Adam BOYLAN G4

Aubin RAHMANI G1

RAPPORT FINAL PROJET ARDUINO

LE DOS D'ANE INTELLIGENT

Polytech Nice Peip 2

SOMMAIRE

I- Introduction

Les ralentisseurs ou dos d'ânes sont des aménagements présents dans la plupart des agglomérations ayant pour but principal d'assurer la sécurité des usagers de la route ainsi que les piétons en imposant aux véhicules de ralentir à l'approche du bombement de la chaussée. Ils sont synonymes d'inconfort pour les conducteurs qui doivent subir cette bosse mais aussi dégradant pour la voiture qui use ses plaquettes de frein, consomme plus de carburant à cause de la réaccélération et parfois même frotte le bas-de-caisse quand les dos d'ânes sont hors normes.

L'idée qui nous est venue était d'élaborer un dos d'âne intelligent capable de changer de forme en fonction de la vitesse d'arrivée du véhicule afin de récompenser les bons comportements et pénaliser les délits. De plus, nous voulions introduire un système de désactivation du radar menant à un dos d'âne abaissé pour les véhicules prioritaires (police, pompiers, …)

Le principe était d'avoir une position initiale dite « à plat » c'est-à-dire le dos d'âne n'existe pas au niveau de la sensation pour le conducteur qui peut poursuivre sa conduite naturellement, et une position finale dite « inclinée » répliquant l'aspect d'un dos d'âne traditionnel.

Pour passer d'une position à l'autre nous avions prévu d'utiliser un radar infrarouge pour mesurer la vitesse du véhicule arrivant, au final nous avons changé légèrement d'approche en optant pour un radar qui envoi des ondes ultrasons et qui déduit la vitesse à partir de la distance parcourue en un certain temps. En ce qui concerne le mécanisme d'inclinaison de chaque face du dos d'âne, nous avions prévu d'utiliser d'une façon ou une autre des moteurs, ce qui a bien été réalisé de manière simplifiée. Enfin, pour le système de véhicules prioritaires nous avions songé à un lecteur de plaque ou un capteur style « télépéage » mais nous avons choisi une télécommande et un capteur infrarouge pour le faire.

II- Les différentes parties du projet

A- Le radar de vitesse

Pour faire fonctionner notre dos d'âne il nous fallait absolument un radar calculant sans cesse la vitesse afin d'obtenir des résultats précis et une activation sans trop d'erreurs du levé des faces du dos d'âne. La question était de savoir comment s'y prendre, avec quels modules et lesquels étaient les plus efficaces. Nous avions à disposition plusieurs modules qu'on pouvait exploiter ou utiliser :

-Le Radar Doppler, un nouveau module jamais utilisé commandé par l'Ecole qui nous a été conseillé. Le principe est simple, le radar envoie une onde et reçois l'onde réfléchis par l'obstacle. Il calcule en plus de la distance, le temps perçu de l'aller-retour de l'onde envoyé. On a donc la vitesse instantanée du véhicule. Cependant, on a très vite abandonné cette idée car le module était beaucoup trop complexe à utiliser et minutieux, il aurait fallu commander un nombre incalculable de petite pièce pour le faire fonctionner.

-Le Radar laser, un module envoyant un laser sur la voiture et réfléchis par cette dernière capte le laser réfléchi. Le module calcule la distance entre la voiture et l'émetteur. De plus, le module calcule également le temps parcouru du laser. Connaissant donc la distance, le temps et la vitesse de la lumière il est très facile de calculer la vitesse.

Nous ne l'avons pas choisi car cette technique n'est pas utilisée sur les radars fixes utilisés en circulation Le radar tronçon calcule la vitesse moyenne d'un véhicule. Il est possible de le réaliser en mettant des capteurs sur le sol qui se déclenche lorsqu'une voiture roule dessus, et s'arrête lorsque le voiture franchis les autres capteurs placés à une distance X. En mesurant le temps de parcours du véhicule, on obtient la vitesse. On ne l'a pas utilisé car cela était trop dur à réaliser et cela ne convient pas en ville, car on peut capter un humain ou un animal ou encore un vélo. Nous avons opté pour le module hc-sr04, un capteur de distance à ultrason permettant d'évaluer les distances entre un objet mobile et les obstacles rencontrés. Il est équipé de deux capsules, un récepteur et un émetteur. Un signal de 10 micro secondes est envoyé sur la broche Trigger pour émettre des ultrasons à 40 KHz via la capsule émettrice du module. La seconde capsule permet de réceptionner l'onde réfléchi de l'ultrason. L'information est envoyée sur la carte Arduino via la broche Echo.

Le calcul de la distance n'est pas fait directement sur le module mais via la carte Arduino grâce au code que nous avons écrit. La distance mesurée est donc égale à la (Durée totale (aller et retour de l'ultrason) * la vitesse du son (0,34 cm/µs)) / 2. Grâce à la distance calculée, nous pensions que nous pouvions utiliser cette relation suivante : v = D/T. Avec T la durée Totale. Cependant, on s'est rendu compte qu'il était impossible d'utiliser cette relation car on aurait une vitesse constante (et fausse) quel que soit la distance. On a donc décidé de créer une fonction qui implémente le calcul de la distance et créer un Delta T, un temps choisi et fixé par nous entre chaque mesure. L'idée est d'envoyer deux ondes d'ultrason à un intervalle donné qui se réfléchissent et qui sont

réceptionnées par le module. Nous avons appelé deux fois la fonction, afin d'avoir la mesure. Nous avons donc la vitesse grâce à cette relation suivante : V = Delta D / Delta T avec D = distance 1 – distance 2 et DeltaT = 500 millisecondes.

Nous obtenons bien une vitesse qui est modifié constamment plus ou moins vite en faisant varier le DeltaT. Nous avons tout converti en secondes.

Grâce à cette vitesse nous avons pu débloquer toutes les autres fonctionnalités du projet. Nous avons donc converti la vitesse en km/h et affiché cette dernière sur un écran LCD (développer plus ?) de 16 pixels sur 2 lignes. Nous avons choisi le message Speed (vitesse en français), suivi de la vitesse qui est modifié constamment en fonction de la vitesse du véhicule, et suivi de « km/h ». Sur la deuxième ligne nous avons mis un message personnalisé selon la vitesse du conducteur. Si le véhicule dépasse les 0.8 km/h, (à notre échelle), le message signalait le conducteur de ralentir, et si le conducteur conduisait à une allure normale, (inférieur à 0.8 km/h), le message signale que le conducteur à une bonne allure.

B- La maquette du dos d'âne

Naturellement, ce n'était pas envisageable de concevoir un dos d'âne à l'échelle de vraies voitures, nous avons donc choisi de faire une maquette miniature pour des petites voitures Arduino. Il n'y avait pas grand-chose à faire pour la réaliser : un support sur lequel seront attachés les deux faces du dos d'âne avec une charnière. Il n'y avait qu'une seule règle à respecter : avoir une maquette la plus à plat possible pour avoir une traversée optimale lorsque le dos d'âne est escamoté. Comme support nous avons pris une planche en bois très fine de 80*40 cm sur lequel nous avons attaché de part et d'autre, avec une charnière flexible permettant le mouvement, chaque face du dos d'âne elles aussi très fines de 40*30 cm.

Hélas, nous nous sommes rendus compte par la suite que le moteur permettant l'inclinaison devait être placé légèrement sous le support, de fait pour combler à cela nous avons placer des pieds de 0.8cm aux quatre coins du support pour éviter que la maquette tienne en équilibre sur les moteurs.

C- Le système d'inclinaison mécanique

Pour passer de la position initiale à plat à une position inclinée, il fallait songer à un mécanisme capable de soulever chacune des faces du dos d'âne des façon synchronisée et relativement rapide.

Premièrement il fallait considérer un angle d'environ 10° pour obtenir un sommet à environ 7cm. Pour y arriver, l'idée était de placer en dessous de chacune des faces un moteur à courant continu qui une fois mis en rotation, permettrait l'inclinaison avec son hélice pour ensuite le maintenir dans cette position et résister au poids de la voiture surmontant.

Il fallait que l'axe de rotation de l'hélice soit au même niveau que le support, autrement dit qu'une partie du moteur soit en dessous du support. De fait, nous avons percer deux trous de la taille des moteurs directement en dessous de chaque face pour ensuite les fixer.

Il suffisait désormais de programmer l'angle de rotation de chaque moteur c'est-à-dire 90° afin d'avoir une amplitude maximale en veillant bien à paramétrer un des angles à l'opposé afin d'avoir deux moteurs qui tournent dans le sens opposé. Puisque les deux moteurs étaient reliés au même programme il n'y avait pas de souci de décalage : les deux faces montaient en même temps au même niveau.

D- La télécommande infrarouge

Pour répondre à la question sur les véhicules prioritaires, on avait d'abord envisagé d'utiliser l'intelligence artificiel qui permettrait de reconnaître les véhicules prioritaires que cela soit un véhicule de police, de pompier ou encore de d'ambulance. On avait donc pour idée de prendre beaucoup de photos de ces véhicules et de les stocker dans la mémoire de l'intelligence artificielle. Or cela était impossible pour nous de réaliser cela car on aurait perdu beaucoup trop de temps pour une fonction annexe du projet, bien qu'importante.

Nous nous sommes donc dirigés vers une télécommande infrarouge qui quand on active cette dernière, enclenche bloque le système de monté du dos d'âne. Cela permettrait aux véhicules prioritaires de ne pas ralentir et ne pas perdre de temps grâce à la communication rapide de la télécommande à infrarouge.

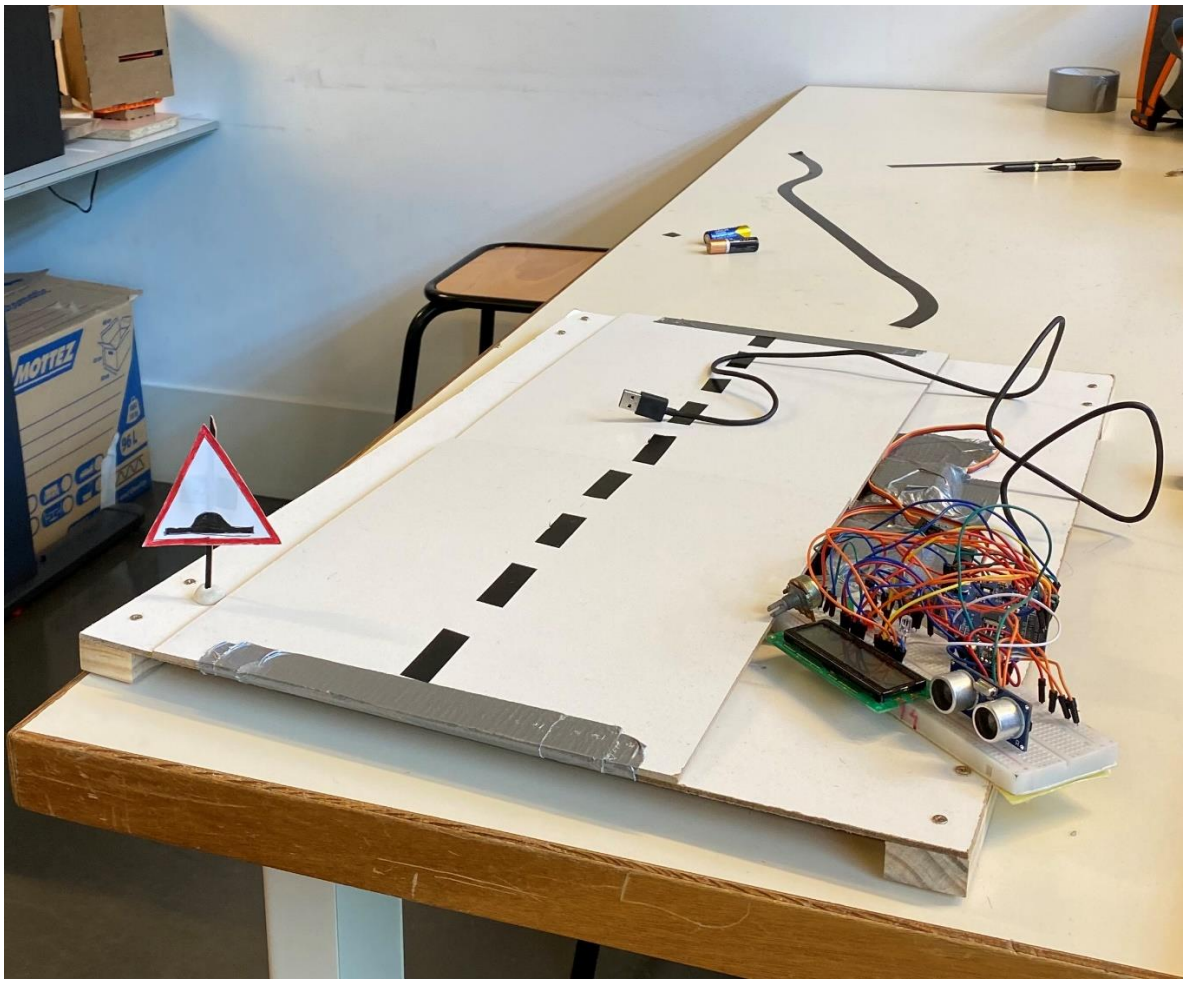
Le module que nous avons utilisé pour le projet le kit infrarouge GT017, composé d'une télécommande infrarouge de 38kHz. La télécommande dispose de 17 boutons avec 4 boutons directionnels dont la flèche du haut et celle du bas. Ce kit est aussi composé d'un récepteur et d'une led bleu que l'on a utilisé pour le bon fonctionnement du module.

La télécommande infrarouge envoie un code différent pour chaque bouton pressé, mais aussi un code général envoyé pour n'importe quel bouton.

On a donc converti grâce à Arduino le code en base HEXADECIMA ce qui nous a permis d'ériger un tableau avec le code hexadécimal de chaque bouton. Nous avons fait autant de variables que de bouton (avec leur code respectif). Cependant on s'est rendu compte que le temps de latence avant de recevoir le code d'un bouton était beaucoup trop long et que cela ne collait pas avec la fonction de base qui était d'envoyer un message via la télécommande le plus rapidement possible pour ne pas faire monter le dos d'âne.

On a donc utilisé le code hexadécimal fixe envoyer pour chaque bouton. Si le récepteur reçois un quelconque message provenant de la télécommande infrarouge, on bloque tout le programme pendant 15 secondes qui permet donc de laisser le dos d'âne à plat. Le radar arrête donc de prendre la vitesse en compte et bloque l'affichage sur l'écran LCD.

Voici une photo du projet final :

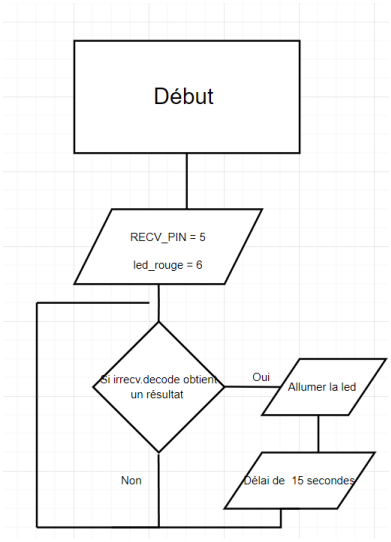


Lien de la démonstration vidéo : https://youtu.be/PnD7mtUFSY8

III- L'algorithme des fonctions

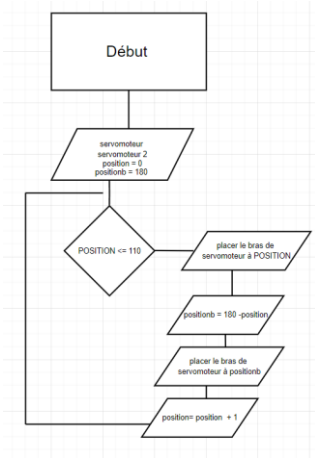
Le code est très long et comprend plusieurs fonctions, c'est pour cela que nous avons découpé en plusieurs algorithme :

La télécommande Infra rouge :



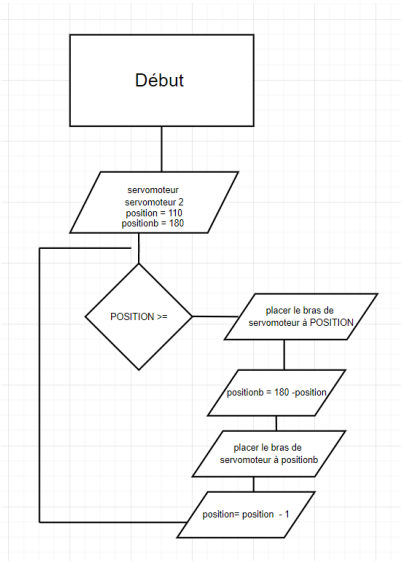
Ici on a une idée très simple du programme. On fait d'abord appeler à la bibliothèque Ir Remote. On initialise une variable (entier) RECV_PIN. Si on reçoit un quelconque signal, on allume la led rouge et on met en pause tout le programme pendant 15 secondes. Si aucun signal n'est reçu, on continu de chercher.

Montée des bras du dos d'âne :



Ici on utilise deux servomoteurs, initialisé en entrée, puis on utilise un for qui permet d'avoir une montée des bras avec un pas de 1. On commence à la position initiale 0 et on monte petit à petit. Le deuxième servomoteur doit tourner de l'autre sens mais avec la même position initiale et d'arriver (ici 110), d'où le 180 – position. On appellera cette fonction « dos_se_leve ».

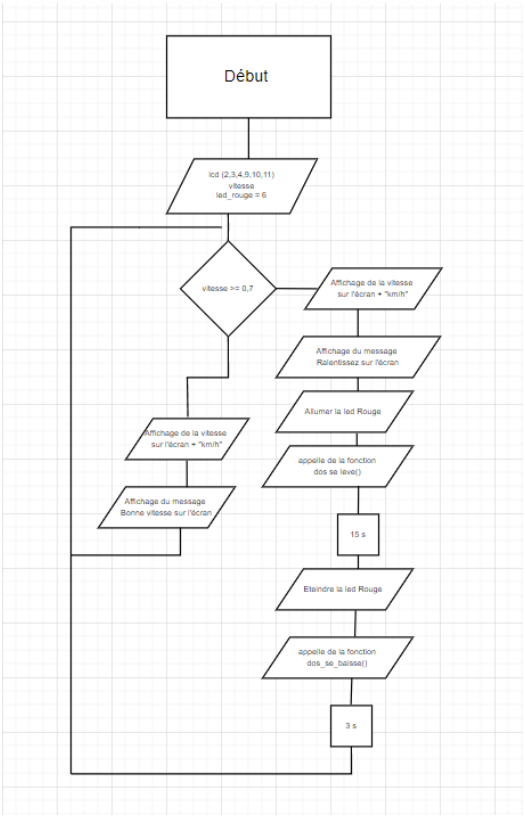
Descente des bras du dos d'âne :



Descente des bras du dos d'âne :

Ici c'est pratiquement le même code, mais cette fois-ci on part de la position haute (110) et on descend avec un pas de 1 jusqu'à atteindre la position finale 0. On appellera cette fonction « dos_se_baisse ».

Radar et écran LCD :



Nous avons fait un code plus complexe. Nous avons calculé la vitesse dans l'initialisation grâce à une fonction distance que l'on appelle deux fois. On soustrait la première distance et on divise le résultat par un delta T (temps) donné (500 micro secondes).

La vitesse est calculée indéfiniment et si elle est supérieure à 0,7 on affiche la vitesse sur la première ligne et le message « Ralentissez » sur la deuxième ligne de l'écran LCD.

On allume la led rouge. On appelle ensuite la fonction dos_se_leve (montré juste avant), et on instaure un temps de 15 secondes pour laisser le temps à la voiture de franchir le dos d'âne. Ensuite on éteint la led rouge et on appelle la fonction dos_se_baisse et on instaure un délai de 3 secondes seulement.

Si la vitesse est inférieure à 0,7, le dos d'âne reste à la position basse (0) et affiche seulement la vitesse sur la première ligne et le message suivant sur la deuxième ligne « Bonne vitesse »

IV- Planning

	Adam	**Aubin**
SEANCE 1	Recherche des dimensions et matériaux du dos d'âne	Initiation au capteur de vitesse Doppler
SEANCE 2	Dimensions et réflexion au mécanisme	Capteur laser avec télécommande
SEANCE 3	Recherche du mécanisme et moteurs	Ecriture code pour capteur
SEANCE 4	Oral intermédiaire, début code moteur	Début de recherche pour le radar ultrasons
SEANCE 5	Début de la maquette au FabLab	Avancée code et branchements
SEANCE 6	Code mettant en relation radar et moteur	Finalisation du radar à ultrason
SEANCE 7	Assemblage maquette et placement moteurs	Synchronisation des moteurs et code
SEANCE 8	Dernières modifications et vérifications	Dernières modifications et calibrage du radar

V- Difficultés rencontrées

En ce qui concerne les difficultés rencontrées, c'est principalement au niveau de la gestion du temps et le respect des délais imposés.

Au niveau du mécanisme d'escamotage, nous avions eu du mal initialement à concrétiser nos idées, principalement parce que nous voulions faire des choses trop compliquées mais une fois lancés ces problèmes se sont résolus en utilisant un système simplifié.

Concernant le radar à vitesse, nous avions été confrontés à de nombreux bugs au niveau de la vitesse mesurée souvent erronée car la sonde était très sensible à quelque mouvement dans les alentours. Pour remédier à cela, nous avons passé beaucoup de temps à calibrer le radar en faisant de nombreux test pour déterminer les conditions idéales pour une exécution réussie.

Au niveau du code Arduino, il était parfois difficile de comprendre de nous même ce qu'il n'allait pas et nous nous retrouvions souvent bloqués avant de demander de l'aide aux professeurs. Heureusement, grâce à leur aide et celui de nos camarades et internet, nous avions réussi à écrire un code correct.

VI- Améliorations envisageables

Nombreuses sont les éventuelles améliorations ou aspects que nous aurions aimé différemment. D'une part, lors de l'inclinaison un léger espace se forme entre chacune des faces, ce qui n'est pas dérangeant à cette échelle mais pourrait l'être pour des vraies voitures. Il faudrait songer à un moyen de combler l'écart soit avec un système de

coulissage des faces pour se rejoindre en un sommet ou bien l'ajout d'une pièce permettant de relier les deux parties.

En ce qui concerne le radar, les ondes utilisées sont peu pratiques car elles entrainent trop d'erreurs de mesure. Il faudrait utiliser un radar avec des laser, plus fiable et mieux placé pour pouvoir être qualifié d'optimiser. Qui plus est, intégrer un système de flash afin de verbaliser les véhicules en excès de vitesse est également une amélioration possible.

D'autre part, concernant le système pour les véhicules prioritaires, la télécommande s'avère souvent peu pratique, surtout en cas d'urgence ou de perte. Nous aurions aimé créer un système de lecteur de plaque capable de rapidement distinguer l'arrivée d'un véhicule autorisé à circuler au-dessus de la vitesse maximale.

Enfin, l'objectif final sera d'élargir ce projet à taille réelle afin d'avoir un dos d'âne intelligent fonctionnel pour des voitures réelles.

VII- Conclusion et perspectives

Pour conclure, le projet nous a beaucoup apporté notamment au niveau de l'organisation du travail, l'aspect technique concernant l'électronique avec la complexité des branchements et leur précision, avec une seule erreur le projet ne fonctionne plus. Cela nous a permis d'améliorer nos capacités à programmer en Arduino en manipulant beaucoup de fonctions et de code. Nous avons pu réaliser un dos d'âne en miniature et un radar en miniature fonctionnels.

Cependant avec l'expérience acquise, et avec 9 semaines de plus, je pense que nous pourrions nettement améliorer le radar, avec un plus grand écran LCD (ressemblant au radar pédagogique), permettant une meilleure visibilité et plus de pixels afin d'avoir plus d'éléments sur l'écran. D'autre part, il serait intéressant de rendre le mécanisme d'inclinaison plus rapide et plus solide afin de résister à des charges plus lourdes.

VIII- Bibliographie

Tope inteligente : https://youtu.be/7jnKqTnYysQ

GO TRONIC:

- https://www.gotronic.fr/art-module-de-detection-us-hc-sr04-20912.htm
- https://www.gotronic.fr/art-telecommande-ir-gt017-29003.htm
- https://www.gotronic.fr/cat-afficheurs-lcd-413.htm
- https://www.gotronic.fr/cat-servomoteurs-1084.htm

Mon coyote : https://www.moncoyote.com/fr/radars-routiers.html

Plaisir Arduino : https://plaisirarduino.fr/les-fonctions/

Conrad: https://www.conrad.fr/c/servomoteurs-c19788

Kit-embrayage : https://www.kit-embrayage.fr/blog/servomoteur-fonctionnement-programmation-signes-dusure-et-prix/