RAPPORT DE STAGE

*IUT DE CACHANT

Université PARIS SUD

De 2 juin à 15 juillet

GEORGES DAHER

Mon stage portera sur le développement d'une partie robotique – MECATRONIQUE au sein du **CRIIP (Centre de ressources et d'innovation pédagogiques de l'IUT).

Mon stage est divisé en deux parties:

1. Language V+ (simulation sur robot baby foot) .

2. robot autonome et capable d'aller d'un point A a un point B tout en évitant des obstacles, présent sur la piste à des emplacements aléatoires. (Arduino)

Georges DAHER

Bs Génie Mécatronique

REMERCIMENTS

Je tiens à remercier dans un premier temps, Pr. Souhil MEGHERBI, Directeur de l'IUT de Cachan, Université Paris sud, France de m'avoir acceptés en tant que stagiaires au sein de l'université.

Je remercie également madame Pascal VAREIL pour son accueil chaleureux.

Je remercie également Mr. Bertrand MANUEL, Mr. DEXTER, pour leurs soutiens techniques.

D'une façon plus générale, je remercie l'université, pour l'intérêt qu'ils mon porté tout au long du stage ainsi que pour leur aide et précisions.

Je remercie de même, mon doyen de faculté Dr. Elias KHALIL pour son encadrement pendant celui-ci.

*IUT DE CACHAN

Créé en 1966 à la suite du Décret du 7 janvier 1966 instaurant les Instituts Universitaires de Technologie, l'IUT de Cachan a toujours conservé une caractéristique essentielle : il propose des spécialités industrielles articulées autour du Génie Electrique et Informatique Industrielle et du Génie Mécanique et Productique.

En restant fidèle à ce choix, l'IUT de Cachan a su s'adapter à l'évolution de la technologie et aux demandes des entreprises. La liste des spécialités actuelles à l'IUT le montre :

électronique, électrotechnique, télécommunications, informatique industrielle, mécatronique, automatique et automatismes, robotique, mécanique, productique, CAO, CFAO et DAO.

L'IUT a aussi su adapter ses cursus et ses méthodes pédagogiques à la demande des publics concernés. Il propose maintenant des parcours de formation à temps plein, avec un stage final (statut étudiant) ou par apprentissage (statut apprenti), de formation ouverte à distance, de formation continue tout au long de la vie avec validation des acquis de l'expérience et alternance adaptée à la situation des stagiaires.

Les diplômes préparés à l'IUT s'inscrivent dans le schéma européen Licence Master Doctorat (LMD) des Universités, soit au niveau L : DUT (bac +2) et Licence Professionnelle (bac +3), soit au niveau M : Diplôme d'ingénieur en partenariat préparé par l'apprentissage dans le cadre de l'IFIPS (filière ingénieur de l'UPS).

Pour répondre aux missions qui lui sont assignées, l'IUT est structuré en 3 départements d'enseignement disposant d'équipes éducatives pluridisciplinaires :

Génie Electrique & Informatique Industrielle 1

Génie Electrique & Informatique Industrielle 2

Génie Mécanique et Productique

Ces Départements sont appuyés par un ensemble de services communs. Au total c'est plus de 160 personnels titulaires de l'Education Nationale à plein temps qui sont au service de la réussite des quelques 1100 étudiants et stagiaires qui fréquentent, chaque année, l'IUT de Cachan.

POUR PLUS D'INFORMATIONS

 scolarite.iut-cachan@u-psud.fr

 01.41.24.11.00

IUT de Cachan
9 avenue de la division Leclerc
94230 Cachan

** CRIIP

Le CRIIP est un Centre de Recherche et d'Ingénierie Industrielle ou Pédagogique. Il a le statut d'un centre de ressources, a été créé en novembre 2000, regroupe et valide les multiples actions de recherche et de collaborations industrielles ou pédagogiques menées par les trois départements de l'IUT de Cachan : GEII1, GEII2 et GMP.

Le CRIIP travaille sur des projets en collaborations avec des partenaires industriels. Ces derniers permettent aux étudiants de l'IUT de s'impliquer directement, dans le cadre de projets tuteurés et des travaux de réalisation, au développement et à la réalisation de projets industriels et technologiques. Le CRIIP fournit ainsi la logistique nécessaire à chaque projet.

Les trois départements GE1, GE2 et GMP se répartissent quatre activités :

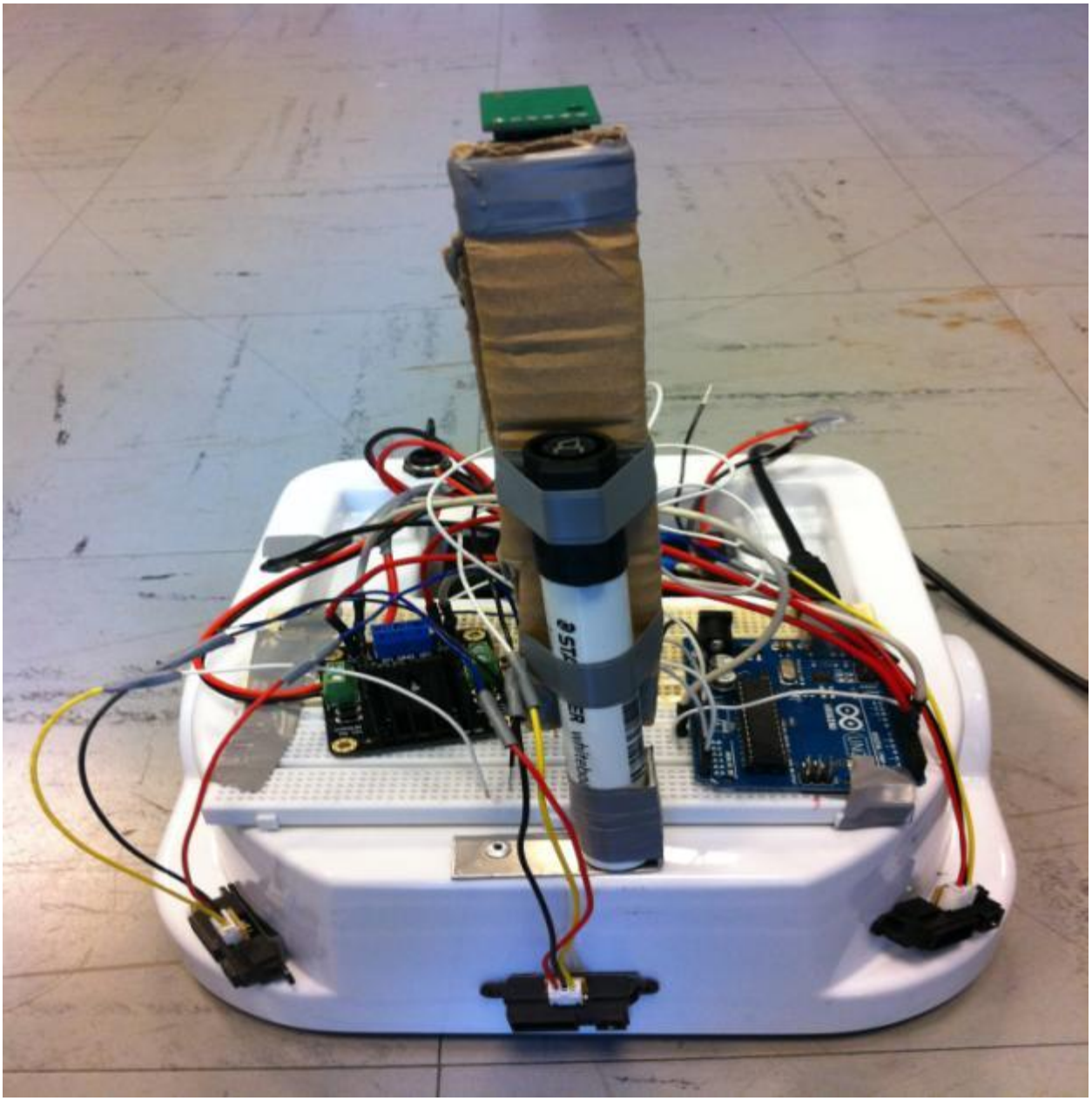
- La recherche fondamentale et appliquée,
- La recherche pédagogique,
- Le transfert de technologie,

La coopération pédagogique internationale.

Sommaire

Fin

B. ROBOT AUTONOME



Arduino introduction

Le robot autonome est muni d'une carte arduino (son cerveau).

C'est quoi arduino :

En quelques mots :

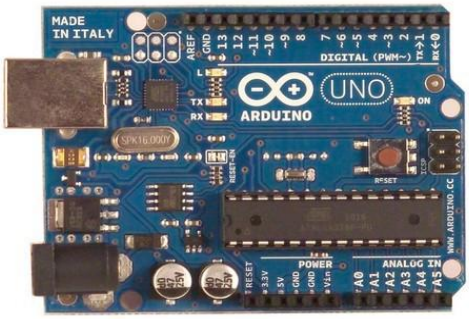
Arduino = 1 carte a microcontrôleur + 1 outil de développement + 1

Communauté active.

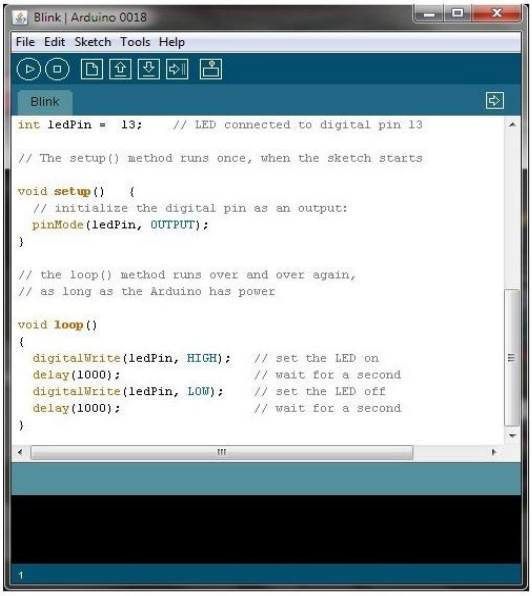
Le logiciel et le matériel sont open-source.

La « philosophie »

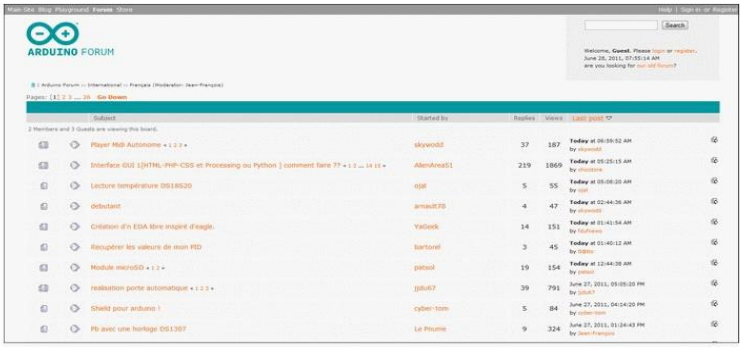
L'idée est d'utiliser la carte Arduino comme un macro-composant dans des applications de
Prototypage électronique. Le concepteur n'a plus qu'à développer des interfaces et
Programmer le macro-composant pour réaliser son application !



1 carte à micro-contrôleur



1 outil de développement



1 communauté active

Les avantages

- Pas cher !
- Environnement de programmation clair et simple.
- Multiplateforme : tourne sous Windows, Macintosh et Linux.
- Nombreuses librairies disponibles avec diverses fonctions implémentées.
- Logiciel et matériel open source et extensible.
- Nombreux conseils, tutoriaux et exemples en ligne (forums, site perso etc...)
- Existence de « shield » (boucliers en français) : ce sont des cartes supplémentaires

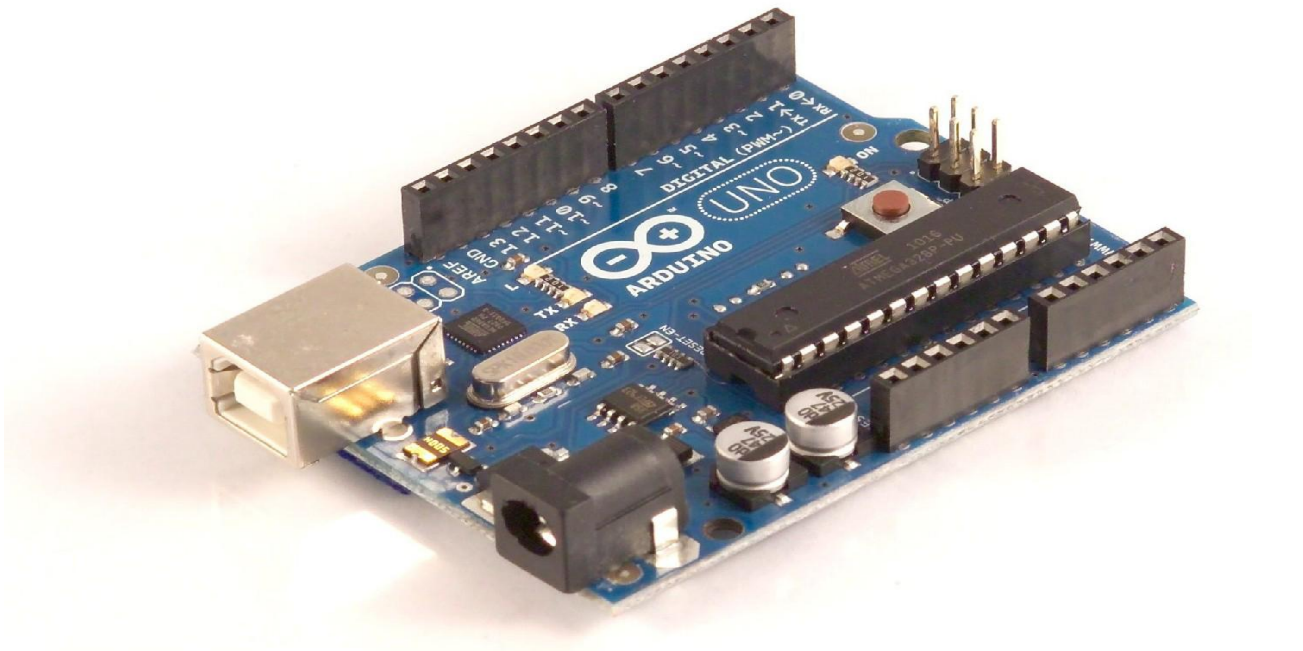
qui se connectent sur le module Arduino pour augmenter les possibilités comme par

exemple : afficheur graphique couleur, interface Ethernet, GPS, etc...

Par sa simplicité d'utilisation, Arduino est utilisé dans beaucoup d'applications comme
l'électronique industrielle et embarquée, le modélisme, la domotique mais aussi dans des
domaines différents comme l'art contemporain ou le spectacle !

La carte Arduino uno

Il existe plusieurs types de cartes, j'ai commencé avec une carte Arduino uno (carte
basique, au dimensions voisines de celle d'une carte bancaire).

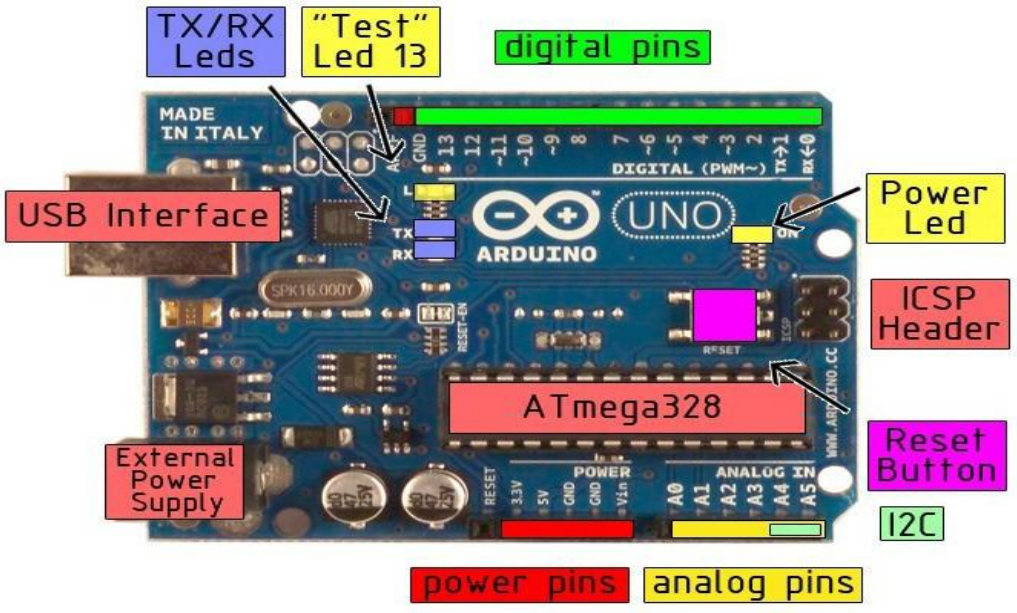


La carte Arduino uno

Caractéristiques de la carte Arduino uno :

Micro contrôleur : ATmega328
Tension d'alimentation interne = 5V
tension d'alimentation (recommandée)= 7 à 12V, limites =6 à 20 V
Entrées/sorties numériques : 14 dont 6 sorties PWM
Entrées analogiques = 6
Courant max par broches E/S = 40 mA
Courant max sur sortie 3,3V = 50mA
Mémoire Flash 32 KB dont 0.5 KB utilisée par le bootloader
Mémoire SRAM 2 KB
mémoire EEPROM 1 KB
Fréquence horloge = 16 MHz
Dimensions = 68.6mm x 53.3mm

La carte s'interface au PC par l'intermédiaire de sa prise USB.
La carte s'alimente par le jack d'alimentation (utilisation autonome) mais peut être
alimentée par l'USB (en phase de développement par exemple).
Louis

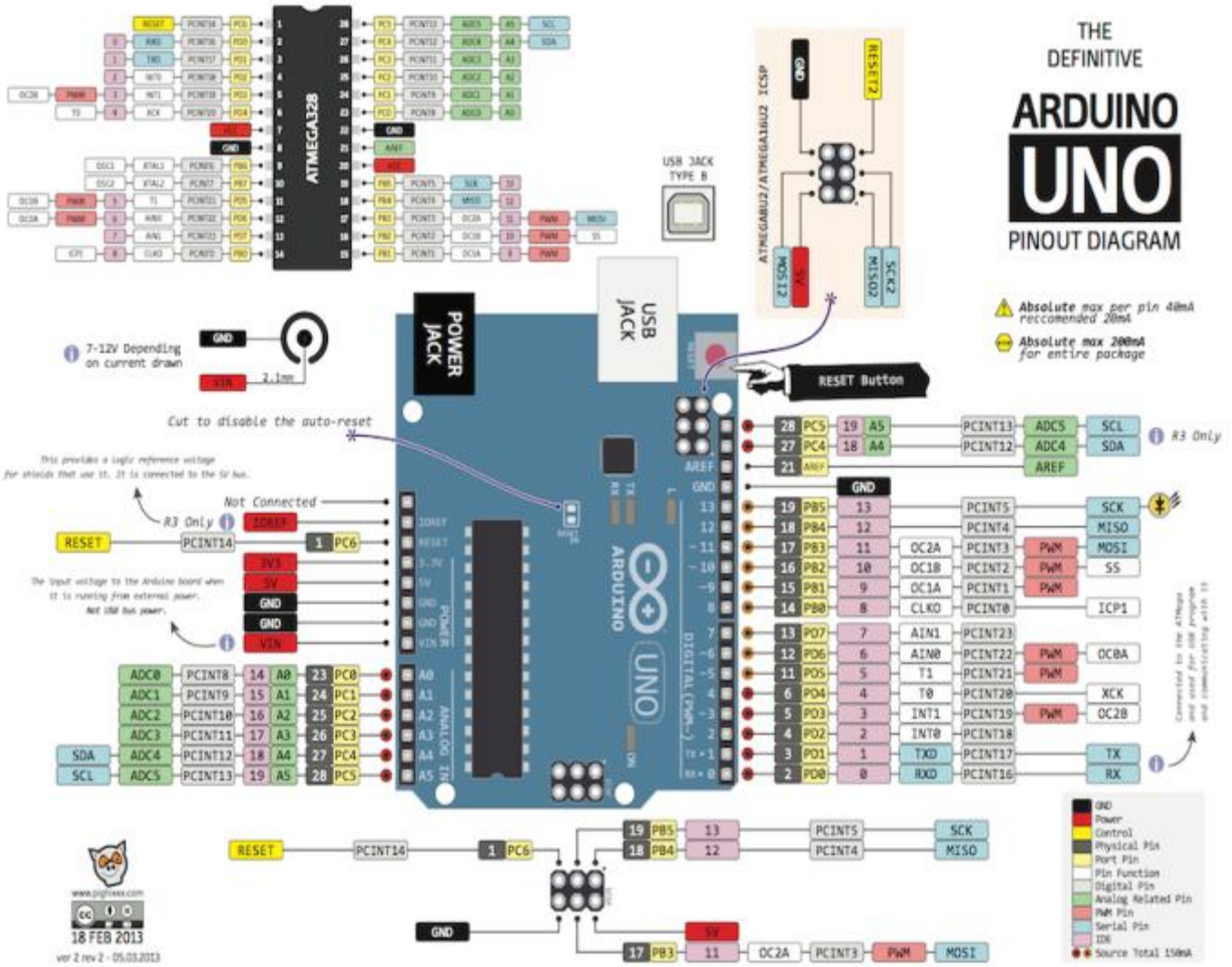


Les « shields »

Il existe de nombreux shields que l'on traduit parfois dans les documentations par «boucliers ». Personnellement, le terme « extension » me paraitrait plus approprié. Un « shield » Arduino est une petite carte qui se connecte sur une carte Arduino pour augmenter ses fonctionnalités. Quelques exemples de « shields » :

- Afficheur graphique
- Ethernet et carte SD
- GPS
- Carte de prototypage (type labdec)
- etc...

Louis



Développement d'un projet

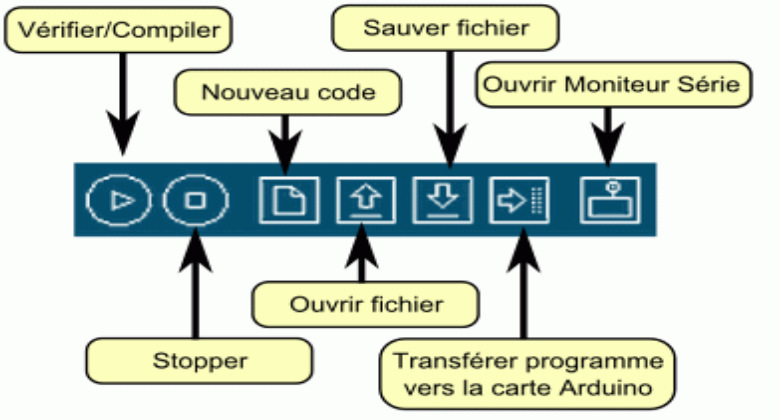
Le développement sur Arduino est très simple :

- on code l'application : Le langage Arduino est basé sur les langages C/C++ , avec des fonctions et des librairies spécifiques à Arduino (gestions des e/s).
- on relie la carte Arduino au PC et on transfert le programme sur la carte,
- on peut utiliser le circuit !

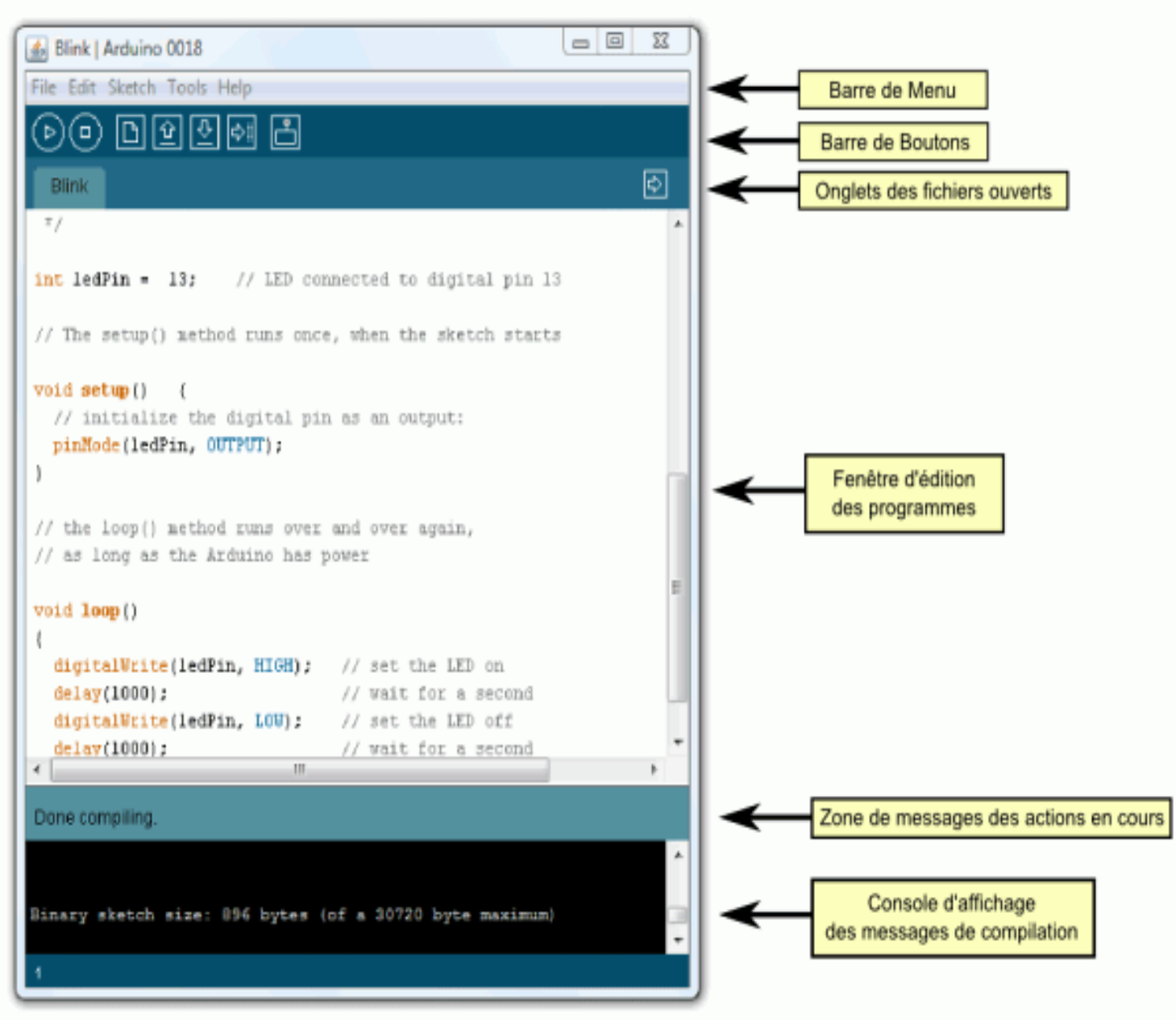
Le logiciel de programmation des modules Arduino est une application Java multiplateformes (fonctionnant sur tout système d'exploitation), servant d'éditeur de code et de compilateur, et qui peut transférer le firmware (et le programme) au travers de la liaison série (RS232, Bluetooth ou USB selon le module).

Le logiciel est très simple à prendre en main, il existe de très bon tutoriaux très bien faits avec même des explications en français. De très nombreux exemples sont fournis.

Les fichiers exemples sont vraiment bien documentés et permettent de coder des choses très compliquées sans trop d'efforts. Les bibliothèques fournies permettent d'utiliser des composants complexes très simplement en quelques lignes très claires (afficheur ou liaison SPI etc..).



détail de la barre de boutons



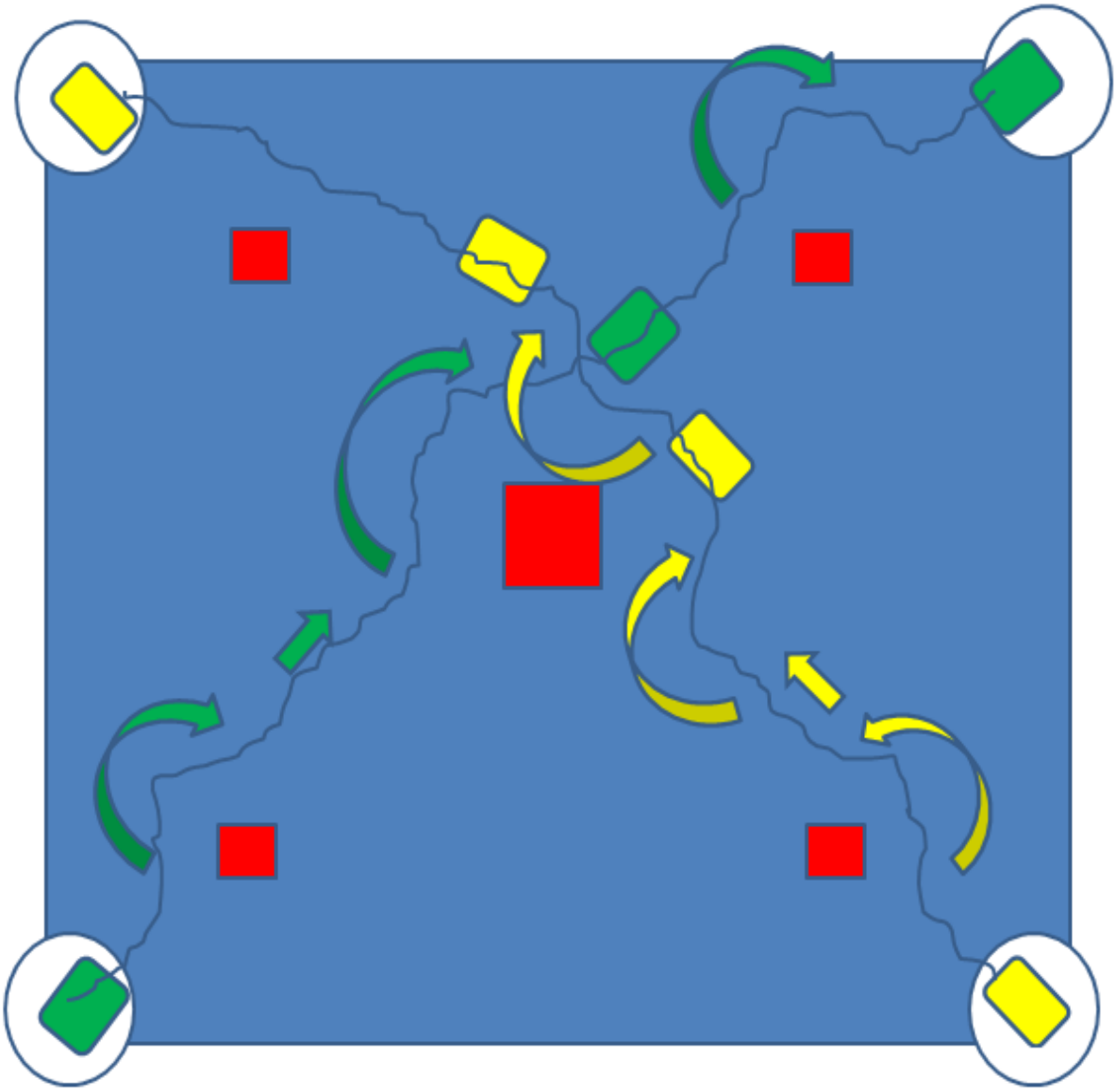
Liens

Le site Arduino : http://www.arduino.cc

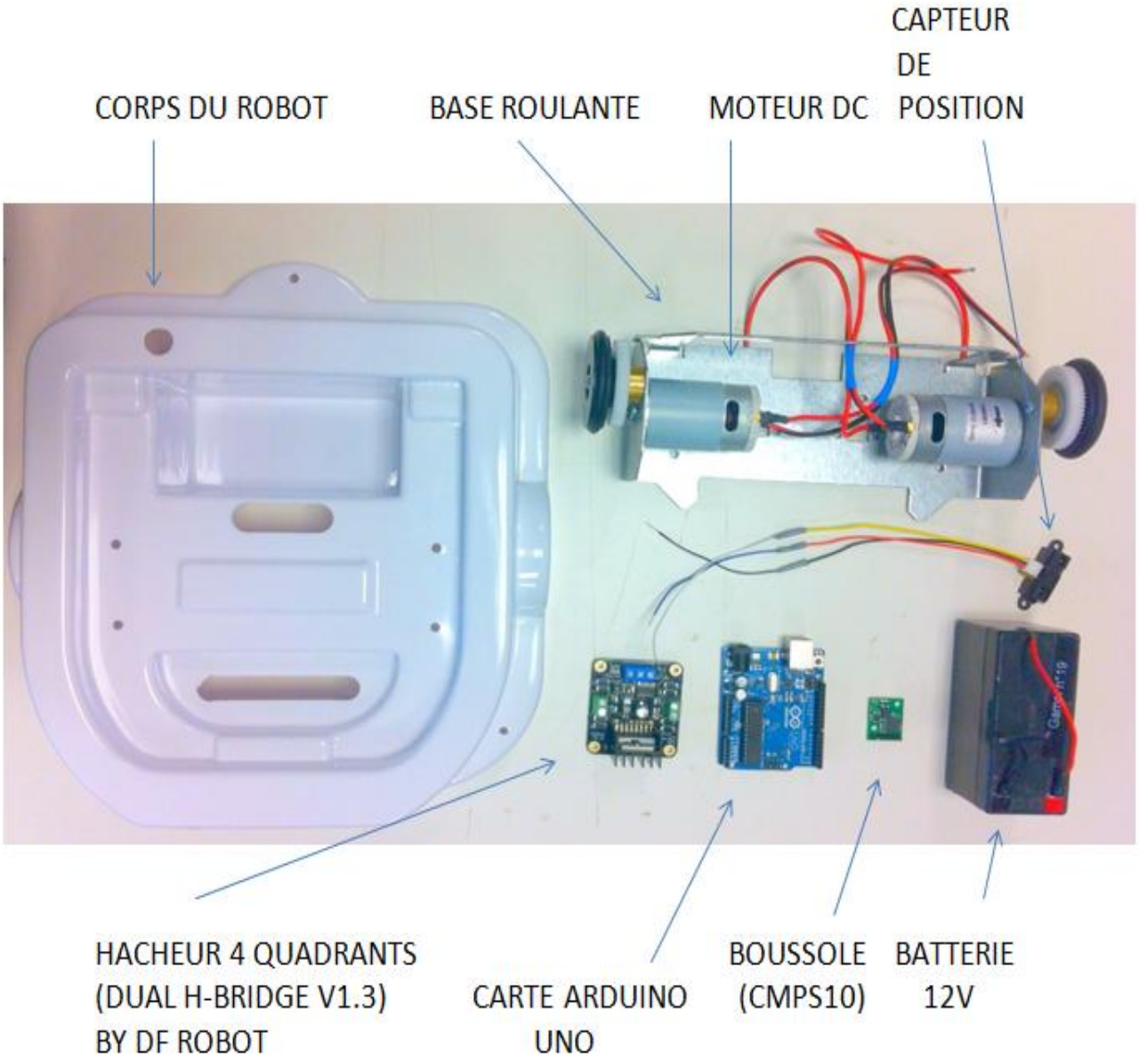
Traduit en français (partiellement) : http://www.arduino.cc/fr

2) CAHIER DES CHARGES

Les règles du concours stipulent que le robot doit parcourir la diagonale d'une piste carrée de 8m par 8m, le robot se dirige vers son point d'arrivée {à l'aide d'une boussole située sur celui-ci.
La piste sera également parsemée d'obstacle, le robot devra donc les éviter puis rejoindre son point d'arrivée.



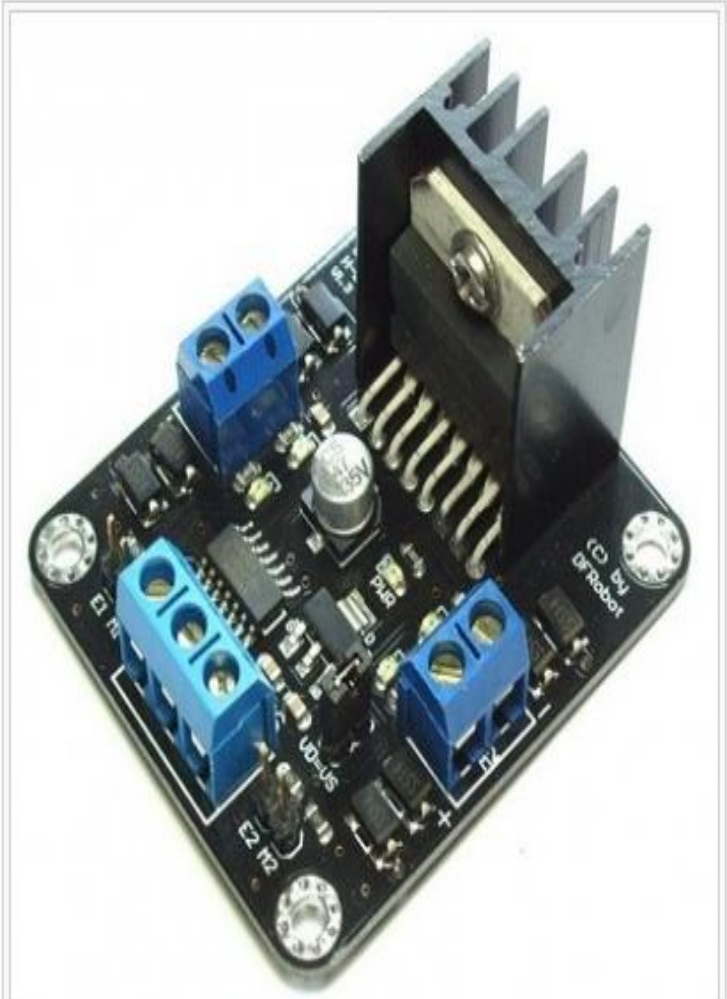
3) LES COMPOSANTS DU ROBOT



Le hacheur (dual h- bridge)

Introduction

This is a 4.8-46V, 2A Dual Motor Controller which is the revised version of the DF-MDV1.0. Its performance has been improved greatly. It can bear larger current due to the increased haetsink dissipation. It is easy to control, using LGS's outstanding high-power motor driver chip, the L298N. This chip allows for direct drive of two bi-directional DC motors, and incorporates high-speed short diodes for protection. Drive current up to 2A per motor output. The driver uses a broad-brush design to reduce wire resistance.



MD1.3 2A Dual Motor Controller (SKU: DRI0002)

Specifications

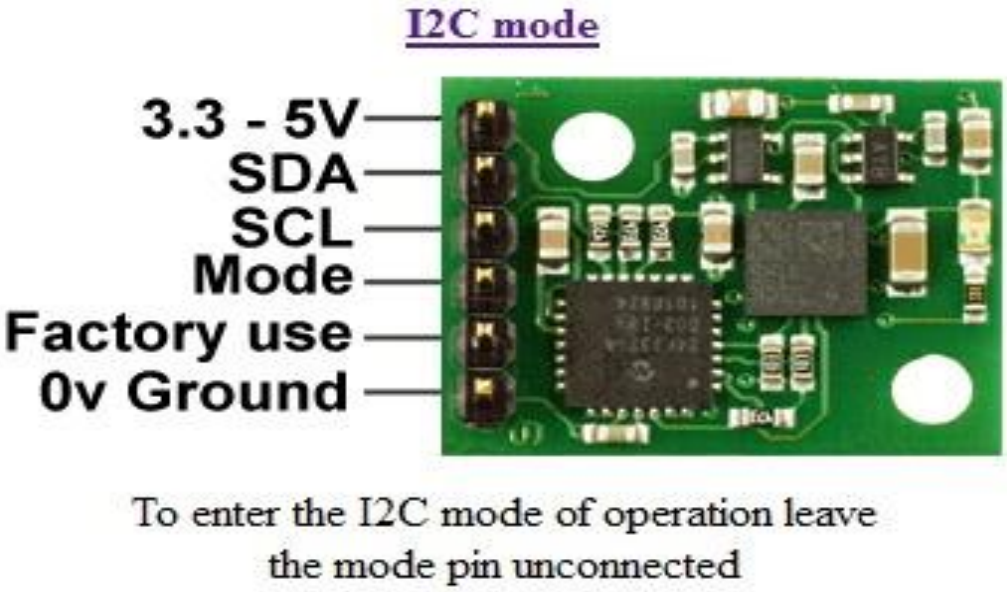
- The logic part of the input voltage: 6 ~ 12V
- Driven part of the input voltage Vs: 4.8 ~ 46V
- The logical part of the work current Iss: 36mA
- Drive part of the operating current Io: 2A
- Maximum power dissipation: 25W (T = 75 degree Celsius)
- Control signal input level:
- High level: 2.3V = Vin = Vss
- Low:-0.3V = Vin = 1.5V
- Operating temperature: -25 degree Celsius ~ +130 degree Celsius
- Drive Type: Dual high-power H-bridge driver
- Module Size: 47 mm × 53mm
- Module Weight: About 29g

La boussole (CMPS10)

Présentation

Le module CMPS10 est une boussole d'inclinaison compensée. Employant un magnétomètre 3 axes et un accéléromètre à 3 axes et un puissant processeur 16-bit, le CMPS10 a été conçu pour éliminer les erreurs causées par l'inclinaison de la carte. Le CMPS10 produit un résultat de 0-3599 représentant 0 à 359,9 ou de 0 à 255. La sortie des trois capteurs de mesure de composants z du champ magnétique X, Y et, en même temps que le tangage et le roulis sont utilisés pour calculer le palier, chacun de ces composants sont également disponibles dans la forme brute. Le module CMPS10 nécessite une alimentation à 3,3 - 5 V et dessine une 25mA nominale de courant. Il y a trois façons d'obtenir le roulement du module. Une interface série, une interface I2C ou une sortie PWM.

L'interface I2C est l'interface qu'on doit utiliser pour la relier à la carte arduino.

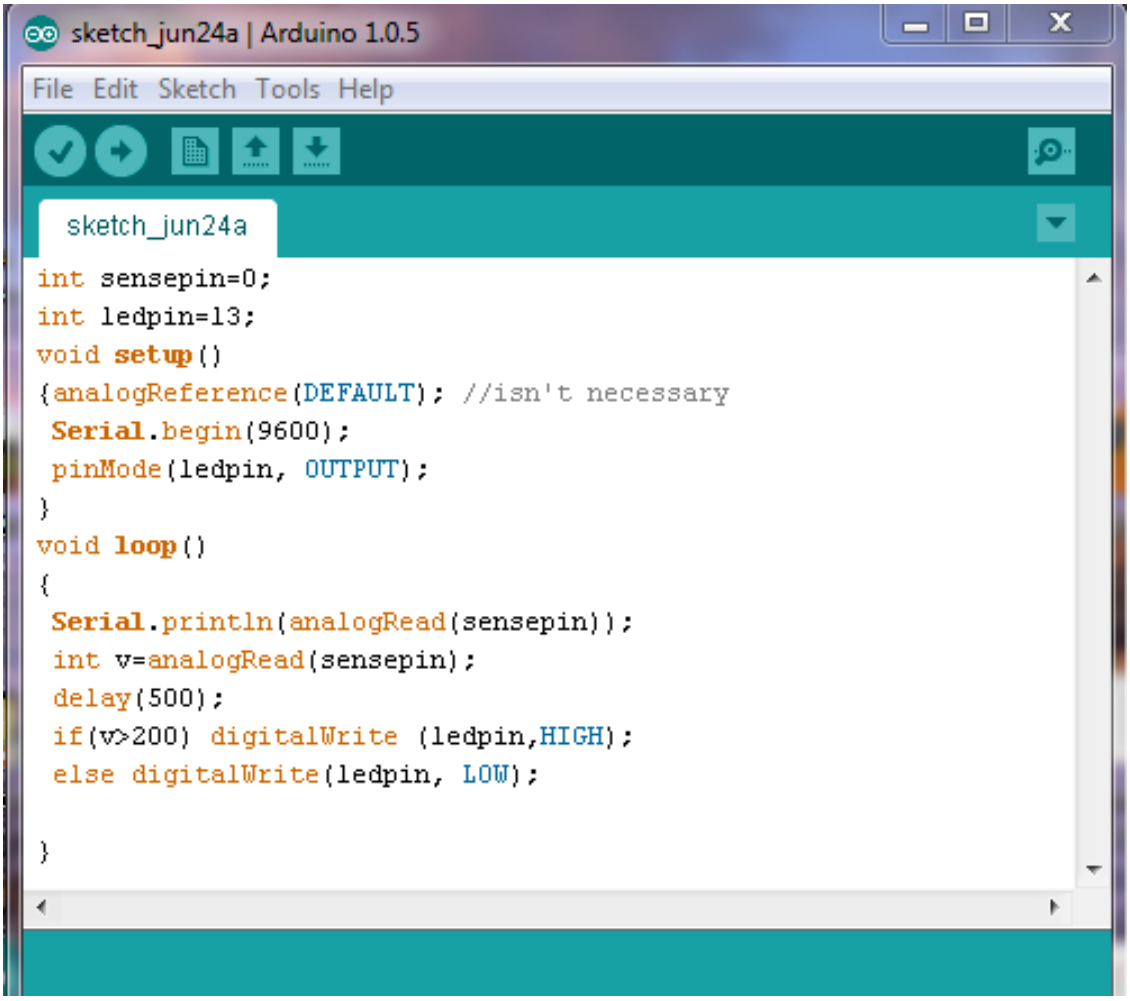


4) LES PROGRAMMES EL LES TESTS REALISES

4.1) TEST CAPTEUR DE POSITION

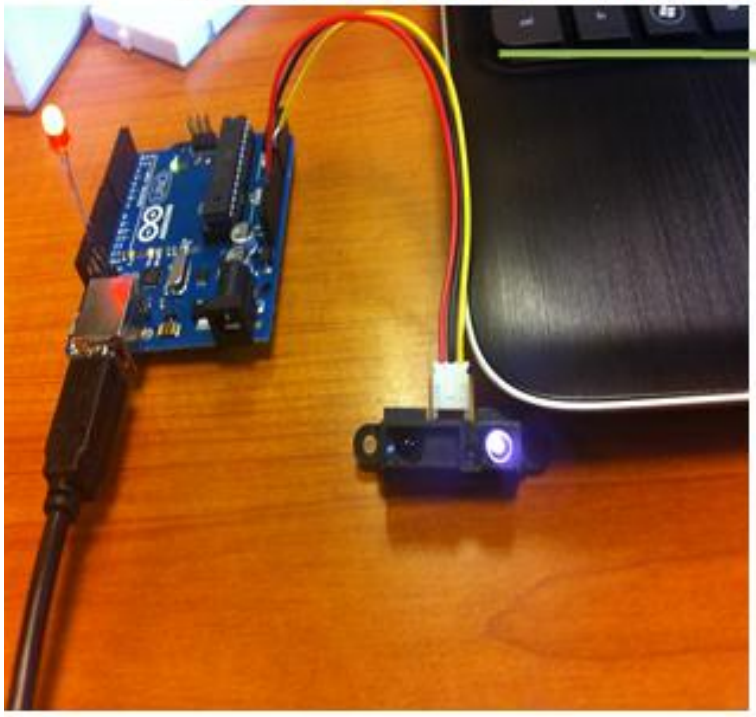
Le capteur de position est un capteur infra rouge qui détecte l'existence et la distance d'un objet.

Le test réalise si dessous consiste à allume une LED si la distance de l'Object par rapport au capteur est moins que 20cm



LA CAMERA ICI EST L'OBJET détecté PAR LE CAPTEUR

CAMERA PROCHE (LA LED S'ALLUME)



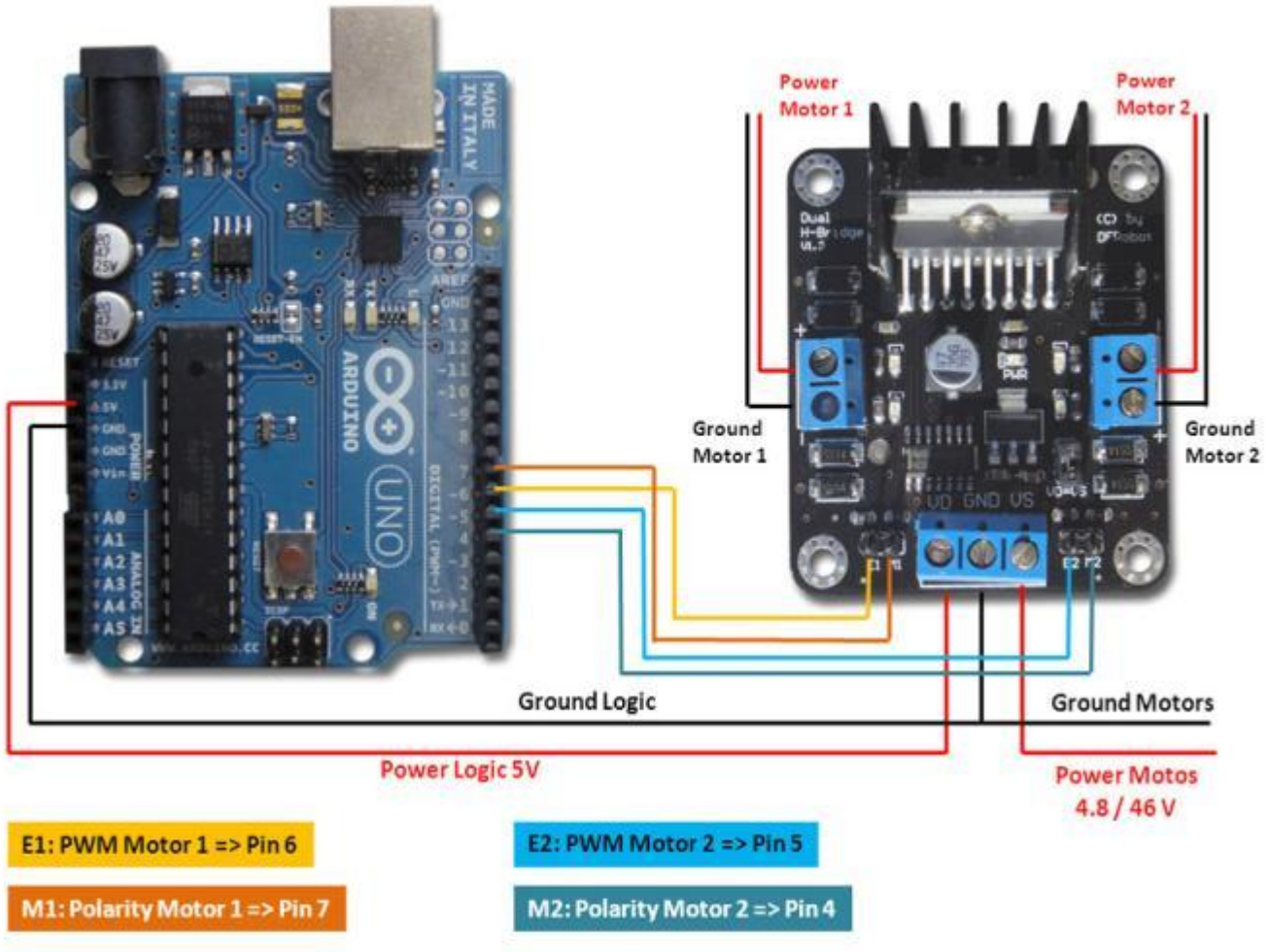
V<200

CAMERA LOIN (LA LED NE S'ALLUME PAS)

V>200

4.2) TEST MOTEURS

Pour pouvoir faire fonctionner les 2 moteurs (changeant leurs directions et leurs vitesses) on a besoin d'un hacheur 4 quadrants.

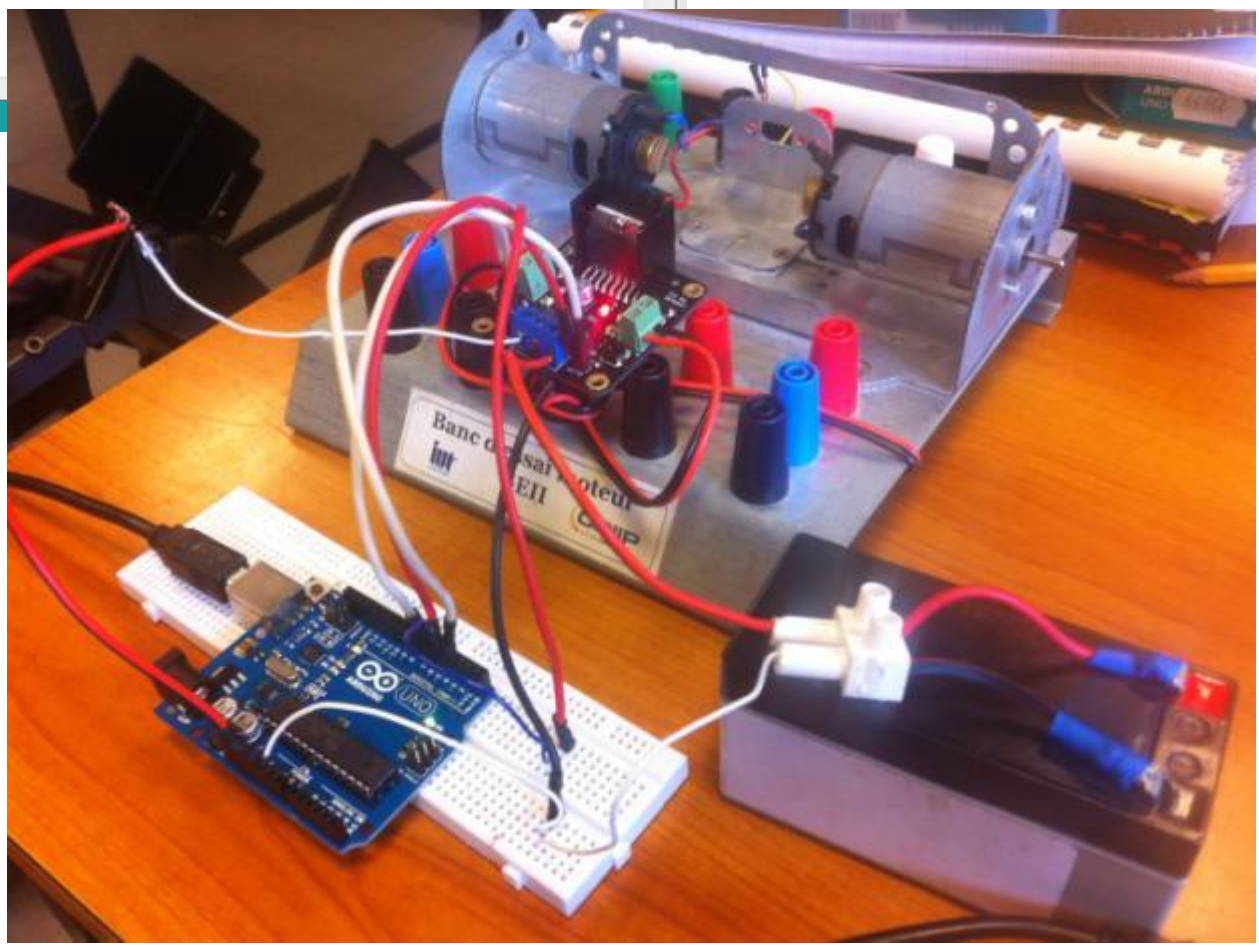


motortest1 | Arduino 1.0.5

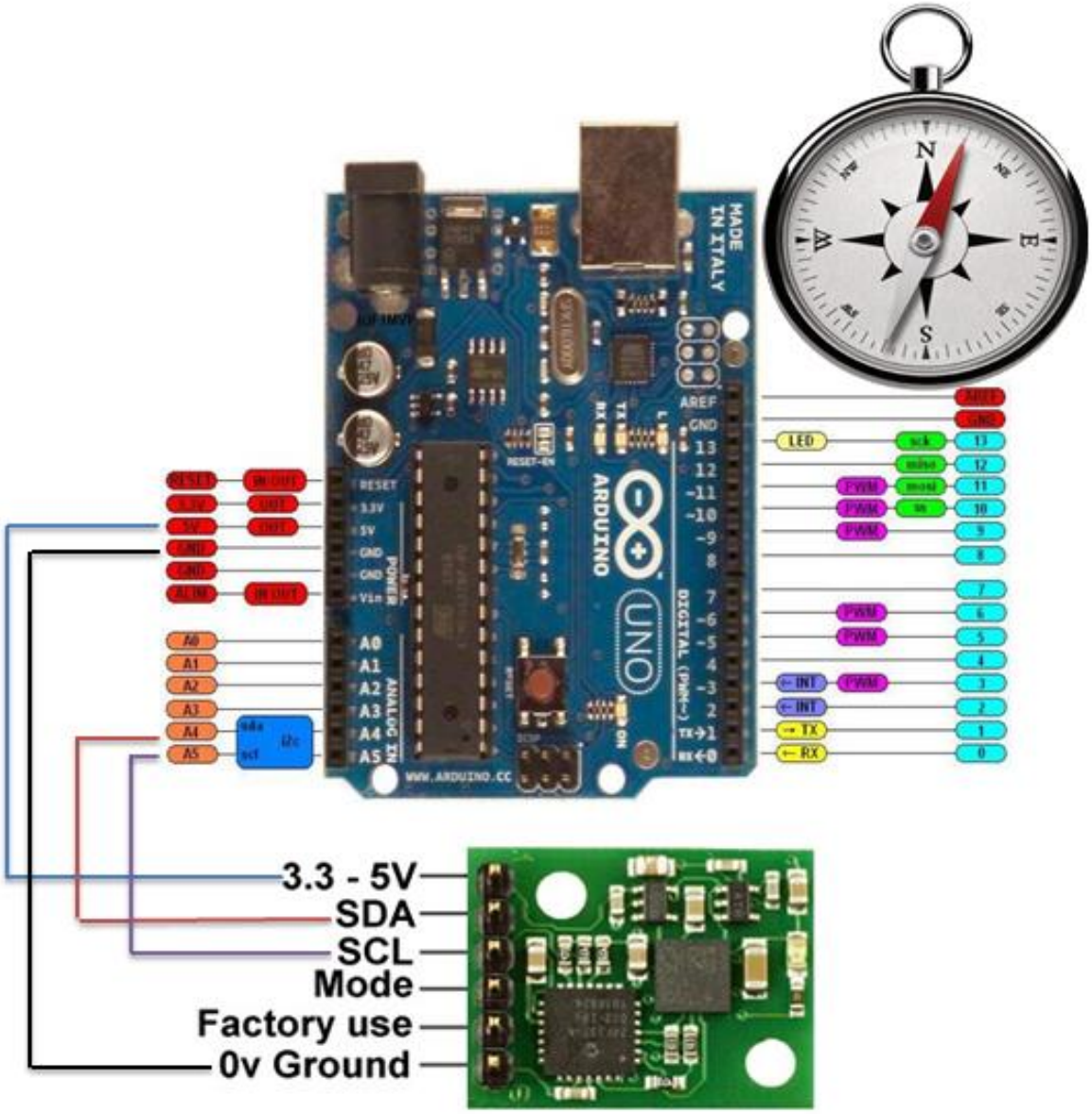
File Edit Sketch Tools Help

motortest1 §

```
int E1=6;
int M1=7;
int E2=5;
int M2=4;

void setup()
{
  pinMode(M1, OUTPUT);
  pinMode(M2, OUTPUT);
}
void loop()
{
  int value;
  for(value=0 ; value<= 255; value+=5)
  {
    digitalWrite(M1, HIGH);   //direction >>>>>>
    digitalWrite(M2, LOW);    //direction <<<<<<
    analogWrite(E1, value);   //pwm speed control
    analogWrite(E2, value);   //pwm speed control
    delay(200);
  }
}
```



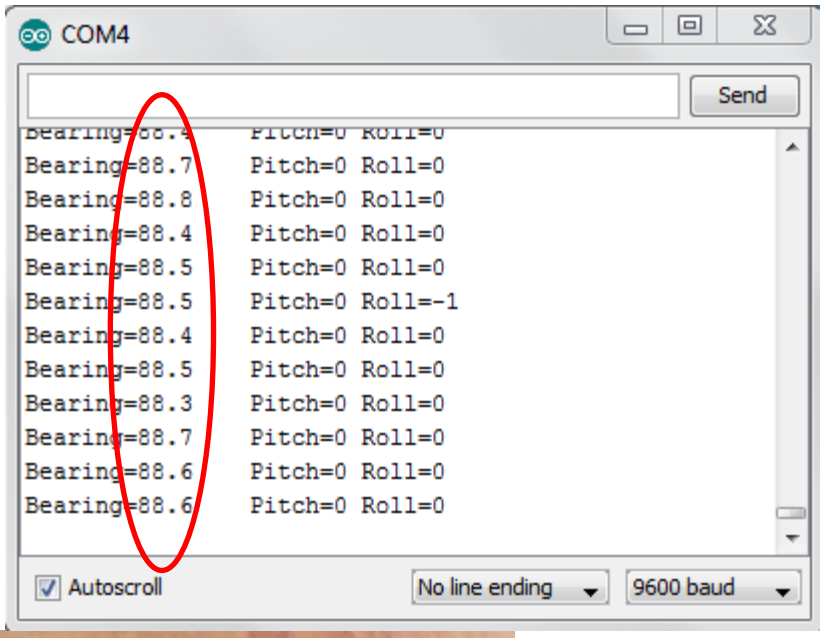
4.3) TEST DE LA BOUSSOLE

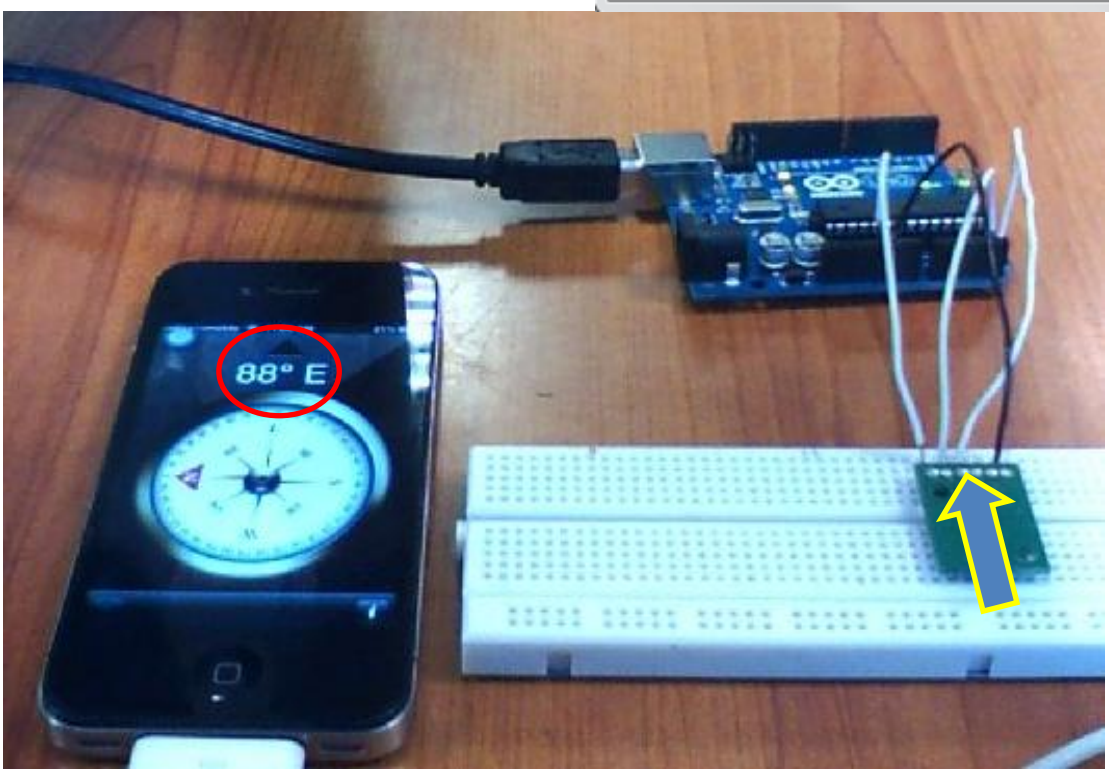


PROGRAMME

```
#include <Wire.h> // Use I2C library
#define ADDRESS 0x60 // Address of CMPS10

void setup(){
Serial.begin(9600);
Wire.begin(); // Connect to I2C
}

void loop(){
byte byteHigh, byteLow, fine; // byteHigh/byteLow store high and low bytes of
the bearing
// fine stores decimal place of bearing
char pitch, roll; // Stores pitch and roll (signed values)
int bearing; // Full bearing

Wire.beginTransmission(ADDRESS); // begin communication with CMPS10
Wire.write(2); // Start read
Wire.endTransmission();
Wire.requestFrom(ADDRESS, 4); // Request 4 bytes from CMPS10
while(Wire.available() < 4); // Wait for the bytes to arrive
byteHigh = Wire.read(); // Store values
byteLow = Wire.read();
pitch = Wire.read();
roll = Wire.read();
 bearing = ((byteHigh<<8) + byteLow) / 10; // Calculate full bearing
fine = ((byteHigh<<8) + byteLow) % 10; // Calculate bearing decimal
print_data(bearing, fine, pitch, roll); // Print data

delay(500);
}
```

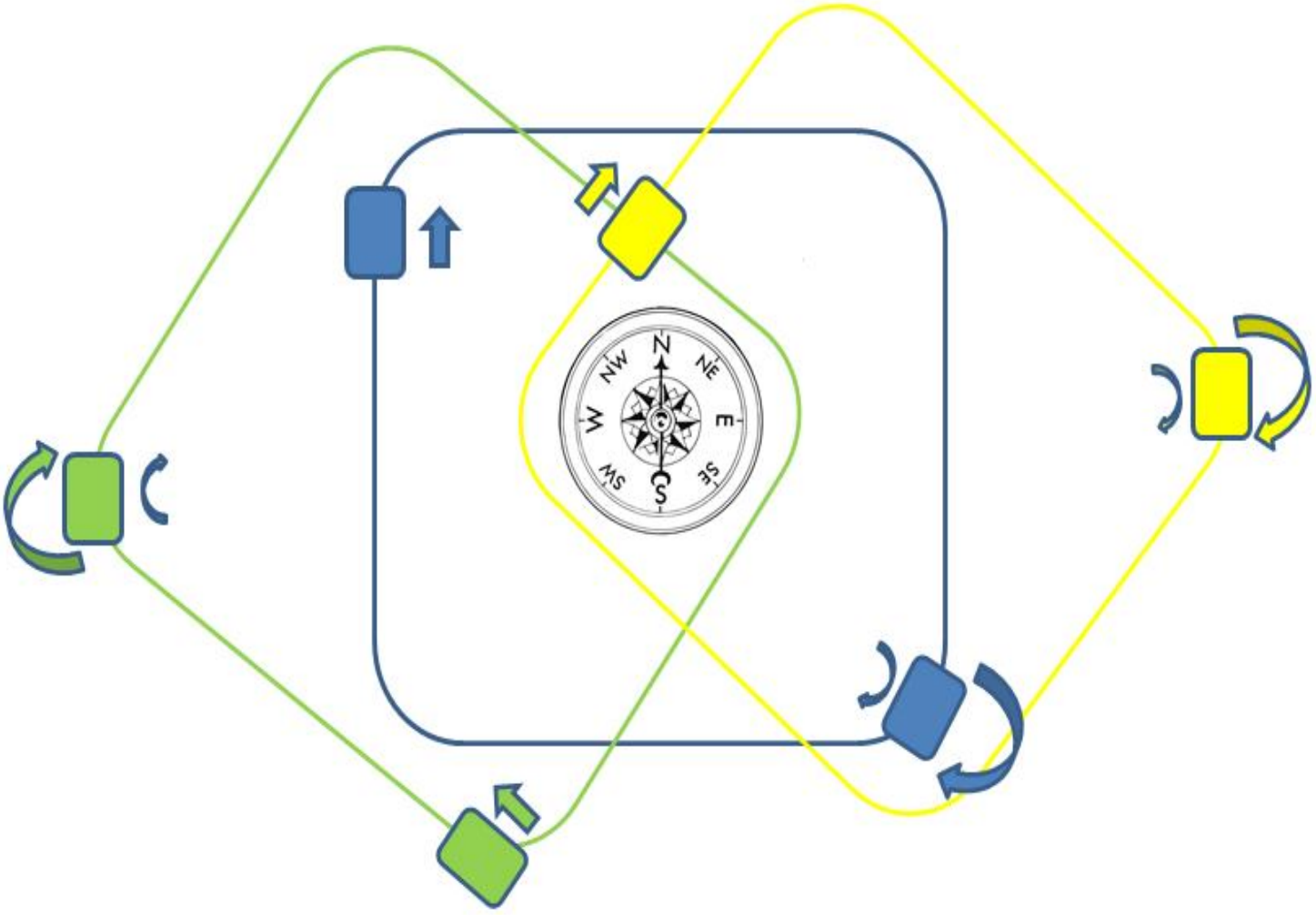
```
void print_data(int bearing, int fine, int pitch, int roll) {
// Print data to Serial Monitor window
Serial.print("Bearing=");
Serial.print(bearing, DEC);
Serial.print(".");
Serial.print(fine, DEC);

Serial.print("\t");
Serial.print("Pitch=");
Serial.print(pitch, DEC);

Serial.print("\t");
Serial.print("Roll=");
Serial.print(roll, DEC);
Serial.println("");
}
```





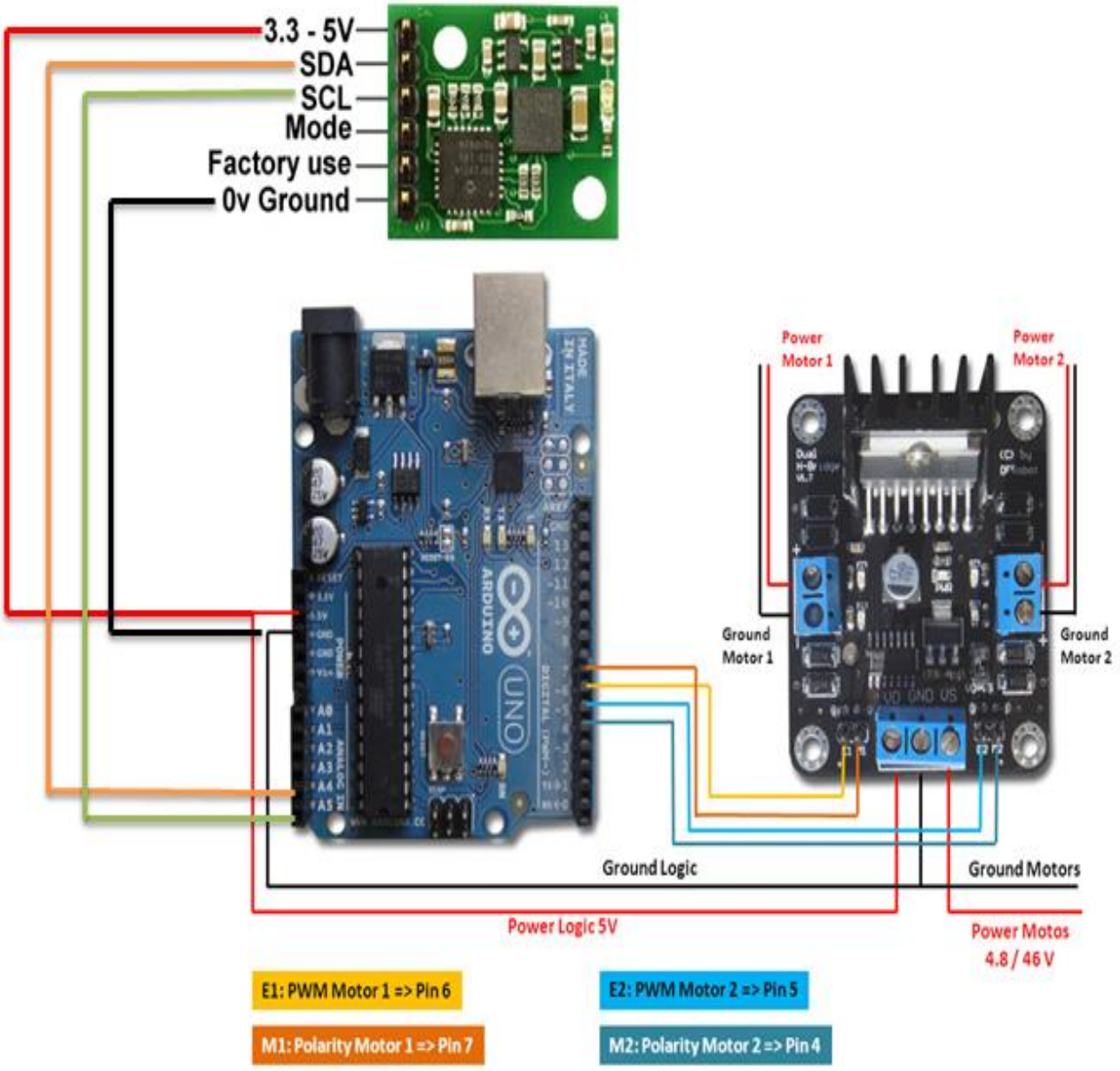
4.4) TEST « CARRE »

Dans ce test le robot doit faire un carre grâce à la boussole !!!!



Le robot chaque 2 seconde doit ajouter 90 degrés à sa direction désiré, direction initiale = direction de la boussole .

Les connections :



CODE «PROGRAMME»

```
#include <Wire.h>                 // Use I2C library
#define ADDRESS 0x60              // Address of CMPS10
int E1=6;
int M1=7;
int E2=5;
int M2=4;
float speedLeft=(245/2);        // <<245 pas 255 car il y a une diff entre
float speedRight=(255/2);     //  les deux moteurs >>
int x = LOW;
int direc;
unsigned long timerA;           //on initialize le timer

void setup(){
Serial.begin(9600);             //serial to simulate on the screen
Wire.begin();                   // Connect to I2C
pinMode(M1, OUTPUT);
pinMode(M2, OUTPUT);
}

void loop(){

   digitalWrite(M1, HIGH);
   digitalWrite(M2, HIGH);
   analogWrite(E1, speedRight);
   analogWrite(E2, speedLeft);
```

```
 if(x==HIGH)
 {if(millis()-timerA >= 2000)        // chaque 2 s ajoute 90 degrés
   { direc=direc+90;
     if(direc > 360)
      { direc= direc-360;}
     timerA=millis();                 //bi ballish el timer min jdid
   }                                  //reset the timer
 }
else
{  timerA=millis();                  //set the timer


}

 byte byteHigh, byteLow, fine; // byteHigh/byteLow store high and low
//bytes of the bearing
// fine stores decimal place of bearing
char pitch, roll; // Stores pitch and roll (signed values)
int bearing; // Full bearing


Wire.beginTransmission(ADDRESS); // begin communication with
CMPS09
Wire.write(2); // Start read
Wire.endTransmission();
Wire.requestFrom(ADDRESS, 4);      // Request 4 bytes from CMPS10
while(Wire.available() < 4);          // Wait for the bytes to arrive
byteHigh = Wire.read();              // Store values
byteLow = Wire.read();
pitch = Wire.read();                 //pas nécessaire dans ce programme
roll = Wire.read();                  //pas nécessaire dans ce programme
```

```
bearing = ((byteHigh<<8) + byteLow) / 10;    // Calculate full bearing
fine = ((byteHigh<<8) + byteLow) % 10;       // Calculate bearing decimal
print_data(bearing, fine, pitch, roll);         // Print data

if(x== LOW)
{  //delay(500);
   direc = bearing;
   x = HIGH;
}


 int diff = bearing - direc;
 // modify degress
  if(diff > 180)
    diff = -360+diff;
  else if(diff < -180)
    diff = 360+diff;

  // Make the robot turn to its proper orientation
 // diff = map(diff, --,-- ,--,--);

  if(diff > 0) {
                              // keep the right wheel spinning,
                              // change the speed of the left wheel
    speedLeft = ((245-diff)/2);
    speedRight = (255/2);
  } else {
                             // keep the right left spinning,
                             // change the speed of the left wheel
    speedLeft = (245/2);
```

```
  speedRight = ((255+diff)/2);

 }
}
void print_data(int bearing, int fine, int pitch, int roll) {
// Print data to Serial Monitor window    pour la simulation
Serial.print("Bearing=");        //print the bearing
Serial.print(bearing, DEC);
Serial.print(".");
Serial.print(fine, DEC);
Serial.print("\t");
Serial.print("Pitch=");
Serial.print(pitch, DEC);
 Serial.print("\t");
Serial.print("Roll=");
Serial.print(roll, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("direc=");         //print direction
Serial.print(direc, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("speedLeft=");      //print the value of the speed (left)
Serial.print(speedLeft, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("speedRight=");    //print the value of the speed (right)
Serial.print(speedRight, DEC);
Serial.println("");
}
```

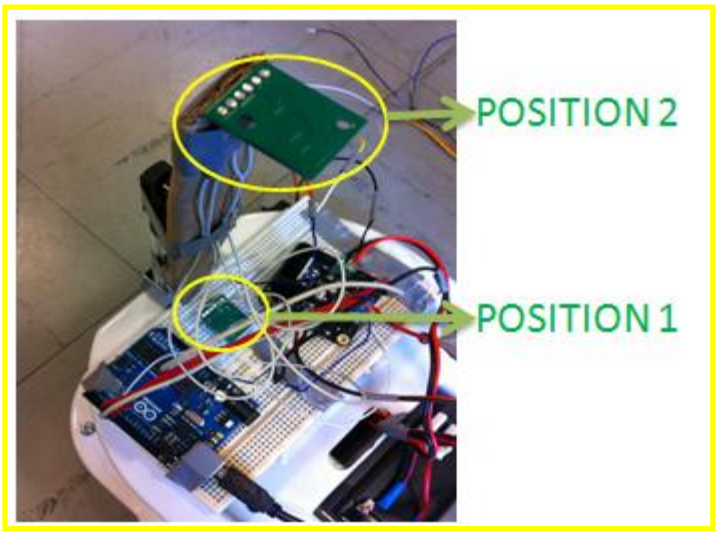
Problèmes et solutions :

_ Les 2 DC moteurs ne sont pas fiable 100 % et il Ya un décalage entre eux, pour cela c'est mieux d'utilise un servomoteur (car c'est plus précis qu'un moteur DC, moins lourd, et plus facile à mettre en œuvre.

Définition : un servomoteur est un moteur conçu pour produire des mouvements précis d'un élément mécanique selon une commande externe.
En d'autres termes, c'est un moteur qui va à une position précise en fonction du signal d'entré.

_la boussole est très sensible.

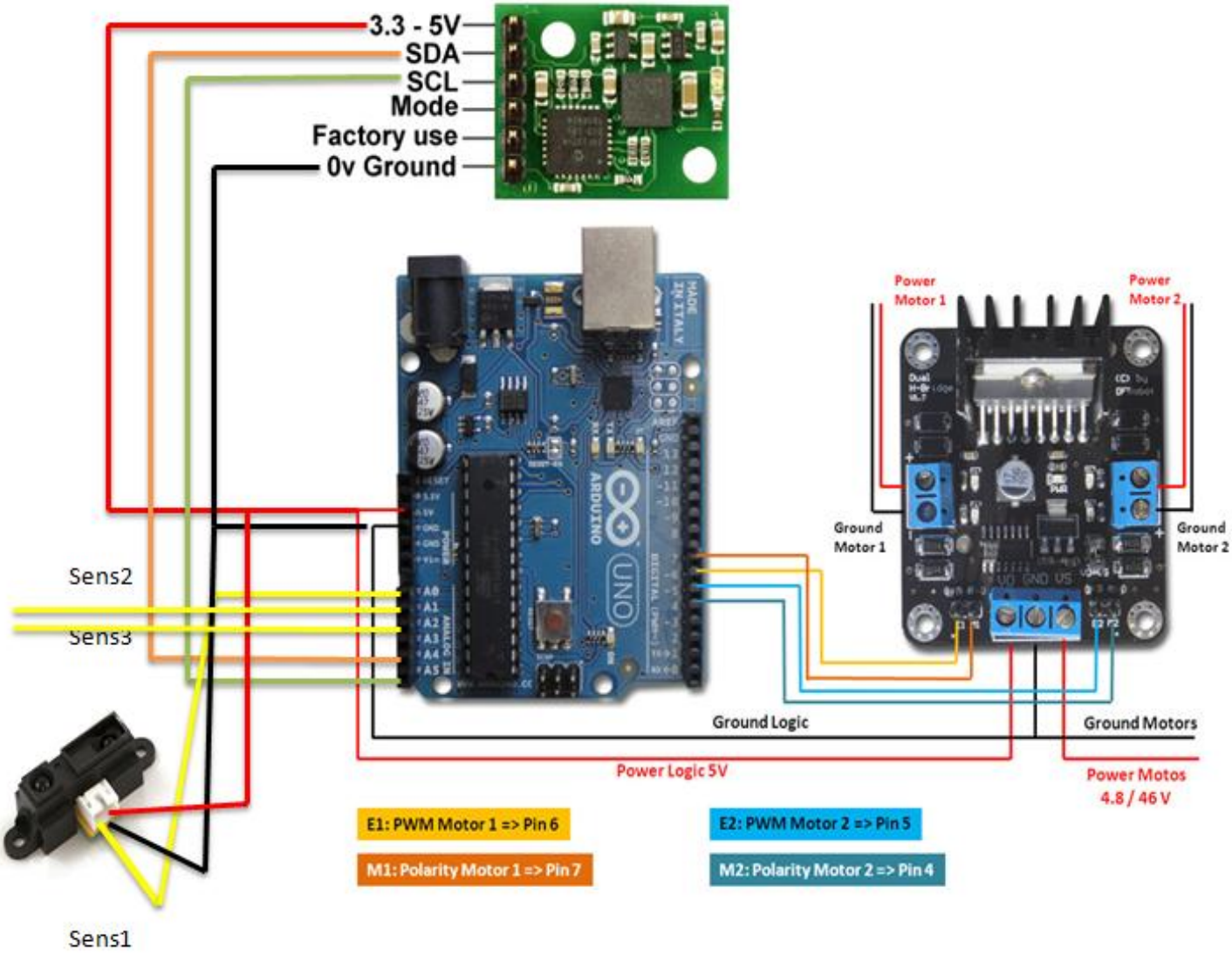
Si la boussole est en position 1 :
La boussole devient proche des Moteurs ce qui crée un champ magnétique, la boussole est alors influencer par ce champ, ce qui trouble le robot.
Pour cela l'une des solutions qui à réduit cette influence est de mettre la boussole sur un barre Loin des moteurs (position2), pour avoir une Précision plus exacte.



LE TEST FINALE

Il faut maintenant rejoindre les programmes (test), pour avoir Le programme final.

N.B : je n'ai pas utilisé un capteur (noir et blanc)
Car je n'ai pas eu le temps de créer la piste.



LE PROGRAMME

```
//georges daher
//les valeurs des moteurs ne sont pas exacte

#include <Wire.h> // Use I2C library
#define ADDRESS 0x60 // Address of CMPS10
int E1=6;
int M1=7;
int E2=5;
int M2=4;
int sens1=0;
int sens2=1;
int sens3=2;
float speedLeft=(245);
float speedRight=(255);
int x = LOW;
int direc;
unsigned long timerA;

void setup(){
Serial.begin(9600);
Wire.begin();          // Connect to I2C
pinMode(M1, OUTPUT);
pinMode(M2, OUTPUT);
}
 void loop()
{
  digitalWrite(M1, HIGH);
   digitalWrite(M2, HIGH);
   analogWrite(E1, speedRight);
   analogWrite(E2, speedLeft);
   int s1=analogRead(sens1);
   int s2=analogRead(sens2);
   int s3=analogRead(sens3);
```

```
  byte byteHigh, byteLow, fine; // byteHigh/byteLow store high and low bytes of
//the bearing
    // fine stores decimal place of bearing
    char pitch, roll; // Stores pitch and roll (signed values)
    int bearing; // Full bearing
    Wire.beginTransmission(ADDRESS); // begin communication with CMPS09
    Wire.write(2); // Start read
    Wire.endTransmission();
    Wire.requestFrom(ADDRESS, 4); // Request 4 bytes from CMPS10
    while(Wire.available() < 4); // Wait for the bytes to arrive
    byteHigh = Wire.read(); // Store values
    byteLow = Wire.read();
    pitch = Wire.read();
    roll = Wire.read();
    bearing = ((byteHigh<<8) + byteLow) / 10; // Calculate full bearing
    fine = ((byteHigh<<8) + byteLow) % 10; // Calculate bearing decimal
    print_data(bearing, fine, pitch, roll); // Print data
    if(x== LOW)
    {//delay(500);
    direc = bearing;
    x = HIGH;
    }
 int diff = bearing - direc;
   // modify degress
  if(diff > 180)
    diff = -360+diff;
  else if(diff < -180)
    diff = 360+diff;
    if(diff > 0) {
    // keep the right wheel spinning,
    // change the speed of the left wheel
```

```
 speedLeft = (245-diff);
   speedRight = (255);
  }
 Else
 {
 // keep the right left spinning,
 // change the speed of the left wheel
    speedLeft = (245);
   speedRight = (255+diff);

  }

    if( s1>100 && s2>s3)
    {
//analogWrite(E1, 255);
// analogWrite(E2, 0);
 speedLeft=0;                          //les valeurs ne sont pas exacte
 speedRight=255;                        //les valeurs se change d'un
delay(300);                           //les valeurs se change d'un
                                       //moteurs DC
                                      //if faut faire beaucoup des expériences
                                       //pour pouvoir a la fin adapter les
                                       //meilleurs valeurs


  }
    else if ( s1>100 && s2<s3)
    {
  //  analogWrite(E1, 0);
//    analogWrite(E2, 255);

     speedLeft=255;
    speedRight=0;
```

```
  delay(300);

}


   else if(s1<100 && s2>150)
   { // analogWrite(E1, 255);
     // analogWrite(E2, 0);

     speedLeft=0;
     speedRight=255;
     delay(200);

}


   else if(s1<100 && s3>150)
   {// analogWrite(E1, 0);
 //analogWrite(E2, 255);
 speedLeft=255;
  speedRight=0;
    delay(200);
}
  }
void print_data(int bearing, int fine, int pitch, int roll) {
// Print data to Serial Monitor window
Serial.print("Bearing=");
Serial.print(bearing, DEC);
Serial.print(".");
Serial.print(fine, DEC);
Serial.print("\t");
Serial.print("Pitch=");
Serial.print(pitch, DEC);
```

```
 Serial.print("\t");
Serial.print("Roll=");
Serial.print(roll, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("direc=");
Serial.print(direc, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("speedLeft=");
Serial.print(speedLeft, DEC);
Serial.println("");
Serial.print("\t");
Serial.print("speedRight=");
Serial.print(speedRight, DEC);
Serial.println("");
}
```

Conclusion : du robot autonome

Le robot autonome m'a permis de bien comprendre ce qui se passe et comment gérer les problèmes.
Mais l'une des propositions est d'utiliser des servomoteurs (plus exacte et précis) Pour ne pas perdre le temps à faire des essais (sur un dc moteur et sans encoder)

NB : la simulation sur serial est exacte. Mais en réel plusieurs problèmes se pose (il faut faire beaucoup des essais pour arriver à une solution). Mais pour ne pas perdre le temps c'est mieux d'utiliser des servomoteurs au lieu des DC moteurs, même si les servomoteurs sont un peu moins rapides mais plus exacte. En tout cas c'est très difficile de faire rouler le robot en grande vitesse car la boussole va être perturbé et ensuite le robot.

Conclusion

Pour conclure, du point de vue professionnel ce stage m'a apporté de l'expérience vis-à-vis des aléas que je peux rencontrer, de l'adaptation à de nouveaux matériels.

Du point de vue plus personnel, le fait d'avoir voyagé m'a apportée beaucoup que ce soit par la découverte d'une culture, mais aussi par le mode de vie des français qui diffèrent que celui d'un libanais. Cette expérience je la considère comme très positive, elle m'a appris à débrouiller dans la vie professionnelle et même personnelle, dans la vie d'un homme responsable, j'ai appris beaucoup de chose à travers les gens et nos responsables ici, et améliorer beaucoup d'autre comme les méthodes de communication et géré le temps.

Pour terminer ce stage a été très bénéfique du point de vue personnel et restera une bonne expérience et professionnel.