



# Composants

## Schémas elec

## Communication arduinos

## Moteurs roues

### Moteur pince

## Moteur élévate

Ultrason

## Camera

LIDAR

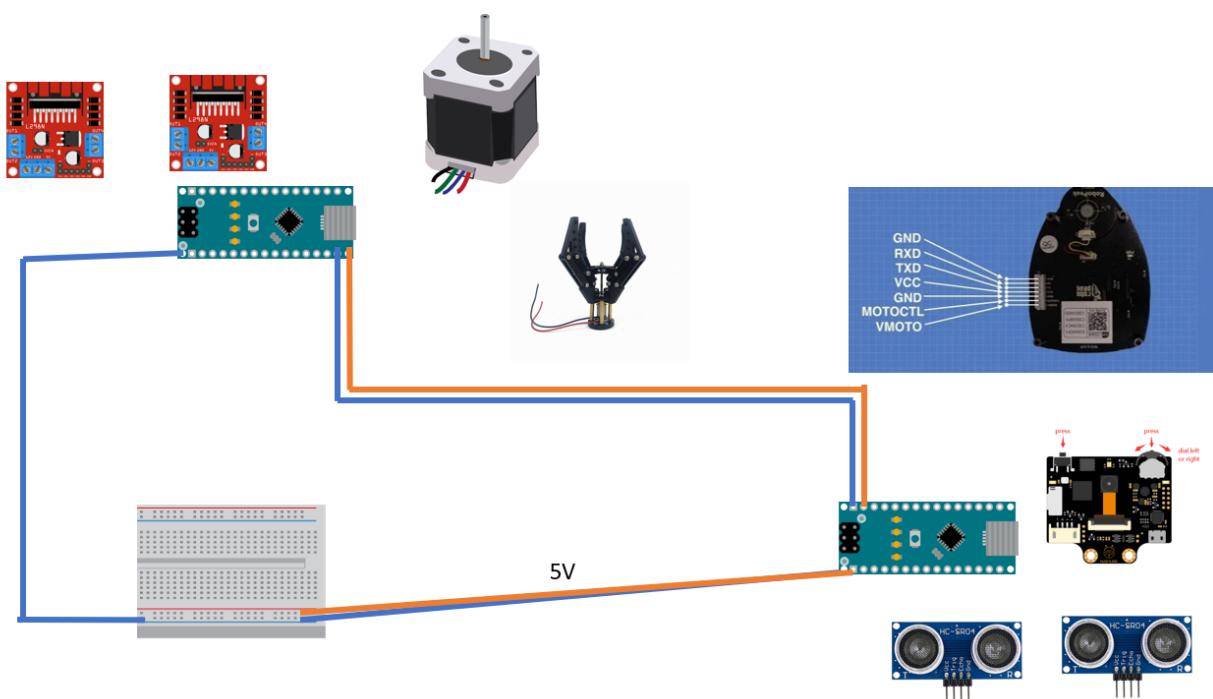
### Ensemble capteurs

Sabemos algo

Attention certains pin analog ne sont pas utilisables en digital <https://forum.arduino.cc/t/can-i-use-all-the-analog-pins-of-arduino-nano-as-digital/922215/2>

2000

Arduino	Fils	
Nano	TX et RX	5 6
Nano	TX et RX	6 5

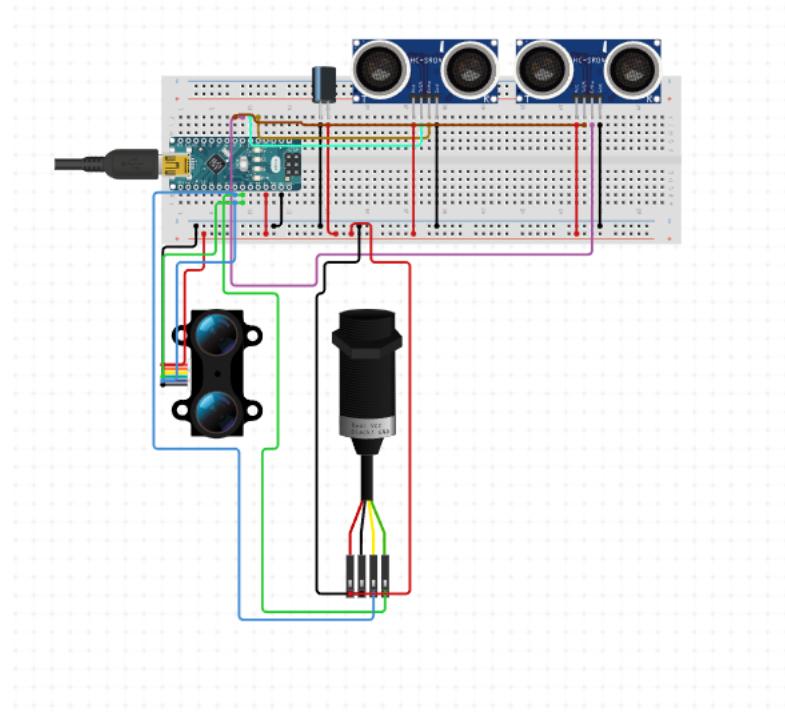


## Capteurs :

Nb fils :

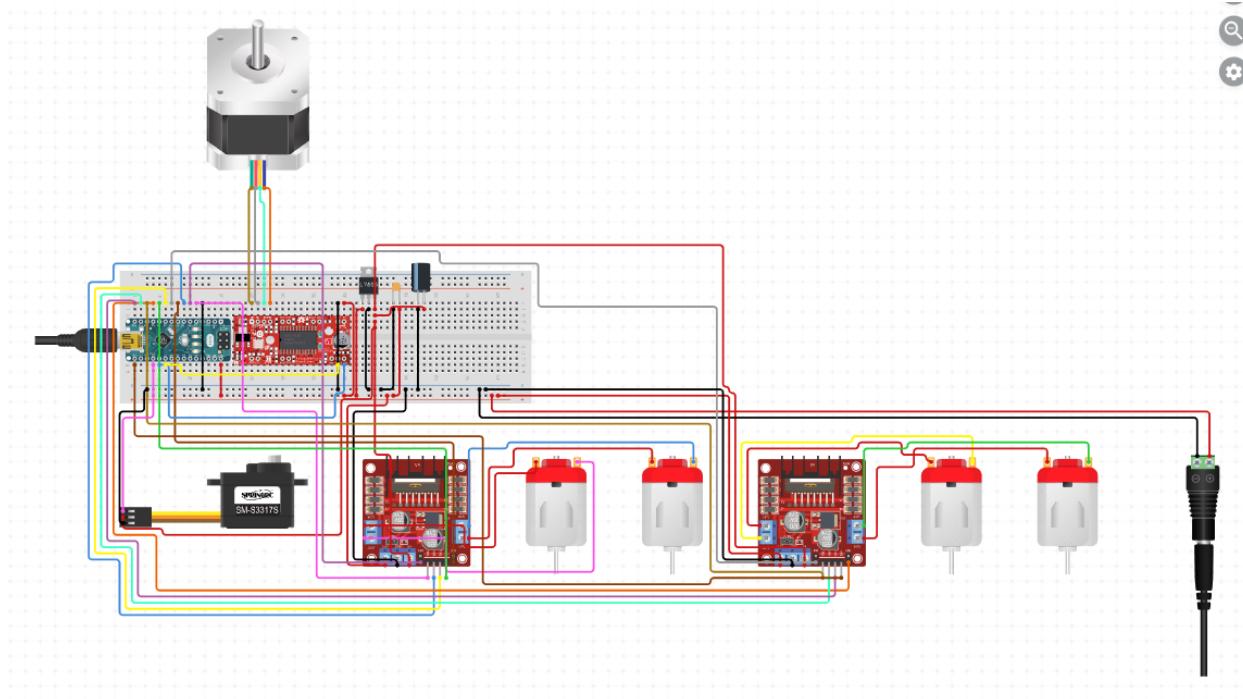
Capteur	Alim	Données	
---------	------	---------	--

Capteur	Alim	Données	
Camera	GND+5V	2	T : D3 R : D4
Ultrason	GND+5V	2	Echo : 10 Trig : 9
Ultrason	GND+5V	2	Echo : 12 Trig : 11
LIDAR	2* (GND+5V)	3	



Actionneurs :

Actionneurs	Alim	Données	
Moteurs roues	2*(GND+5V)+2*(GND+12V)	2*6 fils	ENA 12 ENB 7 ENA A0 ENB A5
Moteur pince	GND+5V	1	D4
Moteur élévateur	GND + 5V + GND + 12V	4	



## Communication arduinos

Pb : port série utilisé par l'ordinateur

Solution librairie altsoftserial [https://www.pjrc.com/teensy/td\\_libs\\_AltSoftSerial.html](https://www.pjrc.com/teensy/td_libs_AltSoftSerial.html)

Ou sur arduino nano : softwareserial example (les bases : <https://forum.arduino.cc/t/serial-input-basics-updated/382007/11> )

UART mauvais bail ?

Welcome to the forum.

May I ask why are you doing this to yourself? SoftwareSerial is an old invention to save a penny for another microcontroller. When you sell millions of units it may be worthwhile doing this kind of tricks. For hobby use I would only recommend this for nostalgic reasons. Otherwise you can use an Arduino that has enough hardware UARTs or use another peripheral (SPI, I2C, ...).

UARTs are asynchronous and use a clock-less signal. That means the master can start a transmission at any time and there is no signal to tell the slave when the data is valid. Master and slave agree on the baudrate (well the programmer does) that tells the slave when the data is valid. There are multiple sources of error (even when using a hardware UART)

- clock frequency tolerance on both master and slave
- some baudrates cannot be created from a given clock frequency e.g., 16MHz / 115200
- other interrupts (SoftwareSerial only)
- high baudrates tend to have higher error rates

<https://forum.arduino.cc/t/communicating-two-arduino-nano-using-softwareserial/895622/4>

En tout cas petit test qui fonctionne :

arduino\_uart\_com

```

if(inByte == 'c') {
    Serial.println("Message received !!!");
}
}

// blank line to separate data from the two ports:
//Serial.println();
}

*/
//transmitter
SoftwareSerial portTwo(8, 9);
char bb[] = {"cCa"};
void setup() {
    // Open serial communications and wait for port to open:
    // Start each software serial port
    portTwo.begin(9600);
}

void loop() {
    // Now listen on the second port
    portTwo.write(bb[random(0, 3)]);
    portTwo.flush();
    delay(2000);

    // blank line to separate data from the two ports:
    Serial.println();
}

```

COM8

```

22:43:03.866 -> CCCcMessage received !!!
22:43:11.866 -> cMessage received !!!
22:43:13.900 -> cMessage received !!!
22:43:15.875 -> cMessage received !!!
22:43:17.892 -> cMessage received !!!
22:43:19.872 -> CCCMessage received !!!
22:43:25.850 -> CcMessage received !!!
22:43:29.885 -> cMessage received !!!
22:43:31.862 -> CCCCMessage received !!!
22:43:41.893 -> cMessage received !!!
22:43:43.898 -> CCCCcMessage received !!!
22:43:53.884 -> CCCMessage received !!!
22:43:59.920 -> cMessage received !!!
22:44:01.879 -> C

```

Autoscroll  Show timestamp

```

#include <SoftwareSerial.h>

/*
//receiver
// software serial #1: RX = digital pin 10, TX = digital pin 11
SoftwareSerial portOne(10, 11);

void setup() {
    // Open serial communications and wait for port to open:
    Serial.begin(9600);
    while (!Serial) {
        ; // wait for serial port to connect. Needed for native USB port only
    }
    // Start each software serial port
    portOne.begin(9600);
}

void loop() {
    // By default, the last initialized port is listening.
    // when you want to listen on a port, explicitly select it:
    portOne.listen();
    // while there is data coming in, read it
    // and send to the hardware serial port:
    while (portOne.available() > 0) {
        char inByte = portOne.read();
        Serial.write(inByte);

        if(inByte == 'c') {
            Serial.println("Message received !!!");
        }
    }

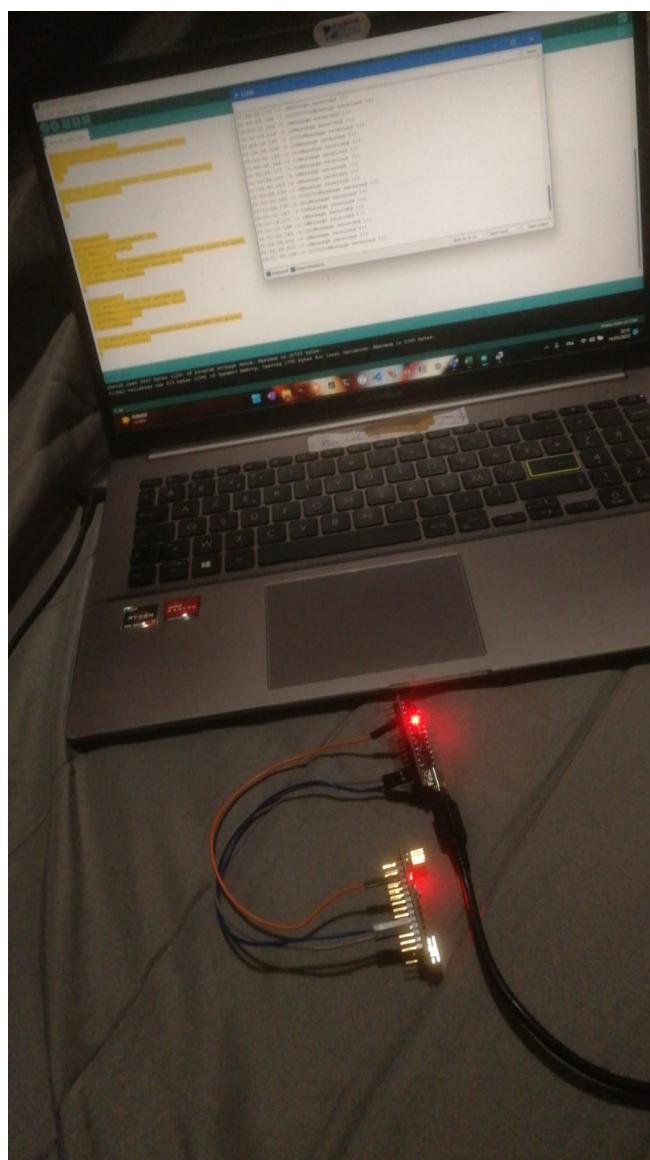
    // blank line to separate data from the two ports:
    //Serial.println();
}

```

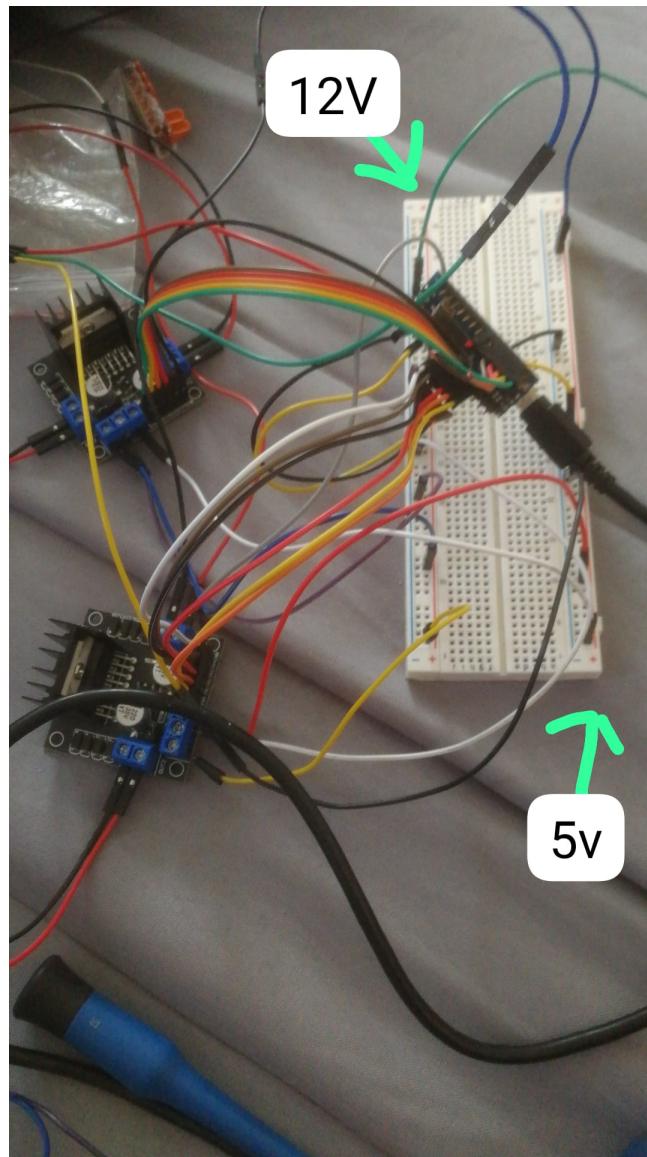
```
}

//transmitter
SoftwareSerial portTwo(8, 9);
char bb[] = {"cca"};
void setup() {
    // Open serial communications and wait for port to open:
    // Start each software serial port
    portTwo.begin(9600);
}
void loop() {
    // Now listen on the second port
    portTwo.write(bb[random(0, 3)]);
    portTwo.flush();
    delay(2000);

    // blank line to separate data from the two ports:
    Serial.println();
}
```



## Moteurs roues



```
*****//  
// Association des entrées du L298N, aux sorties utilisées sur notre Arduino Uno //  
*****//  
#define borneENA      12    // On associe la borne "ENA" du L298N à la pin D10 de l'arduino  
#define borneIN1      11    // On associe la borne "IN1" du L298N à la pin D9 de l'arduino  
#define borneIN2      10    // On associe la borne "IN2" du L298N à la pin D8 de l'arduino  
#define borneIN3      9     // On associe la borne "IN3" du L298N à la pin D7 de l'arduino  
#define borneIN4      8     // On associe la borne "IN4" du L298N à la pin D6 de l'arduino  
#define borneENB      7     // On associe la borne "ENB" du L298N à la pin D5 de l'arduino  
  
*****//  
#define miborneENA     A0    // On associe la borne "ENA" du L298N à la pin D10 de l'arduino  
#define miborneIN1     A1    // On associe la borne "IN1" du L298N à la pin D9 de l'arduino  
#define miborneIN2     A2    // On associe la borne "IN2" du L298N à la pin D8 de l'arduino  
#define miborneIN3     A3    // On associe la borne "IN3" du L298N à la pin D7 de l'arduino  
#define miborneIN4     A4    // On associe la borne "IN4" du L298N à la pin D6 de l'arduino
```

```

#define m1borneENB      A5      // On associe la borne "ENB" du L298N à la pin D5 de l'arduino

//*****
// SETUP //
//*****
void setup() {
    // Configuration de toutes les pins de l'Arduino en "sortie"
    pinMode(borneENA, OUTPUT);
    pinMode(borneIN1, OUTPUT);
    pinMode(borneIN2, OUTPUT);
    pinMode(borneIN3, OUTPUT);
    pinMode(borneIN4, OUTPUT);
    pinMode(borneENB, OUTPUT);

    pinMode(m1borneENA, OUTPUT);
    pinMode(m1borneIN1, OUTPUT);
    pinMode(m1borneIN2, OUTPUT);
    pinMode(m1borneIN3, OUTPUT);
    pinMode(m1borneIN4, OUTPUT);
    pinMode(m1borneENB, OUTPUT);
}

//*****
// Boucle principale : LOOP //
//*****
void loop() {

    // Configuration du L298N en "marche avant", pour le moteur connecté au pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN1, HIGH);           // L'entrée IN1 doit être au niveau haut
    digitalWrite(borneIN2, LOW);            // L'entrée IN2 doit être au niveau bas
    digitalWrite(m1borneIN1, HIGH);          // L'entrée IN1 doit être au niveau haut
    digitalWrite(m1borneIN2, LOW);           // L'entrée IN2 doit être au niveau bas

    // Et on lance le moteur (branché sur le pont A du L298N)
    lancerRotationMoteurPontA();

    // Configuration du L298N en "marche avant", pour le moteur connecté au pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN3, HIGH);           // L'entrée IN1 doit être au niveau haut
    digitalWrite(borneIN4, LOW);            // L'entrée IN2 doit être au niveau bas
    digitalWrite(m1borneIN3, HIGH);          // L'entrée IN1 doit être au niveau haut
    digitalWrite(m1borneIN4, LOW);           // L'entrée IN2 doit être au niveau bas

    // Et on lance le moteur (branché sur le pont A du L298N)
    lancerRotationMoteurPontB();

    // Puis on configure le L298N en "marche arrière", pour le moteur câblé sur le pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN1, LOW);             // L'entrée IN1 doit être au niveau bas
    digitalWrite(borneIN2, HIGH);            // L'entrée IN2 doit être au niveau haut

    // Et on relance le moteur (branché sur le pont A du L298N)
    lancerRotationMoteurPontA();

    // Puis on configure le L298N en "marche arrière", pour le moteur câblé sur le pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN3, LOW);             // L'entrée IN1 doit être au niveau bas
    digitalWrite(borneIN4, HIGH);            // L'entrée IN2 doit être au niveau haut
    digitalWrite(m1borneIN3, LOW);           // L'entrée IN1 doit être au niveau bas
    digitalWrite(m1borneIN4, HIGH);          // L'entrée IN2 doit être au niveau haut

    // Et on relance le moteur (branché sur le pont A du L298N)
    lancerRotationMoteurPontB();

    ouvrir();
    delay(1000);
    prise();
    delay(2000);
    ouvrir();
    delay(1000);
    fermer();

}

//*****
// Fonction : lancerRotationMoteurPontA()                                //
// But :      Active l'alimentation du moteur branché sur le pont A      //
//            pendant 2 secondes, puis le met à l'arrêt (au moins 1 seconde)  //
//*****
void lancerRotationMoteurPontA() {
    digitalWrite(borneENA, HIGH);        // Active l'alimentation du moteur 1
    digitalWrite(m1borneENA, HIGH);       // Active l'alimentation du moteur 1
    delay(2000);                      // et attend 2 secondes
}

```

```

digitalWrite(borneENA, LOW);           // Désactive l'alimentation du moteur 1
digitalWrite(m1borneENA, LOW);        // Active l'alimentation du moteur 1
delay(1000);                         // et attend 1 seconde
}

void lancerRotationMoteurPontB() {
    digitalWrite(m1borneENB, HIGH);    // Active l'alimentation du moteur 1
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 1
    delay(2000);                     // et attend 2 secondes

    digitalWrite(m1borneENB, LOW);     // Désactive l'alimentation du moteur 1
    digitalWrite(borneENB, LOW);       // Active l'alimentation du moteur 1
    delay(1000);                     // et attend 1 seconde
}

```

Roues mecanum tableau direction infos : <https://dronebotworkshop.com/mecanum/>

Référence vue de face avec pince devant et arduino derrière

Moteur 1 carte 1 sens avant : 01

Moteur 2 carte 1 sens avant : 10

	Devant	Droite	Gauche	Derrière	
IN1	0	1	0	1	
IN2	1	0	1	0	
IN3	1	1	0	0	
IN4	0	0	1	1	
m1_IN1	0	1	0	1	
m1_IN2	1	0	1	0	
m1_IN3	1	1	0	0	
m1_IN4	0	0	1	1	

```

//active les 4 moteurs en sens avant
void devant(){
    digitalWrite(borneENA, HIGH);      // Active l'alimentation du moteur 1
    digitalWrite(m1borneENA, HIGH);    // Active l'alimentation du moteurs 1
    digitalWrite(m1borneENB, HIGH);    // Active l'alimentation du moteur 2
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 2

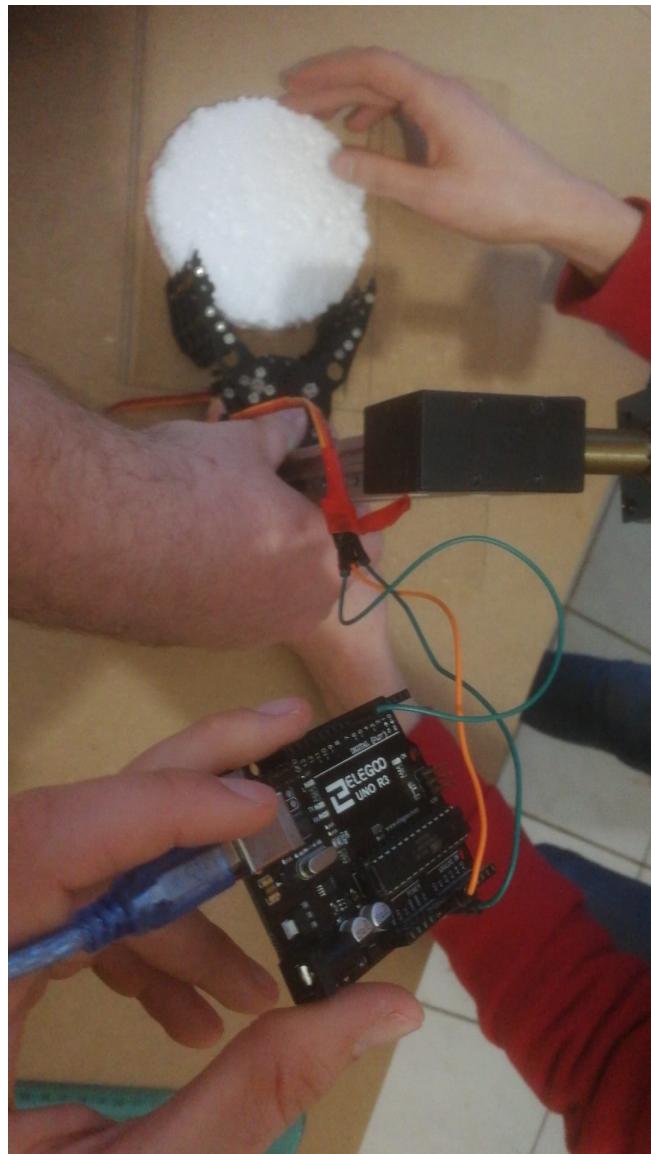
    //sens des moteurs
    digitalWrite(borneIN1, LOW);
    digitalWrite(borneIN2, HIGH);
    digitalWrite(m1borneIN1, LOW);
    digitalWrite(m1borneIN2, HIGH);

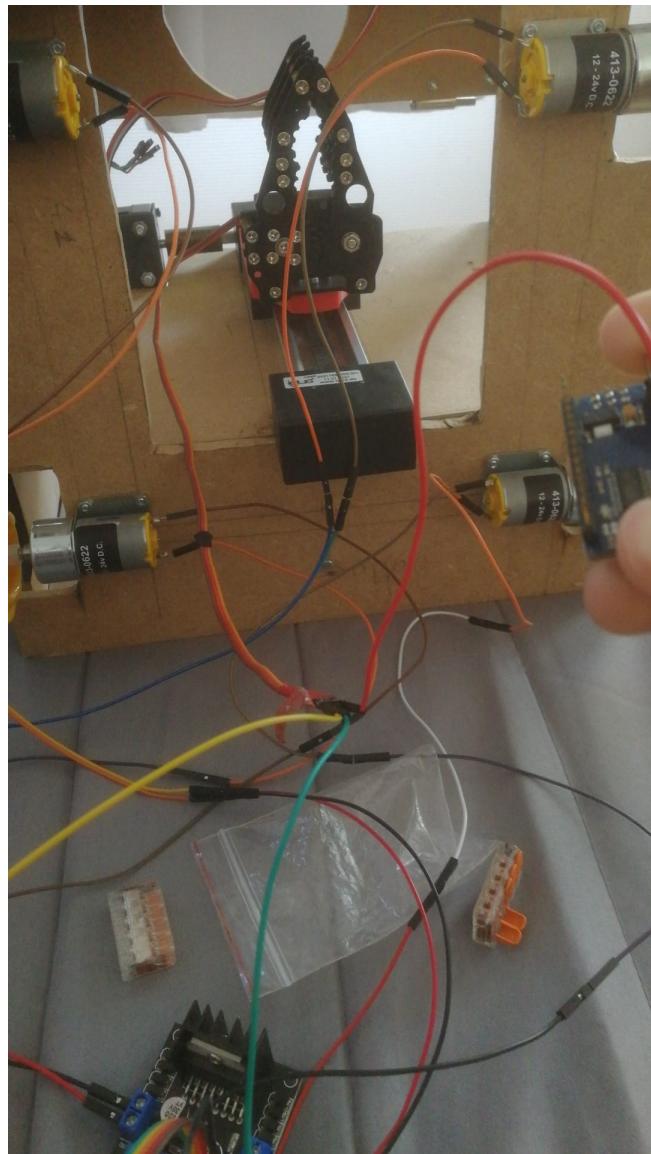
    digitalWrite(borneIN3, HIGH);
    digitalWrite(borneIN4, LOW);
    digitalWrite(m1borneIN3, HIGH);
    digitalWrite(m1borneIN4, LOW);
    delay(3000);

    //désactive moteurs
    digitalWrite(borneENA, LOW);
    digitalWrite(m1borneENA, LOW);
    digitalWrite(m1borneENB, LOW);
    digitalWrite(borneENB, LOW);
    delay(3000);
}

```

## Moteur pince

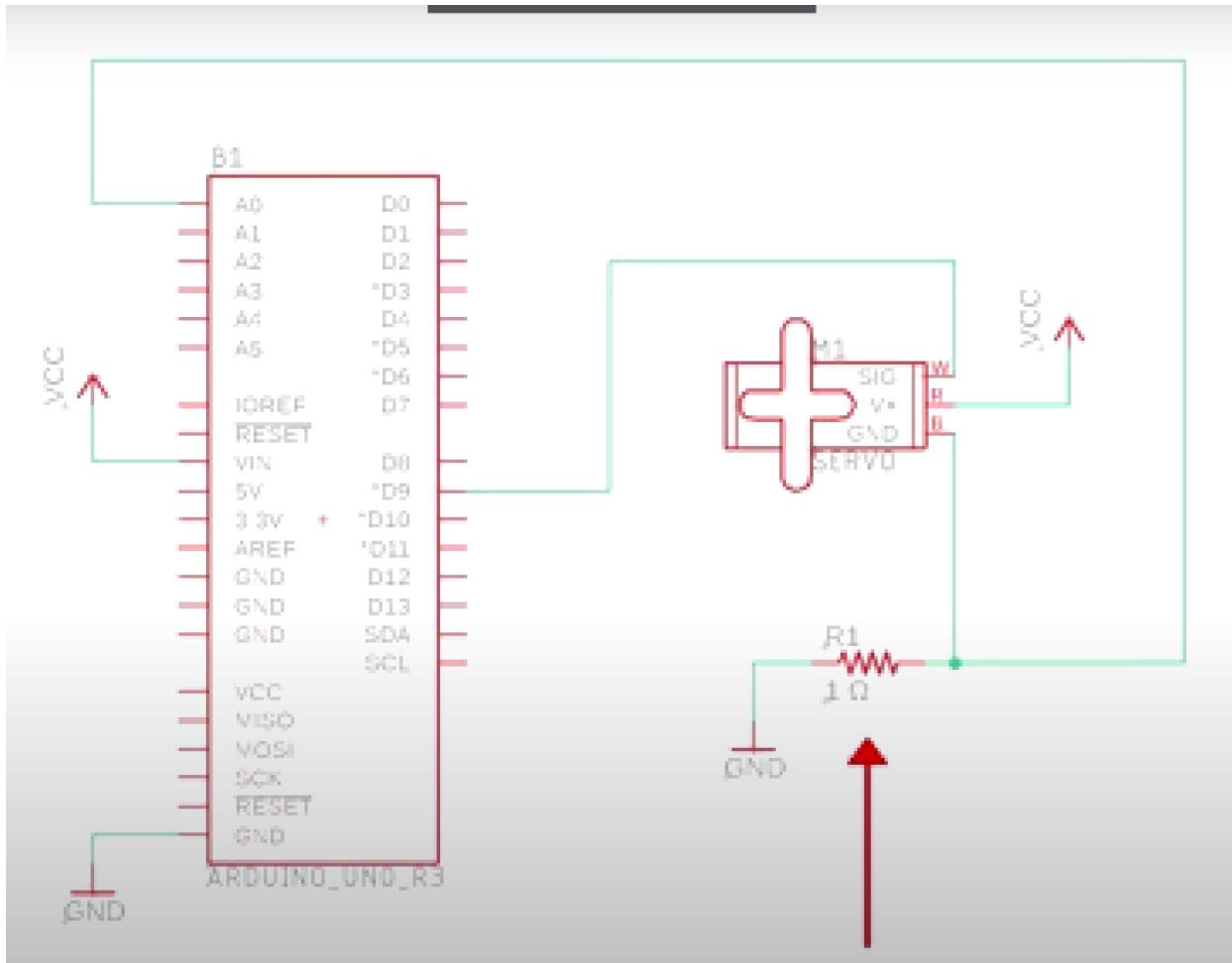




Jaune = signal, milieu = + et violet = gnd

Contrôle de la pince avec détection surtension

Pour la surtension : <https://youtu.be/LKJLCJvyVdk>



### Résistance faible

```
/****************************************************************************
yt : https://youtu.be/LKJLCJvyVdk
Board : arduino uno
****

#include <Servo.h>

Servo myservo1;

int pos = 0;      // variable to store the servo position
int servoPin = A0;
int sensorValue;
int threshold = 30;

//Initialize the servo
void serv_setup() {
  myservo1.attach(9);
  Serial.begin(9600);
  pinMode(INPUT, A0);
}

void detecSurtension(){
  sensorValue = analogRead(servoPin);
  Serial.println(sensorValue);
  if(sensorValue >= threshold){
    myservo1.write(180);
    delay(1000);
  }
}

void fermer() {
```

```

for (pos = 180; pos >= 80; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo1.write(pos);           // tell servo to go to position in variable 'pos'
    delay(15);                  // waits 15ms for the servo to reach the position
}
detecSurtension();

}

void ouvrir() {
    for (pos = 110; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
        // in steps of 1 degree
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'

        //sensorValue = analogRead(servoPin);
        //Serial.println(sensorValue);
        delay(15);                  // waits 15ms for the servo to reach the position
    }

    sensorValue = analogRead(servoPin);
    Serial.println(sensorValue);
}

void prise() {
    for (pos = 180; pos >= 110; pos -= 1) { // goes from 180 degrees to 0 degrees
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'

        //sensorValue = analogRead(servoPin);
        //Serial.println(sensorValue);
        delay(15);                  // waits 15ms for the servo to reach the position
    }

    detecSurtension();
}

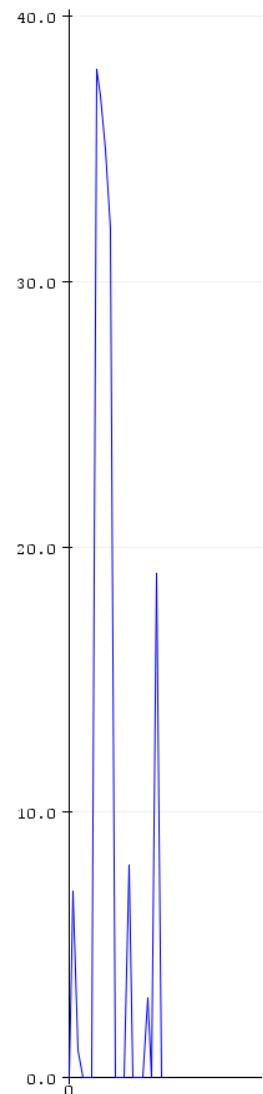
void setup() {
    serv_setup();           //Initialize the servo
}

void loop() {
//  read value

    //stop effort
    detecSurtension();

    ouvrir();
    delay(1000);
    prise();
    delay(5000);
    ouvrir();
    delay(1000);
    fermer();
}

```



Choix du threshold

### Moteur élévateur

# Stepper Motor Operation

## Bipolar Stepper

- 4 Wire Connection
- Needs Voltage Reversal
- Higher Torque
- Slower Max Speed
- Advanced Controller

## Unipolar Stepper

- 5 or 6 Wire Connection
- No Voltage Reversal
- Lower Torque
- Higher Max Speed
- Simple Controller

# Stepper Motor Specs

## Step Angle

- Number of degrees shaft advances per step
- Sometimes specified as *Steps per Revolution*
- Easy to convert between two specifications

$$1.8^\circ \text{ per step} = 200 \text{ steps per revolution}$$

Yt vidéo explicative : <https://youtu.be/0qwrnUeSpYQ>.

<https://www.instructables.com/Arduino-6-wire-Stepper-Motor-Tutorial/>

Le moteur <https://fr.rs-online.com/web/p/moteurs-pas-a-pas/1805279>

Problème pas de driver compatible avec l'arduino nano

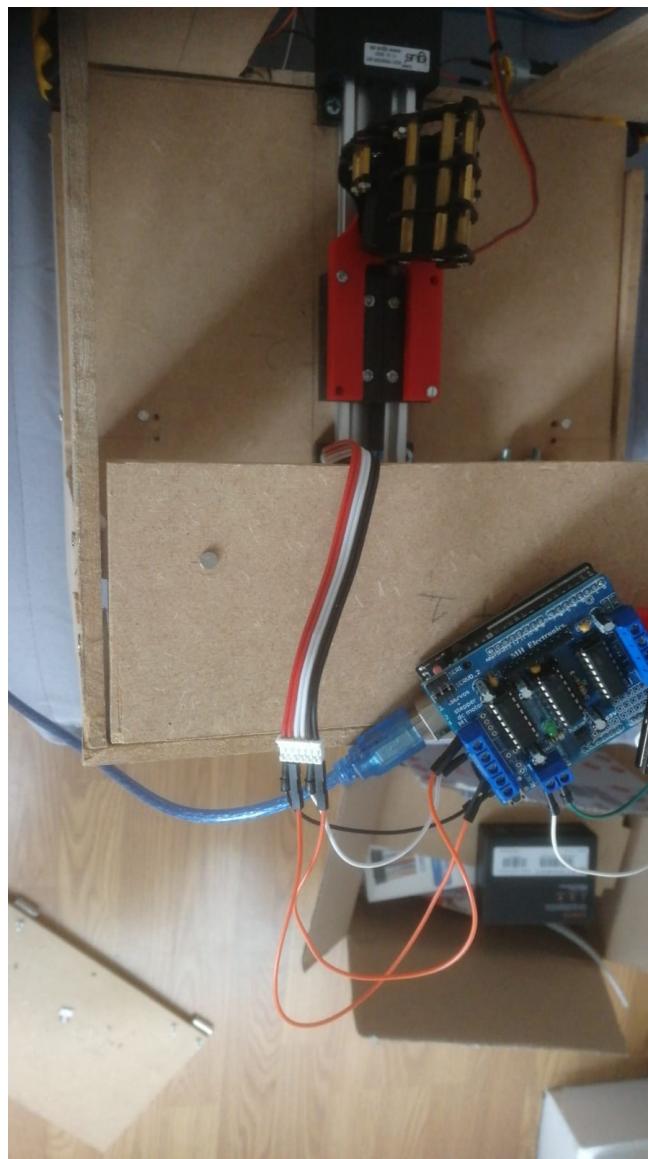
Par exemple : <https://www.pololu.com/category/120/stepper-motor-drivers>

Tutoriel complet

<https://how2electronics.com/control-stepper-motor-with-a4988-driver-arduino/>

test fonctionnel :

vidéo



```
#include <AFMotor.h>
AF_Stepper motor(200, 1); //360/1.8
//AF_DCMotor motor(1, MOTOR12_64KHZ);

void setup() {
    motor.setSpeed(20);
    motor.onestep(FORWARD, SINGLE);
    motor.release();
    delay(1000);
    //motor.setSpeed(200);

    motor.step(100, BACKWARD, DOUBLE);
    // motor.release();
}
void loop() {
    //motor.step(100, BACKWARD, DOUBLE);
    //delay(100);
    // motor.step(100, FORWARD, SINGLE);
    // delay(1000);
    // motor.step(100, BACKWARD, SINGLE);
```

```

// motor.step(100, FORWARD, DOUBLE);
// motor.step(100, BACKWARD, DOUBLE);
// motor.step(100, FORWARD, INTERLEAVE);
// motor.step(100, BACKWARD, INTERLEAVE);
// motor.step(100, FORWARD, MICROSTEP);
// motor.step(100, BACKWARD, MICROSTEP);
/*
motor.run(FORWARD);
delay(1000);
motor.run(BACKWARD);
delay(1000);
motor.run(RELEASE);
delay(1000); */
}

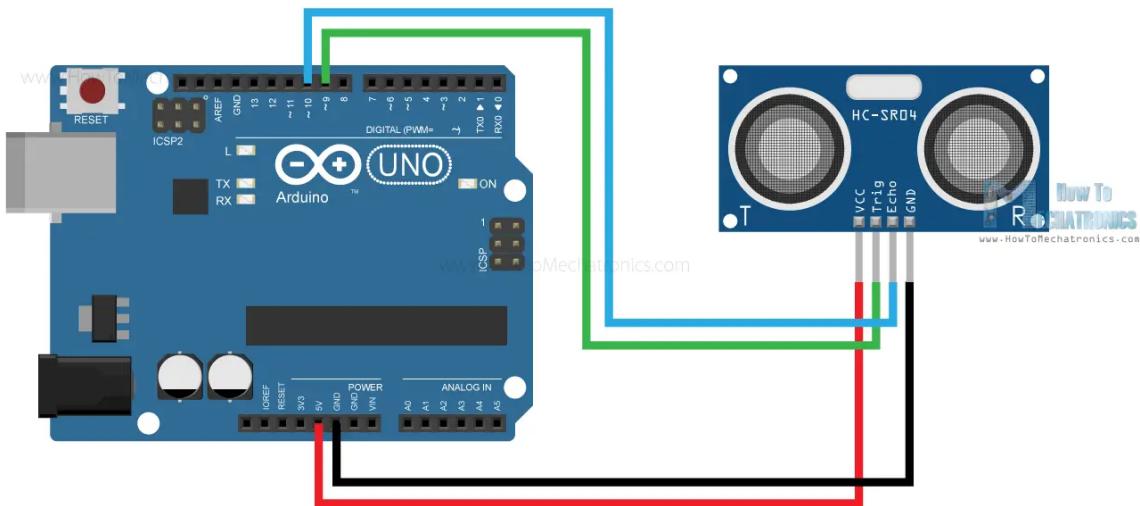
```

(pour d'autres types de stepper motor, ceux qui ont 5 fils, on peut utiliser ça <https://www.instructables.com/Arduino-Nano-and-Visuino-Control-Stepper-Motor-Wit/>)

## Ultrason

<https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>

### HC-SR04 Ultrasonic Sensor and Arduino Wiring



```

// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
}
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
}

```

```

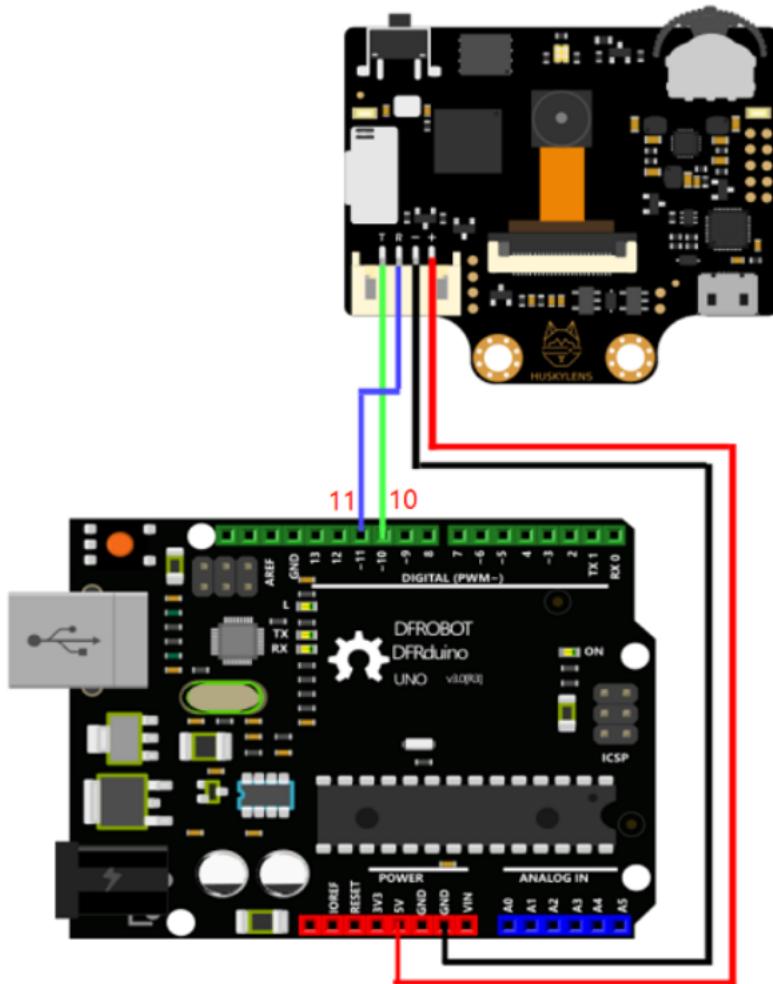
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance = duration * 0.034 / 2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}

```

## Camera

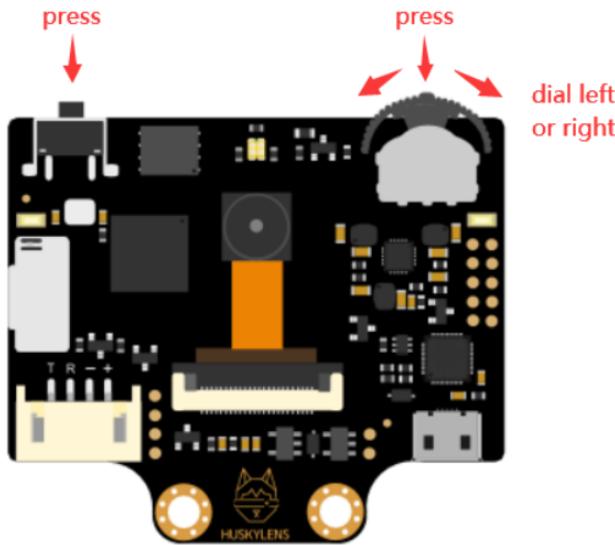
[https://wiki.dfrobot.com/HUSYLENS\\_V1.0\\_SKU\\_SEN0305\\_SEN0336#target\\_1](https://wiki.dfrobot.com/HUSYLENS_V1.0_SKU_SEN0305_SEN0336#target_1)

Connection Diagram



Réglage de la camera manuel

- Dial the "function button" to left or right to switch different functions.
- Short press the "Learning button" to learn the specified object; long press the "Learning button" to continuously learn the specified object from different angles and distances; if HuskyLens has learned the object before, short press the "Learn button" to make it forget.
- Long press the "function button" to enter into the second-level menu(parameter setting) in the current function. Dial left, right or short press the "function button" to set related parameters.



```
#include "HUSKYLENS.h"
#include "SoftwareSerial.h"

HUSKYLENS huskylens;
SoftwareSerial mySerial(10, 11); // RX, TX
//HUSKYLENS green line >> Pin 10; blue line >> Pin 11
void printResult(HUSKYLENSResult result);

void setup() {
    Serial.begin(115200);
    mySerial.begin(9600);
    while (!huskylens.begin(mySerial))
    {
        Serial.println(F("Begin failed!"));
        Serial.println(F("1.Please recheck the \"Protocol Type\" in HUSKYLENS (General Settings>>Protocol Type>>Serial 9600)"));
        Serial.println(F("2.Please recheck the connection."));
        delay(100);
    }
}

void loop() {
    if (!huskylens.request()) Serial.println(F("Fail to request data from HUSKYLENS, recheck the connection!"));
    else if(!huskylens.isLearned()) Serial.println(F("Nothing learned, press learn button on HUSKYLENS to learn one!"));
    else if(!huskylens.available()) Serial.println(huskylens.read().command);//Serial.println(F("No block or arrow appears on the screen!"))
    else
    {
        Serial.println(F("#####"));
        while (huskylens.available())
        {
            HUSKYLENSResult result = huskylens.read();
            printResult(result);
        }
        delay(10000);
    }
}

void printResult(HUSKYLENSResult result){
    if (result.command == COMMAND_RETURN_BLOCK){
```

```

        Serial.println(String() +F("Block:xCenter=") +result.xCenter+F("yCenter=") +result.yCenter+F("width=") +result.width+F("height=") +re
    }
    else if (result.command == COMMAND_RETURN_ARROW){
        Serial.println(String() +F("Arrow:xOrigin=") +result.xOrigin+F("yOrigin=") +result.yOrigin+F("xTarget=") +result.xTarget+F("yTarget=")
    }
    else{
        Serial.println("Object unknown!");
    }
}

```

Résultat d'une détection

```

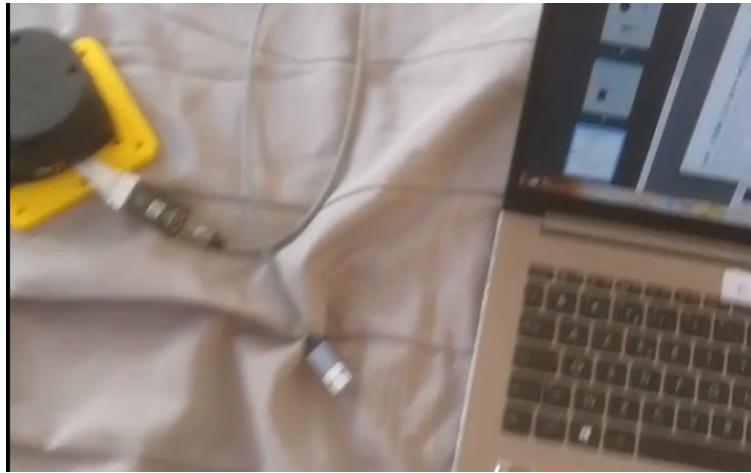
20:29:20.342 -> 255
20:29:20.342 -> 255
20:29:20.389 -> 255
20:29:20.389 -> 255
20:29:20.436 -> 255
20:29:20.436 -> 255
20:29:20.483 -> #####
20:29:20.483 -> Block:xCenter=170,yCenter=84,width=84,height=112,ID=0

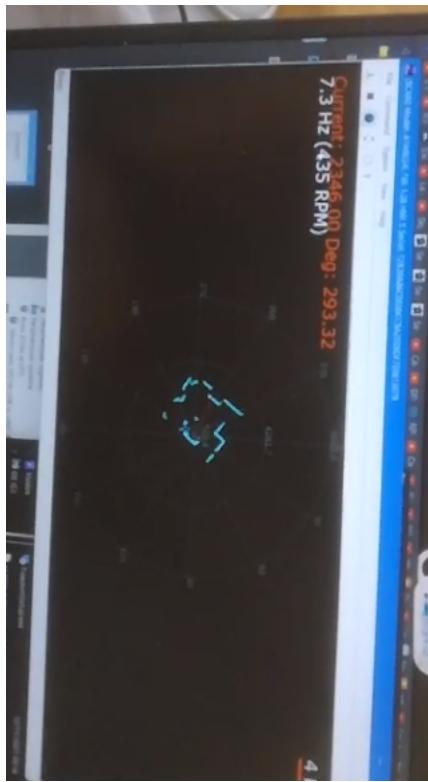
```

Tester de détecter les bloc puis diriger le robot en fonction

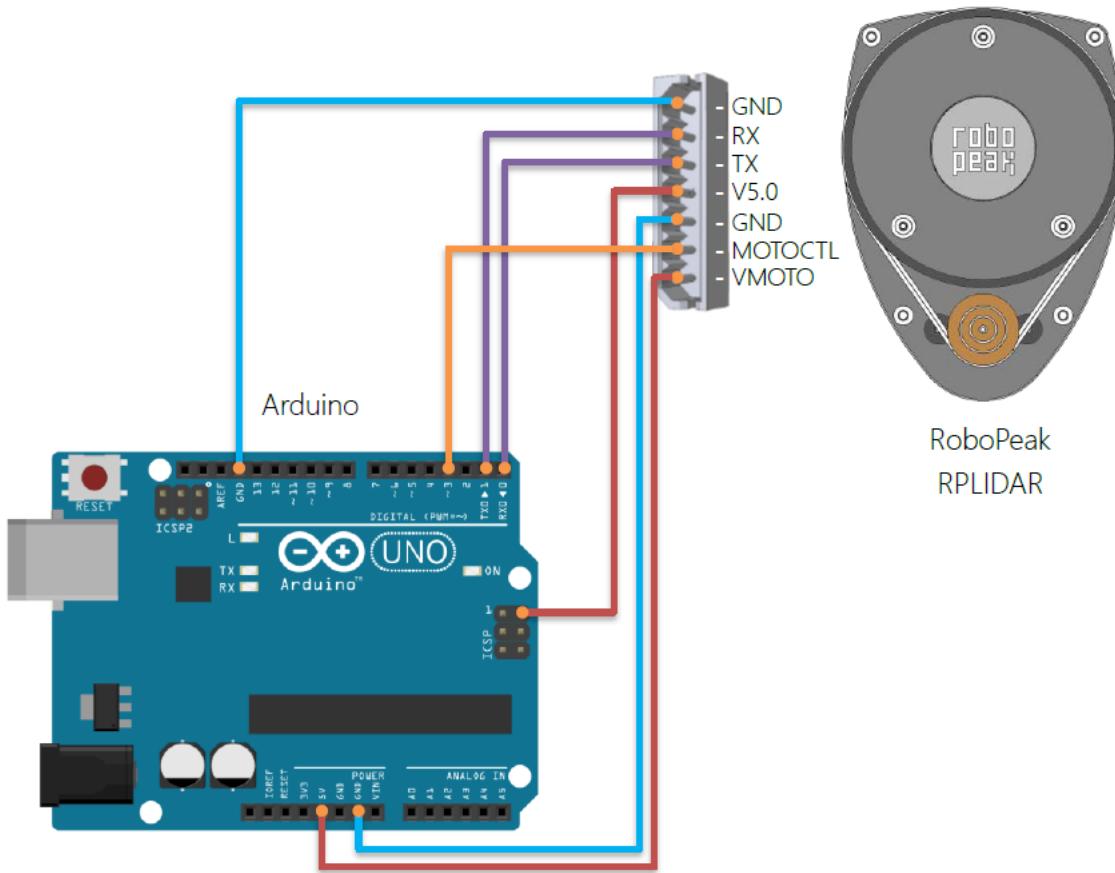
## LIDAR

test basique [https://bucket-download.slamtec.com/e680b4e2d99c4349c019553820904f28c7e6ec32/LM108\\_SLAMTEC\\_rplidarKit\\_usermanual\\_A1M8\\_v1.0\\_en.pdf](https://bucket-download.slamtec.com/e680b4e2d99c4349c019553820904f28c7e6ec32/LM108_SLAMTEC_rplidarKit_usermanual_A1M8_v1.0_en.pdf)





avec arduino <https://mevihub.com/rplidar-a1-laser-scanner-with-arduino/>



test recuper données LIDAR sur 1 arduino et envoyer le résultat à un autre arduino

## Ensemble capteurs

Objectif : envoie info détection capteurs

```

//*****
// Entrées-sorties capteurs
//*****


// ultrason
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;

//com : transmitter
#include <SoftwareSerial.h>
SoftwareSerial portTwo(6, 5);

void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input

  // Open serial communications and wait for port to open:
  // Start each software serial port
  portTwo.begin(9600);
}

void loop() {

  // Clears the trigPin
  digitalWrite(trigPin, LOW);

```

```

delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance = duration * 0.034 / 2;
/*
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
*/
if(distance < 30){
    // ouverture fermeture pince
    portTwo.write('o');
    portTwo.flush();
    delay(1000);
}
else if(distance >40 && distance <90) {
    //reculer
    portTwo.write('c');
    portTwo.flush();
    delay(1000);
}
else {
    portTwo.write('F');
    portTwo.flush();
}
}

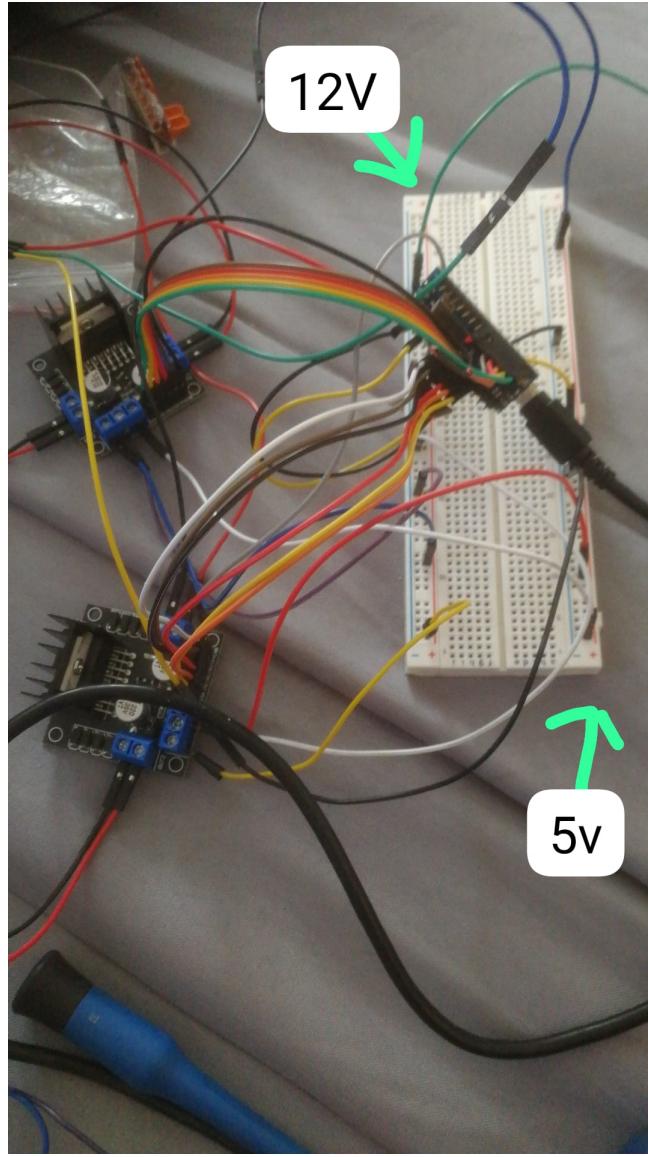
```

## Ensemble actionneurs

### ▼ 1er test

Vérifier que tous les actionneurs fonctionnent

cablage



## Vidéo :

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/1943295b-f3aa-49e9-8cdb-589b9a562c76/moteurs.mp4>

```
/*
 Fichier:      MarcheArretMoteurL298N
 Description:   Exemple de code Arduino, permettant de faire tourner un moteur à courant continu (marche avant/marche arrière), via un
 Auteur:       Jérôme, Passion-Électronique (https://passionelectronique.fr/)
 Création :    05.05.2021
 */

//*****//
// Association des entrées du L298N, aux sorties utilisées sur notre Arduino Uno //
//*****//

#define borneENA     12      // On associe la borne "ENA" du L298N à la pin D10 de l'arduino
#define borneIN1     11      // On associe la borne "IN1" du L298N à la pin D9 de l'arduino
#define borneIN2     10      // On associe la borne "IN2" du L298N à la pin D8 de l'arduino
```

```

#define borneIN3      9      // On associe la borne "IN3" du L298N à la pin D7 de l'arduino
#define borneIN4      8      // On associe la borne "IN4" du L298N à la pin D6 de l'arduino
#define borneENB      7      // On associe la borne "ENB" du L298N à la pin D5 de l'arduino

//********************************************************************//  

#define m1borneENA     A0    // On associe la borne "ENA" du L298N à la pin D10 de l'arduino
#define m1borneIN1     A1    // On associe la borne "IN1" du L298N à la pin D9 de l'arduino
#define m1borneIN2     A2    // On associe la borne "IN2" du L298N à la pin D8 de l'arduino
#define m1borneIN3     A3    // On associe la borne "IN3" du L298N à la pin D7 de l'arduino
#define m1borneIN4     A4    // On associe la borne "IN4" du L298N à la pin D6 de l'arduino
#define m1borneENB     A5    // On associe la borne "ENB" du L298N à la pin D5 de l'arduino

//pince
#define pinceP 4
#include <Servo.h>

Servo myservo1;
int pos = 0;      // variable to store the servo position

void fermer() {
    for (pos = 180; pos >= 80; pos -= 1) { // goes from 180 degrees to 0 degrees
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

void ouvrir() {
    for (pos = 110; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
        // in steps of 1 degree
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

void priser() {
    for (pos = 180; pos >= 110; pos -= 1) { // goes from 180 degrees to 0 degrees
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

//*****
// SETUP //
//*****//
void setup() {
    myservo1.attach(pinceP);
    // Configuration de toutes les pins de l'Arduino en "sortie" (car elles attaquent les entrées du module L298N)
    pinMode(borneENA, OUTPUT);
    pinMode(borneIN1, OUTPUT);
    pinMode(borneIN2, OUTPUT);
    pinMode(borneIN3, OUTPUT);
    pinMode(borneIN4, OUTPUT);
    pinMode(borneENB, OUTPUT);

    pinMode(m1borneENA, OUTPUT);
    pinMode(m1borneIN1, OUTPUT);
    pinMode(m1borneIN2, OUTPUT);
    pinMode(m1borneIN3, OUTPUT);
    pinMode(m1borneIN4, OUTPUT);
    pinMode(m1borneENB, OUTPUT);
}

//*****
// Boucle principale : LOOP //
//*****//
void loop() {

    // Configuration du L298N en "marche avant", pour le moteur connecté au pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN1, HIGH);          // L'entrée IN1 doit être au niveau haut
    digitalWrite(borneIN2, LOW);           // L'entrée IN2 doit être au niveau bas
    digitalWrite(m1borneIN1, HIGH);         // L'entrée IN1 doit être au niveau haut
    digitalWrite(m1borneIN2, LOW);          // L'entrée IN2 doit être au niveau bas

    // Et on lance le moteur (branché sur le pont A du L298N)
    lancerRotationMoteurPontA();

    // Configuration du L298N en "marche avant", pour le moteur connecté au pont A. Selon sa table de vérité, il faut que :
    digitalWrite(borneIN3, HIGH);          // L'entrée IN1 doit être au niveau haut
    digitalWrite(borneIN4, LOW);           // L'entrée IN2 doit être au niveau bas
    digitalWrite(m1borneIN3, HIGH);         // L'entrée IN1 doit être au niveau haut
}

```

```

digitalWrite(miborneIN4, LOW); // L'entrée IN2 doit être au niveau bas

// Et on lance le moteur (branché sur le pont A du L298N)
lancerRotationMoteurPontB();

// Puis on configure le L298N en "marche arrière", pour le moteur câblé sur le pont A. Selon sa table de vérité, il faut que :
digitalWrite(borneIN1, LOW); // L'entrée IN1 doit être au niveau bas
digitalWrite(borneIN2, HIGH); // L'entrée IN2 doit être au niveau haut

// Et on relance le moteur (branché sur le pont A du L298N)
lancerRotationMoteurPontA();

// Puis on configure le L298N en "marche arrière", pour le moteur câblé sur le pont A. Selon sa table de vérité, il faut que :
digitalWrite(borneIN3, LOW); // L'entrée IN1 doit être au niveau bas
digitalWrite(borneIN4, HIGH); // L'entrée IN2 doit être au niveau haut
digitalWrite(miborneIN3, LOW); // L'entrée IN1 doit être au niveau bas
digitalWrite(miborneIN4, HIGH); // L'entrée IN2 doit être au niveau haut

// Et on relance le moteur (branché sur le pont A du L298N)
lancerRotationMoteurPontB();

ouvrir();
delay(1000);
prise();
delay(2000);
ouvrir();
delay(1000);
fermer();

// Puis... on boucle à l'infini !
}

//*****//
// Fonction : lancerRotationMoteurPontA() //
// But : Active l'alimentation du moteur branché sur le pont A //
// pendant 2 secondes, puis le met à l'arrêt (au moins 1 seconde) //
//*****//
void lancerRotationMoteurPontA() {
    digitalWrite(borneENA, HIGH); // Active l'alimentation du moteur 1
    digitalWrite(miborneENA, HIGH); // Active l'alimentation du moteur 1
    delay(2000); // et attend 2 secondes

    digitalWrite(borneENA, LOW); // Désactive l'alimentation du moteur 1
    digitalWrite(miborneENA, LOW); // Active l'alimentation du moteur 1
    delay(1000); // et attend 1 seconde
}

void lancerRotationMoteurPontB() {
    digitalWrite(miborneENB, HIGH); // Active l'alimentation du moteur 1
    digitalWrite(borneENB, HIGH); // Active l'alimentation du moteur 1
    delay(2000); // et attend 2 secondes

    digitalWrite(miborneENB, LOW); // Désactive l'alimentation du moteur 1
    digitalWrite(borneENB, LOW); // Active l'alimentation du moteur 1
    delay(1000); // et attend 1 seconde
}

```

### ▼ Une fois câblé

Lis infos de l'arduino des capteurs, si lettre o → ouvre la pince

```

/*
Fichier: MarcheArretMoteurL298N
Description: Exemple de code Arduino, permettant de faire tourner un moteur à courant continu (marche avnat/marche arrière), via un
Auteur: Jérôme, Passion-Électronique (https://passionelectronique.fr/)
Création : 05.05.2021
*/

//*****//
// Sorties moteurs //
//*****//

#define borneENA 12 // On associe la borne "ENA" du L298N à la pin D10 de l'arduino
#define borneIN1 11 // On associe la borne "IN1" du L298N à la pin D9 de l'arduino
#define borneIN2 10 // On associe la borne "IN2" du L298N à la pin D8 de l'arduino
#define borneIN3 9 // On associe la borne "IN3" du L298N à la pin D7 de l'arduino
#define borneIN4 8 // On associe la borne "IN4" du L298N à la pin D6 de l'arduino
#define borneENB 7 // On associe la borne "ENB" du L298N à la pin D5 de l'arduino

```

```

//*****
#define m1borneENA      A0      // On associe la borne "ENA" du L298N à la pin D10 de l'arduino
#define m1borneIN1      A1      // On associe la borne "IN1" du L298N à la pin D9 de l'arduino
#define m1borneIN2      A2      // On associe la borne "IN2" du L298N à la pin D8 de l'arduino
#define m1borneIN3      A3      // On associe la borne "IN3" du L298N à la pin D7 de l'arduino
#define m1borneIN4      A4      // On associe la borne "IN4" du L298N à la pin D6 de l'arduino
#define m1borneENB      A5      // On associe la borne "ENB" du L298N à la pin D5 de l'arduino

//pince
#define pinceP 4

//Com arduino : receiver
#include <SoftwareSerial.h>
SoftwareSerial portOne(6, 5);
#include <Servo.h>

Servo myservo1;
int pos = 0;      // variable to store the servo position

void fermer() {
    for (pos = 180; pos >= 80; pos -= 1) { // goes from 180 degrees to 0 degrees
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

void ouvrir() {
    for (pos = 110; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
        // in steps of 1 degree
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

void prise() {
    for (pos = 180; pos >= 110; pos -= 1) { // goes from 180 degrees to 0 degrees
        myservo1.write(pos);           // tell servo to go to position in variable 'pos'
        delay(15);                  // waits 15ms for the servo to reach the position
    }
}

//*****
// SETUP //
//*****
void setup() {

    //recup infos sur l'ordi
    Serial.begin(9600);
    while (!Serial) {
        ; // wait for serial port to connect. Needed for native USB port only
    }

    //com
    portOne.begin(9600);
    //sortie servo pince
    myservo1.attach(pinceP);
    // Configuration de toutes les pins de l'Arduino en "sortie"
    pinMode(m1borneENA, OUTPUT);
    pinMode(m1borneIN1, OUTPUT);
    pinMode(m1borneIN2, OUTPUT);
    pinMode(m1borneIN3, OUTPUT);
    pinMode(m1borneIN4, OUTPUT);
    pinMode(m1borneENB, OUTPUT);

    pinMode(m1borneENA, OUTPUT);
    pinMode(m1borneIN1, OUTPUT);
    pinMode(m1borneIN2, OUTPUT);
    pinMode(m1borneIN3, OUTPUT);
    pinMode(m1borneIN4, OUTPUT);
    pinMode(m1borneENB, OUTPUT);
}

//*****
// Boucle principale : LOOP //
//*****
void loop() {
    //devant():
    //droite();
    //gauche();
}

```

```

//com
portOne.listen();
//Serial.println("a");
while (portOne.available() > 0) {
    char inByte = portOne.read();
    //Serial.println("Bledans");
    Serial.write(inByte);
    switch (inByte) {
        case 'c':
            Serial.println("\nAAAAA");
            derriere();
            break;
        case 'o':
            Serial.println("\nOUVRE");
            ouvrir();
            delay(1000);
            prise();
            delay(1000);
            break;
        default:
            // statements
            break;
    }
}

//derriere();
/*
ouvrir();
delay(1000);
prise();
delay(2000);
ouvrir();
delay(1000);
fermer();
*/
}

//active les 4 moteurs en sens avant
void devant(){
    digitalWrite(borneENA, HIGH);      // Active l'alimentation du moteur 1
    digitalWrite(miborneENA, HIGH);    // Active l'alimentation du moteur 1
    digitalWrite(miborneENB, HIGH);    // Active l'alimentation du moteur 2
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 2

    //sens des moteurs
    digitalWrite(borneIN1, LOW);
    digitalWrite(borneIN2, HIGH);
    digitalWrite(miborneIN1, LOW);
    digitalWrite(miborneIN2, HIGH);

    digitalWrite(borneIN3, HIGH);
    digitalWrite(borneIN4, LOW);
    digitalWrite(miborneIN3, HIGH);
    digitalWrite(miborneIN4, LOW);
    delay(3000);

    //désactive moteurs
    digitalWrite(borneENA, LOW);
    digitalWrite(miborneENA, LOW);
    digitalWrite(miborneENB, LOW);
    digitalWrite(borneENB, LOW);
    delay(3000);
}

//active les 4 moteurs en sens avant
void droite(){
    digitalWrite(borneENA, HIGH);      // Active l'alimentation du moteur 1
    digitalWrite(miborneENA, HIGH);    // Active l'alimentation du moteur 1
    digitalWrite(miborneENB, HIGH);    // Active l'alimentation du moteur 2
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 2

    //sens des moteurs
    digitalWrite(borneIN1, LOW);
    digitalWrite(borneIN2, HIGH);
    digitalWrite(miborneIN1, LOW);
    digitalWrite(miborneIN2, HIGH);

    digitalWrite(borneIN3, LOW);
    digitalWrite(borneIN4, HIGH);
    digitalWrite(miborneIN3, LOW);
}

```

```

digitalWrite(miborneIN4, HIGH);
delay(3000);

//désactive moteurs
digitalWrite(borneENA, LOW);
digitalWrite(miborneENA, LOW);
digitalWrite(miborneENB, LOW);
digitalWrite(borneENB, LOW);
delay(3000);
}

//active les 4 moteurs en sens avant
void gauche(){
    digitalWrite(borneENA, HIGH);      // Active l'alimentation du moteur 1
    digitalWrite(miborneENA, HIGH);    // Active l'alimentation du moteurs 1
    digitalWrite(miborneENB, HIGH);    // Active l'alimentation du moteur 2
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 2

    //sens des moteurs
    digitalWrite(borneIN1, HIGH);
    digitalWrite(borneIN2, LOW);
    digitalWrite(miborneIN1, HIGH);
    digitalWrite(miborneIN2, LOW);

    digitalWrite(borneIN3, HIGH);
    digitalWrite(borneIN4, LOW);
    digitalWrite(miborneIN3, HIGH);
    digitalWrite(miborneIN4, LOW);
    delay(3000);

    //désactive moteurs
    digitalWrite(borneENA, LOW);
    digitalWrite(miborneENA, LOW);
    digitalWrite(miborneENB, LOW);
    digitalWrite(borneENB, LOW);
    delay(3000);
}

//active les 4 moteurs en sens derriere
void derriere(){
    digitalWrite(borneENA, HIGH);      // Active l'alimentation du moteur 1
    digitalWrite(miborneENA, HIGH);    // Active l'alimentation du moteurs 1
    digitalWrite(miborneENB, HIGH);    // Active l'alimentation du moteur 2
    digitalWrite(borneENB, HIGH);      // Active l'alimentation du moteur 2

    //sens des moteurs
    digitalWrite(borneIN1, HIGH);
    digitalWrite(borneIN2, LOW);
    digitalWrite(miborneIN1, HIGH);
    digitalWrite(miborneIN2, LOW);

    digitalWrite(borneIN3, LOW);
    digitalWrite(borneIN4, HIGH);
    digitalWrite(miborneIN3, LOW);
    digitalWrite(miborneIN4, HIGH);
    delay(1000);

    //désactive moteurs
    digitalWrite(borneENA, LOW);
    digitalWrite(miborneENA, LOW);
    digitalWrite(miborneENB, LOW);
    digitalWrite(borneENB, LOW);
    delay(3000);
}

/*
void moveMotors(int speedRF, int speedLF, int speedRR, int speedLR, byte dircontrol) {

    // Moves all 4 motors
    // Directions specified in direction byte

    // Right Front Motor
    digitalWrite(MF_AI1, bitRead(dircontrol, 7));
    digitalWrite(MF_AI2, bitRead(dircontrol, 6));
    ledcWrite(mtrRFPwmchannel, abs(speedRF));

    // Left Front Motor
    digitalWrite(MF_BI1, bitRead(dircontrol, 5));
    digitalWrite(MF_BI2, bitRead(dircontrol, 4));
    ledcWrite(mtrLFPwmchannel, abs(speedLF));
}

```

```

// Right Rear Motor
digitalWrite(MR_AI1, bitRead(dircontrol, 3));
digitalWrite(MR_AI2, bitRead(dircontrol, 2));
ledcWrite(mtrRRpwmchannel, abs(speedRR));

// Left Rear Motor
digitalWrite(MR_BI1, bitRead(dircontrol, 1));
digitalWrite(MR_BI2, bitRead(dircontrol, 0));
ledcWrite(mtrLRpwmchannel, abs(speedLR));
}

void stopMotors() {

    // Stops all motors and motor controllers
    ledcWrite(mtrRFpwmchannel, 0);
    ledcWrite(mtrLFpwmchannel, 0);
    ledcWrite(mtrRRpwmchannel, 0);
    ledcWrite(mtrLRpwmchannel, 0);

    digitalWrite(MF_AI1, 0);
    digitalWrite(MF_AI2, 0);
    digitalWrite(MF_BI1, 0);
    digitalWrite(MF_BI2, 0);
    digitalWrite(MR_AI1, 0);
    digitalWrite(MR_AI2, 0);
    digitalWrite(MR_BI1, 0);
    digitalWrite(MR_BI2, 0);
}
*/

```