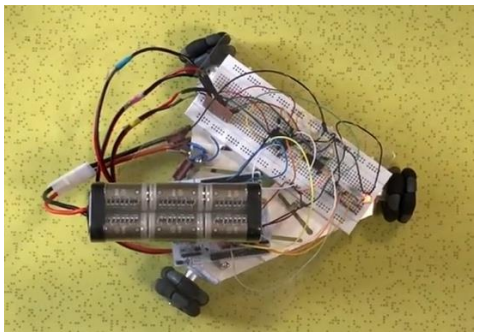
Rapport technique : Robot omnidirectionnel



Introduction :

L'objectif du projet est de réaliser un robot fonctionnant avec 3 roues alimentées chacune par un moteur à courant continue. Son nom, omnidirectionnel explicite son mode de déplacement : le robot peut se déplacer de la même façon dans toutes les directions permises par l'orientation de ses roues. Le déplacement dans une direction est assuré par deux roues uniquement. Nous voulions de plus que ce robot soit télécommandé à distance et qu'il puisse s'arrêter automatiquement quand il rencontre un objet. Nous avions également pour objectif bonus de rendre le robot autonome en utilisant la fonction d'arrêt automatique et en le faisant choisir une direction aléatoirement. De même nous souhaitions augmenter le nombre de directions possibles en utilisant les 3 roues en même temps. Pour réaliser ces fonctionnalités nous avons utilisés des cartes Nucléo comme microcontrôleurs, une pour le robot et une pour les moteurs. Dans le but de contrôler facilement la vitesse des moteurs nous avons utilisés les sorties PWM de la carte qui envoie une tension en créneaux et dont on gère le rapport cyclique. Pour faire communiquer ces cartes nous avons utilisés des modules radiofréquences.

Les différents éléments :

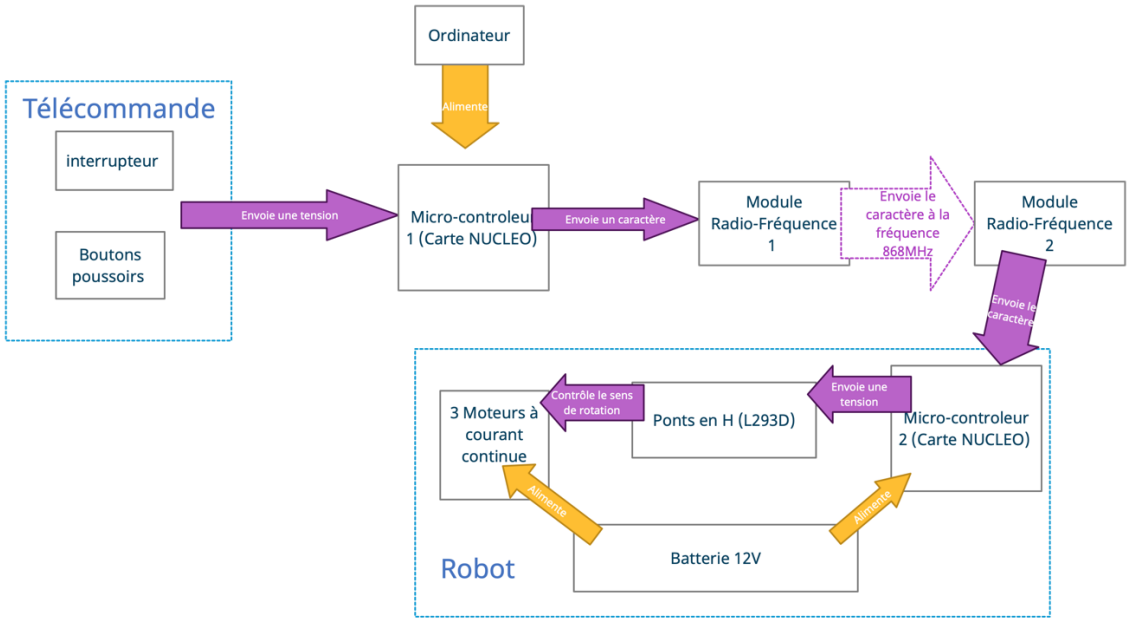
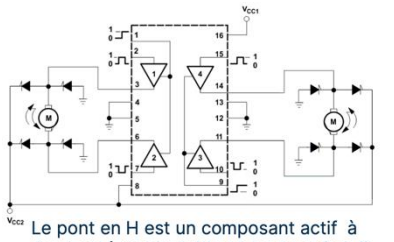


Figure 1 : Schéma bloc regroupant tous les éléments du projet

Tout d'abord, nous nous sommes intéressés aux moteurs à courant continue. A priori ils fonctionnent dans un sens unique, en fonction du sens dans lequel ils sont branchés. Pour pallier cela et permettre de choisir facilement le sens de rotation des roues pour obtenir 6 directions au total, avec à chaque fois 2 roues en fonctionnement, nous avons utilisés des ponts en H qui permettent un branchement des moteurs « dans les deux sens ». Nous avons ainsi pu, utilisé chaque moteur dans le sens que nous voulions en utilisant simplement 6 sorties PWM.



Le pont en H est un composant actif à deux coté manipulable en tout ou rien. Il permet de bloquer la tension dans un sens de rotation.

Pour alimenter la carte nucléo du robot ainsi que les moteurs, alimentés en 12V, nous avons utilisé une batterie 12V.
Pour réaliser la télécommande nous avons utilisé des boutons poussoirs, reliés à la carte nucléo, la pression sur un bouton déclenche l'envoie d'un caractère à la seconde carte nucléo, qui, recevant ce caractère fait tourner les moteurs en conséquence. Elle est composée de 3 boutons de directions, 2 boutons de rotations servant à se déplacer plus facilement si besoin, un bouton pour augmenter le rapport cyclique, et donc la vitesse des moteurs, un bouton pour réduire la vitesse des moteurs, et un interrupteur permettant d'inverser le sens de rotation des moteurs et donc obtenir les 3 directions opposées aux 3 premières.
Les modules radio-fréquence envoie un caractère unique avec une fréquence de porteuse de 868MHz.

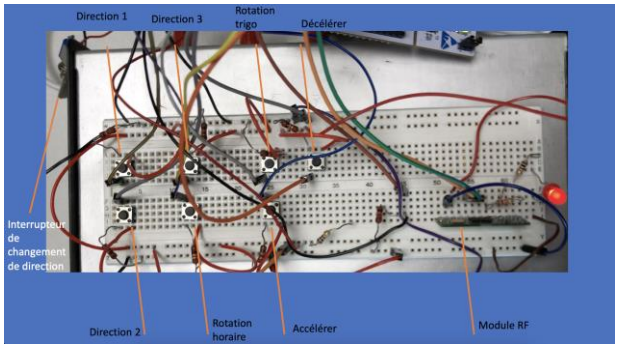


Figure 2 : Télécommande réelle

Schémas électriques :

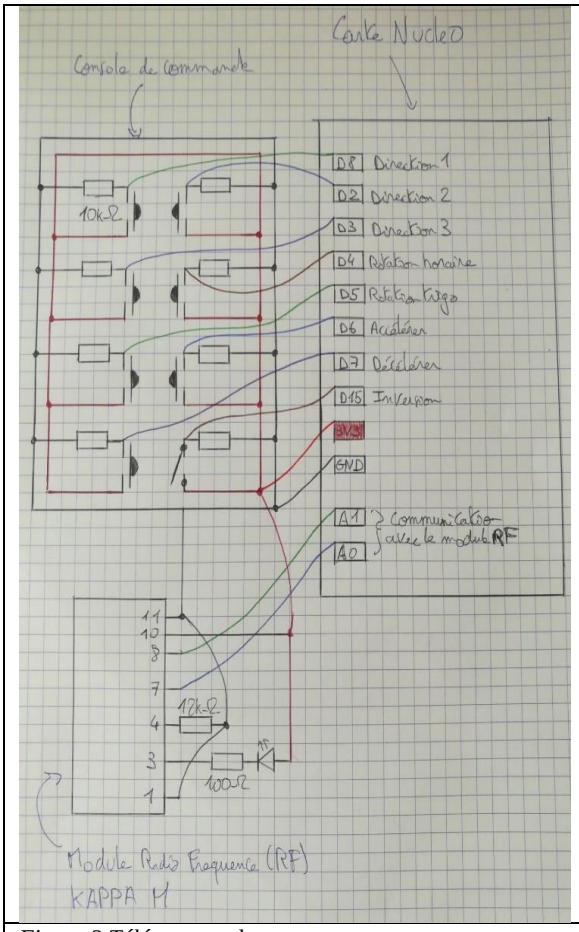


Figure 3 Télécommande

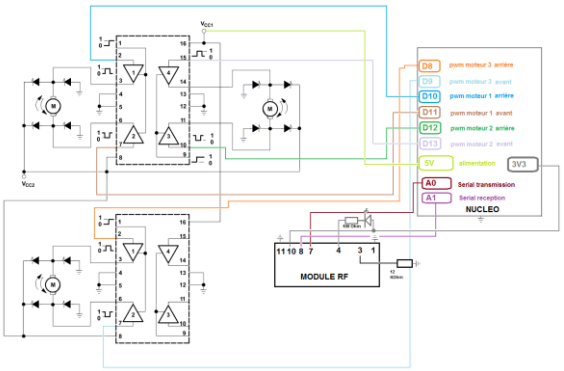


Figure 4 Robot

Algorithmes :

Algorithme type de la télécommande :
Les commandes de base sont les rotations, et la gestion du rapport cyclique.
L'idée est d'envoyé un caractère qui correspond à la pression sur un bouton de la télécommande.

On définit les broches
Tant que l'interrupteur est dans la position 0
Pression sur un bouton poussoir relié à une broche de la carte nucléo
Envoie un caractère via le module RF correspondant aux commandes de base et aux direction 1,2,3
Else (interrupteur position 1)
Pression sur un bouton poussoir relié à une broche de la carte nucléo
Envoie un caractère via le module RF correspondant aux commandes de base et aux direction 4,5,6

Algorithme type du robot :
Pour gérer le sens de direction des moteurs, nous utilisons 6 sorties PWM reliées aux ponts en H. Une sortie PWM correspond à un sens de rotation d'un moteur.

On définit les broches

On définit la variable rc : rapport cyclique avec une valeur de 0.8
Tant que qu'il y a une réception d'un caractère par le module RF
Les sorties PWM sont fixées avec un rapport cyclique de 0
On met dans la variable mode le caractère reçu
Si mode = 1
Deux sorties PWM sont fixées avec rc, les autres à 0
Else if mode = 2 (resp 3,4,5,6)
Deux sorties autres PWM sont fixées avec rc, les autres à 0
Else if mode = + (resp -)
Augmente (resp diminue) la valeur de rc
Else if mode = T (resp H)
Trois sortie PWM sont fixées avec rc, les autres à 0 //le but est d'obtenir une rotation dans le sens trigo ou horaire.

Amélioration de des possibilités de mouvement du robot (non testée par manque de temps) :

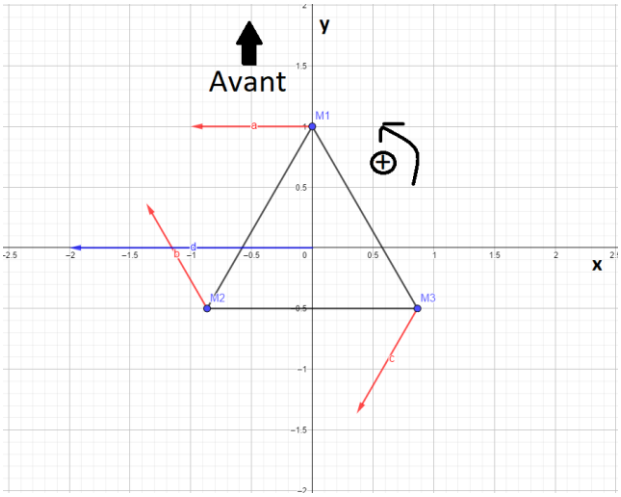


Figure 5 Schéma d'orientation du robot

Le robot a trois degrés de liberté : deux de translation (selon x et y) et un de rotation (selon l'axe sortant de la feuille). On peut former à partir de ça une base orthonormale des degrés de liberté que l'on représente par le **vecteur (X Y R)**. L'idée est alors d'exprimer le rapport cyclique de chaque moteur en fonction de ce vecteur (calculs de projection):

$$\begin{pmatrix} M_1 \\ M_2 \\ M_3 \end{pmatrix} = \begin{pmatrix} -0.81 & 0 & 0.58 \\ 0.41 & -0.71 & 0.58 \\ 0.41 & 0.71 & 0.58 \end{pmatrix} . \begin{pmatrix} X \\ Y \\ R \end{pmatrix}$$

M_x : « Moteur n°x »
Le signe indique le sens de rotation

Exemple de commande : avancer vers l'avant.

$$\begin{pmatrix} M_1 \\ M_2 \\ M_3 \end{pmatrix} = \begin{pmatrix} -0.81 & 0 & 0.58 \\ 0.41 & -0.71 & 0.58 \\ 0.41 & 0.71 & 0.58 \end{pmatrix} . \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ -0.71 \\ 0.71 \end{pmatrix}$$

NB : La matrice utilisée est orthogonale de sorte que le robot est une vitesse constante quelque que soit la direction dans laquelle il se déplace.

Le prototype a d'abord été construit avec une unique carte nucléo, au lieu d'envoyer un caractère, la pression sur un bouton poussoir réalisait direction les actions de la carte nucléo du robot. Le passage à deux cartes a simplement été une répartition du code entre les deux cartes avec une adaptation liée à la communication entre les deux cartes.

Valider le système final :

Au cours du montage du prototype il fallait constamment vérifier chaque élément séparément pour mettre d'identifier plus facilement les problèmes sur le projet complet.

La télécommande :

Pour vérifier le bon fonctionnement de la télécommande il fallait procéder à deux niveaux, au niveau de l'émetteur et du récepteur. Pour cela nous avons connecté les deux cartes nucléo à l'ordinateur dans le but d'utiliser Teraterm pour avoir un retour sur les boucles dans lesquels le microcontrôleur rentrait. Nous avons ainsi pu vérifier que par pression sur un bouton poussoir, la carte rentrait dans la bonne boucle et donc devait en théorie envoyer le bon caractère. Nous validons donc le fonctionnement de la carte et des boutons/interrupteur en eux-mêmes.
Pour vérifier cela on regarde sur la deuxième carte nucléo si celle-ci rentre dans la boucle correspondant au caractère envoyé. C'est le cas, les deux cartes communiquent correctement entre elles, la seconde carte nucléo peut alors commander les moteurs de la façon qui correspond au bouton sur lequel l'opérateur appuie.

Les moteurs :

Pour tester les moteurs nous avons simplement essayé de brancher directement aux bornes du moteurs une alimentation 12V. Nous pouvons ainsi nous assurer que les moteurs fonctionnent dans les deux sens

Le pont en H :

Une partie importante a été de tester les ponts en H. Une fois le câblage réalisé nous avons utilisé un oscilloscope pour contrôler chaque sortie du pont en H avec et sans pression sur les boutons poussoirs. Nous avons eu des difficultés avec ces tests car il y avait une tension délivrée en continue aux moteurs, après avoir revérifié les branchements et changer de pont en H nous sommes arrivés à la conclusion qu'il s'agissait de la carte nucléo.

La carte nucléo :

Après avoir déterminé que la carte nucléo posait problème, nous nous sommes rendu compte qu'elle envoyait un signal de ses sorties PWM alors qu'elle n'était pas dans une boucle de mise en marche des moteurs (cela a été vérifié par Teraterm). Après de nombreux essais de changement de broches le problème s'est réglé de lui-même. Nous n'avons malheureusement pas compris ce qui posait problème cependant comme ce problème en particulier a bloqué tout le projet pendant environ 8h, nous étions heureux qu'il soit résolu.

Finalement :

Chaque élément fonctionnant correctement il a été facile de tester le robot dans son entièreté. Celui-ci a fonctionné correctement, chaque pression sur un des boutons poussoirs agissaient correctement sur les moteurs. La portée de la télécommande est d'environ 10m. Cependant la perte de temps liée à la carte nucléo a entrainé une révision des objectifs. Les objectifs bonus n'ont bien-sûr pas été traité mais nous n'avons également pas pu nous intéresser au fonctionnement et à l'intégration au robot des capteurs d'obstacle.

Planning (et commentaire de rétroplanning) :

Nous nous organiserons en deux équipes deux 2 : Une au cablâge/conception matérielle du robot (Alexandre et Batuhan)
Une à la programmation et création de la télécommande (Gautier et Grégoire)

1ere séance : découverte du projet, des moteurs, pont en H

2e séance : Écriture du cahier des charges/schéma bloc. Câblage sommaire des trois moteurs et de tous les boutons direction sur le robot avec une seule carte Nucléo.

3e séance : -Câbler les trois moteurs, dans un premier temps 3 boutons: un pour chaque sens de chaque moteur et valider le bon fonctionnement.
-Piloter chacun des moteurs indépendamment avec une carte nucléo. (à la fin de cette séance nous avions presque fini le programme de la séance 4)

4e séance : - Câblage d'une télécommande fonctionnelle comme indiqué plus haut, reliée par un fil au robot.
-Ecriture du code traduisant les les commandes en tensions pour les moteurs.
(Après quelques réglages dans le code nous avons essayer de faire communiquer les deux cartes nucléo, ce que nous avons fait mais en filaire tout d'abord)

5e séance : -Séparer la télécommande du robot; ie faire communiquer 2 cartes nucléo à distance (Durant cette séance nous avons pris en main les modules RF pour faire communiquer à distance les deux cartes, cependant nous avons eu un problème que l'on pensait venir d'un des moteurs, il tournait sans raison)

6e séance : -câblage des capteurs de distance (Durant cette séance nous avons recâbler totalement le robot pensant que le problème du moteur pouvait venir d'un mauvais câblage ou du pont en H, nous avons alors procédé aux tests expliqués plus haut sur ce composé)
-code pour arrêter le robot

7e séance : Mise en commun de tous les éléments pour un test de toutes les fonctionnalités du robot (-bonus si nous avons du temps). (Durant cette séance nous avons dû recâbler entièrement le robot que nous avions retrouvé décâbler, nous avions le même problème de carte nucléo qu'y s'est résolu en fin de séance. C'est là que la majorité des tests de la télécommande ont été réalisé).

ANNEXES :

CODE POUR L'ÉMETTEUR

//L'émetteur se contente de tester la pression sur chaque bouton pour envoyer au recepteur un "mode" de fonctionnement du robot.

```
#include "mbed.h
DigitalIn direction1(A0);
DigitalIn direction2(D2);
DigitalIn direction3(D3);
DigitalIn rotationhor(D4);
DigitalIn rotationtri(D5);
DigitalIn vitesseplus(D6);
DigitalIn vitessemoins(D7);
DigitalIn sens(D15); // important que ce soit un interrupteur physique
DigitalOut onoff(D14);

Serial emission(A1, A0);   //Connecté au module RF
Serial pc(USBTX, USBBX);

int main() {
        pc.baud(115200);
        emission.baud(19200);
        onoff = 1;

        while (1)
        {
        while (sens==1){        // Le code est répété deux fois en deux boucle while gérées par l'interrupteur. La seule chose qui change dans ces deux  boucles est que les directions passent en propulsion et non plus en traction.


        if (direction2==1)
                {
        pc.printf("direction2");  //test pour voir si la pression sur le bouton était détectée, c'est bien le cas
        emission.putc('2');
        }

        else  if (rotationhor==1) {


        emission.putc('H');

        }


        else  if (rotationtri==1) {
```

```
        emission.putc('T');
        }


        else if (vitesseplus==1)
        {
        emission.putc('+');

        }
        else if (vitessemoins==1)
                {emission.putc('-');
        }

        else  if (direction1==1) {

        emission.putc('1');

        }



        else if (direction3==1)

        {emission.putc('3');

 }

        else
        {       emission.putc('S');  //ce S n'est plus réutilisé spécifiquement à la réception puisque le else (qui coupe tous les moteurs) agit dès qu'autre chose que les modes définis est reçu.
                }

        }


        while (sens==0) {       //deuxième boucle qui change les inverses les directions et renvoie donc les modes "4, 5, 6" pour les directions et les mêmes modes pour les autres commandes


        if (direction2==1)
                {
        emission.putc('5');
        }

        else  if (rotationhor==1) {
        pc.printf("rotahor");

        emission.putc('H');

        }
```

```
        else  if (rotationtri==1) {
  pc.printf("rotatri");

        emission.putc('T');
         }

        else if (vitesseplus==1)
        {
        emission.putc('+');



        }
        else if (vitessemoins==1)
                { emission.putc('-');
        }

        else  if (direction1==1) {

        emission.putc('4');


        }

        else if (direction3==1)

        emission.putc('6');



        else {
        emission.putc('S');}

        }
        }
        }
```

CODE POUR LE RECEPTEUR

//On agit ici uniquement sur le rapport cyclique rc et sur les sorties Pwm en lisant les “modes” envoyés par la carte émettrice.

```
#include "mbed.h"

PwmOut moteur1a(D11);
PwmOut moteur1r(D10);
PwmOut moteur2a(D13);
PwmOut moteur2r(D12);
PwmOut moteur3a(D9);
PwmOut moteur3r(D8);
```

```
Serial reception(A0, A1);
Serial pc(USBTX, USBRX);
int main() {


        double rc = 0.5;
        reception.baud(19200);
        pc.baud(115200);
        char mode;

        moteur1a.period_ms(10);
        moteur1r.period_ms(10);
        moteur2a.period_ms(10);
        moteur2r.period_ms(10);
        moteur3a.period_ms(10);
        moteur3r.period_ms(10);

        moteur1a.write(0.0);
        moteur1r.write(0.0);
        moteur2a.write(0.0);
        moteur2r.write(0.0);
        moteur3a.write(0.0);
        moteur3r.write(0.0);

        while (1)
        {  pc.printf("test");  //C'était pour tester si la boucle tournait et elle tourne bien

        while (reception.readable()) {

        mode = reception.getc();
         pc.printf("le mode est repéré");  //je voulais voir si on rentrait dans cette boucle
conditionnelle mais non.


        if (mode=='T')

        {
        moteur1a.write(0);
        moteur1r.write(rc);
        moteur2a.write(0);
        moteur2r.write(rc);
        moteur3a.write(0);
        moteur3r.write(rc);}


        else if (mode=='H')

        {
        moteur1a.write(rc);
        moteur1r.write(0);
        moteur2a.write(rc);
        moteur2r.write(0);
        moteur3a.write(rc);
        moteur3r.write(0);}
```

```
else if (mode=='1')

{
moteur1a.write(0);
moteur1r.write(0);
moteur2a.write(0);
moteur2r.write(rc);
moteur3a.write(rc);
moteur3r.write(0);}

else if (mode=='2')

{
pc.printf("mode 2 est activé");         //test pour voir si il lisait le mode
moteur1a.write(rc);
moteur1r.write(0);
moteur2a.write(0);
moteur2r.write(0);
moteur3a.write(0);
moteur3r.write(rc);}

else  if (mode=='3') {

moteur1a.write(0);
moteur1r.write(rc);
moteur2a.write(rc);
moteur2r.write(0);
moteur3a.write(0);
moteur3r.write(0);

}

else if (mode=='4')

{
moteur1a.write(0);
moteur1r.write(0);
moteur2a.write(rc);
moteur2r.write(0);
moteur3a.write(0);
moteur3r.write(rc);}

else if (mode=='5')

{
moteur1a.write(0);
moteur1r.write(rc);
moteur2a.write(0);
moteur2r.write(0);
moteur3a.write(rc);
moteur3r.write(0);}
```

```
        else if (mode=='6')

        {
        moteur1a.write(rc);
        moteur1r.write(0);
        moteur2a.write(0);
        moteur2r.write(rc);
        moteur3a.write(0);
        moteur3r.write(0);}



                else if (mode=='+')
        {

        if (rc+0.05>1)
                rc=1.0;

        else
                rc=rc+0.05;

        }
        else if (mode=='-')
                {
        if (rc-0.05<0.30)
                rc=0.30;
        else rc=rc-0.05;
        }

        else {                  //Si on appuie sur aucun bouton les moteurs sont à 0

                moteur1a.write(0);
                moteur1r.write(0);
                moteur2a.write(0);
                moteur2r.write(0);
                moteur3a.write(0);
                moteur3r.write(0);
                }


        }
  }
}
```