PEDROLI Johan

Rapport de stage de l'année 4

Informatique et Électroniques des Systèmes Embarqués

PROJET FROLL'N'ROLL

9 mai 2023 - 11 août 2023

Remerciements

Un grand merci à Roger Pissard-Gibollet, Christophe Braillon et à l'ensemble de l'équipe du SED pour l'accueil et l'aide qu'ils m'ont apporté.
Merci aussi à Siméon qui m'a donné des conseils en tant que stagiaire plus ancien.
Merci à Benoît, Marco et tous les bénévoles pour l'organisation du Fabrikarium à Palavas.
Merci à Etienne pour la relecture.
Et merci à Sylvain Toru d'avoir été mon enseignant référent durant ce stage.

Sommaire :

Glossaire

I2C	Inter-Integrated Circuit
CNA	Convertisseur numérique-analogique
MQTT	Message Queuing Telemetry Transport
BLE	Bluetooth Low Energy
IDE	Environnement de développement
PCB	Printed Circuit Board
MHK	My Human Kit, association partenaire
SED	Service Expérimentation et Développement
ToF	Time of Flight

I. Introduction

Le porteur de projet est Etienne Moullet, post-doctorant à l'Inria dans les équipes de recherches Camin et Willow.

Étienne, sur la figure 1, est atteint de tétraplégie partielle. La tétraplégie est une paralysie totale ou partielle des membres inférieurs et supérieurs. Ceci se caractérise par une absence de mouvements du fait de lésions à la moëlle épinière. Etienne peut par exemple mobiliser ses épaules mais a une force de serrage des mains nulle.
Pour se déplacer, Etienne utilise un fauteuil roulant à assistance électrique.
Afin de faciliter la commande de son fauteuil, Etienne porte le projet Froll'N'Roll Il vise à modifier le contrôle des moteurs présents sur les roues du fauteuil.

Figure 1 : Image d'Etienne Moullet sur son fauteuil à assistance électrique

Pour commencer, je vais vous présenter là où j'ai travaillé avec les différentes instances.
Ensuite, je vais détailler le but du projet ainsi que où je l'ai trouvé au début de mon stage dans la partie Contexte du projet. Dans la partie gestion de projet, je vous expliquerai ma gestion du temps ainsi que les outils que j'ai utilisés. Puis, je vous montrerai ce que j'ai fait dans la partie Travaux réalisés.

II. Cadre de travail

1. Inria Grenoble - Rhône-Alpes

L'Inria est un établissement public créé le 3 janvier 1967. Il est spécialisé en mathématiques appliquées et en informatique. Celui-ci a pour mission de développer la recherche et de valoriser les techniques de l'information et de la communication. L'Inria est composé de 10 centres de recherche, 9 en France et 1 au Chili et est composé de 3900 scientifiques. Le centre de Grenoble a été créé en 1992 et est actuellement dirigé par Frédéric Desprez.

2. My Human Kit et les Humanlabs

My Human Kit (MHK) est une association créée en 2014 qui vise à faciliter la vie des personnes handicapées. Le concept de cette association est que des personnes avec un handicap viennent avec une idée et qu'une aide de conception et de réalisation soit apportée. Cette association est basée à Rennes. Les Humanlabs sont des fablabs qui ont décidé de dédier un atelier pour trouver des solutions en lien avec le handicap et de s'associer à MHK. Les projets portés par les Humanlabs sont documentés et accessibles librement.

3. Humanlab Inria

Humanlab Inria est une action exploratoire et vise à devenir un projet sur le long terme. Elle a pour but de tisser des liens entre les Humanlabs et l'Inria.

4. Équipe intégrée

Pour ce stage, j'ai travaillé pour Humanlab Inria dans le service Service Expérimentation et Développement (SED) à Inria Grenoble.
J'ai notamment été encadré par Roger Pissard-Gibollet et Christophe Braillon.

III. Contexte du projet

1. But du projet

Le but de ce projet est de modifier un fauteuil roulant à assistance électrique pour répondre à la demande du porteur de projet Etienne.
Un fauteuil roulant à assistance électrique est un fauteuil manuel avec des roues spécifiques (voir figure 2). Lorsque l'utilisateur utilise la main courante (barre métallique faisant le tour de la roue) pour avancer, reculer ou freiner, un moteur l'assiste afin de lui faciliter les déplacements.
Cependant, Etienne a une force de serrage de ses mains très faible. Pour bouger la roue, il utilise l'adhérence entre sa main et la barre. Pour améliorer l'adhérence, il a ajouté une gaine plastique à la main courante et il porte des gants. Malgré l'amélioration, lorsqu'il pleut, il peut manquer d'adhérence et donc ne pas pouvoir freiner. Cela peut être dangereux notamment en descente où il ne peut s'arrêter.
Afin de pallier à ce souci, Etienne voudrait que l'on puisse commander le moteur à partir d'un capteur sans contact.
Nous avons pensé à 2 modes de contrôle différents. Dans un premier mode, le capteur renverra une valeur correspondant à la puissance que le moteur va fournir. Ainsi, à puissance fournie fixe, la vitesse, elle, variera.
Pour le second mode, le capteur renverra une vitesse désirée. Ce sera à notre microcontrôleur de calculer la puissance nécessaire que le moteur doit fournir.

Figure 2 : Fauteuil roulant à assistance électrique avec roues M12 d'Alber

2. Attentes du porteur de projet

Les attentes techniques que l'on m'a fixées sont dans un premier temps de comprendre le système et concevoir, si c'était possible, un moyen de s'interfacer dans ce système sans compromettre la sécurité.

Dans un second temps, l'idée serait de réaliser un asservissement en vitesse. En plus de cela, le projet est open-source, il faudra donc que je réalise une documentation afin que ce soit reproductible pour le plus grand nombre. Le choix de la carte et des composants sont libres.
Etienne souhaite conserver le mode de contrôle d'origine. Il faudra donc qu'il puisse changer facilement de mode.

3. Etat de l'art

La seule chose que j'ai trouvé sur internet qui aurait pu ressembler à notre projet aurait été de rajouter une roue motrice à l'avant (voir figure 2). Cependant, Etienne souhaite pouvoir changer facilement de mode entre celui existant et le nôtre. Ce système aurait été trop volumineux et aurait nécessité une adaptation pour lui.

Figure 3 : F-55 de Sunrise Medical

IV. Gestion de projet

1. Gestion du temps

Je suis la première personne à avoir analysé en détail le système. Il était difficile d'estimer le temps nécessaire à celui-ci. Même si la première analyse a pu être rapide, j'ai dû y revenir plus tard car, certaines sécurités n'avaient pas été trouvées.
Au début de la 5ème semaine, j'ai pu participé à un fabrikarium. Il s'agit d'un sorte d'hackathon non compétitif mais collanoratif organisé par le Humanlab Saint pierre, fablab dépendant de l'institut St pierre à Palavas. Durant 3 jours, par équipe de 10, on travaille sur un projet. Certains projets peuvent être soutenu par une entreprise partenaire qui délègue des salariés bénévoles.
Le fabrikarium a été un moment clé où j'ai pu mieux définir le projet et définir quel capteur allait être utilisé.

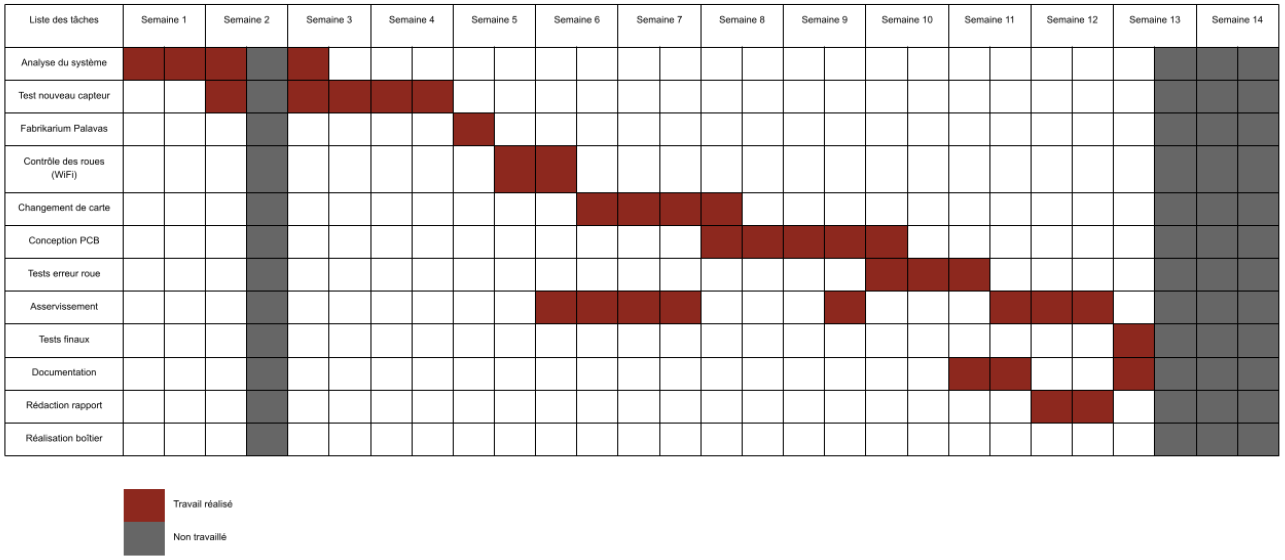


Figure 3 : Diagramme de Gantt

2. Outils utilisés

1. Git et Gitlab

Pour partager des documents, l'Inria a mis en place un espace Gitlab qui utilise Git.
Git est un logiciel de gestion de versions décentralisé. Il permet de sauvegarder des documents et également de gérer leur version. Par exemple, si l'on modifie un document et que l'on y insère une erreur, il est possible de retrouver la version précédente.

2. Mattermost

Mattermost est un service de messagerie instantanée libre. C'est une alternative open-source du logiciel Slack. Il permet un chat interne organisé par “salons”. L'Inria héberge un serveur interne de ce type. Par exemple, pour mon travail je l'ai quotidiennement utilisé pour échanger sur différents salons: équipe SED, action Humanlab Inria, projet Froll'n'Roll et individuellement.

3. KiCad

KiCad est un logiciel libre qui permet de concevoir des schémas électriques, puis l'implémentation des composants et le routage pour définir des dossiers de fabrication de circuits imprimés (ou PCB en anglais pour Printed Circuit Board). Sur ceux-ci, il nous reste juste à souder les différents composants. Cela permet de ne pas avoir de fils qui pourraient se décrocher et d’avoir un produit fini et propre. J’ai eu la chance de pouvoir me former à ce logiciel afin de concevoir deux circuits imprimés.

4. Visual Studio Code

Visual Studio Code est un environnement de développement (IDE) qui permet d’écrire du code. Pour un langage donné, celui-ci peut proposer par l'intermédiaire de plugin, des facilités pour l'écriture du code: la colorisation syntaxique, l'autocomplétion, la navigation par fonction, la refactorisation, le debug, etc…

5. Arduino IDE

Cet IDE est dédié au développement sur Arduino et je l'ai utilisé pour compiler et téléverser et communiquer avec mes programmes sur les différentes cartes que j’ai utilisées durant ce stage. J'ai trouvé cet outil plus accessible que l'extension proposée sous Visual Studio Code.

V. Travaux réalisés

1. Analyse du système existant

1. Matériel pour les tests

Au début de mon stage, j'ai utilisé les M5StickC [1] Plus, boîtier qui est composé d'un ESP-Pico comme microcontrôleur, d'un écran et de capteurs. Ce boîtier est idéal pour les tests car les capteurs sont déjà soudés à la carte, il y a un écran pour afficher des données et des librairies très claires. Cependant, nous n'utilisons pas de M5StickC Plus pour le produit final (détail dans *V.2.1 Idée proposée*). Ce boîtier dispose aussi d'un Convertisseur Numérique-Analogique (CNA) qui renvoie une tension entre 0 et 3.3V. Il s'agit d'un MCP4725 de Microchip[2]. Ce n'était pas suffisant pour imiter le comportement du capteur qui peut renvoyer une tension jusqu'à 4.7V. Nous avons aussi utilisé une carte d'évaluation SOT23-6 qui est composée d'un CNA. Nous l'avons utilisé car ça nous a permis d'utiliser le CNA que nous allons utiliser pour le produit final et d'utiliser une platine d'essais plutôt que de souder.

2. Protocole de communication pour les tests

La carte qui est sur la roue tourne en même temps que celle-ci. Pour les tests, il était nécessaire de pouvoir lui envoyer des données. Pour cela, nous avons utilisé le protocole Message Queuing Telemetry Transport (MQTT). Dans notre cas, la communication se faisait en WiFi.

Premièrement, un appareil va faire tourner le "broker". Il s'agit d'un programme qui reçoit des messages et les renvoie aux "clients" qu'ils lui ont demandés. Il agit en tant que relais. Un appareil peut être broker et client à la fois.

Le MQTT fonctionne par abonnement. C'est-à-dire que des clients (appareil) se connectent au réseau MQTT et s'abonnent à des "topic". Lorsqu'un client envoie un message, il spécifie un "topic" et toutes les personnes qui se sont abonnés à celui-ci le reçoivent.

Dans notre cas,la carte sur la roue crée un réseau wifi et héberge le "broker". Ensuite, mon ordinateur se connecte à ce même réseau. Ainsi, sur mon ordinateur, je peux communiquer avec la carte qui est sur la roue.

J'ai utilisé la librairie qui s'appelle TinyMQTT [3].

3. Outils développés pour les tests

Afin de mieux visualiser les données, mon tuteur a créé une librairie. Celle-ci contient des fonctions qui permettent de s'abonner aux topics demandés et d'enregistrer les données dans un fichier et de les afficher grâce à matplotlib. A l'aide de cette librairie, j'ai pu créer un programme qui prend en argument le type de signal que je désirais et que j'envoyais à la carte, ensuite, le programme récupère les données et me l'affiche à l'écran. Cela m'a permis de gagner beaucoup de temps. Au départ, je modifiais les valeurs à la main puis je téléversais. Entre chaque test, c'était environ 2 minutes de perdues. Avec le programme, je pouvais relancer un test quelques secondes après le précédent. De plus, visualiser les données m'ont permis de m'apercevoir que le moteur limitait la vitesse à un maximum.

4. Capteur existant

Le capteur (figure 4) est fixé au niveau de la jante de la roue. Il s'agit d'un capteur magnétique à effet Hall. Lorsque l'utilisateur bouge la main courante, celle-ci déplace un aimant. Le capteur mesure le champ magnétique et renvoie une tension comprise entre 0.3V et 4.7V. Ce

capteur est alimenté en 5V. Au repos, la tension renvoyée est de 2.5V. A l'aide d'une molette, nous pouvons jouer sur le déplacement de l'aimant. Ainsi, la tension renvoyée pourra être dans un intervalle plus petit ce qui va influencer la puissance fournie par le moteur.

Figure 4 : Capteur originel

5. Sécurités

Lors des tests et notamment lorsque j'ai voulu m'occuper de l'asservissement, la roue se mettait souvent en défaut. Au début, j'ai fait des tests sans outils particuliers. Ces tests m'ont permis de comprendre plusieurs choses :

- Au démarrage, la tension en sortie du capteur doit être égale à 0. La carte attend que le capteur renvoie 2.5V pour prendre en compte la commande. C'est à partir de là que le capteur va commander le moteur. Une pente n'est pas obligé, un créneau ne mettra pas la roue en défaut. Une attente de 2 secondes suffit.

- Ensuite, la carte vérifie son fonctionnement en mesurant la tension entre le Vcc et la masse du capteur. Pour éviter que le système ne se bloque, il faut qu'il y ait une consommation. Celle de la carte suffit à éviter que la roue ne se mette en défaut.

- La tension renvoyée doit être comprise entre 0.3V et 4.7V.

Pour la dernière sécurité, j'ai automatisé au maximum les tests (détail dans *V.1.3 Outils développés pour les tests*). Nous avons supposé que la commande devait revenir au signal de repos régulièrement.
En effet, lorsque la roue est en mouvement, la commande ne peut pas rester trop importante trop longtemps. Par exemple, si l'on met la commande à 4.7V, la roue se mettra en défaut après environ 1.5 secondes. Il faut par conséquent que la commande redescende à 2.5V de temps en temps. Sur la figure 5, il y a les signaux extrêmes. Ttotal correspond à une période avec Tlibre où la tension renvoyée peut varier de 0.3V à 4.7V puis 5ms avec une tension renvoyée qui sera forcément égale à

une tension de repos.

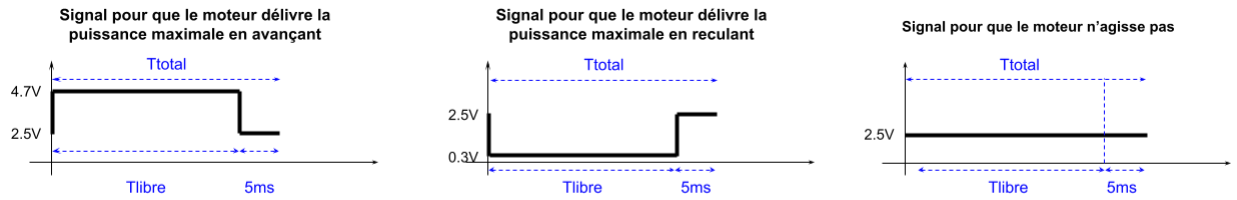


Figure 5 : Signaux extrêmes renvoyés à la roue

En répétant le premier signal de la figure 5 (ci-dessus) et en mesurant la vitesse, nous obtenons la figure 6. Ce graphique a été réalisé avec la roue en l'air donc peu de frottement. Nous voyons qu'il y a une vitesse limite d'environ 510°/s.

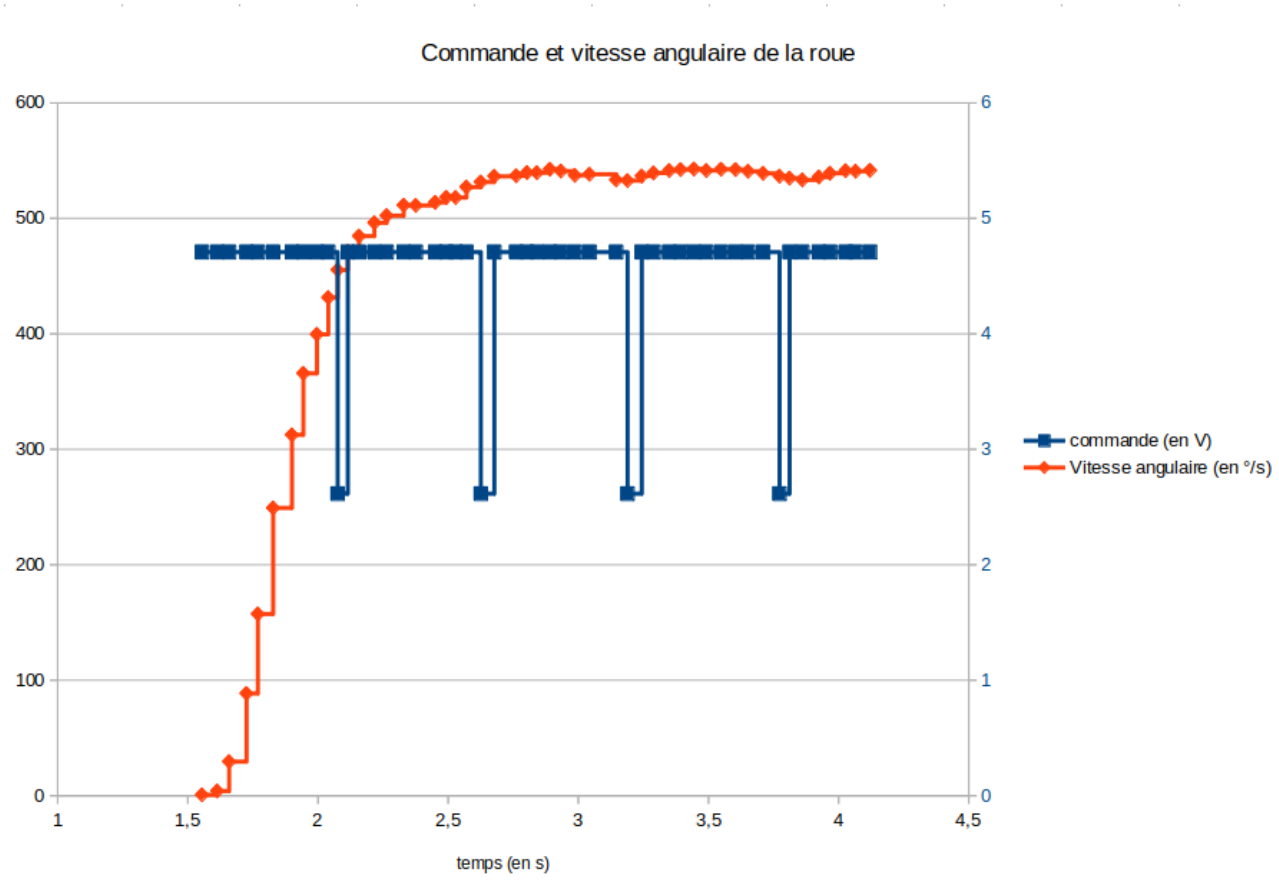


Figure 6 : Vitesse angulaire et commande lors d'un test

2. Conception du nouveau système

1. Idée non retenue

Nous avions comme idée d'utiliser la distance entre un capteur et une main comme moyen de commande.
On souhaitait que le capteur soit fixé sur la roue afin de n'avoir qu'une seule carte.
La problématique que nous avions, est que la roue tourne, il fallait donc trouver un moyen pour que l'utilisateur puisse commander la roue quelque soit l'angle qu'elle avait.
Au départ, nous pensions utiliser un capteur capacitif. Celui-ci a besoin d'une plaque conductrice qui aurait fait le tour de la roue (plaque cuivrée sur la figure 7). Le principe est que lorsque l'on rapproche la main, la tension évolue.
Pour cela, j'ai utilisé le code disponible sur le site aranacorp [4].
Cependant, la variation possible de la tension était très faible. Il aurait facilement été instable. De plus, un petit peu d'eau sur la plaque aurait suffit à tromper le capteur.

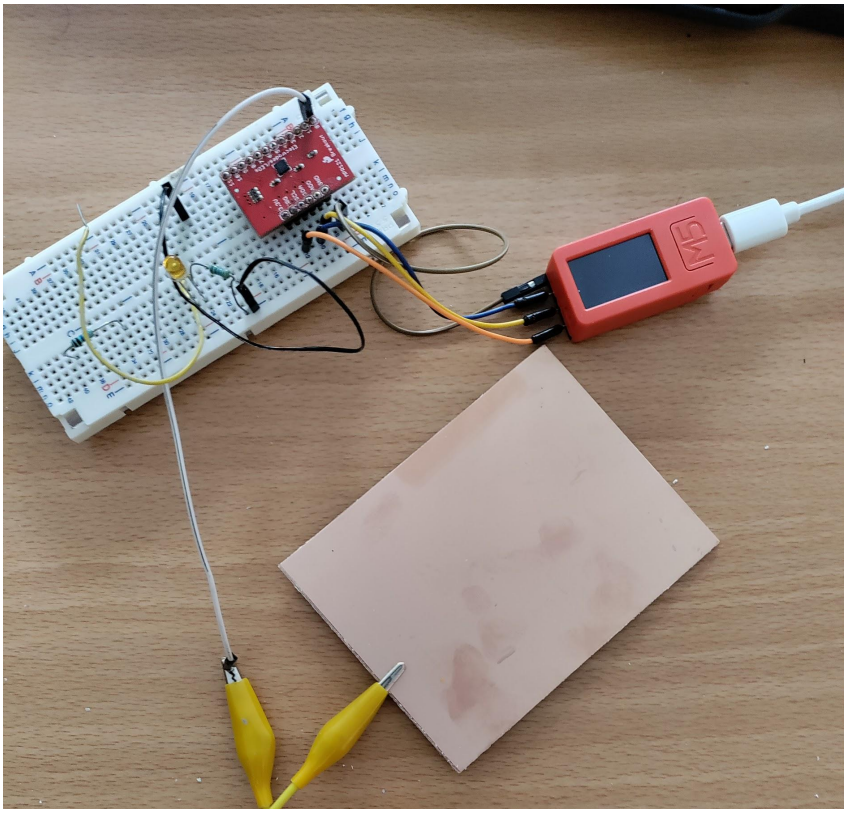


Figure 7 : Test d'un capteur capacitif

2. Idée retenue

Il y a 2 cartes : une au niveau de la roue et une au niveau de la main de l'utilisateur. Sur la carte de la main, il y a un capteur (détail partie *V.2.3 choix du capteur*), la carte envoie à celle de la roue une commande de vitesse.
Sur la carte de la roue, il y a un switch qui permet de choisir entre le mode de commande existant ou le nôtre. Cette carte reçoit une commande de vitesse. Elle réalise un asservissement en vitesse avec un PID (Proportionnel Intégral Dérivé). Ensuite, elle communique avec un CNA par Inter-Integrated Circuit (I2C) qui renvoie à la roue un signal analogique.
Les deux cartes communiquent sans fil.

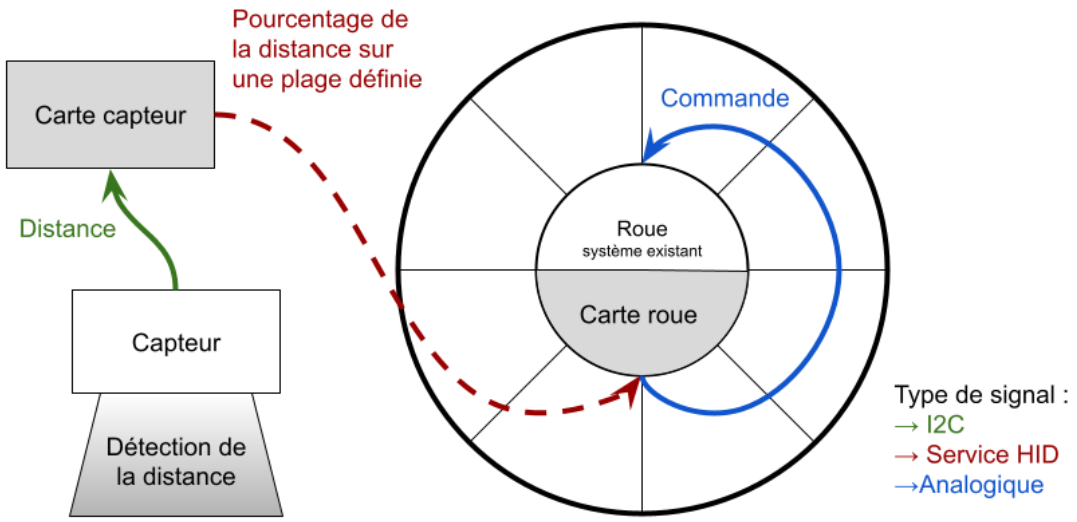


Figure 8 : Organisation générale

3. Choix du capteur

Nous avons choisi d'utiliser la distance entre la main de l'utilisateur et un objet comme moyen de commande.
Pour cela, nous utilisons un capteur ToF (Time of Flight) car il permet de détecter tous les matériaux. Nous avons choisi le capteur Adafruit VL53L0X [5] car celui-ci communique en I2C. Nous avons défini que la mesure serait comprise entre 20 et 200 millimètres (paramètres changeables dans un fichier config).
Entre 0 et 20mm, la vitesse désirée est la vitesse maximale en arrière.
La vitesse désirée augmente proportionnellement à la distance jusqu'à atteindre la vitesse maximale en avançant.
Si le capteur mesure une vitesse hors de cet intervalle ou n'en mesure plus, une donnée spéciale est renvoyée et la roue se met en roue libre.

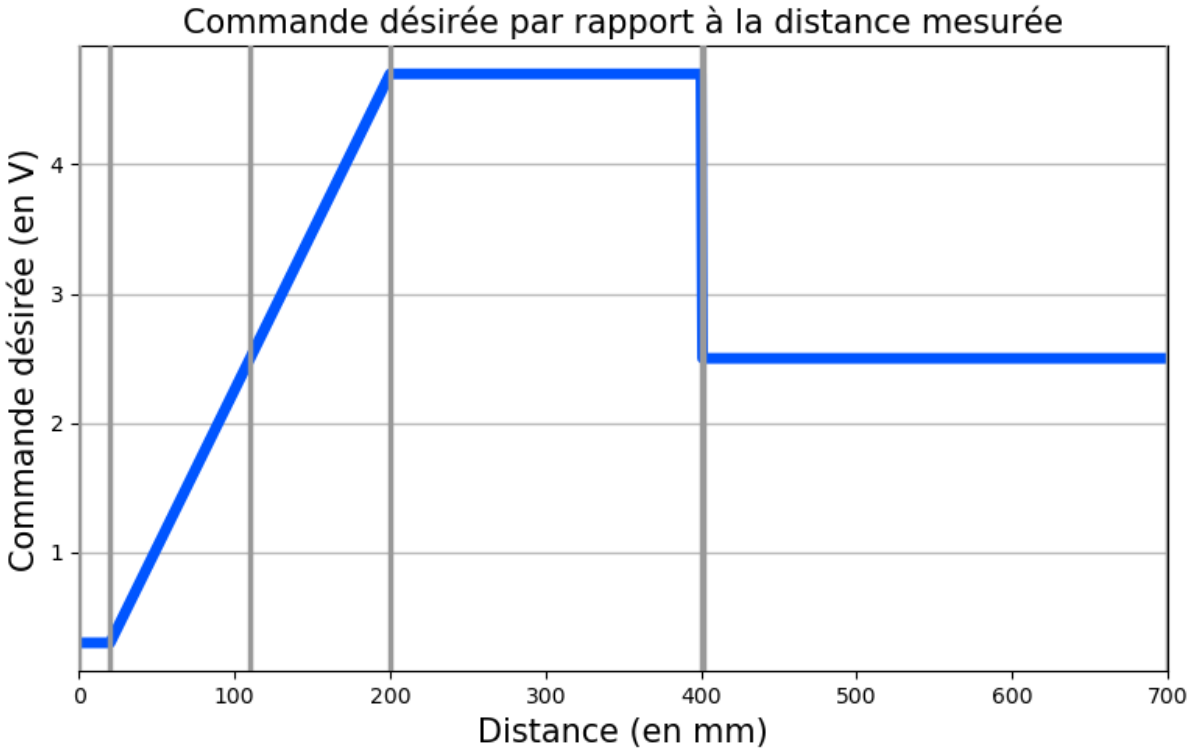


Figure 9 : Commande désirée par rapport à la distance mesurée par le capteur

4. Choix de la communication

Il s'agit d'un service Human Interface Device (HID) par le Bluetooth Low Energy (BLE). Nous avons choisi le BLE car il est économe en énergie. Ce service est notamment utilisé pour les claviers car il utilise des évènements'. Cela permet une plus grande vitesse du système.
La donnée envoyée est un caractère. Afin que les données soient des caractères imprimables, nous avons décidé que les données aient 50 valeurs possibles. Nous les convertissons en caractères imprimables qui sont ensuite envoyés.

5. Choix des cartes

Au départ, nous avions des M5StickC Plus qui ont été très pratiques pour les tests. Cependant, pour la version finale, nous avons privilégié l'utilisation de 2 cartes légèrement différentes. La carte qui sera au niveau du capteur est une carte Adafruit Feather nRF52840 Express [6]. Nous allons aussi utiliser la carte Adafruit Feather nRF52840 Sense [7] qui sera positionnée sur la roue. C'est la même carte que la précédente mais dotée d'un gyromètre.
Nous avons privilégié ces roues car elles ont une moins grande consommation électrique.

6. Réalisation du PCB

J'ai réalisé 2 PCBs. Il s'agit de cartes électroniques où l'on vient ajouter des composants dessus. J'ai commencé par dessiner la carte avec le capteur dessus. Sur KiCad (présenté partie *IV.2.3 Kicad*), il y a 2 grandes étapes à faire. La première est la schématique, nous ajoutons tous les composants sur une feuille et on crée les liens entre les broches(voir *Annexes 1 et 3*). Dans un second temps, nous nous occupons du routage entre les composants. A l'écran, nous voyons les

empreintes (version 2 dimensions des composants). Nous pouvons les déplacer puis on crée réellement les pistes (liens) entre chaque broche. Nous ne pouvons pas avoir deux pistes qui se croisent. Lorsque le PCB sera imprimé, cela correspond à un fil. Si deux pistes se croisent, alors le signal d'une broches ira à des broches non désirées.
Sur les annexes 2 et 4, vous pouvez voir le routage des 2 cartes.

Sur KiCad, il existe des bibliothèques avec des composants déjà enregistrés. C'est le cas de tous les composants exceptés le capteur ToF. Il a donc fallu le dessiner. Heureusement, il y avait dans la documentation de celui-ci un schéma qui permettait la réalisation de l'empreinte. Nous avons commandé sur JLCPCB [8] (entreprise qui imprime les PCBs) et les circuits qui sont arrivés quelques jours plus tard.

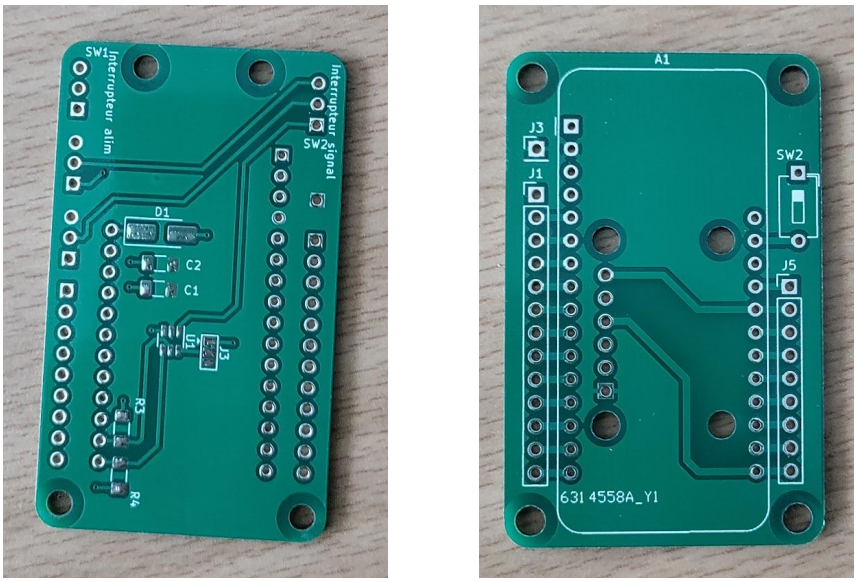


Figure 10 : PCBs après impression

7. Signal envoyé

Nous avons vu dans la partie V.1.5 sécurités que la commande devait revenir à la tension de repos (2.5V) de temps en temps.
Au départ, nous pensions utiliser une sorte de PWM. J'avais mesuré que pour que le moteur accélère suffisamment vite avec moi dessus, il fallait une période de 300ms. Si cette période était moins longue alors le système aurait mis trop de temps à monter à sa vitesse maximale. Mais nous voulions que cette valeur soit la plus faible possible afin que la période de rafraîchissement soit la plus grande possible. Dans cette période, il y avait au minimum 5ms au repos et le reste était à l'état haut (soit 4.7V ou 0.3V selon le sens de la roue). Si nous ne souhaitions pas que le moteur délivre le maximum de sa puissance alors la durée à l'état haut diminuait et celle à l'état bas augmentait. Cela impliquait 2 soucis. Premièrement, les 300ms correspondent à mon fauteuil et à mon poids, c'est une variable que chaque utilisateur aurait dû modifier. Deuxièmement, pour mon cas, nous aurions eu une fréquence d'échantillonnage de 3Hz, ce qui est extrêmement faible.
Pour ce type d'utilisation, 10Hz est un minimum.
Nous avons retenu une méthode différente. La nouvelle période d'échantillonnage est de 5ms. A chaque période, le système envoie une commande au moteur comprise entre 0.3 et 4.7V. Mais pour que la roue ne se mette pas en défaut, toutes les secondes, nous envoyons une commande de 5ms de la tension de repos. Ainsi, nous obtenons une fréquence d'échantillonnage de 200Hz. Le fait de forcer que toutes les secondes, la commande renvoyée soit au repos ne se ressent pas dû à l'inertie du système. Cependant, durant mes tests où la roue était en l'air, nous pouvions voir que la roue ralentissait.
Sur la figure 11, on peut voir un exemple avec la carte qui envoie une rampe sur la commande.

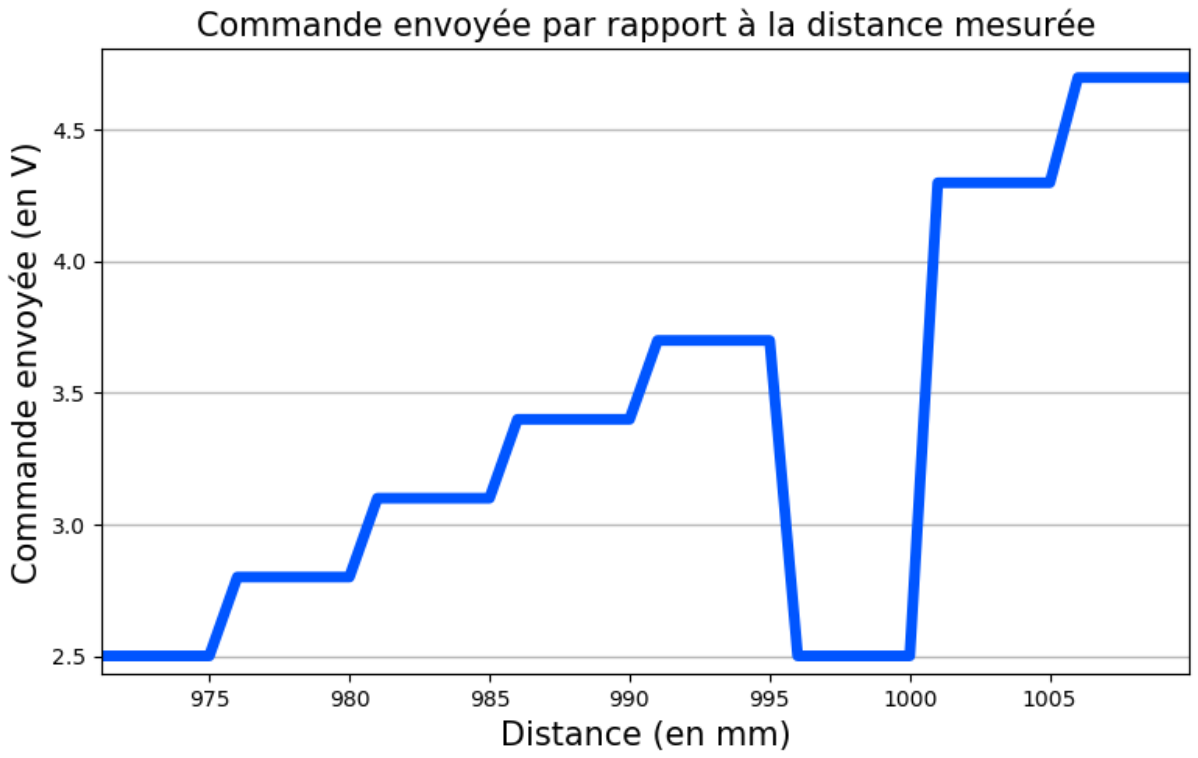


Figure 11 : Exemple de commande en rampe avec la gestion de sécurité

8. Organisation du code

Afin de faciliter l'utilisation du code, celui-ci a été divisé en plusieurs modules.
Pour la carte de la roue et du capteur, il y a le fichier principal avec le setup() et le loop(). Ce sont les deux fonctions obligatoires sur Arduino. Il commence par exécuter la fonction setup() puis il exécute en boucle la fonction loop() jusqu'à l'extinction de la carte. Il y a aussi des fichiers qui contiennent les fonctions BLE dans chaque dossier. Ceux-ci ne sont pas destinés à être modifiés.
Pour la carte du capteur, j'ai créé des fichiers pour le capteur ToF. Il n'y a qu'une seule fonction et elle renvoie un pourcentage. Afin de pouvoir les envoyer avec le service HID du BLE, les données sont sur 50 et non sur 100. Cela correspond à la distance mesurée par le capteur par rapport à la distance maximum. J'ai choisi de mettre cette fonction dans un fichier séparé car il est possible qu'une personne récupère le projet mais qu'en voulant l'adapter, elle change de capteur.
Enfin, pour chaque carte, il existe un fichier de configuration avec toutes les constantes. Il sera donc plus facile pour les personnes de modifier certaines valeurs qui leur seront propres.

9. Améliorations possibles

Aujourd'hui, il y a des boîtiers mais ceux-ci ne sont pas étanches. Pour une utilisation quotidienne, cela va poser des soucis.
On se pose encore la question de l'asservissement en vitesse. C'est-à-dire que le capteur ne renverrait pas une puissance que le moteur doit fournir mais une vitesse désirée. Ce serait ensuite à la carte de calculer quelle puissance le moteur doit délivrer afin d'arriver à cette vitesse. La commande serait différente et c'est une question d'ergonomie.
Enfin, le dernier changement que j'aurai voulu réaliser concerne la commande désirée en fonction de la distance. Actuellement, comme on peut le voir figure 9, si la distance détectée par le capteur est nulle, la commande sera d'aller à la vitesse maximale en reculant. Par conséquent, cela complique l'action d'éteindre le moteur. En ajoutant une zone comme sur la figure 12, cela permettrait d'arrêter la roue en sécurité.

Il est souvent utile d'utiliser la roue libre. Celle-ci n'est plus entraînée par le moteur et freine doucement.. C'est uniquement au milieu de la zone définie (110mm pour notre cas) que le moteur est à l'arrêt. En ajoutant une zone "morte" comme sur la figure 12, cela permettrait de laisser le moteur à l'arrêt plus facilement.

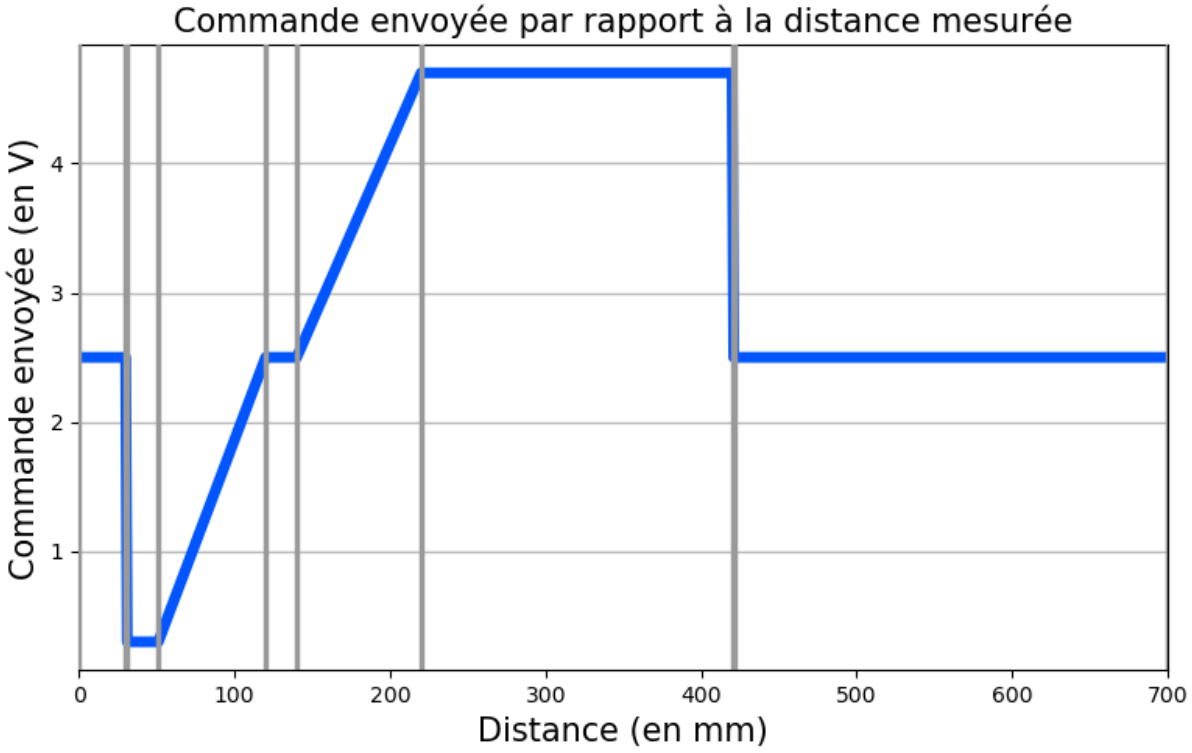


Figure 12 : Nouvelle commande désirée par rapport à la distance mesurée par le capteur

VI. Livrables

A la fin de mon stage, j'ai pu démontrer la faisabilité du projet et arriver à un prototype avancé. J'ai réalisé la documentation sur le travail que j'ai effectué. Elle est actuellement sur un serveur privé mais sera publiée dès que le projet sera vraiment terminé.
Celle-ci explique comment fonctionne le moteur et comment ajouter un capteur sans que la roue ne se mette en défaut (Partie V.2.8 Améliorations possibles).
Le code de la carte du capteur a été réalisé.
Parmi les attentes d'Etienne, il y avait d'avoir 2 modes de commande : en boucle ouverte et fermée.
Le code pour la boucle ouverte a été réalisée et testée. Celui pour la commande en vitesse a été réalisée mais je n'ai pas eu le temps de le tester. Dans le temps qui m'était imparti, il était plus important de rédiger la documentation.

VII. Conclusion

1. Conclusion du projet

A l'heure actuelle, on dispose d'un prototype qui fonctionne en environnement contrôlé. Il reste encore du travail de finition pour disposer d'un prototype utilisable 'tous les jours'. En particulier, il faut réaliser un boitier pour protéger le système de l'eau et des poussières. Il faut également ajouter un interrupteur pour passer du mode standard au mode capteur. Cet ajout d'interrupteur a d'ailleurs été prévu sur le PCB.

2. Conclusion personnelle

Ce stage a été extrêmement enrichissant pour moi. J'ai pu trouver un projet qui m'intéressait sur le plan technique mais aussi personnel. J'ai découvert un milieu qui mêlait ingénierie et associatif dont j'ignorais l'existence. Ainsi, ce stage m'a donné une raison de plus de devenir ingénieur.
Ce stage m'a permis de développer des compétences techniques notamment concernant la conception d'un prototype. En effet, j'ai dû concevoir un produit en respectant le cahier des charges. Pour cela, j'ai choisi les composants en vérifiant qu'ils étaient compatibles.
Une autre des compétences que j'ai pu améliorer est le partage des connaissances. En effet, ce projet étant accessible à tous, je me suis occupé d'expliquer le produit ainsi que l'algorithme afin que chacun puisse s'en servir ou l'améliorer.

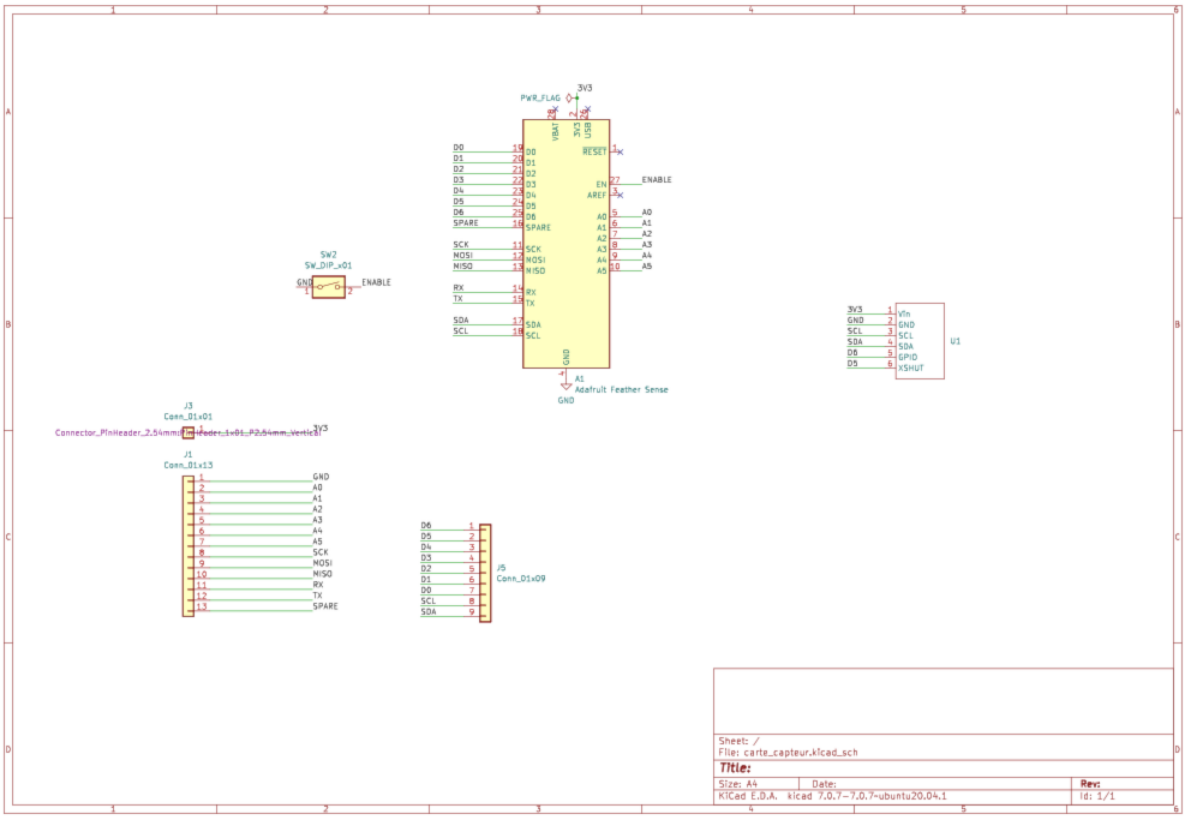
VIII. Bibliographie

[1] Carte de développement M5StickC Plus : https://docs.m5stack.com/en/core/m5stickc_plus
[2] Convertisseur Numérique-Analogique MCP4725 : https://ww1.microchip.com/downloads/aemDocuments/documents/MSLD/ProductDocuments/DataSheets/MCP4725-Data-Sheet-20002039E.pdf
[3] TinyMQTT : https://www.arduino.cc/reference/en/libraries/tinymqtt/
[4] Capteur capacitif : https://www.aranacorp.com/fr/utilisation-dun-capteur-capacitif-avec-arduino/
[5] Capteur ToF Adafruit VL53L0X : https://learn.adafruit.com/adafruit-vl53l0x-micro-lidar-distance-sensor-breakout
[6] Adafruit Feather nRF5280 Express : https://learn.adafruit.com/introducing-the-adafruit-nrf52840-feather
[7] Adafruit Feather nRF5280 Sense : https://learn.adafruit.com/adafruit-feather-sense/overview

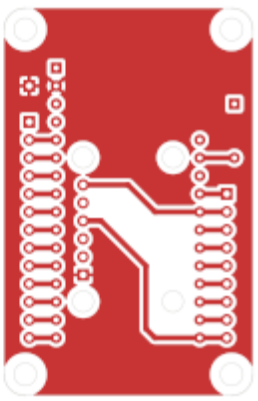
[8] JLCPCB : https://jlcpcb.com/

IX. Annexes

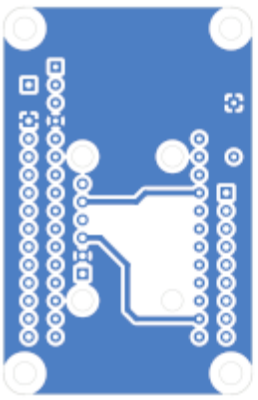
Annexe 1 : Schématique de la carte capteur



Annexe 2 : Routage de la carte capteur

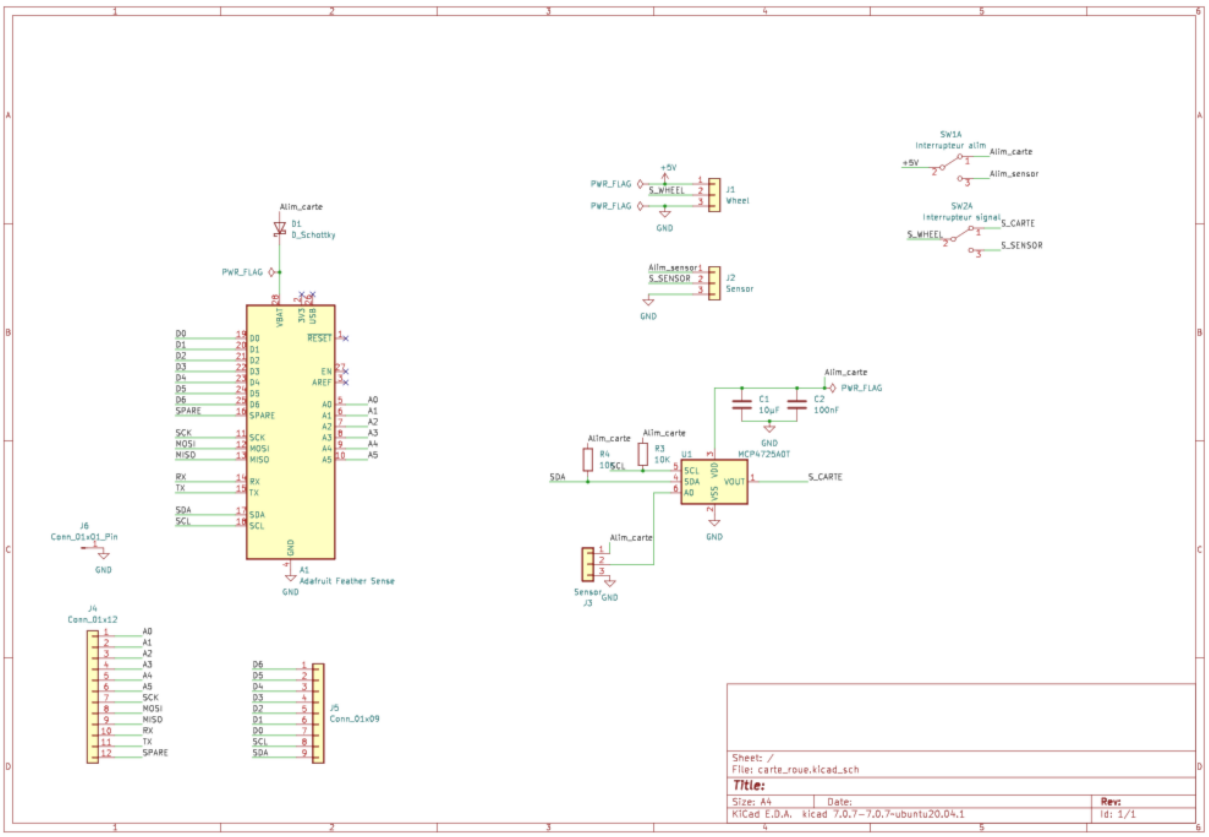


Face avant

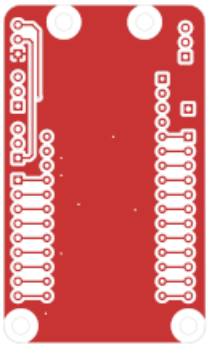


Face arrière

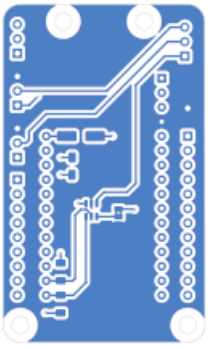
Annexe 3 : Schématique de la carte de la roue



Annexe 4 : Routage de la carte de la roue



Face avant



Face arrière

Etudiant : Johan Pédroli

Année d'étude dans la spécialité : IESE4

Entreprise : Inria Grenoble-Rhône Alpes
Adresse complète : 655 Av. de l'Europe,
38330 Montbonnot-Saint-Martin
Téléphone : 0476615200

Responsable administratif : M. Frédéric Desprez
Courriel : frederic.desprez@inria.fr

Tuteur de stage : M. Roger Pissard-Gibollet
Téléphone : 0612150293
Courriel : roger.pissard@inria.fr

Enseignant-référent : M. Sylvain Toru
Téléphone : 0476827915
Courriel : Sylvain.Toru@univ-grenoble-alpes.fr

Titre : Projet Froll'n'Roll

Dans ce rapport, je vais vous présenter ce que j'ai réalisé durant mon stage à l'Inria Grenoble qui a duré 3 mois. Travaillant pour le projet Humanlab Inria, je me suis occupé de modifier un fauteuil roulant à assistance électrique. Suite à la demande d'un des utilisateurs, j'avais pour mission de contrôler le moteur présent dans les roues en installant le système entre le capteur originel et la carte de contrôle. Cependant, je n'étais pas dans l'entreprise qui a conçu ce fauteuil et n'avais aucun accès aux documents techniques. Une grande partie de mon travail a donc été de comprendre quelles sécurités avaient été mises en place par le constructeur et comment nous pouvions diriger le moteur sans interagir avec celles-ci. En parallèle, j'ai cherché quel capteur pouvait correspondre à l'utilisation du porteur de projet. Ce projet a pour but d'être partagé au plus grand monde. Il sera donc sous licence open-source. Une partie de mon travail a donc été d'établir une documentation pour que des personnes initiées puissent le réaliser. Je vais donc vous expliquer comment je suis passé roue avec le capteur originel à une roue contrôlée par une roue dirigée par une carte avec un capteur de distance qui fonctionne dans un environnement contrôlé.

In this report, I'm going to present what I achieved during my 3-month internship at Inria Grenoble. Working for the Humanlab Inria project, I was in charge of modifying an

electrically-assisted wheelchair. At the request of one of the users, I had to control the motor in the wheels by installing the system between the original sensor and the control board. However, I wasn't in the company that designed the wheelchair and had no access to the technical documents. So a big part of my job was to understand what safeguards had been put in place by the manufacturer, and how we could steer the motor without interacting with them. At the same time, I looked for a sensor that would be suitable for the project leader's application. The aim of this project is to be shared with as many people as possible. It will therefore be released under an open-source license. So part of my job was to draw up documentation so that people with experience could build it. So I'm going to explain how I went from a wheel with the original sensor to a wheel controlled by a card-driven wheel with a distance sensor that works in a controlled environment.