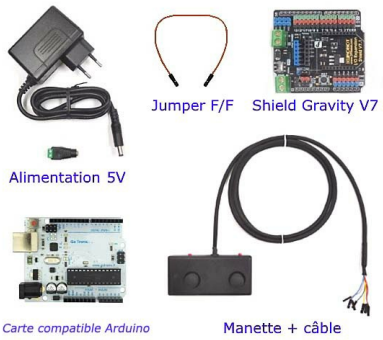


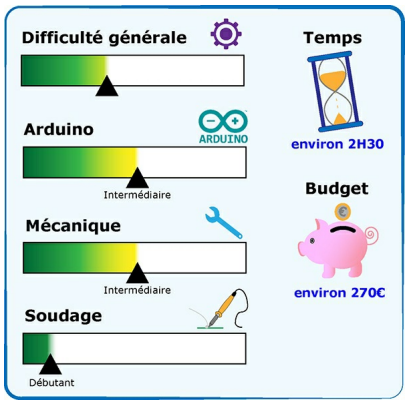
Contrôle d’un bras robotique Joy-It

Aujourd’hui, nous allons contrôler un bras robotique grâce à une carte Arduino et à une manette.

Matériel pour la réalisation :

- Une carte compatible Arduino Uno
- Un kit GoTronic pour JoyIT comprenant :
  - *Un bras robotique JoyIT*
  - *Une manette et son câble (à souder et monter soi-même)*
  - *Un jumper femelle-femelle*
  - *Une carte Shield Gravity V7*
  - *Une alimentation +5 V (câble + bornier + bloc secteur)*





Montage des éléments

Pour construire le robot, aidez-vous du guide de montage. Nous vous conseillons d’utiliser des écrous indesserrables.

Pour monter la manette, veuillez vous référer au mode d’emploi.

Insérez le Shield Gravity sur la carte Arduino Uno.

Étape 1, reliez le **VCC A**nalogique et le **VCC D**igital entre eux avec le Jumper F/F. Ce raccordement est indispensable pour que tout le Shield soit correctement alimenté !

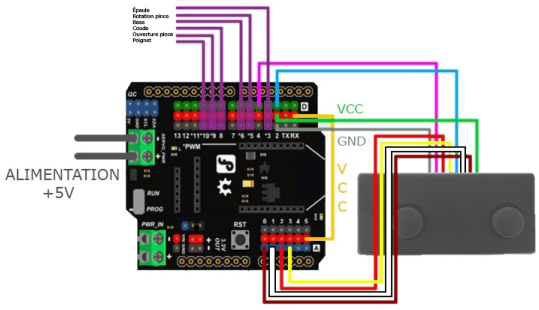
Nous allons concevoir un programme permettant au robot de pouvoir suivre une ligne noire tracée sur un circuit au fond blanc. À chaque fois que le robot détectera qu’il s’éloigne ou qu’il n’est plus sur la ligne noire, il bougera de façon à retrouver la ligne noire. Exemple, s’il s’éloigne à gauche de la ligne, il reviendra automatiquement à droite.

Étape 2, raccordez la manette au Shield Gravity de la façon suivante :

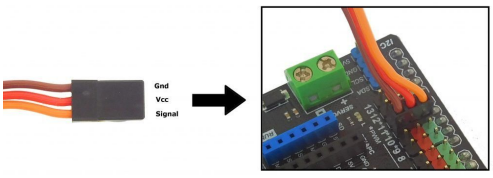
- le fil **GRIS** au **GND**
- le fil **ROSE** en **D4**

- le fil **ROUGE** en **A2**
- le fil **JAUNE** en **A3**

- le fil **BLEU** en **D2**
- le fil **BLANC** en **A1**
- le fil **MARRON** en **A0**
- le fil **VERT** au **VCC**

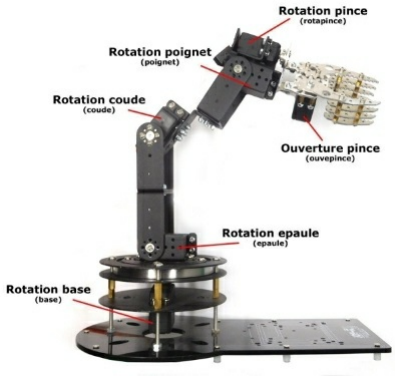


Étape 3, nous allons relier les moteurs du bras robotique au Shield Gravity.
Chaque moteur est composé de 3 fils : un **GND** (marron), un **VCC** (rouge), et une sortie numérique (orange).
Il faudra donc relier de la manière suivante :



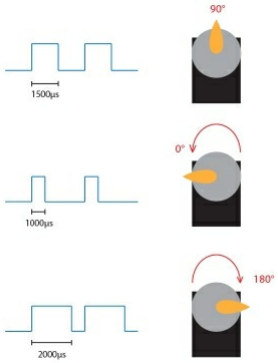
Raccordez :

- le moteur de la **base** en **D6**
- le moteur du **coude** en **D9**
- le moteur du **poignet** en **D5**
- le moteur de la **rotation de la pince** en **D11**
- le moteur de l'**ouverture de la pince** en **D10**



Fonctionnement des servomoteurs :

Le servomoteur est un moteur permettant de tourner sur une course limitée (180°).
Il se pilote par des largeurs d'impulsions. Avec Arduino, on peut contrôler
ces largeurs d'impulsions grâce à l'instruction writeMicroseconds.
La plage 0° – 180° se traduit alors par 1000µs – 2000µs.



Programme Arduino

Nous allons éditer un programme permettant de contrôler les différent moteurs du robot grâce à la manette :

- un mouvement **horizontal** du **Joystick gauche** pilotera la **base**
- un mouvement **vertical** du **Joystick gauche** pilotera le **coude**
- un mouvement **horizontal** du **Joystick droit** pilotera le **poignet**
- un mouvement **vertical** du **Joystick droit** pilotera l'**épaule**
- les **clics** des **Joysticks** piloteront l'**ouverture de la pince**
- les **boutons poussoirs** piloteront la **rotation de la pince**




```
#include <Servo.h>

Servo base; Servo epaule; Servo coude; Servo poignet; Servo rotapince; Servo ouvepince;

int J1, J2, J3, J4;
int BP1, BP2;

int pulse1, pulse2, pulse3, pulse4, pulse5, pulse6;
int pas1, pas2, pas3, pas4;

void setup() {

 base.attach(6); //sortie numérique 6
 epaule.attach(3); //sortie numérique 3
 coude.attach(9); //sortie numérique 9
 poignet.attach(5); //sortie numérique 5
 rotapince.attach(11); //sortie numérique 11
 ouvepince.attach(10); //sortie numérique 10

 pinMode(4, INPUT_PULLUP); //sortie numérique 4
 pinMode(2, INPUT_PULLUP); //sortie numérique 2
 //utilise le VCC et la résistance de pullup de l'Arduino >> évite de rajouter un VCC et une résistance dans le montage

 J1 = J2 = J3 = J4 = 0;
 BP1 = BP2 = 1;

 //initialisation des moteurs en position de départ

 pulse1 = pulse2 = pulse3 = pulse4 = pulse5 = 1500; //90°
 pulse6 = 2000; //180°

}
```

```
void loop() {

 J1 = analogRead(A2); J3 = analogRead(A0); //Joystick gauche mouvement horizontal et vertical
 J2 = analogRead(A3); J4 = analogRead(A1); //Joystick droit mouvement vertical et horizontal
 BP1 = digitalRead(4); BP2 = digitalRead(2); //Bouton poussoir gauche et droit

 base.writeMicroseconds(pulse1);
 epaule.writeMicroseconds(pulse2);
 coude.writeMicroseconds(pulse3);
 poignet.writeMicroseconds(pulse4);
 rotapince.writeMicroseconds(pulse5);
 ouvepince.writeMicroseconds(pulse6);

 //lecture J1
 if (J1 > 1000) {
   pulse6 -= 10;  //action sur la fermeture de la pince
   if (pulse6 < 1650) {
     pulse6 = 1650; /*[2]*/
   }
 }
 else
 { //action sur la base
   pas1 = map(J1, 200, 800, -10, 10); //[0]
   if (pas1 < 5 && pas1 > -5) {} //[1]
   else
   {
     pulse1 += pas1;
     if (pulse1 > 2200) {
       pulse1 = 2200; //[2]
     }
     if (pulse1 < 800) {
       pulse1 = 800; //[3]
     }

   }
 }

 //lecture J2 et action sur le coude
 pas3 = map(J2, 100, 900, -10, 10);
 if (pas3 < 5 && pas3 > -5) {} //[1]
 else
 {
   pulse3 += pas3;
   if (pulse3 > 2100) {
     pulse3 = 2100; //[2]
   }
   if (pulse3 < 900) {
     pulse3 = 900; //[3]
   }
 }

 //lecture J3 et action sur l'épaule
 if (J3 > 1000) {
   pulse6 += 10;  //action sur la l'ouverture de la pince
   if (pulse6 > 2400) {
     pulse6 = 2400; /*[3]*/
   }
 }
 else
 {
   pas2 = map(J3, 200, 800, -10, 10);
   if (pas2 < 5 && pas2 > -5) {} //[1]
   else
   {
     pulse2 -= pas2;
     if (pulse2 > 2200) {
       pulse2 = 2200; //[2]
     }
     if (pulse2 < 1000) {
       pulse2 = 1000; //[3]
     }
   }
 }

 //lecture J4 et action sur le poignet
 pas4 = map(J4, 100, 900, -10, 10);
 if (pas4 < 5 && pas4 > -5) {} //[1]
 else
 {
   pulse4 -= pas4;
   if (pulse4 > 2100) {
     pulse4 = 2100; //[2]
   }
   if (pulse4 < 900) {
     pulse4 = 900; //[3]
   }
 }

 //lecture BP1 et BP2, et action sur la rotation de la pince
 if (BP1 == LOW) {
   pulse5 -= 10;
   if (pulse5 < 600) {
     pulse5 = 600; /*[2]*/
   }
 }
 if (BP2 == LOW) {
   pulse5 += 10;
   if (pulse5 > 2100) {
     pulse5 = 2100; /*[3]*/
   }
 }

 delay(15);

}

/* [0] Un joystick peut varier de 0 à 1023. Suivant la plage qui nous intéresse
  l'instruction "map()" va permettre de faire une mise à l'échelle
  [1] offset
  [2] borne minimale : début de course
  [3] borne maximale : fin de course */
```

Mise en marche

Une fois le programme téléversé, retirez le câble USB de l'Arduino.

Câblez le bloc d'alimentation 5 V au Shield Gravity et branchez la prise au secteur.

Tout est prêt !