PROJET DE FIN D'ÉTUDES

FAUTEUIL ROULANTE ELÉCTRIQUE

Félix Candela Fernández
Tuteurs :
Romuald Stock
Georges Stentz

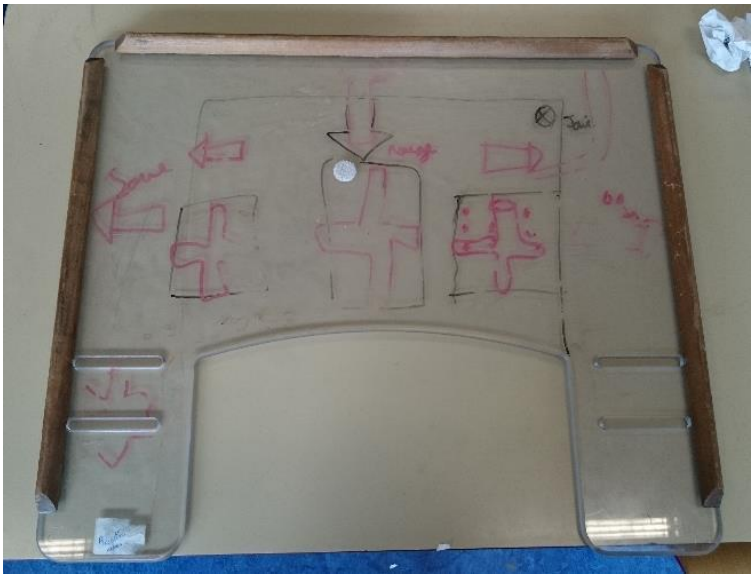
Table des matières

Objectif du projet

- Remplacer le Joystick d'un fauteuil roulant électrique pour un outil qui rend possible le déplacement pour des enfants polyhandicapés.
- Développer l'interface qui aide aux enfants dans l'apprentissage.

Développement du projet

Suite à la première visite au centre Handas nous avons convenu de faire l'outil dans une planche en plastique similaire à celles déjà utilisées.



L'idée était de faire des trous dans la planche pour y mettre les capteurs. Due au manque de place, nous avons convenu de faire la marche arrière en déclenchant les capteurs de droite et gauche au même temps. **Dans les prochains outils il faudra discuter cette idée avec les ergothérapeutes du centre car c'est très difficile pour les enfants d'apprendre ce mouvement.**

Pour l'interface nous avions pensé de faire un trou avec la forme d'une flèche, le couvrir avec une plaque en plastique translucide du couleur et mettre des LEDs au-dessous pour l'allumer. Avec cette idée, on aurait besoin de moins LEDs pour allumer, se traduisant dans une mineure consommation.

Finalement nous avons décidé de ne faire que les trous nécessaires pour les LEDs et faire la forme d'une flèche avec un papier de couleur translucide et le mettre sur la planche.

Sur la photo on voit les emplacements pour les capteurs qui auraient été dans une croix pour les réglages de position. Finalement, on a rejeté cette idée car, dans la pratique, la main couvrait presque toute la surface où le capteur allait se déplacer.

Après le premier essai avec le prototype, on a vu que l'enfant, au lieu de mettre la main sur le capteur, mettait tout de suite la main sur la flèche. Alors, nous avons dû modifier la position des capteurs et les mettre juste à côté des flèches.

Pour les capteurs de direction, on les a mis sous les flèches, comme ça, quand l'enfant allait vers la flèche, il passait la main sur le capteur. La flèche d'avancer nous l'avons dessiné dans le capteur. Pour que les LEDs puissent se voir, on les a mis dans la pointe de la flèche dans le support final. Puisque nous avons vu dans les différents essais que c'était facile que les fils se débranchait. Nous avons dû ajouter un couvercle sous la plaque.

Fonctionnement des composants

Capteur d'ultrasons HC-SR04

Le capteur qui a été utilisé dans notre projet est le capteur d'ultrasons HC – SR04.

Nous l'avons choisi pour les suivantes raisons :

- Il peut être facilement utilisé avec Arduino, la carte que nous allons utiliser.
- Il a un prix réduit, ce qui faciliterai fabriquer dans une échelle plus grande.
- Il a une petite taille, qui nous permet de le placer en plusieurs endroits.

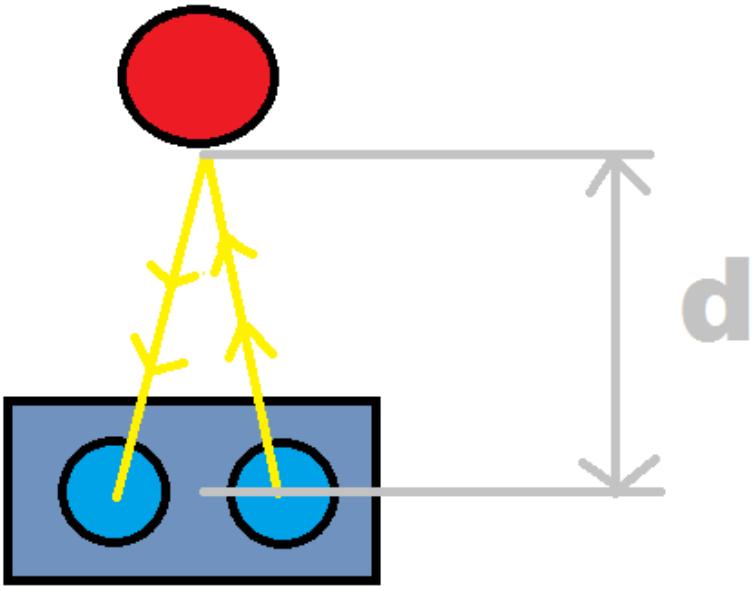
C'est un capteur qui peut détecter un objet dans une distance entre 3 cm et 3 m. Les objets détectés doivent être dans un cône de 15º partant du cylindre. Il envoie une signal d'ultrason avec le cylindre que l'on peut voir à droite, qui s'appelle Trig. Si cette signal trouve un obstacle, elle se réfléchie et arrive à l'autre cylindre, Echo. Avec le temps qui prend le signal pour aller et retourner, on peut savoir la distance à laquelle se trouve l'objet.

Il a quatre connexions :

- En partant de gauche la connexion Vcc qui doit être connecté à la sortie de 5V de notre carte.
- Puis le Trig. Elle doit être connecté dans un PIN défini comme OUTPUT. Cette sortie va recevoir un pulse de 10 µs qui fera fonctionner le capteur.
- Echo : Elle doit aller vers une sortie définie comme INPUT. De cette sortie partira le signal du temps qui a pris pour aller et retourner.

Pour calculer la distance après avoir reçu le signal du capteur (en microsecondes), il ne faut que la transformer parmi la vitesse du son. Comme on voit dans le dessin le temps qu'on reçoit est deux fois la distance par la vitesse du son. Alors la distance sera le temps fois la vitesse du son (on prendra 340 m/s) divisé par deux. Pour montrer la distance en centimètres la formule finale sera :

$$d = 0.017 * t$$



Carte Seeeduino Lotus

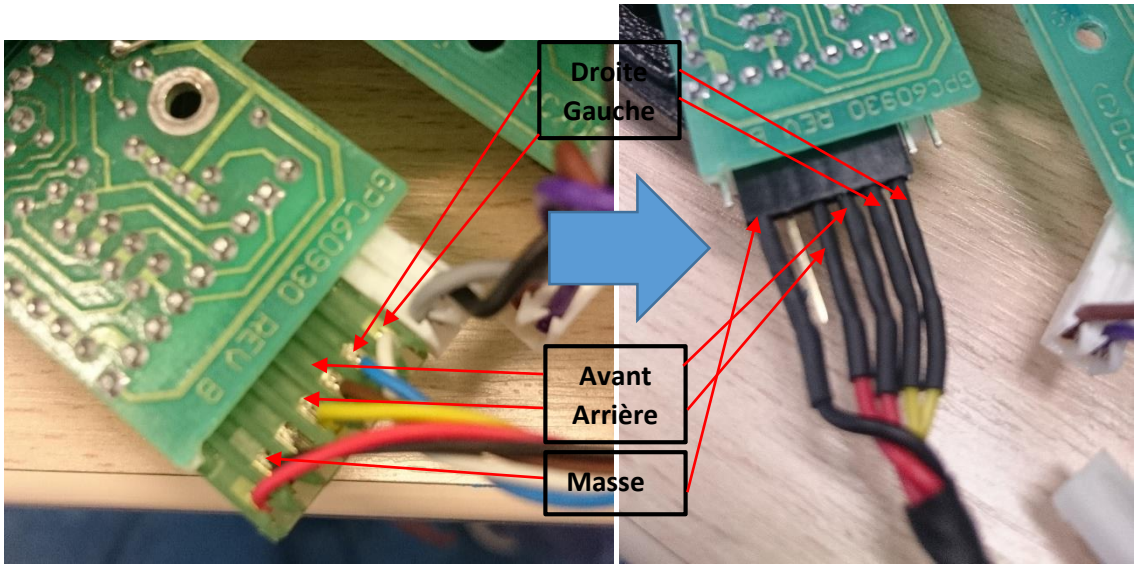
La carte qui aura la fonction de processeur sera une Seeeduino Lotus. Elle est équivalente à la carte Arduino UNO mais elle possède des prises pour rendre le branchement des capteurs plus facile.

Avec cette sorte de carte, on peut programmer dans un langage spécial des instructions et les insérer dans la carte. En fonction des gadgets qui soient connectés à la carte nous pouvons l'utiliser avec beaucoup de buts.

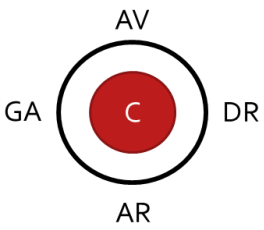
C'est qui va faire la carte c'est d'envoyer une impulsion qui fait fonctionner le capteur, et après transformer l'impulsion reçue du capteur en distance comme nous avons déjà vu. En fonction de la valeur de cette distance, il allumera ou pas les LEDs et modifiera la tension envoyée au fauteuil pour le faire se déplacer.

Commande moteur

Pour la commande du fauteuil nous avons court-circuité la commande arrière en la remplaçant par un signal provenant de la carte.



Pour obtenir les signaux qui faisaient fonctionner le fauteuil, nous avons vu qu'il y avait cinq fils qui envoyaient le signal pour déplacer le fauteuil. Un qui envoyait la courant à terre et quatre autres. Pour analyser l'interaction entre les quatre fils et les mouvements du fauteuil, nous avons remué le joystick et mesuré la tension qu'il y avait dans chaque fil. On présente un graphique avec les mouvements et les signaux correspondants :



	Centre	Avant	Arrière	Gauche	Droite
Jaune	2.5 V	4 V	0.8 V	2.5 V	2.5 V
Brun	2.5 V	1.15 V	4 V	2.5 V	2.5 V
Bleu	2.5 V	2.5 V	2.5 V	0.8 V	4 V
Blanc	2.5 V	2.5 V	2.5 V	4 V	0.8 V

Suite à ces résultats, on peut conclure :

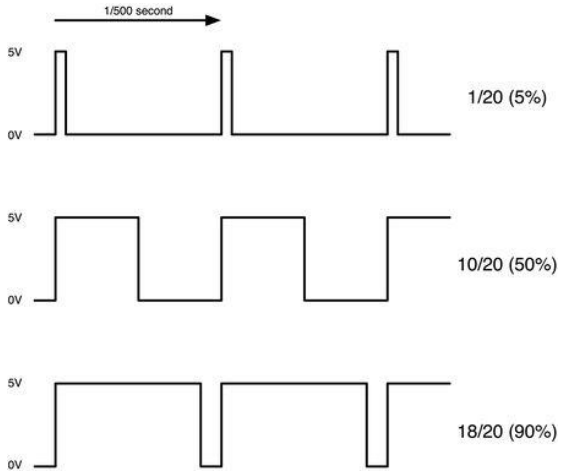
- En repos, la tension de chaque fil est 2.5V.
- Étant donné que si on avance ou l'on recule, la tension des fils bleu et blanc reste constant, ce sont les fils jaune et brun qui servent à avancer et reculer.
- D'un autre côté, après le même raisonnement, on déduit que ce sont les fils bleu et blanc les chargés de tourner à droite et à gauche.
- Dans la couple de fils qui font le mouvement, ils adoptent des valeurs entre 1 et 4 voltes.

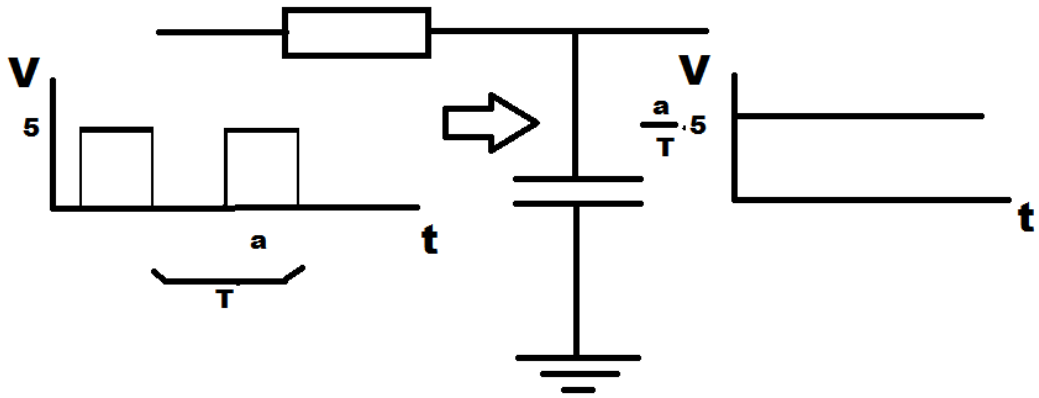
Pour avancer et tourner au même temps, on peut pressentir qu'il y aurait une combinaison des voltages entre les quatre fils.

Dans notre projet, on a décidé de donner toujours les valeurs maximums afin de le faire plus simple. On pourrait augmenter le voltage des fils en fonction de la distance au capteur pour régler la vitesse du fauteuil, néanmoins, peut-être, ça ne serait pas utile pour les enfants qui ont des problèmes pour s'approcher au capteur.

Comme La tension qui envoie la carte Arduino (en notre cas Seeeduino) est PWM, c'est-à-dire Pulse Width Modulation, on doit ajouter un circuit auxiliaire qui transforme cette tension PWM en une tension continue du valeur moyenne. On l'a réussi grâce à un circuit avec un condenseur et une résistance montré dans l'image suivante.

Le signal PWM change ses valeurs entre 0V et 5V pour réussir une tension moyenne entre ces deux valeurs. Le valeur moyenne dépend du temps que la signal est à 5V. On peut régler ce temps avec la programmation mettant un numéro entre 0 (0V) et 255 (5V) en la fonction AnalogicWrite. La fréquence de ce signal est de 500Hz.





Programmation

Dans cette section, on verra la programmation qui mène le fauteuil. Le programme est composé de différentes sections :

D'abord la partie de définition de variables comme sortes de numéros et de sorties/entrées :

```
int ledA = 14, ledG = 15, ledD = 16, ledR = 17, trigG = 4, echoG = 5, trigA = 2, echoA = 3,
trigD = 7, echoD = 8, jaune = 11, brun = 10, bleu = 9, blanc = 6, ledj = 18, pot, h;
long tempsG, distanceG, tempsA, distanceA, tempsD, distanceD;
void setup()
{
//On define les LEDs comme OUTPUTs.
  pinMode (ledA, OUTPUT);
  pinMode (ledG, OUTPUT);
  pinMode (ledD, OUTPUT);
  pinMode (ledR, OUTPUT);
  pinMode (ledj, OUTPUT);
//Les TRIGs des capteurs seront sorties puisqu'on envoie l'impulsion avec eux.
  pinMode (trigG, OUTPUT);
  pinMode (trigA, OUTPUT);
  pinMode (trigD, OUTPUT);
//LEs ECHOs des capteurs seront des entrées pour recevoir l'impulsion.
  pinMode (echoG, INPUT);
  pinMode (echoA, INPUT);
  pinMode (echoD, INPUT);
//Les fils pour la commande.
  pinMode (bleu, OUTPUT);
  pinMode (jaune, OUTPUT);
  pinMode (blanc, OUTPUT);
  pinMode (brun, OUTPUT);
  Serial.begin(9600);
```

La définition des variables doit être dans une seule ligne, de cette la visualisation est plus facile.

Ensuite, nous voyons le fonctionnement du potentiomètre pour régler la distance à laquelle les mains seront détectées. L'entrée analogique du potentiomètre retourne une valeur entre 1024 et 0 en fonction de la position. On profite ça pour varier la distance de détention en modifiant une variable h :

```
void loop()
{
//En fonction de la signal qui retourne du potenciometre le valeur de la distance de detection changera
  pot=analogRead(19);
  if (pot>=800)
  {
    h=3;
  }
  else if (pot>=600)
  {
    h=6;
  }
  else if (pot>=400)
  {
    h=9;
  }
  else if (pot>=200)
  {
    h=12;
  }
  else
  {
    h=15;
  }
  delay (100);
  digitalWrite(ledj, HIGH);
```

Ultérieurement, le capteurs d'ultrason et la conversion du temps de trajet en distance :

```
//Capteur Ultra son droite
  digitalWrite(trigD, LOW);
  delayMicroseconds(5);
  digitalWrite(trigD, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigD, LOW);
  tempsD = pulseIn(echoD, HIGH);
  delayMicroseconds(20);

//Capteur Ultra son gauche
  digitalWrite(trigG, LOW);
  delayMicroseconds(5);
  digitalWrite(trigG, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigG, LOW);
  tempsG = pulseIn(echoG, HIGH);
  delayMicroseconds(20);

//Capteur Ultra son Avant
  digitalWrite(trigA, LOW);
  delayMicroseconds(5);
  digitalWrite(trigA, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigA, LOW);
  tempsA = pulseIn(echoA, HIGH);
  delayMicroseconds(20);

 //On change le temps retourné en distance parmi la vitesse du son.
   distanceG = 0.017 * tempsG;
   distanceA = 0.017 * tempsA;
   distanceD = 0.017 * tempsD;
```

Pour finir, avec les données que nous avons reçues du potentiomètre et des capteurs, nous devons allumer les LEDs et envoyer les tensions correspondantes :

```
//Marche arrière
  if (distanceG < h && distanceD < h && distanceA > h)
  {
//On allume les LEDs necessaires
    digitalWrite(ledG, LOW);
    digitalWrite(ledA, LOW);
    digitalWrite(ledD, LOW);
    digitalWrite(ledR, HIGH);
//On envoie les tension precises
    analogWrite(jaune, 46);
    analogWrite(brun, 201);
    analogWrite(bleu, 130);
    analogWrite(blanc, 130);
  }
//Gauche
  else if (distanceG < h && distanceD > h && distanceA > h)
  {
    digitalWrite(ledG, HIGH);
    digitalWrite(ledA, LOW);
    digitalWrite(ledD, LOW);
    digitalWrite(ledR, LOW);
    analogWrite(jaune, 130);
    analogWrite(brun, 130);
    analogWrite(bleu, 46);
    analogWrite(blanc, 201);
  }
```

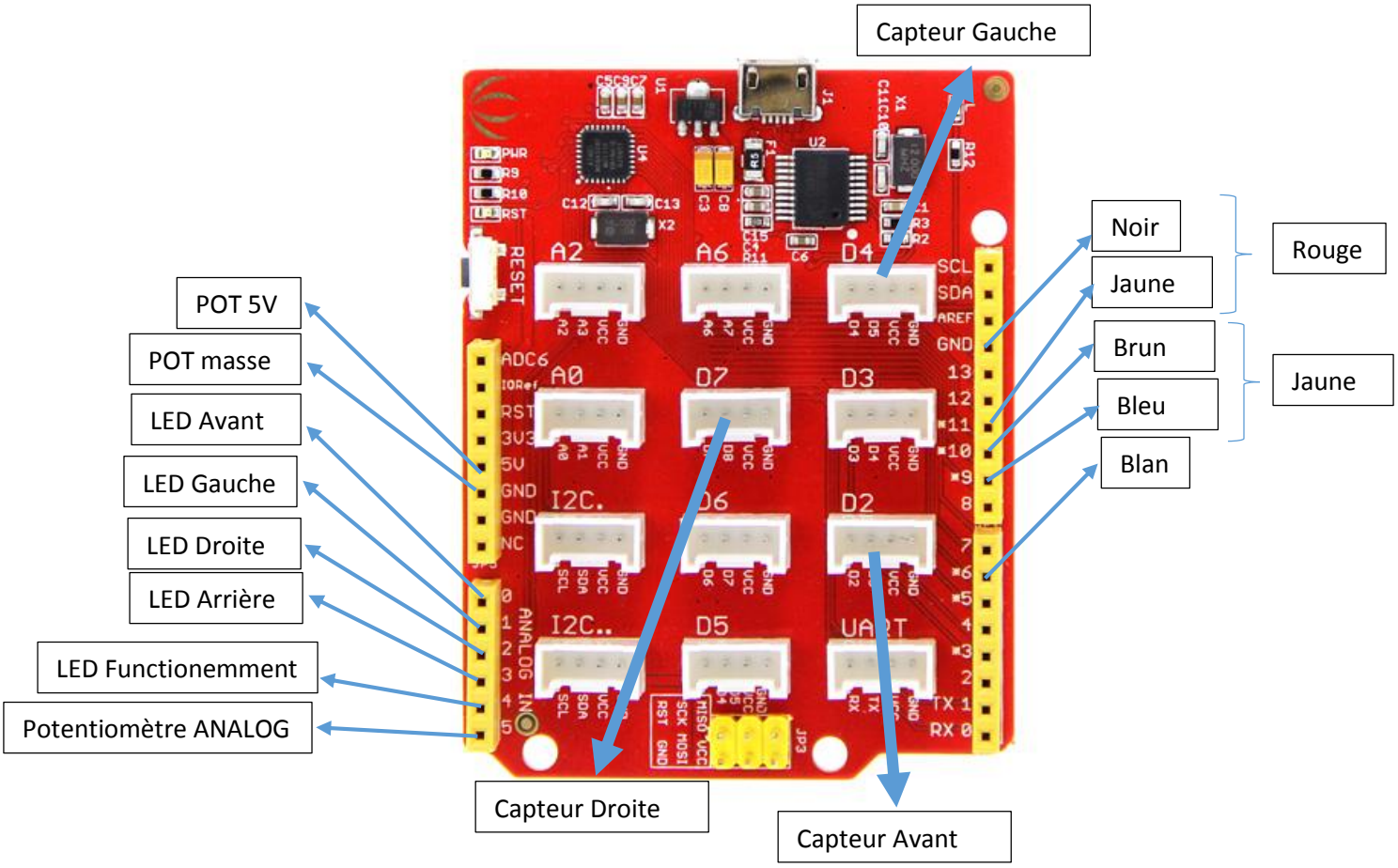
```
//Droite
  else if (distanceG > h && distanceD < h && distanceA > h)
  {
    digitalWrite(ledG, LOW);
    digitalWrite(ledA, LOW);
    digitalWrite(ledD, HIGH);
    digitalWrite(ledR, LOW);
    analogWrite(jaune, 130);
    analogWrite(brun, 130);
    analogWrite(bleu, 201);
    analogWrite(blanc, 46);
  }
//Avant
  else if (distanceA < h && distanceG > h && distanceD > h)
  {
    digitalWrite(ledG, LOW);
    digitalWrite(ledA, HIGH);
    digitalWrite(ledD, LOW);
    digitalWrite(ledR, LOW);
    analogWrite(jaune, 201);
    analogWrite(brun, 46);
    analogWrite(bleu, 130);
    analogWrite(blanc, 130);
  }
```

Pour éviter possibles fautes de programmation et en cas de repos :

```
//Autres cas
  else
  {
    digitalWrite(ledG, LOW);
    digitalWrite(ledA, LOW);
    digitalWrite(ledD, LOW);
    digitalWrite(ledR, LOW);
    analogWrite(jaune, 130);
    analogWrite(brun, 130);
    analogWrite(bleu, 130);
    analogWrite(blanc, 130);
  }
```

Branchements

On présente ici tous les branchements de la carte et ses positions :



Résumé des outils utilisés

Pendant le développement de notre outil, nous avons eu besoin d'utiliser plusieurs outils pour usiner les différentes parties, comme la planche, les supports des capteurs, le câblage, etc. Ensuite les outils utilisés et où on les a utilisés :

Scie :

Utilisée pour :

- ✓ Donner la forme à la planche du prototype.
- ✓ Couper les supports des capteurs de plastique.
- ✓ Couper la place pour mettre le support des capteurs dans le dessin final.

Perceuse :

Utilisée pour :

- ✓ Trous pour les cylindres des capteurs dans la planche.
- ✓ Trous pour la fixation des capteurs aux support en plastique dans le prototype.
- ✓ Trous pour fixer les capteurs à la planche prototype.
- ✓ Trous pour les LEDs.
- ✓ Trous pour les pinces pour fixer la planche au fauteuil.
- ✓ Trous pour fixer la batterie à la planche prototype.

Fer à braser

Utilisé pour souder :

- ✓ Les capteurs aux câblage.
- ✓ Le circuit des LEDs.
- ✓ Le circuit avec les condenseurs pour la commande moteur.

Plieuse

Machine pour plier le plastique. On doit appuyer la partie du plastique que nous voulons plier sur un fil d'acier qui se chauffe. Après un minute (le temps dépend de l'épaisseur de la planche que nous voulons plier) on doit plier manuellement la planche.

Utilisée pour :

- ✓ Plier le couvercle inférieur.