

Rapport Pédagogique De Stage

Meuble mobile : Gestion de l'autonomie de la base

Tuteur Industriel : Lydie CAETANO

Tuteur Pédagogique : Jean-Luc PHILIPPE



Ce stage a été effectuée du 4 Mai au 28 Août 2020 (avec une interruption entre le 22 Juillet et le 24 Août 2020) pour la Chaire M@D à l'ENSIBS Lorient.

Killian ROLLAND

Élève-Ingénieur en Mécatronique à l'ENSIBS Lorient

killian.rolland29@gmail.com

Table des matières

I.	Résumés	2
1.	Résumé en français	2
2.	Abstract	2
II.	Introduction	2
III.	Présentation de l'entreprise	3
IV.	Contexte	3
V.	Présentation de la mission.....	4
VI.	Démarche	5
1.	Autonomie de gestion de la batterie	6
2.	Intelligence de déplacement.....	7
3.	Interaction entre le robot et l'utilisateur ou l'extérieur	10
VII.	Résultats	11
VIII.	Gestion du projet	13
IX.	Retour d'expérience.....	15
1.	REX technique	15
2.	REX pédagogique.....	15
X.	Conclusion.....	16
XI.	Bibliographie	17

Table des illustrations

Figure 1 : Photographie du TurtleBot 3 Waffle Pi	5
Figure 2 : Topic ROS intitulé « battery_state »	6
Figure 3 : Photographie montrant le robot à côté de sa base factice	7
Figure 4 : Schéma présentant le protocole pour les mesures d'évitements d'obstacles dynamiques	8
Figure 5 : Interface Web permettant de piloter le robot.....	10
Figure 6 : Fonctionnement du système.....	12

I. Résumés

1. Résumé en français

Ce document est le rapport pédagogique de mon stage réalisé au sein de la Chaire M@D à Lorient. Ce stage avait pour but de poursuivre les travaux déjà réalisés par d'autres élèves sur le meuble mobile, afin de rendre ce dernier autonome. Le rapport ci-présent explique les démarches que j'ai réalisées et permet de décrire les avancements du projet et les travaux effectués pour y parvenir.

Mots clés : Robotique, Navigation autonome, Aide à la personne, ROS, gestion d'énergie, meuble mobile.

2. Abstract

This document is the report of my internship realised for the Chaire M@D in Lorient. This internship had the goal to follow the work previously done by other students on the mobile furniture, in order to make it autonomous. The report explains the steps I realised and describe the advancements on the project and the works I carried out to achieve it.

Keywords: Robotics, autonomous navigation, help to the person, ROS, energy monitoring, mobile furniture.

II. Introduction

J'ai réalisé mon stage de deuxième année de mécatronique pour la Chaire M@D. Le but de ce stage était de poursuivre les nombreux développements qui avait été déjà été effectués sur une base motorisée : le meuble mobile.

L'objectif final est d'avoir une base totalement autonome, capable d'apporter des objets à une personne, et de gérer elle-même son niveau de batterie (en allant se recharger seule). Ce sujet allie des notions de développements informatiques et d'électronique, qui sont des sujets s'alliant avec les enseignements dispensés par ma formation.

Ma tutrice d'entreprise est Lydie CAETANO tandis que mon tuteur pédagogique est Jean-Luc PHILIPPE.

III. Présentation de l'entreprise

La chaire M@D a été inaugurée en Novembre 2017. C'est un projet mené conjointement par l'ENSIBS et l'IMT Atlantique, en partenariat avec le centre de rééducation de Kerpape.

L'objectif de cette chaire est de répondre à des problématiques que des personnes en situation d'handicap ou en perte d'autonomie rencontrent dans leur habitat. La chaire M@D cherche donc à développer des systèmes qui facilitent le maintien à domicile des personnes ayant besoin d'aide pour les gestes de la vie quotidienne, tout en luttant contre l'isolement des personnes.

L'équipe à plein temps de la Chaire M@D est composée de Bruno JANET (titulaire de la Chaire), Jean-Luc PHILIPPE (Co-directeur scientifique, Référent scientifique de Lorient), André THEPAUT (Co-directeur scientifique, Référent scientifique de Brest), Véronique BOSC-BUREL (Secrétaire générale) et de Lydie CAETANO et Abbas RAMADAN qui sont ingénieurs de recherche.

IV. Contexte

Ce projet vient d'une rencontre entre la Chaire M@D et le CCAS (Centre Communal d'Action Sociale) de Lorient. En effet, il y a un besoin d'apporter des objets du quotidien à des personnes dont les déplacements sont difficiles, même à domicile. Le projet du meuble mobile a donc pour objectif de proposer des services à domicile personnalisés afin de permettre à ces personnes de maintenir leur autonomie à domicile.

Le robot doit donc pouvoir se déplacer en autonomie à domicile (savoir où il est, où il doit aller, et ce, de manière autonome), rejoindre une personne afin de lui apporter un objet, mais aussi avoir une autonomie énergétique (il ne doit pas nécessiter d'être constamment branché au secteur pour fonctionner).

De nombreux groupes d'élèves ont déjà travaillé sur le projet, mon stage demandait donc de s'interfacer avec les développements déjà faits et d'ajouter des fonctionnements.

V. Présentation de la mission

Avant le début de mon stage, le meuble mobile est capable de se déplacer correctement et est alimenté par une batterie. On peut alors piloter le robot en lui donnant des commandes par clavier, ou alors lui demander de se déplacer d'un point A à un point B en autonomie avec ROS.

Ainsi, la mission de mon stage était dans un premier temps de prendre connaissance des développements faits précédemment à l'aide de rapports et documents livrés.

Ensuite, l'objectif était de finaliser les développements existants concernant l'intelligence de déplacement et l'autonomie en batterie.

Dans la finalité il était attendu de ma part de prendre en main le système de monitoring de la batterie (gestion intelligente de la batterie) et de l'interfacer avec la partie intelligence du système afin de réaliser une machine d'état adaptée à un scénario d'usage (une personne appelle le meuble, le meuble apporte un objet et rechargement de la batterie du meuble).

Une évaluation de la partie évitement d'obstacles était également requise (test des performances, paramétrages, ajout de capteurs si nécessaire, etc.) afin de s'assurer que la navigation du robot dans son environnement sera sécurisée et que sa capacité à éviter des obstacles mouvants (ou non) sera suffisamment bonne.

Une étude était aussi nécessaire quant à l'interaction entre le meuble et l'utilisateur afin de décider comment l'utilisateur pourrait appeler le meuble et que le meuble puisse la rejoindre (manière de faire l'appel, localisation de la destination du meuble dans sa cartographie).

VI. Démarche

Dans le contexte sanitaire (Covid-19) de cette année, il a fallu revoir, au début du stage, les conditions de réalisation du stage et ses objectifs.

Pour des raisons pratiques et sécuritaires, il a donc été décidé que ce stage pouvait être réalisé en télétravail. Dans un premier temps je pouvais effectuer le travail de recherche et développement sur un autre robot que le meuble mobile : le TurtleBot3 qui se trouvait chez moi puisque j'avais effectué un projet avec la Chaire M@D quelques semaines avant le stage.

Ce robot (voir *Figure 1 : Photographie du TurtleBot 3 Waffle Pi*) mesure 281mm de long, 306mm de large pour une hauteur de 141mm. Il peut supporter une charge allant jusqu'à 30 kg et une autonomie de batterie de 2h (pour 2h30 de chargement sur secteur). Il est pilotable sous ROS et possède de nombreuses applications open sources créées par le constructeur (Robotis).

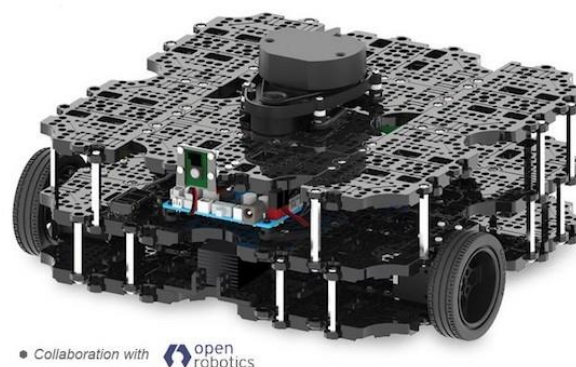


Figure 1 : Photographie du TurtleBot 3 Waffle Pi

Ainsi, dans un premier temps mon travail a été de déterminer les différents aspects du stage et de les adapter légèrement afin de les développer sur le TurtleBot3.

Les différents aspects du stage qui sont ressortis sont les suivants :

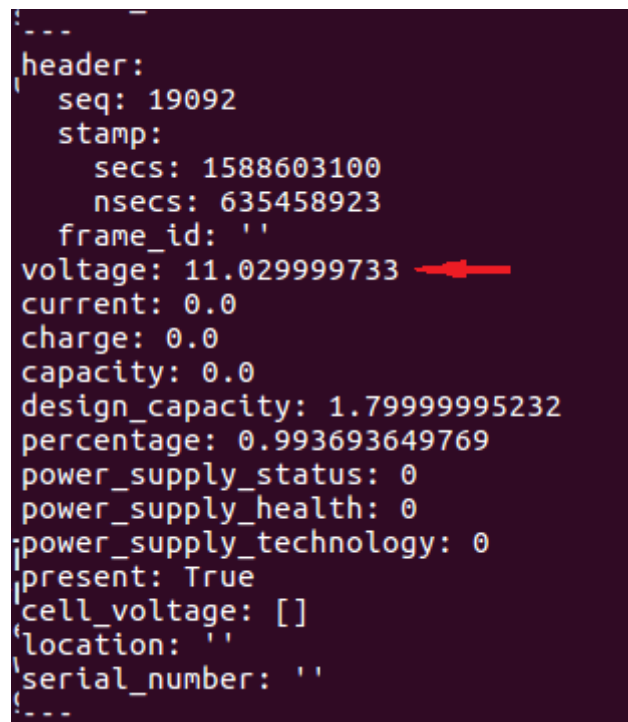
1. L'autonomie de gestion de la batterie ;
2. L'intelligence de déplacement ;
3. L'interaction entre le robot et l'utilisateur ou l'extérieur.

Voici donc les démarches que j'ai réalisé pour chaque partie :

1. Autonomie de gestion de la batterie

A bord du TurtleBot, on peut trouver deux cartes. Une Raspberry Pi qui sert à l'intelligence du robot où l'on retrouve Linux installé (Raspbian) avec ROS. Il y a une seconde carte : l'OpenCR qui sert à fournir la puissance aux différents composants du robot (Caméra, Lidar, Moteurs).

Cette carte de puissance est commandée par des codes Arduino, et il se trouve que l'on peut récupérer grâce à cela la tension de la batterie en temps réel. Cette tension est retrouvable dans les topics ROS (voir Figure 2 : Topic ROS intitulé « battery_state ») lorsque l'on pilote le robot.



```
header:
  seq: 19092
  stamp:
    secs: 1588603100
    nsecs: 635458923
  frame_id: ''
voltage: 11.029999733
current: 0.0
charge: 0.0
capacity: 0.0
design_capacity: 1.79999995232
percentage: 0.993693649769
power_supply_status: 0
power_supply_health: 0
power_supply_technology: 0
present: True
cell_voltage: []
location: ''
serial_number: ''
```

Figure 2 : Topic ROS intitulé « battery_state »

En effet, on remarque ici que l'on peut accéder à la tension de la batterie en temps réel dans la partie à côté de la flèche rouge « voltage ». Cette valeur est relevée par un capteur de tension présent dans la carte de puissance « OpenCR » du robot.

J'ai donc décidé de créer un nœud ROS qui va récupérer la valeur de cette tension de batterie en temps réel et qui va faire en sorte que le robot aille à une position désignée comme sa station de charge si son niveau de batterie est trop faible.

De cette manière, si le robot arrive à une tension de batterie inférieure à 5%, le robot va se déplacer à un lieu proche de sa « base de recharge », puis va chercher un marqueur qu'il reconnaîtra via sa caméra, qui lui indiquera plus précisément où se trouve sa base, pour qu'il puisse la rejoindre.

Afin de rendre la batterie lisible pour l'utilisateur, j'ai mis en place une démarche de calcul simple qui permet de transformer la tension de la batterie en % de batterie restant : lorsque la batterie est considérée comme vide (=0%) on a 11V et lorsque la batterie est comme pleine (=100%) on a environ une tension de 12,42V.

Comme $(12,42-11)/100 = 0.0142$, on en déduit qu'à chaque fois que la batterie perd 0.0142V elle perd 1% de batterie.

En réalité Cela n'est pas exact puisque cette batterie est une batterie au lithium et on a donc une tension qui est stabilisée à 11,1 V (elle va rester quelques temps à ce niveau de tension avant que celle-ci chute). Cependant cela permet d'avoir malgré tout une idée de la charge actuelle de la batterie même si ce pourcentage descendra plus vite à certains moments qu'à d'autres.

Ainsi mon nœud ROS intitulé « battery_monitoring » publiera en temps réel le pourcentage de batterie restant avec le simple calcul : $(\text{tension_actuelle} - 11)/0.0142$.

Pour information, le Turtlebot ne possède pas de station de rechargement, ainsi le fonctionnement du rechargement autonome est ici factice (voir *Figure 3 : Photographie montrant le robot à côté de sa base factice*), puisque c'est à l'utilisateur d'éteindre le robot et de brancher la batterie du robot.



Figure 3 : Photographie montrant le robot à côté de sa base factice

2. Intelligence de déplacement

Pour l'intelligence de déplacement, il fallait dans un premier temps, comme indique dans la présentation de la mission, pouvoir évaluer les capacités d'évitement du robot.

J'ai ainsi mis en place un protocole permettant de prendre des mesures sur les évitements d'obstacles dynamiques auxquels pourrait être confronté le robot.

Ainsi à l'aide de 2 ballons de tailles différentes lancés à plusieurs vitesses différentes, j'ai pu évaluer les cas où le robot réagissait suffisamment vite et précisément afin d'éviter la collision avec un obstacle. J'ai ainsi réalisé une multitude de vidéos qui montrent chaque essai ainsi qu'un tableur Excel qui caractérise chaque essai par le type de balle, la vitesse de lancer, le résultat obtenu, la qualité du temps de réaction, etc.

Ce document est trouvable dans le Google Drive en annexe [11] : (Chaire_M@D_Stage2020-Meuble_Mobile&TurtleBot_Killian_ROLLAND/Documentation/Obstacles_dynamiques/Evitement_dynamique_V2.xlsx dans l'arborescence).

Les vitesses étaient relevées à l'aide de marqueurs aux sols et d'un chronomètre avec des trajectoires effectuées comme sur le schéma *Figure 4* : Schéma présentant le protocole pour les mesures d'évitements d'obstacles dynamiques (les balles en rouge et le robot en bleu).

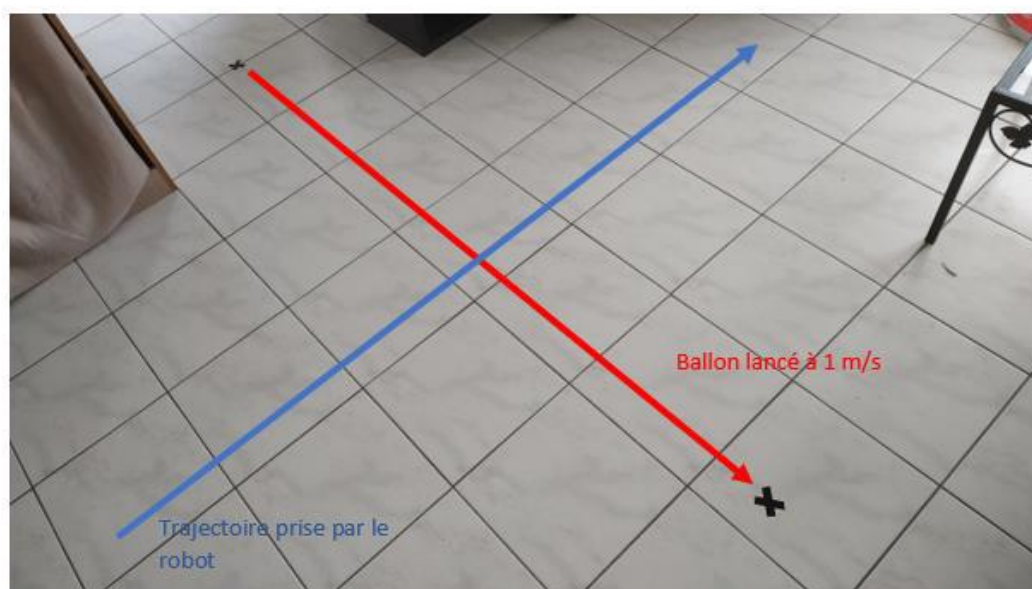


Figure 4 : Schéma présentant le protocole pour les mesures d'évitements d'obstacles dynamiques

Concernant l'intelligence de déplacement du robot à proprement parler, je me suis inspiré de forums et tutoriaux (notamment un tutoriel ROS numéroté [7] dans la bibliographie de ce document) afin d'obtenir un algorithme capable de commander la navigation autonome du robot via ROS. Ainsi, une fois la cartographie de l'environnement du robot faite (voir document de déploiement qui explique les procédures à suivre sur le Google Drive : *Documentation/Document_deploiement_TurtleBot3.pdf*), on peut récupérer plusieurs points de cette carte et les enregistrer afin de décider d'y envoyer le robot.

J'ai donc créé un nœud ROS intitulé « simple_navigation_goals », qui va recevoir des ordres envoyés par un utilisateur et qui va indiquer au robot qu'il doit se rendre à un endroit voulu. Cela fonctionne dans la mesure où la navigation ROS a déjà été installée sur le robot et qu'elle a été correctement paramétrée.

Comme je viens de le décrire rapidement, la gestion d'un appel par l'utilisateur a été choisi de manière que ce soit l'utilisateur qui indique sa position au robot. En effet, ce que j'ai jugé le plus faisable est que le robot enregistre plusieurs endroits de l'appartement puisque, il suffirait alors à la personne d'indiquer où elle se trouve (du type salon, chambre, cuisine) pour que le robot la rejoigne. Cette solution est d'autant moins gênante du fait que les utilisateurs seront pour la plupart dans une situation d'handicap ou de perte d'autonomie, ce qui fait que les possibilités de leur localisation dans leur logement à un instant t ne sont pas nombreuses.

L'idée est que ces ordres soient envoyés simplement par l'utilisateur, c'est ce que traite l'aspect suivant.

3. Interaction entre le robot et l'utilisateur ou l'extérieur

Pour permettre à l'utilisateur d'être en interaction avec le robot, j'ai décidé de passer par un téléphone portable de type smartphone Android (Modèle Xiaomi Redmi Note 6 Pro sous Android 9.0).

Dans un premier temps j'avais développé un système où la personne se connectait au master ROS via l'application JuiceSSH, cependant l'interaction était assez médiocre puisque l'on avait simplement un terminal ou des informations étaient affichées et où l'utilisateur devait rentrer des chiffres afin d'indiquer ce qu'il souhaite faire du robot. Ce fonctionnement est illustré dans des vidéos que j'ai réalisées pour montrer les différents pilotages que l'on pouvait faire (voir le lien Google Drive en annexe Videos/Pilotages_via_JuiceSSH).

Par la suite je me suis renseigné sur les applications Android et ce que l'on pouvait en faire en utilisation avec ROS. J'ai finalement décidé que la solution pour avoir une interface graphique pour l'utilisateur serait de développer une application Web. Ainsi, toute personne possédant un appareil qui peut se connecter à un site Web (PC, téléphone, tablette, etc.) pourrait piloter le robot. L'apparence finale de cette interface Web est la suivante, présentée Figure 5 sur un smartphone :



Figure 5 : Interface Web permettant de piloter le robot

Ainsi dans ce cas de figure la personne a accès au pourcentage de batterie actuel, à la vue caméra du robot, a des boutons permettant d'envoyer des destinations voulues pour le robot, à un onglet information qui permet de donner les états actuels du robot et à un onglet permettant d'avoir accès à un joystick afin de déplacer le robot manuellement en cas de problème.

Les démarches pour les 3 aspects du stage ont donc été décrites dans les parties précédente qui sont l'autonomie de gestion de la batterie, l'intelligence de déplacement et l'interaction entre le robot et l'utilisateur.

Ainsi, l'explication de la mise en place de ce fonctionnement est trouvable dans la bibliographie [9], et les vidéos se trouve dans l'onglet « Vidéos » de ce même Google Drive. La dernière partie restante du travail est alors la partie où il faut adapter les codes au meuble mobile afin d'obtenir un fonctionnement similaire à celui décrit précédemment.

Dans un premier temps j'ai dû prendre connaissance des aspects techniques du meuble, d'autant plus que le simple pilotage des moteurs sans ROS ne fonctionnait plus. Après étude des plans électroniques du système j'ai pu résoudre ce problème. En effet on voit dans l'annexe [8] que sur le schéma PCB, c'est une vue de dos de la carte, et qu'il faut donc bien brancher l'alimentation 12V là où elle est indiquée et non pas sur l'alimentation secondaire comme le branchement était fait.

Le problème suivant est que le dernier groupe ayant travaillé sur le meuble a rencontré un crash ROS qui fait que ce qui fonctionnait à l'origine (navigation autonome) ne fonctionne maintenant plus.

A l'heure actuelle je peux lancer rviz (et la navigation stack), indiquer au robot où il se situe sur la carte. Lorsque je lui demande d'aller à un point il va se mettre à tourner sur lui-même et le terminal m'indique qu'il n'arrive pas à trouver sa position sur la carte

VII. Résultats

Les résultats concernant la partie avec le TurtleBot3 sont assez bons. Il suffit à l'utilisateur d'avoir un robot correctement paramétré (cartographie du domicile réalisée, points que le robot doit connaître enregistrés, base du robot connue et installée). Ensuite il pourra depuis un smartphone (ou autre) se connecter à un navigateur Web et à l'aide de l'adresse IP du master ROS, avoir l'interface que l'on peut voir Figure 5 : Interface Web permettant de piloter le robot), se référer à l'annexe [9] de la bibliographie.

Concernant l'étude des capacités d'évitement d'obstacle du robot, le protocole établi a retiré la conclusion que dans la plupart des cas le robot a été capable d'éviter la balle (en s'arrêtant ou en la contournant) malgré quelques erreurs occasionnelles. Ces erreurs ont été généralement dues à une erreur au niveau du Lidar qui voyait un obstacle invisible (possiblement une poussière qui se place sur la trajectoire du laser) et qui donc cherchait à éviter ce dernier ; ou encore à un manque de capacité de réaction du robot dû à l'apparition

d'un obstacle soudain (du type la balle a rebondi contre un mur et reviens subitement sur la trajectoire du robot).

La gestion de la batterie fonctionne correctement et dès que le robot se trouve en situation de batterie faible, il avertira l'utilisateur avant de se rendre à sa base et attendre sa mise en charge.

Un souci reprochable à ce système est le besoin d'avoir un PC qui fait office de master ROS. En effet, la carte Raspberry Pi du TurtleBot ne peut pas intégrer toutes les utilisations ROS nécessaires pour la navigation, il faut donc que la partie Master se fasse sur un PC. Le fonctionnement est en fait le suivant (cf. Figure 6) :

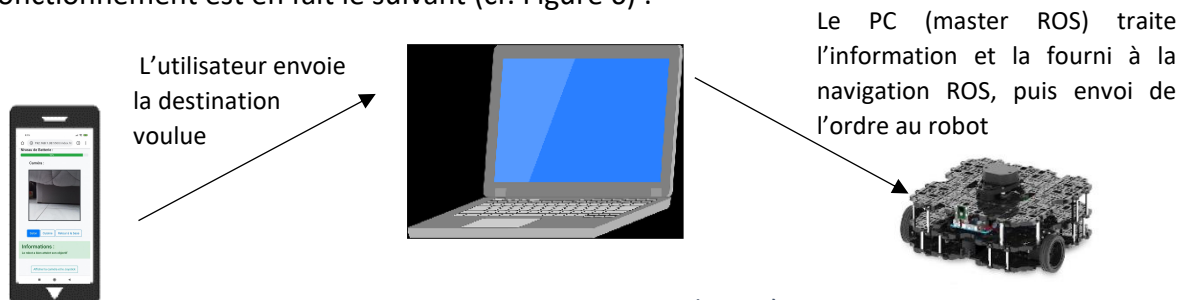


Figure 6 : Fonctionnement du système

Ce souci est un peu plus détaillé dans le document numéro [10] de la bibliographie.

VIII. Gestion du projet

Lors de ce projet j'ai pu réaliser divers documents d'explications (caméra, protocole, etc.), un journal de bugs, plusieurs vidéos et j'ai créé un GitHub permettant à toute personne d'accéder librement à tous les algorithmes que j'ai développés et qui sont nécessaires au fonctionnement du système. Tous ces documents sont trouvables dans la partie Documentation de mon Google Drive (lien n°[11] dans la bibliographie), tandis que mon GitHub est trouvable au lien [12].

Chaque semaine j'envoyais un compte-rendu relatant des avancées, problèmes ou questionnement que j'ai eu pendant la semaine (annexe [13]). Comme le stage était en télétravail, il y a eu plusieurs visioconférences qui permettaient de faire le point sur le stage et de discuter des difficultés.

Concernant les différentes tâches effectuées et leur moment d'exécution durant mon stage, vous trouverez ci-dessous les tâches et les Gantt initiaux et finaux.

Tâche	Titre de la tâche
TA01	Etude de la configuration actuelle du meuble (documentation, ressources, annales).
TA02	Choix des solutions répondant aux problèmes
TA03	Protocole de mesure d'évitement
TA04	Intégration de l'intelligence de déplacement.
TA05	Intégration de l'intelligence de gestion de la batterie.
TA06	Intégration du mécanisme d'appel du meuble.
TA07	Développement d'une interface graphique.
TA08	Adaptation au meuble mobile.

Ci-dessous le Gantt Initial et Effectif :

Légende : En bleu : Gantt Initial / En Orange : Gantt Effectif / En Violet : Chevauchement des deux.

Semaine	1	2	3	4	5	6	7	8	9	10	11	12
TA01	Blue	Blue					Blue			Orange	Orange	
TA02	Violet	Violet	Orange	Orange	Orange				Orange			
TA03		Violet	Orange	Orange								
TA04			Blue	Blue	Violet	Violet	Violet					
TA05				Blue	Violet	Violet	Violet	Orange				
TA06					Orange	Violet	Violet	Violet	Violet	Orange		
TA07								Orange	Violet	Violet	Blue	
TA08							Blue	Blue	Blue	Violet	Violet	Violet

Le Gantt initial présent ci-dessus était une estimation du temps qu'allait prendre les différentes tâches ainsi que le moment où j'allais les démarrer.

Les écarts sont principalement dus au temps de recherches pour certaines parties qui ont été plus longs que prévu. Par exemple la décision du système d'appel a demandé beaucoup de recherche, puisque le but était de faire un état de l'art de comment on peut donner une localisation à un robot afin que ce dernier puisse rejoindre une personne.

De plus j'ai passé du temps à faire des recherches autour de la caméra du TurtleBot afin de déterminer si elle pouvait être utile lors de la détection d'obstacles ou pour la navigation. Cette étude est trouvable dans l'annexe [15].

Quelques bugs m'ont aussi fait perdre du temps (ils sont indiqués dans un *Journal_de_bugs*) puisqu'il a fallu les résoudre ou les contourner (annexe [16]).

Il est donc très intéressant de constater que l'on néglige facilement le temps que l'on passera à résoudre des bugs dans un projet. Cette facette est une partie intrinsèque du métier à laquelle il faut savoir penser lorsque l'on veut planifier un projet (principalement les durées de chaque tâche relative à ce projet).

Finalement, la prise en main du meuble a été assez compliquée puisqu'une simple erreur de branchement m'a demandée un certain temps d'étude des documents pour comprendre ce qui n'allait pas. Le fait que la navigation ROS ne fonctionnait plus comme avant sur le meuble m'a aussi demandé beaucoup d'essais et de paramétrages pour essayer de régler ce problème.

Ces problèmes et paramétrages essayés sont trouvables dans les comptes-rendus Hebdomadaires n°11 et 12 dans la Bibliographie [13].

IX. Retour d'expérience

1. REX technique

Le projet peut encore être amélioré sur une multitude de points.

Ce point a été traité dans le document [10] de la bibliographie et est plus détaillé en aspects techniques.

La mise en place du système peut être un peu longue puisqu'il faut faire la cartographie de l'appartement, enregistrer tous les points voulus et modifier les codes avec les nouveaux points, voire modifier l'interface graphique pour ajouter/enlever des boutons. Cela pourrait donc être automatisé par un algorithme qui pourrait effectuer les enregistrements des points et modifications voulues par simple demande de l'utilisateur dans un terminal lors de l'installation.

Il faudrait aussi trouver un moyen d'avoir la totalité de ROS embarquée sur le robot et ne pas avoir besoin de passer par un PC pour utiliser le système. Une perspective pourrait être de changer de système d'exploitation pour la Raspberry Pi (sur le TurtleBot) ou encore de changer de carte (passage à une Odroid par exemple).

Finalement le développement/utilisation d'une synthèse vocale pourrait rendre le système bien plus interactif du point de vue de l'utilisateur puisque le robot pourrait énoncer en temps réel qu'il se déplace vers lui, qu'il retourne à sa base, qu'il n'a plus de batterie, etc.

2. REX pédagogique

Lors de ce stage j'ai pu acquérir une multitude de compétences qui me seront utiles dans la suite du déroulement de ma carrière.

Premièrement les recherches que j'ai dû mener sur la robotique (et plus particulièrement dans un contexte d'aide à la personne) m'ont permis d'améliorer grandement mes connaissances sur le sujet et n'ont fait qu'accroître mon intérêt pour ce domaine.

J'ai pu continuer à découvrir ROS, sur un robot que j'utilisais déjà pour un projet scolaire dans la Chaire M@D, ce qui m'a permis de découvrir de nouvelles facettes de ce système et de mieux comprendre son fonctionnement.

Ce stage m'a donné l'opportunité d'apprendre une multitude de langages de programmation puisque je n'avais encore jamais codé en C++, Javascript, HTML, Arduino et les formats ROS. J'ai donc pu découvrir les subtilités des algorithmes de pilotages par ROS, ainsi que la création d'une page Web permettant de piloter un robot depuis un smartphone.

X. Conclusion

Ce stage m'a été très bénéfique et j'ai apprécié son déroulement jusqu'à la fin. Je suis malgré tout déçu de ne pas avoir pu transposer complètement le système fonctionnel du TurtleBot vers le meuble mobile. La multitude d'outils de programmation que j'ai découvert et utilisé ne pourra que m'être bénéfique dans la suite de ma carrière. Le fait d'utiliser de nouvelles technologies dans un but d'aide à la personne procure aussi une sensation d'être utile et le fait de savoir que l'on développe quelque chose qui pourrait fortement aider quelqu'un est très bénéfique pour le moral.

Pour le bon déroulement de mon stage je remercie Lydie CAETANO et Jean-Luc PHILIPPE qui ont pu être à l'écoute de mes problèmes et qui ont toujours su me donner des conseils avisés me permettant d'avancer dans mon projet. Je remercie aussi Maël CHEVANCHE, Pierre BOMEL et Éric SENN qui ont pu m'apporter de l'aide lorsque j'avais des questions à leur poser.

XI. Bibliographie

- [1] Le site de la Chaire M@D : <https://chaire-mad.fr/>
- [2] Le site de Robotis, constructeur du TurtleBot3, avec tous les tutoriels du robot : https://emanual.robotis.com/docs/en/platform/turtlebot3/getting_started/#getting-started
- [3] GitHub, site où j'ai pu trouver beaucoup de réponse à mes questions concernant ROS et où j'ai pu poster mes algorithmes : <https://github.com/>
- [4] Séries de vidéos ayant inspiré mon travail concernant l'interface Utilisateur/Robot ainsi que l'intelligence de déplacement : <https://www.youtube.com/watch?v=m-UyentHOwM&t=4s>
- [5] Forums de Raspberry Pi pour les problèmes concernant la Raspberry Pi 3B+ : <https://www.raspberrypi.org/forums/>
- [6] Google Drive du stagiaire de l'année précédente :
Meuble_Mobile_2019_Riwan_GIRONNET/Documentation
- [7] Tutoriel sur le Wiki ROS qui montre comment envoyer le robot à un endroit en envoyant des commandes à la pile de navigation du robot : <http://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals>
- [8] Meuble_Mobile_2019_Riwan_GIRONNET/Documentation/Meuble_Mobile/Conception_PCB/Carte-draw.png
- [9] Chaire_M@D_Stage2020-Meuble_Mobile&TurtleBot_Killian_ROLLAND
/Documentation/Document_deploiement_TurtleBot3.pdf
- [10] Chaire_M@D_Stage2020-Meuble_Mobile&TurtleBot_Killian_ROLLAND
/Documentation/Perspectives_et_Solutions
- [11] Lien vers le Google Drive du stage : <https://drive.google.com/drive/folders/10cN4ByttwTlIeknRdFRT4O2PrBo6WrrW?usp=sharing>
- [12] <https://github.com/KillianRolland/KillianRolland-Navigation-goals-and-battery-monitoring-TB3>
- [13] <https://drive.google.com/drive/u/1/folders/100neon9A6J9STOeXsIlgk36zPgZTwUhp>
- [14] Chaire_M@D_Stage2020-Meuble_Mobile&TurtleBot_Killian_ROLLAND
/Documentation/Raspberry_Pi_Camera
- [15] Chaire_M@D_Stage2020-Meuble_Mobile&TurtleBot_Killian_ROLLAND
/Documentation/Journal_de_bugs.xlsx