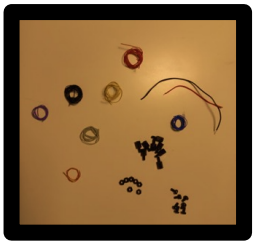
Projet Arduino
Voiture robotisée & contrôlée par Bluetooth

Bernard Anwar (en collaboration avec Chillat Quentin)

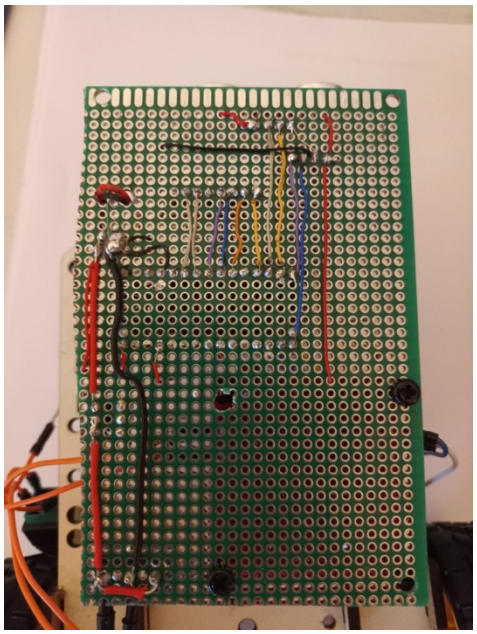
Rapport Semaine 8 (le 22/02/2024)

Matériels utilisés

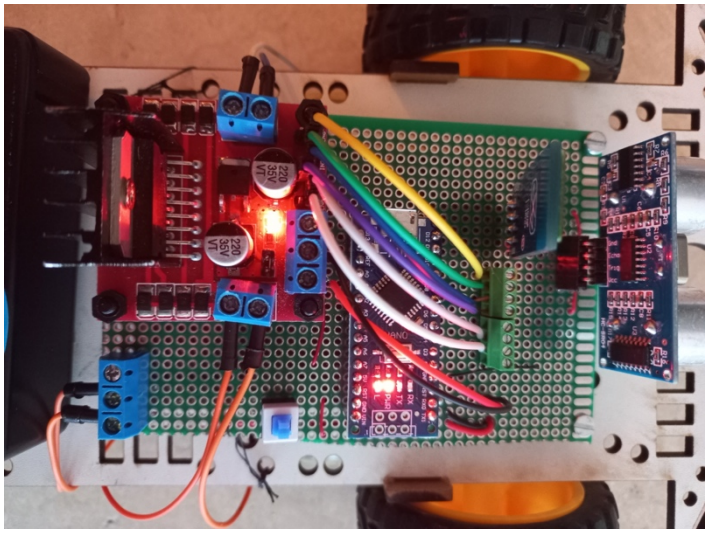
- Des fils
- Visserie et entretoises



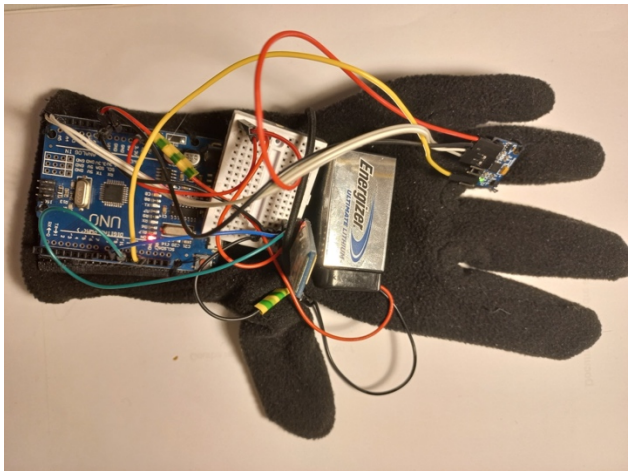
Câblage et Assemblage



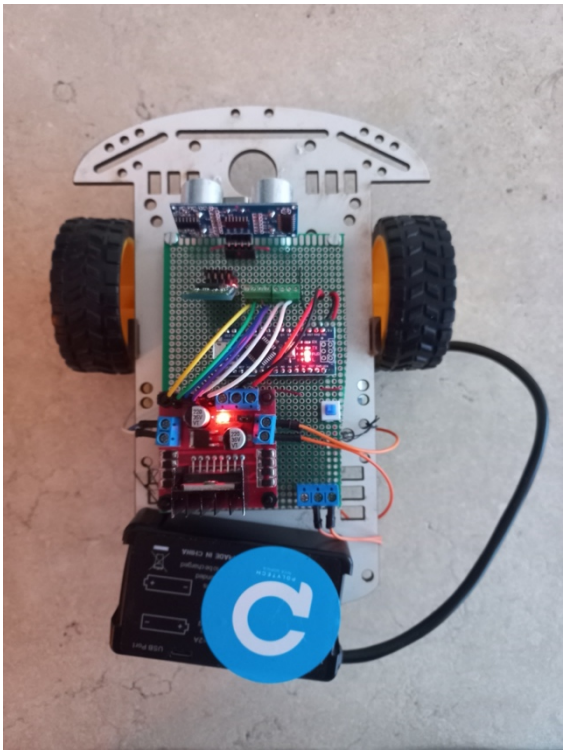
Vue dessous



Vue dessus

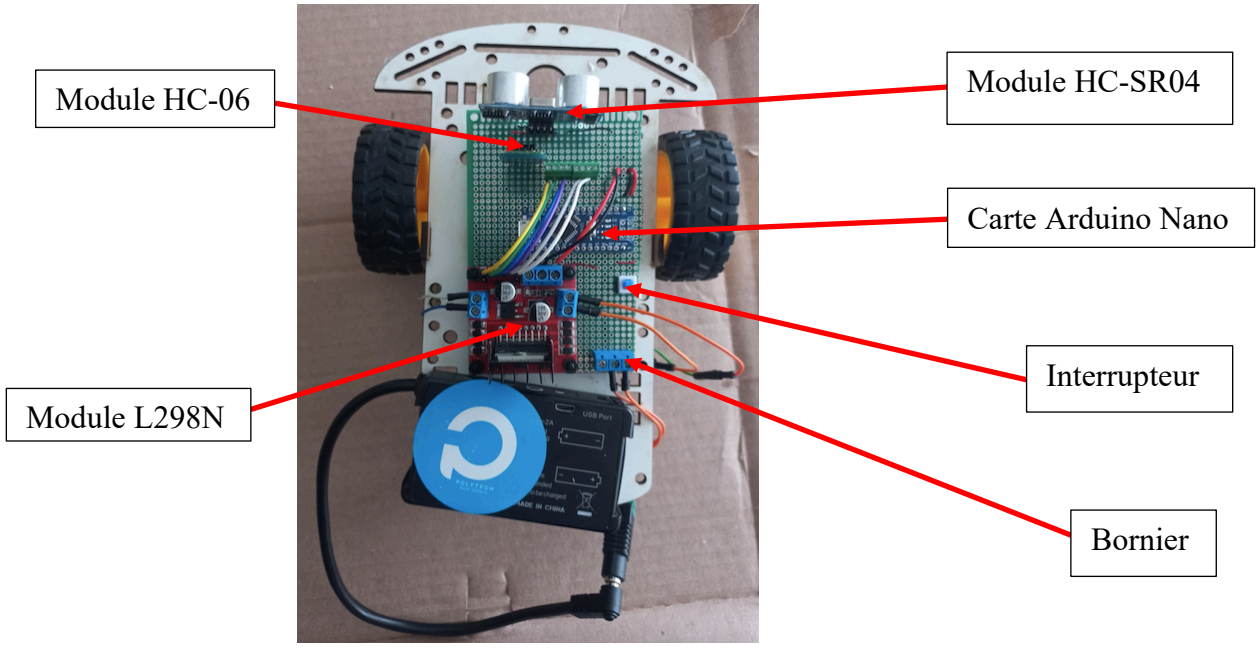


Gant pour diriger



Vue dessus

Répartition des composants sur la carte PCB



Tâches effectuées :

Rappel : Nous remplaçons le montage composé de la carte Arduino, implémenté sur la voiture, par un montage avec une carte PCB double face, pour pouvoir éventuellement habiller la voiture d'une coque.

Cette séance a été principalement dédiée à la vérification des soudures, des connexions et des modules, pour résoudre le problème du non fonctionnement du montage installé sur la voiture. Nous avons commencé par corriger les soudures de la séance précédente. Cela a nécessité de la précision et de la patience pour rétablir les connexions entre les différents composants. Nous avons remarqué qu'il manquait la connexion de la masse sur la carte Arduino. Nous l'avons donc également soudée. Nous avons ensuite corrigé l'inversion entre le VCC et le GND du module HC-SR04.
De plus, le circuit était en court-circuit entre la masse et l'alimentation. Pour résoudre cela, nous avons dû ajuster les soudures et indiquer clairement la séparation entre elles.
Après toutes ses manipulations, le système ne fonctionnait toujours pas.
Nous avons alors isolé chaque composant pour déceler l'origine du problème.
Nous nous sommes rendu compte que les modules HC-05 et HC-06 arrivaient à communiquer sans la batterie du gant (pile). Nous avons donc remplacé la pile du gant par une autre pile. Malheureusement, la carte Arduino a grillé. Nous avons dû la changer. Pour finir, nous avons testé notre nouvelle installation qui a enfin fonctionnée.

Tableau de câblage des différents modules à la carte Arduino Nano

Module	Pin module	N° I/O Arduino Nano
L298N (Pont en H)	ENA	3()
	IN1	4
	IN2	5
	IN3	6
	IN4	7
	ENB	9()
HC-SR04	Trig	10
	Echo	11
HC-06	RX	12
	TX	13

Programme de communication

Codage récepteur (HC-06)

```
#include <SoftwareSerial.h>
SoftwareSerial bluetoothSerial(13, 12); // RX, TX (Connectez RX du HC-06 à 10 et TX à 11)

int16_t accelerometer_data[2] = {0};
String receivedString = "";            // Chaîne pour stocker les données reçues

const int trig=10;
const int echo=11;

float distance;
float lecture_echo;

int ENA=3; // Moteur DROIT
int IN1=4;
int IN2=5;

int ENB=9; //Moteur GAUCHE
int IN3=6;
int IN4=7;

void setup() {
  Serial.begin(38400);
  bluetoothSerial.begin(38400); // Initialisez le port série pour la communication Bluetooth

  pinMode(ENA,OUTPUT);
  pinMode(ENB,OUTPUT);
  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);

  analogWrite(ENA,255);
  analogWrite(ENB,255);

}

void loop() {

  pinMode(trig,OUTPUT);   //trig sur sortie 10
  pinMode(echo,INPUT);    //echo sur entrée 10

  digitalWrite(trig,HIGH);  //on allume le signal
  delayMicroseconds(10);
  digitalWrite(trig,LOW);   //on eteint le signal
  lecture_echo=pulseIn(echo,HIGH);    //on enregistre le temps d'allumage du signal
  distance=lecture_echo*0.017;    //on converti le temps en distance
distance=(temps*2)/0.034=temps*0.017




  // Vérifier s'il y a des données disponibles sur Bluetooth
  while (bluetoothSerial.available()) {
    char receivedChar = bluetoothSerial.read();

    // Si une virgule ou un saut de ligne est reçu, cela signifie la fin d'une valeur
    if (receivedChar == ',' || receivedChar == '\n') {
      // Convertir la chaîne reçue en entier
      int receivedValue = receivedString.toInt();

      // Mettre à jour les valeurs du tableau d'accéléromètre à partir des données reçues
      for (int i = 0; i < 2; i++) {
        if (accelerometer_data[i] == 0) {
          accelerometer_data[i] = receivedValue;
          break;
        }
      }

      // Réinitialisez la chaîne pour la prochaine valeur
      receivedString = "";

      // Si le tableau est complètement construit, afficher chaque valeur
      if (accelerometer_data[0] != 0 && accelerometer_data[1] != 0 ) {
        Serial.print("Tableau d'accéléromètre complet : ");
        Serial.print(accelerometer_data[0]);
        Serial.print(", ");
        Serial.print(accelerometer_data[1]);
```

```
        Serial.print(" Distance :");
        Serial.println(distance);

      if(distance>5){
          if(accelerometer_data[0]<-10000){ //avancer
            Serial.println(" Avance");
            digitalWrite(IN1,HIGH);
            digitalWrite(IN2,LOW);
            digitalWrite(IN3,LOW);
            digitalWrite(IN4,HIGH);
            }
          else if(accelerometer_data[0]>10000){//reculer
            Serial.println(" Recule");
            digitalWrite(IN1,LOW);
            digitalWrite(IN2,HIGH);
            digitalWrite(IN3,HIGH);
            digitalWrite(IN4,LOW);
            }

          else if(accelerometer_data[0]<-10000){//tourner a gauche
            Serial.println(" Tourne a gauche");
            digitalWrite(IN1,HIGH);
            digitalWrite(IN2,LOW);
            digitalWrite(IN3,HIGH);
            digitalWrite(IN4,LOW);
            }

          else if(accelerometer_data[0]>10000){//tourner a droite
            Serial.println(" Tourne a droite");
            digitalWrite(IN1,LOW);
            digitalWrite(IN2,HIGH);
            digitalWrite(IN3,LOW);
            digitalWrite(IN4,HIGH);
            }
          else{ //static
            digitalWrite(IN1,LOW);
            digitalWrite(IN2,LOW);
            digitalWrite(IN3,LOW);
            digitalWrite(IN4,LOW);
            }

            // Réinitialisez le tableau pour la prochaine série de valeurs
            memset(accelerometer_data, 0, sizeof(accelerometer_data));
          }
      else {
        if(accelerometer_data[1]>10000){//reculer
            digitalWrite(IN1,LOW);
            digitalWrite(IN2,HIGH);
            digitalWrite(IN3,HIGH);
            digitalWrite(IN4,LOW);
            }
        else{ //static
            digitalWrite(IN1,LOW);
            digitalWrite(IN2,LOW);
            digitalWrite(IN3,LOW);
            digitalWrite(IN4,LOW);
            }

          // Réinitialisez le tableau pour la prochaine série de valeurs
            memset(accelerometer_data, 0, sizeof(accelerometer_data));
          }
      }
    }
    else {
      // Ajoutez le caractère à la chaîne en cours de réception
      receivedString += receivedChar;
    }
  }
}
```

Codage émetteur (HC-05)

```
#include <Wire.h>
#include <SoftwareSerial.h>

SoftwareSerial bluetoothSerial(10, 11); // RX, TX (Connectez RX du HC-05 à 10 et TX à 11)
#define MPU6050_ADDRESS 0x68        // Adresse I2C du MPU6050
int16_t accelerometer_x, accelerometer_y, accelerometer_z;
char tmp_str[7]; // variable temporaire utilisée dans la fonction de conversion

char* convert_int16_to_str(int16_t i) { // convertit int16 en chaîne de caractères. De plus,
les chaînes résultantes auront la même longueur dans le moniteur de débogage.
  sprintf(tmp_str, "%6d", i);
  return tmp_str;
}

void setup() {
  Serial.begin(38400);
  Wire.begin();
  Wire.beginTransmission(MPU6050_ADDRESS); // Commence une transmission à l'esclave I2C
(carte GY-521)
  Wire.write(0x6B); // Registre PWR_MGMT_1
  Wire.write(0); // réglé à zéro (réveille le MPU-6050)
  Wire.endTransmission(true);
  bluetoothSerial.begin(38400);
}

void loop() {
  // Lire les données de l'accéléromètre (MPU6050)
  Wire.beginTransmission(MPU6050_ADDRESS);
  Wire.write(0x3B); // Adresse du registre de l'accélération X haute
  Wire.endTransmission(false);
  Wire.requestFrom(MPU6050_ADDRESS, 7 * 2, true);

  accelerometer_x = Wire.read() << 8 | Wire.read();
  accelerometer_y = Wire.read() << 8 | Wire.read();
  accelerometer_z = Wire.read() << 8 | Wire.read();

  // Envoyer les données à HC-06 sous forme de chaînes de caractères
  bluetoothSerial.print(convert_int16_to_str(accelerometer_x));
  bluetoothSerial.print(",");
  bluetoothSerial.println(convert_int16_to_str(accelerometer_y));
  /*bluetoothSerial.print(",");
  bluetoothSerial.println(convert_int16_to_str(accelerometer_z));*/

  // Attendre un court délai pour que les données soient envoyées
  delay(500);
}
```