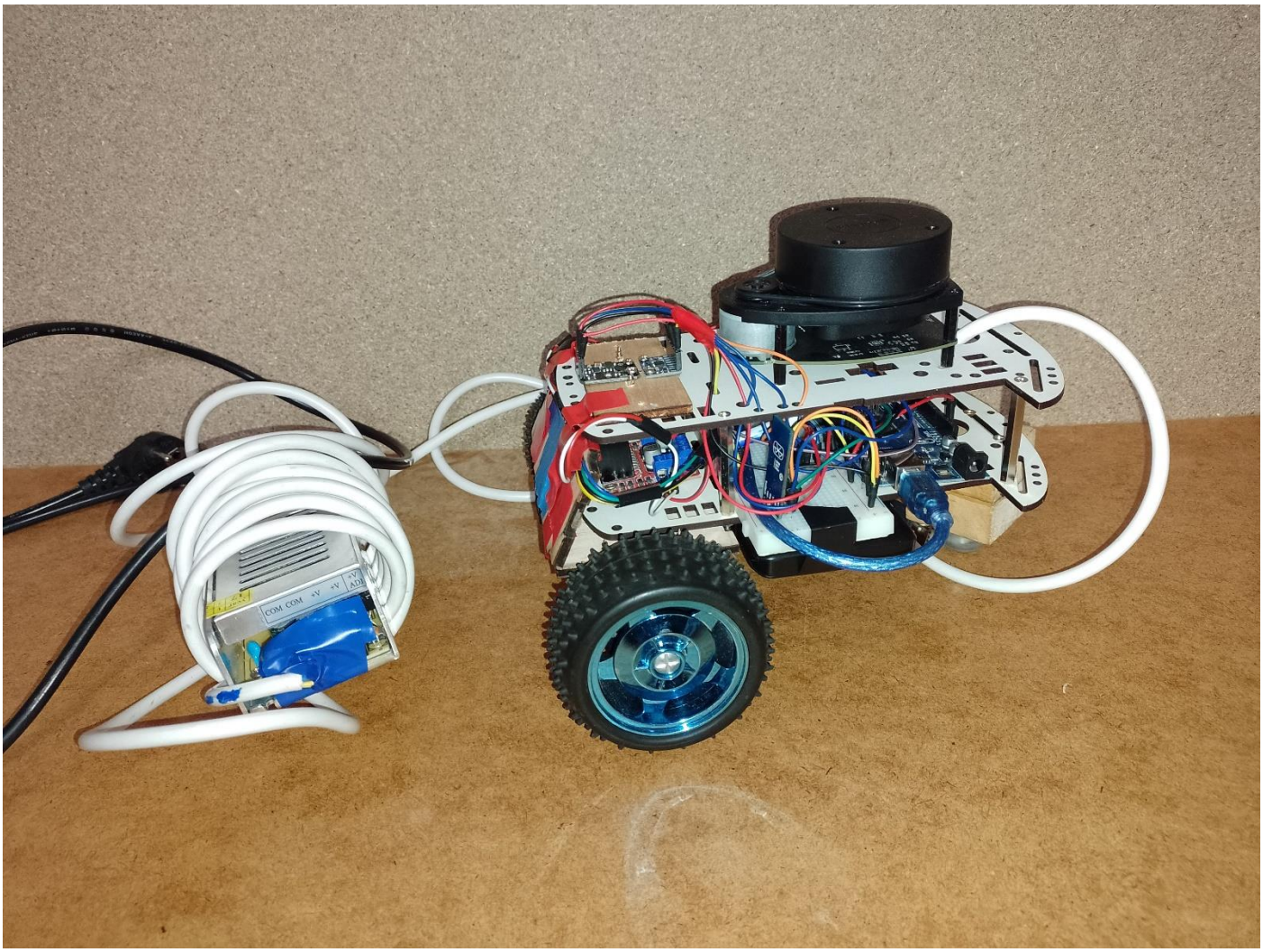
Vincent Callegari G1

Projet ARDUINO Transporteur autonome :

Rapport Final de projet



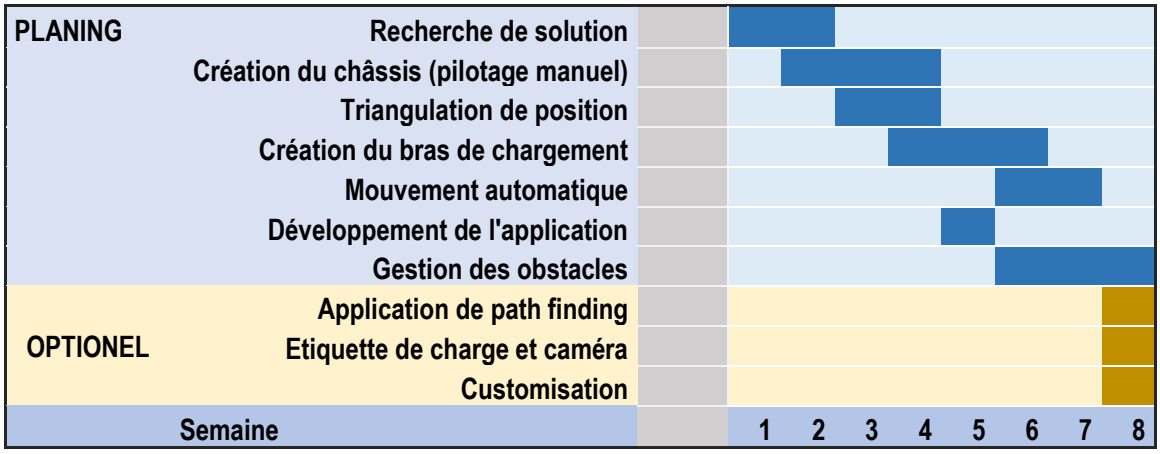
Sommaire :

1.Objectif :

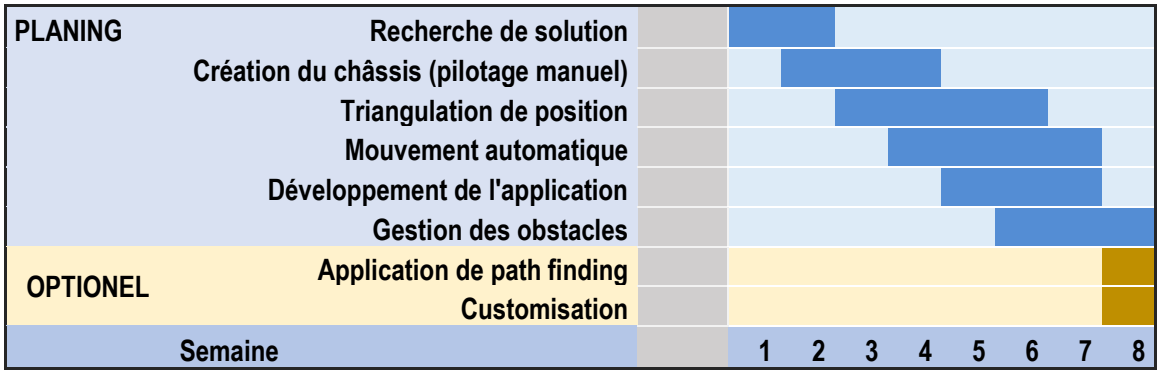
L'objectif du projet était de mettre au point un véhicule capable de se déplacer de manière autonome dans un espace 2D (véhicule terrestre).

Le robot devait également transporter une charge mais l'idée a été laissée de côté avant le début du projet car les contraintes de temps était trop courte.

Voici le planning tel qu'il avait été prévu au départ.



Bien sûr, toutes les tâches en rapport avec les charges ont été mises de côté et le planning prévu ressemblait donc plus à ça :



Le projet devait avoir des modules permettant de calculer la position du robot, des modules pour détecter les obstacles, un moyen de locomotion (roues, chenille …) et d'un module Bluetooth.

Finalement les différents modules utilisés par le projet ont été les suivants :

- Lidar (détection des obstacles)
- Magnétomètre (calculer la position)
- Accéléromètre (calculer la position)
- Module HC-06 (Bluetooth)
- Moteur 12V avec encodeurs (Moyen de Locomotion)

2.Structure du projet :

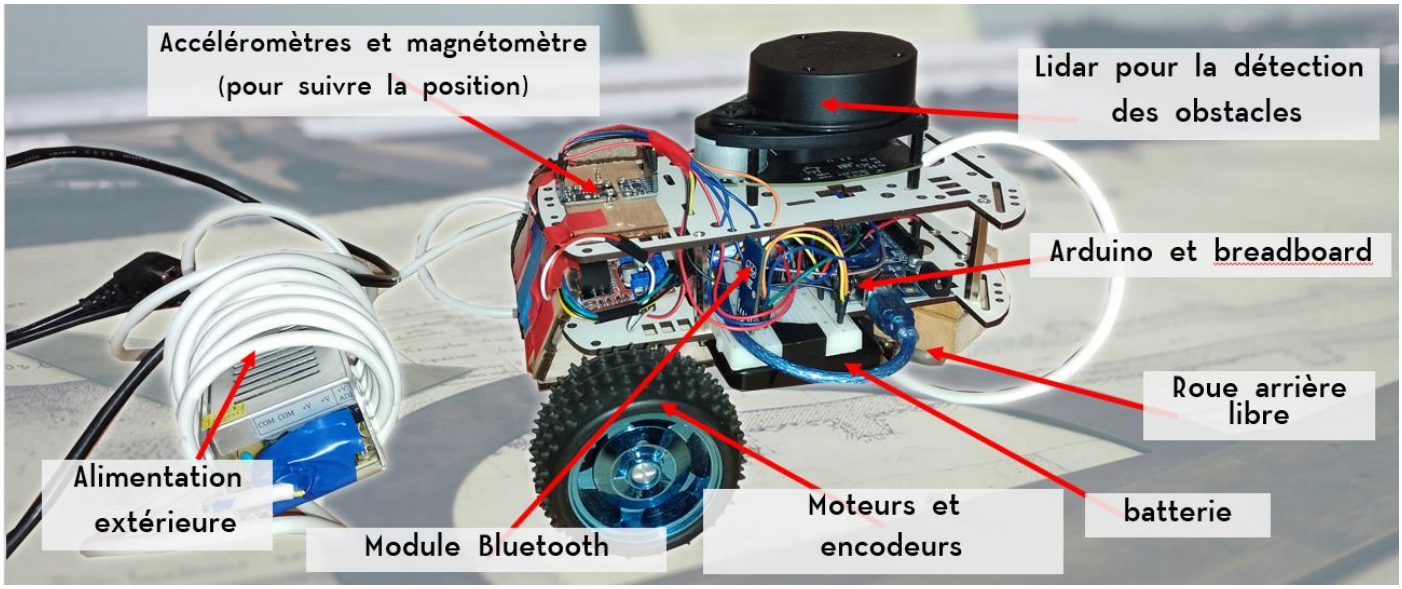
Le châssis consiste en deux plaques en bois découpé au laser et relié entre elles. Le lidar est placé sur le dessus pour ne pas être gêné par les roues les autres composants. J'ai aussi placé les capteurs de position (magnétomètre et accéléromètre) sur le dessus car les capteurs sont assez petits pour ne pas interrompre le laser du lidar.

J'ai placé en dessous la carte Arduino et la Breadboard. Il y a également le contrôleur du moteur.

Les moteurs et encodeurs sont fié sous le châssis.

Pour les alimentations j'ai utilisé un bloc d'alim externe de 12V avec une rallonge pour les moteurs car le prototype était trop petit pour contenir une batterie de 12V.

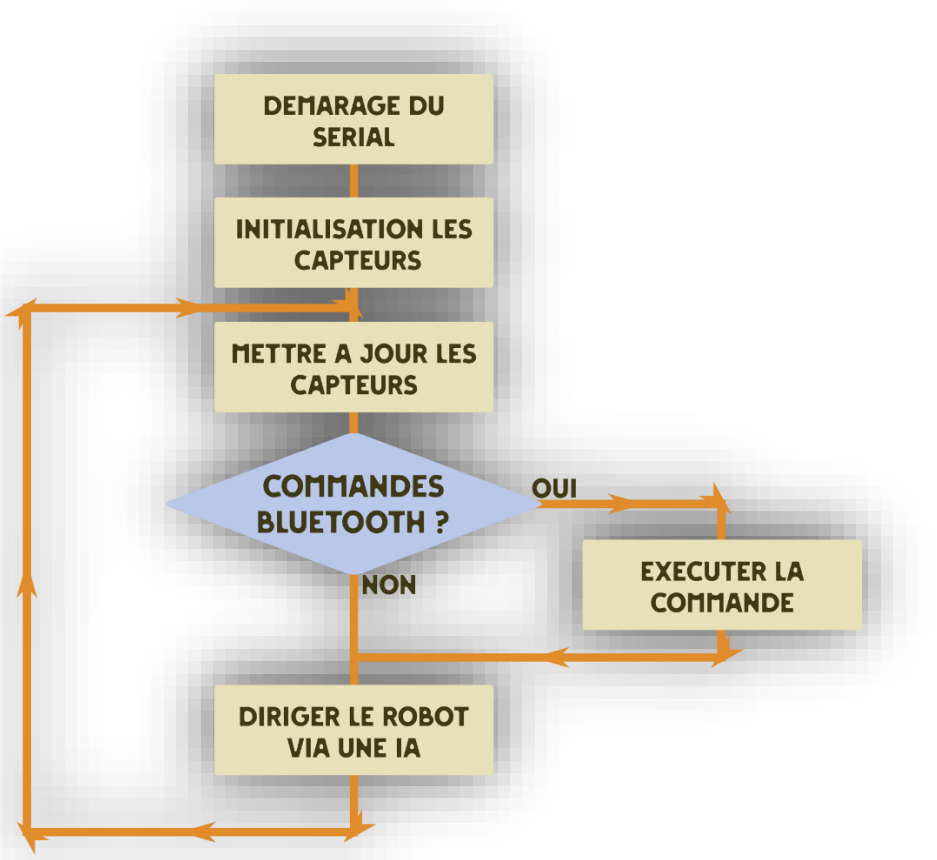
Pour l'Arduino j'ai utilisé une batterie portable de smartphone mais elle s'est avérée trop faible pour alimenter le lidar (branché sur l'Arduino). J'ai donc remplacé la batterie par une autre plus puissante.

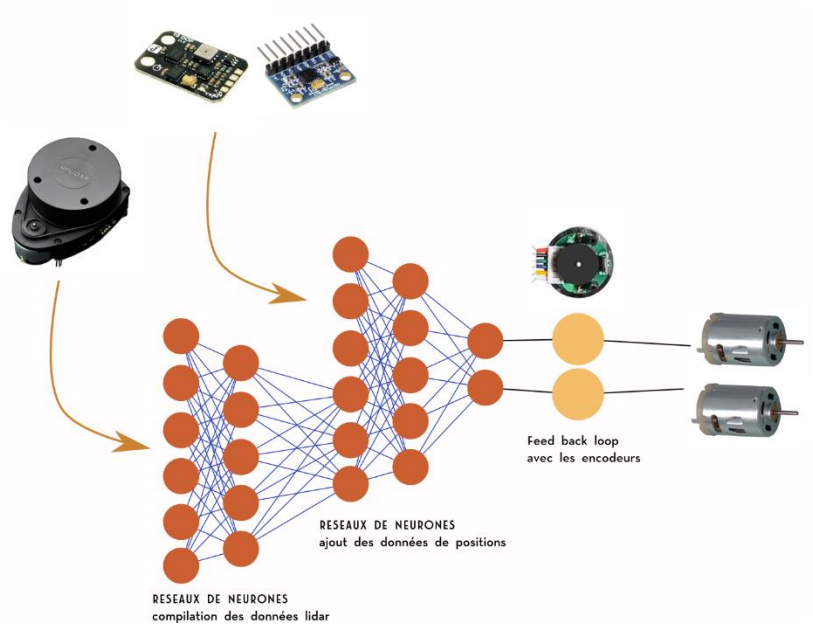


3.Algorithmes :

Algorithme initial :

Le programme utilise une intelligence artificielle pour déterminer quelles vitesses donner aux moteurs. L'intelligence artificielle consiste en un réseau de neurones optimisé par algorithme génétique.

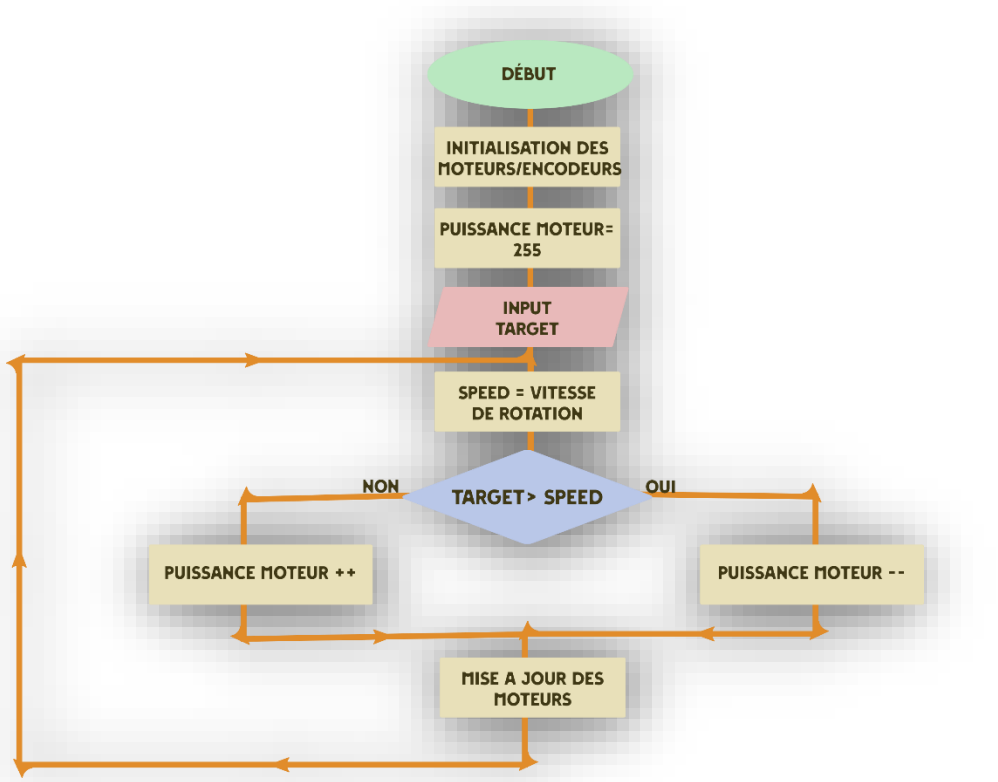




FEEDBACK LOOP DES ENCODEURS :

Voici l'algorithme des Feedback Loop mise en place pour les encodeurs, l'algorithme ne montre la Feedback Loop que d'un seul des deux encodeurs pour simplifier le schéma.

La cible sera définie par d'autres morceaux du programme.



CALCUL DE LA POSITION :

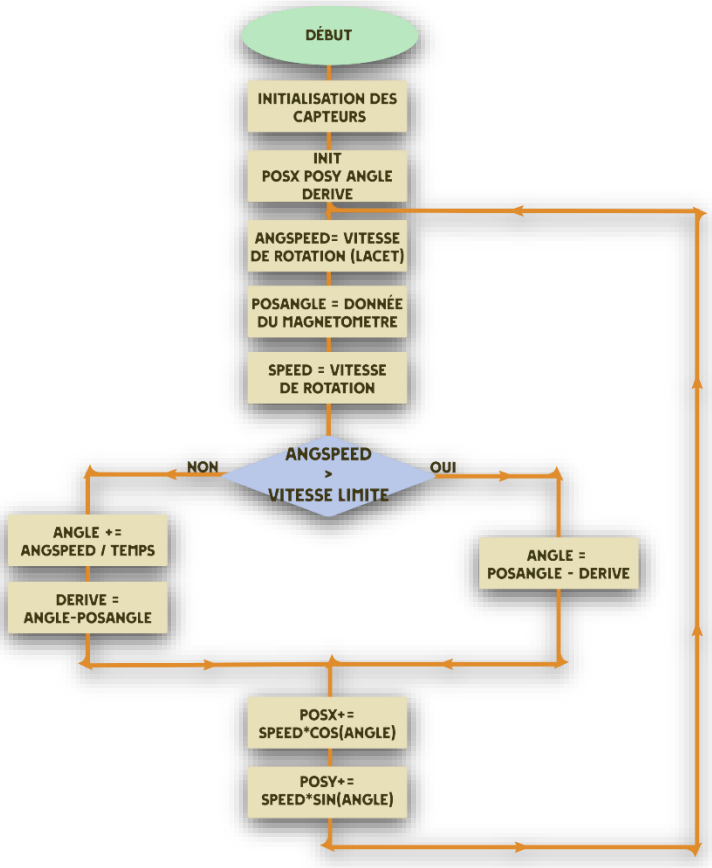
L'algorithme du système de positionnement à été plus difficile à mettre en place que prévu car les capteurs se sont révélés être moins précis. Les angles récupérés par le magnétomètre dérivaient avec le temps (le rendant inutilisable à long terme) et l'accéléromètre faisait des mesures imprécises avec des grandes vitesses.

J'ai choisi un système en base de Frenet pour calculer les positions de la voiture pour ça il faut mesurer la vitesse du véhicule et son orientation le long de l'axe vertical.

Pour les vitesses j'ai choisi d'utiliser les encodeurs aux lieux de l'accéléromètre car l'accélération de la gravité influençaient les valeurs selon les légères pentes du terrain et il aurait fallu faire encore plus de mesure d'angle pour calculer les vraies accélérations.

Pour la mesure de l'angle j'ai cumulé les deux capteurs en utilisant l'accéléromètre à faible vitesse angulaire (donc la majeure partie du temps) puis le magnétomètre à forte vitesse angulaire (donc pendant de courte période).

Voici l'algorithme final pour le calcul des positions :



UTILISATION DU LIDAR :

J'ai rencontré de nombreux problèmes avec le lidar, notamment des problèmes de communications qui ont été réglé en utilisant une alimentation plus forte, cependant même la batterie utilisée dans le prototype final est un peu trop faible pour l'alimenter normalement (ou peut être que la carte Arduino limite le courant d'une manière ou d'une autre).

En dehors de ces problèmes j'ai réussi à faire fonctionner le lidar correctement et l'algorithme du lidar consiste en un algorithme très simple qui récupère ses données en boucles.

Dans le cas où les données du lidar serait trop nombreuse j'avait pensé à un système d'alternance en prenant les données des angles pairs puis impairs Mais je n'ai as eu le temps de mettre en place ce système.

Algorithme final :

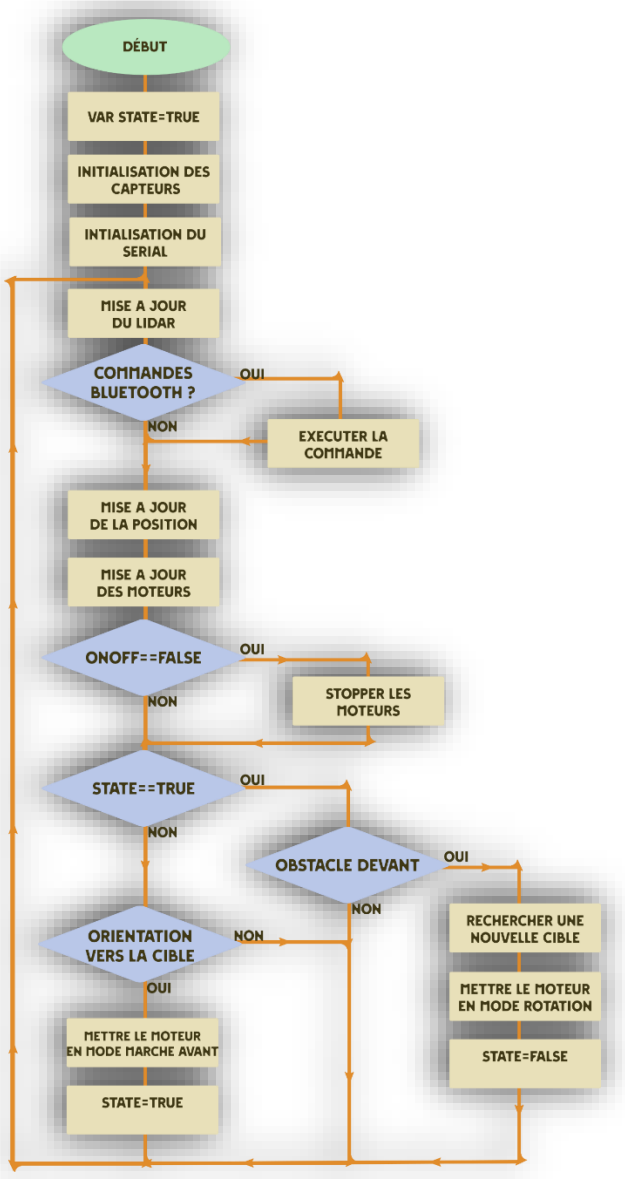
Ayant eu des problèmes avec le programme de l'intelligence artificielle (voir partie 3). J'ai dû faire un programme plus simple :

Le programme oscille entre deux phases :

- Une phase de recherche dans laquelle le robot s'oriente dans la direction avec le plus le plus lointain ;
- Et une phase de « route » durant laquelle le robot se dirige en ligne droite jusqu'au prochain obstacle

Il y a eu quelque problème de code que je n'ai pas eu le temps de résoudre avant la fin du projet, le problème principal étant le fait que le lidar à besoin d'une grande puissance de calcul ce qui réduit la fréquence des boucles ce qui pose un problème avec les Feedback Loop des encodeurs.

Vous pouvez voir le résultat de l'algorithme final dans cette vidéo de démonstration : https://youtu.be/Rj1Y-65wC_8



3. Logiciel de simulation :

Le Logiciel a été programmé en JAVA avec la librairie Processing pour simplifier la conception de l'interface.

J'ai créé mes propres librairies de réseaux de neurones et d'algorithme génétique.

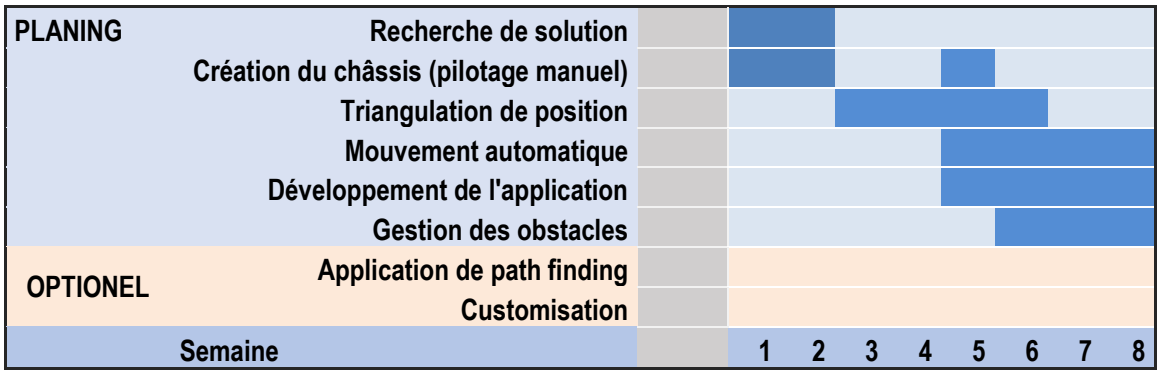
Le programme consiste en un environnement contenant des murs (lignes) et des versions virtuelles des voitures simulant leur mouvement et leur LIDAR (raycast).

Les versions virtuelles de la voiture sont lâchées au début d'un simple parcours, l'objectif des voitures est d'aller le plus loin possible dans le circuit. Les résultats ont montré que l'algorithme fonctionnait, mais pas suffisamment pour être utilisé dans le projet.

Le principal problème était que l'intelligence artificielle n'était pas suffisamment polyvalente.

5. Conclusion :

La répartition finale des tâches :



Finalement le robot est capable d'éviter les obstacles et de connaître sa position. L'ensemble des capteurs sont opérationnels même si je n'ai pas eu le temps de faire un vrai « cerveau » pour le robot.

La prochaine étape serait de travailler pleinement sur la gestion précise des déplacements notamment en faisant converger le robot vers la cible voire enregistrer les données du lidar pour avoir une carte de l'environnement puis de calculer un chemin efficace par pathfinding.

6. Bibliographie :

//https://github.com/adafruit/Adafruit_Sensor
//https://github.com/adafruit/Adafruit_MPU6050
//https://github.com/robopeak/rplidar_arduino
//https://wiki.dfrobot.com/10_DOF_Sensor__SKU_SEN0140
//http://bucket.download.slamtec.com/e680b4e2d99c4349c019553820904f28c7e6ec32/LM108_SLAMTEC_rplidarkit_usermaunal_A1M8_v1.0_en.pdf
//https://www.aranacorp.com/fr/utilisation-dun-module-mpu6050-avec-arduino/
//https://automaticaddison.com/calculate-pulses-per-revolution-for-a-dc-motor-with-encoder/