



Projet Master Ipsa

Projet pédagogique Mbot2

5TS1
Said Afrite
Salomé Mathé
Samantha Saint-Lary
Odile Schaeffer

30 janvier 2026
M. Ortega & M. Labsir

Introduction - Contexte

Mise en place d'une plateforme robotique



Une avancée pédagogique



Intelligence artificielle embarqué



Systèmes autonomes



Introduction – Objectifs

Concevoir + déployer une plateforme robotique pédagogique

→ 5 robots mBot2



Introduction – Objectifs

Concevoir + déployer une plateforme robotique pédagogique

→ 5 robots mBot2

- 1 Prise en main des robots
- 2 Recherche et découverte des fonctionnalités
- 3 Implémentations de codes

→ **Objectif final : traitement de données avec Matlab**

Sommaire

01

Prise en main des robots

02

Codes et réalisations

03

Critiques et améliorations



01 Prise en main des robots

I. Prise en main des robots

A. Découverte du matériel

Robot Mbot2 : composants

mBot2 shield

compatible with a variety of external components ,and includes a built-in lithium-ion battery.

Quad RGB sensor

Four sensor probes support color recognition ,as well as basic and advanced line detection programs.

CyberPi

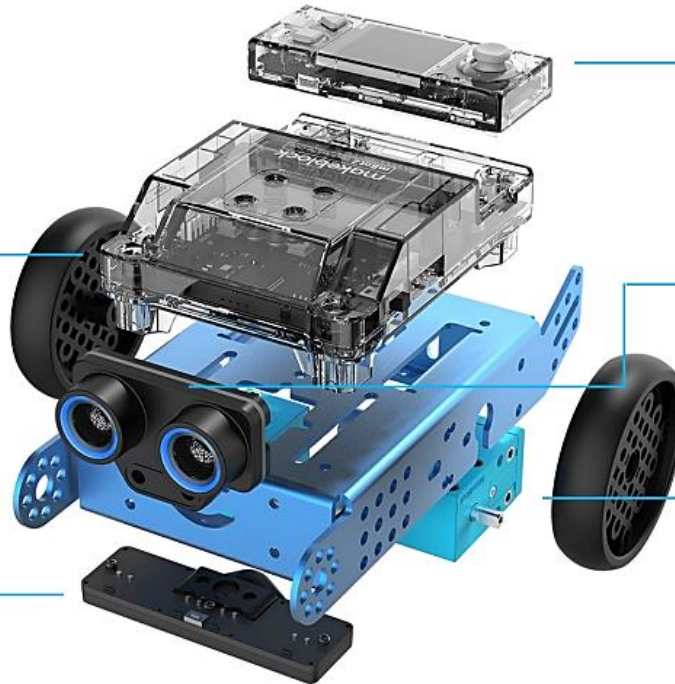
ESP32 microprocessor for wireless communication ,and compatibility with block-based and Python coding.

Ultrasonic Sensor 2

Object detection is accompanied with 8 programmable LEDs for an enhanced interaction.

Encoder Motors

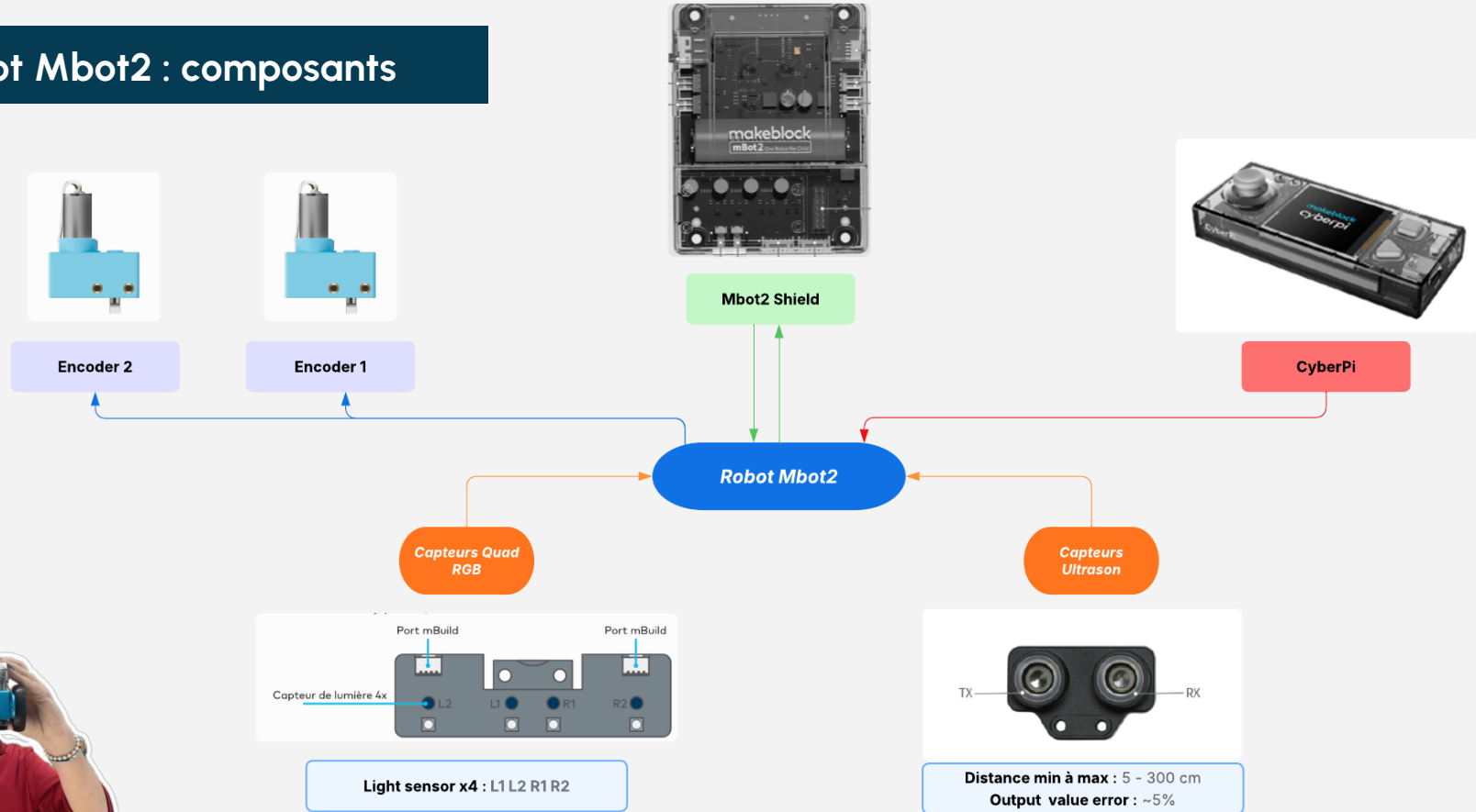
1 degree detection accuracy ,distance traveled, and up to 200 RPM can be precisely controlled.



I. Prise en main des robots

A. Découverte du matériel





Robot Mbot2 : composants



I. Prise en main des robots

A. Découverte du matériel

Robot Mbot2 : avantages

Etapes d'utilisation	Note
Montage	
Première prise en main	
Accès à l'information	
Utilisation du logiciel	



Facile



Abordable/Moyen

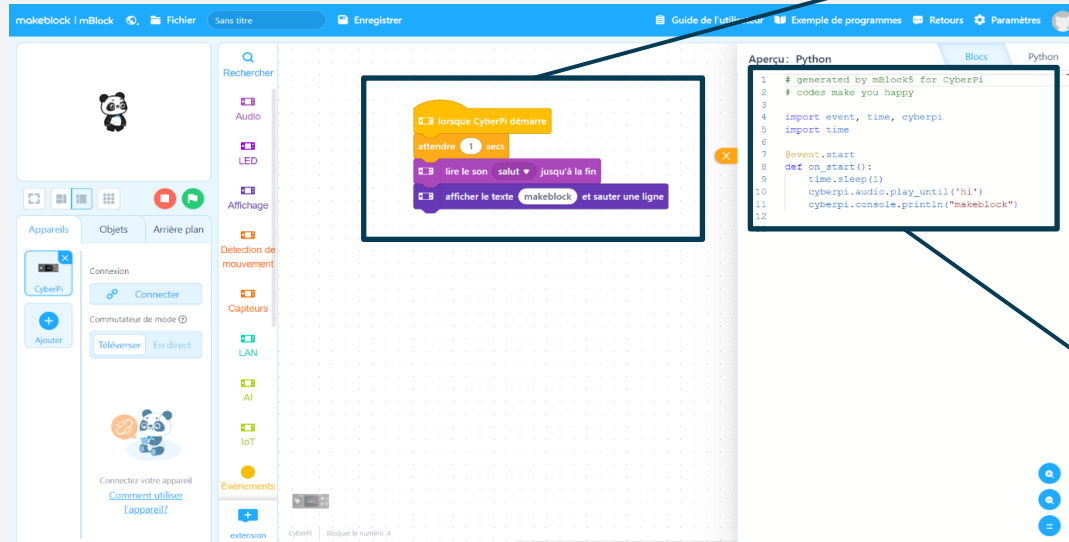


Difficile

I. Prise en main des robots

A. Découverte du matériel

Logiciel mBlock



```
1 # generated by mBlock5 for CyberPi
2 # codes make you happy
3
4 import event, time, cyberpi
5 import time
6
7 @event.start
8 def on_start():
9     time.sleep(1)
10     cyberpi.audio.play_until('hi')
11     cyberpi.console.println("makeblock")
12
```

{ Pédagogique et pensé pour tous

I. Prise en main des robots

B. Réalisation d'une vidéo explicative





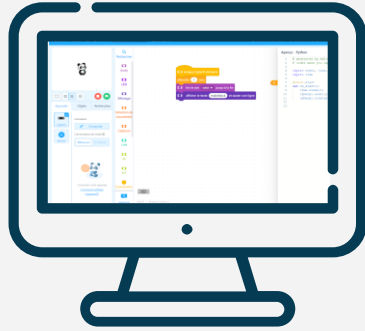
— 02 Codes et réalisations

A. Premiers codes : implémentation simple



II. Codes et réalisations

A. Premiers codes : implémentation simple



Python sur mBlock



mBot2 Exécution



Phase simple

Nombreuses ressources disponibles



Apprentissage rapide

Prise en main du mBot2

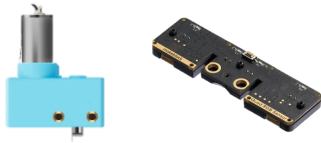


II. Codes et réalisations

A. Premiers codes : implémentation simple

Trois codes fondamentaux

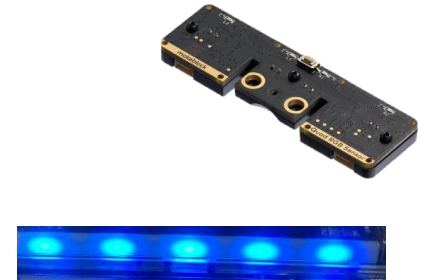
Suivi de ligne



Eviter un obstacle



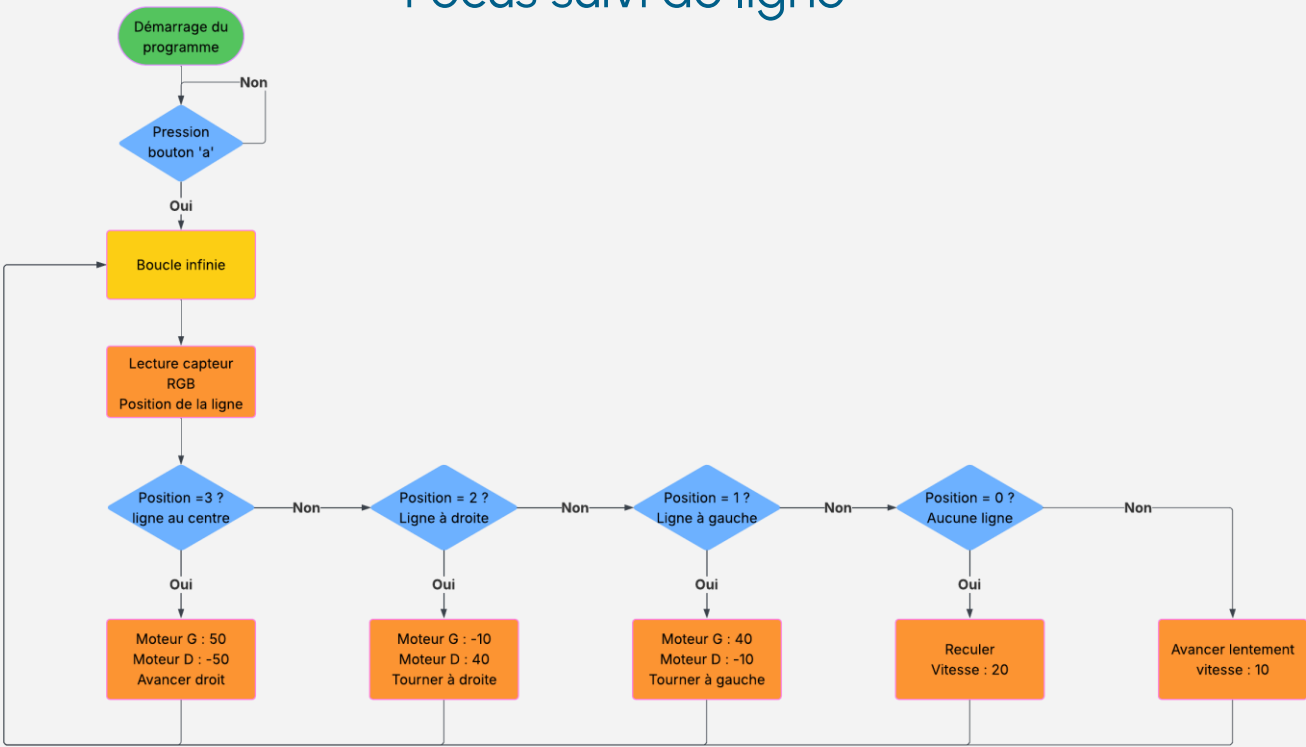
Affichage RGB - LED



II. Codes et réalisations

A. Premiers codes : implémentation simple

Focus suivi de ligne



II. Codes et réalisations

A. Premiers codes : implémentation simple

Focus suivi de ligne

Problèmes rencontrés

- ➡ Utilisation des modes prédéfinis du capteur (valeurs 0, 1, 2, 3)
- ➡ Mouvements saccadés du robot
- ➡ Corrections brutales de trajectoire
- ➡ Une seule roue tournait pour corriger

➡ **Nécessité d'optimiser l'algorithme**

II. Codes et réalisations

A. Premiers codes : implémentation simple

Focus suivi de ligne

Améliorations implémentées

1. Analyse fine des capteurs RGB
Utilisation de **pourcentages de couleur** au lieu de valeurs discrètes
2. Vitesse différentielle adaptative
Réduction progressive de la vitesse en fonction de l'écart
3. Contrôle des deux roues simultanément
Corrections plus fluides par ajustement des **deux moteurs**

➔ **Trajectoire beaucoup plus fluide et précise**

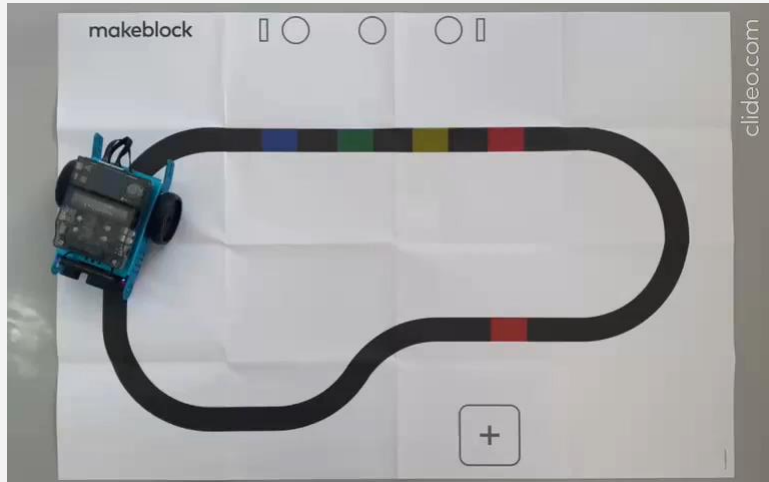


II. Codes et réalisations

A. Premiers codes : implémentation simple

Focus suivi de ligne

Résultats

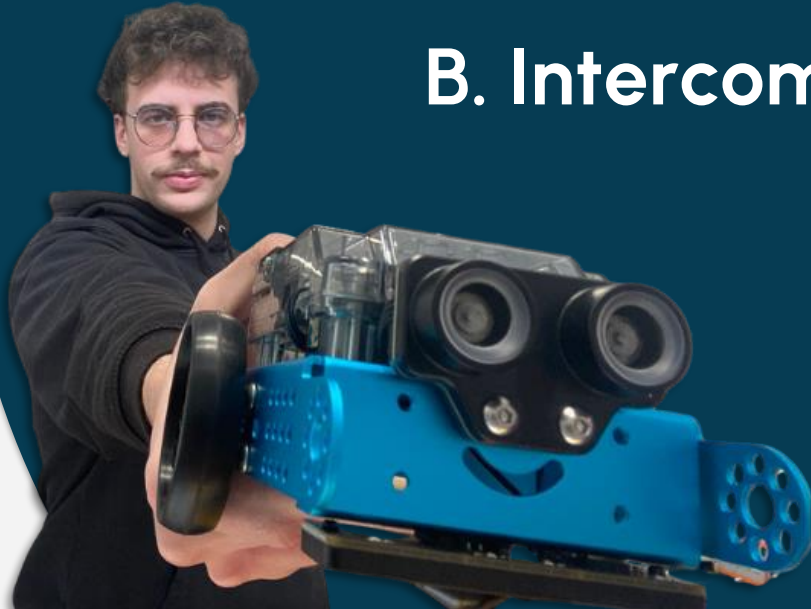


- ➡ Trajectoire fluide
- ➡ Correction progressive
- ➡ Suivi précis et stable
- ➡ Vitesse adaptative



Logique binaire (0/1/2/3) à analyse continue (%) ➡ contrôle plus fin et performances améliorées

B. Intercommunication



II. Codes et réalisations

B. Intercommunication



Python sur mBlock

Wi-Fi



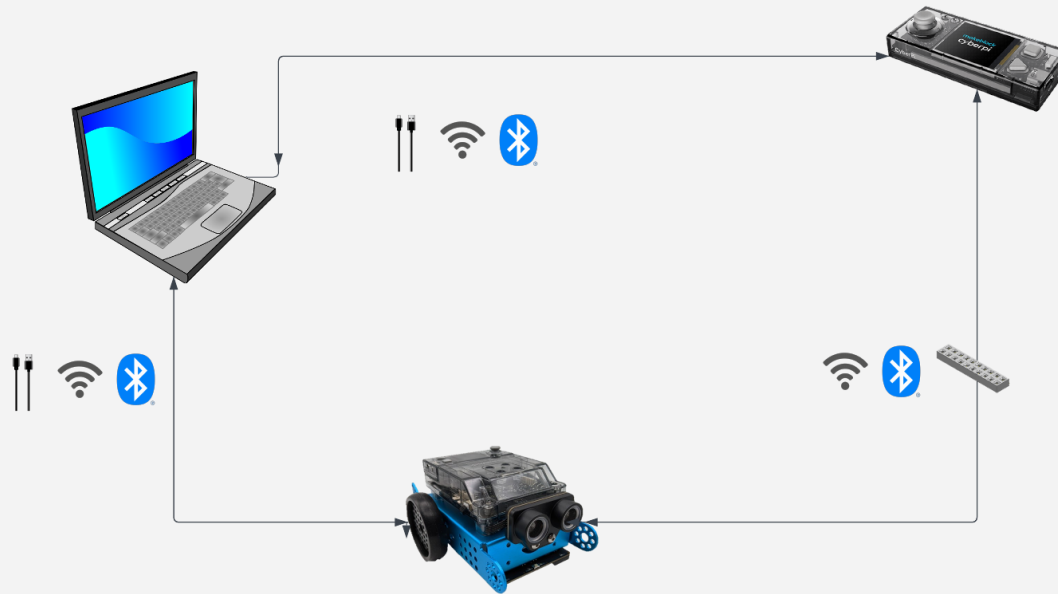
mBot2 Exécution



Phase modérée
Ressources limitées

II. Codes et réalisations

B. Intercommunication



Utilisation du Wi-Fi

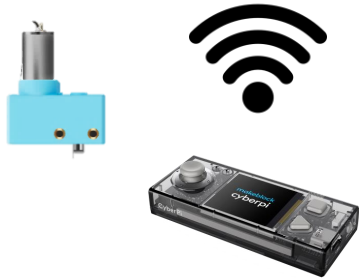
Connection commune
Code mBlock

II. Codes et réalisations

B. Intercommunication

Trois codes fondamentaux

Une télécommande
plusieurs robots



Un robot plusieurs
télécommandes



Communication en
chaîne



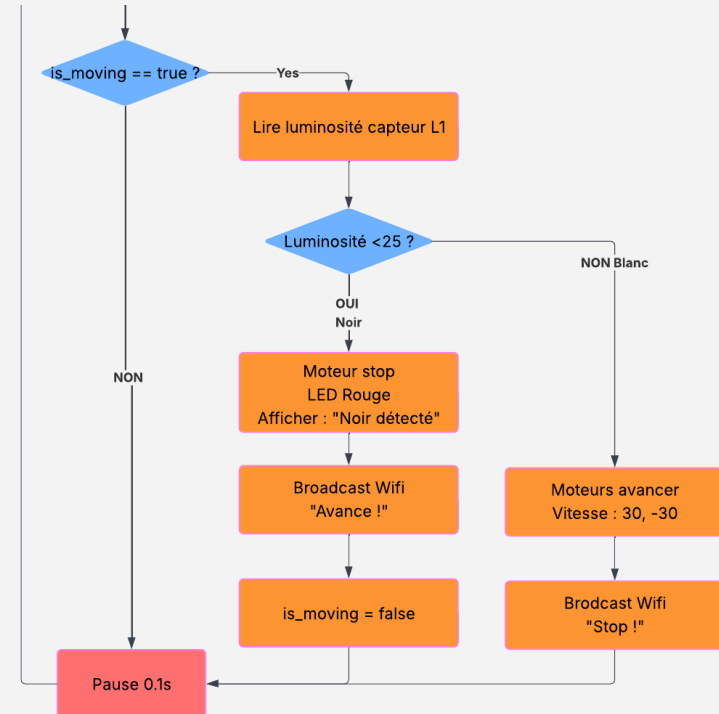
II. Codes et réalisations

B. Intercommunication

Focus communication en chaine

Problèmes rencontrés

- ➡ Saturation du canal de communication.
- ➡ Latences liées à la qualité du reseau.



II. Codes et réalisations

B. Intercommunication

Focus communication en chaine

Améliorations implémentées

1. Mise en place d'une pause entre les envois de données.
2. Mise en place d'un protocole de nettoyage du canal à chaque nouvelle exécution.

➔ **Meilleur réactivité et comportement plus prévisible**

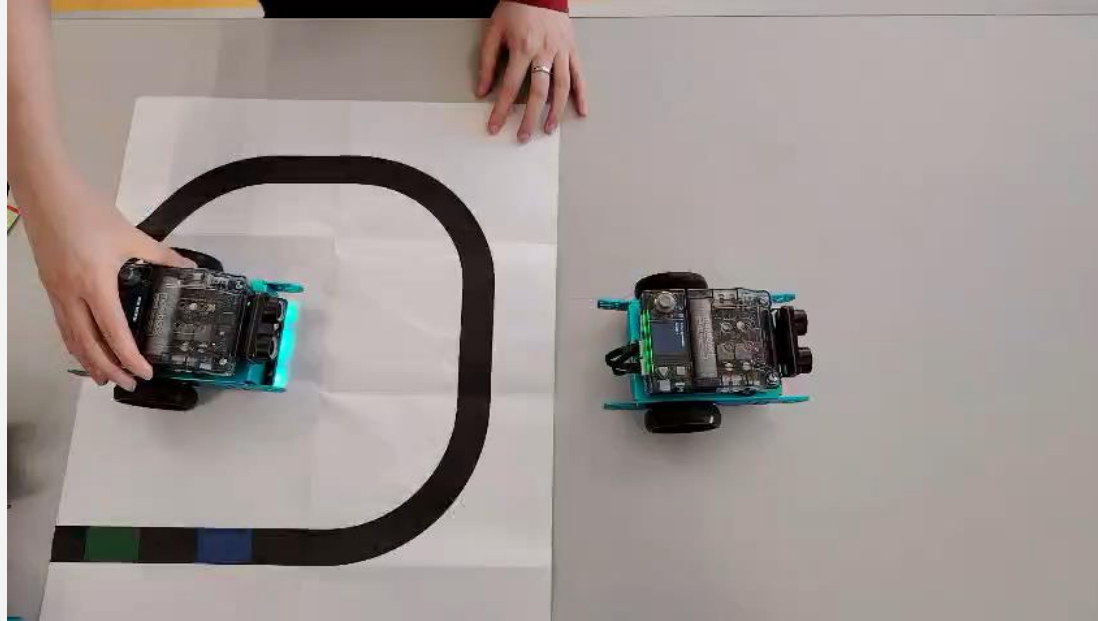


II. Codes et réalisations

B. Intercommunication

Focus communication en chaîne

Résultats

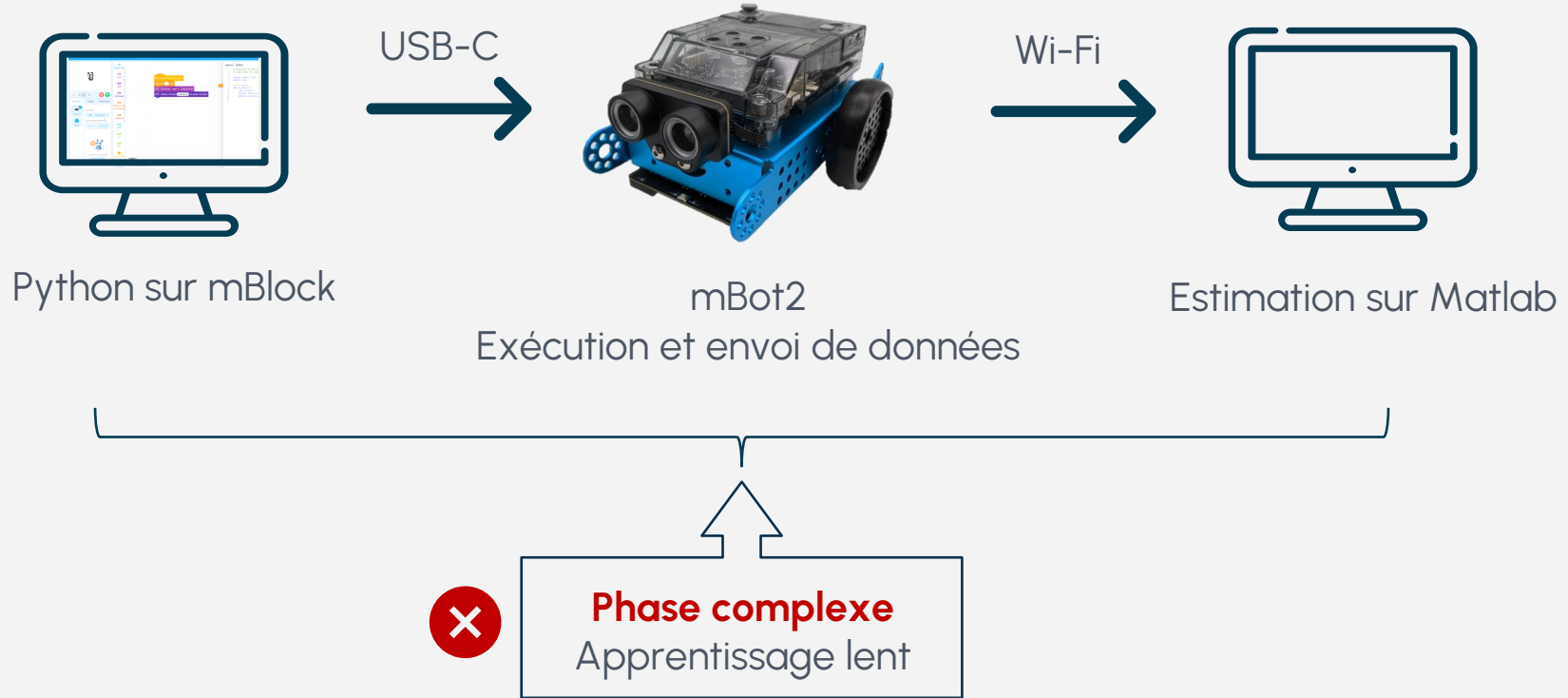




C. Extraction et exploitation des données

II. Codes et réalisations

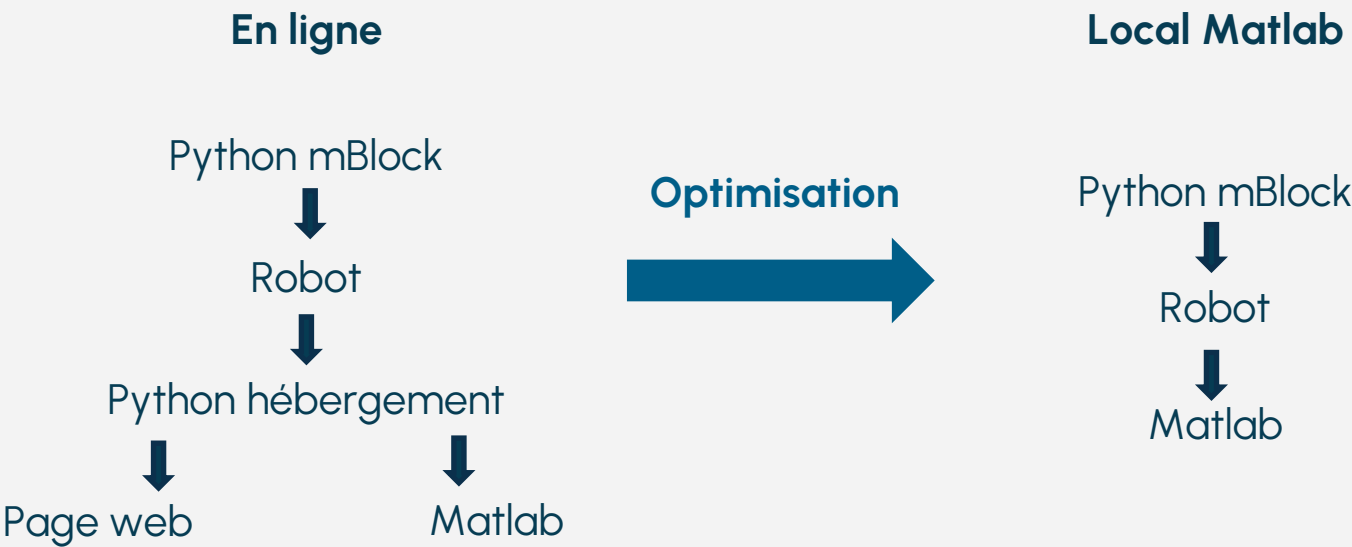
C. Extraction et exploitation des données



II. Codes et réalisations

C. Extraction et exploitation des données

Utilisation des serveurs



II. Codes et réalisations

C. Extraction et exploitation des données

Trois codes fondamentaux

Extraction de distance



Suivi de ligne et
extraction de distance
et RGB



Estimation de
trajectoire



II. Codes et réalisations

C. Extraction et exploitation des données

Focus : Estimation de trajectoire



- ➡ Utiliser le joystick
- ➡ Envoyer la position dans le canal
- ➡ Calculer la trajectoire avec vitesse et angle imposés

Problèmes rencontrés

- ➡ Envoi de messages encombrants dans le canal
- ➡ Latence
- ➡ Mauvais ratio trajectoire réelle/estimée
- ➡ Pas de moyens de comparer les trajectoires réelles/estimées

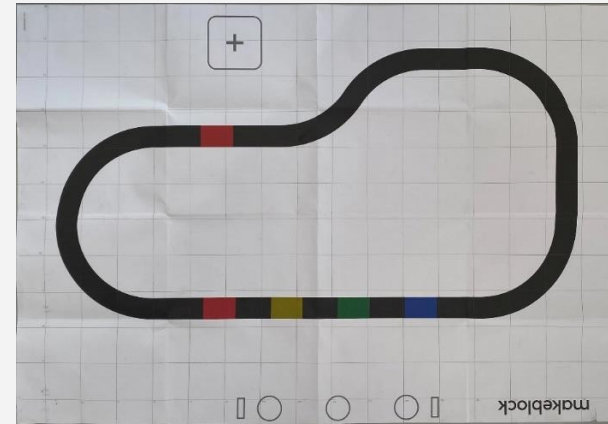
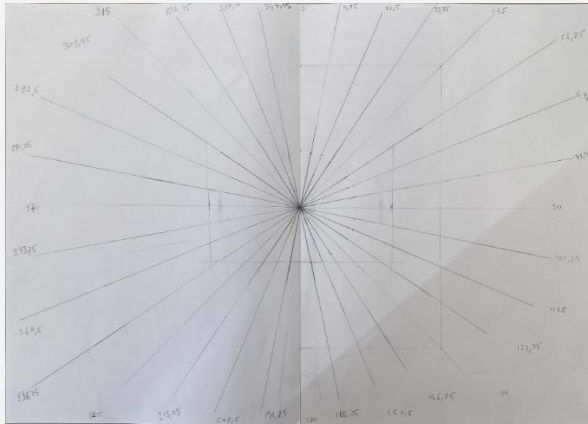
II. Codes et réalisations

C. Extraction et exploitation des données

Focus : Estimation de trajectoire

Calibrage

Feuilles de calibrage → pour trouver un rapport entre déplacement réel et estimé



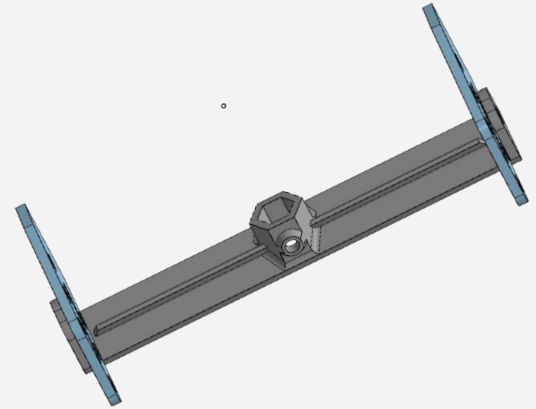
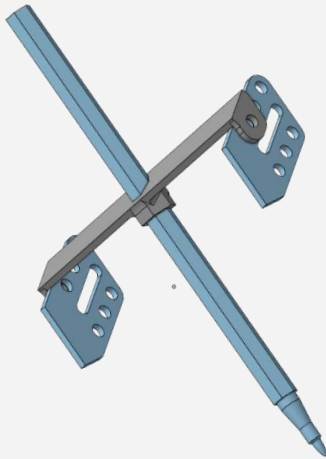
➡ Calibrage angulaire et linéaire

II. Codes et réalisations

C. Extraction et exploitation des données

Focus : Estimation de trajectoire

Création d'un support de stylo



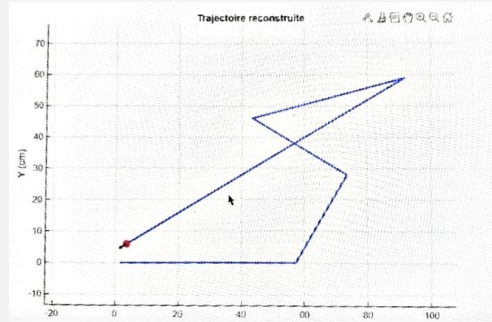
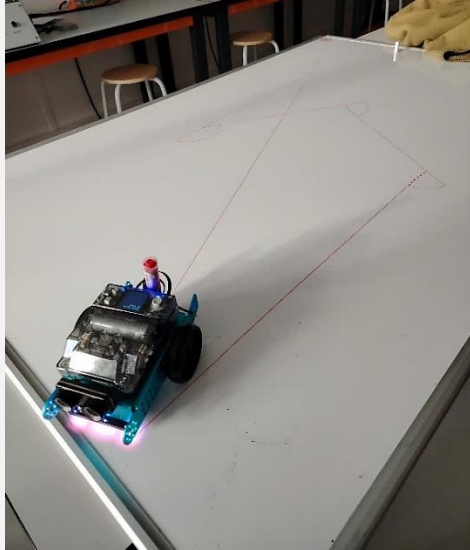
➡ Comparer tracé réel et estimé du robot

II. Codes et réalisations

C. Extraction et exploitation des données

Focus : Estimation de trajectoire

Résultats



- ➡ Trajectoire fluide
- ➡ Retour en temps-réel
- ➡ Optimisation du canal



Calibrage que pour un couple vitesse linéaire et angulaire.



Support de crayon décalé du centre de rotation



Glissement des roues difficile à quantifier



— 03 Critiques et améliorations

III. Critiques et amélioration

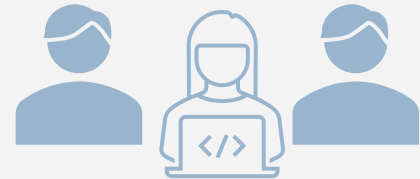
Extraction des données

- Manque de ressources et documentations
 - Pour récupérer les données des capteurs



Logistique

- Robots seulement disponibles à l'IPSA
- Devoir coder sur un seul ordinateur



III. Critiques et amélioration

Améliorations possibles

- Ajouter d'autre capteur (vitesse...)
- Caméra intelligente



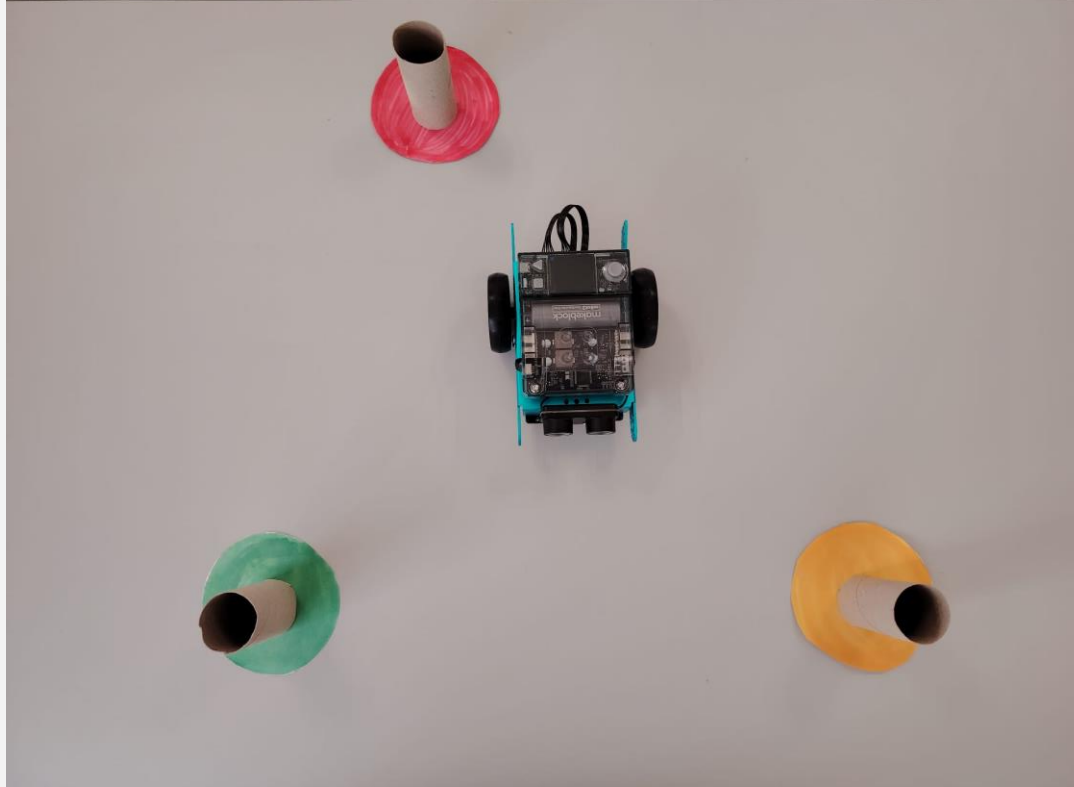
Pour aller plus loin

- Amélioration des codes déjà implémentés (optimisation, méthode de calibrage...)
- Accès à plus de documentation



III. Critiques et amélioration

Landmarks



Conclusion





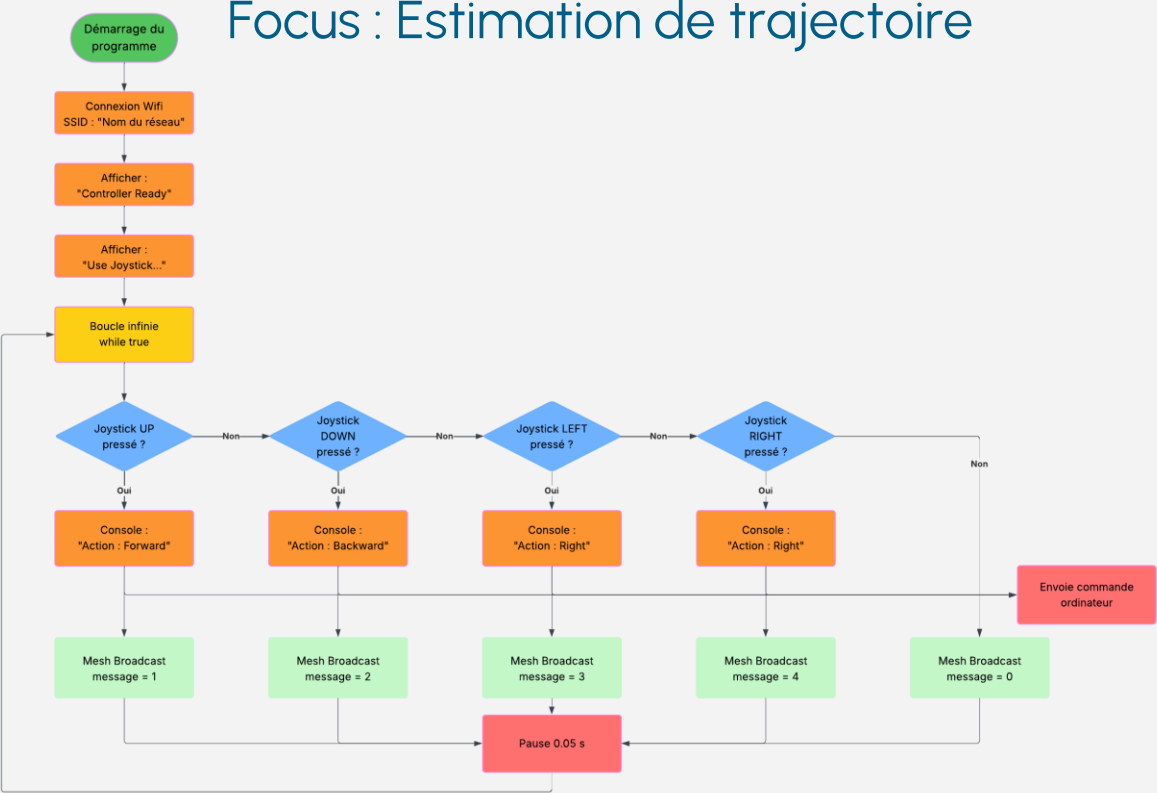
Merci pour votre écoute



II. Codes et réalisations

C. Extraction et exploitation des données

Focus : Estimation de trajectoire



```
1
2  # Wi-Fi credentials
3
4  WIFI_SSID = "Nom_du_réseau"
5
6  WIFI_PASSWORD = "Mot_de_passe_du_réseau"
7
8  # Connect to Wi-Fi
9
10 cyberpi.display.show_label("Connecting WiFi", 16)
11 cyberpi.wifi.connect(WIFI_SSID, WIFI_PASSWORD)
12
13 while not cyberpi.wifi.is_connected():
14     time.sleep(0.5)
15
16 cyberpi.display.show_label("WiFi Connected", 16)
17 time.sleep(1)
18
```

```

1  #computer IP and port
2
3  SERVER_IP = "10.254.141.2" # <-- Your computer IP
4  SERVER_PORT = 5000
5
6  # Create TCP socket
7
8  sock = socket.socket()
9  sock.connect((SERVER_IP, SERVER_PORT))
10
11  cyberpi.display.show_label("Connected to PC", 16)
12  time.sleep(1)
13

```

```
C:\Users\saman>ipconfig
```

```
Carte réseau sans fil Wi-Fi :
```

```

Suffixe DNS propre à la connexion. . . :
Adresse IPv6 de liaison locale. . . . :
Adresse IPv4. . . . . :
Masque de sous-réseau. . . . . :
Passerelle par défaut. . . . . :

```

