

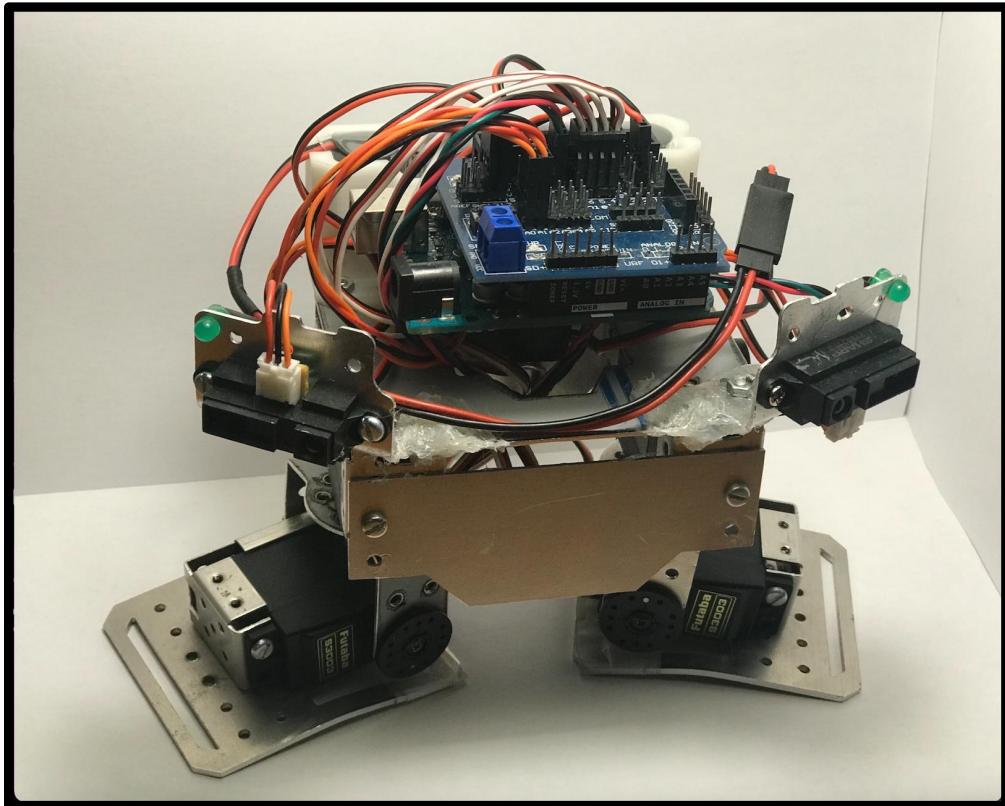


UNIVERSITÉ
CÔTE D'AZUR

RAPPORT ROBOTIQUE

MASTER 1 ESTEL

ROBOT EVITEUR D'OBSTACLE



Étudiants :

GOUFFI Djamel
CHEGGOUR Selina
MBOCK Thérèse

Encadrant :

RIBERO Jean-Marc

Année 2020/2021

TABLE DES MATIERES

Introduction.....	3
Première partie.....	4
I. Description du système.....	5
II. Grafcet du robot.....	6
III. Cahier des charges.....	8
Deuxième partie.....	9
I. Vue globale.....	10
II. La carte Arduino.....	11
1. Choix du type de la carte.....	11
2. Composantes de la carte Arduino UNO.....	12
III. EasyCard (Sensor Sheild v5.0).....	12
IV. Capteur sharp infrarouge 2Y0A21.....	13
V. Servomoteur.....	15
1. Composition d'un servomoteur.....	15
2. Commande d'un servomoteur à base de NE55.....	16
VI. Batterie.....	18
VII. Buzzer	18
VIII. LED.....	19
IX. Interrupteur.....	19
Troisième partie.....	20
I. Schéma du robot.....	21
II. Simulation des différents circuits.....	21
1. Servomoteur.....	21
2. Capteur Infrarouge.....	24
3. Buzzer.....	26
III. Code Final.....	27
1. ÉCLAIREMENTS ET COMMENTAIRES.....	27
Conclusion.....	28

Introduction

Durant le premier semestre de notre année de MASTER 1 ESTEL, nous avons suivi le module Robotique. Avant de passer à la pratique et de construire notre propre robot, nous avons d'abord vu ce qu'est la robotique en théorie.

L'origine du mot robot provient de la langue tchèque dans laquelle son ancêtre « *robota* » signifie travail forcé. Dans le Petit Larousse, on définit un robot comme étant *un appareil automatique capable d'exécuter des opérations selon un programme fixe ou modifiable*. L'Association Française de Normalisation (A. F. N. O. R.) précise tout de même qu'un robot a la capacité *de manipuler des matériaux, des pièces, des outils et des dispositifs spécialisés, au cours de mouvements variables et programmés pour l'exécution d'une variété de tâches*. Du loisir à l'industrie ou encore du domaine médical au service domestique, on retrouve les robots omniprésents dans notre environnement. L'électronique étant un secteur en continue évolution, le robot est donc toujours sujet à de nouvelles innovations.

Au cours de l'histoire, on peut distinguer trois générations de robots.
La première génération est celle du robot passif. Il peut effectuer des tâches complexes mais de façon répétitive sans prendre en compte les facteurs de l'environnement qui l'entoure, il s'adapte donc très peu.

La seconde génération est celle du robot actif. Il est capable de visualiser son environnement grâce à des capteurs. Les capteurs correspondent aux différents sens qu'on voudra attribuer au robot. Il existe une multitude de capteurs : photo-électronique, de température, à ultrasons, etc. Le robot sait donc s'adapter à son environnement en choisissant le bon comportement.

Enfin, la dernière génération est celle du robot intelligent. Ici, on parle d'Intelligence Artificielle (IA). Elle vise à mimer le fonctionnement du cerveau humain. En plus des capteurs physiques, cela permet aux machines d'imiter la logique de l'Homme lorsqu'il s'agit de prendre des décisions complexes.

Dans notre cas, nous avons pour objectif de construire un robot actif ayant deux missions à accomplir. Nous nous demanderons dans un premier temps comment mettre sur pied un robot capable d'éviter des obstacles en s'aidant de capteurs sensoriels ? Ensuite nous nous demanderons comment optimiser ce robot afin de le rendre le plus rapide le plus rapide possible afin de le mettre en compétition avec les autres robots de la promotion ?
Pour répondre à ces questions, nous verrons tout d'abord l'étude fonctionnelle de notre robot, puis ses matériaux et composants et nous finirons avec les simulations.

Première partie

Étude fonctionnelle

I. Description du système

Comme il l'a été mentionné précédemment, nous allons construire un robot éviteur d'obstacle. Nous verrons ici quel schéma basique de robotique nous devrons suivre puis nous le comparerons aux caractéristiques du robot que nous mettrons sur pied.

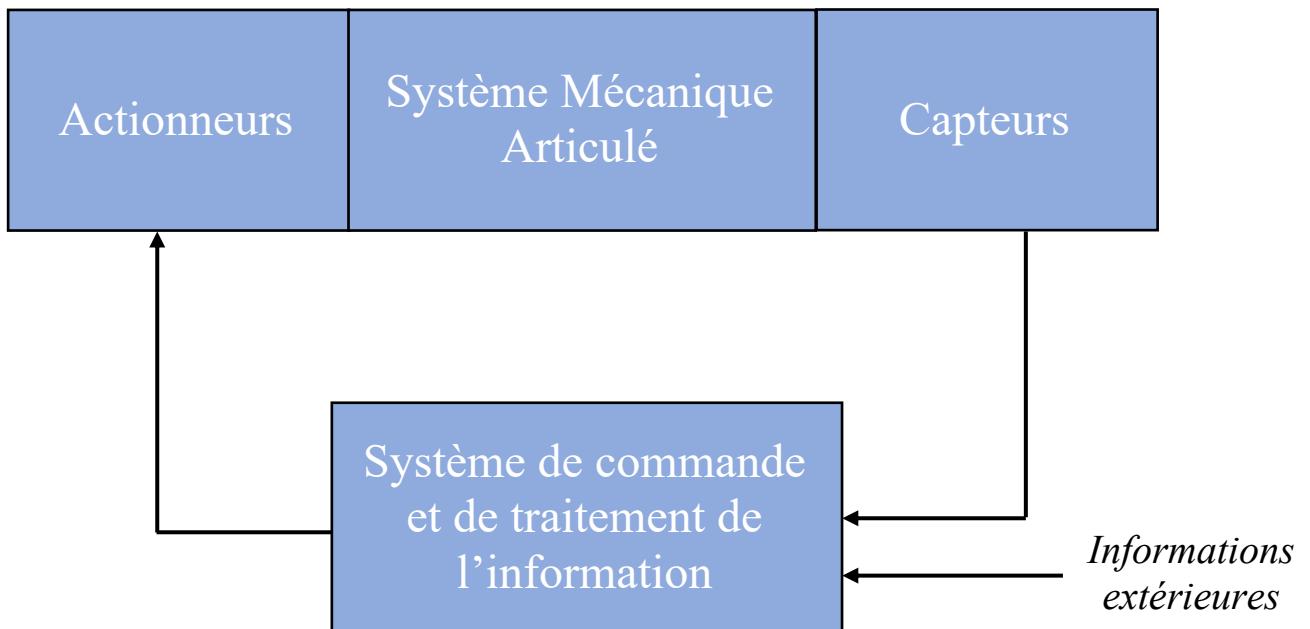


Figure 1 – Les quatre parties principales d'un robot en général

Dans le cas d'un robot possédant un bras articulé, le Système Mécanique Articulé serait justement le mécanisme de ce bras qui aurait pour rôle d'amener la pince (qui est aussi appelée organe terminale) dans une situation donnée (position et orientation) selon des caractéristiques de vitesse et d'accélération. Son architecture rigide est assemblée par des articulations. L'appellation « actionneur » désigne tous les systèmes concernant la mise en mouvement, le changement de structure (attraper un objet, pousser, tirer...). Un actionneur est déclenché par une sortie du système de commande du robot. Ce système de commande est lié au traitement de l'information car il reçoit les informations obtenues par les capteurs. Les capteurs forment le lien entre l'environnement et le robot même si des informations extérieures peuvent ne pas être détectées par ces derniers.

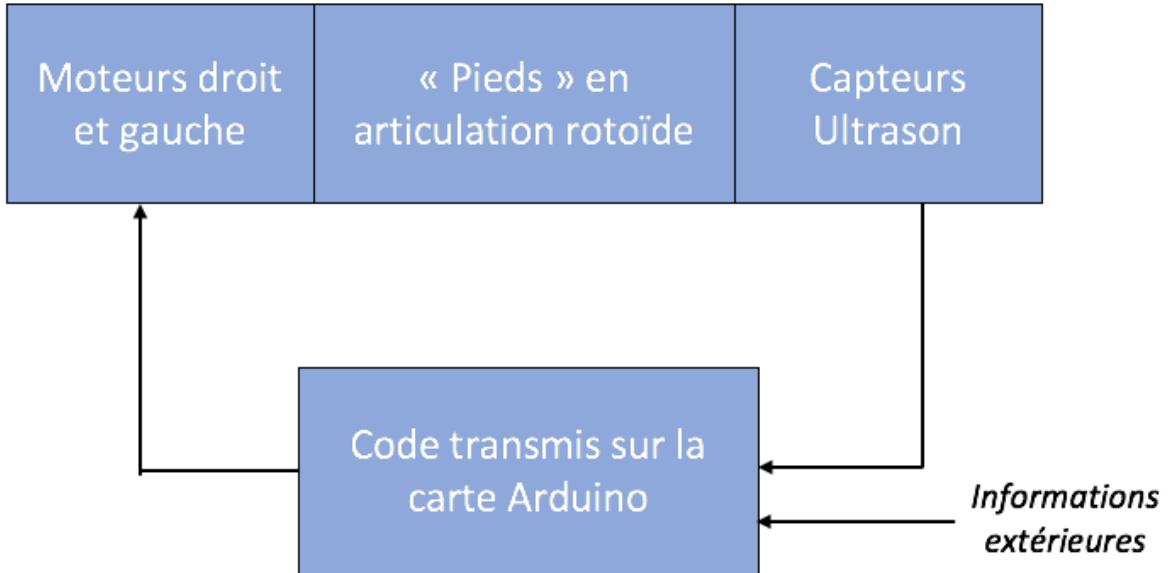


Figure 2 – Les quatre parties principales de notre robot éviteur d’obstacle

Dans notre situation, les pieds droit et gauche du robot représentent le système mécanique articulé. On dit qu’ils sont en articulation rotatoire car il s’agit d’une articulation de type pivot réduisant le mouvement autour d’un axe. Les actionneurs sont les moteurs rattachés à chacun des pieds. Ils sont activés par le code implémenté dans la carte Arduino présente sur le robot. Les capteurs que nous allons utiliser seront des capteurs infrarouges. Ces derniers auront pour rôle de calculer la distance entre le robot et les obstacles qu’il pourrait rencontrer. De là, le robot devra choisir une stratégie afin d’éviter l’obstacle détecté.

II. Graftet du robot

Nous allons maintenant mettre en place le graftet à l’aide des informations données précédemment. Il précisera également les différentes stratégies que pourra adopter le robot. Dans le graftet voici les conditions que vous retrouverez :

- m : mise en marche
- o : présence d’un obstacle
- d : la distance la plus courte est mesurée par le capteur droit
- g : la distance la plus courte est mesurée par le capteur gauche
- e : les deux capteurs mesurent des distances égales

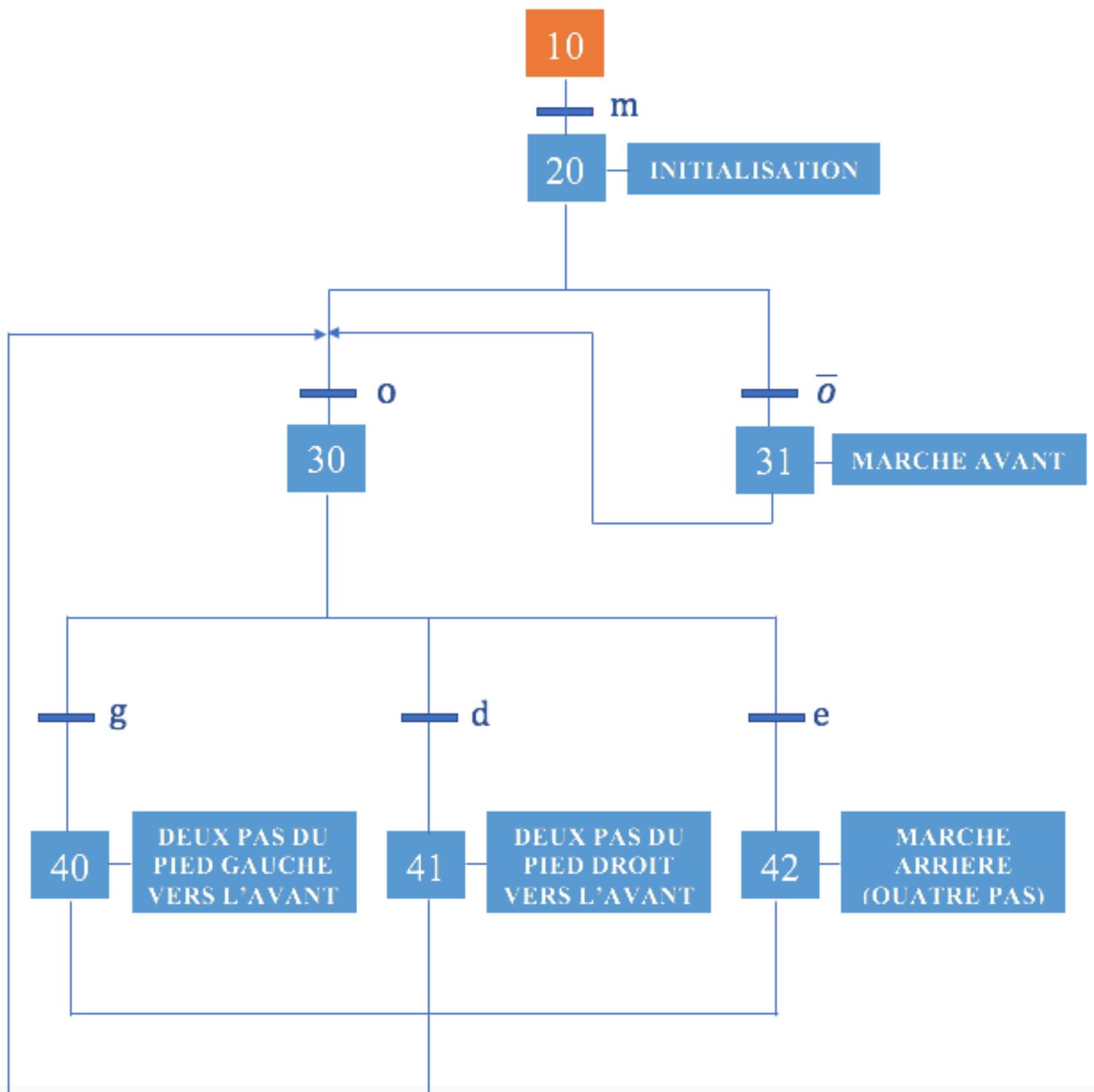


Figure 3 – Grafcet fonctionnel de notre robot éviteur d’obstacle

Au total trois stratégies sont mises en place. La première sera lorsque le robot rencontrera un obstacle à sa gauche. Dans cette situation, il faut le faire tourner vers la droite et pour ce faire on ne bouge que son pied gauche. S'il voit un obstacle sur sa droite, il procèdera de la même façon mais fera deux pas avec son pied droit pour tourner à gauche. Dans le cas précis où les capteurs droit et gauche calculent un obstacle équidistant, le robot reculera de quatre pas avant de recalculer la distance aux obstacles dans sa nouvelle position. Peu importe la posture dans laquelle se trouve le robot, chaque fois qu'il sera allumé ce dernier se mettra en position zéro (initialisation), c'est-à-dire stable sur ses deux pieds, prêt à marcher.

III. Cahier des charges

Voici les fonctionnalités principales que le robot devra remplir à la fin de notre projet :

- Se déplacer dans toutes les directions
- Être à l'équilibre après chaque déplacement
- Capter l'existence d'un obstacle
- Analyser les données reçues par les capteurs de façon précise
- Choisir la stratégie adaptée à sa position face à un obstacle

Dans un second temps :

- Être capable de « courir » le plus vite possible
- Rester à l'équilibre lors de sa course
- Ne pas dévier de sa piste

Il faut maintenant se concentrer sur les matériaux et composants qui nous permettrons de remplir toutes ces conditions.

Deuxième partie

Matériaux et composants

Le choix des matériaux est une tâche fondamentale et très complexe. Pour réussir à bien l'aborder, nous avons posé un groupe de critères que nous devions satisfaire afin de faire les bons choix. Les critères imposés sont les suivants :

- Au niveau des matériau : caractéristiques mécaniques, esthétiques, thermiques, électroniques, économiques, environnementales, physiques.
- Au niveau du procédé : volume, masse, géométrie, taille de la série, caractéristiques économiques et environnementales.

I. Vue globale

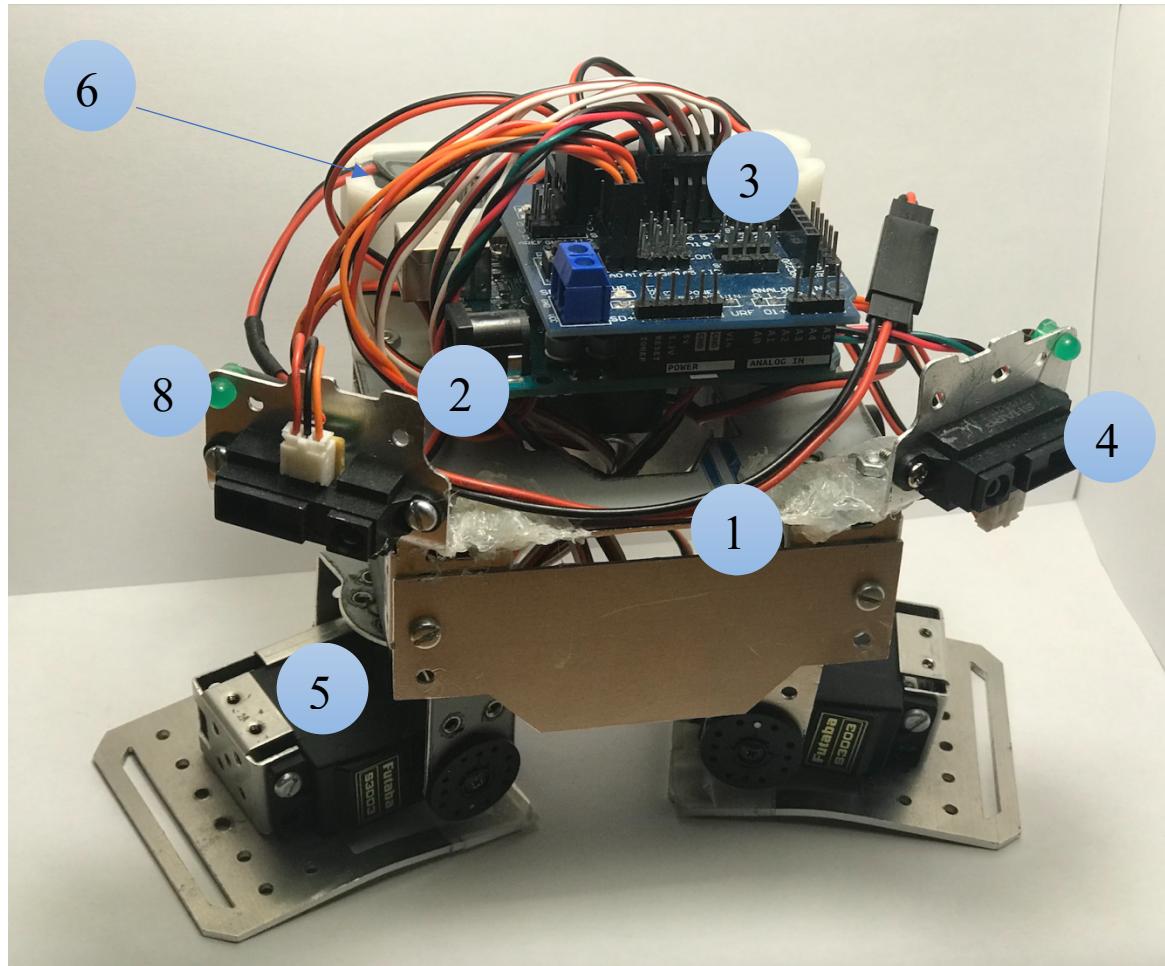


Figure 4 – Châssis du robot

L'idée est d'utiliser une carte Arduino pour lire les données reçues par un, ou plusieurs, capteur(s) à infrarouge. Lorsque le capteur détecte un obstacle devant lui, l'Arduino doit envoyer des instructions aux servomoteurs qui contrôle les mouvements du robot pour l'éviter. Le robot sera constitué du châssis et des composantes électroniques (Arduino, capteur infrarouge, servomoteurs, ...) fixé à ce dernier à l'aide de pièces en aluminium et de vis (ou simplement à l'aide d'une colle). Les matériaux dont nous avons besoin pour réaliser ce robot sont :

1. Un châssis de robot bipède.
2. Une carte Arduino UNO.
3. Une EasyCard(assemblée, soudée).
4. Deux capteurs sharp infrarouge 2Y0A21.
5. Quatre servomoteurs.
6. Une batterie de 6V et son chargeur.
7. Un buzzer.
8. Deux LED.
9. Un interrupteur.

II. La carte Arduino

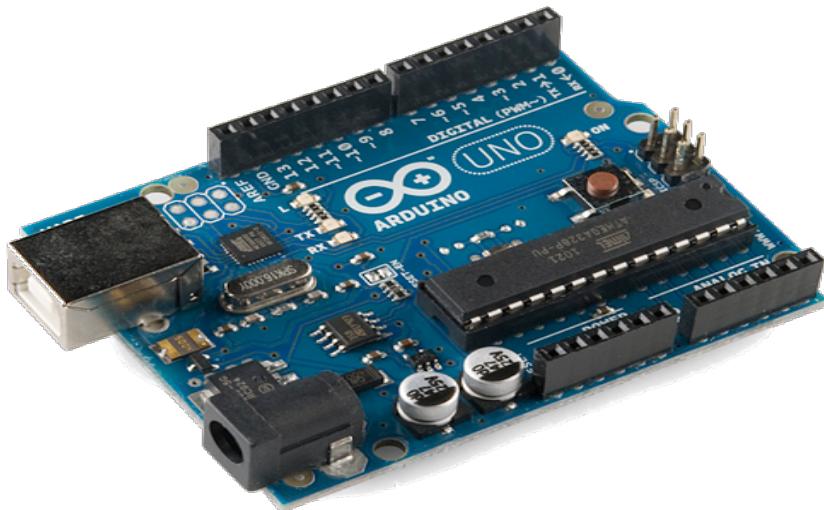


Figure 5 – Carte Arduino UNO

Une carte Arduino est une petite carte électronique équipée d'un microcontrôleur. Le microcontrôleur permet, à partir d'événements détectés par des capteurs, de programmer et commander des actionneurs ; la carte Arduino est donc une interface programmable. Elle peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des tâches très diverses comme le contrôle des appareils domestiques, éclairage, chauffage, le pilotage d'un robot, de l'informatique embarquée, etc.

1. Choix du type de la carte

Il existe plusieurs types de cartes Arduino, nous nous sommes orientés uniquement vers le type le plus fréquent, qui est la carte Arduino UNO. Pour réaliser ce montage, on doit assurer la liaison entre chaque composante électronique et notre carte Arduino.

Au total, nous aurons besoin d'une carte Arduino qui contient au moins 9 broches numériques dont deux PWM. Elle devra être capable d'exécuter le plus vite possible l'algorithme gravé dans sa mémoire. En revanche, la carte Arduino UNO est la carte la plus convenable à cette situation puisqu'elle possède 14 broches numériques (dont 6 avec PWM) et 6 broches d'entrée analogique.

La carte UNO ainsi que les autres types cartes Arduino (Micro et Mega) ont chacune leur propre capacité de traitement. Parlons à présent de leurs fréquences/vitesses d'horloge respectives. La fréquence/vitesse d'horloge de ces cartes indique simplement la vitesse avec laquelle elles peuvent exécuter une commande. Les trois cartes possèdent toutes la même vitesse d'horloge, soit 16 MHz.

Au niveau de la mémoire La Uno et la Micro possèdent toutes les deux une mémoire Flash de 32 ko, tandis que la Mega 2560 en propose 256 ko, soit 8 fois plus d'espace mémoire !

La mémoire Flash représente simplement la taille maximale du code ou du modèle que vous pouvez charger sur votre Arduino. Si votre code est lourd, la Mega 2560 est donc la solution idéale.

2. Composantes de la carte Arduino UNO

Une carte Arduino UNO se compose essentiellement de :

- Prisejack: Permet de brancher une alimentation (pile, batterie, adaptateur secteur,+ au centre Vin 7 à 12 V).
- Microcontrôleur : stocke le programme et l'exécute.
- Entrées analogiques : Permettent de brancher des capteurs et des détecteurs analogiques.
- Connexion USB (Universal Serial Bus) : Permet d'alimenter la carte en énergie électrique (5V). Permet de téléverser le programme dans la carte.
- Entrées et sorties numériques (Digital) : Permet de brancher des actionneurs. Permet de brancher des détecteurs.

III. EasyCard (Sensor Sheild v5.0)

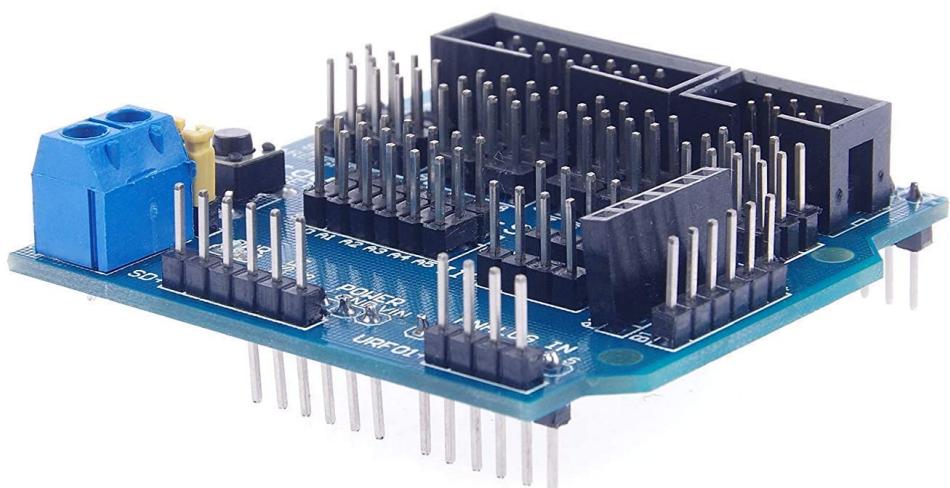


Figure 6 – EasyCard

Le bouclier d'extension d'E / S Arduino fournit un moyen facile de connecter des capteurs, des servos et un périphérique RS485 à la carte Arduino. Il étend les broches d'E / S numériques et d'entrée analogique d'Arduino avec alimentation et GND. Il fournit également des broches PWM séparées qui sont compatibles avec le connecteur servo standard. Une autre caractéristique unique est que le blindage d'E / S a un convertisseur RS485 intégré qui permet à Arduino de communiquer avec les périphériques RS485. La prise de communication offre un moyen extrêmement simple de brancher un module sans fil tel que le module RF APC220 ou le module DF-Bluetooth. Il dispose d'une entrée d'alimentation individuelle pour les servos. Un cavalier d'alimentation servo permet à l'utilisateur de sélectionner en utilisant une alimentation externe ou une alimentation interne pour piloter les servos.

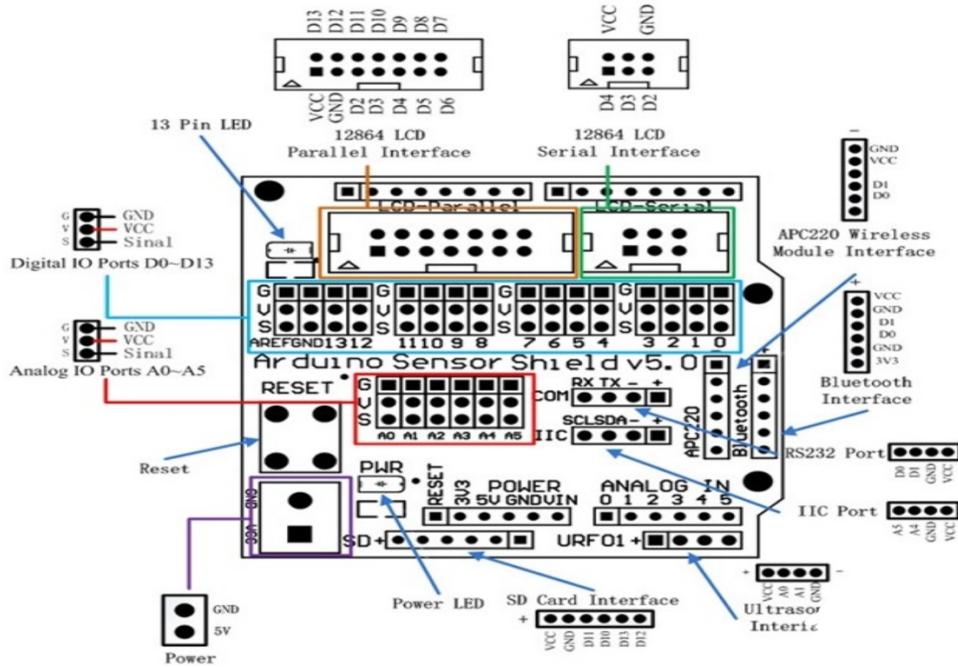


Figure 7 - Schéma électrique d'une EasyCard

IV. Capteurs sharp infrarouge 2Y0A21

Le détecteur de distance **Sharp GP2Y0A21YK** (10-80cm) est un senseur de proximité infrarouge évaluant la distance sur base de la quantité de lumière infrarouge reçue en retour.



Figure 8 – Capteur infrarouge

Sa technologie infrarouge offre une évaluation de la distance moins rigoureuse qu'un senseur ultrason mais reste cependant très économique. Il permet donc la détection d'obstacles (et jugement de distances) à prix abordable.

Ce senseur dispose d'une électronique embarquée qui le rend à la fois autonome et permet de proposer une interface simple à mettre en œuvre. En effet, le senseur fournit une tension analogique qui est proportionnelle à la distance à laquelle l'objet est détecté.

La tension de sortie varie de 2,3 - 0,4 Volts pour une distance variant respectivement 10 - 80 cm. Il suffit donc de brancher ce senseur sur une entrée Analogique pour évaluer la distance.

D'après le graph tension /distance : il est difficile de mesurer une distance entre 5 et 10 cm, tout comme une distance supérieure à 50 cm (pour un objet non massif). Les corps massifs sont détectés jusqu'à 70 cm sans problème.

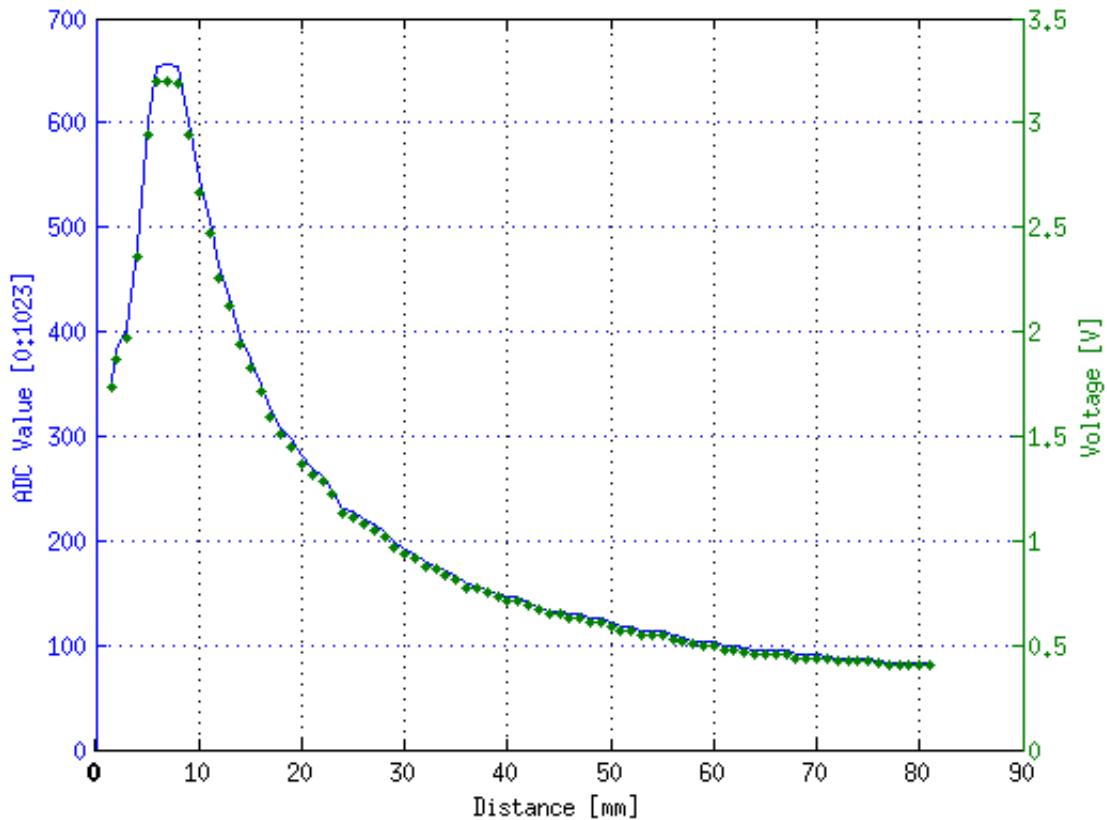


Figure 9 – Graph de correspondance tension/distancce

V. Servomoteur

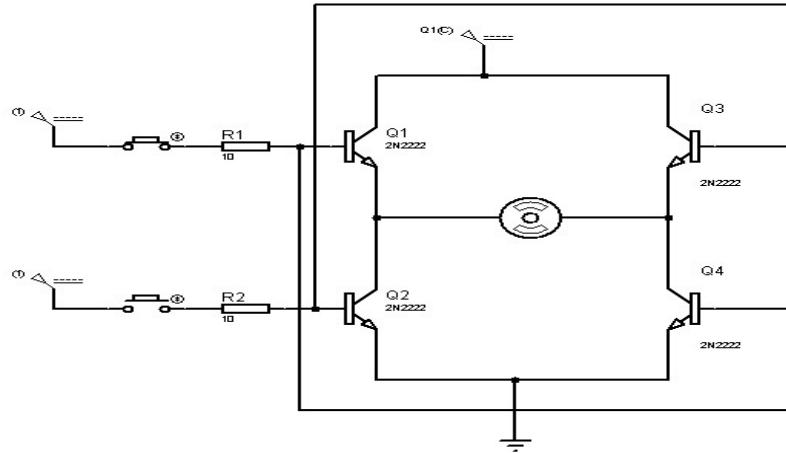


Figure 10 – Schéma d'un pont en H réalisé par Proteus à base de transistors

Un servomoteur est un système asservi (ou système bouclé). Il s'agit d'un moteur à courant continu enfermé dans un boîtier de petite taille avec des constituants électroniques et mécaniques. Sa fonctionnalité est de tenir une position angulaire donnée fournie au système en entrée sous forme d'un signal électrique. On en trouve de toutes les tailles et de toutes les puissances. La plupart du temps, la sortie peut se positionner entre 0 et 180°. Il en existe également dont la sortie peut se débattre sur seulement 90° et d'autres, ayant un plus grand débattement, sur 360°. Ceux qui ont la possibilité de faire plusieurs tours sont souvent appelés servo-treuils. Enfin, les derniers, qui peuvent faire tourner leur axe sans jamais se percuter, sont appelés servomoteurs à rotation continue. Les servomoteurs sont très fréquemment employés dans les applications de modélisme. Par exemple, pour piloter le safran d'un bateau, le gouvernail d'un avion ou bien même les roues d'une voiture téléguidée.

1. Composition d'un servomoteur

Le servomoteur est composé de plusieurs éléments visibles et invisibles :

- Un moteur à courant continu
- Des engrenages pour former un réducteur (en plastique ou en métal)
- Un capteur de position de l'angle d'orientation de l'axe (un potentiomètre)
- Une carte électronique pour le contrôle de la position de l'axe et le pilotage du moteur à courant continu
- Les fils, qui sont au nombre de trois
- L'axe de rotation sur lequel est monté un accessoire en plastique ou en métal
- Le boîtier qui le protège

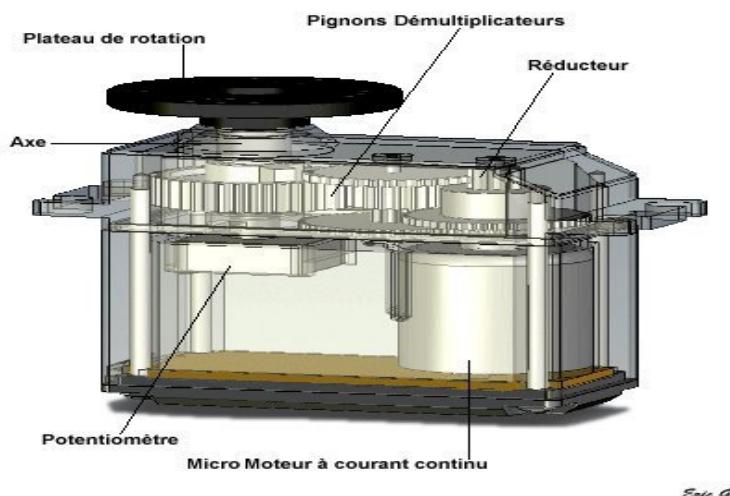


Figure 11 – Vue interne d'un servomoteur

Principe de fonctionnement d'un servomoteur :

La plupart des servomoteurs sont commandés par l'intermédiaire d'un câble électrique à trois fils qui permet d'alimenter le moteur et de lui transmettre des consignes de position sous forme d'un signal codé en largeur d'impulsion plus communément appelé PWM. Cela signifie que c'est la durée des impulsions qui détermine l'angle absolu de l'axe de sortie et donc la position du bras de commande du servomoteur. Le signal est répété périodiquement, en général toutes les 20 millisecondes, ce qui permet à l'électronique de contrôler et de corriger continuellement la position angulaire de l'axe de sortie, cette dernière étant mesurée par le potentiomètre.

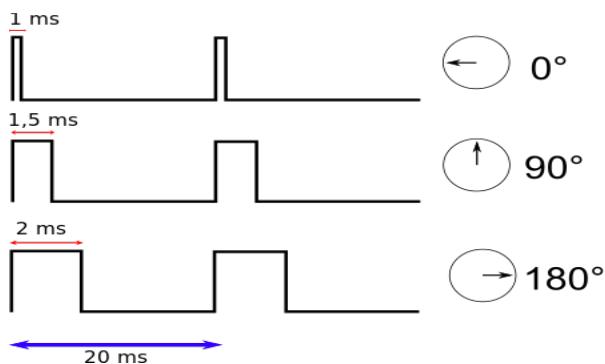


Figure 12 – Simulation d'un pont en H réalisée par Proteus à base des transistors

2. Commande d'un servomoteur à base de NE555

On cherche dans cette sous partie à savoir comment piloter la position angulaire de l'axe d'un servomoteur par un potentiomètre en utilisant quelques composants électroniques simples. On réalise le schéma suivant où le NE555 génère une impulsion périodique. En

faisant varier la valeur du potentiomètre nous obtenons la commande en position du servomoteur :

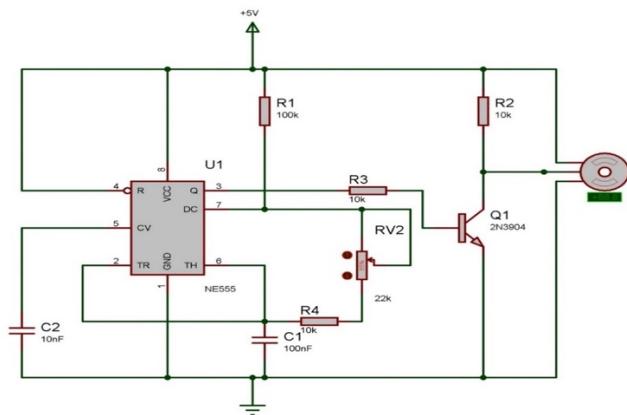
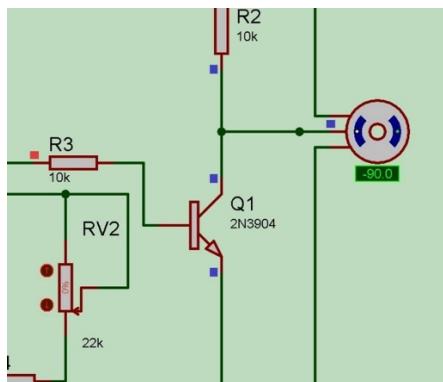
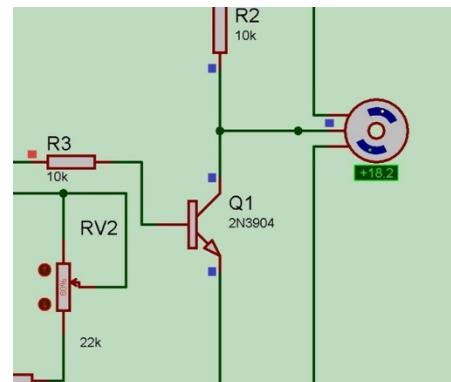


Figure 13 – Schéma de commande d'un servo-moteur à base d'un NE555

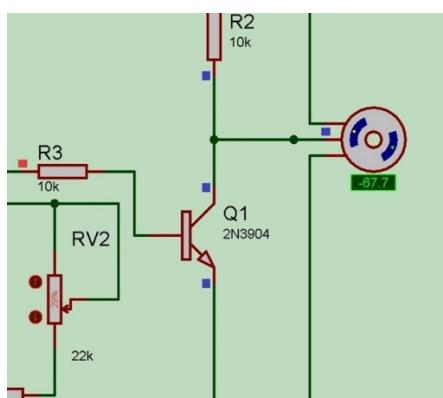
Résultats :



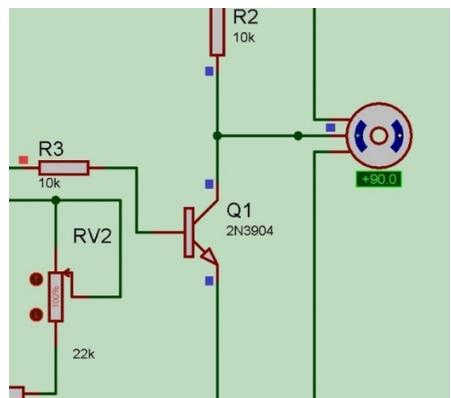
Le potentiomètre en 0% et le servomoteur indique -90



Le potentiomètre en 80% et le servomoteur indique +18



Le potentiomètre en 20% et le servomoteur indique -67



Le potentiomètre en 100% et le servomoteur indique +90

VI. Batterie



Figure 14 – Batterie

La batterie **LRP XTEC RX-PACK STRAIGHT 2/3A NIMH** permet d'alimenté le robot, Composé de 5 piles possédant une grande capacité de 1600mAh et une tension nominale de 6V. Leurs cellules sont très robustes et ont donc un taux d'auto-décharge très faible.

VII. Buzzer



Figure 15 – Buzzer

Un buzzer piézoélectrique est une sorte de haut-parleur mais de faible puissance qui va émettre un son en fonction de la fréquence et amplitude de vibration. Il permet de jouer des notes et de recréer des mélodies simples. Son rôle dans notre robot et de mettre un son court d'une durée de 5ms à chaque fois qu'il rencontre un obstacle.

VIII. LED



Figure 16 – LED

La LED (Light- Emitting Diode) est un composant électrique capable d'émettre de la lumière verte, ou autre couleur selon les modèles, lorsque'il est parcouru par un courant électrique. Elle est utilisée pour afficher des informations sous formes d'états lumineux. L'intensité de la lumière émise varie en fonction de l'intensité du courant qui la traverse.

IX. Interrupteur

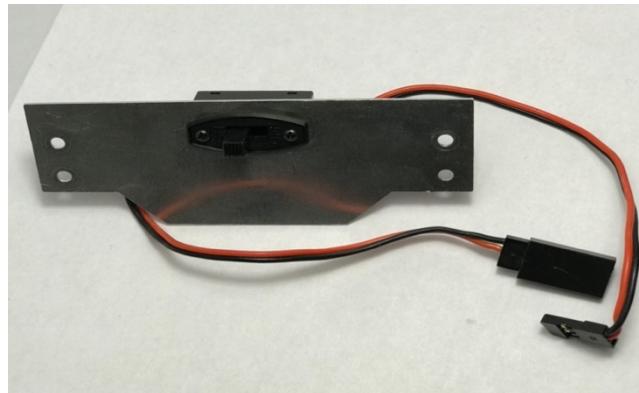


Figure 17 – Interrupteur à bascule

Dispositif permettant d'interrompre ou de laisser passer un flux électrique dans un circuit électrique.

Nous avons choisi l'interrupteur à bascule car il est utilisé lorsqu'on veut garder l'état souhaité jusqu'à la prochaine étape.

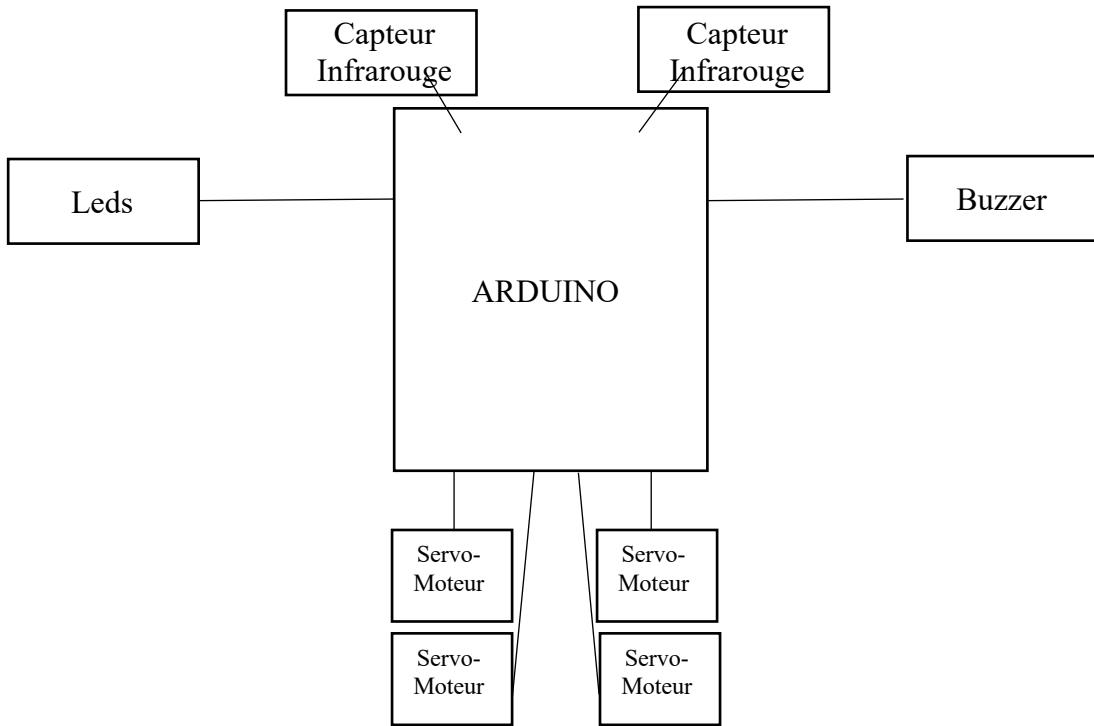
Le choix des matériaux était la phase la plus délicate dans notre projet. En effet, il ne portait pas seulement sur un aspect purement technique répondant à des exigences fonctionnelles, mais aussi à des attentes relevant des préférences des utilisateurs dans le cadre d'un marché spécifique. Globalement, le choix des matériaux a été analysé sous l'angle de l'ingénierie des matériaux, du temps de réponse, sans oublier enfin le côté esthétique (le design industriel).

Troisième partie

Simulations

La simulation de notre robot était la dernière tâche qu'on devait réaliser, le montage a déjà été terminé et le dispositif mécatronique déjà mise en place, il ne manquait qu'un code pour assimiler ces dispositifs puis vérifier si les mesures qu'on a pris nous ont emmené à concilier entre le côté technique et esthétique.

I. Schéma du robot



II. Simulation des différents circuits

1. Servomoteur

Dans cette partie, nous avons fait en sorte de programmer la marche du robot. Pour cela, nous avons eu recours à une bibliothèque prédéfinie afin de faire pivoter les moteurs tout en veillant à les synchroniser afin de créer l'équilibre nécessaire pour faire avancer ou reculer notre robot.

• Marche Avant

Une marche en avant est essentielle pour ce projet précisément. Sur ce, nous avons écrit un sketch qui fait en sorte que nos 4 moteurs soient programmés à des angles permettant de faire des pas pour une démarche en avant.

Le code suivant reporte clairement le code servant à faire faire les pas à notre robot.

```
#include <Servo.h>

Servo monServo_Rightup;
Servo monServo_Leftup;
Servo monServo_Rightlow;
Servo monServo_Leftlow;

int i = 0;

///////////////////////////////
int infraRed_right = 0; // Capteur connecté sur la broche analoge 0
int infraRed_left = 1; // Capteur connecté sur la broche analoge 1
int infra_right = 0; // Variable identifiant la valeur du capteur 1
int infra_left = 0; // Variable identifiant la valeur du capteur 2
int led_right = 12; // Led droite
int led_left = 13; // Led Gauche
int buzzer = 8; // Buzzer
void setup()
{
    // On connecte les servos aux bons pins de l'arduino
    monServo_Rightup.attach(4);
    monServo_Leftup.attach(5);
    monServo_Rightlow.attach(6);
    monServo_Leftlow.attach(7);
    monServo_Leftlow.write(60);
    monServo_Leftup.write(95);
    monServo_Rightup.write(85);
    monServo_Rightlow.write(130);
    delay(500);
    pinMode(led_right, OUTPUT);
    pinMode(led_left, OUTPUT);
    pinMode(buzzer, OUTPUT);
    digitalWrite(led_right, HIGH);
    digitalWrite(led_left, HIGH);
    digitalWrite(buzzer, LOW);
    Serial.begin(9600);
}

void loop() {
    monServo_Rightlow.write(140); //angle bas 140
```

```

delay(100);
monServo_Rightup.write(70); //angle avant 70
delay(100);
monServo_Rightlow.write(110); // angle haut
delay(100);
monServo_Rightup.write(105); //angle arriere
delay(100);

monServo_Leftlow.write(45); //angle bas
delay(100);
monServo_Leftup.write(105); //angle avant 105
delay(100);
monServo_Leftlow.write(80); // angle haut
delay(100);
monServo_Leftup.write(70); //angle arriere
delay(100);
}

```

- **Marche Arrière**

Tout comme la marche en avant, il est nécessaire dans certains cas que notre robot fasse une marche dans le sens opposé afin d'éviter des obstacles.
Le code ci-dessous démontre les instructions codées afin de faire marcher en sens opposé notre robot.

```

#include <Servo.h>

Servo monServo_Rightup;
Servo monServo_Leftup;
Servo monServo_Rightlow;
Servo monServo_Leftlow;

int i = 0;

///////////////////////////////
int infraRed_right = 0; // Capteur connecté sur la broche analoge 0
int infraRed_left = 1; // Capteur connecté sur la broche analoge 1
int infra_right = 0; // Variable identifiant la valeur du capteur 1
int infra_left = 0; // Variable identifiant la valeur du capteur 2
int led_right = 12; // Led droite
int led_left = 13; // Led Gauche
int buzzer = 8; // Buzzer
void setup()
{

    // On connecte les servos aux bons pins de l'arduino
    monServo_Rightup.attach(4);
    monServo_Leftup.attach(5);

```

```

monServo_Rightlow.attach(6);
monServo_Leftlow.attach(7);
monServo_Leftlow.write(60);
monServo_Leftup.write(95);
monServo_Rightup.write(85);
monServo_Rightlow.write(130);
delay(500);
pinMode(led_right, OUTPUT);
pinMode(led_left, OUTPUT);
pinMode(buzzer, OUTPUT);
digitalWrite(led_right, HIGH);
digitalWrite(led_left, HIGH);
digitalWrite(buzzer, LOW);
Serial.begin(9600);
}

void loop() {
    monServo_Leftup.write(40);//agnle arriere 40
    delay(60);
    monServo_Leftlow.write(80);// angle haut
    delay(60);
    monServo_Leftup.write(90); //angle avant 90
    delay(60);
    monServo_Leftlow.write(45); //angle bas
    delay(60);

    monServo_Rightup.write(120);//angle arriere 120
    delay(50);
    monServo_Rightlow.write(110);// angle haut 110
    delay(50);
    monServo_Rightup.write(90);//angle avant 90
    delay(50);
    monServo_Rightlow.write(140);//angle bas 140
    delay(50);
}

```

2. Capteur Infrarouge

Pour que notre robot évite les obstacles, il lui faut d'abord les détecter. Afin d'accomplir cela, nous avons programmé nos 2 capteurs infra-rouge qui servent de lecteur de distance entre le robot et ce qu'il rencontre sur son chemin. Ainsi les obstacles seront évités selon la proximité de ce dernier avec les objets.

Cela a été accompli en ayant recours à une bibliothèque prédéfinie dans nous avons fait usage afin de calculer en centimètres les distances en question et par la suite inciter notre robot à changer de trajectoire si nécessaire.

La lecture des valeurs analogiques de nos capteurs a été faite grâce au programme suivant :

```
/*SHARP GP2Y0A21YK0F IR distance sensor with Arduino and SharpIR library  
example code. More info: https://www.makerguides.com */
```

```
// Include the library:  
  
#include <SharpIR.h>  
  
// Define model and input pin:  
#define IRPin A0  
#define model 1080  
  
// Create variable to store the distance:  
int distance_cm;  
  
/* Model :  
   GP2Y0A02YK0F --> 20150  
   GP2Y0A21YK0F --> 1080  
   GP2Y0A710K0F --> 100500  
   GP2YA41SK0F --> 430  
*/  
  
// Create a new instance of the SharpIR class:  
SharpIR mySensor = SharpIR(IRPin, model);  
  
void setup() {  
    // Begin serial communication at a baudrate of 9600:  
    Serial.begin(9600);  
}  
void loop() {  
    // Get a distance measurement and store it as distance_cm:  
    distance_cm = mySensor.distance();  
    // Print the measured distance to the serial monitor:  
    Serial.print("Mean distance: ");  
    Serial.print(distance_cm);  
    Serial.println(" cm");  
    delay(1000);  
}
```

Après exécution de ce code, afin de mesurer les distances en utilisant l'un des capteurs, nous avons obtenu les valeurs suivantes :

On remarque clairement que l'affichage varie selon les distances lues par le capteur infrarouge.

3. Buzzer

Ce composant a été introduit dans le code du robot afin d'émettre des signaux sonores à chaque rencontre d'obstacles.

Les instructions qui ont fait que notre buzzer fonctionne à la fréquence désirée est le suivant :

```
int i = 0;  
void setup()  
{  
    digitalWrite(buzzer, LOW);  
}  
void loop() {  
    digitalWrite(buzzer, HIGH);  
    delay(50);  
    digitalWrite(buzzer, LOW);  
}
```

III. Code Final

En raison de sa dimension importante, un fichier séparé contenant le code final accompagnera le rapport de ce robot.

1. ECLAIREMENTS ET COMMENTAIRES :

Pour résumer notre robot procède ainsi :

- Pour commencer, à l'aide d'une batterie et un switch, nous alimentons notre circuit qui contrôle le robot et ses mouvements afin qu'il puisse fonctionner.
- Par la suite, le robot se mettra directement en marche comme il a été programmé tout en prenant simultanément des mesures de distances avec les objets ou obstacles en vue afin de les éviter par la suite.
- Le programme a été doté de multiples conditions permettant de contrôler le robot. Dans le cas où aucun obstacle ou objet a été détecté et que la distance mesurée est supérieure à 12 cm alors le robot continuera sa marche tout droit.
- S'il rencontre un obstacle près du capteur gauche alors il tournera dans ce sens-là afin de l'éviter, et si l'objet a plutôt été détecté du côté droit alors le robot va tourner du même côté (droit) tout en évitant l'obstacle.
- Dans le cas où la distance mesurée par les deux capteurs sera égale en même temps, le robot se doit de faire une marche arrière suivie de quatre pas en arrière du côté gauche, cela le fera sortir d'un recoin.
- A chaque fois qu'un obstacle sera détecté le buzzer lancera une alerte sonore.

Conclusion

En conclusion, nous avons pu réaliser un montage mécanique ainsi qu'électronique d'un robot éviteur d'obstacle à l'aide de multiples composants et une carte Arduino.

En somme, notre montage a parfaitement fonctionné et les performances désirées ont été atteintes.

Après réalisation d'un premier robot, un deuxième a été monté avec de plus haute performance afin d'accomplir le meilleur record possible d'une distance concourue en un moindre temps.