**Développement du logiciel de commande**

Dans cette section, le travail repose en grande partie sur la base d’un code donné par nos prédécesseurs qui avait pour ambition de gérer la commande du robot.

Nous avions, dès lors, plusieurs fichiers à notre disposition dans le dossier qui nous avait été communiqué :

* Un dossier Arduino pour contrôler les moteurs
* Un fichier python codant le correcteur PID
* Un pour traiter l’image
* Un pour faire fonctionner le robot manuellement ou de façon autonome : il lance le robot

Pour rappel, la fonction finale qui doit être remplie par le robot consiste à être capable de suivre une personne. Au regard de l’état d’avancement du projet, nous restreignons nos ambitions en limitant notre objectif à un suivi de ligne blanche pré tracée.

En effet, le rapport de nos prédécesseurs indiquait que le protocole mis en place répondait déjà à ce problème. Mais après des essais et des vérifications nous en sommes arrivés à la conclusion qu’il n’en était rien : la commande manuelle marche bien (mise à part la partie mécanique qui ne suivait pas : les roues glissaient trop, le robot avait du mal à tourner) mais la commande autonome ne marche pas du tout : que la ligne aille vers la gauche ou vers la droite, le robot tourne dans les deux cas à droite, et la pente de la ligne n’est pas du tout prise en compte. Cependant, les calculs dans le rapport de nos prédécesseurs sont bons, nous nous en sommes donc servis pour refaire entièrement le code de cette partie autonome. Cependant, cette partie étant conséquente et ayant des contraintes plus importantes pour le projet, nous avons assez vite arrêter de m’y aventurer, d’autant plus qu’une fois les formules réécrites, le robot n’était pas encore tout à fait opérationnel car il restait les coefficients du correcteur à ajuster. Actuellement, le robot tourne donc dans le bon sens, prend plus ou moins la pente de la ligne en compte (va moins vite si la ligne est en angle droit avec le robot et proche de celui-ci), mais n’est pas encore capable de la suivre.

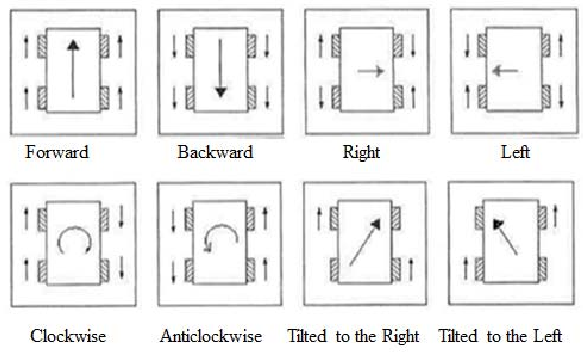
Ensuite, après avoir installé le nouveau châssis, les nouveaux moteurs et les nouvelles roues, nous sommes passés à une nouvelle partie : l’exploitation du plein potentiel de ces roues. En effet, avant, les roues étaient commandées par paires : les roues droites étaient commandées ensembles avec la même consigne, et les roues gauches entre elles aussi. Cela permettait au robot d’avancer, reculer, et tourner tout en faisant ces mouvements (il devait aussi tourner sur lui-même, mais il est facile de comprendre que ce n’est pas possible avec des roues « normales » en parallèle).

Avec les roues *mécanum*, le robot peut maintenant se déplacer sur les côtés, tourner sur lui-même, et se déplacer en diagonale… à condition que les 4 roues soient commandées indépendamment. C’est cette partie sur laquelle nous avons beaucoup travaillé par la suite. En effet, avant, le circuit imprimé permettait de commander les deux côtés indépendamment, car des courts-circuits étaient placés entre les roues avants et arrières pour les commander ensembles. En enlevant ces courts-circuits, il n’y a alors plus que les roues avant qui sont commandées par le L298N du circuit-imprimé (composant permettant de délivrer une tension à deux moteurs, via modulation de largeur de pulsation, et de changer leur sens de rotation). Heureusement la carte Arduino du robot (la Roméo BLE) possède aussi un L298N intégré, permettant alors de commander les 4 moteurs indépendamment. Il a juste fallut encore enlever quelques courts-circuits sur l’Arduino, puis comprendre comment modifier le code et quels branchements faire pour pouvoir l’utiliser. La partie « quels branchements faire » a pris 4 semaines, puis nous sommes passés au code. Après avoir fait quelques tests du code, qui marchait comme prévu, nous l’avons encore modifié pour qu’il corresponde à une approche plus intuitive et rapide que nous n’avions pas prévu au départ.

A picture containing text, different, orange, same

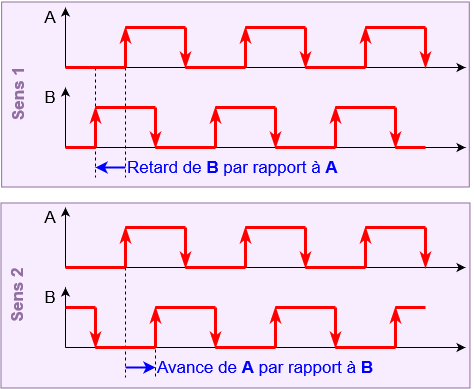
Description automatically generated

*Figure i+1 : Vue de dessous des roues : les rouleaux permettent d’avoir des résultantes uniquement selon les directions des flèches oranges. Il est alors très facile de comprendre comment les mouvements suivants sont possibles :*



*Figure i+2 : mouvements possibles grâce aux roues mécanum (vue de dessus)*

Diagram

Description automatically generated with medium confidence

*Figures i+3 : Principe des encodeurs double voie*

Directement après, nous avons voulu utiliser les codeurs des moteurs, pour vérifier précisément les caractéristiques des moteurs notamment. Deux semaines ont été nécessaires à la compréhension de la structure du code à adopter. Nous avons alors compris que nous ne pourrions pas prendre en compte les codeurs des 4 moteurs : déjà, il n’y a pas assez de pins restants sur la carte Arduino ; ensuite, même parmi ces pins, tous ne sont pas utilisables pour les codeurs, qui nécessitent des pins « interrupt ». Ces pins spéciaux permettent d’appeler une fonction spéciale dès que l’état du pin change. Cette fonction permets de calculer l’intervalle de temps qu’il a fallu à la roue pour passer d’un angle à un autre, et donc de savoir à quelle vitesse va la roue, et donc le robot.

Même après avoir compris cela, nous n’arrivions pas à faire marcher le code pour les encodeurs, nous pensions même que les codeurs ne marchaient pas. Alors à l’oscilloscope, nous avons vérifié que les codeurs marchaient bien. Dans la même après-midi, nous avons réussi à faire marcher le code pour les encodeurs, et nous avons alors déterminés la vitesse du robot : 80cm/s. Cela n’est pas encore l’objectif, mais nous avons multiplié par 5 la vitesse d’origine, et l’objectif peut être atteint en mettant une source de tension de 9V (au lieu des 6V actuels) bien que les moteurs sont faits pour fonctionner à 6V.