# Présentation du Projet Long

Samuel ELBEZ & Pierrick JACQUETTE 21200353 & 21305551

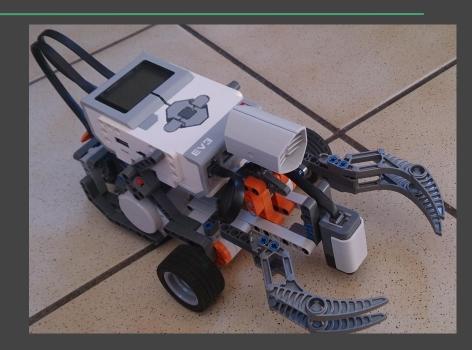


### Définition du système

Robot Lego EV3 Mindstorm (2 motors et d'un capteur HiTechnicColorSensor)

Lejos: JAVA

Suiveur de ligne



### Les fonctionnalités

Afficher les valeurs lues par le capteur

Apprentissage des couleurs

Suiveur de ligne:

- l'un se focalise sur la rapidité
- l'autre retrace le parcours

# Les différents algorithmes

	Temps en seconde	Avantages	Inconvénients
Simple	35	Sur la ligne	Lent
PID	30	Rapide	Sur l'interligne Inutilisable avec une fonctionnalité annexe
Draw	42	Sur la ligne Dessine le circuit	Lent

### Développement dans le Temps

- 1. Reconnaissance des couleurs 1 mois capteur défectueux
- 2. 1ère version de l'algorithme + affinement 1 mois
- 3. Evolution en Algorithme Simple 2 semaines
- 4. Gestion des points 4 mois API peu clair
- 5. Implémentation d'une version avec PID 1 semaine

### Conception

Pour chaque problème, un package et une version de Robot différente

Robot gère l'API LEJOS et les périphériques LEGO

#### Besoins:

- Apprendre l'API LEJOS souvent @deprecated
- Bases en JAVA

### Gestion de projet

Beaucoup de réflexion à deux

1ère implémentation à deux

Test et implémentation solo car une seule machine

Tests sur circuits personnels ou celui fournit le mercredi

## Programmation - pivot du robot sur lui-même

```
private int pivot() {
this.ev3.stop();
int vitesseRight = this.ev3.getVitesseRight();
int vitesseLeft = this.ev3.getVitesseLeft();
int c = (vitesseLeft < vitesseRight) ? 0 : 1;</pre>
                                                                    // je recupere la direction dans lequel le robot tourne
long times2 = 250:
long times = System.currentTimeMillis();
this.ev3.setVitesse(TP, TP);
                                                                     // je remet une vitesse constante
while (seenColor != ev3.SUIVRE) {
                                                                    // tant quil a pas retrouve la ligne
    if (System.currentTimeMillis() - times > times2) {
                                                                     // swap direction
                                                                     // je change la direction
        times2 += (c < 3) ? (c % 2 == 0) ? 250 : 260 : (c % 2 == 0) ? 500 : 520;
       times = System.currentTimeMillis();
   } else {
       if (c % 2 == 0) {
                                                                    // choisis la direction dans laquel je tourne
            ev3.rotateD();
                                                                     // a droite
        } else {
                                                                     // ou
            ev3.rotateG();
                                                                    // a gauche
    seenColor = find.whatColor(ev3.lireColor());
                                                                    // lire la couleur
this.ev3.stop():
this.ev3.setVitesse(vitesseLeft, vitesseRight);
                                                                    // j'inverse la vitesse des roues par rapport à la vitesse initiale
return c % 2;
```

### Conclusion

Nous avons appris les bases de la robotique et développement MindStorm

#### Pour le futur :

- Continuer Algorithme des points en zigzaguant (pas au point sur GIT)
- Passer à un Arduino ou similaire, pour amélioration en rapidité et complexité