Rapport Projet SSL

Partie Capteurs et communication :

Avec Clément SARRAUD, David CHAUVET et Valentin BAUDET

Table des matières

I. Présentation du projet :

Le but de se projet est de réaliser un robot holonome pouvant participer à la robocup 2020 à Bordeaux. Cette compétition a pour but de faire s'affronter des équipes de six robots pour disputer des matchs de football. Les robots doivent être capable de se déplacer dans toutes les directions, se sont des robots holonomes et ils sont contrôlés par un ordinateur qui possède une vision générale du jeu grâce à des caméras présentes au-dessus du terrain.

Par ailleurs, afin de mener à bien ce projet, nous avons quelques contraintes qu'il est nécessaire de prendre en compte :

- Le robot doit obligatoirement être commander grâce à un microcontrôleur ARM de la famille STM32.
- Le robot doit être holonome et équipé de 4 roues holonomes.
- Le robot doit avoir un dribleur à l'avant de sorte à pouvoir contrôler la balle
- Le robot doit avoir un kickeur lui permettant de tirer la balle
- Le robot doit avoir une forme cylindrique avec un diamètre et une hauteur n'excédant pas 25 cm.
- Le robot doit être alimenté par une batterie LiPo 3S de 12V
- Le robot doit embarquer la gestion de la charge de la batterie ainsi que l'état de charge.
- Le robot doit avoir un capteur à effet Hall sur chaque moteur, ainsi qu'un capteur angulaire magnétique, une centrale inertielle, une barrière infrarouge et deux capteurs optique.
- Le robot doit être capable de communiquer avec l'ordinateur avec un module nRF24L01 afin de transmettre des informations sur l'état du robot et de réaliser les ordres qui lui sont transmis par l'ordinateur.
- Le robot doit avoir une autonomie de 20 à 35 minutes

Dans le cahier des charges, il existe d'autre contraintes tels que des contraintes liées au matériels et logiciels à utiliser qui sont Altium Designer et la programmation des microcontrôleurs de la famille des STM32 en langage C et/ou python, des contraintes environnementales, puisque le robot doit pouvoir fonctionner en intérieur sur de la moquette et des contraintes économiques, puisque le coût total du robot ne doit pas excéder 500 euros.

Afin de réaliser au mieux ce projet et en respectant le cahier des charges, il nous est également possible d'utiliser les moyens que l'on possède à l'école, tels que :

- Les outils de CAO
- Les moyens de réalisations des PCB
- Les appareils de test ou de mesure
- Les cartes de développement de la famille des STM32

Nous pourrons également utiliser les documentations techniques et spécifiques nécessaire à la réalisation de notre projet en ligne ou hors ligne.

II. Gestion du projet

1. Répartition des taches :

Comme l'indique le cahier des charges, il existe plusieurs fonctions principales que nous devons partager. Les différentes tâches du projet sont :

- Réalisation des ESC
- Contrôle des moteurs avec gestion PWM et asservissement des roues
- Calcul du déplacement du robot
- Mise en œuvre du dribleur
- Mise en œuvre du kicker
- Communication Radio Fréquence
- Conception de la carte mère
- Gestion de la batterie

Après avoir vu avec les personnes du projet et au vu des contraintes que nous avons dû au fait que David soit dans l'obligation de partir en janvier 2019, nous avons décidé de partager les taches du projet comme suit :

a. Clément :

- Mise en œuvre du dribleur
- Mise en œuvre du kicker

} Fonction FP6

b. David :

- Réalisation des ESC Fonction FP5

c. Matthieu :

- Contrôle des moteurs avec gestion PWM et asservissement des roues
- Calcul du déplacement du robot

} Fonction FP3

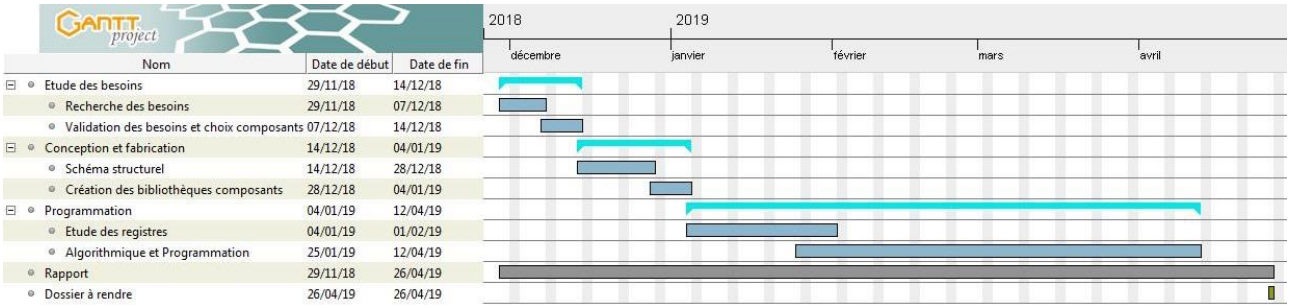
- Communication Radio Fréquence Fonction FP2

d. Valentin :

- Conception de la carte mère Fonction FP4
- Gestion de la batterie Fonction FP1

2. Planification :

Afin de réaliser et d'assimiler au mieux le travaille que je dois faire, j'ai établi un diagramme de Gantt prévisionnel. Ce diagramme a donc pour but de prévoir le temps que je vais allouer à chaque partie du projet.



Nom	Date de début	Date de fin
Etude des besoins	29/11/18	14/12/18
Recherche des besoins	29/11/18	07/12/18
Validation des besoins et choix composants	07/12/18	14/12/18
Conception et fabrication	14/12/18	04/01/19
Schéma structurel	14/12/18	28/12/18
Création des bibliothèques composants	28/12/18	04/01/19
Programmation	04/01/19	12/04/19
Etude des registres	04/01/19	01/02/19
Algorithmique et Programmation	25/01/19	12/04/19
Rapport	29/11/18	26/04/19
Dossier à rendre	26/04/19	26/04/19

Figure 1 : Gantt prévisionnel Robot SSL

III. Analyse Fonctionnelle :

1. Schéma fonctionnel des capteurs :

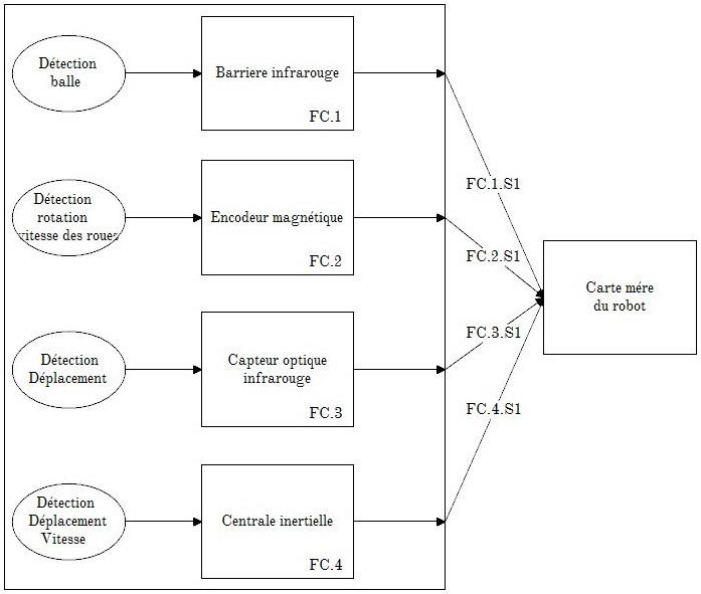


Figure 2 : Schéma fonctionnel de niveau 1 des capteurs

Ce schéma fonctionnel est un schéma fonctionnel de niveau1, il sert à expliquer de manière très généralistes les fonctions que l'on retrouve pour les capteurs.

2. Schéma fonctionnel de la radiocommunication :

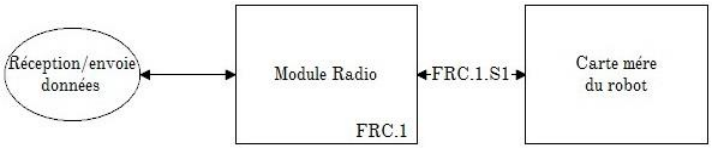


Figure 3 : Schéma fonctionnel de niveau 1 de la radiocommunication

Grâce à ce schéma, on remarque les liaisons et les fonctions qui existent pour la communication entre le robot et l'ordinateur.

3. Description des signaux :

- FC.1 : Fonction Capteur 1, elle représente la fonction de la barrière infrarouge qui détecte la balle dans le dribbler. Lié à cette fonction, on retrouve la fonction
- FC.1.S1 : Fonction Capteur 1 secondaire 1, elle lie la Fonction Capteur 1 avec la carte mère. Elle utilise une communication de tout ou rien qui vient dire à la carte s'il y a une balle ou s'il n'y a rien.

- FC.2 : Fonction Capteur 2, elle sert à détecter la vitesse et le sens de rotation des roues.
- FC.2.S1 : Fonction Capteur 2 Secondaire 1, elle sert à lier la Fonction Capteur 2 avec la carte mère du robot. Elle utilise une communication SPI (Serial Peripheral Interface).

- FC.3 : Fonction Capteur 3, elle a pour but de détecter et de mesurer la valeur de déplacement du robot ainsi que sa vitesse.
- FC.3.S1 : Fonction Capteur 3 secondaire 1, elle lie la Fonction Capteur 3 et la carte mère grâce à une communication SPI (Serial Peripheral Interface).

- FC.4 : Fonction Capteur 4, la fonction permet de mesurer l'accélération et le déplacement du robot.
- FC.4.S1 : Fonction Capteur 4 Secondaire 1, elle lie la Fonction Capteur 4 à la carte mère du robot. Elle utilise un protocole de communication I^2C (Inter-Integrated Circuit).

- FRC.1 : Fonction RadioCommunication 1, elle a pour but de réaliser la communication entre le robot et l'ordinateur.
- FRC.1.S1 : Fonction RadioCommunication 1 Secondaire 1, elle permet de lier la Fonction RadioCommunication 1 avec la carte. Cette communication utilise un protocole SPI (Serial Peripheral Interface).

IV. Analyse des besoins :

Comme vu précédemment, il est nécessaire d'ajouter différents capteurs afin de s'assurer du bon fonctionnement du robot.
Ainsi les fonctions à traiter sont les suivantes :

- Vitesse et sens de rotation de chaque roue
- Vitesse du robot
- Déplacement du robot
- Position du robot dans l'espace
- Détecter la présence de la balle

Afin de réaliser ces fonctions, nous allons utiliser plusieurs capteurs. Cependant, comme il l'est demandé dans le cahier des charges, nous devons réaliser une redondance. Autrement dit, afin d'être sûr de nos valeurs, plusieurs capteurs vont réaliser la même fonction.

De plus, comme nous cherchons à diminuer au maximum les dépenses, j'ai utilisé certains capteurs que nous possédions déjà à l'école : tels que les encodeurs et les capteurs optique infrarouge.

1. Barriere infrarouge dribbler :

Il est nécessaire de déterminer si la balle est présente devant les dribbler du robot. Ainsi, il est important d'apporter une solution technique fiable. Le problème lié à la détection par infrarouge est qu'il est possible que la lumière ambiante vienne altérer la valeur reçue par le capteur et ainsi indiquer au robot qu'il n'a pas la balle alors qu'il l'a.

Pour éviter ce problème, il est préférable d'utiliser un angle de directivité plutôt faible pour le récepteur. Cette question ne se pose pas pour l'émetteur.
De plus, il est nécessaire de faire attention à ce que la longueur d'onde émise puisse bien être lu par le récepteur.

Mon choix c'est donc tourné vers une diode électroluminescente infrarouge en temps qu'émetteur et d'un récepteur infrarouge pour détecter la présence ou l'absence de la balle.

Pour réaliser la barrière, je place une led infrarouge émettant avec un angle de 10°. Je cherche à obtenir un angle le plus petit possible, de sorte que l'on minimise au mieux les interférences dû au facteurs environnant. Mon choix c'est donc porté sur la led OFL-5102 ayant une longueur d'onde de 940 nm à un courant de 20 mA, un angle d'émission de 10° et une tension de fonctionnement 1.25V.

Dans l'axe de la led, je place un récepteur infrarouge. Ce récepteur fonctionne avec un courant de 5 mA et une tension allant de -0.3 à +6V. De plus je sais que le récepteur et la led sont compatibles puisque le spectre de réception est suffisamment large. Sa réception est optimale pour une longueur d'onde allant de 900 à 970 nm.

Le fonctionnement du capteur est le suivant :

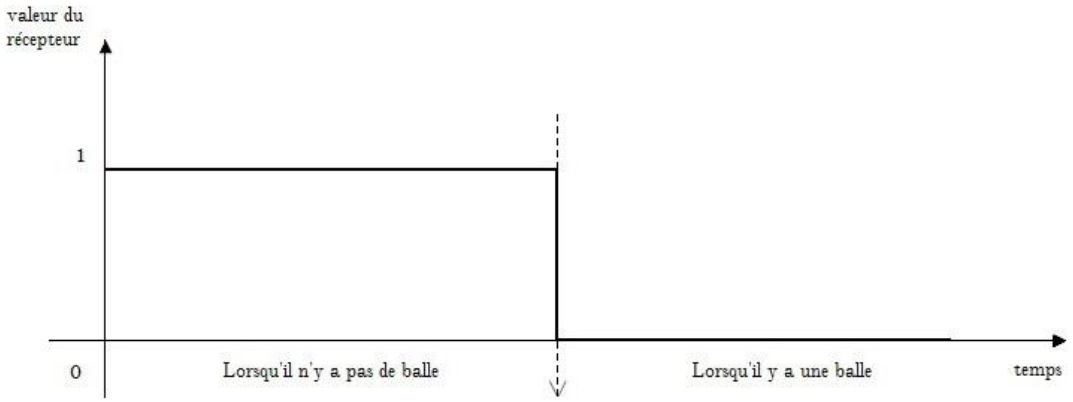


Figure 4 : Schéma explicatif du fonctionnement du capteur IR du dribleur

2. Encodeurs :

Pour être sûr de la vitesse de déplacement du robot, et de son bon contrôle, il est nécessaire d'asservir la vitesse et le sens de rotation de chaque moteur. Afin de réaliser cet asservissement, on utiliser des encodeurs.

Les encodeurs sont des capteurs à effet hall, ainsi, à la sortie de chaque capteur, on observe un signal PWM (Pulse Width Modulation). On connait la fréquence du signal, on connait le nombre de point (précision) du capteur. Ainsi on peut facilement déterminer la vitesse de chaque roue. De plus, comme un capteur possède deux canaux, il est possible de déterminer le sens de rotation en fonction du déphasage (avance ou retard de phase).

Il existe déjà une technologie d'encodeur présente à l'école : le capteur AS5048A.

Il est possible de récupérer les valeurs de ce capteur grâce à un protocole de communication SPI ou bien de la sortie PWM présente sur le capteur. Le capteur possède une résolution de conversion de 14 bits, peut être alimenter en 3.3V ou en 5V.

3. Capteur optique infrarouge :

Afin d'avoir des informations sur le déplacement du robot, on peut utiliser un capteur optique infrarouge. Ce capteur indique donc le déplacement du robot.

Ainsi, il est possible de déterminer son déplacement par rapport au point de départ.

Je peux utiliser les capteurs qui sont à l'école qui sont les ADNS3080.

4. Centrale inertielle :

Une centrale inertielle, aussi appelée IMU (Inertial Measurement Unit) regroupe plusieurs capteurs tels qu'un accéléromètre, un gyroscope ainsi qu'un compas magnétique. Son but est donc de donner l'accélération du robot (grâce à l'accéléromètre), la vitesse angulaire (à l'aide du gyroscope) et l'orientation du robot par rapport au champ magnétique terrestre (dû au compas).

Dans notre cas, l'utilisation d'une centrale inertielle sert à plusieurs choses :

- Fournir une redondance d'information en particulier pour la vitesse globale du robot.
- Estimer sa position par rapport à son point de départ.

- Apporter la certitude que le robot se trouve bien à la cage de l'équipe adverse et qu'il ne va pas tirer dans son propre camp.

Minimu-9 v3 (pololu) GY-525 MPU6050

5. Communication Radio :

Pour la communication radio, le module est imposé. C'est le module nRF24L01.

Il a pour but de réaliser la communication entre l'ordinateur et le robot, de lui transmettre les ordres de déplacements mais aussi toutes informations utiles pour le robot. Cependant, ce module ne va pas juste être utilisé de sorte que l'ordinateur envoie les ordres au robot ; mais il servira également à transmettre les informations du robot :

- Information de batterie
- Information de position et de vitesse
- Information de possession de balle

Pour conclure sur les besoins et en particulier sur les capteurs, on peut constater que certains ont des fonctions très proches voir même identique. Le fait que les capteurs remplissent les mêmes fonctions est voulu. C'est à la fois une demande du cahier des charges et à la fois une nécessité pour être sûr que les valeurs de vitesses, de déplacements sont bonnes.

V. Analyse structurelle :

1. Schéma Altium :

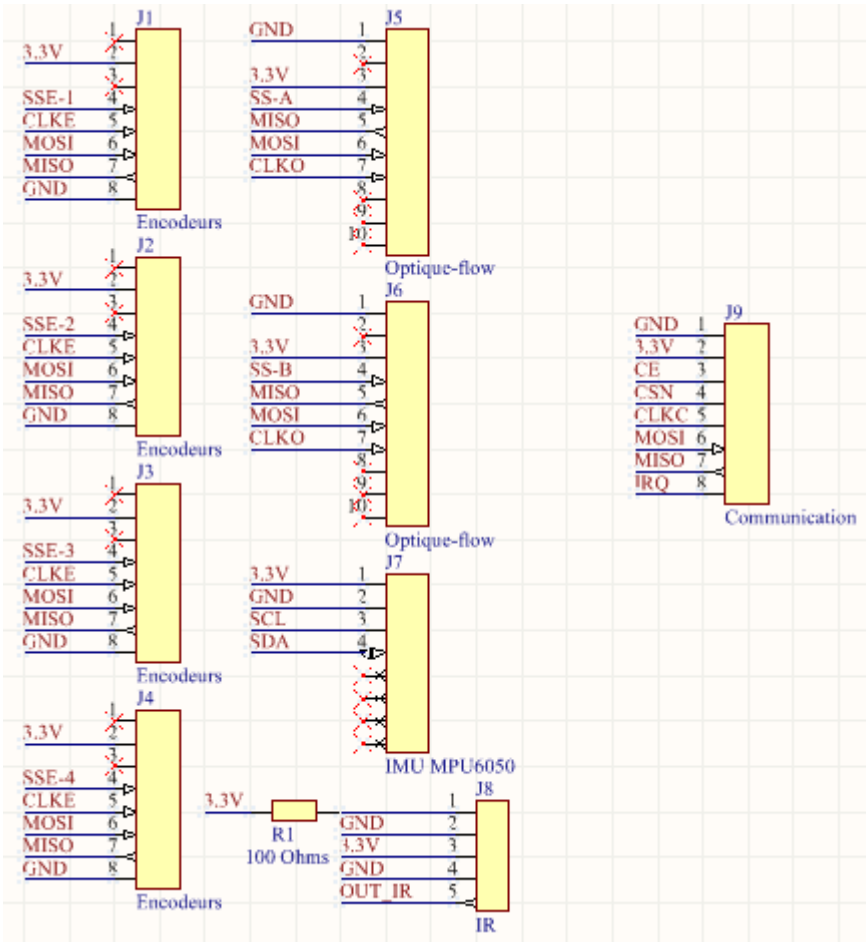


Figure 5 : Schématique Altium capteurs/communication

Cette schématique permet de mettre en évidence les différents branchements des capteurs ainsi que de la partie communication.
On remarque donc que le protocole de communication majoritaire est la communication SPI (Serial Peripheral Interface). De plus, l'alimentation de toutes les parties se fait en 3.3V. Ce qui facilite au maximum le travail de Valentin dans sa partie alimentation puisque je ne lui impose pas une multitude de tension.
Néanmoins, pour la fabrication, c'est Valentin qui se chargera d'implémenter mes connecteurs sur la carte mère de sorte à pouvoir relier les capteurs et le module radio à la carte mère.

2. Analyse structurelle :

Dans le but de bien comprendre comment fonctionne la partie capteur et la partie communication, il est important de les expliquer en détails.
C'est le rôle de cette partie :

a. Encodeurs :

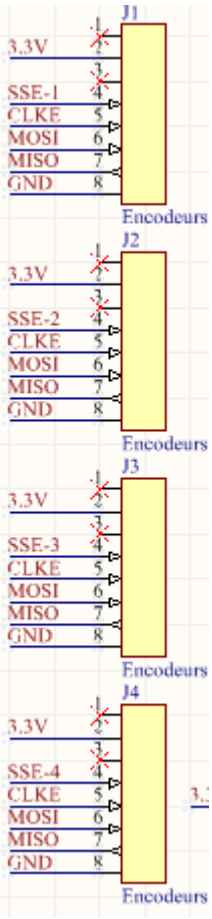


Figure 6 : Schéma des encodeurs

Les encodeurs ou capteurs à effets Hall sont les capteurs fixer à la partie roue plus moteur et qui permettent de mesurer avec certitude la vitesse de rotation du moteur. Or comme la roue est fixé directement sur le moteur, la vitesse de la roue correspond à la vitesse du moteur.

A l'aide du schéma, on remarque que le capteur supporte la communication SPI et est alimenté en 3.3V.

Le bus de communication SPI se définit par :

- SS : Slave Select, qui permet de sélectionner l'esclave
- CLK : Clock, cette broche permet d'apporter à l'esclave la clock du microcontrôleur (communication synchrone)
- MOSI : Master Out Slave In, broche ou l'esclave reçoit les données venant du microcontrôleur
- MISO : Master In Slave Out, broche avec laquelle l'esclave envoi la donnée au microcontrôleur.

b. Capteur optique :

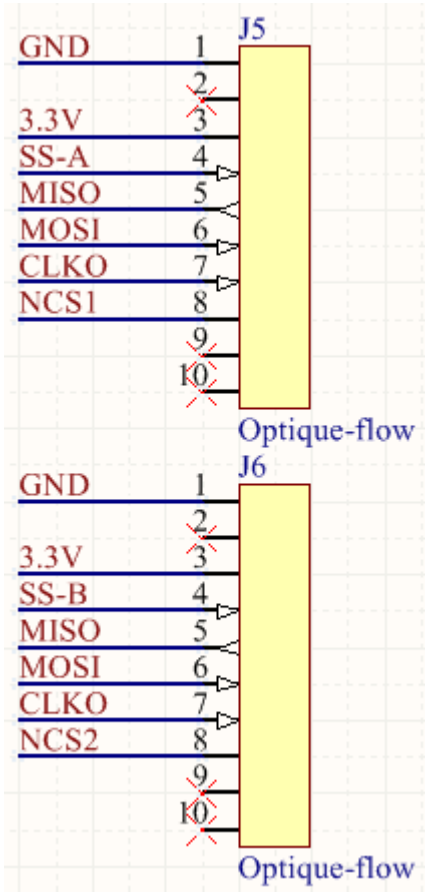


Figure 7 : Schéma capteur optique souris

Ce schéma met en évidence comment connecter les capteurs de souris à la carte microcontrôleur.
On remarque qu'il fonctionne grâce à une communication SPI (Serial Peripheral Interface) et est alimenté la aussi en 3.3V.

- SS : Slave Select, permet de définir avec quel esclave le microcontrôleur communique
- MISO : Master In Slave Out, broche de données allant de l'esclave vers le maitre.
- MOSI : Master Out Slave In, broche de données allant du maitre à l'esclave
- CLK : Clock, permet de définir la clock et donc la vitesse de transmission (synchrone)
- NCS : Chip Select : permet d'activer ou de désactiver le capteur (actif entrée à l'état bas).

c. Centrale Inertielle :

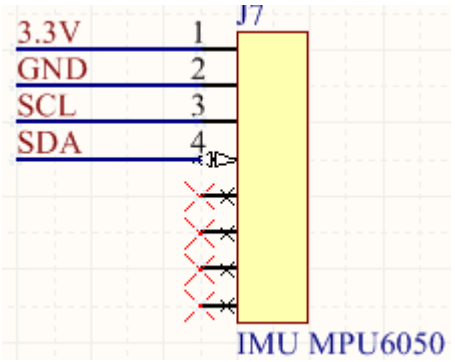
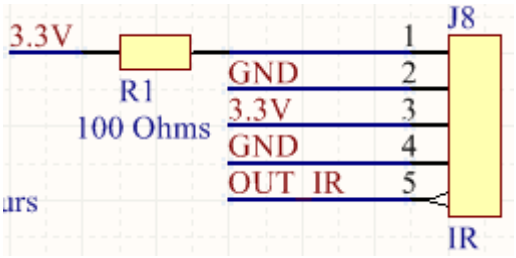


Figure 8 : Schéma de l'IMU

Ce schéma représente la central inertielle (IMU), la communication utilisée par ce capteur est une communication I^2C (Inter Integrated Circuit) :

- SCL: Serial Clock Line, broche de synchronization d'horloge bidirectionnel
- SDA : Serial Data Line, broche contenant la donnée transmise

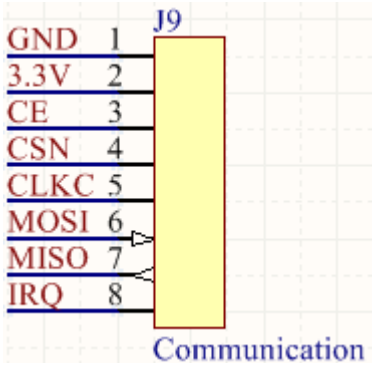
d. Barrière infrarouge :



Ce schéma explique le fonctionnement de la barrière infrarouge,
Les broches 1 et 2 alimente une LED infrarouge
Les broches 3,4 et 5 correspondent au récepteur infrarouge

- OUT_IR : sortie du récepteur infrarouge, état haut ou état bas en fonction de si la balle est devant le dribleur ou non.

e. nRF24L01 :



La communication se fait grâce au module nRF24L01, qui fournie une communication à 2.4GHz, avec un protocole de communication SPI (Serial Peripheral Interface) :

- CE : Chip Enable, permet d'activer ou de désactiver le module
- CSN : Chip Select, permet de choisir l'esclave avec qui communiquer
- CLK : Clock, synchronise la clock du maitre avec l'esclave pour une communication synchrone
- MOSI : Master Out Slave In, broche de données reçu par l'esclave
- MISO : Master In Slave Out, broche de données envoyé au maitre
- IRQ : Interrupt, broche d'interruption

VI. Gestion du déplacement :

Le robot est holonome, il est donc capable de se déplacer sur tous les axes d'un plan sans avoir à modifier l'orientation de ses roues.
Comme nous cherchons à réduire les prix au maximum, nous allons réutiliser la structure de basse du robot.
Nous connaissons donc les angles de chaque roue par rapport au point milieu.
Ainsi, on a :

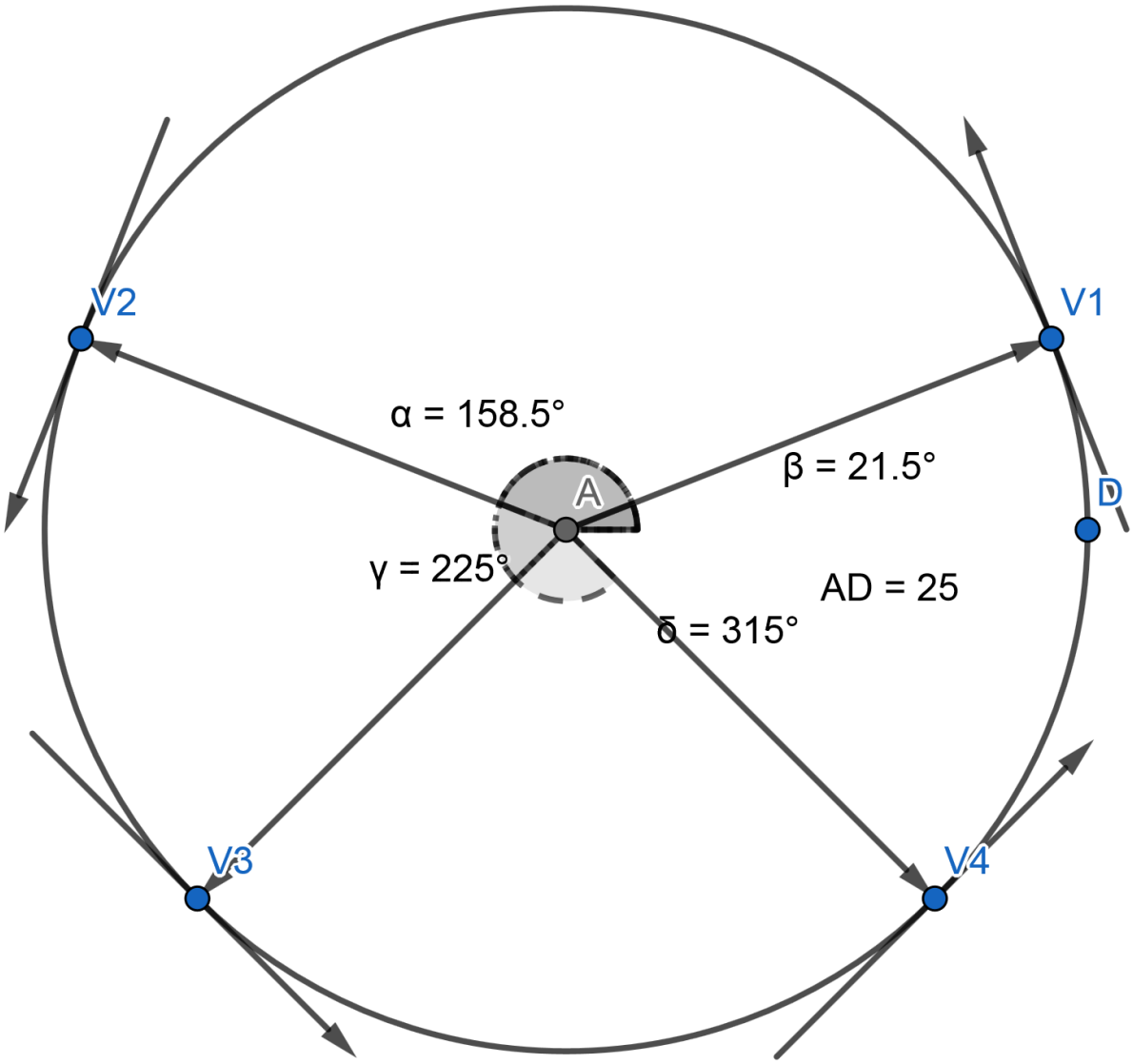


Figure 9 : Représentation schématique des vecteurs déplacements

Grâce à cette figure, on comprend mieux le fonctionnement du robot ainsi que son déplacement. Chaque groupe moteur et roue est représenté par un vecteur.

Afin de gérer au mieux le déplacement du robot, il est nécessaire de bien comprendre comment fonctionne le robot.

Le robot a donc 4 roues holonomes, Néanmoins, du fait que l'on ait récupéré la structure de base du robot, toutes les roues ne sont pas situées à $k.\frac{\pi}{2}$. Seules les deux roues arrière du robot sont bien situées à $\frac{5}{2}.\frac{\pi}{2}$ et à $\frac{7}{2}.\frac{\pi}{2}$.Les deux roues avant sont quant à elle à $21.5°$ et $158.5°$. Cet angle est du au fait que nos prédécesseurs avaient prévu beaucoup de place pour la partie dribleur et kickeur.

1. Roue V1 :

a. Calcul de $\overline{V1}$:

Le vecteur $\overline{V1}$ est donc situé à un angle de $21.5°$ par rapport à l'axe AD, qui représente ici l'axe des x.
On retrouve donc la relation :

$$x\overline{V1} = \cos(21.5 + 90)$$

et

$$y\overline{V1} = \cos(21.5 + 90)$$

Ainsi, on obtient :

$$x\overline{V1} = -0.366$$

Et

$$y\overline{V1} = 0.930$$

b. Vitesse et déplacement de $\overline{V1}$:

On sait que le vecteur $\overline{V1}$ est représenté par l'association de deux vecteurs grâce à la relation de Chasles. Le vecteur x$\overline{V1}$ et le vecteur y$\overline{V1}$. Cette addition donne donc la valeur de déplacement en x et en y de la roue 1.

$$V1 = x\overline{V1} + y\overline{V1}$$

Par ailleurs, on sait que la roue fait 0.06m, et que la roue est directement connectée au moteur. De plus, on estime que la roue doit permettre d'atteindre une vitesse de 3m/s.
On obtient donc une vitesse de la roue 1 à :

$$rotation\ du\ moteur\ 1 = \frac{3.V1.60}{2.\pi.0.06}$$

Avec :
3 : vitesse du robot (exprimée en m/s)
V1 : déplacement en x et y du vecteur de la roue1
60 : conversion de seconde à minute
2.π.0.06 : périmètre de la roue

2. Roue V2 :

a. Calcul de $\overline{V2}$:

Le vecteur $\overline{V2}$ est lui situé à $158.5°$.

On a donc la relation :

$$x\overline{V2} = \cos(158.5 + 90)$$

Et

$$y\overline{V2} = \sin(158.5 + 90)$$

Donc

$$x\overline{V2} = -0.366$$

Et

$$y\overline{V2} = -0.93$$

b. Vitesse et déplacement de V2 :

Avec la même relation que pour la roue1, on obtient :

$$V2 = x\overline{V2} + y\overline{V2}$$

Ainsi, la vitesse de rotation du moteur 2 est de :

$$rotation\ du\ moteur\ 2 = \frac{3.V2.60}{2.\pi.0.06}$$

3. Roue V3 :

a. Calcul de $\overline{V3}$:

Le vecteur $\overline{V3}$ est lui situé à $225°$.

On a donc la relation :

$$x\overline{V3} = \cos(225 + 90)$$

Et

$$y\overline{V3} = \sin(225 + 90)$$

Donc

$$x\overline{V3} = 0.707$$

Et

$$y\overline{V3} = -0.707$$

b. Vitesse et déplacement de V3 :

Avec la même relation que pour la roue1, on obtient :

$$V3 = x\overline{V3} + y\overline{V3}$$

Ainsi, la vitesse de rotation du moteur 3 est de :

$$rotation\ du\ moteur\ 3 = \frac{3.V3.60}{2.\pi.0.06}$$

4. Roue V4 :

a. Calcul de $\overline{V4}$:

Le vecteur $\overline{V4}$ est lui situé à $315°$.

On a donc la relation :

$$x\overline{V4} = \cos(315 + 90)$$

Et

$$y\overline{V4} = \sin(315 + 90)$$

Donc

$$x\overline{V4} = 0.707$$

Et

$$y\overline{V4} = 0.707$$

b. Vitesse et déplacement de V4 :

Avec la même relation que pour la roue1, on obtient :

$$V4 = x\overline{V4} + y\overline{V4}$$

Ainsi, la vitesse de rotation du moteur 4 est de :

$$rotation\ du\ moteur\ 4 = \frac{3.V4.60}{2.\pi.0.06}$$

5. Exemple de déplacement :

Ainsi, si je cherche à réaliser un déplacement vers l'avant pour que le robot aille tout droit.

Pour aller tout droit, je dois jouer avec le sens des vecteurs seulement puisque leur direction ne change pas. Donc je me retrouve avec le vecteur $\overline{V1}$ positif, le vecteur $\overline{V2}$ négatif, le vecteur $\overline{V3}$ négatif et le vecteur $\overline{V4}$ positif.

La représentation vectorielle est donc la suivante :

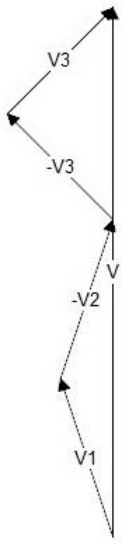


Figure 10 : Représentation vectorielle du déplacement

Lorsque l'on cherche à aller tout droit, il est important de jouer avec le sens des vecteurs. Sachant que $\overline{V1}$ et $\overline{V2}$ sont symétrique par rapport à l'axe des Y, afin d'aller tout droit il est nécessaire que l'un soit l'opposé de l'autre. De même pour les vecteurs $\overline{V3}$ et $\overline{V4}$.

Comme on cherche à aller tout droit, on ne se préoccupe que de l'axe des Y.
Il me suffit donc de réaliser le calcul suivant :

- Moteur1 : $x\overline{V1} = 0$ et $y\overline{V1} = 0.93$ ➔ V1$= x\overline{V1} + y\overline{V1} = 0.93$
- Moteur2 : $x\overline{V2} = 0$ et $y\overline{V2} = -0.93$ ➔ -V2$= -(x\overline{V2} + y\overline{V2}) = 0.93$
- Moteur3 : $x\overline{V3} = 0$ et $y\overline{V3} = -0.707$ ➔ -V3$=-(\ x\overline{V3} + y\overline{V3}) = 0.707$
- Moteur4 : $x\overline{V4} = 0$ et $y\overline{V4} = 0.707$ ➔ V4$= x\overline{V4} + y\overline{V4} = 0.707$

Ainsi, ce vecteur direction nous permet de définir le coefficient que doit prendre chaque moteur de sorte que chaque roue tourne dans le bon sens et à la bonne vitesse.

- Moteur1 : $\frac{3.V1.60}{2.\pi.0.06} = 444$ tour/min
- Moteur2 : $\frac{3.V2.60}{2.\pi.0.06} = -444$ tour/min
- Moteur3 : $\frac{3.V3.60}{2.\pi.0.06} = -337.5$ tour/min
- Moteur4 : $\frac{3.V4.60}{2.\pi.0.06} = 337.5$ tour/min

Grâce à ces calculs, il est donc possible de contrôler chaque roue de sorte que le robot aille dans la direction voulue. Il suffit juste de jouer sur le sens et la vitesse de rotation.

Je calcule chaque indépendamment l'action de chaque roue, pour obtenir un x global pour le robot et un y global pour le robot.

6. Calcul du PID :

Pour pouvoir assurer le contrôle du robot, il est nécessaire de mettre en place un PID (Proportionnel Intégrale Dérivé). Ce système traité logiciellement sert à asservir au mieux les moteurs de sorte que chaque roue tourne à la vitesse souhaitée.
Afin de réaliser cet asservissement, on va utiliser les capteurs à effets Hall (encodeurs) qui vont nous donner avec certitude la valeur de la vitesse de rotation des roues.

a. Proportionnel :

L'aspect de proportionnalité dans le PID sert à réaliser une correction instantanée de l'erreur perçue entre la valeur de déplacement donnée par l'ordinateur (commande) et la valeur mesurée à l'aide des capteurs.
On va nommer cet aspect Kp.
K pour le fait que ce soit un coefficient de proportionnalité et p pour se souvenir qu'il s'agit bien de la proportionnelle.

$$commande\_P = Kp.erreur$$

Avec :
Kp : coefficient de proportionnalité à régler
Erreur : différence entre la commande et la mesure (commande-mesure)

Néanmoins, il est aussi important de prendre en compte le fait que le coefficient de proportionnalité n'est pas suffisant. En effet, si l'erreur est trop grande, alors le système devient instable en appliquant des corrections non réalisables.
Ainsi, on ajoute d'autre composante

b. Intégrale :

Pour améliorer la précision et la qualité du système, on ajoute un autre aspect à notre asservissement, l'intégrale.
En effet, si on utilise seulement la proportionnelle, on ne pourra pas rester à la vitesse voulue. Il est donc nécessaire de rajouter cet aspect qui va prendre en compte la proportionnelle et l'ajouter à la somme des erreurs.
Ainsi on a :

$$commande\_I = Kp.erreur + Ki.somme\_erreur$$

Avec :
Kp : coefficient de proportionnalité de la proportionnelle
Ki : coefficient de proportionnalité de l'intégrale
Somme_erreur : somme des erreurs commise au cours du temps

c. Dérivée :

Afin, pour limiter au maximum les erreurs et les délais. Ainsi, que pour rendre le système le plus stable possible, il est indispensable de rajouter un aspect de dérivée, qui va ajouter à la proportionnelle un coefficient de dérivée multiplié par erreur actuel moins l'erreur à t-1.
Ainsi, on a :

$$commande\_D = Kp.erreur + Kd.(erreur - erreur_{précédente})$$

Avec :
Kp : coefficient de proportionnalité de la proportionnelle
Kd : coefficient de proportionnalité de la dérivée
$Erreur_{précédente}$: erreur à t-1

d. Représentation du PID :

Pour le moment, il me reste à réaliser le schéma du PID, le coder en C sur ARM, tester les capteurs et la communication et réaliser le code.
Par la suite après les tests sur capteurs, je pourrai être sûr de mon schéma altium.