definieron una serie de escenarios de acuerdo al tipo de relación espacial entre emisor y receptor.
- Se buscó un lugar dentro del edificio del Área Científica, en el campus de la universidad de A Coruña, donde poder realizar diferentes mediciones correspondientes a los escenarios previamente identificados.
- Una vez obtenidos los datos en bruto, se generaron diferentes estadísticos sobre ellos usando diferentes tamaños de ventana. Estos nuevos datos se incorporaron como conjuntos de entrenamiento, validación y prueba para una implementación del algoritmo support vector machine (SVM).
- Se generaron diversas ejecuciones del algoritmo empleando diferentes combinaciones de estadísticos, con el fin de encontrar aquella combinación que obtenía una mayor índice de éxito a la hora de clasificar las medidas de test.
- Por último, a la vista de los resultados se extrajeron las conclusiones pertinentes. En este

caso, se comprobó que utilizando solo la media del *ranging* y el RSS era suficiente para obtener buenos resultados de clasificación.

C.1.2 Capítulo 3: Análisis del impacto de NLOS en los resultados de localización

En este capítulo se detallaron los resultados obtenidos tras estudiar cómo la identificación de las medidas de *ranging* UWB procedentes de un escenario NLOS consigue mejorar el rendimiento de los algoritmos de localización que las usan.

Siguiendo con la idea original de implementar un sistema capaz de detectar y mitigar los errores derivados de la ausencia de línea de visión directa en dispositivos de localización de bajo coste que emplean tecnología UWB, este capítulo analizó cual sería el impacto que se obtendría finalmente en las estimaciones de posición en caso de llevar a cabo dicha implementación.

Para ello, se utilizaron las muestras obtenidas durante una nueva campaña de medidas *indoor* para construir un simulador de escenarios de localización virtuales. La idea de este simulador era la de poder generar estimaciones de *ranging* reales para cualquier punto dentro de un escenario virtual, de forma que estas mediciones se pudiesen pasar a un algoritmo de posicionamiento que realizaría la estimación final de posición. Estas medidas de *ranging* serían de clase LOS o NLOS dependiendo de una probabilidad establecida para cada ejecución de la simulación. Así, este simulador se puede configurar con los siguientes parámetros:

- Número de dispositivos UWB (balizas de referencia) presentes en el escenario.
- Probabilidad de cada dispositivo de generar una muestra correspondiente al escenario NLOS
- Puntos de una trayectoria o *waypoints* que el dispositivo móvil simulado seguiría durante la prueba.

Los resultados obtenidos confirmaron que el uso de información a priori sobre la naturaleza NLOS de algunas medidas UWB puede tener un impacto significativo en el resultado final de posicionamiento. Ya sea ignorando las medidas de este tipo o ajustando la covarianza del error, los datos mostrados en este capítulo demostraron que el beneficio final de la aproximación propuesta puede ser sustancial en términos de error medio de posicionamiento.

C.1.3 Capítulo 4: Detección y mitigación NLOS

En este capítulo se detallan los trabajos realizados para la implementación y prueba de un mecanismo de clasificación y mitigación de medidas UWB obtenidas en un escenario NLOS usando técnicas de machine learning (ML) y hardware *low-cost*.

Tras los resultados descritos en los capítulos anteriores, en donde se confirmaron las mejoras posibles en las estimaciones de localización al detectar las medidas UWB procedentes de un entorno NLOS, el siguiente paso era el de construir un sistema clasificador y mitigador basado en técnicas de ML y probarlo con medidas reales.

En este capítulo se detallan las diferentes tareas llevadas a cabo para construir este sistema. Se realizó una nueva campaña de medidas con hardware real y sus resultados se utilizaron para implementar diversos clasificadores y mitigadores usando distintos algoritmos. Una parte de las muestras obtenidas durante la campaña de medidas se utilizó posteriormente para analizar el rendimiento de dichos elementos en diferentes situaciones: intentando clasificar solo entre medidas LOS y NLOS o haciéndolo entre los tres tipos LOS, NLOS-*Soft* y NLOS-*Hard*.

Finalmente se compararon sus resultados tanto entre los distintos algoritmos implementados como entre las diferentes situaciones planteadas. La conclusión al finalizar este capítulo fue que especialmente para el caso con solo dos clases la clasificación (LOS y NLOS) y mitigación funcionaba con un índice de acierto considerable, reduciendo los valores de error medio del *ranging* por debajo de los originales.

C.1.4 Capítulo 5: Validación cruzada

En este capítulo se presentó un estudio sobre como la aplicación de modelos de clasificación y mitigación basados en técnicas de aprendizaje automático podían ser aplicados en escenarios indoor diferentes. Se buscó de esta forma obtener un análisis del rendimiento de dichos sistemas cuando su conjunto de entrenamiento había sido capturado en un escenario completamente diferente al usado posteriormente en la aplicación de los modelos.

En este capítulo se abordó por tanto la última fase del proceso, que es la de validar la propuesta en un entorno real. Existen diversos trabajos que han usado anteriormente este enfoque (modelos basados en aprendizaje automático y su impacto en la localización), pero todos ellos adolecían de una considerable limitación. Y es que en estos trabajos los modelos y las sucesivas pruebas de localización se obtenían siempre sobre el mismo escenario. Aunque este enfoque era válido para obtener los límites de la aproximación, dejaba en el aire el cómo de general era la solución. Es decir, si los modelos entrenados en un escenario podían ser aplicados sin modificación para operar sobre medidas de otro escenario.

En este capítulo se detallan los experimentos realizados para intentar analizar esta situación, esto es, entrenar los algoritmos de clasificación y mitigación en un escenario y analizar su impacto en la localización en otro escenario diferente. El trabajo se desarrolló en las siguientes fases:

- Realización de dos campañas de medida. Para obtener los conjuntos de entrenamiento y prueba, se tomaron medidas en dos escenarios indoor diferentes.
- Entrenamiento de los modelos. Tras la obtención de las medidas, se realizó la configuración y entrenamiento de diversos algoritmos de clasificación y mitigación, incluyendo entre ellos el multilayer Perceptron (MLP).
- Estimación de posición. Tras emplear los clasificadores y mitigadores obtenidos anteriormente sobre las medidas capturadas en el segundo escenario, se usaron sus resultados para alimentar distintos algoritmos de localización y generar estimaciones de

posición.

- Análisis de resultados. Por último, se compararon las posiciones estimadas con las posiciones reales en cada instante, de cara a analizar el error obtenido para cada una de las configuraciones contempladas.

Los resultados arrojaron que el proceso de clasificación era perfectamente exportable a escenarios diferentes al de entrenamiento, obteniéndose una gran mejora con respecto al error de posicionamiento. La mitigación, sin embargo, no consiguió mejorar los resultados originales, debido a las pequeñas diferencias existentes entre la relación *ranging*-RSS presentes en ambos escenarios.

C.2 Simulación de dispositivos UWB para localización

La segunda parte de esta memoria se centra en la descripción de una plataforma de simulación basada en Gazebo y Robot Operating System (ROS) sobre la que se construyó un simulador de alto nivel de dispositivos UWB.

El trabajo realizado en esta parte se describió en el capítulo 6. Para crear el simulador, se utilizaron las medidas reales capturadas en los anteriores experimentos para entrenar una serie de algoritmos ML de forma que para un valor de distancia real se obtuviese una estimación de medida simulada. Esta estimación dependía del escenario concreto existente entre la *tag* y la baliza simuladas. Así, si el escenario era LOS se utilizaba un modelo, si era NLOS-*Soft* otro y si era NLOS-*Hard* otro diferente.

Para saber que situación aplicar, se creó un modelo de rayos de forma que desde cada baliza se compruebe si había línea de visión directa con el *tag*. En caso de existir, se utilizaba el modelo de LOS. En caso contrario, si un obstáculo impide la visión directa entre ambos, se comprueba como de importante es esa obstrucción. En caso de que no superase un determinado umbral de opacidad, el modelo NLOS-*Soft* es aplicado. Por último, en caso de que el obstáculo bloquee por completo la línea de visión directa, se realiza una búsqueda discreta para intentar comprobar si algún rebote de la señal puede llegar hasta el *tag*. Para ello se aplicaba el modelo de rayos y se comprueban los rebotes contra el suelo o las paredes del escenario simulado. Si mediante un rebote la visión es posible, entonces se aplica el modelo NLOS-*Hard* sobre la distancia resultante.

El simulador se probó de diferentes maneras. Primero se realizó una copia del escenario en donde se habían tomado las medidas de entrenamiento, y se comprobó que los valores proporcionados por el simulador era similares especialmente para el caso LOS y NLOS-*Soft*. Se probó también a mas alto nivel replicando el escenario de validación utilizado en el Capítulo 5. En este caso se compararon los resultados finales de posicionamiento al aplicar las distintas configuraciones que se habían probado en su momento: con y sin ignorar los valores NLOS y con y sin mitigación.

Además, el simulador se aplicó sobre un problema real habitual en la localización en

interiores, la localización de palés dentro de una nave industrial. En este caso se optó por localizar las carretillas elevadoras en lugar de los palés en sí. Para ello se simularon diferentes escenarios y un vehículo equipado con distintos sensores: un *tag* UWB, sensores inerciales y un sensor óptico PX4 Flow. Los resultados de las simulación mostraron como los valores de error cometido disminuían al añadir mas sensores, y como la posición de las balizas UWB en configuraciones NLOS provocaba una gran desviación en las posiciones estimadas con respecto a las reales.

C.3 Plataforma RTLS multi-tecnología: diseño y evolución

En el último capítulo de la tesis se presentó una plataforma RTLS multi-tecnología. La plataforma es un método flexible que permite desacoplar el trabajo de los diferentes actores normalmente implicados en este tipo de sistema. Así, la plataforma se convierte en el nodo de unión entre las redes de sensores, los location based services (LBS) y clientes finales, los gestores de alertas y los creadores de algoritmos de posicionamiento. Todos ellos pueden realizar su labor sin afectar al resto de entidades.

La plataforma ha sido validada durante los últimos años mediante su uso en diferentes proyectos de localización con diversas empresas y sectores, utilizando distintas tecnologías de localización. Como consecuencia de esta iteración con la operativa real de dichas empresas, surgieron diferentes mejoras y cambios en la plataforma.

En este último capítulo se detalla tanto el planteamiento original de la plataforma como el uso que se le ha dado en estos proyectos y finalmente se detallan también todas las mejoras y cambios que ha sufrido en los últimos tiempos y que la han dotado todavía de mayor flexibilidad.

Appendix IV

List of Acronyms

ACK	acknowledgement
AOA	angle of arrival
AOD	angle of departure
AOI	area of interest
API	application programming interface
BLE	bluetooth low energy
BPM	burst position modulation
BPSK	binary phase shift keying
CIR	channel impulse response
CRC	Cyclic redundancy check
CSMA/CA	carrier sense multiple access with collision avoidance
CSS	chirp spread spectrum
DAA	detect and avoid
ECDF	empirical cumulative distribution function
EIRP	isotropic radiated power
ETSI	European Telecommunications Standards Institute
FCC	Federal Communications Commission
FFD	full functional devices
GLM	generalized linear model
GP	Gaussian process
HTTP	hypertext transfer protocol
JSON	javascript object notation
IC	integrated circuit
IR	infrared
IEKF	iterative extended Kalman filter
IMU	inertial measurement unit
k-NN	k-nearest neighbors
LBS	location based services

LDC	low duty cycle
LED	leading edge detection
LLS	linear least squares
LO	Local oscillator
LOS	line-of-sight
M2M	machine to machine
MAC	medium access control
MAE	mean absolute error
MCU	microcontroller unit
ML	machine learning
MLP	multilayer Perceptron
MQTT	MQ telemetry transport
NLOS	non-line-of-sight
NLS	nonlinear least squares
NN	neural networks
OOK	on-off keying
PAM	pulse amplitude modulation
PAN	personal area network
PHR	physical layer header
PHY	physical layer
PPM	pulse position modulation
PN	pseudo noise
PRF	pulse repetition frequency
PSD	power spectral density
QOS	quality of service
RF	radio-frequency
RFD	reduced functional devices
ROS	Robot Operating System
RSS	received signal strength
RTLS	real-time location system
SHR	synchronization header
SFD	start of frame delimiter
SVM	support vector machine
TCP	transmission control protocol
TDOA	time difference of arrival
TOA	time of arrival
TOF	time of flight
TWR	two-way ranging
UDP	user datagram protocol

URL uniform resource locator

UWB ultra-wideband

References

- [AC05] S Al-Jazzar and J Caffery Jr. “**New algorithms for NLOS identification**”. *Proc. of Proc. of the 14th IST Mobile and Wireless Communications Summit*. 2005.
- [ACH+09] K. Achutegui, L. Martino, J. Rodas, C.J. Escudero, and J. Míguez. “**A Multi-Model Particle Filtering Algorithm for Indoor Tracking of Mobile Terminals Using RSS Data**”. *Proc. of IEEE Control Applications (CCA) & Intelligent Control (ISIC)*. 2009, pp. 1702–1707.
DOI: 10.1109/CCA.2009.5280960.
- [ACH+10] K. Achutegui, J. Rodas, C.J. Escudero, and J. Míguez. “**A Model-Switching Sequential Monte Carlo Algorithm for Indoor Tracking with Experimental RSS**”. *Proc. of Indoor Positioning and Indoor Navigation (IPIN)*. 2010, pp. 1–8.
DOI: 10.1109/IPIN.2010.5648053.
- [ACH+11] K. Achutegui, J. Rodas, C. J. Escudero, and J. Míguez. “**Bayesian Filtering Methods for Target Tracking in Mixed Indoor/Outdoor Environments**”. *Proc. of ICST Mobile Lightweight Wireless Systems (MOBILIGHT)*. Vol. 81. 2011, pp. 169–185.
DOI: 10.1007/978-3-642-29479-2_13.
- [ACH+12] K. Achutegui, J. Míguez, J. Rodas, and C.J. Escudero. “**A Multi-Model Sequential Monte Carlo Methodology for Indoor Tracking: Algorithms and Experimental Results**”. *ELSEVIER Signal Processing*, vol. 92, no. 11, 2012, pp. 2594–2613.
DOI: 10.1016/J.SIGPRO.2012.03.017.
- [AM18] Fahad Al-homayani and Mohammad Mahoor. “**Improved indoor geomagnetic field fingerprinting for smartwatch localization using deep learning**”. *Proc. of 2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE. 2018, pp. 1–8.
- [ANY19] Anyplace. **Anyplace web site**. Available online: <https://anyplace.cs.ucy.ac.cy/>. Accessed Oct. 2019.
- [BAM17] Alexander Buyval, Ilya Afanasyev, and Evgeni Magid. “**Comparative analysis of ros-based monocular slam methods for indoor navigation**”. *Proc. of Ninth International Conference on Machine Vision (ICMV 2016)*. Vol. 10341. International Society for Optics and Photonics. 2017, 103411K.

- [BAN+19] A Banks, E Briggs, K Borgendale, and R Gupta. “**Mqtt version 5.0**”. *OASIS Standard*, 2019.
- [BAR+12] Valentín Barral, Javier Rodas, José A García-Naya, and Carlos J Escudero. “**A novel, scalable and distributed software architecture for software-defined radio with remote interaction**”. *Proc. of 2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 80–83.
- [BAR+16a] Valentín Barral, Pedro Suárez-Casal, Carlos J. Escudero, and José A. García-Naya. “**Assessment of UWB ranging bias in multipath environments**”. *Proc. of Proc. of 7th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Alcalá de Henares, Madrid, Spain, 2016, pp. 1–4. ISBN: 978-1-5090-2424-7.
- [BAR+16b] Valentín Barral, Pedro Suárez-Casal, Carlos Escudero, and José A. García-Naya. “**Assessment of UWB ranging bias in multipath environments**”. *Proc. of Proc. of the International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Alcalá de Henares, Spain, 2016.
- [BAR+16c] Paolo Barsocchi, Antonino Crivello, Davide La Rosa, and Filippo Palumbo. “**A multisource and multivariate dataset for indoor localization methods based on wlan and geo-magnetic field fingerprinting**”. *Proc. of 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE. 2016, pp. 1–8.
- [BAR+19a] Valentín Barral, Carlos J. Escudero, José A. García-Naya, and Roberto Maneiro-Catoira. “**Nlos identification and mitigation using low-cost uwb devices**”. *Sensors*, vol. 19, no. 16, 2019. ISSN: 1424-8220.
DOI: 10.3390/s19163464. Online access: <https://www.mdpi.com/1424-8220/19/16/3464>.
- [BAR+19b] Valentín Barral, Carlos J. Escudero, Pedro Suárez-Casal, and José A. García-Naya. “**Impact of nlos identification on uwb-based localization systems**”. *Proc. of 10th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. Pisa, Italy, 2019.
- [BAR+19c] Valentín Barral, Carlos Escudero, José García-Naya, and Pedro Suárez-Casal. “**Environmental cross-validation of nlos machine learning classification/mitigation with low-cost uwb positioning systems**”. *Sensors*, 2019. Under second revision.
- [BAR+19d] Valentín Barral, Pedro Suárez-Casal, Carlos J. Escudero, and José A. García-Naya. “**Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments**”. *Electronics*, vol. 8, no. 10, 2019.
DOI: 10.3390/electronics8101152. Online access: <https://www.mdpi.com/2079-9292/8/10/1152>.
- [BAR19a] Valentin Barral. **Environment cross validation of NLOS machine learning classification/mitigation in low-cost UWB positioning systems - dataset**. 2019.
DOI: 10.21227/rhhs-fw33. Online access: <http://dx.doi.org/10.21227/rhhs-fw33>.

- [BAR19b] Valentin Barral. **Gazebo 2 ros source code repository**. Available online: <https://github.com/valentinbarral/gazebo2ros>. Accessed Oct. 2019.
- [BAR19c] Valentin Barral. **Gazebo forklift simulation source code repository**. Available online: <https://github.com/valentinbarral/gazeboforkliftsimulation>. Accessed Oct. 2019.
- [BAR19d] Valentin Barral. **Gazebo sensors plugins source code repository**. Available online: <https://github.com/valentinbarral/gazebosensorplugins>. Accessed Oct. 2019.
- [BAR19e] Valentin Barral. **NLOS classification based on RSS and ranging statistics obtained from low-cost UWB devices - dataset**. 2019.
DOI: 10.21227/swz9-y281. Online access: <http://dx.doi.org/10.21227/swz9-y281>.
- [BAR19f] Valentin Barral. **Ros kfpos source code repository**. Available online: <https://github.com/valentinbarral/roskfpos>. Accessed Oct. 2019.
- [BAR19g] Valentin Barral. **Ros logs scripts source code repository**. Available online: <https://github.com/valentinbarral/roslogscripts>. Accessed Oct. 2019.
- [BAR19h] Valentin Barral. **Ros msgs source code repository**. Available online: <https://github.com/valentinbarral/rosmsgs>. Accessed Oct. 2019.
- [BAR19i] Valentin Barral. **Ros toa source code repository**. Available online: <https://github.com/valentinbarral/rostoa>. Accessed Oct. 2019.
- [BAR19j] Valentin Barral. **Ros uwb reader source code repository**. Available online: <https://github.com/valentinbarral/rosuwbreader>. Accessed Oct. 2019.
- [BAR19k] Valentín Barral. **Multi-sensor accurate forklift location and tracking simulation in industrial indoor environments - dataset**. 2019.
DOI: 10.21227/b9tf-fv74. Online access: <http://dx.doi.org/10.21227/b9tf-fv74>.
- [BEG19] Valentín Barral, Carlos J. Escudero, and José A. García-Naya. “**Nlos classification based on rss and ranging statistics obtained from low-cost uwb devices**”. *Proc. of 27th European Signal Processing Conference (EUSIPCO)*. A Coruña, Spain, 2019.
- [BER+18] Rafael Berkvens, Frederik Smolders, Ben Bellekens, Michiel Aernouts, and Maarten Weyn. “**Comparing 433 and 868 mhz active rfid for indoor localization using multi-wall model**”. *Proc. of 2018 8th International Conference on Localization and GNSS (ICL-GNSS)*. IEEE. 2018, pp. 1–6.
- [BET91] Bruno Betrò. “**Bayesian methods in global optimization**”. *Journal of Global Optimization*, vol. 1, no. 1, 1991, pp. 1–14.
- [BG14] Andrew Banks and Rahul Gupta. “**Mqtt version 3.1. 1**”. *OASIS standard*, vol. 29, 2014, p. 89.
- [BHM98] Joan Borras, Paul Hatrack, and Narayan B Mandayam. “**Decision theoretic framework for NLOS identification**”. *Proc. of Proc. of the 48th IEEE Vehicular Technology Conference (VTC)*. Vol. 2. 1998, pp. 1583–1587.

- [BOS19] BOSCH. **Bosh BNO055**. Available online: https://www.bosch-sensortec.com/bst/products/all_products/bno055. Accessed Oct. 2019.
- [BRE+84] L Breiman, JH Friedman, R Olshen, and CJ Stone. “**Large margin dags for multiclass classification**”. *Proc. of Classification and Regression Trees*. Wadsworth, 1984.
- [BRE05] Gary Breed. “**A summary of fcc rules for ultra wideband communications**”, 2005.
- [CAR+15] Daniel Carrillo, Victoria Moreno, Benito Úbeda, and Antonio Skarmeta. “**Magifinger: 3d magnetic fingerprints for indoor location**”. *Sensors*, vol. 15, no. 7, 2015, pp. 17168–17194.
- [CES19] CESGA. **Centro tecnológico de supercomputación de galicia web site**. Available online: <https://www.cesga.es/en/cesga>. Accessed Oct. 2019.
- [CRR18] Adrian Cosma, Ion Emilian Radoi, and Valentin Radu. **Camloc: pedestrian location detection from pose estimation on resource-constrained smart-cameras**. 2018. arXiv: 1812.11209 [cs.CV].
- [DAR+09] Davide Dardari, Andrea Conti, Ulric Ferner, Andrea Giorgetti, and Moe Z Win. “**Ranging with ultrawide bandwidth signals in multipath environments**”. *Proceedings of the IEEE*, vol. 97, no. 2, 2009, pp. 404–426.
- [DAS+11] Marzieh Dashti, Mir Ghoraiishi, Katsuyuki Haneda, and Jun-ichi Takada. “High-precision time-of-arrival estimation for uwb localizers in indoor multipath channels”. *Novel Applications of the UWB Technologies*. IntechOpen, 2011.
- [DEC+10] Nicolò Decarli, Davide Dardari, Sinan Gezici, and Antonio Alberto D’Amico. “**Los/nlos detection for uwb signals: a comparative study using experimental data**”. *Proc. of IEEE 5th International Symposium on Wireless Pervasive Computing 2010*. IEEE. 2010, pp. 169–173.
- [DEC04] J. Decuir. **Two way time transfer based ranging**. IEEE 802.15-04a documents 15-04-0573-00-004a. 2004.
- [DEC14] Decawave. *Sources of error in DW1000 based two-way ranging (TWR) schemes*. Decawave, 2014.
- [DEC19a] DecaWave. **Decawave evk1000 evaluation kit**. Available online: <https://www.decawave.com/product/evk1000-evaluation-kit/>. Accessed Oct. 2019.
- [DEC19b] DecaWave. **Decawave web site**. Available online: <http://www.decawave.com/>. Accessed Oct. 2019.
- [DEC19c] Decawave. **Decawave DW1000 IC**. Available online: <https://www.decawave.com/product/dw1000-radio-ic/>. Accessed Oct. 2019.

- [DRU+97] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. “**Support vector regression machines**”. *Proc. of Advances in neural information processing systems*. 1997, pp. 155–161.
- [ETS14] ETSI. “**302 065 V2. 1.1**”. *Short Range Devices (SRD) using Ultra Wide Band technology (UWB)*, 2014.
- [GAZ19a] Gazebo. **Gazebo building editor**. Available online: <http://gazebosim.org/blog/buildingeditor>. Accessed Oct. 2019.
- [GAZ19b] Gazebo. **Gazebo simulator**. Available online: <http://gazebosim.org>. Accessed Oct. 2019.
- [GBG16] Ruben Gomez-Ojeda, Jesus Briaes, and Javier Gonzalez-Jimenez. “**PI-svo: semi-direct monocular visual odometry by combining points and line segments**”. *Proc. of 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 4211–4216.
- [GEL74] A. Gelb. *Applied Optimal Estimation*. MIT Press, 1974. ISBN: 9780262570480.
- [GMK07] Mohammad Ghavami, Lachlan Michael, and Ryuji Kohno. *Ultra wideband signals and systems in communication engineering*. John Wiley & Sons, 2007.
- [GOL05] Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.
- [GQ16] Xingbin Ge and Zhiyi Qu. “**Optimization wifi indoor positioning knn algorithm location-based fingerprint**”. *Proc. of 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)*. IEEE. 2016, pp. 135–137.
- [GU+18] Fuqiang Gu, Kourosh Khoshelham, Chunyang Yu, and Jianga Shang. “**Accurate step length estimation for pedestrian dead reckoning localization using stacked autoencoders**”. *IEEE Transactions on Instrumentation and Measurement*, 2018.
- [GUR+17] Karthikeyan Gururaj, Anojh Kumaran Rajendra, Yang Song, Choi Look Law, and Guofa Cai. “**Real-time identification of nlos range measurements for enhanced uwb localization**”. *Proc. of 2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE. 2017, pp. 1–7.
- [GÜV+07] İsmail Güvenç, Chia-Chin Chong, Fujio Watanabe, and Hiroshi Inamura. “**NLOS identification and weighted least-squares localization for UWB systems using multipath channel statistics**”. *EURASIP Journal on Advances in Signal Processing*, 2007.
- [HAR61] Herman O Hartley. “**The modified gauss-newton method for the fitting of non-linear regression functions by least squares**”. *Technometrics*, vol. 3, no. 2, 1961, pp. 269–280.

- [HAS+15] Masoumeh Hasani, Jukka Talvitie, Lauri Sydanheimo, Elena-Simona Lohan, and Leena Ukkonen. “**Hybrid wlan-rfid indoor localization solution utilizing textile tag**”. *IEEE Antennas and Wireless Propagation Letters*, vol. 14, 2015, pp. 1358–1361.
- [HAU+10] Daniel Hauschildt, Jurgen Kemper, Nicolaj Kirchhof, Benedict Juretko, and Holger Linde. “**Real-time scene simulator for thermal infrared localization**”. *Proc. of Proceedings of the 2010 Winter Simulation Conference*. IEEE. 2010, pp. 879–890.
- [HBB02] J. Hightower, B. Brumitt, and G. Borriello. “**The location stack: a layered model for location in ubiquitous computing**”. *Proc. of Mobile Computing Systems and Applications, 2002. International Workshop on*. IEEE. 2002, pp. 22–28.
- [HC11] Chih-Ning Huang and Chia-Tai Chan. “**Zigbee-based indoor location system by k-nearest neighbor algorithm with weighted rssi**”. *Procedia Computer Science*, vol. 5, 2011, pp. 58–65.
- [HL04] Haibo Hu and Dik-Lun Lee. “**Semantic location modeling for location navigation in mobile environment**”. *Proc. of IEEE International Conference on Mobile Data Management, 2004. Proceedings. 2004*. IEEE. 2004, pp. 52–61.
- [HON+13] Dominik Honegger, Lorenz Meier, Petri Tanskanen, and Marc Pollefeys. “**An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications**”. *Proc. of Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 1736–1741.
- [HTJ16] Ngoc-Huynh Ho, Phuc Truong, and Gu-Min Jeong. “**Step-detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone**”. *Sensors*, vol. 16, no. 9, 2016, p. 1423.
- [HTS08] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. “**Mqtt-s—a publish/subscribe protocol for wireless sensor networks**”. *Proc. of 2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE’08)*. IEEE. 2008, pp. 791–798.
- [HUA+17] Lvwen Huang, Siyuan Chen, Jianfeng Zhang, Bang Cheng, and Mingqing Liu. “**Real-time motion tracking for indoor moving sphere objects with a lidar sensor**”. *Sensors*, vol. 17, no. 9, 2017, p. 1932.
- [IBE12] Héctor José Pérez Iglesias, Valentín Barral, and Carlos J Escudero. “**Indoor person localization system through rssi bluetooth fingerprinting**”. *Proc. of 2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 40–43.
- [IND19a] Indoor Atlas. **Indoor Atlas web site**. Available online: <https://www.indooratlas.com/>. Accessed Oct. 2019.
- [IND19b] Google Maps Indoor. **Google maps indoor web site**. Available online: <https://www.google.com/maps/about/partners/indoormap/>. Accessed Oct. 2019.

- [ISO19a] ISO. **ISO/IEC 19762-5:2008 - Automatic Identification and Data Capture (AIDC) techniques. Harmonized vocabulary. Part 5: Locating systems**. Available online: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50718. Accessed Oct. 2019.
- [ISO19b] ISO. **ISO/IEC 24730-1:2006 - Real-Time Locating Systems (RTLS). Part 1: Application Program Interface (API)**. Available online: http://www.iso.org/iso/catalogue_detail.htm?csnumber=38840. Accessed Oct. 2019.
- [JAR+17] Omar Jaradat, Irfan Sljivo, Ibrahim Habli, and Richard Hawkins. “**Challenges of safety assurance for industry 4.0**”. *Proc. of 2017 13th European Dependable Computing Conference (EDCC)*. IEEE. 2017, pp. 103–106.
- [JDW08] Damien B Jourdan, Davide Dardari, and Moe Z Win. “**Position error bound for uwb localization in dense cluttered environments**”. *IEEE transactions on aerospace and electronic systems*, vol. 44, no. 2, 2008, pp. 613–628.
- [JS16] Antonio Ramón Jiménez and Fernando Seco. “**Comparing decawave and bespoon uwb location systems: indoor/outdoor performance analysis**”. *Proc. of 2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE. 2016, pp. 1–8.
- [JU97] Simon J Julier and Jeffrey K Uhlmann. “**New extension of the kalman filter to nonlinear systems**”. *Proc. of Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. International Society for Optics and Photonics. 1997, pp. 182–194.
- [KAR+10] Eirini Karapistoli, Fotini-Niovi Pavlidou, Ioannis Gragopoulos, and Ioannis Tsetsinas. “**An overview of the ieee 802.15. 4a standard**”. *IEEE Communications Magazine*, vol. 48, no. 1, 2010, pp. 47–53.
- [KC15] Dong Ki Kim and Tsuhan Chen. “**Deep neural network for real-time autonomous indoor navigation**”. *arXiv preprint arXiv:1511.04668*, 2015.
- [KUM+16] Anil Kumar Tirumala Ravi Kumar, Bernd Schäufele, Daniel Becker, Oliver Sawade, and Ilja Radusch. “**Indoor localization of vehicles using deep learning**”. *Proc. of 2016 IEEE 17th international symposium on a world of wireless, mobile and multimedia networks (WoWMoM)*. IEEE. 2016, pp. 1–6.
- [KUM+17] GA Kumar, Ashok Patil, Rekha Patil, Seong Park, and Young Chai. “**A lidar and imu integrated indoor navigation system for uavs and its application in real-time pipeline classification**”. *Sensors*, vol. 17, no. 6, 2017, p. 1268.
- [KZP07] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. “**Supervised machine learning: a review of classification techniques**”. *Emerging artificial intelligence applications in computer engineering*, vol. 160, 2007, pp. 3–24.

- [LAR+10] Janire Larranaga, Leire Muguira, Juan-Manuel Lopez-Garde, and Juan-Ignacio Vazquez. “**An environment adaptive zigbee-based indoor positioning algorithm**”. *Proc. of 2010 International Conference on Indoor Positioning and Indoor Navigation*. IEEE. 2010, pp. 1–8.
- [LI+14] Rongbing Li, Jianye Liu, Ling Zhang, and Yijun Hang. “**Lidar/mems imu integrated navigation (slam) method for a small uav in indoor environments**”. *Proc. of 2014 DGON Inertial Sensors and Systems (ISS)*. IEEE. 2014, pp. 1–15.
- [LOC19] Localino. **Localino web site**. Available online: <https://www.localino.net/en/>. Accessed Oct. 2019.
- [LZZ13] Weijie Li, Tingting Zhang, and Qinyu Zhang. “**Experimental researches on an ubw nlos identification method based on machine learning**”. *Proc. of 2013 15th IEEE International Conference on Communication Technology*. IEEE. 2013, pp. 473–477.
- [MA+17] Zixiang Ma, Stefan Poslad, John Bigham, Xiaoshuai Zhang, and Liang Men. “**A ble rssi ranking based indoor positioning system for generic smartphones**”. *Proc. of 2017 Wireless Telecommunications Symposium (WTS)*. IEEE. 2017, pp. 1–8.
- [MAA+09] A. Maali, H. Mimoun, G. Baudoin, and A. Ouldali. “**A new low complexity NLOS identification approach based on UWB energy detection**”. *Proc. of Proc. of the IEEE Radio and Wireless Symposium*. 2009, pp. 675–678.
DOI: 10.1109/RWS.2009.4957442.
- [MAR+10] S. Marano, W. M. Gifford, H. Wymeersch, and M. Z. Win. “**NLOS identification and mitigation for localization based on UWB experimental data**”. *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 7, 2010, pp. 1026–1035. ISSN: 0733-8716.
DOI: 10.1109/JSAC.2010.100907.
- [MAR63] Donald W Marquardt. “**An algorithm for least-squares estimation of nonlinear parameters**”. *Journal of the society for Industrial and Applied Mathematics*, vol. 11, no. 2, 1963, pp. 431–441.
- [MEN+18] Germán Mendoza-Silva, Philipp Richter, Joaquín Torres-Sospedra, Elena Lohan, and Joaquín Huerta. “**Long-term wifi fingerprinting dataset for research on robust indoor positioning**”. *Data*, vol. 3, no. 1, 2018, p. 3.
- [MS00] Matthew D Mullin and Rahul Sukthankar. “**Complete cross-validation for nearest neighbor classifiers.**” *Proc. of ICML*. 2000, pp. 639–646.
- [MUR07] William S Murphy. “**Determination of a position using approximate distances and trilateration**”. *Colorado School of Mines*, 2007.

- [MUS+] Ardiansyah Musa, Gde Dharma Nugraha, Hyojeong Han, Deokjai Choi, Seongho Seo, and Juseok Kim. “**A decision tree-based nlos detection method for the uwb indoor location tracking accuracy improvement**”. *International Journal of Communication Systems*, e3997.
- [NLL17] Diederick C Niehorster, Li Li, and Markus Lappe. “**The accuracy and precision of position and orientation tracking in the htc vive virtual reality system for scientific research**”. *i-Perception*, vol. 8, no. 3, 2017, p. 2041669517708205.
- [NOR12] Abdelmoumen Norrdine. “**An algebraic solution to the multilateration problem**”. *Proc. of Proceedings of the 15th international conference on indoor positioning and indoor navigation, Sydney, Australia*. Vol. 1315. 2012.
- [NOR19] Nordic Semiconductor. **Nordic nRF52832**. Available online: <https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832>. Accessed Oct. 2019.
- [NW72] John Ashworth Nelder and Robert WM Wedderburn. “**Generalized linear models**”. *Journal of the Royal Statistical Society: Series A (General)*, vol. 135, no. 3, 1972, pp. 370–384.
- [NXP19] NXP. **Nxp MPL3115A2**. Available online: <https://www.nxp.com/part/MPL3115A2>. Accessed Oct. 2019.
- [OHI05] Ian Oppermann, Matti Hämäläinen, and Jari Inatti. ***UWB: theory and applications***. John Wiley & Sons, 2005.
- [OPE19] OpenID. **OpenID**. Available online: <http://openid.net/>. Accessed Oct. 2019.
- [PCS00] John C Platt, Nello Cristianini, and John Shawe-Taylor. “**Large margin dags for multiclass classification**”. *Proc. of Advances in neural information processing systems*. 2000, pp. 547–553.
- [POZ19a] Pozyx. **Pozyx registers overview**. Available online: <https://www.pozyx.io/product-info/developer-tag/datasheet-register-overview>. Accessed Oct. 2019.
- [POZ19b] Pozyx. **Pozyx web site**. Available online: <http://www.pozyx.com/>. Accessed Oct. 2019.
- [PUM+17] Albert Pumarola, Alexander Vakhitov, Antonio Agudo, Alberto Sanfeliu, and Francese Moreno-Noguer. “**Pl-slam: real-time monocular visual slam with points and lines**”. *Proc. of 2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 4503–4508.
- [PX419] Px4. **Px4 autopilot firmware**. Available online: <https://github.com/PX4/Firmware>. Accessed Oct. 2019.
- [QUU19] Quuppa. **Quuppa web site**. Available online: <https://quuppa.com/>. Accessed Oct. 2019.

- [RAN+18] Jesperi Rantanen, Laura Ruotsalainen, Martti Kirkko-Jaakkola, and Maija Mäkelä. “**Height measurement in seamless indoor/outdoor infrastructure-free navigation**”. *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 4, 2018, pp. 1199–1209.
- [RAP+96] Theodore S Rappaport et al. *Wireless communications: principles and practice*. Vol. 2. Prentice Hall PTR New Jersey, 1996.
- [RAS03] Carl Edward Rasmussen. “**Gaussian processes in machine learning**”. *Proc. of Summer School on Machine Learning*. Springer. 2003, pp. 63–71.
- [RBE13] Javier Rodas, Valentín Barral, and Carlos Escudero. “**Architecture for multi-technology real-time location systems**”. *Sensors*, vol. 13, no. 2, 2013, pp. 2220–2253.
- [RG17] Antonio Ramón Jiménez Ruiz and Fernando Seco Granja. “**Comparing ubisense, bespoon, and decawave uwb location systems: indoor performance analysis**”. *IEEE Transactions on instrumentation and Measurement*, vol. 66, no. 8, 2017, pp. 2106–2117.
- [RID+15] Mohamed Er Rida, Fuqiang Liu, Yassine Jadi, Amgad Ali Abdullah Algawhari, and Ahmed Askourih. “**Indoor location position based on bluetooth signal strength**”. *Proc. of 2015 2nd International Conference on Information Science and Control Engineering*. IEEE. 2015, pp. 769–773.
- [RJ15] S Ravindra and SN Jagadeesha. “**Time of arrival based localization in wireless sensor networks: a non-linear approach**”. *Signal & Image Processing*, vol. 6, no. 1, 2015, p. 45.
- [RN10] Carl Edward Rasmussen and Hannes Nickisch. “**Gaussian processes for machine learning (gpml) toolbox**”. *Journal of machine learning research*, vol. 11, no. Nov, 2010, pp. 3011–3015.
- [ROD+12] Javier Rodas, Valentin Barral, Carlos J Escudero, and Robert Langwieser. “**A solution for optimizing costs and improving diversity of rfid readers**”. *Proc. of 2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE. 2012, pp. 84–88.
- [ROD15] Javier Rodas. “Architecture for multi-technology real-time location systems”. PhD thesis. Universidade da Coruña, 2015. Online access: <http://hdl.handle.net/2183/15769>.
- [ROS19a] ROS. **Gazebo2rviz ros node**. Available online: <http://wiki.ros.org/gazebo2rviz>. Accessed Oct. 2019.
- [ROS19b] ROS. **Mavros ROS package**. Available online: <http://wiki.ros.org/mavros>. Accessed Oct. 2019.
- [ROS19c] ROS. **ROS**. Available online: <http://http://www.ros.org>. Accessed Oct. 2019.
- [ROS19d] ROS. **Teleop_twist_joy ros node**. Available online: http://wiki.ros.org/teleop_twist_joy. Accessed Oct. 2019.

- [ROS19e] ROS. **Turtlebot3_gazebo ros node**. Available online: http://wiki.ros.org/turtlebot3_gazebo. Accessed Oct. 2019.
- [SEV19] Sevenix. **Sevenix Ingeniería S.L.** Available online: <https://www.sevenix.es>. Accessed Oct. 2019.
- [SGD08] Mare Srbinovska, Cvetan Gavrovski, and Vladimir Dimcev. “**Localization estimation system using measurement of rssi based on zigbee standard**”. *Proc. of Conference Proceedings of the 17th International Scientific and Applied Science Conference (Electronics 2008)*. 2008, pp. 48–50.
- [SJ18] Fernando Seco and Antonio Jiménez. “**Smartphone-based cooperative indoor localization with rfid technology**”. *Sensors*, vol. 18, no. 1, 2018, p. 266.
- [SL09] Marina Sokolova and Guy Lapalme. “**A systematic analysis of performance measures for classification tasks**”. *Information Processing & Management*, vol. 45, no. 4, 2009, pp. 427–437.
- [SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. “**Practical bayesian optimization of machine learning algorithms**”. *Proc. of Advances in neural information processing systems*. 2012, pp. 2951–2959.
- [SN10] Samer S Saab and Zahi S Nakad. “**A standalone rfid indoor positioning system using passive tags**”. *IEEE Transactions on Industrial Electronics*, vol. 58, no. 5, 2010, pp. 1961–1970.
- [ST13] Andy Stanford-Clark and Hong Linh Truong. “**Mqtt for sensor networks (mqtt-sn) protocol specification**”, 2013.
- [SV04] J. Schiller and A. Voisard. *Location-based Services*. Elsevier, 2004.
- [TAY94] James D Taylor. *Introduction to ultra-wideband radar systems*. CRC press, 1994.
- [TBF05] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN: 0262201623.
- [TEC19] Tech Solutions. **Tech Solutions web site**. Available online: <https://www.techsolutions.co.za/rfid-pallet-tracking/>. Accessed Oct. 2019.
- [TER+17] Marco Terán, Juan Aranda, Henry Carrillo, Diego Mendez, and Carlos Parra. “**IoT-based system for indoor location using bluetooth low energy**”. *Proc. of 2017 IEEE Colombian Conference on Communications and Computing (COLCOM)*. IEEE. 2017, pp. 1–6.
- [THA+14] Dinesh Thangavel, Xiaoping Ma, Alvin Valera, Hwee-Xian Tan, and Colin Keng-Yan Tan. “**Performance evaluation of mqtt and coap via a common middleware**”. *Proc. of 2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP)*. IEEE. 2014, pp. 1–6.

- [TIA+15] Qinglin Tian, Zoran Salcic, I Kevin, Kai Wang, and Yun Pan. “**A multi-mode dead reckoning system for pedestrian tracking using smartphones**”. *IEEE Sensors Journal*, vol. 16, no. 7, 2015, pp. 2079–2093.
- [TOU19] Touch Path. **Touch Path web site**. Available online: <https://touchpath.com/>. Accessed Oct. 2019.
- [UDC19] UDC. **Area científica location**. Available online: <https://goo.gl/maps/59pCfNgZ75Siy6gf7>. Accessed Oct. 2019.
- [VAP95] Vladimir N Vapnik. “**The nature of statistical learning theory**”. *Proc. of The nature of statistical learning theory*. 1995.
- [VB07] S. Venkatesh and R. M. Buehrer. “**Non-line-of-sight identification in ultra-wideband systems based on received signal statistics**”. *IET Microwaves, Antennas and Propagation*, vol. 1, no. 6, 2007, pp. 1120–1130. ISSN: 1751-8725. DOI: 10.1049/iet-map:20060273.
- [VER19] Vero Solutions. **Vero Solutions web site**. Available online: <https://vero.solutions/warehouse-real-time-location-systems/>. Accessed Oct. 2019.
- [VWW05] Peter James Vial, Beata J. Wysocki, and Tadeusz A. Wysocki. “**An ultra wide band simulator using matlab/ simulink**”. *Proc. of 8th International Symposium on DSP and Communication Systems DSPCS'2005*. 2005, pp. 231–236.
- [WAN+18] Yun-Ting Wang, Chao-Chung Peng, Ankit Ravankar, and Abhijeet Ravankar. “**A single lidar-based feature fusion indoor localization algorithm**”. *Sensors*, vol. 18, no. 4, 2018, p. 1294.
- [WAN15] Yue Wang. “**Linear least squares localization in sensor networks**”. *Eurasip journal on wireless communications and networking*, vol. 2015, no. 1, 2015, p. 51.
- [WH07] Chin-Der Wann and Chih-Sheng Hsueh. “**NLOS mitigation with biased kalman filters for range estimation in UWB systems**”. *Proc. of Proc. of IEEE Region 10 Conference (TENCON)*. 2007, pp. 1–4. DOI: 10.1109/TENCON.2007.4429031.
- [WOL92]
Jacques Wolfmann. “**Almost perfect autocorrelation sequences**”. *IEEE Transactions on Information Theory*, vol. 38, no. 4, 1992, pp. 1412–1418.
- [WS00] Moe Z Win and Robert A Scholtz. “**Ultra-wide bandwidth time-hopping spread-spectrum impulse radio for wireless multiple-access communications**”. *IEEE Transactions on communications*, vol. 48, no. 4, 2000, pp. 679–689.

- [XIA+14] Zhuoling Xiao, Hongkai Wen, Andrew Markham, Niki Trigoni, Phil Blunsom, and Jeff Frolik. “**Non-line-of-sight identification and mitigation using received signal strength**”. *IEEE Transactions on Wireless Communications*, vol. 14, no. 3, 2014, pp. 1689–1702.
- [XIA+15] Hao Xia, Xiaogang Wang, Yanyou Qiao, Jun Jian, and Yuanfei Chang. “**Using multiple barometers to detect the floor location of smart phones with built-in barometric sensors for indoor positioning**”. *Sensors*, vol. 15, no. 4, 2015, pp. 7857–7877.
- [XML19] XML. **Extensible markup language (xml)**. Available online: <http://www.w3.org/XML/>. Accessed Oct. 2019.
- [XU+17] He Xu, Ye Ding, Peng Li, Ruchuan Wang, and Yizhu Li. “**An rfid indoor positioning algorithm based on bayesian probability and k-nearest neighbor**”. *Sensors*, vol. 17, no. 8, 2017, p. 1806.
- [YAM19] Yaml. **Yaml ain’t markup language**. Available online: <http://www.yaml.org>. Accessed Oct. 2019.
- [YAN+06] Wang Yang, Chen Peipei, Zhi Xinwei, Zhang Qinyu, and Zhang Naitong. “**Characterization of indoor ultra-wide band NLOS channel**”. *Proc. of Proc. of the IEEE Annual Wireless and Microwave Technology Conference*. IEEE. 2006, pp. 1–5.
- [YAS+17] Muneer Bani Yassein, Mohammed Q Shatnawi, Shadi Aljwarneh, and Razan Al-Hatmi. “**Internet of things: survey and open issues of mqtt protocol**”. *Proc. of 2017 International Conference on Engineering & MIS (ICEMIS)*. IEEE. 2017, pp. 1–6.
- [YE+11] Tingcong Ye, Michael Walsh, Peter Haigh, John Barton, and Brendan O’Flynn. “**Experimental impulse radio ieee 802.15. 4a uwb based wireless sensor localization technology: characterization, reliability and ranging**”. *Proc. of ISSC 2011, 22nd IET Irish Signals and Systems Conference, Dublin, Ireland. 23-24 Jun 2011*. Institution of Engineering and Technology. 2011.
- [YS15] Chouchang Yang and Huai-Rong Shao. “**Wifi-based indoor positioning**”. *IEEE Communications Magazine*, vol. 53, no. 3, 2015, pp. 150–157.
- [ZFB10] Piero Zappi, Elisabetta Farella, and Luca Benini. “**Tracking motion direction and distance with pyroelectric ir sensors**”. *IEEE Sensors Journal*, vol. 10, no. 9, 2010, pp. 1486–1494.
- [ZHA+18] Mingyang Zhang, Yingyou Wen, Jian Chen, Xiaotao Yang, Rui Gao, and Hong Zhao. “**Pedestrian dead-reckoning indoor localization based on os-elm**”. *IEEE Access*, vol. 6, 2018, pp. 6116–6129.