



MASTER OF SCIENCE
IN ENGINEERING

Hes·SO

Haute Ecole Spécialisée
de Suisse occidentale

Fachhochschule Westschweiz

University of Applied Sciences and Arts
Western Switzerland

Master of Science HES-SO in Engineering
Av. de Provence 6
CH-1007 Lausanne

Master of Science HES-SO in Engineering

Orientation : Technologies de l'information et de la communication (TIC)

AI-enhanced LoRa Based Indoor Localization System

Fait par

Manon Racine

Sous la direction de
Dr. Nuria Pazos
HE-Arc

St-Imier, HE-Arc, 9 janvier 2019

Résumé

Le défi d'avoir une localisation intérieure est de plus en plus présent dans la vie de tous les jours et devient d'une très grande utilité pour l'industrie, mais également pour effectuer de la surveillance de personnes par exemple. En extérieur, il existe une multitude de technologies le permettant (GPS, le Wifi, la vision, etc.) alors qu'en intérieur très peu de technologies ont fait leurs preuves.

L'objectif de ce projet d'approfondissement est d'étudier la possibilité d'amélioration de la précision du positionnement intérieur grâce à des algorithmes d'apprentissage tels que "support vector machine", "random forest", "K-nearest neighbour", etc. Le système de positionnement de départ est basée sur la technologie LoRa 2.4GHz intégrée dans un circuit radio SX1280 de chez Semtech.

Un logiciel d'apprentissage a été développé afin de tester différents scénarios par rapport à des mesures qui ont été prises dans le laboratoire de la HE-ARC à St-Imier. Ces différents scénarios comportent des points qui ont été mesurés aux mêmes endroits sans bouger l'espion (Sx - points de référence) et aussi aux mêmes endroits mais à des moments différents en ayant légèrement modifié la position (SxT - points de test) et également des points placés autour des points de référence (Tx - point pour l'évaluation des performances).

Les résultats obtenus sont encourageants concernant la classification, mais il serait nécessaire d'améliorer le système en intégrant une régression pour parvenir à localiser des points complètement inconnus. Il serait également nécessaire de retoucher le pré-traitement des mesures brutes ainsi que simplifier la procédure de récolte de données.

Abstract

The challenge of having an indoor location is more and more present in everyday life and becomes very useful for the industry, but also for monitoring people for example. Outside, there is a multitude of technologies allowing it (GPS, Wifi, vision, etc.) while inside very few technologies have proven themselves.

The aim of this project is to study the possibility of improving the accuracy of the indoor positioning thanks to machine learning algorithms such as "support vector machine", "random forest", "K-nearest neighbor" etc. The initial positioning system is based on the LoRa 2.4GHz technology integrated in Semtech's SX1280 radio circuit.

A machine learning software has been developed to test different scenarios with respect to measurements taken in the HE-ARC's laboratory in St-Imier. These different scenarios include points that have been measured in the same places without moving the spy (Sx - reference points) and also in the same places but at different times having slightly modified the position (SxT - test points) and also points placed around reference points (Tx - point for performance evaluation).

The results obtained are encouraging with regard to classification, but it would be necessary to improve the system by integrating a regression in order to locate completely unknown points. It would also be necessary to review the pre-processing of raw measurements as well as simplify the data collection procedure.

Remerciements

Dans un premier temps, je tiens à remercier chaleureusement Mme Nuria Pazos qui a suivi mon travail minutieusement et qui y a investi de son temps. Elle a été d'un soutien précieux avec toujours un mot d'encouragement positif.

Sans la collaboration de mon ancien collègue Michael Mueller, la reprise de sa thèse de Master aurait été bien plus compliquée. Il a été d'un grand soutien dans les premiers pas de ce projet.

Ensuite, j'aimerais dire un tout grand merci à mes collègues qui m'ont aidé de près ou de loin avec leurs conseils et leur partage d'idées. Un merci particulier à Arnaud Gay Des Combes et Olivier Gloriod qui m'ont apporté un soutien considérable dans le langage Python.

Un merci également à notre très cher concierge Pierre pour avoir laissé mon installation en place durant de longues semaines malgré les dérangements occasionnés.

Pour terminer avec une note d'humour, je tiens à remercier mon chat qui m'a servi de bouillotte durant mes longues soirées de rédaction.

Table des matières

Résumé / Abstract	iii
Remerciements	v
1. Introduction	1
1.1. Contexte	1
1.2. Aspect novateur	2
1.3. Structure du rapport	2
2. État de l'art	3
2.1. Classification des algorithmes d'apprentissage	3
2.2. Algorithme de classification	4
2.2.1. Decision Tree	4
2.2.2. Naïve Bayes	4
2.2.3. Bayesian Network	4
2.2.4. K-Nearest Neighbor	5
2.2.5. SMO / SVM	5
2.2.6. AdaBoost	5
2.2.7. Bagging	5
2.3. Algorithme de régression	6
2.3.1. Support Vector Machine pour la régression (SVR)	6
2.3.2. Gaussian Process (GP)	6
2.4. Détection d'outliers [FP17]	7
2.5. Étude des travaux existants	7
2.5.1. Analyse comparative de différents algorithmes pour le positionnement intérieur [Sin+15]	7
2.5.2. Machine learning pour diminuer l'erreur de positionnement UWB [Hen+12]	10
2.5.3. Amélioration du positionnement en utilisant "conformal prediction" [KN17]	11
2.6. Choix et conclusion	14
3. Prise de mesures	15
3.1. Plan de mesure	15
3.2. Setup pour la prise de mesures	15
3.2.1. Matériel utilisé	17
3.3. Programme de prise de mesure [Mic17]	18
3.3.1. Architecture logicielle	18
3.3.2. Prise de données brutes	20
3.3.3. Prise de données de positions convergées	22
4. Traitement des données et classification	25
4.1. Structure du traitement	25
4.2. Extraction des caractéristiques	26
4.3. Reconnaissance - Algorithme	28
4.3.1. Hyper paramètres	28
4.4. Validation et résultats	29
4.4.1. Résultats en utilisant les données de "ranging"	30
4.4.2. Résultats en utilisant les données de position convergées	37
4.5. Résumé des résultats	39
5. Problèmes rencontrés et améliorations	41

6. Conclusion	43
Bibliographie	45
A. Cahier des charges	49
B. Planning	55
C. Journal de travail	57

Table des figures

1.1.	État de l'art des différentes méthodes de positionnement intérieur [LLI]	1
2.1.	Comment choisir un algorithme ML [Ped+11]	3
2.2.	a) exemple de classification b) exemple de régression	4
2.3.	Phase de construction et d'apprentissage [Sin+15]	8
2.4.	Résultats des précisons pour la classification par bâtiments [Sin+15]	8
2.5.	Résultats de la précision pour la classification par étage.[Sin+15]	9
2.6.	Résultats de la précision pour la classification par région. [Sin+15]	9
2.7.	Résultat de la précision des algorithmes d'apprentissage.[Sin+15]	9
2.8.	Résultats du temps de traitement des algorithmes d'apprentissage[Sin+15]	10
2.9.	Ranging error - LOS and NLOS [Hen+12]	11
2.10.	Les deux phases du fingerprinting	12
2.11.	Performance de la précision du fingerprint lors de la compétition Microsoft IPSN 2014 competition [KN17]	12
2.12.	Performance de la précision de W-KNN, Naïve Bayes et réseau de neurones, mais aussi l'impacte du nombre de données d'entraînement[KN17]	13
2.13.	Summary of the three fingerprinting test beds used in this article [KN17]	14
3.1.	Plan des mesures effectuées et du setup	16
3.2.	Montre le setup pour la prise de mesure	16
3.3.	Montre le plan réel du laboratoire	17
3.4.	Montre le plan réel du laboratoire	17
3.5.	Blocs principaux du logiciel de prise de mesure	18
3.6.	Interface graphique du soft de prise de mesures	19
3.7.	Fonctionnement du mode spy et signification des mesures brutes	20
3.8.	Diagramme expliquant la prise de mesures des valeurs de ranging	21
3.9.	Structure des données des mesures de localisation	22
3.10.	Diagramme expliquant la prise de mesures des positions convergées	23
3.11.	Structure des données de mesures contenant uniquement la valeur des positions convergées	24
4.1.	Flux de traitement de l'information	25
4.2.	Données sous forme de dataframe pour le traitement et l'extraction des caractéristiques	27
4.3.	Partage des données	28
4.4.	Influence des paramètres C et gamma	29
4.5.	Représentation de la précision et du rappel [WIKI]	30
4.6.	Représentation des points non convergé par rapport aux positions réelles	31
4.7.	Matrice de confusion obtenue en utilisant les points Sx et les caractéristiques de "ranging"	32
4.8.	Matrice de confusion obtenue en utilisant les points de tests SxT et les caractéristiques de "ranging" de la 1ère et 2ème mesures de l'esclave	32
4.9.	Matrice de confusion obtenue en utilisant les points de tests SxT et les caractéristiques de "ranging" de la 2ème mesure de l'esclave uniquement	33
4.10.	Matrice de confusion obtenue en utilisant les points Sx et la caractéristique de position	34
4.11.	Matrice de confusion obtenue en utilisant les points SxT et la caractéristique de position	34
4.12.	Matrice de confusion obtenue pour la meilleure solution en utilisant les données des points Sx	35
4.13.	Matrice de confusion pour la meilleure solution en utilisant les points de tests SxT	36
4.14.	Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés autour des points entraînés	36
4.15.	Représentation des points convergés par rapport aux positions réelles	37

4.16. Matrice de confusion pour les valeurs de la position convergée avec des points positionnés à la même place	38
4.17. Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés à la même place	38
4.18. Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés autour des points entraînés	39
4.19. Résumé des différents résultats obtenus qui sont présentés dans ce rapport	39

1. Introduction

1.1. Contexte

La précision du positionnement intérieur reste quelque chose de très important, car il peut être utile dans plusieurs domaines comme : la gestion de stocks, la localisation de personnes âgées dans des homes, la localisation de personnes pour la surveillance, etc.

Les systèmes de localisation basés sur GPS souffrent de la détérioration de la précision et sont presque indisponibles dans les environnements intérieurs. Pour les environnements intérieurs, de nombreuses technologies de systèmes de positionnement ont été conçues sur la base de la vision, de la détection infrarouge ou ultrasonore, des champs magnétiques de la terre, des accéléromètres / gyromètres, des balises BLE ou de la communication WiFi. Bien que la création de ces nouvelles applications ait été couronnée de succès, le coût de ces récepteurs, leur consommation d'énergie et leur limitation aux environnements extérieurs excluent de nombreuses applications.

La Figure 1.1 montre un graphique comparatif de la précision de positionnement concernant différentes technologies [LLI]. Ce graphique montre qu'un système utilisant la vision apporte une grande précision. Mais, il peut poser des problèmes dans certaines applications où il serait nécessaire de mettre beaucoup de caméras et cela implique souvent des traitements assez complexes. En comparaison, un système à ultrason qui est un peu comparable à un système infrarouge est meilleur marché. Il est également nécessaire de placer un grand nombre de capteurs selon les utilisations. Concernant les systèmes RFID, ils ont une portée très courte qui se limite à quelques mètres et la précision est dégradée par rapport aux systèmes précédents. Les deux derniers systèmes qui sont les réseaux sans fil (Bluetooth et WLAN) possèdent beaucoup d'avantages, car ils sont largement répandus et possèdent une bonne couverture. Un gros inconvénient de ces deux technologies est qu'elles sont gourmandes en énergie et ne peuvent pas fonctionner sur batterie.

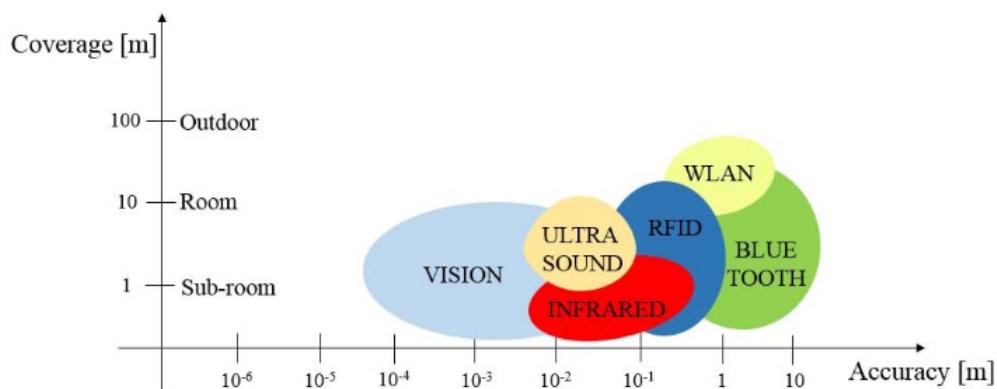


Figure 1.1. – État de l'art des différentes méthodes de positionnement intérieur [LLI]

La localisation avec LoRa est une possibilité séduisante et probablement l'une des meilleures candidates pour le positionnement intérieur. Le faible coût des infrastructures et des noeuds finaux ainsi que la disponibilité pourraient permettre de nombreuses nouvelles applications. Il n'est donc pas surprenant que les chercheurs et les entités commerciales se soient mis au travail sur ce problème au cours des derniers mois. Cependant, plusieurs défis restent à relever pour qu'un tel système devienne pratique.

LoRa peut également être utilisée pour un positionnement intérieur. À ce jour, très peu d'expériences sont disponibles pour la conception de tels systèmes et c'est pourquoi il est nécessaire d'effectuer différentes

études (dont ce travail) afin de connaitre la précision qui peut être atteinte. Un autre aspect très important qui fait de la technologie LoRa est une très bonne candidate pour du positionnement intérieur est sa très faible consommation.

L'objectif général de ce projet est d'étudier diverses technologies d'apprentissage (Machine Learning) permettant d'améliorer la précision de la localisation en intérieur sur la base de la technologie LoRa et en utilisant le mode "ranging" qui permet de mesurer des temps de vol.

1.2. Aspect novateur

Ce travail d'approfondissement va permettre d'évaluer une nouvelle approche pour effectuer du positionnement intérieur. En s'appuyant sur les capacités étendues des nouveaux circuits intégrés LoRa, ce projet développera et déployera un système de localisation capable d'améliorer la précision de la position atteinte par les systèmes de localisation basés sur LoRa existants et reposants sur des mécanismes TDOA ou de "ranging". À cette fin, une exploration et une comparaison des différentes techniques "machine learning/deep learning" pour l'amélioration de la précision du positionnement basé sur le "fingerprinting" seront effectuées.

L'aspect novateur du projet est d'intégrer au système de positionnement basé sur LoRa un mécanisme d'apprentissage de la position afin d'augmenter la précision du positionnement en intérieur.

1.3. Structure du rapport

Ce rapport va décrire dans un premier temps l'état de l'art des techniques à utiliser, en particulier les systèmes de localisation intérieurs basés sur des techniques d'apprentissage.

Toute la partie concernant la prise de mesures pour obtenir des données sera expliquée afin de comprendre la mise en place et pouvoir la reproduire. Cette étape est très importante dans le projet, car elle va permettre d'évaluer le système. Pour se faire, un plan de mesure a été mis en place ainsi qu'une description du matériel à utiliser. Une brève explication du logiciel de prise de mesure est également présentée afin de comprendre comment les données sont récupérées et structurées.

Ensuite, basé sur les études, le choix de l'algorithme sera détaillé ainsi que son implémentation. La présentation de quelques résultats intéressants sera également détaillée en fin de rapport.

Finalement, une perspective pour le futur sera donnée afin de pouvoir poursuivre le travail et aller plus loin dans l'analyse des données et l'apprentissage avec un algorithme de régression plutôt que de classification.

2. État de l'art

Ce chapitre passe en revue les solutions existantes qui parlent d'application des méthodes d'apprentissage pour l'amélioration de la précision de la localisation intérieure. L'exploration de ces sujets va permettre de choisir la solution la plus adaptée pour remplir les objectifs de ce travail.

Pour ce faire, quatre ouvrages ont été étudiés. Les trois premiers traitent du positionnement intérieur aidé avec des algorithmes d'apprentissage (Machine Learning). Le quatrième a été sélectionné, car il traite de la gestion de "outliers", c'est à dire comment détecter des points aberrants et qui pourraient fausser les mesures.

Grâce à l'étude des structures existantes, une première classification selon le type d'apprentissage et selon les algorithmes de classification utilisés peut être établie. En plus, un aperçu de la façon de détecter les "outliers" est soulevé.

2.1. Classification des algorithmes d'apprentissage

Il existe plusieurs catégories d'estimateur et dans ces catégories, il y a plusieurs algorithmes à choix. La Figure 2.1 permet de sélectionner différents algorithmes en fonction de ce qu'il est nécessaire d'obtenir. On remarque sur cette figure qu'il y a quatre groupes distincts : classification, regression, clustering et dimensionality reduction. En partant depuis le point "start" et en répondant aux différentes questions, cela va proposer un algorithme qui permet de répondre à nos besoins.

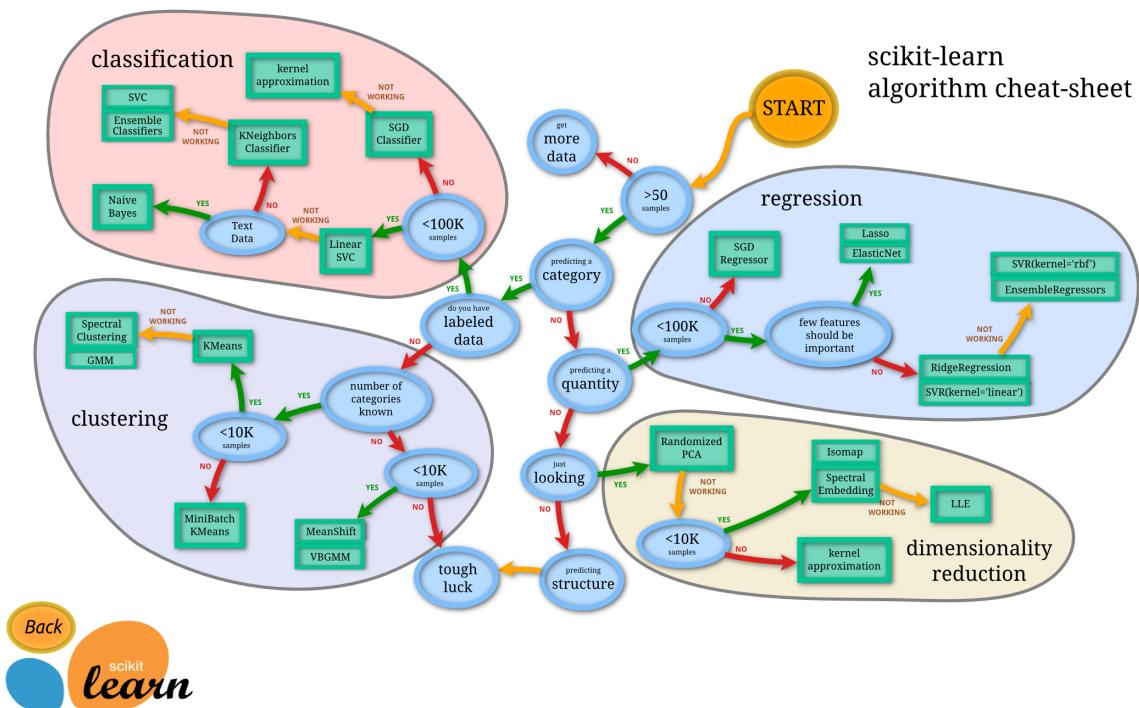


Figure 2.1. – Comment choisir un algorithme ML [Ped+11]

Dans le cas de ce travail, il y a deux possibilités, soit d'utiliser une classification soit d'utiliser une régression. Dans la Figure 2.2, on peut visualiser la différence entre les deux types d'estimateurs. Dans le cas de la classification, on va apprendre à notre algorithme différentes positions et l'algorithme sera ensuite capable de fournir en sortie une de ses positions. Donc, il sort une région plutôt qu'une coordonnée. Si l'entraînement de l'algorithme a été fait avec un maillage fin alors le résultat sera fin. Lorsqu'un point est détecté, il sera classifié selon la position la plus proche comme on le voit sur la Figure 2.2 avec le carré vert et le carré rouge. Dans une régression, la problématique est un peu différente. L'algorithme sera également entraîné avec différentes positions, mais ensuite, par régression l'algorithme va tenter d'estimer la position du nouveau point en calculant sa coordonnée alors que pour une classification il s'agit de positionner ce nouveau point dans une classe.

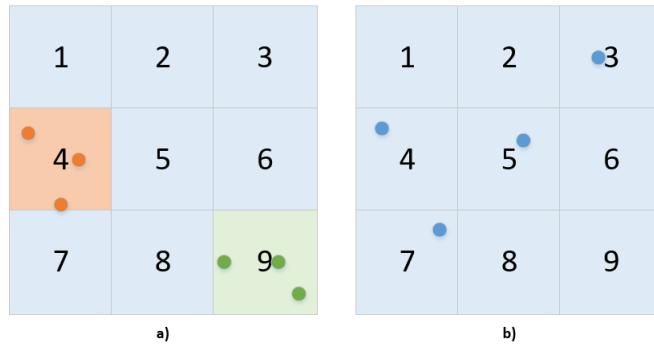


Figure 2.2. – a) exemple de classification b) exemple de régression

2.2. Algorithme de classification

Dans cette section, quelques algorithmes de classification utilisés dans ces études [FP17] [KN17] sont brièvement décrits et analysés.

2.2.1. Decision Tree

L'arbre de décisions est une méthode très connue en "machine learning". Il possède des noeuds de décisions (non-terminal), des branches, et des noeuds feuilles (terminal) qui représentent les caractéristiques, condition et les classes. À chaque noeud de décision on sait quelles branches suivre et lorsque l'algorithme atteint un noeud final, le label contenu dans ce même noeud est retourné comme étant la classe. L'ID3 de Quinlan et son successeur, C4.5, sont les plus populaires parmi les algorithmes d'arbre de décision [JR14].

2.2.2. Naïve Bayes

Le classificateur Naïve Bayes [GHP95] basé sur le théorème de Bayes est un algorithme d'apprentissage supervisé [CS13]. Il est robuste aux données bruyantes, faciles à construire, affiche une grande précision et rapidité lorsqu'il est appliqué à de grandes bases de données et exécute des modèles de classification plus complexes. Par conséquent, il est largement utilisé dans les tâches de classification. Il calcule la probabilité de chaque attribut dans les données en supposant qu'elles sont également importantes et indépendantes les unes des autres. Cette hypothèse est appelée indépendance conditionnelle de classe [WA13] [SS14].

2.2.3. Bayesian Network

L'algorithme de réseau bayésien est largement utilisé pour la classification et est basé sur le théorème de Bayes où la probabilité conditionnelle sur chaque nœud est calculée et forme un réseau bayésien. Il

s'appelle également réseau de croyance ou réseau occasionnel. Le réseau bayésien à deux parties nommées qualitatives et quantitatives, qui sont la structure topologique du réseau bayésien et respectivement le tableau de probabilité conditionnelle (CPT) [DLo7].

Le réseau bayésien est un graphe acyclique dirigé où chaque nœud représente un attribut des données et un ensemble de distributions de probabilité. Ces distributions donnent les probabilités pour la valeur de chaque nœud.

2.2.4. K-Nearest Neighbor

Le classificateur K-Nearest Neighbor (K-NN) [DWDMK91] est également connu sous le nom de classificateur basé sur la distance qui classe les instances en fonction de leur similarité. C'est l'un des algorithmes les plus populaires de l'apprentissage automatique. C'est un type d'apprentissage paresseux dans lequel la fonction n'est approchée que localement et tout calcul est retardé jusqu'à la classification. Le point inconnu dans K-NN est assigné à la classe la plus commune parmi ses K-plus proches voisins. Lorsque K = 1, le point inconnu se voit attribuer la classe du point d'apprentissage le plus proche dans l'espace des motifs [CAG13].

2.2.5. SMO / SVM

L'algorithme d'optimisation séquentielle minimale (SMO - Sequential minimal optimization) [J.98] est représenté par John C. Platt pour la formation du classificateur de vecteurs de support à l'aide des noyaux polynomiaux ou RBF. C'est l'un des algorithmes les plus courants pour la classification des grandes marges par SVM. Il remplace globalement toutes les valeurs manquantes et transforme les attributs nominaux en attributs binaires. SVM est une technique de classification basée sur la technologie des réseaux de neurones utilisant la théorie de l'apprentissage statistique [PH14]. Il recherche un hyperplan optimal linéaire afin de maximiser la marge de séparation entre la classe positive et la classe négative. En pratique, la plupart des données ne sont pas linéairement séparables ; ainsi, pour rendre la séparation possible, la transformation est effectuée à l'aide d'une fonction du noyau. L'entrée est transformée en un espace caractéristique de dimensions supérieures à l'aide d'une cartographie non linéaire. Une décision sur la fonction du Kernel est nécessaire pour implémenter SVM. Le Kernel définit la classe de fonction [SMKN13].

2.2.6. AdaBoost

AdaBoost (Adaptive Boosting) [YRE96] est un algorithme d'apprentissage d'ensemble. Généralement, il peut être utilisé avec des algorithmes de Machine learning faibles pour améliorer leurs performances. Il est simple à mettre en œuvre, rapide et moins susceptible d'avoir un overfitting. Il améliore les algorithmes de classification instables tels que J48, DecisionStump, etc. L'idée derrière cet algorithme est d'obtenir un classificateur très précis en combinant de nombreux classificateurs faibles. Il fonctionne en exécutant de manière répétée un algorithme d'apprentissage faible donné sur diverses distributions sur les données d'apprentissage, puis en combinant les classificateurs produits par l'apprenant faible en un classificateur composite unique [RRE13]. Les classificateurs de l'ensemble sont ajoutés un par un, de sorte que chaque classificateur suivant est entraîné sur des données difficiles pour les membres précédents de l'ensemble. Les poids sont définis sur les instances du jeu de données, en suivant une règle selon laquelle les instances difficiles à classer prennent plus de poids. Cette règle conduit les classificateurs ultérieurs à se concentrer sur eux [SOLISP10].

2.2.7. Bagging

Le Bagging [L.96] crée des sacs de données de la même taille que le jeu de données d'origine en appliquant une sélection aléatoire à différents sous-ensembles des données d'apprentissage avec de nombreux exemples qui apparaissent plusieurs fois. Ce processus est appelé réPLICATION bootstrap des données d'entraînement. L'idée derrière cette technique est de construire différents classificateurs en utilisant ces sous-ensembles.

Chaque sous-ensemble est utilisé pour entraîner un classificateur individuel. Cette approche d'ensemble utilise le nombre de classificateurs a priori [L.96].

2.3. Algorithme de régression

Dans cette section, quelques algorithmes de régression utilisés dans cette étude [Hen+12] sont brièvement décrits et analysés.

2.3.1. Support Vector Machine pour la régression (SVR)

Support Vector Machine (SVM) est une méthode d'apprentissage supervisée utilisée pour la classification. Cette méthode peut être étendue pour résoudre des problèmes de régression et cette méthode s'appelle Support Vector Regression (SVR).

Support Vector Regression (SVR) est considéré comme une technique non paramétrique, car il dépend du Kernel.

Il existe trois implémentations différentes de Support Vector Regression : SVR, NuSVR et LinearSVR. LinearSVR fournit une implémentation plus rapide que SVR, mais ne considère que les noyaux linéaires, tandis que NuSVR implémente une formulation légèrement différente de SVR et de LinearSVR.[Ped+11]

2.3.2. Gaussian Process (GP)

Gaussian processes a récemment gagné en intérêt du côté des algorithmes d'apprentissage. Cela, car il forme un bon framework pour résoudre des problèmes de régression [CCo6].

C'est une méthode d'apprentissage supervisée pour résoudre des problèmes de régression comme dit ci-dessus, mais également des problèmes de probabilités.

Les avantages du processus gaussien sont :

1. La prédiction interpole les observations (au moins pour les noyaux normaux).
2. La prédiction est probabiliste (gaussienne) afin que l'on puisse calculer des intervalles de confiance empiriques et décider en fonction de ceux-ci si l'adaptation convient de réajuster (adaptation en ligne, adaptation adaptative) la prévision dans une région d'intérêt.
3. Polyvalent : différents noyaux peuvent être spécifiés. Les noyaux communs sont fournis, mais il est également possible de spécifier des noyaux personnalisés.

Les inconvénients des processus gaussiens sont :

1. Ils ne sont pas clairsemés, c'est-à-dire qu'ils utilisent l'ensemble des informations sur les échantillons / caractéristiques pour effectuer la prédiction.
2. Ils perdent leur efficacité dans les espaces de grandes dimensions, notamment lorsque le nombre de caractéristiques dépasse quelques dizaines.

[Ped+11].

2.4. Détection d'outliers [FP17]

Cette section donne un aperçu de la manière de traiter les "outliers - valeurs aberrantes", c'est-à-dire les points qui ne sont pas cohérents lors d'une mesure. Selon Barnet et Lewis [VT94], un "outliers" est défini comme étant une observation qui semble incompatible avec le reste d'un ensemble de données. Garder un "outliers" dans un jeu de données peut amener à de mauvais résultats, il est donc important de les détecter correctement. Il existe différentes méthodes pour déterminer ces "outliers" :

1. Grubbs' test : détecte un "outliers" en supposant une distribution normale.
2. Tietjen-Moore test : C'est une généralisation de Grubbs' test pour détecter de multiple outliers. Il a cependant un inconvénient, il est nécessaire de connaître le nombre exact d'outliers.
3. Generalized Extreme Studentized Deviate (ESD) : c'est également une généralisation du test Grubbs', mais il n'est pas nécessaire de connaître à l'avance le nombre d'outliers. Ce test nécessite uniquement une limite supérieure pour le nombre suspect d'outliers.[Esd]

2.5. Étude des travaux existants

Les trois documents ne peuvent pas être comparés à proprement dit, car ils traitent de différentes manières. Un parle d'algorithme de classification et compare différents algorithmes existants. Un autre document traite de régression et compare également différents algorithmes. Le dernier document parle de la prédition conforme (CP - conformal prediction) qui va permettre d'améliorer encore les résultats avec un algorithme choisi.

C'est pourquoi les documents sont résumés en mettant les points importants en évidence séparément. Ils seront décrits plutôt que comparés.

2.5.1. Analyse comparative de différents algorithmes pour le positionnement intérieur [Sin+15]

Cette publication parle d'une analyse comparative entre différents algorithmes de "machine learning" pour du positionnement intérieur. L'étude est basée sur un positionnement "fingerprint" ce qui permet de cartographier un endroit à l'aide de la force du signal réceptionné (RSS - Received Signal Strength). Dans cet article, les algorithmes de "machine learning" sélectionnés sont comparés en termes de précision de positionnement et de temps de calcul.

Au cours des expériences, la base de données UJIIndoorLoc est utilisée. La classification est effectuée en premier lieu en utilisant le jeu de données d'origine en considérant les valeurs RSS des 520 points d'accès sans fil (WAP - wireless access points) et les nouveaux attributs définis en tant que «cellule» qui composent les attributs : BuildingID, Floor, SpaceID et RelativePos. Ensuite, une nouvelle méthode est proposée : «Séparation déductive pour le positionnement intérieur (DESIP - Deductive Separation for Indoor Positioning)». Dans cette méthode, tout d'abord, seules les informations de bâtiment et les valeurs RSS mesurées à partir de 520 WAP sont utilisées pour la tâche de classification.

Durant les expériences, des algorithmes tels que le plus proche voisin (NN - nearest neighbor), le SMO, l'arbre de décision (J48), Naïve Bayes et Bayes Net sont utilisés. L'algorithme le plus approprié pour la solution du problème de positionnement intérieur est déterminé en comparant la précision et le temps de calcul de chaque approche.

La base de données entière est séparée de telle sorte que 19'937 enregistrements soient réservés à l'apprentissage et 1'111 enregistrements soient réservés aux tests. Il y a 529 caractéristiques qui sont : les coordonnées où sont prises les empreintes digitales WiFi, telles que bâtiment, étage, espace (bureau, laboratoire, etc.), position relative (dans une pièce ou dans un couloir), etc. Le jeu de données d'apprentissage UJIIndoorLoc comprenant les valeurs RSSI de 520 WAP et une autre «cellule» qui comprend les attributs floor, buildingID, spaceID et relative position de l'ensemble de données d'origine est utilisé pour la tâche de classification. Les étapes des expériences utilisant ce jeu de données sont illustrées à la Figure 2.3. Dans un premier temps, le jeu de données (dataset) est séparé en deux groupes. Un groupe est utilisé pour l'apprentissage

(train dataset) et l'autre partie est utilisée pour valider le résultat des différents algorithmes (test dataset). Le jeu d'entraînement est utilisé pour quatre différents algorithmes.

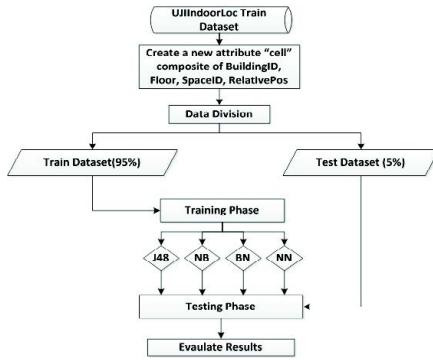


Figure 2.3. – Phase de construction et d'apprentissage [Sin+15]

Dans cet article, les algorithmes suivants ont été comparés dans différentes situations : NN, SMO, J48, Naïve Bayes et BayesNet. La classification est effectuée en trois phases (building,floor and region). La première étape est la classification par bâtiment (Figure 2.4). Le résultat des algorithmes pour cette classification donne BayesNet comme étant le meilleur, car il possède la meilleure précision.

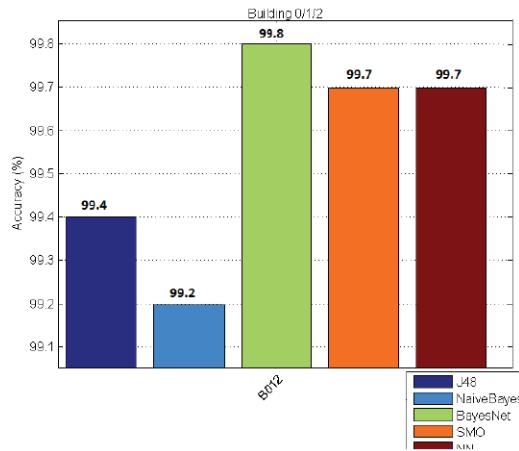


Figure 2.4. – Résultats des précisons pour la classification par bâtiments [Sin+15]

Suite à cela, la classification a été faite en fonction des étages (floors) (Figure 2.5). Dans ce cas, le meilleur algorithme est NN.

Et pour la dernière étape (Figure 2.6), la classification a été faite en fonction de la région et là encore, c'est l'algorithme NN qui est le meilleur.

Les deux tableaux (Figure 2.7 et Figure 2.8) permettent d'avoir une vue d'ensemble de tous les algorithmes appliqués et leur performance en termes de précision et de rapidité. L'algorithme NN est le meilleur pour tous les jeux de données au niveau du temps d'exécution. En ce qui concerne la précision, Bayes Net est meilleur pour la classification "building" par contre NN est meilleur dans tous les autres cas.

Les résultats expérimentaux révèlent que l'algorithme k-Nearest Neighbor (k-NN) est le plus approprié lors du positionnement. En outre, J48 offre des performances quasiment identiques lorsqu'il est utilisé avec des algorithmes itératifs, à savoir AdaBoost et Bagging. Il est à noter que toutes ces méthodes mettent en évidence des classifications. C'est-à-dire que le résultat placera un nouveau point sur l'un des points entraîné et ne donne pas une nouvelle coordonnée comme cela serait le cas d'un résultat par régression.

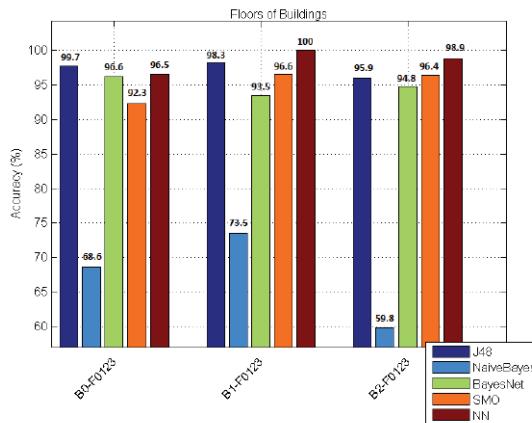


Figure 2.5. – Résultats de la précision pour la classification par étage.[Sin+15]

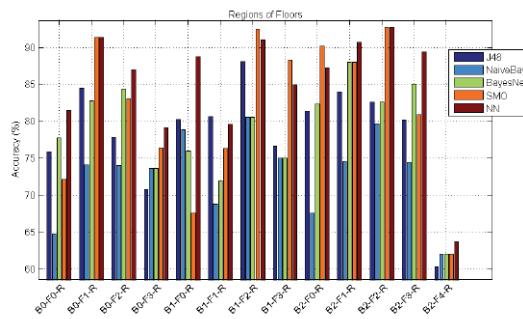


Figure 2.6. – Résultats de la précision pour la classification par région. [Sin+15]

	J48	Naive Bayes	Bayes Net	SMO	NN
B012	99,4	99,19	99,80	99,69	99,69
B0_F0123	97,7	68,58	96,17	92,34	96,55
B0_F0_R	75,93	64,81	77,78	72,22	81,48
B0_F1_R	84,48	74,14	82,76	91,37	91,38
B0_F2_R	77,92	74,03	84,42	83,12	87,01
B0_F3_R	70,83	73,61	73,61	76,39	79,17
B1_F0123	98,28	73,54	93,47	96,56	100
B1_F0_R	80,28	78,87	76,06	67,61	88,73
B1_F1_R	80,64	68,82	72,04	76,34	79,57
B1_F2_R	88,06	80,60	80,59	92,54	91,04
B1_F3_R	76,67	75	75	88,33	85
B2_F01234	95,95	59,77	94,83	96,40	98,87
B2_F0_R	81,37	67,65	82,35	90,19	87,25
B2_F1_R	84	74,67	88	88	90,67
B2_F2_R	82,61	79,71	82,61	92,75	92,75

Figure 2.7. – Résultat de la précision des algorithmes d'apprentissage.[Sin+15]

	J48	Naive Bayes	Bayes Net	SMO	NN
B012	62,41	2,5	11,19	7,25	0,06
B0_F0123	11,74	0,65	2,03	14,31	0,01
B0_F0_R	1,5	0,22	0,7	14,1	0
B0_F1_R	1,71	0,21	0,88	17,93	0
B0_F2_R	2,52	0,2	0,9	18,77	0
B0_F3_R	1,62	0,2	0,88	17,3	0
B1_F0123	10,05	0,57	2,13	11,5	0,02
B1_F0_R	1,68	0,18	0,87	18,22	0
B1_F1_R	1,75	0,21	1,01	13,3	0
B1_F2_R	1,96	0,27	1,11	19,47	0
B1_F3_R	0,95	0,13	1,11	12,1	0
B2_F01234	18,09	1,18	4,4	18,81	0,03
B2_F0_R	2,54	0,24	1,11	21,57	0
B2_F1_R	3,27	0,29	1,15	27,1	0
B2_F2_R	1,86	0,2	1,46	19,76	0

Figure 2.8. – Résultats du temps de traitement des algorithmes d'apprentissage [Sin+15]

2.5.2. Machine learning pour diminuer l'erreur de positionnement UWB [Hen+12]

Les approches classiques pour faire face au défi de localisation dans des environnements encombrés impliquent généralement d'abord la détection de la condition NLOS (NLOS = pas en ligne de vue), puis la prise de mesures appropriées pour prendre en compte la condition NLOS. Toutefois, la grande variété de matériaux et d'environnements d'exploitation variés peuvent impacter les performances de la mesure de ranging, ce qui indique que la distinction entre LOS et NLOS n'est pas toujours significative. Sur la base de cette observation, ils ont opté pour une approche différente dans cet article. Leur approche utilise des techniques de Machine Learning non paramétriques (SVM et GP) pour estimer l'erreur de "ranging" directement à partir de la forme d'onde reçue, sans aucune connaissance a priori ou a posteriori de la condition NLOS.

Cette publication traite de la diminution de l'erreur du mode ranging concernant la localisation UWB (Ultra Wide Band). Plusieurs techniques existent pour diminuer l'erreur de positionnement en détectant ce qui est en ligne de vue (LOS) ou non (NLOS). Ici, une autre technique est exploitée et va directement diminuer cette erreur que ça soit en LOS ou en NLOS. Ils appliquent deux classes de régresseurs non paramétriques pour avoir une estimation de l'erreur de mesure. Afin de valider leurs résultats, ils ont fait une vaste campagne de mesures intérieures. Cette technique montre une amélioration de performances significatives dans divers scénarios par rapport aux approches conventionnelles.

La régression non paramétrique est une forme d'analyse de la régression dans laquelle le prédicteur, ou fonction d'estimation ne prend pas de forme prédéterminée, mais est construite selon les informations provenant des données. La régression non paramétrique exige des tailles d'échantillons plus importantes que celles de la régression basée sur des modèles paramétriques parce que les données doivent fournir la structure du modèle ainsi que les estimations du modèle. [Wika]

Ils se sont appuyés sur des outils d'apprentissage (Machine Learning), et proposent deux techniques de régression pour estimer l'erreur de mesure, en se basant uniquement sur la forme d'onde reçue et la distance estimée.

1. La première technique utilise une régression SVM (support vector machine) pour trouver un hyperplan qui se rapproche de l'erreur de mesure en fonction des données d'apprentissage.
2. La seconde technique utilise un processus gaussien (GP) pour déterminer la distribution a posteriori de l'erreur de mesure, en fonction des données d'apprentissage.

L'erreur de mesure estimée, associée à une mesure de certitude, peut être transmise à un algorithme de localisation. Leurs techniques de régression présentent l'avantage supplémentaire de pouvoir être

appliquées même lorsque les données d'apprentissage ne sont pas étiquetées avec des informations LOS ou NLOS.

Lors des mesures, il existe beaucoup de paramètres qui peuvent créer une erreur. Avec un seul modèle, il est difficile de capturer tous les types de perturbations. Dans cette publication, ils se basent sur 1024 mesures (512 LOS et 512 NLOS) Figure 2.9.

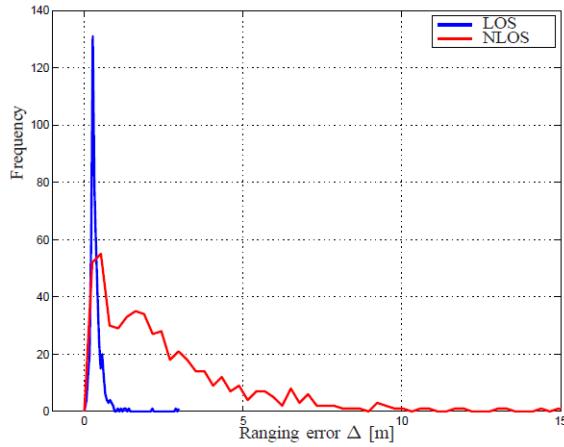


Figure 1. Histogram of the ranging error for the LOS and NLOS condition.

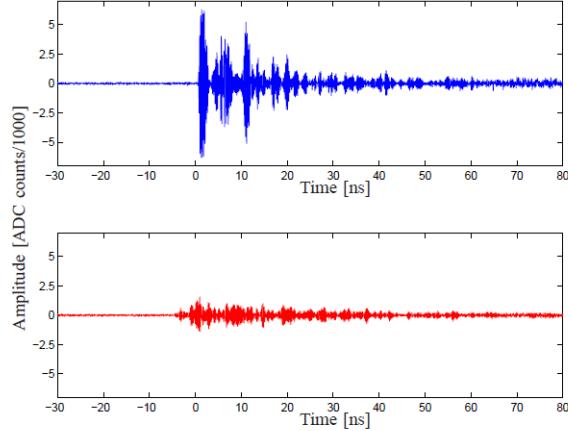


Figure 2. In some situations there is a clear difference between LOS (upper waveform) and NLOS (lower waveform) signals.

Figure 2.9. – Ranging error - LOS and NLOS [Hen+12]

Sur la base d'une vaste campagne de mesures en intérieur avec des radios UWB conformes à la FCC (Federal Communications Commission), ils ont évalué les performances de localisation en termes de probabilité de panne pour différentes stratégies de localisation. Leurs résultats ont révélé que :

1. La minimisation de l₁-norme est plus robuste pour faire face aux valeurs aberrantes (outliers) que la minimisation de l₂-norme, pour une localisation sans atténuation.
2. les contraintes peuvent générer des gains significatifs, en particulier lorsque les exigences de localisation ne sont pas trop strictes.
3. les techniques de régression SVM ou GP offrent des gains de performance supplémentaires pour tous les scénarios considérés.
4. Les techniques de régression SVM ou GP, combinées à la connaissance des contraintes relatives à l'erreur de "ranging", offrent les meilleures performances pour les scénarios considérés.

2.5.3. Amélioration du positionnement en utilisant "conformal prediction" [KN17]

Pareil que dans les deux précédentes publications, le but est d'améliorer la précision du positionnement intérieur où il n'est pas possible d'utiliser un GPS. Cela toujours en tenant compte des problématiques d'un environnement dynamique avec des personnes qui bougent et de l'environnement complexe avec des murs, etc.. Dans cet article, ils ont validé leur solution dans trois immeubles. Cet article est basé sur un positionnement WIFI fingerprinting et se base sur la force du signal reçu (RSSI).

Pour effectuer les mesures il y a deux phases voir Figure 2.10.

La première phase (Off-line training) consiste à décider combien de mesures nous voulons faire (tous les mètres ? centimètres ?), comment sont prises les mesures (plusieurs mesures sont faites à chaque position) et comment labéliser le signal (souvent labellisé avec la coordonnée réelle). Cette phase est très importante et il est nécessaire de bien réfléchir comment procéder (utiliser un robot par exemple). C'est ce qu'on appelle la phase d'entraînement.

Concernant la seconde phase (on-line positioning), c'est de définir le positionnement. Durant cette phase la partie délicate est de définir quel algorithme utiliser. C'est durant cette phase que les algorithmes seront testés.

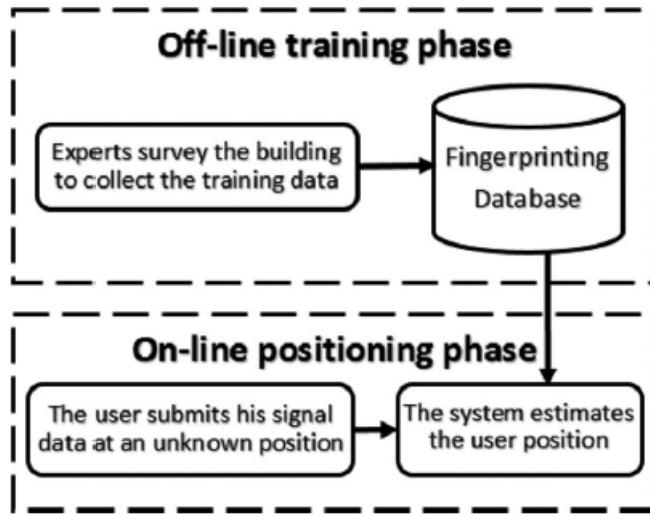


Figure 2.10. – Les deux phases du fingerprinting

Il existe une compétition comparative pour les positionnements intérieurs (Microsoft IPSN 2014) Figure 2.11.

Une localisation précise à l'intérieur peut potentiellement transformer la façon dont les gens naviguent à l'intérieur, de la même manière que le GPS a transformé la façon dont les gens naviguent à l'extérieur. Au cours des 15 dernières années, le monde universitaire et l'industrie ont proposé et expérimenté plusieurs technologies de localisation en intérieur, mais nous n'avons pas encore assisté à des déploiements à grande échelle. Ce concours vise à rassembler des technologies de localisation intérieure en temps réel ou quasi réel et à comparer leurs performances dans le même espace. [Mic]

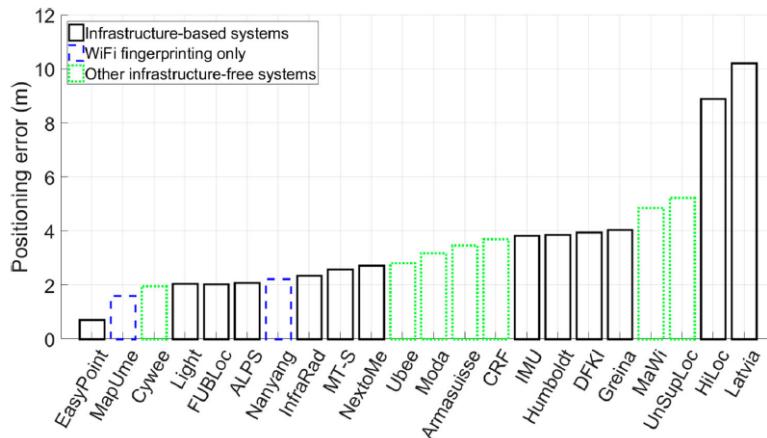


Figure 2.11. – Performance de la précision du fingerprint lors de la compétition Microsoft IPSN 2014 competition [KN17]

Cela permet de mettre en évidence les précisions qui sont obtenues. Les algorithmes qui seront étudiés sont : Weighted K-nearest neighbours (W-KNN), Naïve Bayes Figure 2.12. Tous les mesures et tests sont basés sur le RSS du WIFI.

Les mesures ont été effectuées de trois manières différentes (dans un corridor, sur un étage, sur trois étages - voir dans le document [KN17]). En résumé, la synthèse des trois solutions suggère que, avec uniquement la mesure métrique WiFi RSS, de nombreux algorithmes complexes risquent de ne pas être aussi performants que des algorithmes plus simples. Malgré sa simplicité, W-KNN a excellé dans la plupart des analyses de "fingerprinting". Il convient de noter que le système MapUme, deuxième sur vingt-deux concurrents

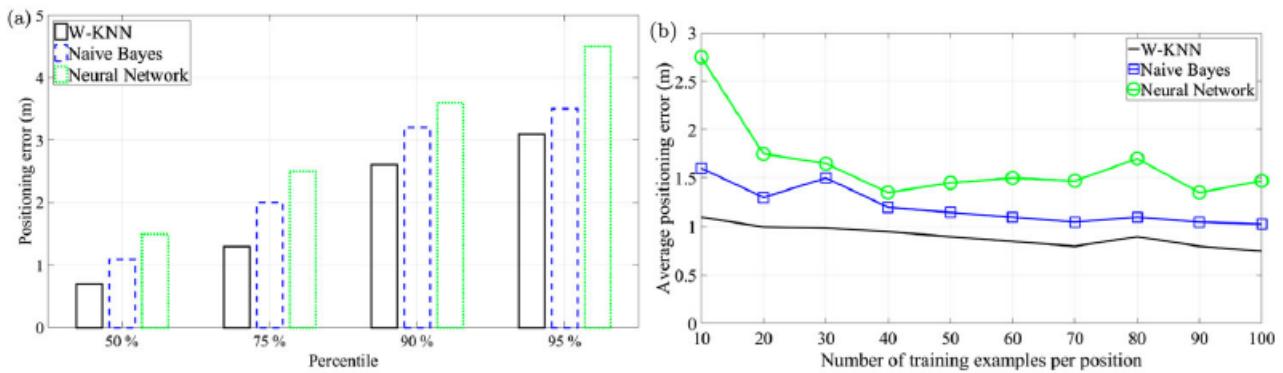


Figure 2.12. – Performance de la précision de W-KNN, Naïve Bayes et réseau de neurones, mais aussi l'impacte du nombre de données d'entraînement[KN17]

du concours Microsoft IPSN 2014, utilisait également W-KNN comme principal algorithme. Cependant, l'approche Naïve Bayes améliore sa précision lorsque le nombre d'entrainements est élevé, ce qui indique qu'au-delà du WiFi RSS, des informations supplémentaires seront nécessaires pour améliorer davantage les performances du "fingerprinting".

Une chose qui est également importante c'est la confiance qu'il y a dans un algorithme. Pour cela, cet article a utilisé "conformal prediction (CP)" afin de donner un indice de confiance. Plus la confiance est souhaité, plus il est nécessaire d'avoir de jeu de données. Les avantages d'utiliser l'indice de confiance sont les suivants :

1. Chaque prédiction est associée à un indice de confiance et permet de dire combien la prédiction est correcte.
2. La prédiction produite par CP est statistiquement correcte sous les paramètres qui ont été choisis dans la phase on-line.
3. le niveau de confiance peut être ajusté pour produire un ensemble de prédiction plus grand ou plus petit.

Afin de valider leur algorithme, ils ont utilisé trois bancs de test Figure 2.13.

1. Royal Holloway : Données récoltées manuellement dans un office standard avec un smartphone
2. Cambridge : Données récoltées automatiquement par un robot dans un environnement assez idéal.
3. UJIIndoorLoc : Utilise le dataset public qui couvre une grande surface intérieure basée sur trois bâtiments.

Pour résumer cet article, il propose une nouvelle approche d'apprentissage basé sur la confiance d'un algorithme permettant d'estimer la position de l'utilisateur à l'intérieur avec la force du signal WiFi. Il introduit une mesure de confiance, non seulement utile pour refléter l'incertitude des prédictions de positionnement, mais également capable d'ajuster la taille de l'ensemble de prédictions en conséquence.

Il a été montré empiriquement que la précision de positionnement était d'environ 2,4 m / probabilité de 75% avec un banc d'essai normal, environ 70 cm / probabilité de 75% avec un banc d'essai idéal et d'environ 8,8 m / probabilité de 75% avec un banc d'essai difficile.

Ces résultats ont surpassé les algorithmes de Machine Learning sans indice de confiance mis à l'essai sur les mêmes bancs d'essai jusqu'à 20% plus précis. Si l'on ne tient pas compte de CP, l'algorithme W-KNN est un peu meilleur que Naïve Bayes.

Les approches présentées dans cet article ne nécessitent pas de carte du bâtiment. Cela pourrait être utile de les avoir afin d'avoir des informations supplémentaires pour supprimer les mauvaises prédictions comme par exemple une personne qui marche dans un mur alors qu'elle est dans le couloir.

Table 4. Summary of the three fingerprinting test beds used in this article.

	Royal Holloway	Cambridge	UJIIndoorLoc
Abbreviation	RH	Cam	UJI
Training examples	13,600	78,000	19,937
Training area	1480 m ²	540 m ²	110,000 m ²
Surveyed space	3 corridors	1 corridor, 1 room	3 buildings
Training time	1 day	2 days	20 days
Granularity	1 m	10 cm for room 30 cm for corridor	Up to 2 m
Test samples	150	100	1,111
Last test sample	Same day	1 day later	3 months later
Measuring time	Working hours	Weekend	Working hours
Measures per location	200	40	Up to 30
Distinct positions	68	1950	933
Orientations per location	4	4	Unknown
Label type	Cartesian (metre scale)	Cartesian (metre scale)	Longitude and latitude
Label generator	Manually by the surveyor	Automatically by Active Bat system	Manually by the surveyor
Total building(s)	1	1	3
Total floor(s)	1	1	13
WiFi APs	131	43	529
Device(s) used	1 phone	1 netbook	25 phones
Surveyor(s)	1 person	1 robot	18 people

Figure 2.13. – Summary of the three fingerprinting test beds used in this article [KN17]

2.6. Choix et conclusion

Le choix de l'algorithme suite à l'analyse de l'état de l'art est une tâche difficile. Il est important de garder en tête l'application finale pour laquelle le système sera utilisé, c'est-à-dire, pour faire du positionnement intérieur.

En premier lieu, il est nécessaire de choisir entre un apprentissage par classification ou par régression. Il est clair que pour une utilisation finale un algorithme traitant de la régression comme SVM serait le mieux adapté. Ceci afin d'entrainer le système avec peu de positions et d'estimer les autres positions.

Cependant, il n'est pas vraiment réaliste de démarrer directement avec une technique pareil sans même savoir si les données disponibles sont exploitables. C'est pourquoi la décision a été de démarrer avec un algorithme de classification dans l'optique d'améliorer le système avec un algorithme de régression.

Suite, à l'étude des documents existants, il aurait été possible de sélectionner l'algorithme KNN qui a donné dans la plupart des cas de bons résultats. Mon choix c'est dirigé vers l'algorithme SVM qui est également bien classé. Ce choix a été fait en grande partie, car il est également cité pour la régression en tant que SVR alors que KNN est uniquement cité pour la classification.

3. Prise de mesures

Ce chapitre traite de la collecte des données qui est un point important de ce travail. Il a été nécessaire de réfléchir à quoi prendre et de quelle manière. Pour ce faire, je me suis servi d'un système de mesures de positionnement intérieur développé par M. Mueller dans le cadre de son travail de Master [Mic17]. Ci-dessous, ce programme sera détaillé plus précisément afin de pouvoir le prendre en main et réaliser les prises de mesures.

Il sera également précisé comment les données ont été structurées afin de pouvoir être utilisée plus tard pour alimenter l'algorithme d'apprentissage.

Finalement, il sera détaillé comment les mesures ont été effectuées selon le plan du laboratoire.

3.1. Plan de mesure

Un plan de mesure idéal avait été imaginé dans un premier temps pour effectuer les premiers essais, voir sur la Figure 3.2. Ce plan n'a pas pu être réalisé exactement de cette manière dû à la configuration de la salle et de la disposition des établis. Comme décrit ci-dessus, la configuration de la salle pour effectuer les mesures est composée de quatre esclaves (Slaves) placés dans les coins de la pièce à mesurer et le maître (master) est placé au centre. Ensuite, l'espion est déplacé à différents endroits pour effectuer la prise de mesures voir sur la Figure 3.8.

Afin de mieux comprendre la méthodologie de mesures, un cas réel a été imaginé. C'est-à-dire, que la réflexion a été faite en partant du principe qu'il faut positionner des objets dans un laboratoire et installer le système sans effectuer un quadrillage trop précis afin que le temps d'action soit le plus court possible.

Les points marqués par Sx (orange foncé) sont les points sur lesquels le système est basé et correspondent à la position de l'espion (spy). Quarante mesures ont été effectuées sur ces six endroits et c'est eux qui seront utilisés pour entraîner le système. Cinq mesures supplémentaires ont été effectuées à ces endroits pour les vérifications. Ces mesures ont été effectuées à des moments différents à l'exception de S5.

Les triangles bleus marqués Tx correspondent à des points de tests et cinq mesures ont été effectuées par point.

À noter que toutes ces mesures ont été effectuées en positionnant un mat muni d'un carton pour poser la carte de l'espion. Donc à chaque changement de position, il est possible que cette dernière varie de quelques centimètres et que l'orientation de l'antenne soit différente.

3.2. Setup pour la prise de mesures

Pour effectuer les prises de position, il est nécessaire d'effectuer un certain nombre de mesures afin d'avoir une quantité acceptable de données. Dans un premier temps, l'idée est de se concentrer sur le plan du laboratoire. Pour une première version d'apprentissage, les mesures seront effectuées dans une seule moitié du laboratoire comme indiqué sur la Figure 3.2.

Le point rond "M" représente le maître, les points ronds "S" représentent les esclaves et finalement les points carrés "E" représentent les espions. C'est sur ces derniers que les mesures seront effectuées.

Il sera nécessaire de prendre quarante mesures sur chaque point. Comme uniquement la première moitié sera considérée, les mesures seront faites sur six points différents donc 240 données seront à disposition. Une mesure consiste à changer les canaux de 1 à 40.

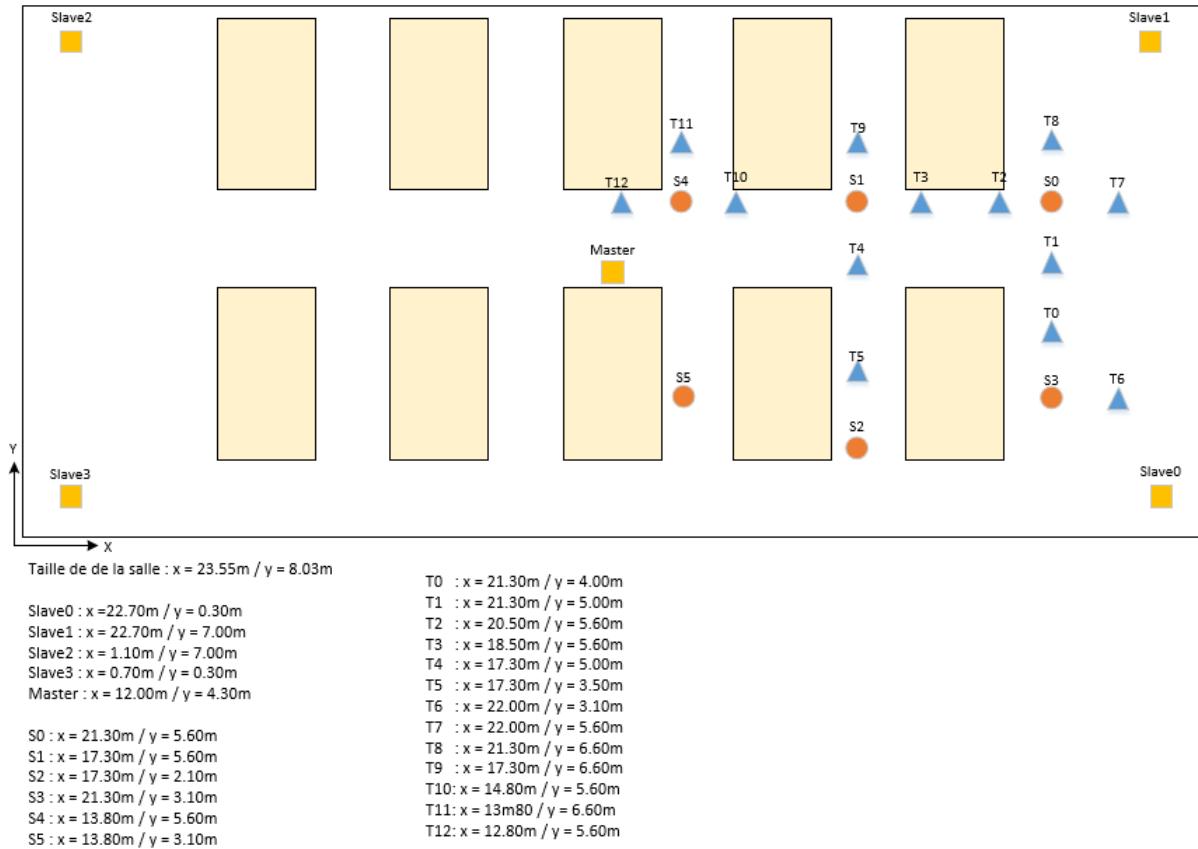


Figure 3.1. – Plan des mesures effectuées et du setup

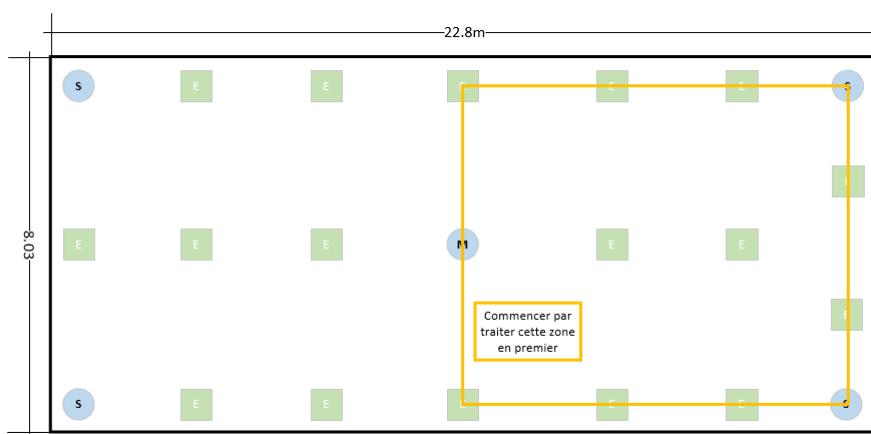


Figure 3.2. – Montre le setup pour la prise de mesure

Le plan de la Figure 3.2 et une simplification du plan réel de la Figure 3.3.

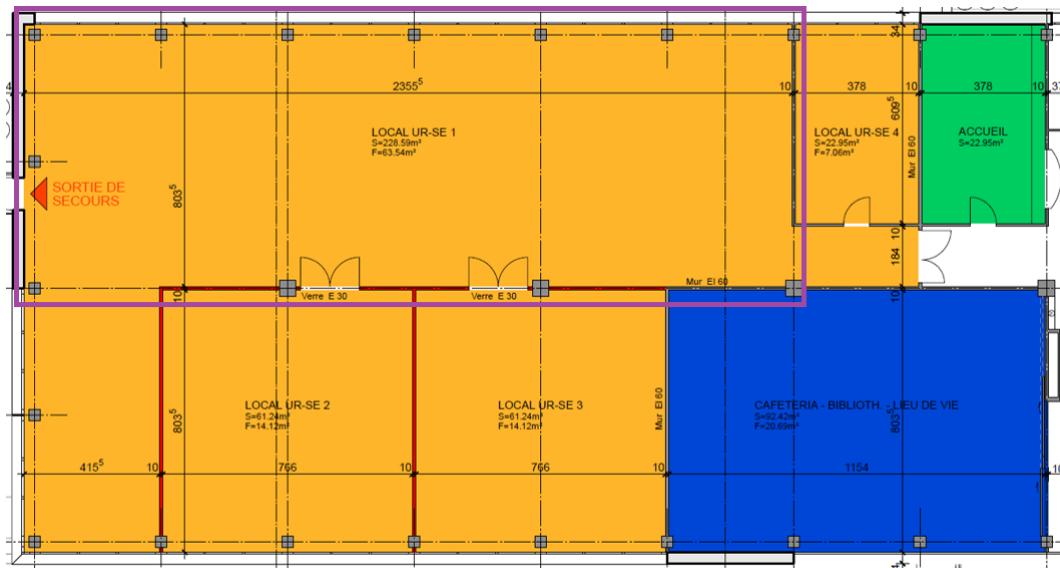


Figure 3.3. – Montre le plan réel du laboratoire

3.2.1. Matériel utilisé

Les cartes utilisées durant ce travail pour les noeuds du système de mesures sont présentées dans la Figure 3.4. Les noeuds sont : les quatre esclaves, le maître et l'espion. La carte provient de chez Semtech et utilise la radio SX1280, elle est interfacée sur une carte Nucleo-F410RB de chez STMicroelectronics.

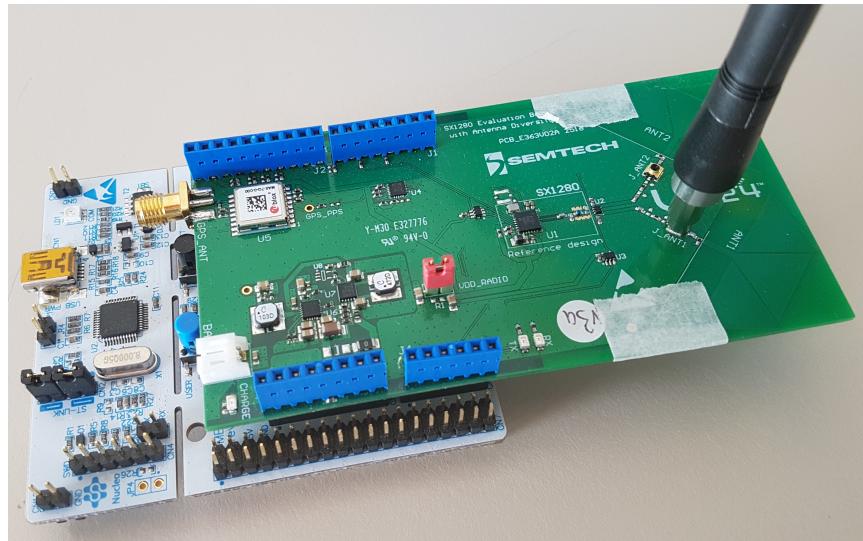


Figure 3.4. – Montre le plan réel du laboratoire

Chaque carte est configurée séparément pour fonctionner selon sa fonction (esclave, maître ou espion). Il suffit de connecter la carte à un ordinateur et de glisser dans le répertoire qui s'ouvre le fichier "image" correspondant à la fonctionnalité.

3.3. Programme de prise de mesure [Mic17]

Cette section introduit le programme de prise de mesure qui avait été développé. Ce programme permet avant tout d'exploiter pleinement le protocole de localisation, ainsi que les algorithmes de résolution de position.

Le système peut être composé d'un ou plusieurs esclaves et d'un ou plusieurs maîtres. Les espions font bien entendu partie intégrante du setup. Le programme effectue des mesures de distance, résout des positions des différents espions et les affiche dans une interface graphique. Le tout est configurable au travers d'un simple fichier de configuration au format texte.

Dans le cadre de ce projet, le fichier de configuration (stage.cfg) contient les informations (position, id,etc.) du "maître" ainsi que des quatre esclaves.

Le programme réalisé pour ce démonstrateur a dû être adapté, car il a été conçu pour effectuer des mesures en continu afin d'améliorer au fil du temps le calcul de la position. Cette façon de faire n'est pas utilisable dans le cadre d'un apprentissage à l'aide d'un algorithme.

3.3.1. Architecture logicielle

Pour avoir une meilleure compréhension la structure du programme de prise de mesures, la Figure 3.5 permet de visualiser les blocs principaux qui interagissent dans le calcul de position.

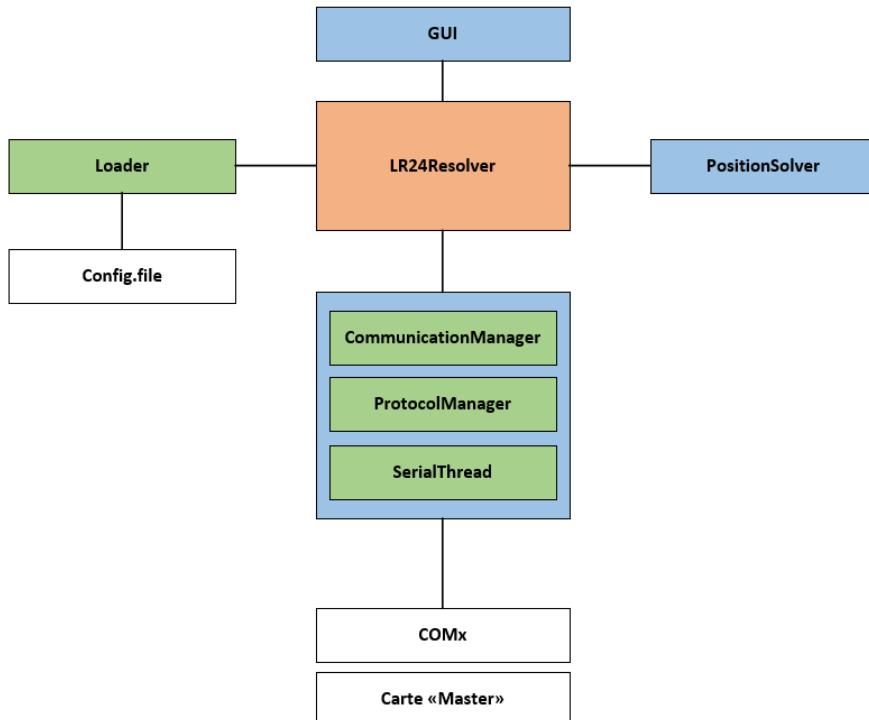


Figure 3.5. – Blocs principaux du logiciel de prise de mesure

Ci-dessous, une description de chaque bloc est présentée :

1. LR24Resolver : Partie centrale du démonstrateur (chef d'orchestre). Il instancie les autres blocs et gère les différents bursts de mesure et de communication).
2. Loader : Chargé de lire le fichier de configuration et de créer un jeu d'objets utilisables facilement par les autres blocs.

3. GUI : Interface graphique du système. Il affiche la position des maîtres (carrés verts), des esclaves (cercles verts), des espions (cercles jaunes) et les distances maître-esclave (cercles rouges).
4. PostionSolver : Chargé de résoudre la position des espions en se basant sur leurs mesures. Il utilise la méthode d'approximation aux moindres carrés avancés.
5. SerialThread : Communication avec la carte embarquée de type maître. Ce bloc utilise le port série standard. Il lit les données disponibles et les remontes à la couche supérieure. Il transmet aussi les bytes reçus de la couche supérieure vers le port série.
6. ProtocolManager : Gestion du format du protocole de communication. Cela signifie qu'il transforme les trames reçues en jeu d'objets utilisables, ou alors, il transforme des objets reçus en une trame valide.
7. CommunicationManager : Gestion des actions (mesure, communication, calibration, etc.). Il reçoit des actions et les exécute au travers du "ProtocolManager".

Une interface graphique est disponible avec le programme lors du lancement. Il est possible de démarrer une mesure, de l'interrompre et de la remettre à zéro, Figure 3.6.

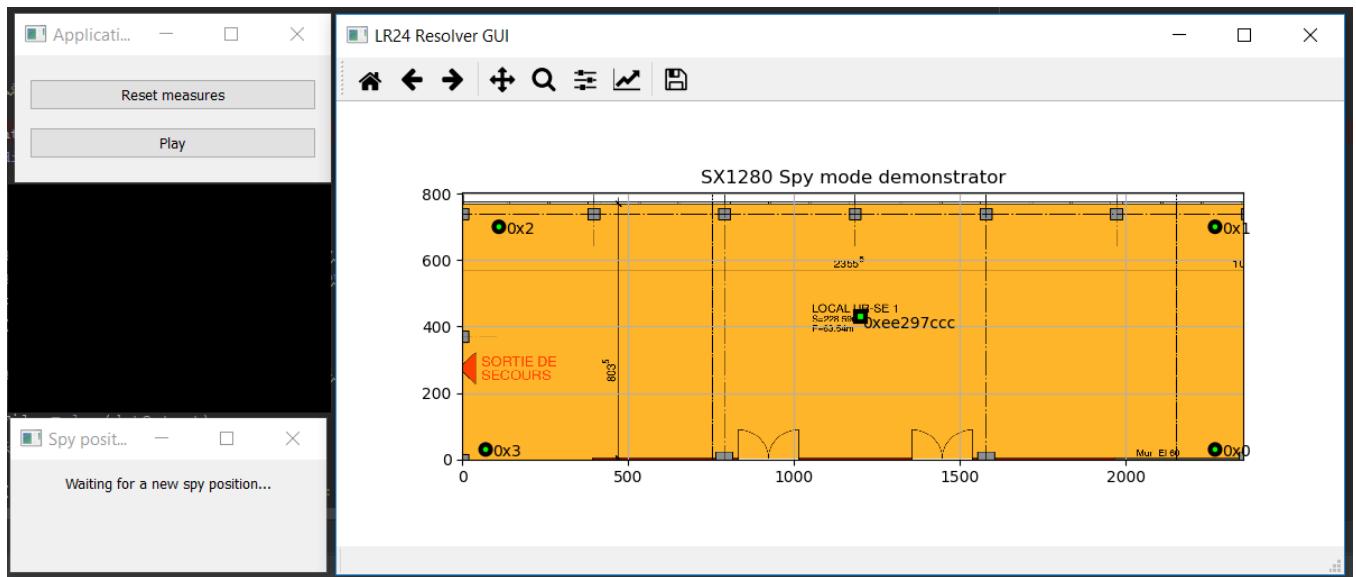


Figure 3.6. – Interface graphique du soft de prise de mesures

Ce programme est fait pour ordonner à la carte embarquée "maître" d'effectuer une série de mesures. Ensuite, dès que le maître reçoit une mesure de l'espion, il la transfère au programme qui tourne sur le PC. Les mesures réceptionnées sont sous forme de burst. Les informations utiles de ces bursts sont :

1. La provenance du burst de mesure (mesure effectuée sur quel esclave ?)
2. La mesure brute de différence de distance pour chaque esclave et pour quarante canaux de fréquence différents.
3. La mesure brute du RSSI

La position de l'espion est calculée au niveau du programme PC (PositionSolver) en fonction des données reçues vu précédemment. Selon le fichier de configuration, les mesures sont faites sur plusieurs canaux (ranging slot number = 40). Le système va effectuer les étapes suivantes en boucle en mesurant séquentiellement chaque esclave :

1. Mesure sur 40 canaux sur un esclave
2. Récupérer les données sur les maîtres
3. Transmettre les données au niveau du PC
4. Mesurer le deuxième esclave
5. Récupérer les données sur le master
6. Estimation de position quand cela est possible

La fonction qui réceptionne les burst entrant et par conséquent la structure des données se trouve dans le callback ci-dessous :

```

1 def new_burst_available(burst_list) :
2     """
3     Callback when new bursts are available
4     :param burst_list: list_of_bursts
5     :return: None
6
7     => It will calculate nodes new positions
8     """
9     for burst in burst_list:
10        position_solver.update_position_solver(burst)
11        system_gui.update_burst_info(burst)

```

Ce callback est reçu dans lr24_resolver.py. C'est une liste de burst (en python). En résumé, les burst reçus stock toutes les mesures pour un couple maître-esclave :

valeurs[0, canal] = mesure brute pour un canal donné
 valeurs[1, canal] = rssi pour un canal donné
 valeurs[2, canal] = erreur pour un canal

Ces données doivent être exploiter pour la suite du travail.

3.3.2. Prise de données brutes

La prise de données brutes consiste à mémoriser les valeurs brutes des informations de différences de distance fournie par l'espion. Cette mesure provient du fonctionnement des cartes embarquées qui sont configurées pour fonctionner en mode espion. Cette donnée se calcule en faisant $M_{Spy} = D_1 + D_2 - D_3$, voir Figure 3.7. Lorsque le système possède quatre esclaves, il y aura une mesure par esclave sur chaque fois les quarante canaux. C'est avec ces dernières qu'une position est estimée.

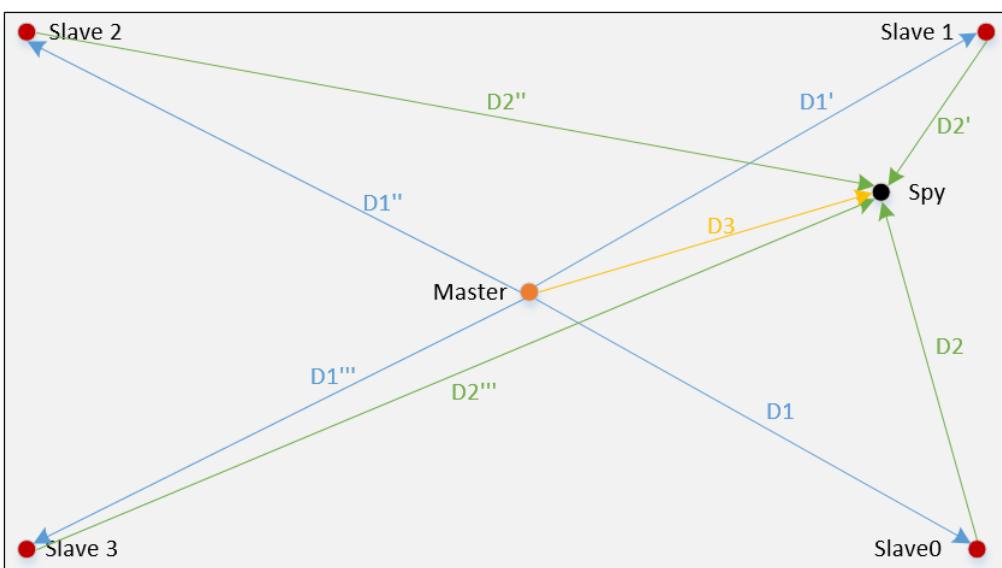


Figure 3.7. – Fonctionnement du mode spy et signification des mesures brutes

Modification du programme lr24_resolver.py

Le programme Lr24_resolver.py n'a pas pu être utilisé de la manière dont il avait été créé, car il était destiné à effectuer des mesures continues afin d'améliorer au fil du temps la précision. Dans le cadre de ce projet, il est nécessaire de pouvoir maîtriser la prise de mesures ainsi que gérer le nombre d'échantillons enregistrés. La Figure 3.8 montre de quelle manière sont prises les données.

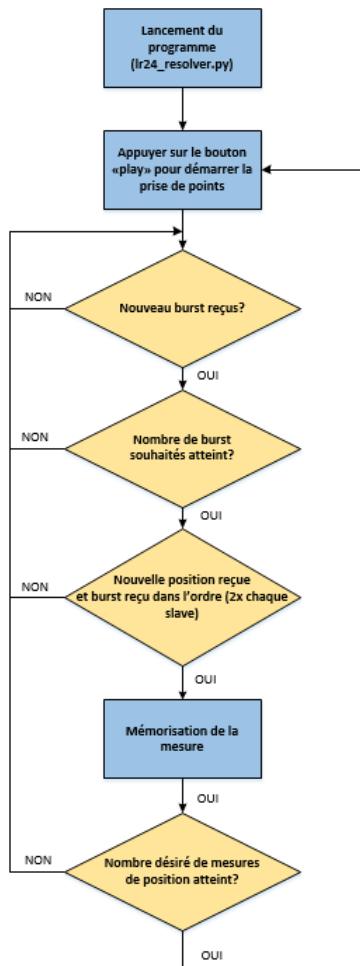


Figure 3.8. – Diagramme expliquant la prise de mesures des valeurs de ranging

Il a été nécessaire de modifier la fonction `new_burst_available(burst_list)` du fichier `Lr24_resolver.py` afin de mémoriser les huit premiers burst qui arrivent depuis la carte "maître". De ces "burst" sera uniquement mémorisée la valeur de la différence de distance calculée par l'espion. Cette différence de distance est obtenue grâce au mode "ranging" de la communication LoRa. Un "burst" comprend quarante informations qui sont liées aux quarante canaux de mesures.

Une fois que les huit bursts ont été reçus, le calcul de position, s'il existe, est mémorisé puis remis à zéro pour la mesure suivante. Il est possible de sélectionner le nombre de mesures désirées grâce à la variable "`measures_count_des`". Quand ce nombre est atteint, le programme s'arrête automatiquement, sinon il continue d'effectuer les mesures.

La fonction "`new_node_position_available(node_pos)`" a également été adaptée. Lorsqu'une mesure de position existe, il y a un contrôle qui est effectué afin de s'assurer que les mesures ont été effectuées dans un ordre précis qui est le suivant : Burst reçu de la mesure sur le slave0 (2x) puis du slave1 (2x) puis du slave2 (2x) et finalement du slave3 (2x). Cela afin de ne pas mélanger les données lors de l'apprentissage à l'aide de l'algorithme SVM.

Finalement, pour que les mesures soient automatiques avec le moins d'interaction possible avec l'utilisateur le fichier gui.py a été modifié pour y ajouter les fonctions software des boutons "stop", "start" et "reset".

Structure des données

Dans la Figure 3.9 est représenté comment sont structurées les mesures récupérées par le programme python (Lr24_resolver.py). Il a été nécessaire de faire une réflexion concernant quoi prendre comme mesure et de quelle manière. Il a été décidé de récupérer les informations ci-dessous. A noter qu'un fichier est créé par endroit de mesure, c'est-à-dire qu'à chaque changement de position un nouveau fichier est construit.

1. Index : Numéro de la mesure pour un point donné
2. X1 : Coordonnée X réelle par rapport aux repères de la salle
3. Y1 : Coordonnée Y réelle par rapport aux repères de la salle
4. X2 : Coordonnée X calculée par le programme python Lr24_resolver.py après avoir reçus huit bursts
5. Y2 : Coordonnée Y calculée par le programme python Lr24_resolver.py après avoir reçus huit bursts
6. meas : Données brutes de la valeur de "ranging" sur les quarante canaux reçues à chaque burst (8x40 valeurs)

Une décision a été prise à ce niveau de ne pas utiliser le RSSI. Effectivement, la prise de mesures est plus complexe qu'espérée et prend passablement de temps. Le RSSI aurait pu être intégré par la suite en pensant qu'une mesure était rapidement faite, mais ce n'a pas été le cas.

Sur la Figure 3.9 la partie "meas" est séparée en quatre parties, car les bursts sont reçus dans l'ordre deux fois pour l'esclave 0, deux fois pour l'esclave 1, deux fois pour l'esclave 2 et finalement deux fois pour l'esclave 3.

Index	X1	Y1	X2	Y2	Slave0 (2x 40 canaux)	Slave1 (2x 40 canaux)	Slave2 (2x 40 canaux)	Slave3 (2x 40 canaux)
'index': index + indexFile, 'x1': 15.2, 'y1': 5.60, 'x2': node_pos_node_pos[0], 'y2': node_pos_node_pos[1], 'meas': dctCapture								

Figure 3.9. – Structure des données des mesures de localisation

3.3.3. Prise de données de positions convergées

La prise de données des positions convergées est similaire à la prise des mesures brutes sauf que lors du lancement de la mesure, il faut laisser la mesure converger sur sa position finale et la mémoriser sans tenir compte des valeurs brutes des différents canaux.

Cette prise de mesure supplémentaire a été réalisée, car les résultats obtenus uniquement avec les valeurs brutes ne sont pas complètement satisfaisants (voir dans le chapitre suivant).

Modification du programme lr24_resolver.py

Pareil que ci-dessus, le programme Lr24_resolver.py n'a pas pu être utilisé comme il avait été créé, car il était destiné à effectuer des mesures continues afin d'améliorer au fil du temps la précision. Dans le cadre de ce projet, il est nécessaire de pouvoir maîtriser la prise de mesures. La Figure 3.10 montre de quelle manière sont prises les données de positions convergées.

Les modifications apportées au logiciel sont similaires à celles détaillées ci-dessus. La seule différence c'est qu'il a été nécessaire de regarder à partir de quel moment la mesure converge et effectuer ce nombre de mesures pour chaque point. La mesure dure environ trois minutes pour obtenir une position convergée ce qui correspond à la réception de 120 burst.

Le résultat de la position est mémorisé à la suite des 120 burst.

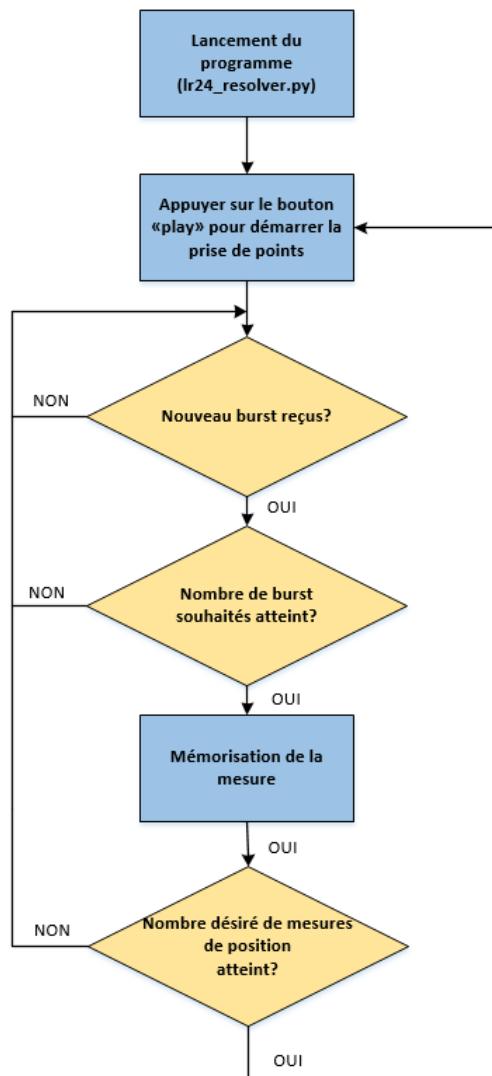


Figure 3.10. – Diagramme expliquant la prise de mesures des positions convergées

Structure des données

La structure des données voir la Figure 3.11 est similaire à la partie ci-dessus sauf que les données brutes ne sont pas mémorisées, car leur nombre serait trop important.

1. Index : Numéro de la mesure pour un point donné
2. X₁ : Coordonnée X réelle par rapport aux repères de la salle
3. Y₁ : Coordonnée Y réelle par rapport aux repères de la salle
4. X₂ : Coordonnée X calculée par le programme python Lr24_resolver.py après avoir convergée (120 bursts)
5. Y₂ : Coordonnée Y calculée par le programme python Lr24_resolver.py après avoir convergée (120 bursts)

Index	X1	Y1	X2	Y2
'index': index + indexFile, 'x1': 15.2, 'y1': 5.60, 'x2': node_pos_node_pos[0], 'y2': node_pos_node_pos[1]				

Figure 3.11. – Structure des données de mesures contenant uniquement la valeur des positions convergées

4. Traitement des données et classification

Comme décidé lors de l'analyse de l'état de l'art et les réflexions de départ, la première analyse qui est faite, est de faire de la classification avant de tester les résultats de la régression.

L'algorithme qui a été sélectionné pour effectuer cette tâche est SVM (Support Vector Machines). Ce chapitre va décrire les différentes étapes qui ont été réalisées pour traiter les données et utiliser l'algorithme.

La programmation a été réalisée sur l'environnement PyCharm PROFESSIONAL 2018.2 et dans le langage Python. Cet environnement m'est le plus familier, car il est étudié dans différents cours, c'est pourquoi mon choix s'est porté sur ce dernier.

4.1. Structure du traitement

Pour le traitement de données pour un apprentissage à l'aide d'un algorithme il est conseillé d'avoir différentes étapes voir Figure 4.1.

1. Acquisition des données : Étape délicate et très importante dans ce projet.
2. Prétraitement : Étape de traitement sur les données brutes se trouvant dans le jeu de données.
3. Extraction des caractéristiques : Étape qui consiste à extraire les caractéristiques qui seront utilisées par l'algorithme pour faire la reconnaissance.
4. Reconnaissance : Utilisation d'un algorithme pour effectuer la reconnaissance des positions.
5. Décision : Cette étape consiste, si cela est nécessaire, à décider du résultat final à l'aide d'une fusion, cela n'a pas été utilisé.

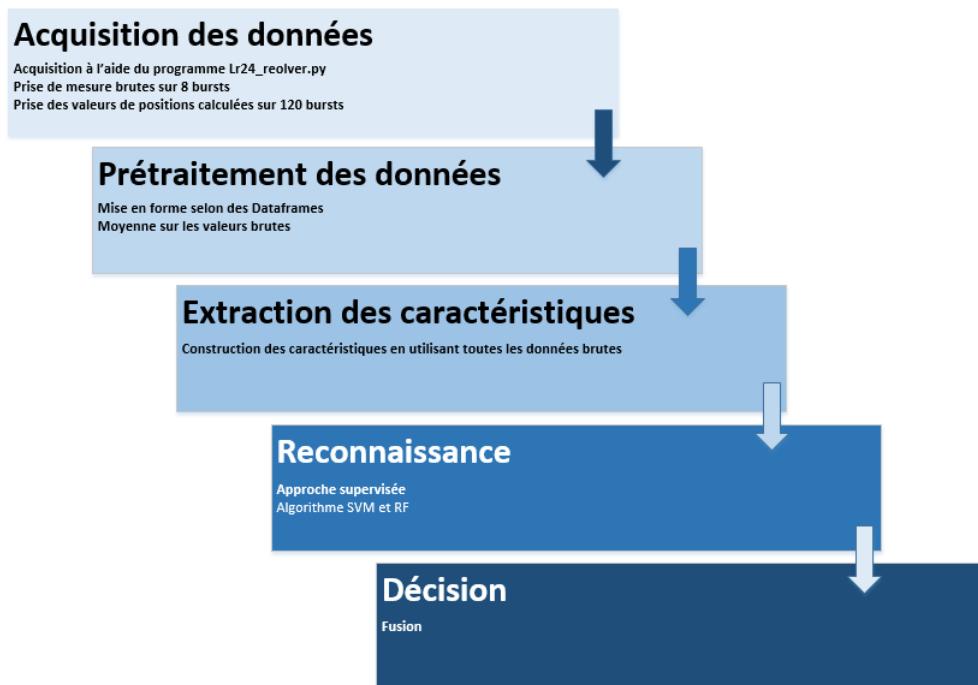


Figure 4.1. – Flux de traitement de l'information

Le premier point est l'acquisition de données qui est décrit au chapitre précédent. C'est une étape primordiale, car c'est grâce à ces données que toute la suite de l'analyse va dépendre. Dans le cadre de ce projet, on remarque vite que si les données ne sont pas prises de manière rigoureuse, cela ne fonctionne pas.

Le prétraitement consiste à utiliser les données disponibles et de les exploiter. Dans un premier temps, les données sont lues depuis un fichier *.npy qui représente des données NumPy. NumPy est une extension du langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux. Plus précisément, cette bibliothèque logicielle libre (open source) fournit de multiples fonctions permettant notamment de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes [Wikc].

Depuis ce fichier, plusieurs traitements sont effectués afin de transformer les données et les mettre dans un format permettant un traitement simple, le format "Dataframe" a été choisi et appartient à la librairie Pandas. Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles [Wikd]. À partir de là, il est possible d'utiliser les données afin d'en extraire des caractéristiques qui est l'étape suivante. Ce point sera détaillé plus loin afin de mieux comprendre ce qui a été utilisé pour que l'algorithme puisse au mieux reconnaître les différentes positions.

L'étape de reconnaissance consiste à entraîner un système avec les caractéristiques sélectionnées à partir des données et ensuite de valider que notre algorithme est capable de reconnaître de nouvelles mesures. Ce point est également détaillé plus bas.

Finalement, dans ce type d'analyse, il existe souvent une partie décision. Cela peut se faire, car deux algorithmes différents sont utilisés et il est nécessaire de décider quel est le meilleur résultat. Dans le cadre de ce projet, il n'a pas été nécessaire d'utiliser cela, car il n'a pas été possible dans le temps imparti de tester plusieurs techniques et donc la fusion n'était pas utile.

4.2. Extraction des caractéristiques

Ce chapitre traite des caractéristiques qui seront utilisées par l'algorithme afin de déterminer à quelle classe elles appartiennent. La Figure 4.2 montre comment se présentent les données. Il est possible de les séparer et d'isoler chaque mesure. Chaque mesure est composée des données pour les quarante canaux (colonne : Canal). Les mesures qui peuvent être exploitées pour effectuer la classification sont les mesures brutes de différence de distance ($Sx.x$). Il serait aussi possible d'exploiter la position fournie par le programme de prise de mesures, mais cette dernière n'est pas assez représentative. C'est pour cette raison que deux types de classification ont été effectués.

La première consiste à utiliser les données brutes et de travailler avec ces dernières pour extraire d'autres caractéristiques. Les principales qui ont été testées sont décrites ci-dessous.

1. Toutes les données (raw) : Cela consiste à prendre toutes les mesures brutes de tous les canaux et de les mettre les unes après les autres. Ce qui veut dire 320 valeurs (quarante canaux fois 2x chaque esclave).
2. Un canal (raw) : Cette façon de faire est identique à la précédente sauf qu'un seul canal est pris en compte ce qui fait 8 valeurs (un canal fois 2x chaque esclave).
3. Moyenne : La moyenne est faite sur les quarante canaux par esclave ce qui va donner huit caractéristiques supplémentaires.
4. Variance : La variance est une mesure qui permet de caractériser la dispersion d'un échantillon. Elle est utilisée pour voir la dispersion des valeurs par esclave sur les quarante canaux.
5. Déviation standard : La déviation standard équivaut à la racine carrée de la variance. C'est donc une mesure de dispersion qui est faite également par esclave sur les quarante canaux.
6. Quantile : Un quantile correspond à séparer les données en partie de taille égale. C'est-à-dire que chaque partie doit contenir le même nombre de données. Cela est également utilisé pour séparer les données au niveau des canaux pour un esclave.

7. Médiane : La médiane est en quelque sorte le quantile qui sépare des données en deux parties de même taille.
8. covariance : La covariance permet de voir la corrélation entre des variables. Cette mesure est effectuée par esclave sur tous les canaux.

La deuxième consiste à utiliser uniquement la position fournie par le programme de prise de mesure. Cette position est mémorisée après avoir convergée et ce qui fait que les caractéristiques disponibles sont la coordonnée X et la coordonnée Y. Comme il n'y a que deux caractéristiques, il sera difficile d'effectuer différents traitements sur ces données si l'algorithme n'arrive pas à différencier les classes.

Les résultats de ces deux différentes approches seront détaillés au chapitre suivant.

	Pos	Meas	Canal	s0.0	s0.1	s1.0	s1.1	
25	0.00000	0.00000	25.00000	147.00000	-1136.00000	1334.00000	-739.00000	
26	0.00000	0.00000	26.00000	701.00000	-26.00000	838.00000	394.00000	
27	0.00000	0.00000	27.00000	615.00000	-77.00000	460.00000	-324.00000	
28	0.00000	0.00000	28.00000	510.00000	298.00000	715.00000	255.00000	
29	0.00000	0.00000	29.00000	497.00000	-406.00000	1277.00000	-259.00000	
30	0.00000	0.00000	30.00000	598.00000	-137.00000	614.00000	514.00000	
31	0.00000	0.00000	31.00000	297.00000	-20.00000	465.00000	-28.00000	
32	0.00000	0.00000	32.00000	780.00000	1869.00000	-358.00000	-388.00000	
33	0.00000	0.00000	33.00000	901.00000	162.00000	384.00000	-511.00000	
34	0.00000	0.00000	34.00000	1110.00000	260.00000	763.00000	5.00000	
35	0.00000	0.00000	35.00000	1067.00000	498.00000	-163.00000	-492.00000	
36	0.00000	0.00000	36.00000	1189.00000	383.00000	718.00000	-230.00000	
37	0.00000	0.00000	37.00000	1.00000	450.00000	328.00000	-107.00000	
38	0.00000	0.00000	38.00000	1117.00000	-14.00000	823.00000	173.00000	
39	0.00000	0.00000	39.00000	-144.00000	61.00000	1721.00000	329.00000	
40	0.00000	1.00000	0.00000	1274.00000	848.00000	829.00000	373.00000	
41	0.00000	1.00000	1.00000	993.00000	299.00000	1221.00000	-9.00000	
42	0.00000	1.00000	2.00000	993.00000	-847.00000	1235.00000	-292.00000	
43	0.00000	1.00000	3.00000	986.00000	581.00000	801.00000	47.00000	
44	0.00000	1.00000	4.00000	424.00000	-143.00000	1652.00000	208.00000	
45	0.00000	1.00000	5.00000	226.00000	394.00000	1959.00000	1062.00000	
46	0.00000	1.00000	6.00000	794.00000	1384.00000	1785.00000	1646.00000	
47	0.00000	1.00000	7.00000	989.00000	480.00000	1630.00000	1124.00000	
48	0.00000	1.00000	8.00000	925.00000	-165.00000	2212.00000	1327.00000	
49	0.00000	1.00000	9.00000	1044.00000	528.00000	877.00000	220.00000	
	s2.0	s2.1	s3.0	s3.1	x1	y1	x2	y2
826.00000	444.00000	1175.00000	-1379.00000	21.30000	5.60000	21.01397	6.65830	
400.00000	469.00000	1169.00000	0.00000	21.30000	5.60000	21.01397	6.65830	
1932.00000	776.00000	2558.00000	1646.00000	21.30000	5.60000	21.01397	6.65830	
1373.00000	1175.00000	1192.00000	698.00000	21.30000	5.60000	21.01397	6.65830	
485.00000	338.00000	1323.00000	480.00000	21.30000	5.60000	21.01397	6.65830	
446.00000	1422.00000	1025.00000	1260.00000	21.30000	5.60000	21.01397	6.65830	
1051.00000	693.00000	330.00000	-182.00000	21.30000	5.60000	21.01397	6.65830	
988.00000	1181.00000	817.00000	1416.00000	21.30000	5.60000	21.01397	6.65830	
438.00000	390.00000	778.00000	463.00000	21.30000	5.60000	21.01397	6.65830	
867.00000	540.00000	485.00000	537.00000	21.30000	5.60000	21.01397	6.65830	
667.00000	1584.00000	490.00000	988.00000	21.30000	5.60000	21.01397	6.65830	
-383.00000	283.00000	1022.00000	572.00000	21.30000	5.60000	21.01397	6.65830	
727.00000	884.00000	760.00000	1232.00000	21.30000	5.60000	21.01397	6.65830	
497.00000	842.00000	891.00000	983.00000	21.30000	5.60000	21.01397	6.65830	
909.00000	1116.00000	1006.00000	1154.00000	21.30000	5.60000	21.01397	6.65830	
182.00000	798.00000	-26.00000	1962.00000	21.30000	5.60000	21.16969	3.75498	
860.00000	285.00000	670.00000	1013.00000	21.30000	5.60000	21.16969	3.75498	
974.00000	907.00000	703.00000	965.00000	21.30000	5.60000	21.16969	3.75498	
1108.00000	791.00000	1139.00000	726.00000	21.30000	5.60000	21.16969	3.75498	
1015.00000	746.00000	531.00000	633.00000	21.30000	5.60000	21.16969	3.75498	
941.00000	688.00000	408.00000	1465.00000	21.30000	5.60000	21.16969	3.75498	
1386.00000	1077.00000	1082.00000	768.00000	21.30000	5.60000	21.16969	3.75498	
633.00000	847.00000	348.00000	990.00000	21.30000	5.60000	21.16969	3.75498	
643.00000	-2748.00000	181.00000	482.00000	21.30000	5.60000	21.16969	3.75498	
1466.00000	1270.00000	899.00000	1050.00000	21.30000	5.60000	21.16969	3.75498	

Figure 4.2. – Données sous forme de dataframe pour le traitement et l'extraction des caractéristiques

4.3. Reconnaissance - Algorithme

Le choix de l'algorithme a été fait par rapport à l'état de l'art qui a été effectué en début de projet. Comme déjà mentionné, l'algorithme qui a été retenu est SVM (Support Vector Machine). Cette solution permet de laisser le projet évoluer en utilisant dans un premier temps la solution pour la classification et ensuite en utilisant SVM pour la régression.

Support Vector Machine est un algorithme supervisé qui permet de faire de la classification en essayant de trouver le meilleur hyperplan. La librairie scikit learn a été utilisée et je me suis servie de sklearn.svm.SVC (System Vector Classification).

L'implémentation de l'algorithme est facile une fois que les labels et les caractéristiques ont été extraits. Dans un premier temps, le canevas de base a été mis en place en utilisant les données brutes des mesures de "ranging". Cela a permis de faire les premiers réglages.

Afin d'entraîner et de tester l'efficacité de l'algorithme, il est nécessaire de partager les données en trois jeux. Il y a un jeu qui est utilisé pour l'entraînement, un jeu qui est utilisé pour valider l'entraînement qui vient d'être réalisé et un troisième jeu qui sert uniquement pour le résultat final. Dans ce dernier jeu, ce sont des données que l'algorithme n'a jamais vues. La Figure 4.3 permet d'illustrer cette séparation. IL est également possible de voir que les données d'entraînement et de validation vont de pair. Pour ma part, j'ai décidé d'effectuer une cross-validation. C'est-à-dire qu'à chaque lancement le partage des données n'est jamais identique. La seule chose qui est respectée est qu'il y aura toujours 20% de données pour la validation et 80% pour l'entraînement. Une autre chose qui est à vérifier c'est que les jeux de données soient balancés. Cela veut dire qu'il doit y avoir le même nombre de données de chaque classe dans chaque jeu. Cela donnerait de mauvais résultats si l'entraînement est fait avec les classes 1,2,et 3 et que la validation se fait avec des données des classes 4 et 5 par exemple.

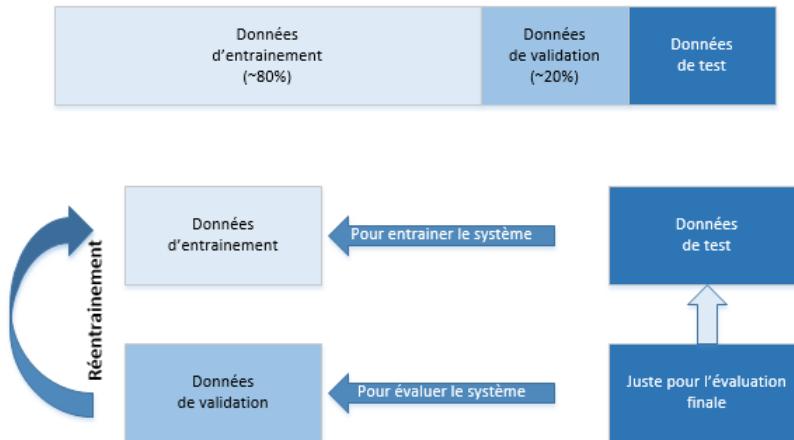


Figure 4.3. – Partage des données

4.3.1. Hyper paramètres

L'algorithme SVM utilise plusieurs paramètres qu'il a été nécessaire de chercher pour obtenir les meilleurs. Seuls le Kernel, le paramètre C et le paramètre gamma ont été recherchés. Une forte valeur de C tente de minimiser les erreurs de classification des données d'entraînement et une faible valeur essaie de maintenir une classification lisse. Concernant le gamma, plus il est grand , plus la cloche de la gaussienne est étroite voir Figure 4.4.

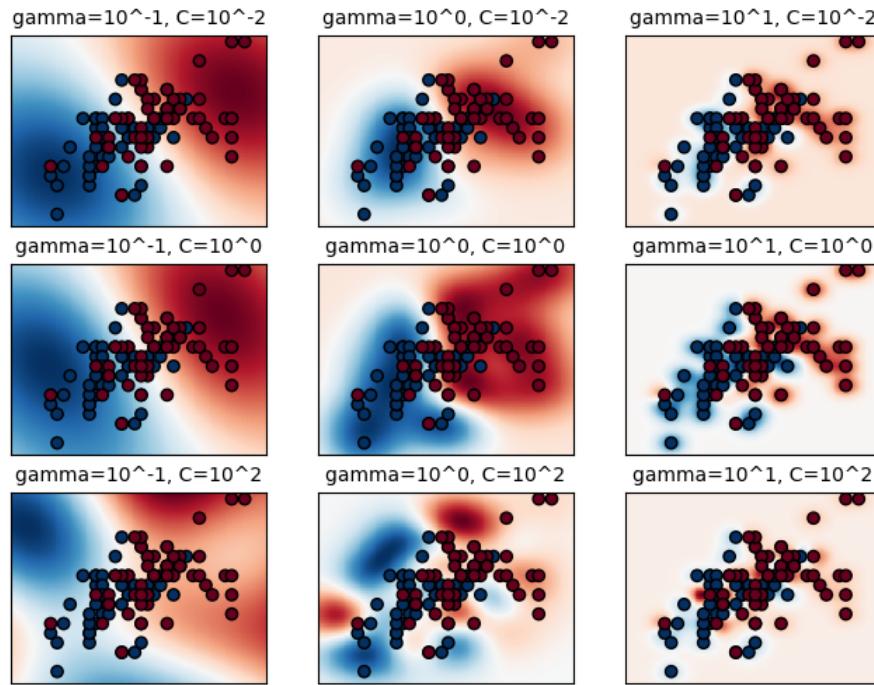


Figure 4.4. – Influence des paramètres C et gamma

La première chose qui a été faite a été de trouver le meilleur Kernel pour cette classification. Les Kernel suivants ont été testés : linear, rbf, poly, sigmoid. Celui qui a donné les meilleurs résultats est le Kernel linear.

Suite à cela, un grid-search est effectué sur les paramètres C et gamma. Les meilleurs résultats obtenus sont avec un C égal à 1 et un gamma égal à 0.001.

4.4. Validation et résultats

Cette section va présenter l'évolution des résultats obtenus au cours du développement. Voici un résumé des données acquises selon la Figure 3.8 :

1. Mesures de "ranging" effectuées sur les points Sx qui correspondent aux points que l'on souhaite retrouver (40x sur chaque point).
2. Mesures de "ranging" effectuées sur les points Sx à un moment différent que les quarante mesures précédentes (5x sur chaque point).
3. Mesures de "ranging" effectuées sur les points Tx. Ces points sont pris afin de voir si le plus proche voisin est retrouvé.
4. Mesures de position effectuées sur les points Sx (40x + 5x sur chaque point).
5. Mesures de position effectuées sur les points Tx (5x sur chaque point)

Ces différentes prises de données permettent de faire plusieurs analyses et comprendre de façon plus claire comment le système se comporte.

Les résultats sont évalués à l'aide des matrices de confusion, mais également à l'aide de la mesure de précision, du F1-Score macro et finalement du F1-Score micro.

La Figure 4.5 montre ce que représente la précision et le rappel. Ces deux notions sont utilisées pour calculer le F1-Score de la manière suivante :

$$\text{F1_score} = 2((\text{Précision} * \text{rappel}) / (\text{Précision} + \text{rappel}))$$

Le micro ou macro-avarage sont utilisés pour analyser les données de manière différente. Pour le calcul du macro-avarage, il faut additionner les précisions de chaque classe et les diviser par le nombre de classes. Dans ce cas, il se pourrait qu'une classe soit très mauvaise et déséquilibrée, mais que les autres remontent le score. Si maintenant on parle du micro-avarage, le calcul se fait en additionnant les vrais positifs et en les divisant par le nombre total de points. Cette manière de faire va donner un score plus bas si une classe est déséquilibrée [Dat].

Dans le cadre de cette analyse, toutes les classes possèdent le même nombre de données.

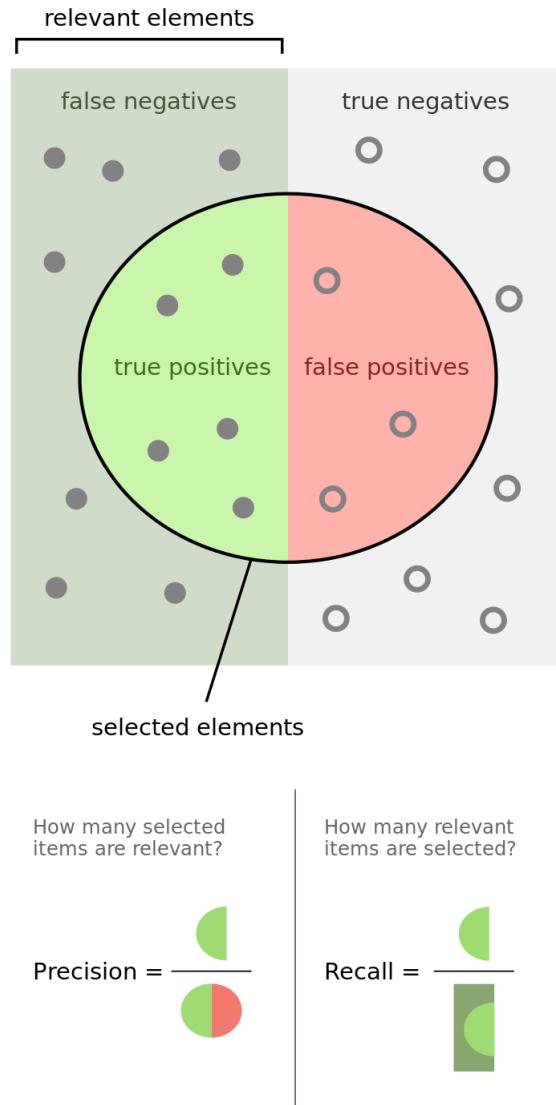


Figure 4.5. – Représentation de la précision et du rappel [WIKI]

4.4.1. Résultats en utilisant les données de "ranging"

Ce chapitre va présenter les résultats qui ont été obtenus à partir des données brutes de "ranging" fournies par le programme de mesures. Plusieurs traitements sont faits sur ces mesures afin d'améliorer les résultats. Le calcul de position est également disponible, mais n'est pas très représentatif, car calculé uniquement à partir de huit valeurs de ranging (deux sur chaque esclave).

Afin d'avoir une meilleure vue de ce que représentent les positions calculées sur peu d'échantillons, la Figure 4.6 montre le positionnement de ces points calculés par rapport à la position réelle de l'espion. Les gros points de couleurs représentent la position réelle et les petits points de couleurs représentent les positions calculées. La croix noire représente la position du master. On remarque que les petits points sont bien plus espacés que lorsqu'on laisse converger la position (Figure 4.15), mais il y a tout de même une séparation entre les ensembles.

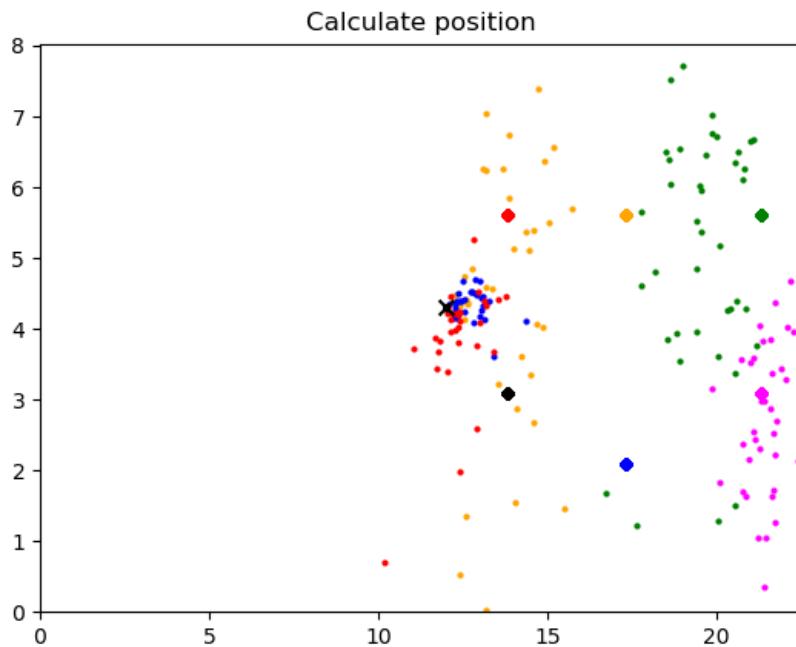


Figure 4.6. – Représentation des points non convergé par rapport aux positions réelles

La Figure 4.6 met en évidence un phénomène qui tend à faire rapprocher les points sur la position du maître. Cet effet qui se trouve dans le laboratoire où les mesures ont été effectuées complique la reconnaissance de position.

Ci-dessous, une énumération des couleurs et de la position associée :

1. Couleur verte (S0) : $x = 21.30m / y = 5.60m$
2. Couleur jaune (S1) : $x = 17.30m / y = 5.60m$
3. Couleur bleue (S2) : $x = 17.30m / y = 2.10m$
4. Couleur rose (S3) : $x = 21.30m / y = 3.10m$
5. Couleur rouge (S4) : $x = 13.80m / y = 5.60m$
6. Couleur noire (S5) : $x = 13.80m / y = 3.10m$

Comme décrit plus haut, toute une série de tests ont été effectués afin de trouver d'une part les meilleurs paramètres et d'autre part de trouver les caractéristiques les plus probantes. Dans tous les essais réalisés, seuls, trois seront détaillés ci-dessous : Résultats obtenus en utilisant uniquement les données de "ranging", résultats obtenus en utilisant uniquement les données de position et finalement présentation de la solution retenue qui utilise la médiane, les données "ranging", les quantiles et la variance.

Résultat en utilisant toutes les données de ranging brutes

Le premier test qui a été effectué a été de prendre toutes les données brutes des valeurs de ranging. Cela consiste à prendre chacune des mesures des quatre esclaves sur les quarante canaux et de les mettre les unes à côté des autres pour en créer une caractéristique. Cela est effectué pour chacune des mesures (quarante par position).

La Figure 4.7 présente le résultat obtenu lorsque l'entraînement est fait avec 32 des 40 mesures sur une même position et testé avec les huit restants. Le résultat obtenu est impressionnant et offre une précision entre 95% et 100%. Les positions sont très bien reconnues. La variation de pourcentage pour la reconnaissance vient du fait qu'à chaque lancement, le programme exclut huit autres mesures.

La matrice de confusion représentée est donnée avec des mesures qui ont été faites sur un même point sans bouger l'espion. Le F1-score macro est de 0.9791 et un F1-score micro de 0.9792.

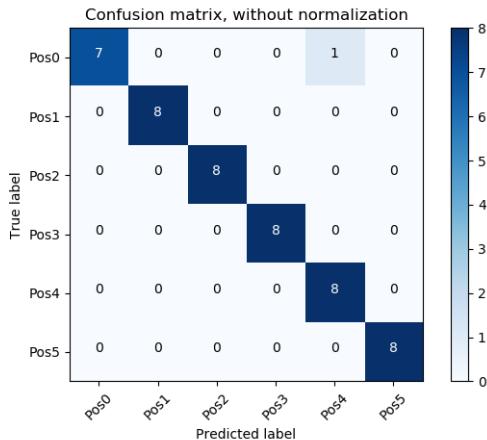


Figure 4.7. – Matrice de confusion obtenue en utilisant les points Sx et les caractéristiques de "ranging"

Afin de valider le système, cinq points supplémentaires par position ont été pris durant une autre période et en ayant probablement une position à peine différente. Il faut préciser que les mesures sur la position 5 ont été effectuées au même moment tant pour les quarante que les cinq mesures. C'est pour cette raison que la détection de cette position est très bonne.

Cela montre une précision de 46.67% ce qui correspond à 14 positions correctement reconnues sur 30. Ces résultats ne sont pas encourageants, car cela veut dire que sitôt que l'espion est positionné différemment, mais au même endroit cela influence grandement la reconnaissance. Il est donc nécessaire d'effectuer un travail supplémentaire pour améliorer cette reconnaissance soit niveau prise de mesure soit niveau du travail sur les caractéristiques. Ce qui a été obtenu dans ce test n'était pas ce qui était attendu au vu des résultats obtenus lors de l'entraînement dans la Figure 4.7.

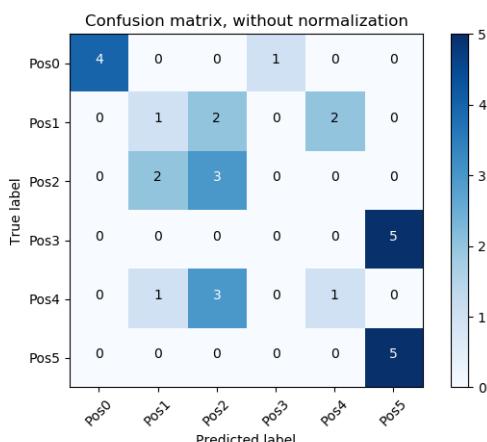


Figure 4.8. – Matrice de confusion obtenue en utilisant les points de tests SxT et les caractéristiques de "ranging" de la 1ère et 2ème mesures de l'esclave

Deux mesures sont effectuées par esclave pour les quarante canaux. L'exemple précédent prend en compte toutes ces mesures. Afin de voir l'effet de ces mesures, un essai a été de prendre uniquement chaque première mesure et ensuite chaque seconde mesure. Étonnement, cela a amélioré les résultats comme le montre la Figure 4.9. Les premières mesures offrent de mauvais résultats alors que prendre uniquement les deuxièmes mesures améliore le score. La précision obtenue est de 53.33% ce qui correspond à 16 positions correctement reconnues sur 30. Difficile à comprendre pourquoi il y a une amélioration. Afin de clarifier cette influence, il serait intéressant de faire plus de deux mesures par esclave et regarder les différents résultats.

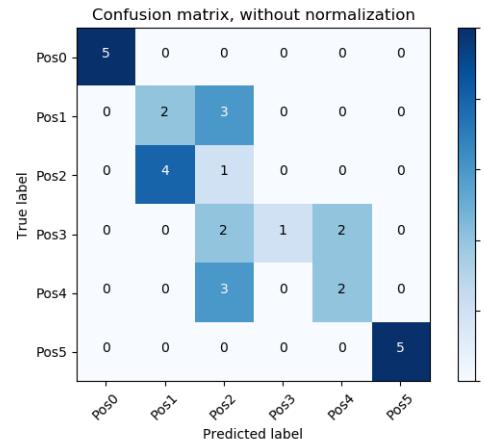


Figure 4.9. – Matrice de confusion obtenue en utilisant les points de tests SxT et les caractéristiques de "ranging" de la 2ème mesure de l'esclave uniquement

Résultat en utilisant la position non convergé

Le résultat qui est présenté ici permet de faire une comparaison entre une mesure qui est convergée (présenté au chapitre suivant) et une mesure qui est calculée uniquement à partir de huit mesures. C'est sans surprise que les résultats qui sont obtenus sont vraiment moins bons.

La Figure 4.11 présente le résultat obtenu lorsque l'entraînement est fait avec 32 des 40 mesures sur une même position et testé avec les huit restants. Le résultat obtenu est impressionnant et offre une précision d'environ 70% contre environ 85% pour la solution avec les positions convergées.

La matrice de confusion représentée est donnée avec des mesures qui ont été faites sur un même point sans bouger l'espion. La précision est de 72.91%, le F1-score macro est de 0.7102 et un F1-score micro de 0.7293. 83.33

Afin de valider le système, cinq points supplémentaires par position ont été pris durant une autre période et en ayant probablement une position à peine différente. Il faut préciser que les mesures sur la position cinq ont été effectuées au même moment tant pour les quarante que les cinq mesures. C'est pour cette raison que la détection de cette position est très bonne.

Cela montre une précision de 43.33% ce qui correspond à 13 positions correctement reconnues sur 30 alors que les résultats pour les positions convergées donnent 83.33% qui correspond à une reconnaissance correcte de 25 positions sur 30.

Meilleur résultat obtenu

Ce chapitre va présenter le meilleur résultat qui a été obtenu ainsi que les caractéristiques qui ont été utilisées. Pour y arriver, les caractéristiques suivantes ont été testées de manière individuelle :

1. Toutes les données "ranging" (raw)

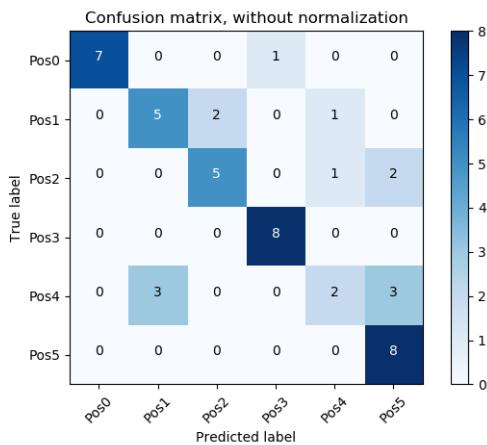


Figure 4.10. – Matrice de confusion obtenue en utilisant les points Sx et la caractéristique de position

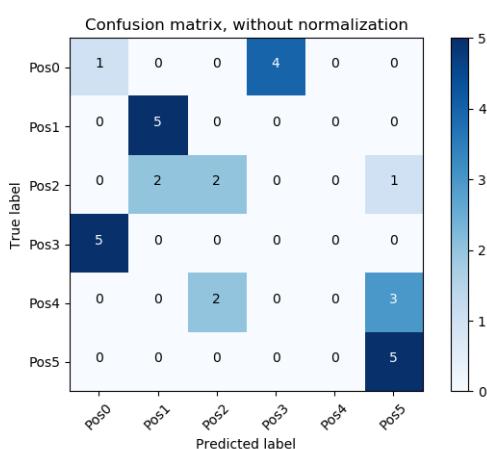


Figure 4.11. – Matrice de confusion obtenue en utilisant les points SxT et la caractéristique de position

2. Donnée "ranging" d'un canal (raw)
3. La moyenne des canaux
4. La variance sur les canaux
5. Déviation standard sur les canaux
6. Quantile effectué sur les canaux
7. Médiane effectuée sur les canaux
8. Covariance des canaux

Ensuite, plusieurs associations ont été faites entre ces différents tests et le meilleur des résultats est obtenu en associant la valeur de la médiane, de la variance, du quantile et des données de "ranging". À noter que pour avoir le meilleur résultat, il a été nécessaire de prendre en compte uniquement la deuxième mesure de ranging sur chaque esclave. Lorsque les différentes caractéristiques sont affichées, il est à noter que l'amplitude des valeurs n'est pas du même ordre et par conséquent pour tenter de les égaliser, la médiane a été multipliée par cinq, le quantile par six et la variance a été divisée par quatre. Cette façon de faire a encore amélioré les résultats pour obtenir ceux présentés ci-dessous.

La Figure 4.12 présente le résultat obtenu lorsque l'entraînement est fait avec 32 des 40 mesures sur une même position et testé avec les huit restants. Cela donne de très bonnes précisions qui se situent entre 83% et 98%. Les positions sont majoritairement bien reconnues. La variation de pourcentage pour la reconnaissance vient du fait qu'à chaque lancement, le programme exclut huit autres mesures.

La matrice de confusion représentée est donnée avec des mesures qui ont été faites sur un même point sans bouger l'espion. La précision dans ce cas est de 97.92% avec un F1-score macro de 0.9791 et un F1-score micro de 0.9792.

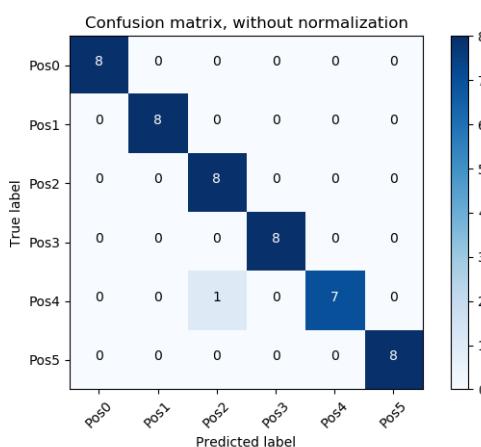


Figure 4.12. – Matrice de confusion obtenue pour la meilleure solution en utilisant les données des points Sx

Maintenant, afin de valider le système cinq points supplémentaires par position ont été pris durant une autre période de la journée et en ayant probablement une position à peine différente. La présentation des précédents résultats montre de très mauvais résultats concernant ce test, mais après le travail effectué et la recherche des meilleures caractéristiques les résultats sont satisfaisants et encourageants. Il faut préciser que les mesures sur la position 5 ont été effectuées au même moment tant pour les quarante que les cinq mesures. C'est pour cette raison que la détection de cette position est excellente.

Cela montre une précision de 73.33% ce qui correspond à 22 positions correctement reconnues sur 30.

Un dernier test a été réalisé afin de voir si les positions Tx sont bien reconnues par rapport à leur plus proche voisin. La Figure 4.14 montre les résultats obtenus lorsque les positions Tx sont utilisées. Les points T_{1/2/7/8} correspondent à la position S₀, T_{3/4/9} correspondent à la position S₁, T₅ correspond à la position S₂, T_{6/10} correspondent à la position S₃ et finalement, T_{10/11/12} correspondent à la position S₄. Comme attendu, les résultats pour cette classification ne sont pas bons et pas utilisables. La précision obtenue est de 31.25% ce qui représente une détection correcte de 20 positions sur 64. Pour obtenir ce résultat, on essaie de classifier les positions Tx selon la position entraînée la plus proche.

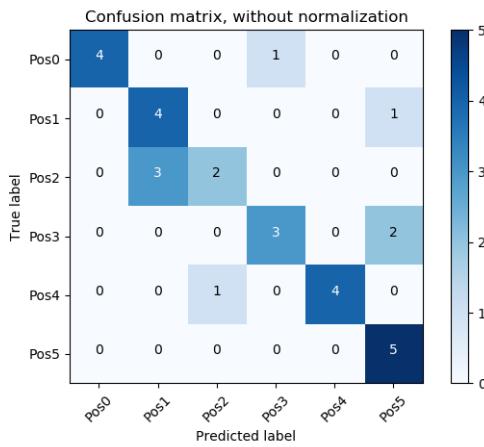


Figure 4.13. – Matrice de confusion pour la meilleure solution en utilisant les points de tests SxT

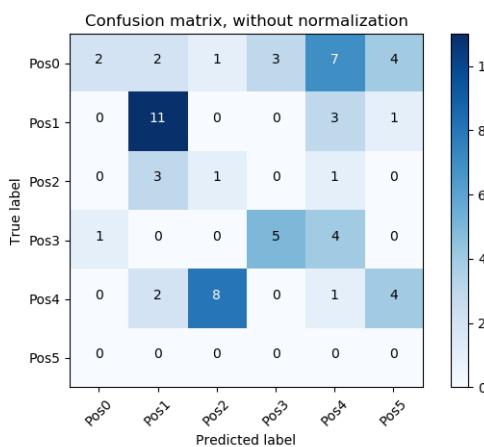


Figure 4.14. – Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés autour des points entraînés

4.4.2. Résultats en utilisant les données de position convergées

Ce chapitre va présenter les résultats qui ont été obtenus à partir des approximations de position fournies par le programme de mesures. Dans cette partie uniquement ces valeurs seront utilisées. Afin d'avoir une meilleure vue de ce que représentent ces positions, la Figure 4.15 montre le positionnement des points calculés par rapport à la position réelle de l'espion. Les gros points de couleurs représentent la position réelle et les petits points de couleurs représentent les positions calculées. La croix noire représente la position du maître.

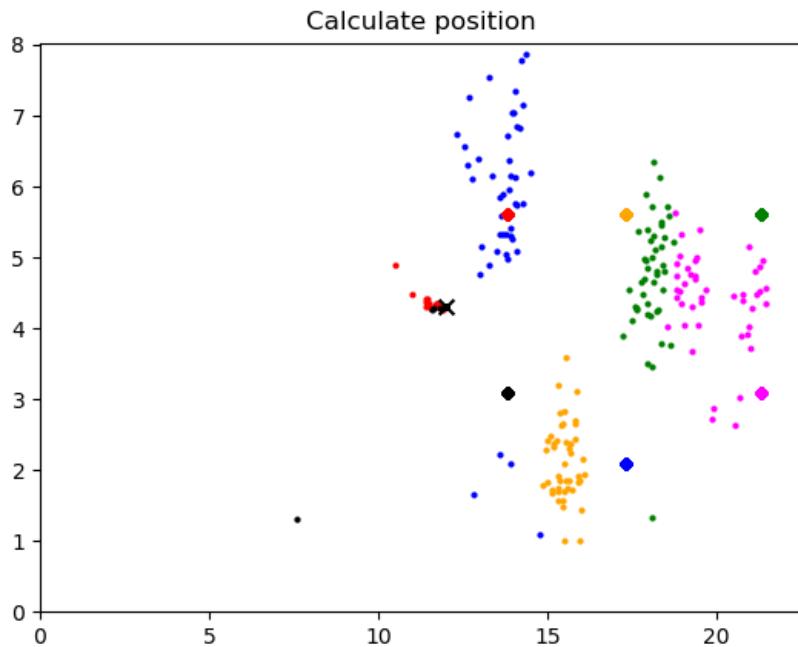


Figure 4.15. – Représentation des points convergés par rapport aux positions réelles

La Figure 4.15 met en évidence un phénomène qui tend à faire correspondre les points proches du maître sur les coordonnées du maître. Cet effet qui se trouve dans le laboratoire où les mesures ont été effectuées complique la reconnaissance de position. De par cette représentation, il est possible de voir à l'oeil nu la séparation des points. Par contre, il est rapidement possible de se rendre compte que si nous ajoutons des points il sera de plus en plus difficile à différencier les emplacements de base de l'espion.

Ci-dessous, une énumération des couleurs et de la position associée :

1. Couleur verte (S0) : $x = 21.30m / y = 5.60m$
2. Couleur jaune (S1) : $x = 17.30m / y = 5.60m$
3. Couleur bleue (S2) : $x = 17.30m / y = 2.10m$
4. Couleur rose (S3) : $x = 21.30m / y = 3.10m$
5. Couleur rouge (S4) : $x = 13.80m / y = 5.60m$
6. Couleur noire (S5) : $x = 13.80m / y = 3.10m$

La Figure 4.16 représente le résultat obtenu lorsque l'entraînement est fait avec 32 des 40 mesures sur une même position et testé avec les huit restants. Cela donne de très bonnes précisions qui se situent entre 81% et 89%. Comme attendu, les positions proches du maître (S4 et S5) se confondent et pas conséquent ne sont pas correctement reconnues. Les autres positions sont majoritairement bien reconnues à l'exception de quelques mesures. La variation de pourcentage pour la reconnaissance vient du fait qu'à chaque lancement, le programme exclu huit autres mesures.

Afin de tester l'algorithme fixé, il est entraîné avec les quarante mesures effectuées sur chaque position de l'espion. Ces quarante mesures sont celles utilisées pour définir les meilleurs paramètres et obtenir les

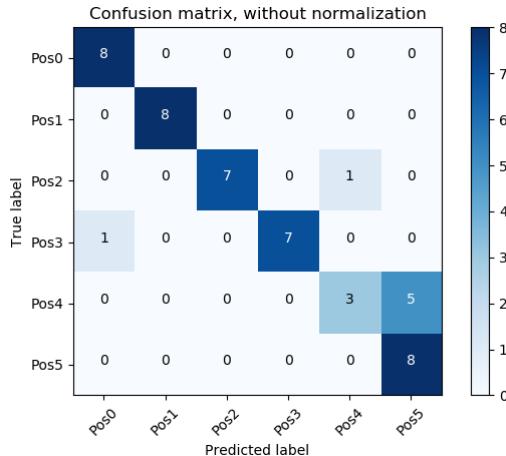


Figure 4.16. – Matrice de confusion pour les valeurs de la position convergée avec des points positionnés à la même place

meilleurs résultats comme présentés dans la Figure 4.15. La Figure 4.17 présente le résultat final obtenu avec les cinq données de test qui avaient été effectué à la même position que les quarante de l'entraînement.

La précision obtenue pour ce test est de 83.33% ce qui correspond à une reconnaissance correcte de 25 positions sur 30. Les positions mal reconnues sont sans surprise les positions S4 et S5.

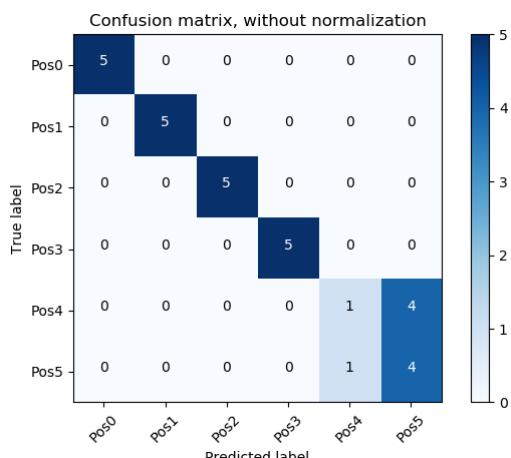


Figure 4.17. – Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés à la même place

La Figure 4.18 montre les résultats obtenus lorsque les positions Tx sont utilisées. Les points T1/2/7/8 correspondent à la position S0, T3/4/9 correspondent à la position S1, T5 correspond à la position S2, T6/10 correspondent à la position S3 et finalement, T10/11/12 correspondent à la position S4. Comme attendu, les résultats pour cette classification ne sont pas bons et pas utilisables. La précision obtenue est de 21.67% ce qui représente une détection correcte de 13 positions sur 60. Pour obtenir ce résultat on essaie de classifier les positions Tx selon la position entraînée la plus proche.

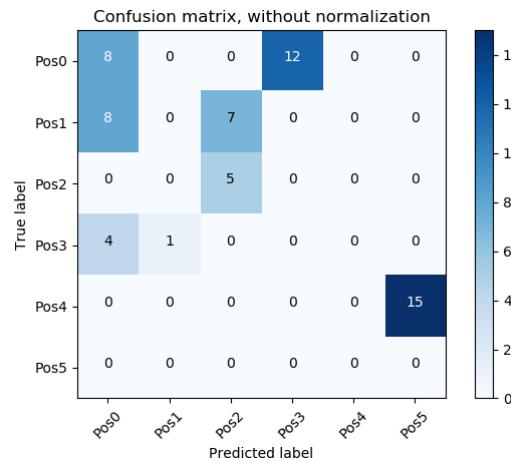


Figure 4.18. – Matrice de confusion pour les valeurs de la position convergée avec des points de tests positionnés autour des points entraînés

4.5. Résumé des résultats

La Figure 4.19 résume les résultats présentés dans ce rapport. La solution la plus optimiste a été mise en vert. Ce n'est pas celle qui donne les meilleurs résultats au niveau des points SxT, mais malgré cela c'est elle qui offre la plus grande liberté de traitements et d'améliorations.

La solution qui utilise uniquement la position convergée donne effectivement de bons résultats, mais si le nombre de points est augmenté ils vont commencer de se mélanger et il ne sera plus possible de les différencier. Les positions proches du maître seront toujours confondues alors qu'avec la solution d'utiliser les données brutes ces positions sont différenciables. Lorsque les données brutes de "ranging" sont utilisées il est possible d'utiliser des traitements sur ces mesures comme la moyenne, la déviation standard, la variance, etc. chose qui n'est pas possible lorsqu'on possède uniquement l'information des coordonnées x et y.

Caractéristiques utilisées	Précision obtenue sur les mesures à des positions 100% identiques (Sx)	Précision obtenue sur les positions identiques mais un autre jour (SxT)	Précision obtenue sur les positions entourant les positions d'entraînement (Tx)
Toutes les données "ranging"	97.92%	46.67%	-
Données de "ranging" 2ème mesure	97.92%	53.33%	-
Données de positions non convergées	72.92%	43.33%	-
Données de "ranging" 2ème mesure / médiane / quantile / variance	97.92%	73.33%	31.25%
Données de positions convergées	85%	83.33%	21.67%

Figure 4.19. – Résumé des différents résultats obtenus qui sont présentés dans ce rapport

Un test supplémentaire a été effectué afin de comparer un autre algorithme. Celui utilisé est "random forest", c'est un des algorithmes cité et utilisé dans l'état de l'art. Les résultats obtenus sont inférieurs avec une précision à 50% ce qui représente une reconnaissance de 15 positions correctes sur 30 contre 22 sur 30 avec l'algorithme SVM.

5. Problèmes rencontrés et améliorations

Ce chapitre traite des problèmes qui ont été rencontrés durant de ce travail d'approfondissement ainsi que des améliorations possibles.

Le projet est basé sur une thèse de Master que Michael Mueller avait réalisé [Mic17]. Il avait mis en place un programme permettant de calculer une position dans un local intérieur. Réutiliser ce programme devait permettre d'acquérir facilement des mesures, mais cela n'a pas été le cas. Dans un premier temps, il a été nécessaire de modifier le programme, car le système de prise de mesures n'était pas adapté. Cela dans le sens qu'il effectuait des mesures en permanence en stockant les résultats et en recalculant à chaque fois une position par rapport à toutes les mesures.

Ce qui a été problématique est la stabilité du système qui à tout moment s'arrête de fonctionner et ne donne plus de résultats utilisables. Les esclaves cessaient de fonctionner de temps en temps et cela faisait qu'une mesure prenait un temps conséquent. Il a aussi été remarqué que lorsque le système fonctionnait durant de longues heures d'affilées les bugs étaient de plus en plus nombreux et le système finissait par n'être plus utilisable.

De plus, lorsque les mesures de positions convergées ont été réalisée afin d'obtenir une série de mesures sur une position (40 mesures sur les 40 canaux), il fallait une demi-journée. Durant ce temps, il faut sans cesse surveiller le système afin de vérifier que les esclaves ne s'éteignent pas.

Au niveau des améliorations, il serait important de travailler sur le programme de prise de mesures afin de comprendre pourquoi le système s'arrête parfois. Actuellement, la prise est automatique, mais si un des esclaves s'arrête, il faut le redémarrer et cela ne peut pas être fait de manière automatique.

Un autre point serait de travailler sur un premier tri des mesures afin d'utiliser uniquement celles qui donnent de bons résultats et supprimer les autres (outlier). Cela permettrait d'avoir de meilleurs résultats finaux. Dans le même ordre d'idée, supprimer les canaux qui seraient bruités.

Concernant la compréhension du système complet, faire les mesures dans un environnement maîtrisé où les réflexions/absorption sont limitées (ce qui n'est pas le cas dans le laboratoire) serait d'une grande aide. La quantité de mesures prises doit également être augmentée afin de pouvoir poursuivre le travail et se pencher sur une analyse par régression.

Une réflexion devrait être faite concernant le positionnement des esclaves et du maître. Il serait éventuellement intéressant d'ajouter un second maître et de les placer différemment, par exemple aux extrémités plutôt qu'au centre du local.

Finalement, l'intégration de la mesure du RSSI pourrait être un candidat très intéressant pour compléter l'analyse et améliorer les prédictions.

6. Conclusion

L'objectif de ce travail était d'étudier la possibilité d'améliorer la précision du positionnement en intérieur à l'aide d'algorithmes d'apprentissage intelligents. La technologie LoRa 2.4GHz a été utilisée pour la mesure de position. Cette technologie possède un mode appelé "ranging" et c'est celui-ci qui a été utilisé durant le projet.

Les mesures ont été prises dans le laboratoire de la HE-ARC à St-Imier. A la suite des résultats obtenus, il aurait été plus judicieux dans un premier temps de faire des mesures dans un local avec moins de reflexions/absorbtions comme une halle de gymnastique par exemple. Les trois noeuds pour la prise de mesure (maître, esclave et espion) ont été placés en hauteur afin d'éviter au maximum les obstacles et être en ligne de mire ("line-Of-Sight"). Dans le cadre d'un déploiement en production, les noeuds ne seront pas forcément en ligne de mire, mais cela a permis de tester et analyser le système en enlevant des perturbateurs.

Les résultats obtenus sont encourageants pour les mesures qui sont faites et testées sur un même point. Cependant, sitôt que la mesure est faite à côté de ces points, l'algorithme n'est plus capable de trouver la position finement. Cela montre qu'il serait nécessaire de quadriller la pièce plus précisément, d'ajouter et fusionner d'autres caractéristiques, comme le RSSI ou améliorer la méthode de prétraitement sur les données récupérées en identifiant, par exemple, des outliers ou des canaux bruités.

Ce travail montre qu'il est possible de positionner précisément un point à un endroit qui a été entraîné alors que sa mesure réelle varie fortement de sa position réelle. Concernant un positionnement à l'aide de la régression, il reste un travail conséquent à faire sur la prise de mesures pour obtenir des résultats. L'étude de cette voie n'a pas été réalisée dans le cadre de ce projet d'approfondissement.

Bibliographie

- [LLI] Mainetti LUCA, Patrono LUIGI et Sergi ILARIA. *A Survey on Indoor Positioning Systems*. Rapp. tech. University of Salento Lecce, ITALY (cf. p. 1).
- [FP17] Bernat Carbones FARGAS et Martin Nordal PETERSEN. « GPS-free Geolocation using LoRa in Low-Power WANs ». In : *Proceedings of 2017 Global Internet of Things Summit (GIoTS)* (2017) (cf. p. 4, 7).
- [Ped+11] F. PEDREGOSA et al. « Scikit-learn : Machine Learning in Python ». In : *Journal of Machine Learning Research* 12 (2011), p. 2825-2830 (cf. p. 3, 6).
- [KN17] KHUONG et NGUYEN. « A performance guaranteed indoor positioning system using conformal prediction and the WiFi signal strength ». In : *Journal of Information and Telecommunication*, 1 :1, 41-65, DOI : 10.1080/24751839.2017.1295659 (2017). URL : <https://doi.org/10.1080/24751839.2017.1295659> (cf. p. 4, 11-14).
- [JR14] Quinlan J. R. « C4. 5 : programs for machine learning ». In : *Elsevier* (2014) (cf. p. 4).
- [GHP95] John G. H. et Langley P. « Estimating Continuous Distributions in Bayesian Classifiers ». In : *11th Conference on Uncertainty in Artificial Intelligence*, pp., 338-345 (1995) (cf. p. 4).
- [CS13] Anuradha C. et Dhall S. « Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm ». In : (2013) (cf. p. 4).
- [WA13] Yotsawat W. et Srivihok A. « Inbound tourists segmentation with combined algorithms using K-Means and Decision Tree ». In : *10th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp.189-194 (2013) (cf. p. 4).
- [SS14] Ureerat S. et P. SINGSRI. « The classifier model for prediction quail gender after birth based on external factors of quail egg ». In : *IEEE 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)* (2014) (cf. p. 4).
- [DL07] Yang D. et Jin-lin L. « Research on personal credit evaluation model based on bayesian network and association rules ». In : *International Conference on Wireless Communications, Networking and Mobile Computing* (2007) (cf. p. 5).
- [DWDMK91] Aha D. W., Kibler D. et Albert M. K. « Instance-based learning algorithms ». In : *Machine Learning*, vol. 6, pp., 37-66 (1991) (cf. p. 5).
- [CAG13] Shah C. et Jivani A. G. « Comparison of data mining classification algorithms for breast cancer prediction ». In : *Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)* (2013) (cf. p. 5).
- [J.98] Platt J. « Fast Training of Support Vector Machines using Sequential Minimal Optimization ». In : *Advances in Kernel Methods - Support Vector Learning* (1998) (cf. p. 5).
- [PH14] Niken P. et Ohwada H. « Applicability of machine-learning techniques in predicting customer defection ». In : *IEEE 2014 International Symposium on Technology Management and Emerging Technologies (ISTMET)* (2014) (cf. p. 5).
- [SMKN13] Obaidullah S. M., Roy K. et Das N. « Comparison of different classifiers for script identification from handwritten document ». In : *IEEE International Conference on Signal Processing, Computing and Control (ISPCC)*, pp.1-6 (2013) (cf. p. 5).
- [YRE96] Freund Y. et Schapire R. E. « Experiments with a new boosting algorithm ». In : *3th International Conference on Machine Learning, San Francisco*, pp. 148-156 (1996) (cf. p. 5).
- [RRE13] Shams R. et Mercer R. E. « Classifying Spam Emails Using Text and Readability Features ». In : *IEEE 13th International Conference on Data Mining (ICDM*, pp. 657-666 (2013) (cf. p. 5).

- [SOLISP10] Sharif S. O., Kuncheva L. I. et Mansoor S. P. « Classifying encryption algorithms using pattern recognition techniques ». In : *IEEE International Conference on Information Theory and Information Security (ICITIS)*, , pp. 1168-1172 (2010) (cf. p. 5).
- [L.96] Breiman L. « Bagging predictors ». In : *Machine Learning*. vol. 24, no. 2, pp.123-140 (1996) (cf. p. 5, 6).
- [Hen+12] Wymeersch HENK et al. *A Machine Learning Approach to Ranging Error Mitigation for UWB Localization*. Rapp. tech. Belgian American Education Foundation, the Charles Stark Draper Laboratory Robust Distributed Sensor Networks Program, the Office of Naval Research Young Investigator Award N00014-03-1-0489, the National Science Foundation under Grants ANI-0335256 and ECCS-0636519., 2012 (cf. p. 6, 10, 11).
- [CCo6] Rasmussen C. et Williams C. « Gaussian Processes for Machine Learning ». In : *Springer* (2006) (cf. p. 6).
- [VT94] Barnett V. et Lewis T. « Outliers in Statistical Data ». In : *3rd ed. Wiley Series in Probability and Mathematical Statistics* (1994). URL : <https://doi.org/10.1080/24751839.2017.1295659> (cf. p. 7).
- [Esd] « lien concernant les ESD ». In : (). URL : <https://www.itl.nist.gov/div898/handbook/eda/section3/eda35h3.htm> (cf. p. 7).
- [Sin+15] Bozkurt SINEM et al. *A Comparative Study on Machine Learning algorithms for Indoor Positioning*. Rapp. tech. This work is supported by The Scientific et Technological Research Council of Turkey (TUBITAK) under grant number 1130024, 2015 (cf. p. 7-10).
- [Wika] « lien concernant la regression non paramétrique ». In : (). URL : https://fr.wikipedia.org/wiki/R%C3%A9gression_non_param%C3%A9trique (cf. p. 10).
- [Mic] « lien concernant le concours Microsoft ». In : (). URL : <https://www.microsoft.com/en-us/research/event/microsoft-indoor-localization-competition-ipsn-2014/> (cf. p. 12).
- [Mic17] Mueller MICHAEL. « Système de localisation et de tracabilité des objets communicant basse consommation ». Mém. de mast. HES-SO Master, 2017 (cf. p. 15, 18, 41).
- [Wikc] « lien concernant NumPy ». In : (). URL : <https://fr.wikipedia.org/wiki/NumPy> (cf. p. 26).
- [Wikd] « lien concernant Pandas ». In : (). URL : <https://fr.wikipedia.org/wiki/Pandas> (cf. p. 26).
- [Dat] « lien concernant le score F1 ». In : (). URL : <https://datascience.stackexchange.com/questions/15989/micro-average-vs-macro-average-performance-in-a-multiclass-classification-settin/16001> (cf. p. 30).
- [Wikb] « lien concernant le score F1 ». In : (). URL : https://en.wikipedia.org/wiki/F1_score, https://fr.wikipedia.org/wiki/Pr%C3%A9cision_et_rappel.

Appendix

Annexe A.

Cahier des charges

1 Introduction

Ce document est le cahier des charges pour le travail d'approfondissement d'Informatique-Systèmes Embarqués 2018 en MRU-TIC à la HE-Arc.

2 Nom du projet

AI-enhanced LoRa Based Indoor Localization System

3 Contexte

3.1 Introduction

GPS-based location systems suffer from accuracy deterioration and are almost unavailable in indoor environments. For indoor environments, many technologies for positioning systems have been devised based on vision, infrared or ultrasound sensing, earth magnetic fields, accelerometers/gyrometers, BLE beacons or WiFi communication. Each of these existing technologies has a related costs, accuracy and maximum coverage trade-off, but a generic indoor location service remains elusive today.

Mandant

n/a

4 Cahier des charges

4.1 But du projet

Building upon the ranging capabilities of newer LoRa ICs, this project will develop and deploy a location system able to enhance the position accuracy reached by existing LoRa-based location systems relying on TDOA or ranging mechanisms. For this purpose, an exploration and comparison of different machine learning/deep learning techniques for fingerprinting based positioning will be performed.

4.2 Spécifications

Les spécifications suivantes doivent être prises en compte :

- Michael Müller. Système de localisation et traçabilité des objets communicants basse consommation. TM MSE-TIC, février 2017

4.3 Travail à réaliser :

Le travail consiste à:

- Etudier le cahier des charges.
- Etudier l'état de l'art des techniques à utiliser dans le cadre du projet, en particulier les systèmes de localisation indoor basés sur des techniques d'apprentissage, et réunir une documentation (env. 20% effort)
- Etablir un planning pour l'ensemble du projet.
- Définir un plan des tests à effectuer.
- Définir les procédures de test.
- Définir le setup pour la collecte de données de localisation
- Prise en main de l'environnement de développement pour les phases de training et du test de la technique d'apprentissage retenue (e.g., PyTorch).
- Implémentation de la solution ML retenue.
- Tester le système selon le protocole préétabli.
- Faire des propositions pour améliorer les performances de l'algorithme et, si possible, les implémenter.
- Rédiger le rapport et documenter l'ensemble du projet.

Délivrables :

- Selon Fiche TA « MSE_HES_SO_Fiche PA_V5.1.pdf », annexée à ce cahier des charges.
- Un guide d'installation et d'utilisation.
- Un démonstrateur fonctionnel.

4.4 Organisation du travail

Chaque deux semaines, l'étudiant envoie un mini-rapport sur maximum deux côtés de pages A4 qui présente le travail réalisé dans les deux dernières semaines de travail et une planification de ce qui sera fait dans les 2 prochaines semaines.

Une rencontre avec le professeur aura lieu au maximum toutes les deux semaines.

4.5 Lieu de travail

Le lieu de travail est laissé au choix de l'étudiant.

Une place lui est néanmoins réservée dans le laboratoire de systèmes embarqués de St-Imier.

4.6 Modification du cahier des charges

Ce cahier des charges est susceptible d'évoluer en fonction des propositions qui seront faites par les différents acteurs du projet.

4.7 Contraintes administratives

L'ensemble des contraintes administratives du projet sont décrit dans les documents :

MSE-Calendrier-academique-2018-2019-8979.pdf

MSE_HES_SO_Fiche PA_V5.1.pdf

4.8 Confidentialité

Ce projet n'est pas confidentiel au sens légal du terme, cependant il est demandé de garder les informations, en particulier techniques, interne au groupe de compétences « Systèmes informatiques embarqués ».

4.9 Compléments :

Aucun

4.10 Particularité :

Toutes les propositions qui vont dans le sens de l'amélioration du projet sont les bienvenues et vivement encouragées.

5 Conclusion

Ce projet est une étape importante pour l'obtention de son Master, fin d'une longue vie d'étude. Il est donc important de bien vivre cette étape de vie et donc de s'investir valablement dans ce projet en y trouvant du plaisir à travailler alors :

ENJOY IT

6 Réalisation :

Candidat :
Mme. Manon Racine

Professeurs responsables:
Mme Nuria Pazos nuria.pazos@he-arc.ch 032 930 22 50

Répertoire de travail :
P:\RaD\040-ProjetsRaD\18TICa09 HES-SO - LoRaLoc\6-PA_RMA

7 Annexes :

Mentionnées.

Annexe B.

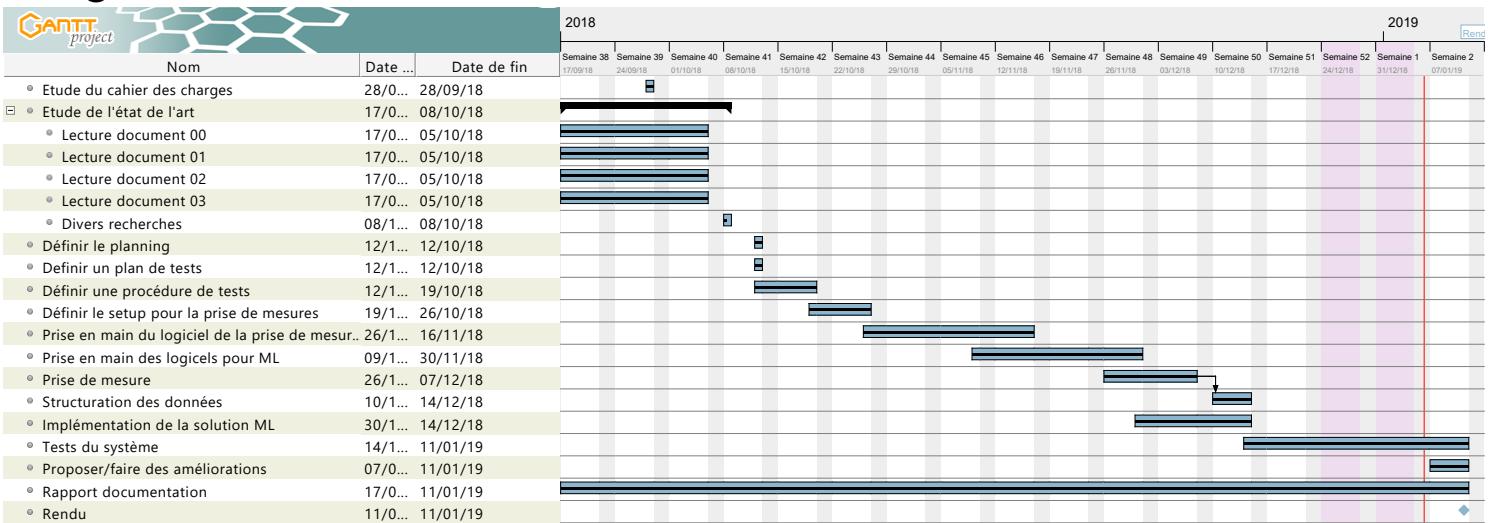
Planning

Untitled Gantt Project

6 janv. 2019

4

Diagramme de Gantt



Annexe C.

Journal de travail

Journal de travail

Date	Semaine	Temps [h]	Tâches	Commentaires
17.09 - 23.09	1	2	Mise en place de latex	
		1	Discussion démarrage	Mise en place du cadre et transmission
		8	Démarrage projet	GPS-free Geolocation using LoRa.pdf -> pour avoir une idée comment supprimer des mesures fausses
24.09 - 30.09	2	3	Lecture	Lecture du document : ML_Indoor Positioning.pdf Lecture du document : ML_UWB Localization.pdf -> très difficile à comprendre et interpréter... Demander conseil à Alexis.
		6	Lecture	Séance de démarrage du projet avec Nuria et clarification du cahier des charges. Lecture des documents concernant l'état de l'art.
		3	Rapport	Mise en place du rapport et démarrage de la rédaction
01.10 - 07.10	3	6	rapport	résumé de l'état de l'art
		3.5	rapport	résumé de l'état de l'art document 03 + recherches sur Internet (pas concluant)
08.10 - 14.10	4	5	rapport	Finalisation de l'état de l'art -> faut encore faire une petite recherche Internet sur le sujet
		8	recherche	Analyse de la base de données UjiIndoorLoc Analyse des différences entre Regression et Classification pour mieux comprendre nos besoins
15.10 - 21.10	5	2	Recherche	Recherche concernant KNN en tant que algo de régression, idem pour SVM (SVR) et random forest.
		6	Rédaction du rapport	introduction
		2	Rédaction du rapport	Etat de l'art
22.10 - 28.10	6	6	Rapport + réflexion plan	Finton de l'état de l'art dans le rapport et début des réflexions concernant la prise de mesure
		1	Discussion	Discussion avec Hatem concernant les différents algorithmes et le choix entre Regression ou Classification. Discussion concernant la prise de données
		6	Soft de mesure	Discussion avec Nuria, début de la prise en main du soft de mesure de Michael Mueller
29.10 - 04.11	7	4	Définition du plan de tests	Réflexion et mise en place du plan de test. (abstract03)
05.11 - 11.11	8	10	Prise en main soft	Mise en marche du soft de prise de mesure dans le laboratoire.
		6	Installation du système au Labo	Le système ne fonctionnant pas dans mon bureau j'ai dû l'installer dans le bureau des étudiants et installer les outils de développement sur mon portable.
		4	Correction Rapport	Faire les corrections du rapport indiquées par Nuria et faire fonctionner la démo au Labo.
12.11 - 18.11	9	8	Modification soft PC	Tentative de mise en forme des données avec start / stop après 4 burst
		6	Amélioration soft PC	Mise en forme de la récupération des données
19.11 - 15.11	10	8	Changement du soft PC	Sauvegarde des données dans un fichier
		3	Prise de mesure	Prise de mesure pour le premier point
26.11 - 02.12	11	6	Prise de mesure	Prise de mesure et adaptation du soft car problème de sauvegarde
		6	Prise de mesure	Prise de mesure plus adaptation du soft afin d'être sûr de prendre des bonnes mesures
03.12 - 09.12	12	8	Soft de mesure	Analysé de l'algorithme qui sera utilisé dans un premier temps pour la classification (SVM)
		8	Récupération des mesures	Récupération des données mesurées et mise en forme afin qu'elles puissent être traitées
		10	Classification + Rapport	Entrainement de l'algo SVM sur 5 points 100% fonctionnels. Validation avec quelques points un peu décalés, cela ne fonctionne plus. Essai avec Random Forest
		3	Prise de mesure + Rapport	Prise de mesure pour le point T7 / T8 / T0 / S3 / T2 / T3 / S5 (40+5) et avancement dans le rapport
10.12 - 16.12	13	8	Prise de mesure + test	Test de différentes manières en ajoutant les points extérieurs (Tx) mais également les points supplémentaires aux postions Sx (SxT) -> mauvais résultats si un point différent est testé
		6	Rapport	Écriture du rapport pour documenter la prise de mesure
		10	Rapport + Mesures	Mesures avec les points convergés (env. 3h par mesures) sur S0 et S1, nouvelle prise de mesure car celle obtenue en utilisant les valeurs brutes ne donne pas des très bons résultats
		7	Rapport + mesures	Mesure sur S1 lorsque les étudiants étaient là et mesures du point S2 en prenant uniquement les positions convergées. Optimisation de l'algorithme.
		5	Rapport + Mesures	Explication des étapes de machine learning réalisée. Mesure du point S3
17.12 - 23.12	14	6	Mesure	Mesures des points S4 et S5, beaucoup de problèmes avec le slave S2 qui n'a pas arrêté de planter
		8	Mesure + rapport + algo	Mesure des points de test Tx, avancement dans le rapport et continuer d'améliorer l'algorithme. J'ai refait le point S1T qui me paraissait faux. Niveau mesure, il me manque T0 et T6
24.12 - 30.12	Vacances	5	Rédaction du rapport	Description de la mise en place de l'algorithme
		5	Rédaction du rapport	Description des résultats obtenus avec les positions convergées
		5	Rédaction du rapport	Description des résultats obtenus avec les mesures courtes
31.12 - 06.01	Vacances	8	Rédaction du rapport	Description des résultats obtenus avec les mesures courtes
		2	Rédaction du rapport	Résumé des résultats / problèmes rencontrés
		7	Rédaction du rapport	Conclusion / résumé
07.01 - 13.01	Rendu	8	Relecture	