



NXP CUP
INTELLIGENT
CAR RACING

ESIÉE

PARIS

une école de la



SOMMAIRE :

I. INTRODUCTION

- La NxP Cup, qu'est-ce que c'est ?
- Explication du projet

II. LE ROBOT

- Les détails techniques
- Les améliorations apportées
- Les problèmes rencontrés

III. LES CONNAISSANCES ACQUISES

- Principe de PID
- La communication PC-composant
- La caméra et le suivi de ligne

IV. CONCLUSION

- Résultat final
- Les améliorations possibles
- Retours personnels vis-à-vis du projet

I. INTRODUCTION

La NxPCup est une compétition automobile destinée à des équipes d'étudiants venus de plusieurs pays, dans laquelle des voitures autonomes montées et développées par ces étudiants s'affrontent.

Au début du projet nous avons reçu le pack de base fourni à tous les participants à la NxP Cup, ainsi que les programmes de test. Le projet peut se découper en plusieurs parties.

La première partie consiste à la phase de compréhension, tests et de familiarisation avec l'environnement de développement, le fonctionnement de la voiture et les codes fournis.

La transition vers la seconde partie s'est faite naturellement - il s'agissait d'amélioration et d'optimisation du code car à terme, la voiture devait se déplacer seule. En effet, la voiture doit concourir en autonomie et réaliser un tour de piste le plus rapidement possible sans sortir de piste. Celle-ci reconnaît les limites du circuit en analysant les images filmées par la caméra.

Enfin, la dernière partie est arrivée après la course. L'objectif principal de l'atelier étant passé, nous avons voulu aller au-delà des attentes et avons entamé des améliorations sur la voiture. Ces améliorations sont minimales mais néanmoins importantes. Nous avons procédé à l'ajout d'un système d'arrêt automatique de la voiture lorsque celle-ci se retrouve bloquée contre un objet ou mur, ou lors d'un choc. Un travail d'optimisation a également été réalisé sur le PID pour obtenir une meilleure trajectoire de la voiture. Enfin, un interrupteur a été ajouté et câblé afin de rendre les tests plus pratiques.

L'atelier avait pour objectif de consolider nos bases en programmation en C et approfondir nos connaissances dans ce langage, dans un environnement de développement peu familier. Egalement, cet atelier constituait une première approche des systèmes embarqués- nous avons appris à contrôler chaque composant de la voiture et à faire le lien entre le code et la caméra, les moteurs, le capteur de vitesse...

II. LE ROBOT

Pour ce projet nous disposons d'un pack de base fourni par la NxP cup. Ce pack contient un châssis composé d'un socle doté de quatre roues, de deux moteurs et un servomoteur. Il y a aussi une batterie, une caméra en noir et blanc et une carte de développement FRDM-KL25Z associé au TFC Shield.

Notre châssis est le model C de dimensions 28.5x16x8 cm.



Châssis, source : google image

Au niveau du châssis, les deux roues avant sont mobiles et reliées au servomoteur pour pouvoir diriger la voiture ensuite les deux roues arrière sont chacune branchée à un moteur.

Le servomoteur utilisé est un moteur rotatif qui permet de placer les roues dans des positions prédéterminées et de les maintenir dans la position choisie.

La batterie est une pile NiMh (**nickel-metal hydride**) rechargeable ayant une tension de 7.2 Volts pour une intensité inférieure ou égale à 3000 mA et étant connectée par des connecteurs Tamiya.

La caméra représente en fait les yeux de la voiture. C'est une caméra en noir et blanc à balayage linéaire avec une résolution de 1 x 128 pixels.



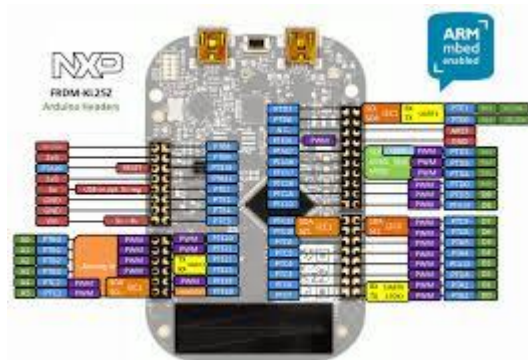
Male

Connecteur
Tamiya, source :
NxP

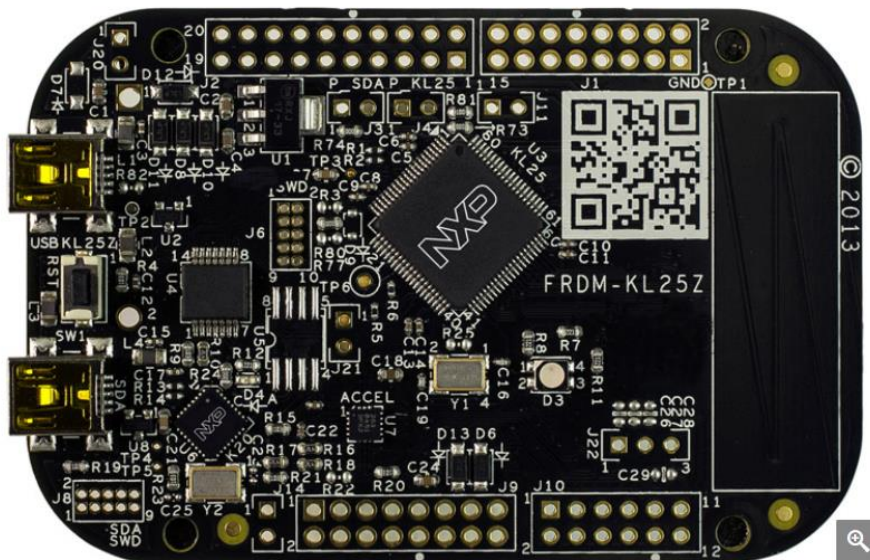
Enfin le programme permettant la navigation du robot est télé versé sur la carte FRDM-KL25Z. Il s'agit d'un microcontrôleur, basé sur le processeur ARM Cortex-M0, qui permet d'effectuer tous les calculs permettant la gestion des périphériques internes. Cette carte est complétée avec le TLC Shield qui, lui, gère les périphériques externes tels que la caméra.



TFC shield, source ; google image



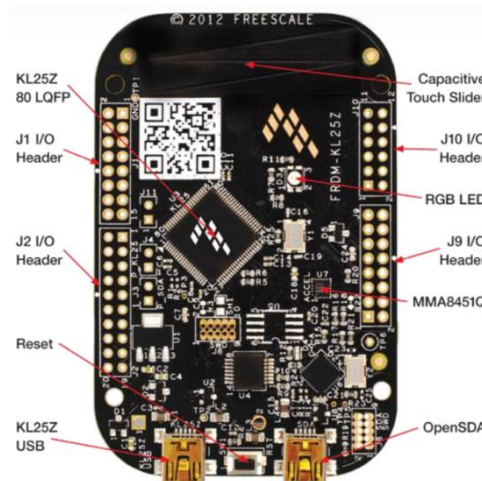
Carte FRDM-KL25Z. source : mbed.com



Source : nxp.com

La voiture doit évoluer d'elle-même le long d'un circuit. Le circuit se compose de lignes droites, carrefours, chicanes et virages. La piste est une bande blanche de 33.5 cm de large et délimités de chaque côté par des bandes noires.

Durant l'atelier nous avons dû faire face à plusieurs imprévus. Lors d'un test, alors qu'on télé versait le programme dans la voiture, celle-ci est partie d'un seul coup à toute allure alors qu'elle était encore branchée à l'ordinateur et le port USB, s'est arraché. Ce problème nous a fortement ralenti et nous avons donc dû changer de carte et pour prévoir à l'avenir ce type de problèmes, nous avons mis un interrupteur au niveau de la batterie.



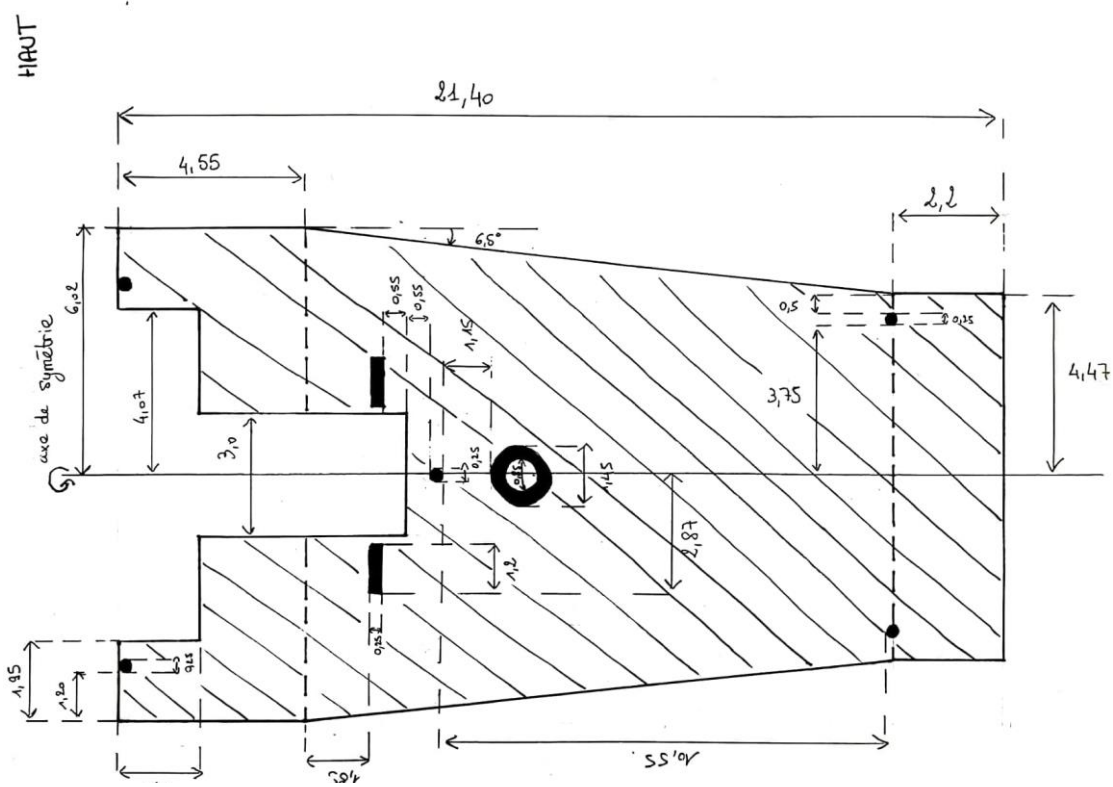
Carte FRDM-KL25Z légendée, source : <http://www.68hc08.net>

Ensuite, nous avons rencontré un problème au niveau des moteurs et cela était lié au pont H. Nous avons découvert qu'il y avait en fait une résistance brûlée et donc les moteurs tournaient en sens inverse l'un par rapport à l'autre.

De plus, les capteurs de vitesse ont dû être refaits par modélisation.

Enfin, nous avons dû gérer les problèmes au niveau du circuit. Tout d'abord, par rapport à la luminosité extérieure, les bandes blanches du circuit ne paraissait pas assez blanche pour la caméra et donc nous devons prévoir une plage de nuances à considérer comme une bande blanche. De plus il était difficile de reconstituer le circuit.

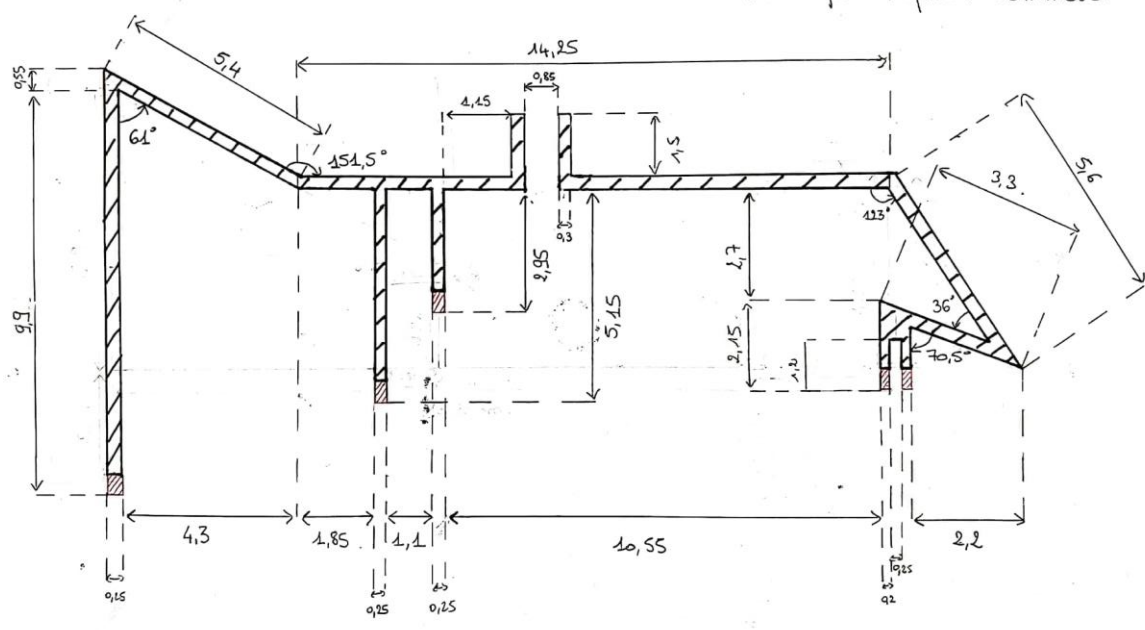
Dans une deuxième partie nous allons parler des améliorations apportées au robot, qu'elles soient plus ou moins importantes. Dans un premier temps l'amélioration majeure que nous avons effectuée est la « carrosserie ». En effet nous avons préalablement nettoyé le robot qui comportait un grand nombre de fils non reliés à la carte ainsi que des parties du châssis non utilisés. Par la suite nous avons modélisé à l'aide du logiciel « 123D Design » la pièce voulue après avoir pris des mesures. Vous trouverez ci-dessous le schéma fait à la main et sous différents points de vue de la pièce.



BAS

$z_{max} = 10,45 \text{ cm}$
 $x_{max} = 21,40 \text{ cm}$
 $y_{min} = 12,05 \text{ cm}$

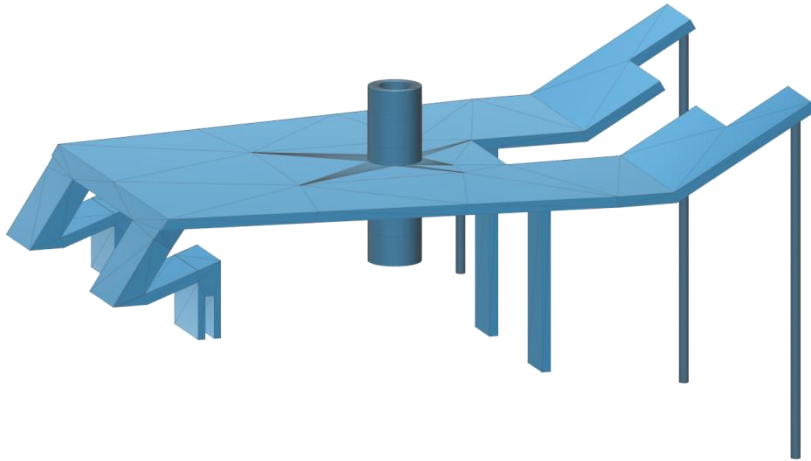
- épaisseur = 0,3 cm quand non indiquée
- unité par défaut : centimètre



GAUCHE

DROITE

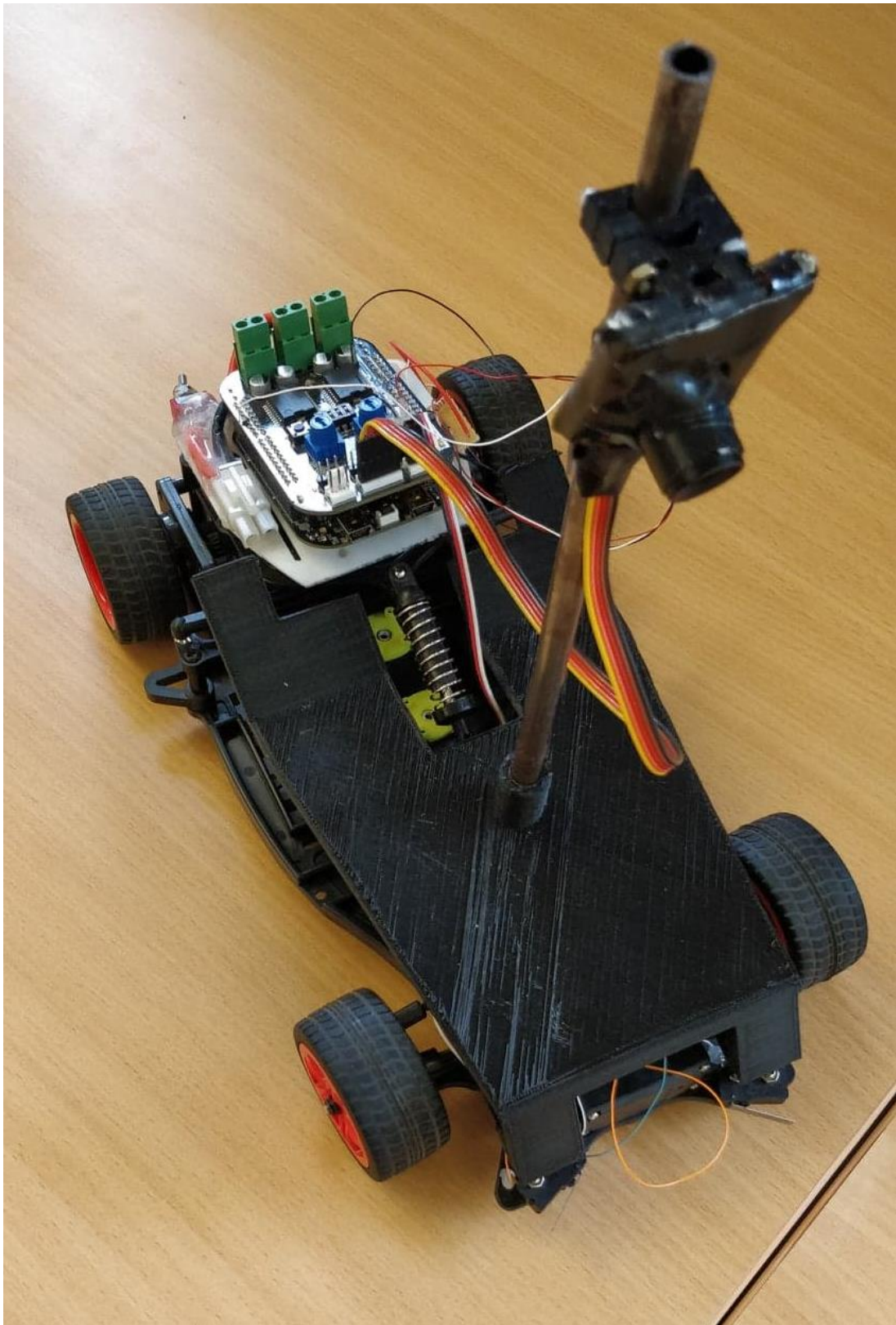
Sur le logiciel de modélisation la pièce a été assez rapide à concevoir malgré sa complexité, le résultat obtenu est le suivant.



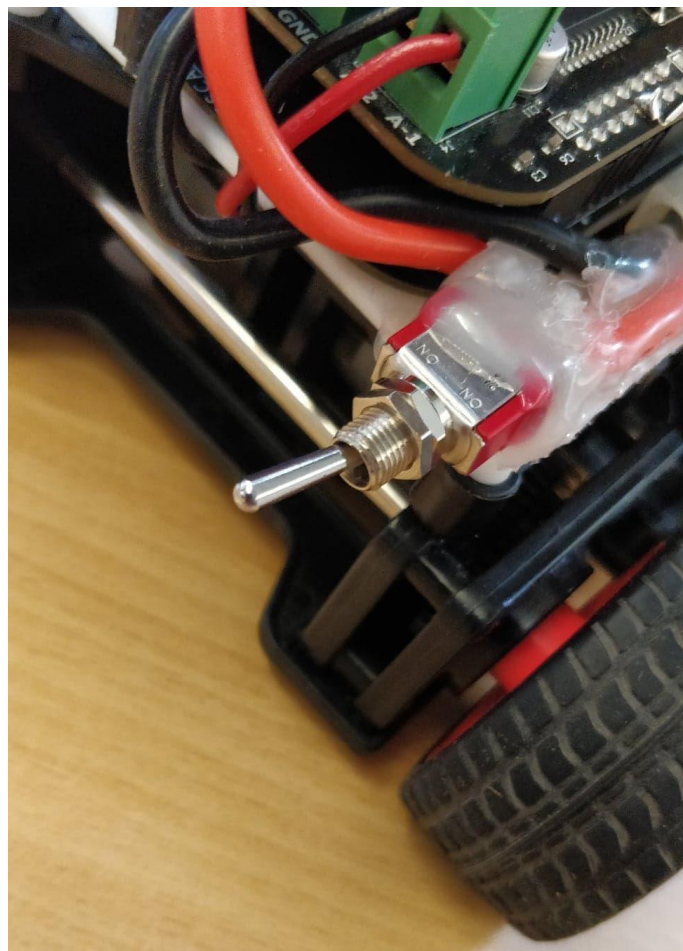
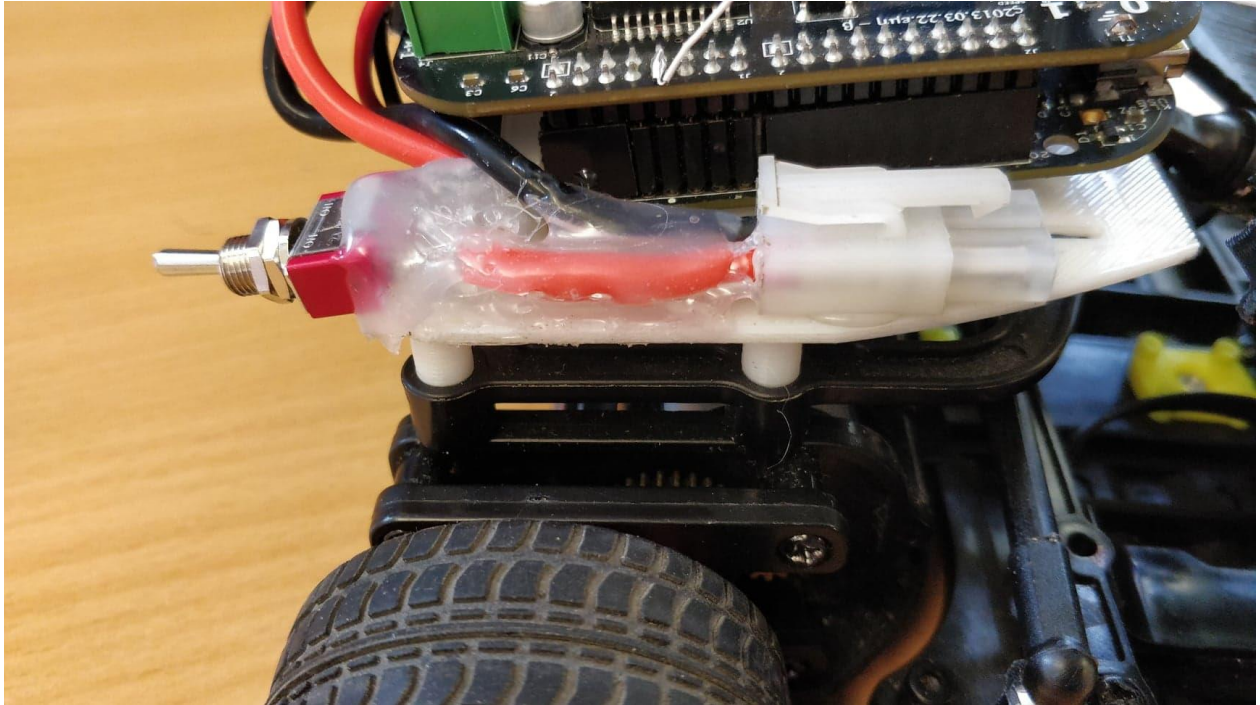
La conception de cette pièce n'apparaît pas seulement comme une amélioration esthétique mais surtout elle pallie à un problème majeur quand nous avons reçu le châssis de la voiture. Effectivement le mât sur lequel est accrochée la caméra était très instable et l'on voulait trouver une solution non seulement belle mais adéquat.

Après avoir modélisé la pièce nous avons utilisé l'imprimante 3D mise à notre disposition (Witbox 1^{ère} génération) pour concrétiser tout ce travail. L'impression a duré presque 12 heures et ne fût pas aux premiers abords une réussite. En effet l'imprimante a, par soucis de facilité, imprimé la pièce à l'envers, mais les trois tiges les plus fines présentes sur les croquis et modélisées n'ont pas pu être fabriquées correctement. Cela a été le cas car le diamètre de celles-ci était trop petit par rapport à la tête d'impression (ou la buse) de l'imprimante. Néanmoins nous nous sommes rendu compte qu'elles n'étaient pas indispensables donc nous avons pris la décision de les retirer définitivement pour obtenir la pièce montée sur le châssis de la voiture ci-dessous.

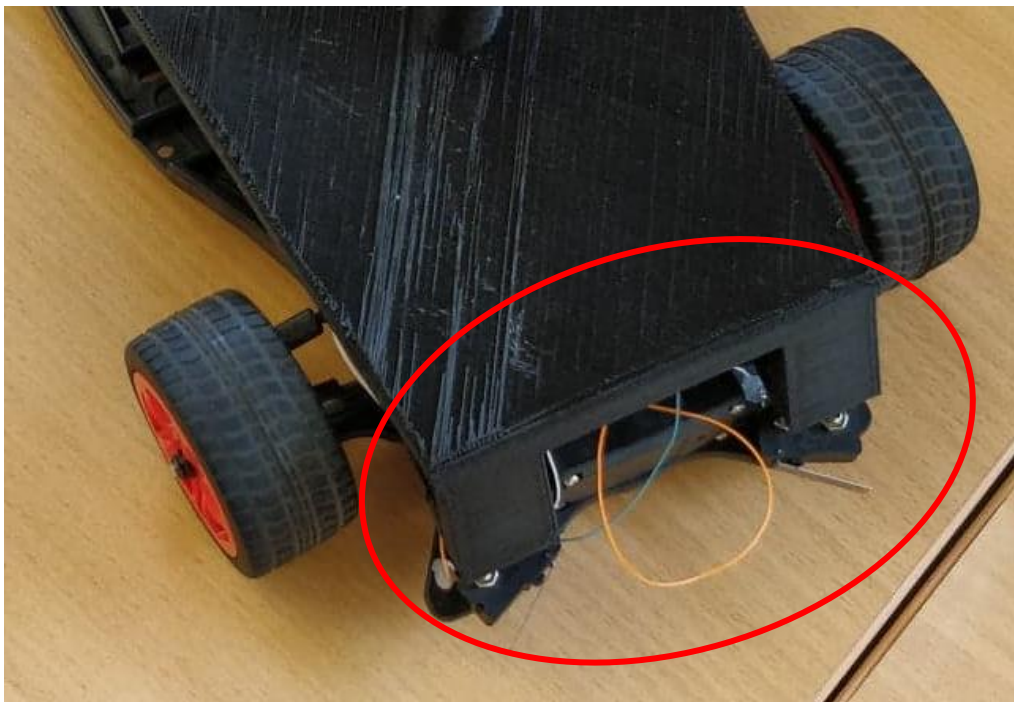




Dans un deuxième temps nous avons procédé à un changement mineur visant à améliorer la manipulation du robot lors des tests, qu'ils soient sur le circuit ou tout simplement sans faire rouler ce-dernier. Pour ce faire nous avons intégré un interrupteur, en amont de la carte. Effectivement après plusieurs essais nous nous sommes rendu compte de l'utilité de ce composant, de plus tous les groupes en possédaient et nous en constatons l'importance. Pour ce faire nous avons tout simplement sectionné les câbles arrivant de la batterie vers la carte, donc ceux alimentant en énergie celle-ci, puis soudé un interrupteur déjà conçu.



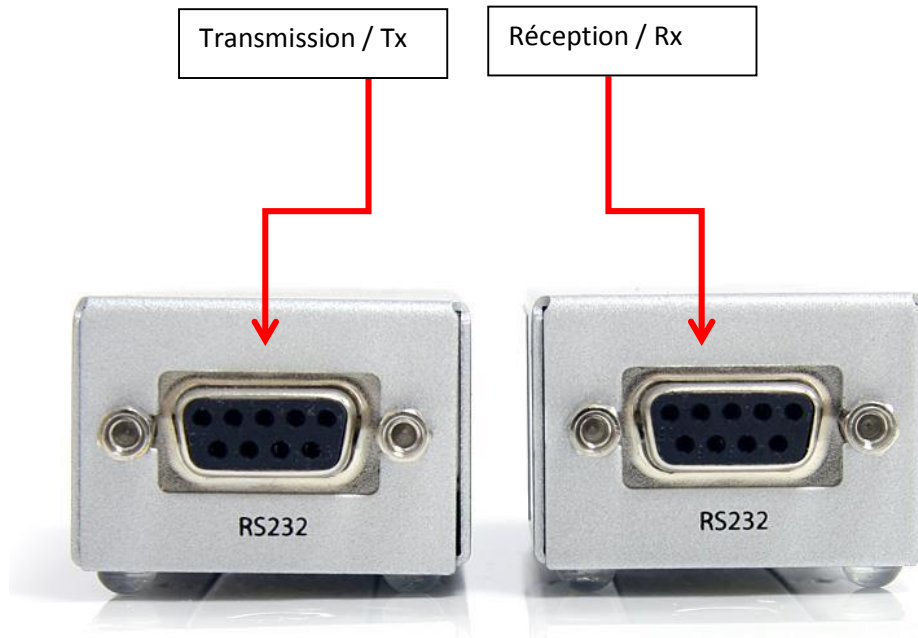
Dans le même ordre d'idée que l'interrupteur, nous avons installé des bumpers à l'avant de la voiture. Ces derniers ont pour but de protéger le moteur principalement contre tout choc pouvant intervenir lors de tests ou autres. Ils sont très importants puisque lors d'un choc contre un mur par exemple la voiture ne le prend pas en compte et ne s'arrête ce qui a pour effet d'échauffer le moteur jusqu'à la rupture irrémédiable de ceux-ci. C'est donc principalement pour cela que les bumpers sont cruciaux, en effet, lors d'un choc frontal le mur appuie en premier sur les tiges métalliques du bumper et déclenche un arrêt moteur. Tout ceci ne se fait pas naturellement et il a fallu procéder à quelques modifications dans le code ainsi que des branchements supplémentaires. Dans un premier temps il a fallu brancher en parallèle les deux bumpers situés de part et d'autre de l'avant de la voiture puis les câbler à la carte pour que ces derniers puissent transmettre des informations sur leur état et que le programme les exploite. Pour ce faire il fallait trouver un port disponible sur la carte, c'est-à-dire un port sur lequel aucune information ne transite. Après l'avoir choisi, nous avons soudé le fil transmettant les informations à ce port puis le deuxième câble à l'alimentation pour qu'ils puissent fonctionner. Dans un deuxième temps il nous fallait traiter ces informations envoyées par les bumpers, nous avons ainsi rajouté une condition de sorti du programme, autrement dit de la boucle principale qui exécute le programme, qui lorsque celle-ci se déclenchait la sortie de boucle et par conséquent l'arrêt du programme. Ci-dessous une photo des deux bumpers situés à l'avant de la voiture.



Dans un dernier temps nous avons amélioré le PID du programme, le PID, qui sera expliqué plus en détail dans une partie par la suite. Pour résumé il permet la correction de la trajectoire et de la vitesse du robot en fonction des précédents mouvements effectués. Cette méthode implique de lourdes fonctions mathématiques très complexes que nous ne pouvions mettre en œuvre dans notre cas, c'est donc pour cela que nous avons effectué un PID que l'on pourrait qualifier de manuel. En effet au fur et à mesure des tests nous modifions les valeurs attribuées aux changements d'angle pour qu'ils soient plus accentués ou moins pour obtenir un robot parcourant correctement le circuit.

III. LES CONNAISSANCES ACQUISES

Dans cette partie nous allons traiter des connaissances acquises au cours de ce projet. Dans un premier temps nous parlerons du principe de base de communication entre un ordinateur et un composant. Cette communication est réalisée à l'aide de deux fils, un pour la transmission et un autre pour la transmission de données. Dans le cas étudié nous parlerons du port RS232 illustré ci-dessous :



À savoir que ce système ne possède pas d'horloge interne et est donc asynchrone, cela signifie que la transmission des données n'est pas effectuée selon les fronts montants d'une horloge mais selon une vitesse de transmission propre à chaque composant. Cette vitesse est exprimée en bits par seconde ou encore en bauds, et dans notre cas elle est de 115200 bits par seconde concernant la transmission des données de la caméra. De plus les informations sont transmises en utilisant la notation binaire mais également des « bits » spéciaux dits de « start » et de « stop ». En effet, ils permettent une distinction claire du message. Enfin pour visualiser les données nous avons utilisé le logiciel Teraterm.

Ensuite nous allons aborder la notion de PID pour Proportionnelle, Intégrale et Dérivée ; elle a pour but de créer une rectification de la trajectoire et de la vitesse automatiquement. Pour ce faire des formules mathématiques sont mises en œuvre, qui réutilise les précédentes données pour pouvoir tirer des conclusions et appliquer des modifications par la suite. Ainsi ce procédé se divise en trois étapes ; premièrement proportionnellement c'est-à-dire la différence entre la consigne envoyée à la voiture et la valeur actuelle multipliée par un coefficient tel que :

$$K_c = e(t)$$

Ensuite cette différence est intégrée suivant des bornes allant de la consigne à la valeur actuelle, puis on multiplie par un autre coefficient comme ci-après :

$$\frac{K_c}{T_i} = \int e(t)dt$$

Enfin on dérive l'erreur actuelle, c'est-à-dire la différence entre la valeur voulue et la valeur obtenue, par rapport au temps et on multiplie par un dernier coefficient tel que :

$$-K_c T_d \frac{dP_V}{dt}$$

Finalement dans une dernière partie nous aborderons les connaissances acquises concernant l'autoguidage et donc par conséquent la caméra, outil principal dans la réalisation de ce procédé.

Dans un premier temps il est important de signifier que la caméra utilisée n'est pas une caméra classique avec laquelle on filme couramment mais une caméra captant une ligne composée de 128 pixels. En effet le traitement d'image est un procédé très lourd pour ce type de carte voire irréalizable dans certains cas, or nous voulions une correction en temps réelle de la trajectoire, de ce fait il fallait un temps minimal entre la réception des données et les consignes envoyées aux moteurs. Par conséquent ce choix est le plus optimal même si nous verrons par la suite qu'il n'est pas le meilleur. Une fois que nous maîtrisons l'utilisation et le traitement des données avec cette caméra, il nous a fallu respecter le cahier des charges qui est de parcourir le circuit, et donc pour ce faire de rester entre deux lignes noires.

Pour pouvoir respecter cette condition il faut tout simplement analyser les pixels récupérés par la caméra et plus précisément la valeur de ceux-ci pour pouvoir en déduire de la couleur de l'objet, et donc de si cette objet est une ligne, le circuit ou encore autre chose. Après avoir fait quelques tests nous nous sommes rendu compte que ce n'était pas une couleur en particulier que nous ciblions mais plutôt une plage qui va du gris plus ou moins foncé au noir. Une fois la ligne repérée il ne reste donc plus qu'à donner les consignes aux moteurs en prenant en compte les précédentes consignes (application du PID).

IV. CONCLUSION

Au terme des quelques mois qui nous ont été donnés pour avancer sur ce projet, nous avons eu l'occasion de réaliser de nombreuses améliorations tant dans le code de la voiture que dans l'esthétique et la maniabilité.

La voiture initialement reçue comportait des défauts et instabilités au niveau de la carrosserie - nous avons pallié à ces problèmes. Nous avons également ajouté deux composants qui ont facilité la manipulation de la voiture. L'interrupteur permet de mieux contrôler le départ et l'arrêt de la voiture, les bumpers forcent l'arrêt en cas de choc et préservent donc la voiture et ses moteurs.

Ces modifications ont amené à de nombreuses modifications dans les programmes fournis et nous a permis de nous familiariser avec le fonctionnement et le comportement des éléments composant la voiture.