BERTIN Loïc

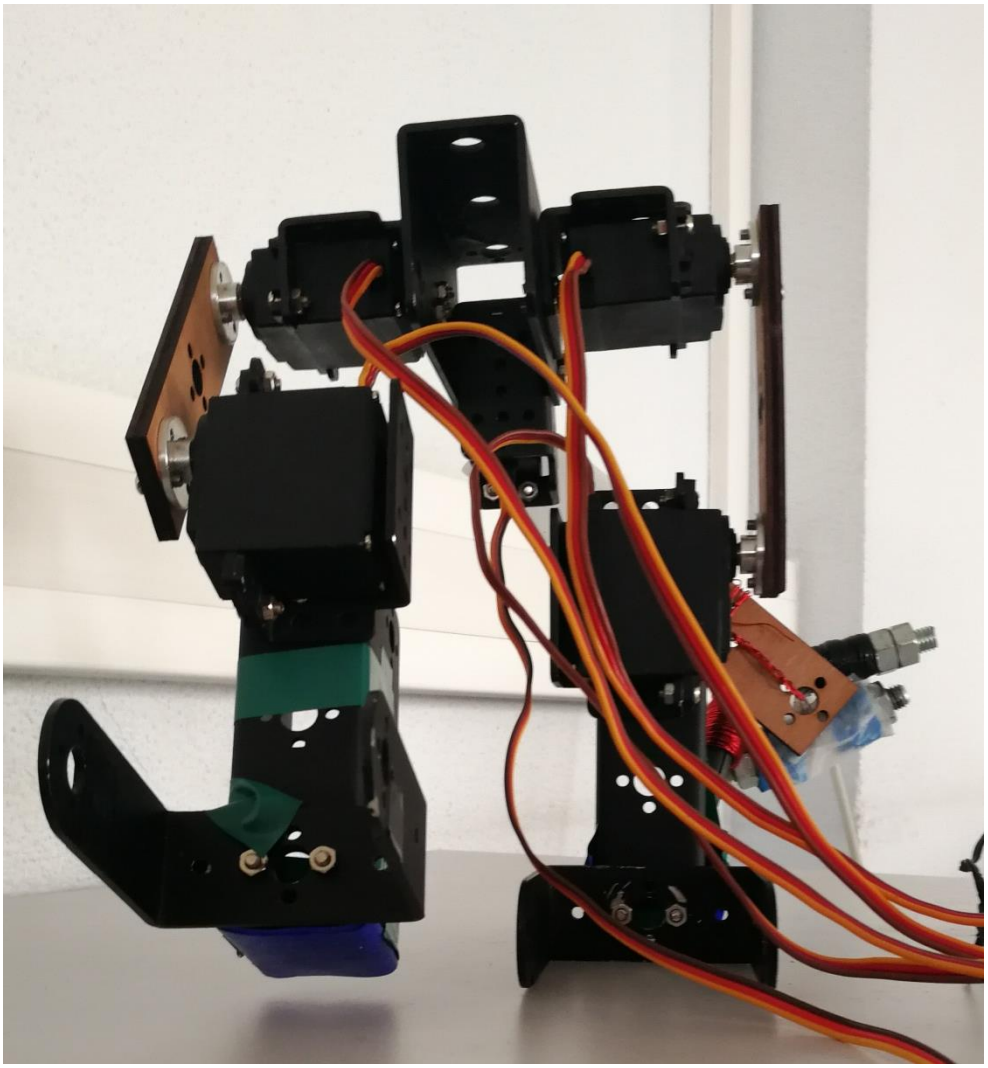
GRAULIER Brice

G3

POLYTECH NICE SOPHIA

Compte Rendu Arduino

Projet Bipède



Année 2017/2018

Sommaire

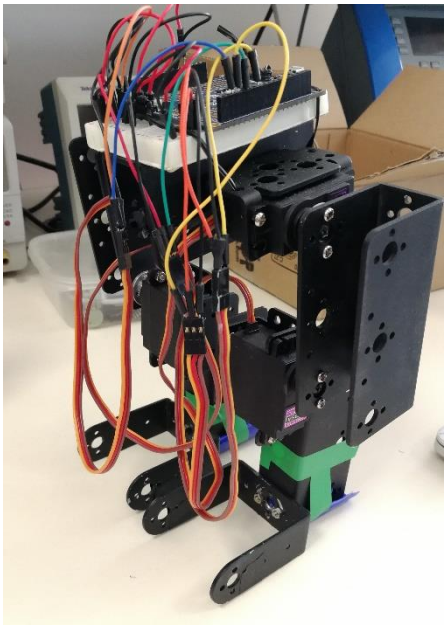
Introduction

On a tous déjà vu un objet, une invention, un concept qui nous fascine, nous c'est le robot Atlas de chez Boston Dynamics. La complexité de sa fabrication, la précision de ces gestes en font probablement aujourd'hui un des meilleurs bipèdes robotisés au monde.

Alors nous on a décidé d'essayer également, après tout pourquoi pas. On s'est équipé d'une bonne dose de patience, de longues heures de réflexion et le tour est joué, on n'a pas Atlas mais nous avons un bipède qui marche, certes plus proche du T-Rex que de l'humain mais il reste cependant, un bipède !

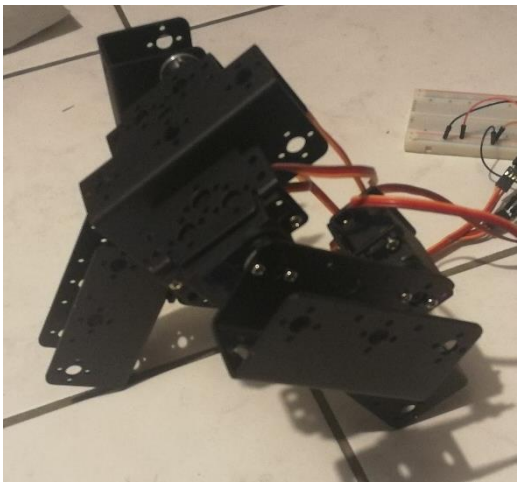
Pour comprendre un peu plus les différences voici quelques photos de Atlas

Et voici le nôtre :



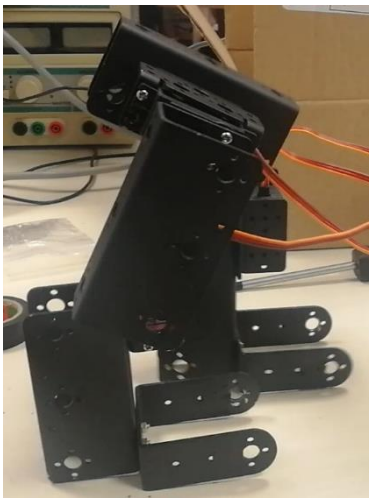
Par contre il ne sait pas faire des saltos, dommage on aurait aimé également.

Evolution physique



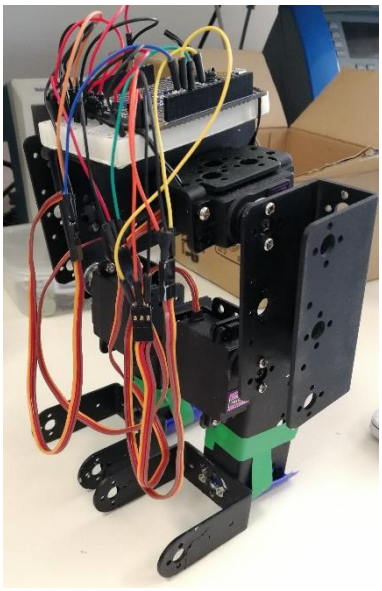
Janvier 2018

Cette version simple était assez basique, pas de pied, pas de queue, il marchait mais glissait, tombait sur le côté assez régulièrement



Février 2018

Sur cette version a été rajouté les pieds qui lui offrent une plus grande stabilité lors de la marche, il ne chutait plus mais glissait sur le sol et faisait du sur-place.



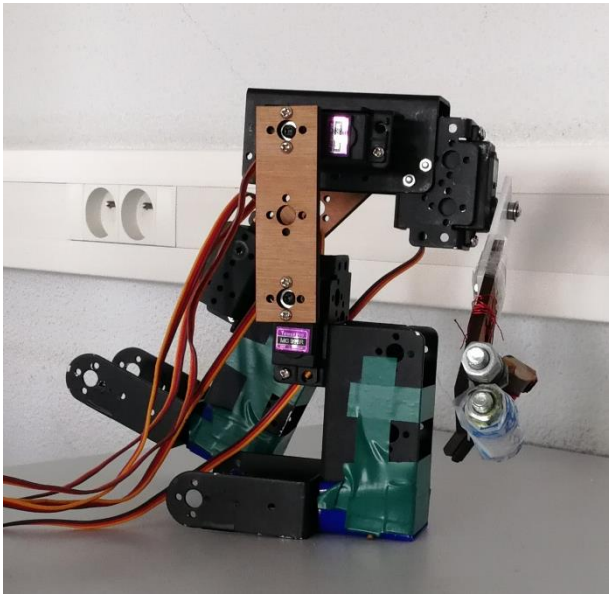
Mars 2018

Cette version-là a eu 2 ajouts majeurs, un qui a été conservé et un qui a été enlevé.

La première a été l'ajout de semelle, une feuille très adhérente fixé avec du scotch.

La seconde a été de monté la plaque et son support sur sa tête. Le jour où on a appliqué ces modifications, le robot marchait très bien, puis nous avons voulu ajouter le module bluetooth et cela a créé plein de faux contact et le robot ne répondait plus du tout comme on désirait.

Même après l'avoir enlevé, le robot avait encore des disfonctionnements, nous avons donc laissé la carte et son support au sol.



Mai 2018

Version finale du robot. On lui a ajouté la queue avec les poids pour qu'il soit plus stable. La plaque a été redescendu au sol à cause des faux contacts de la version précédentes.

Certaines pièces ont été remplacées par des pièces en bois afin d'alléger le robot, découpées à la découpeuse laser.

Evolution du code

Le code du robot est simple. Celui-ci ne fait que contrôler les moteurs.

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

Servo droite; // base 60;
Servo gauche; // base 64
Servo gDroit;
Servo gGauche;
Servo Queue;
int initDroite = 65;
int initGauche = 60;
int initgDroit = 58;
int initgGauche = 55;
int initQueue = 85;
```

Les cinq moteurs du robot sont établis avec une position initiale qui sera l'angle au repos. J'utilise des fonctions de marche simples pour coordonner des mouvements de base. Les enchainements permettent une marche presque fluide.

Au départ le robot avait du mal à lever la jambe et glissait sur le sol. Un ajout de la queue qui vient rééquilibrer permet une marche plus fluide.

```
void rightStep(){

  droite.write(initDroite + 25);
  gDroit.write(initgDroit + 25);
  delay(300);
  gauche.write(initGauche + 30);
  gGauche.write(initgGauche + 30)

  delay(400);

  droite.write(initDroite);
  gDroit.write(initgDroit);
  delay(400);
  gauche.write(initGauche);
  gGauche.write(initgGauche);

  delay(500);

}
```

```
void rightStep(){

  Queue.write(initQueue - 60);
  delay(300);
  droite.write(initDroite + 25);
  gDroit.write(initgDroit + 25);
  delay(300);
  gauche.write(initGauche + 30);
  gGauche.write(initgGauche + 30);

  delay(400);

  droite.write(initDroite);
  gDroit.write(initgDroit);
  delay(400);
  gauche.write(initGauche);
  gGauche.write(initgGauche);
  delay(200);
  Queue.write(initQueue);

  delay(500);

}
```

Les angles sont choisis après plusieurs tests pour être sûr que le robot ne tombe mais fait d'assez grands pas pour se déplacer à une vitesse qui nous convenait. La queue est ramenée assez haute pour ne pas gêner les jambes durant le déplacement mais pas trop pour déséquilibrer le robot.

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

#define TxD 12
#define RxD 11
SoftwareSerial BTSerie(RxD,TxD);


Servo droite; // base 60;
Servo gauche; // base 64
Servo gDroit;
Servo gGauche;
Servo Queue;
int initDroite = 65;
int initGauche = 60;
int initgDroit = 58;
int initgGauche = 55;
int initQueue = 85;
char recvChar;

void setup() {

//BlueTooth Setup
  pinMode(RxD,INPUT);
  pinMode(TxD,OUTPUT);
  BTSerie.begin(9600);


  droite.attach(2);
  gauche.attach(3);
  gDroit.attach(4);
  gGauche.attach(5);
  Queue.attach(7);
//Remettre les jambes à 0
  gauche.write(initGauche);
  droite.write(initDroite);
  gDroit.write(initgDroit);
  gGauche.write(initgGauche);
  Queue.write(initQueue);
  Serial.begin(9600);
}
```

```
void loop() {

  delay(500);

  if (BTSerie.available()) {
    recvChar = BTSerie.read(); //lecture
    if(recvChar=='U'){
      twoStep();
    }else if(recvChar=='L'){
      leftStep();
    }else if(recvChar=='R'){
      rightStep();
    }
    Serial.print(recvChar); //ecriture
  }

  if (Serial.available()) {
    recvChar = Serial.read(); //lecture

    BTSerie.write(recvChar); //ecriture
  }
}
```

L'installation bluetooth est très simple. Les boutons sur le téléphone renvoient juste un caractère pour décider l'action du robot (avancer, tourner à gauche ou tourner à droite)

```
void leftStep(){

  Queue.write(initQueue + 70);
  delay(300);
  gauche.write(initGauche - 25);
  gGauche.write(initgGauche - 25);
  delay(300);
  droite.write(initDroite - 25);
  gDroit.write(initgDroit - 25);

  delay(400);

  gauche.write(initGauche);
  gGauche.write(initgGauche);
  delay(400);
  droite.write(initDroite);
  gDroit.write(initgDroit);
  delay(200);
  Queue.write(initQueue);

  delay(500);

}

void twoStep(){
  leftStep();
  delay(100);
  rightStep();
}
```

Le matériel actuel

Le robot dispose actuellement de 5 servo-moteurs, ils peuvent tous être assimilés à des articulations du corps humains, deux sont utilisés pour les genou (droit et gauche) , deux pour les hanches et un pour la queue, on reviendra sur l'utilité de la queue un peu plus tard.

L'armature est réalisée à partir de pièces en métal et de pièces en bois, découpé à la découpeuse laser au FabLab.

La communication avec le robot se fait grâce à un module bluetooth directement avec le téléphone.

Le robot dispose au niveau de la queue de 4 poids.

En dessous de ses pieds il dispose de semelles qui permettent une meilleure adhérence au sol

En liste :

- 5 servo-moteurs
- 6-10 pièces métallique
- 3 pièces en bois
- 4 poids
- 2 semelles adhérentes

Schéma du robot

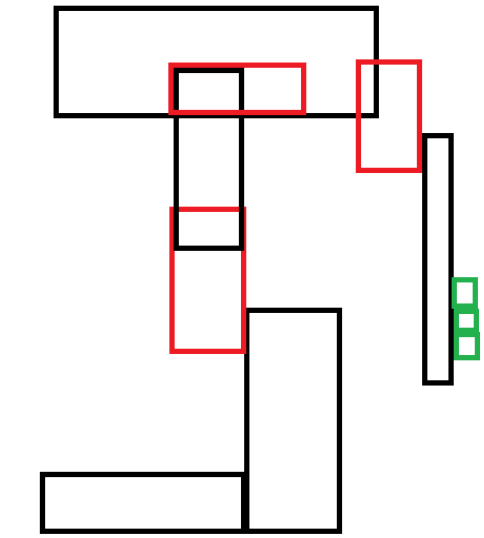


Schéma simplifié vu de côté

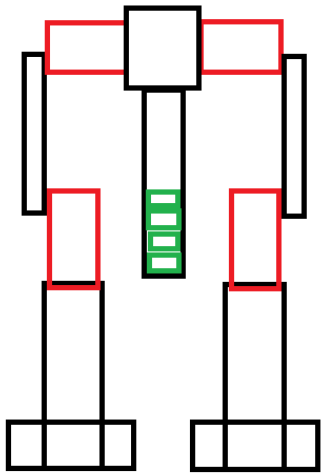


Schéma simplifié vu de devant

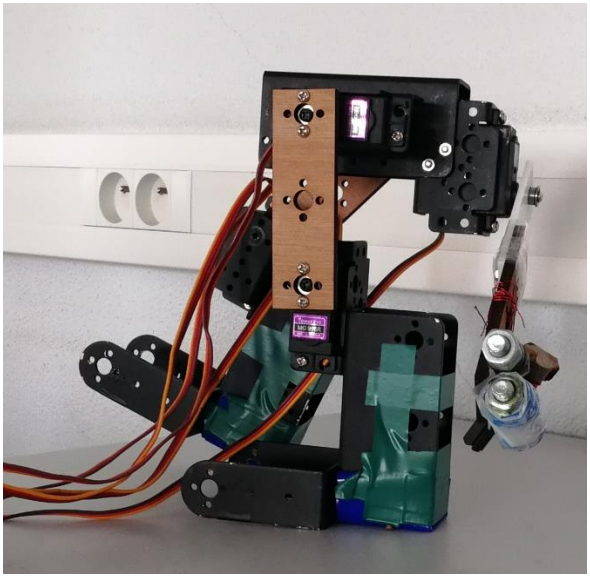


Photo de côté du robot

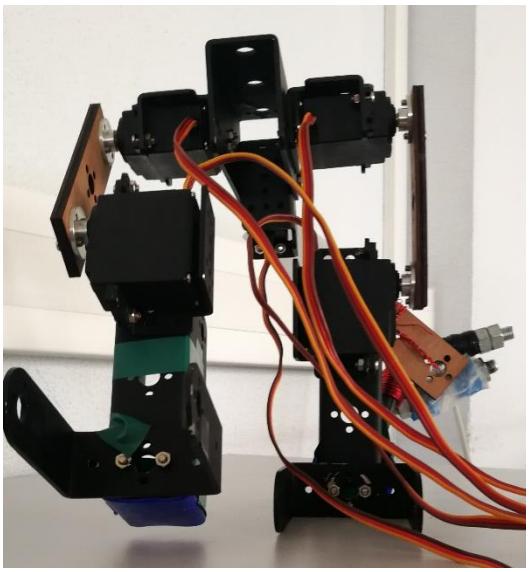


Photo de face du robot

Sur les schéma le rouge correspond aux servo-moteurs et en vert ce sont les poids.

Les servo-moteurs sont très pratique car ils permettent d'imposer un angle et ils s'occupent de le maintenir, ce qui est parfait à la vue du poids du robot, et permettent de faciliter la marche.

Les poids accrochés derrière permettent de déplacer le centre des masses du robot à droite et à gauche afin de lever le pied opposé sans perdre l'équilibre. Sur les photos on voit qu'il n'est pas tenu, son pied droit est levé et les poids sont décalés vers le pied gauche. Il tient donc très bien l'équilibre.

L'algorithme du pas droit, par exemple, se décompose en :

1) Je décale le poids à gauche
2) Je lève ma jambe droite
3) Je plie ma jambe gauche
4) Je ramène ma jambe droite
5) Je déplie ma jambe gauche en remettant le poids au milieu

Problèmes rencontrés

Problème de glisse du robot :

Le robot n'arrivait pas à se « tirer » vers l'avant lorsqu'il faisait un pas et faisait du sur-place.

→ Résolu par l'ajout de semelles sous les pieds qui augmentent énormément l'adhérence du robot.

Problème de poids du robot :

Le robot était trop lourd et avait un équilibre assez précaire lors de sa marche, de plus lors de la marche, les servo-moteurs avaient du mal à bouger le robot

→ Résolu par le changement de 2 pièces en métal et trop grande/large par 2 plaques en bois, plus légère et plus fine.

Problème de stabilité lors de la marche :

Le robot était peu stable et il avait du mal à avancer

→ Résolu par l'ajout d'une queue avec des poids

Problème de faux-contact :

Comme dit précédemment, le robot avait de nombreux problème de faux contact ce qui rendait les servo-moteurs hors de contrôle.

→ Non résolu

Améliorations/ouverture possibles

1) Autonomie

Il serait intéressant de lui rajouter une batterie pour ne plus nécessiter de branchement à l'ordinateur

2) Liberté de mouvement

Il faudrait trouver une solution pour ces problèmes de faux-contact afin de ramener tout l'arduino et sa plaque sur le dessus

3) Esthétisme

Trouver un moyen de cacher les câbles des servo-moteurs.

4) Capteur de distance

Rajouter un capteur de distance qui empêche le robot d'avancer si il est trop près d'un mur.

5) Mouvement

Trouver un moyen de fluidifier ses mouvements et d'en rajouter, par exemple monter une marche.

Conclusion

Notre projet est presque abouti, il reste cependant quelques problèmes à résoudre et des optimisations/ améliorations possibles.

Nous n'avons bien évidemment pas du tout atteint le niveau d'atlas, cependant c'est un bon début !