Rapport technique : voiture autonome

Description du projet

Objectifs

Le produit est une **voiture autonome**, elle doit se **déplacer seule** sur un circuit en **évitant des obstacles**.

Matériel

Nous disposons d'un **châssis** *Tamiya Lancia Delta* de voiture téléguidée, munie d'une batterie et de deux servomoteurs. On l'équipera en plus de capteurs afin d'éviter les différents obstacles. Tout d'abord, nous utiliserons un **capteur infrarouge**, puis plusieurs et nous finirons avec un **LiDAR**, qui est un système plus performant. Nous pouvons piloter la direction des roues avant ainsi que la traction des quatre roues.

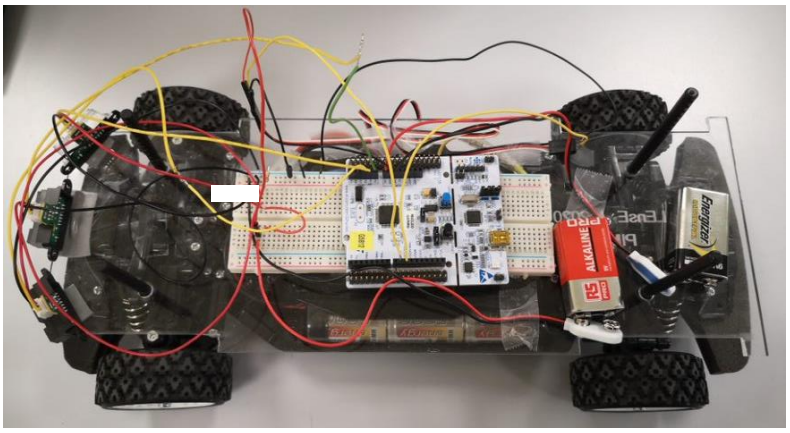

Figure 1 : Châssis de notre voiture équipé de capteurs iR.

Figure 2 : Capteur iR.

Figure 3 : LiDAR.

Problématique

Notre projet a été centré sur la problématique suivante : **quelles sont les étapes à mettre en place afin d'obtenir une voiture autonome dans un environnement avec des obstacles ?**

Cahier des charges

L'objectif est de rendre le véhicule autonome, c'est-à-dire qu'il puisse avancer et tourner en évitant les obstacles qui se présentent à lui.

Pour cela, différents objectifs devront être atteints :

1) la voiture peut se déplacer en ligne droite ;

2) la voiture peut s'arrêter devant un obstacle placé devant elle ;

3) la voiture peut éviter un obstacle qu'elle détecte en le contournant ;

4) la voiture peut traverser le couloir jusqu'au foyer ;

5) la voiture peut traverser le couloir jusque dans le hall d'entrée ;

6) la voiture peut reculer quand le contournement est impossible.

La voiture doit également pouvoir contrôler sa vitesse en fonction de son environnement c'est-à-dire qu'on régule la vitesse selon la proximité des différents obstacles.

Schéma structurel de la voiture

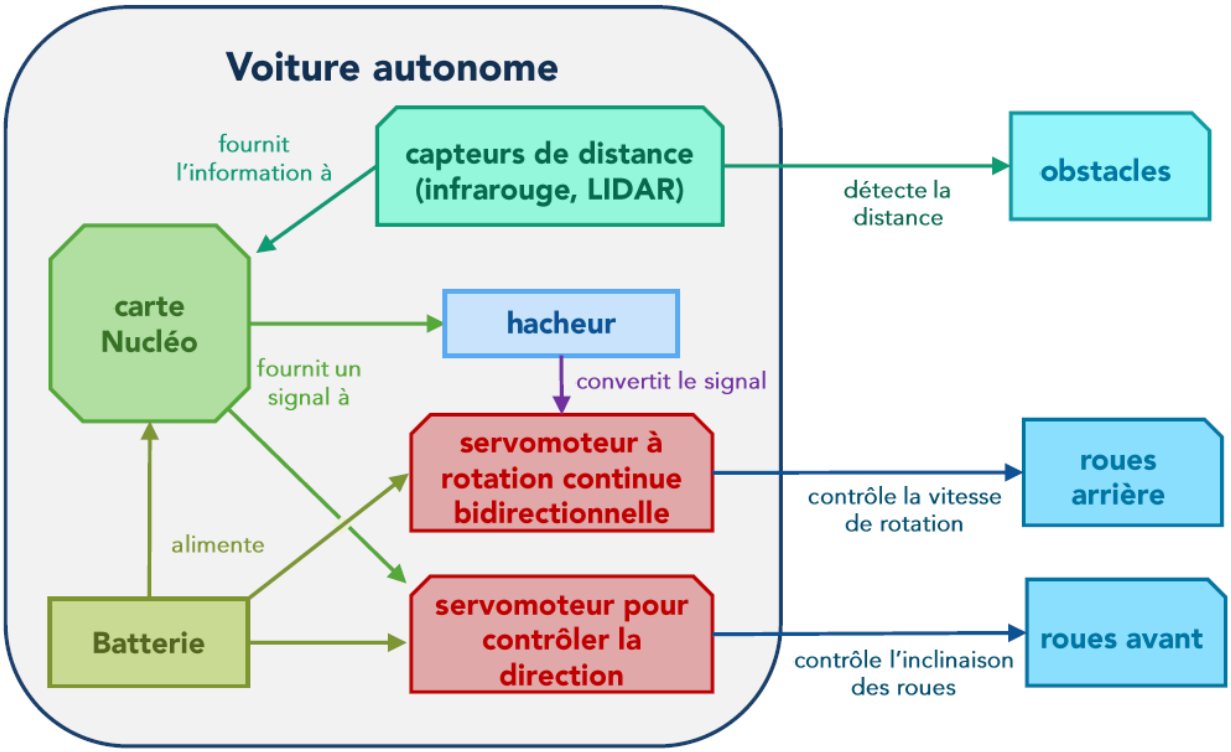

Figure 4 : Schéma général décrivant le fonctionnement de notre voiture autonome.

Découpage fonctionnel

Voici pour commencer un schéma de principe du projet avec les différentes fonctions que l'on a codées :

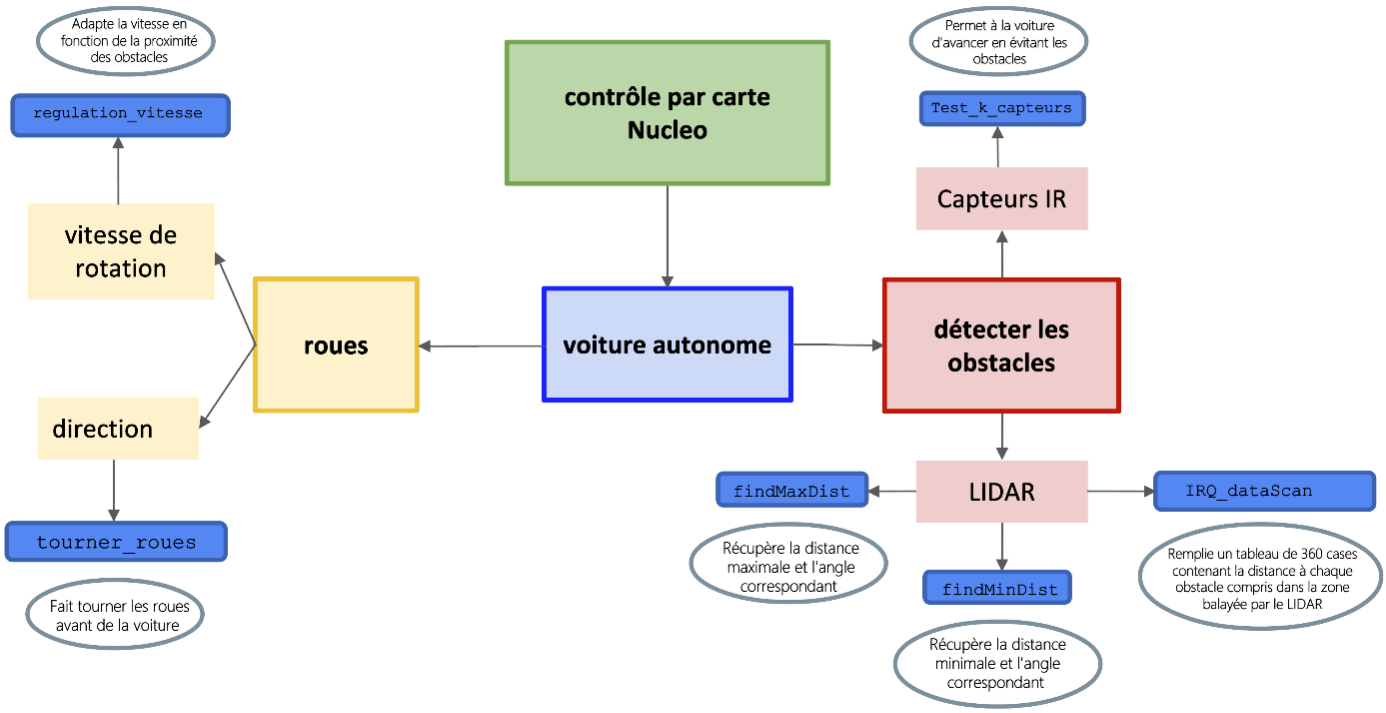

Figure 5 : Schéma général décrivant l'architecture de nos programmes.

Toutes les fonctions ont été réalisées sous MBED. On va donner maintenant une explication rapide de leur fonctionnement et de leur utilisation dans notre code global.

Tourner_roues

Cette fonction est des plus basiques, elle permet **d'orienter les roues** afin de pouvoir faire avancer la voiture dans une direction choisie et non uniquement en ligne droite. Le servomoteur est commandé avec un **signal PWM**. Pour cela on définit une période de 20 ms.

On observe que les roues sont droites pour une largeur du pic du haut de 1300 µs. La valeur maximale pour tourner à droite est 1520 µs et pour tourner à gauche est 1100 µs. Pour des valeurs plus éloignées de ces dernières, on est en butée ce qui n'est pas avantageux car on pourrait détériorer le système.

Roues dans l'axe	Temps haut de 1300 µs
Roues à gauche	Temps haut de 1100 µs
Roues à droite	Temps haut de 1520 µs

Figure 6 : Tableau récapitulatif des données utiles pour contrôler la direction des roues.

Avancer

Cette fonction est également des plus basiques, elle permet de **mettre en rotation les roues** afin de pouvoir faire avancer la voiture. Le servomoteur étant similaire au précédent, est aussi commandé avec un **signal PWM**. Pour cela on définit la même période de 20 ms.

On observe que pour une largeur du pic du haut autour de 1500 µs, la voiture est à l'arrêt. En dessous, les roues tournent en arrière et en dessus vers l'avant. Les valeurs maximales sont 1600 µs pour la rotation vers l'avant et 1460 µs vers l'arrière.

Arrêt	Temps haut de 1500 µs
Max vers l'avant	Temps haut de 1600 µs
Max vers l'arrière	Temps haut de 1460 µs

Figure 7 : Tableau récapitulatif des données utiles pour contrôler la vitesse des roues.

Test_1_capteur

La voiture est à présent équipée d'un **capteur infrarouge** à l'avant. Cette fonction permet à la voiture de se déplacer dans cette configuration. Bien évidemment, les fonctions `avancer` et `tourner_roues` sont utilisées.

Nous avons caractérisé le capteur utilisé :

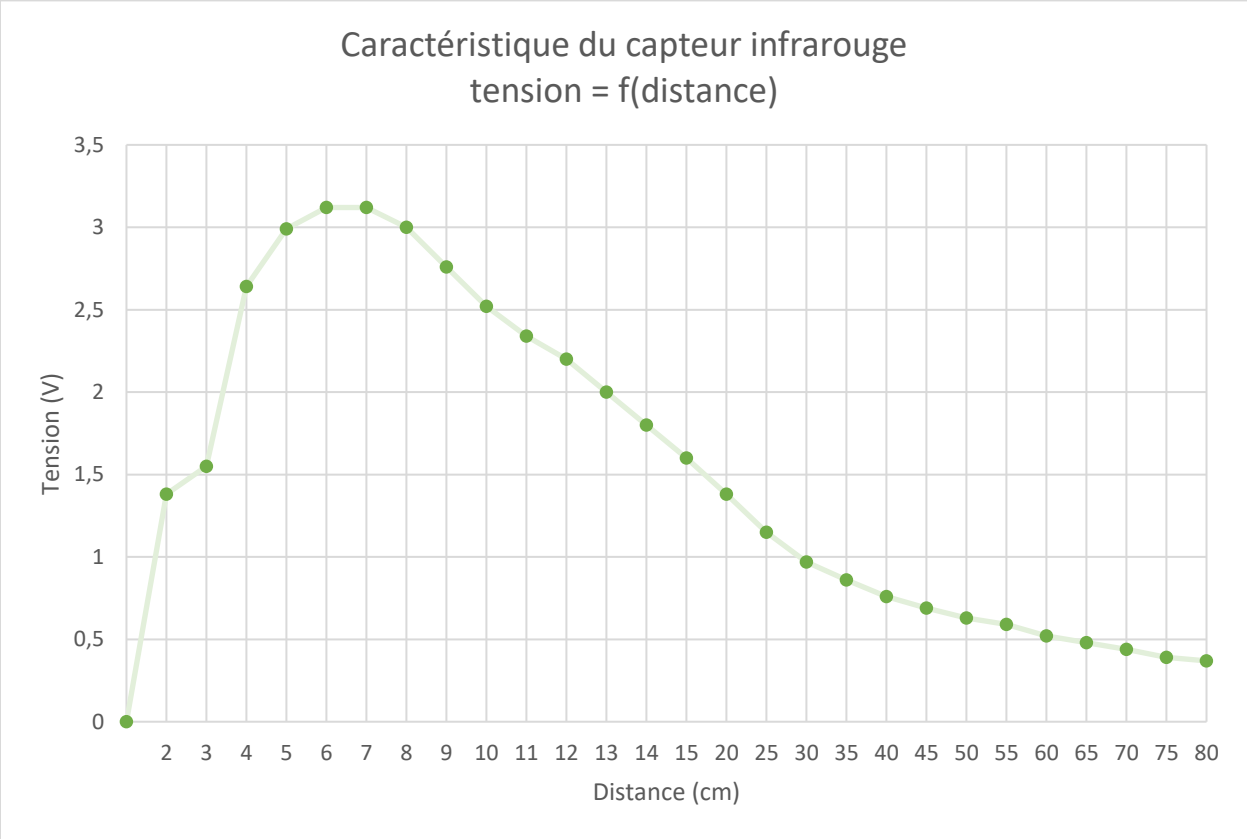

Figure 8 : Caractéristique d'un capteur infrarouge utilisé.

On constate une bonne cohérence avec la courbe théorique fournie par le constructeur :

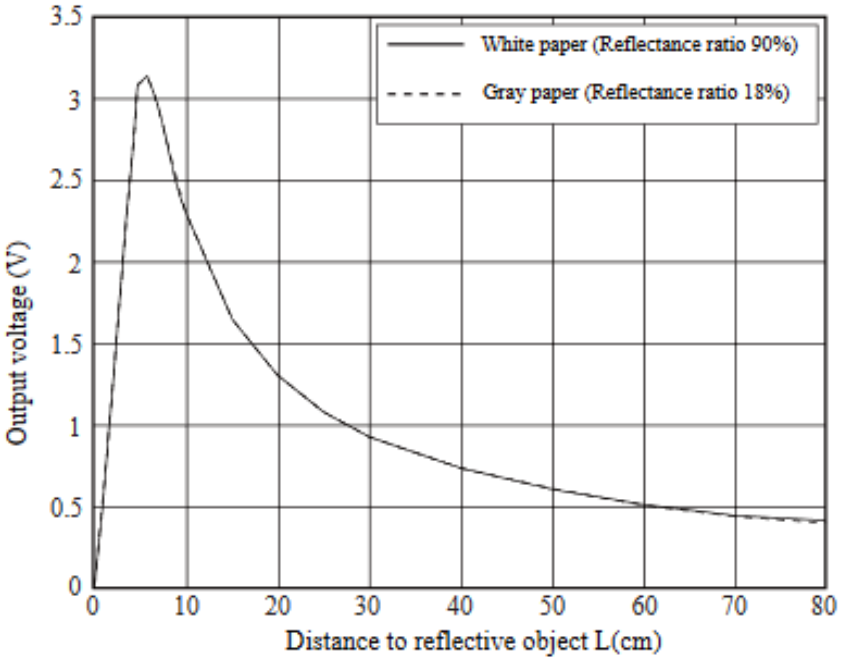

Figure 9 : Caractéristique d'un capteur infrarouge utilisé, fournie par le constructeur.

Avec notre caractéristique expérimentale obtenue, on définit des seuils pour savoir si la voiture doit avancer, ou s'arrêter. Si la tension du capteur est comprise entre 0.8 V et 2.6 V, les roues sont alignées avec le châssis et sont en rotation. Dès que la tension dépasse 2.6 V, la voiture s'arrête.

Cas	Tension	Vitesse (µs)	Rotation (µs)
1	0.8 < U < 2.6	1600	1300
2	U > 2.6	1500	✗

Figure 10 : Récapitulatif des consignes à donner au moteur en fonction de la tension mesurée par le capteur iR.

Test_3_capteurs

La voiture est à présent équipée de **trois capteurs infrarouges** à l'avant. Cette fonction permet à la voiture de se déplacer dans cette configuration en évitant les obstacles. Bien évidemment, les fonctions `avancer` et `tourner_roues` sont utilisées, et les trois capteurs sont identiques.

L'idée est la même que pour la fonction `Test_1_capteur`, on va définir des **seuils** pour laquelle les roues sont mises en rotation et comment elles sont orientées.

Les différentes valeurs sont répertoriées dans le tableau ci-dessous :

Cas	Tension droite (V)	Tension milieu (V)	Tension gauche (V)	Vitesse (µs)	Rotation (µs)
a	✗	1.5 < U < 2.6	1.5 < U < 3	1600	1520
b	✗	U < 0.8	U > 2.5	1600	1520
c	1.5 < U < 3	1.5 < U < 2.6	✗	1600	1100
d	U > 2.5	U < 0.8	✗	1600	1100
e	U > 3	U > 2.6	U > 3	1500	1300

Figure 11 : Récapitulatif des consignes à donner aux servomoteurs en fonction de la tension reçue par les capteurs iR.

Cas a : l'obstacle est trop près à gauche tout en étant détecté par le capteur du milieu

Cas b : l'obstacle est trop près à gauche sans être détecté par le capteur du milieu

Cas c : l'obstacle est trop près à droite tout en étant détecté par le capteur du milieu

Cas d : l'obstacle est trop près à droite sans être détecté par le capteur du milieu

Cas e : l'obstacle est trop proche de l'avant de la voiture

Lidar_basique

A présent, on a remplacé les 3 capteurs infrarouges par un **RPLIDAR 2**. On aura ainsi la connaissance d'un plus grand nombre d'informations afin de diriger au mieux notre voiture. On peut à présent **scanner une zone comprise entre 15 cm et 6 m** du LIDAR et **à 360 degrés**.

Pour commencer, on va utiliser la méthode la plus intuitive possible : le LIDAR nous renvoie les **angles** pour lesquels il y a la **distance maximale et la distance minimale** avec un objet. On va alors agir de sorte que le robot **se déplace dans la direction du maximum**. Il faut également veiller à choisir un intervalle angulaire adéquat car on n'aura aucune idée de la situation de la voiture à l'avant si les deux valeurs sont à l'arrière.

Régulation_vitesse => optimisation

Notre idée était de réguler la vitesse de la voiture en fonction de la taille de la zone vide devant la voiture. Plus l'obstacle est lointain, plus la voiture peut rouler vite. Des obstacles détectés à proximité doivent la faire ralentir.

Le programme a été élaboré. Cependant, il n'a pas pu être testé en situation réelle avec notre voiture. Le programme basique a en effet nécessité d'être amélioré afin de fournir un résultat convenable, visible dans la vidéo de présentation du projet.

Prototype

A) Voiture avec un seul capteur

Voici à quoi ressemblait notre voiture au début du projet (**Figure 12**). On observe que le capteur est placé au centre afin d'éviter tout choc frontal. L'utilisation de la pile permet d'alimenter le capteur en soulageant la batterie.

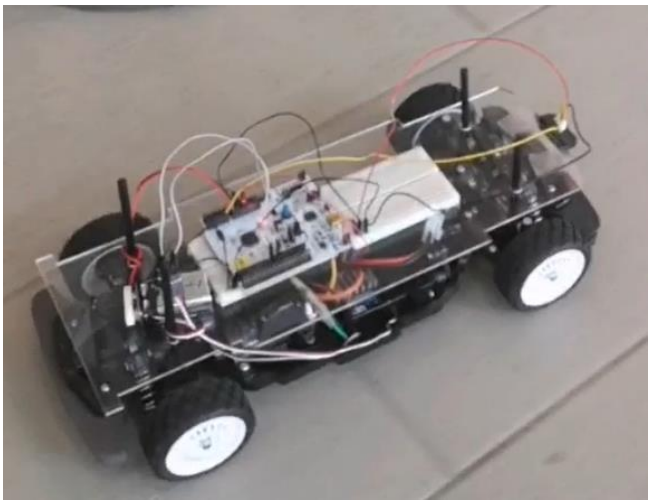

Figure 12 : Châssis de notre voiture au début du projet.

Pour commander la voiture, on utilise la fonction Test_1_capteur. Le cœur du code se présente sous la forme suivante :

main.cpp

```
        if(my_bp == 1){
            servo_avancer.pulsewidth_us(1600);
            servo_tourner.pulsewidth_us(1300);
            mesure = capteur.read_u16();
            tension = mesure/ 65536.0 * 3.3;
            wait_us(20);
            while(tension>0.8 && tension<2.6){
                servo_tourner.pulsewidth_us(1520);    // roue vers la droite
                servo_avancer.pulsewidth_us(1600);
                mesure = capteur.read_u16();
                tension = mesure/ 65536.0 * 3.3;
                wait_us(20);
                }
            while(tension>2.6){
                servo_avancer.pulsewidth_us(1500);
                mesure = capteur.read_u16();
                tension = mesure/ 65536.0 * 3.3;
                wait_us(20);
                }
```

On observe l'initialisation avec le bouton poussoir via le `my_bp==1` qui nous permet de faire démarrer la voiture au moment souhaité.

Les tests nous permettent de bien avancer uniquement en l'absence d'obstacle devant avec la tension inférieure à 2.6 V. On a également ajouté supérieure à 0.8 V pour empêcher le rebond lorsque l'obstacle est trop proche.

B) Voiture avec trois capteurs

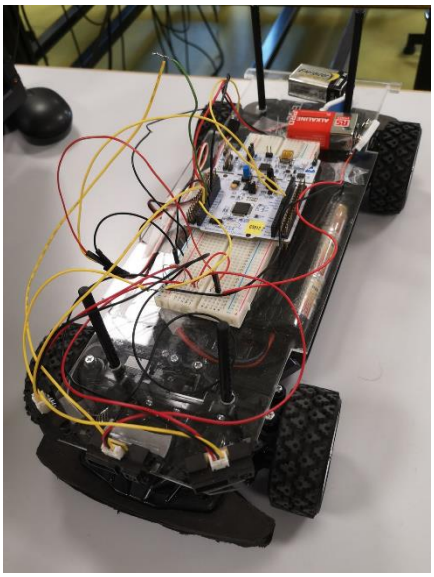

Figure 13 : Châssis de notre voiture équipée des 3 capteurs infrarouge.

Voici à quoi notre voiture ressemble lorsque l'on ajoute les trois capteurs (**Figure 13**).

Le principe général de câblage reste le même mais on constate l'ajout d'une deuxième pile afin de compenser la demande plus importante en courant liée à l'ajout de deux capteurs supplémentaires.

Cette fois-ci nous utilisons la fonction `Test_3_capteurs`. Le cœur du code est téléchargeable en annexe. Il est analogue dans l'utilisation des capteurs à celui réalisé avec un seul capteur.

On a réalisé une disjonction de cas selon le tableau de la partie algorithmique et selon la position de l'objet on oriente les roues de l'autre côté ou on stoppe la voiture.

C) Voiture avec le LiDAR

La voiture est maintenant équipée d'un LiDAR. Un programme fourni par Julien Villemejane permet d'acquérir les données du LiDAR. Plus précisément, on recherche le minimum de distance et l'angle associé ainsi que le maximum de distance et l'angle associé quand le LiDAR fait un tour.

Cela nous permet d'orienter la voiture dans la direction de ce maximum, et ainsi de pouvoir avancer le plus longtemps possible sans rencontrer d'obstacle.

Pour commencer, nous avons cherché le maximum dans un cône de 140° centré dans la direction de déplacement de la voiture. L'idée était ensuite de pouvoir réduire les cônes dans lesquels les mesures sont réalisées en fonction de la vitesse de la voiture.

Test réalisés

A) Voiture avec 1 capteur

On trouvera une vidéo de cette voiture en action via le lien suivant :

https://youtu.be/cYt2C0jj4RU

Cette vidéo est très concluante, la voiture est à présent capable de :

- rouler en ligne droite ;
- s'arrêter en présence d'un obstacle.

Cependant on arrive aux limites de ce modèle car la voiture est incapable d'éviter l'obstacle et donc de rouler sauf en ligne droite.

Afin de pallier ce problème, on décide d'ajouter des capteurs orientés sur les côtés afin d'élargir la zone visible pour la voiture.

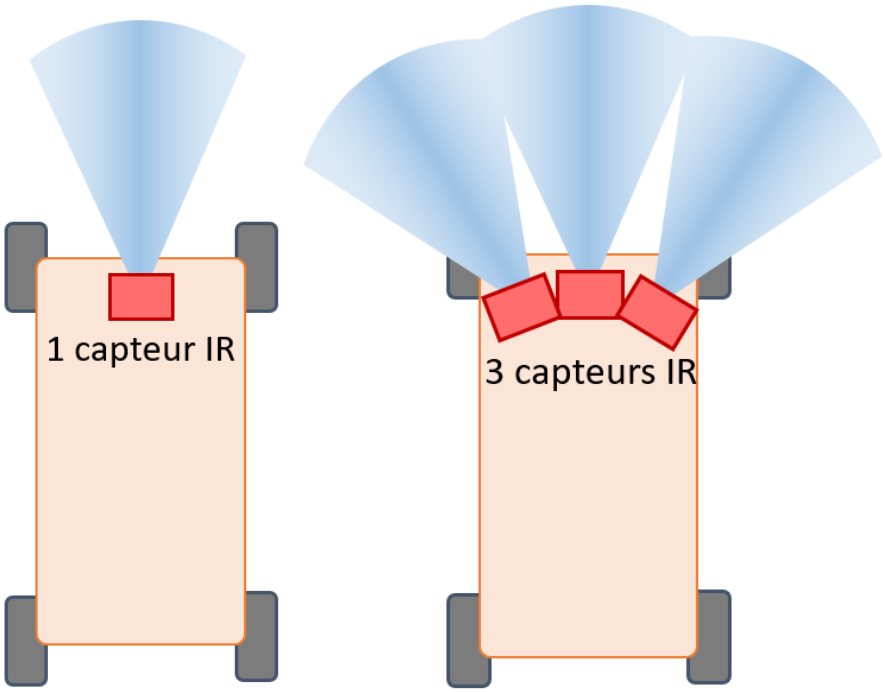

Figure 14 : Cônes de visibilité accessibles avec un capteur ou trois capteurs.

B) Voiture avec 3 capteurs

On trouvera une vidéo de cette voiture en action via le lien suivant :

https://youtu.be/A12S28R7Bhw

Cette vidéo est très concluante car la voiture peut dorénavant :

- éviter un obstacle comme un mur ;
- passer une porte ;
- s'arrêter si un obstacle est trop proche ;
- rouler à côté d'un obstacle courbé ;
- slalomer entre des obstacles proches.

C) Voiture avec le LiDAR

Les tests du programme basique se sont révélés concluants. Cependant, lors de nos premiers essais, au bout d'une dizaine de virages, le LiDAR semblait ne plus transmettre correctement les données à la voiture, et celle-ci se bloquait dans une direction aléatoire, tournant souvent en rond. Cette erreur a pu être résolue en modifiant une boucle `for` et en rectifiant la typographie d'une adresse mémoire en hexadécimal, peut-être à la base du problème.

Une démonstration avec le LiDAR est accessible par ce lien :

https://youtu.be/s-L15xN3l6s

Les problèmes liés à ce programme ont ensuite été résolus, comme cela est visible dans la vidéo de démonstration du projet entier.

Démonstration du projet entier

Une démonstration du projet entier est trouvable ici :

https://youtu.be/udT9UBxOV1g

Conclusion

Nous avons donc caractérisé les capteurs infrarouges puis mis en place un programme simple avec un capteur, puis trois capteurs situés devant la voiture afin de pouvoir éviter des obstacles frontaux et latéraux.

Nous avons ensuite étudié le fonctionnement du LiDAR et nous l'avons mis en place sur la voiture grâce à la carte imprimée par Julien Villemejane, puis nous avons élaboré un programme basique permettant de diriger la voiture vers la direction où l'obstacle est le plus lointain. Ce programme a correctement fonctionné et permis à la voiture d'évoluer pendant plusieurs minutes dans le couloir de manière parfaitement autonome.

Nous avons également élaboré un programme plus complexe permettant de réguler la vitesse selon les situations, mais qui n'a cependant pas pu être testé en situation réelle.

Annexes

Test_3_capteurs :

https://institutoptiquefr-my.sharepoint.com/:u:/g/personal/gonzague_bonin_institutoptique_fr/Ef_BViXLqf5CtJ3c03HPOwIB15b5tCBlExCU2Kg7RhZ7JQ?e=h9zenO

Lidar_basique :

https://institutoptiquefr-my.sharepoint.com/:u:/g/personal/gonzague_bonin_institutoptique_fr/EU_hRAdt87tMkzPAMPJnrilBLSLX7lrFbQaXpCnesicdDQ?e=8iE4Gu

Régulation_vitesse :

https://institutoptiquefr-my.sharepoint.com/:u:/g/personal/gonzague_bonin_institutoptique_fr/EYKceXuj6tlDokMNAyCUanYBYQzeH_vm2gWrYMv2dNA_uw?e=lkNKZO

Rétro-planning

Notre groupe est divisé en deux petites équipes : l'équipe 1 avec Martin et Clément et l'équipe 2 avec Gonzague et Matei.

Séance 1

Équipe 1 :

- test et programmation du servomoteur gérant la direction des roues **OK**
- étalonnage du capteur de direction **OK**

Équipe 2 :

- test et programmation du servomoteur gérant la traction de la voiture (marche avant et arrière) **OK**
- programmation du capteur de direction **OK**

A la fin de la séance, la voiture avance en ligne droite. **OK**

Séance 2

Équipe 1 :

- étalonnage de deux autres capteurs de direction (vérification qu'ils vérifient la caractéristique obtenue en séance 1) **OK**

Équipe 2 :

- programmation de la voiture en ligne droite, qui s'arrête devant un obstacle détecté **OK**
- réglage de la distance d'arrêt **OK**
- début de programmation d'un contournement d'objet situé d'un côté de la voiture **OK**

A la fin de la séance, la voiture avance en ligne droite, s'arrête devant un obstacle et peut contourner un obstacle simple situé d'un côté de la voiture. **OK**

Séance 3

Équipe 1 :

- installation des deux autres capteurs de distance (un sur le devant du véhicule, deux sur les côtés avant-droit et avant-gauche) **OK mais problèmes de puissance d'alimentation**

Équipe 2 :

- programmation des trois capteurs afin de détecter et spatialiser sommairement l'objet, réaction de la voiture adaptée **OK**
- calibration des capteurs et de la distance d'arrêt ou du virage de la voiture **Séance suivante**

A la fin de la séance, la voiture peut éviter un obstacle frontal ou latéral, ou reculer si besoin. **Test réalisé en séance suivante**

Séance 4

Équipe 1 :

- finalisation de la détection d'obstacles par les capteurs de distance **OK, résolution des problèmes de puissance**
- prise en main du Lidar (caractéristiques) **OK**

Équipe 2 :

- finalisation de la détection d'obstacles par les capteurs de distance **OK**
- familiarisation avec la programmation du Lidar **OK, problèmes de communication entre l'ordinateur et le LiDAR**

A la fin de la séance, la voiture peut éviter un obstacle frontal ou latéral, ou reculer si besoin, de manière adaptée et aboutie. Le Lidar a commencé à être étudié théoriquement ou peut-être en pratique. **OK, problèmes de communication en pratique**

Séance 5

Équipe 1 :

- vérification des caractéristiques du Lidar **Non réalisé**

Équipe 2 :

- implantation du Lidar dans le code pour détecter les objets **OK en version très basique, tests préliminaires**

A la fin de la séance, le Lidar a été caractérisé et des premiers tests avec le véhicule ont été menés si le temps le permet. **OK, tests non réalisés de manière concluante**

Séance 6

Équipe 1 :

- étalonnage du Lidar et des distances d'arrêt ou de virage pour éviter les obstacles **OK**

Équipe 2 :

- implantation de ces modifications dans le programme **OK**

A la fin de la séance, la voiture peut éviter des obstacles grâce au Lidar. **OK mais bug : la voiture tourne en rond au bout d'une dizaine de virages**

Séance 7

Équipe 1 :

- test complet avec le Lidar dans le couloir **OK mais bug persistant**

Équipe 2 :

- test complet avec le Lidar dans le couloir **OK**
- éventuelles modifications du programme pour des corrections mineures **OK, bug en cours de résolution**

A la fin de la séance, la voiture peut éviter des obstacles grâce au Lidar, de manière fiable et précise. **OK mais bug pas tout à fait résolu**

Séance 8

Forum de démonstration. **Bug résolu en modifiant le programme : la voiture peut maintenant avancer de manière autonome sur une grande distance !**