



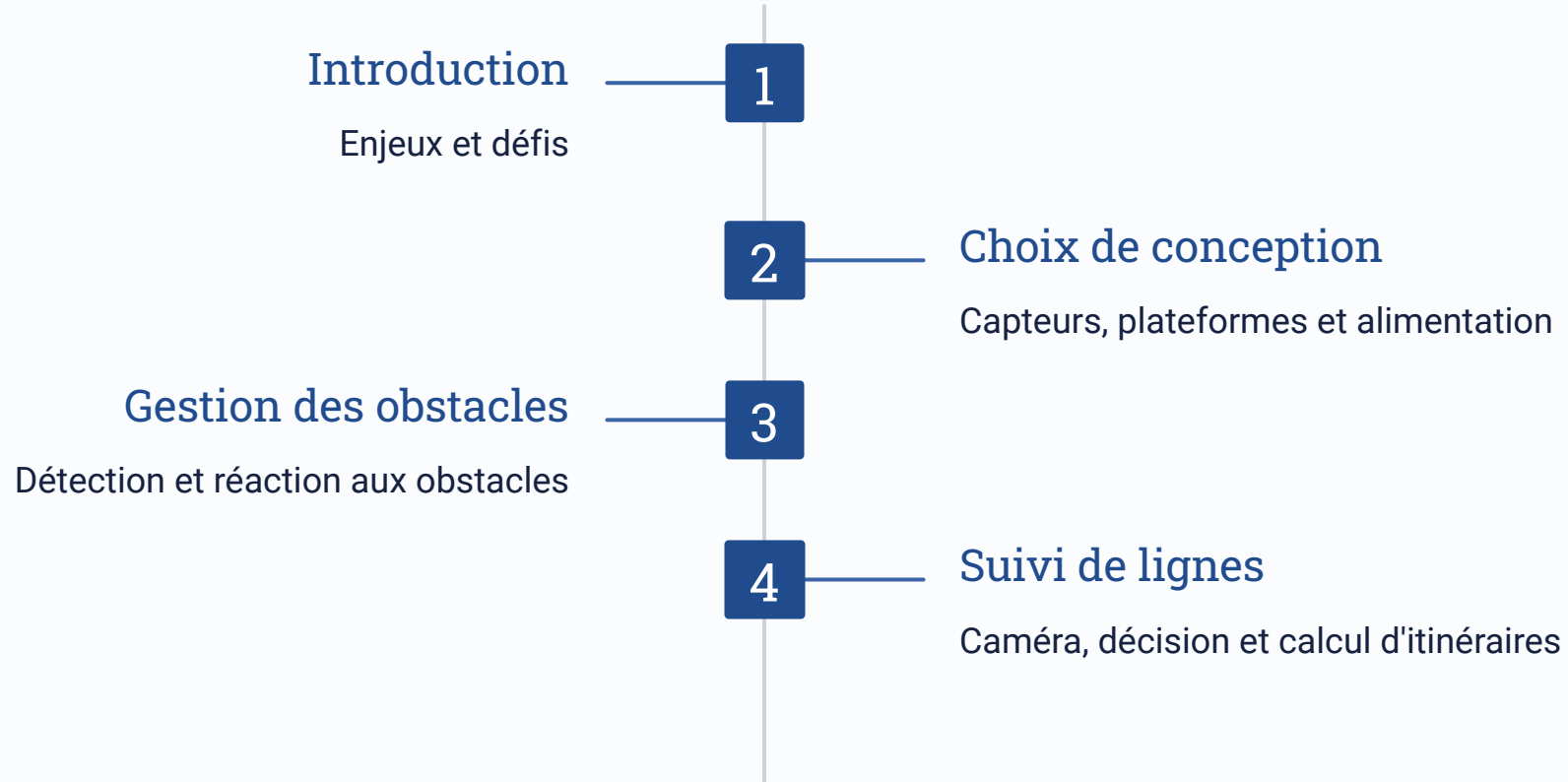
# Enseignement d'intégration

**Véhicules autonomes et connectés**

**Équipe n°4 : FedCS**

Novembre 2024

# Sommaire



# Enjeux et Défis

## 1 Environnement urbain

Livraison à domicile dans un contexte complexe

## 2 Sécurité

Privilégier la sécurité de tous les utilisateurs

## 3 Obstacles

Reconnaissance et gestion des obstacles

## 4 Suivi de route

Navigation précise dans un environnement connu

## 5 Repérage

Localisation précise sur un quadrillage pour navigation



# Répertoire des tâches accomplies



Configuration de la caméra



Création d'une classe Robot



Contrôle des mouvements du robot



Mise en place de la communication série



Développement d'une fonction de suivi de ligne



Détection et gestion des intersections



Intégration de la détection d'obstacles



Implémentation d'une fonctionnalité de balayage



Implémentation d'un retour automatique à la base

# **I. Choix techniques**

# Choix Techniques - Capteurs

## Capteurs à infrarouge

Portée de 80cm

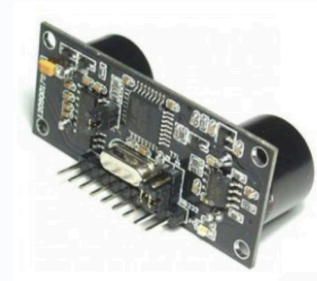
Faisceau laser : très précis mais surface de détection faible



## Capteur à ultrasons

5m de portée et fenêtre de détection de 60°

Ultra son : difficulté de preciser la position de l'obstacle



# Choix Techniques - Alimentation

## Batterie rechargeable

Autonomie et fiabilité optimisées

## Avantages

Flexibilité et adaptation aux  
besoins énergétiques

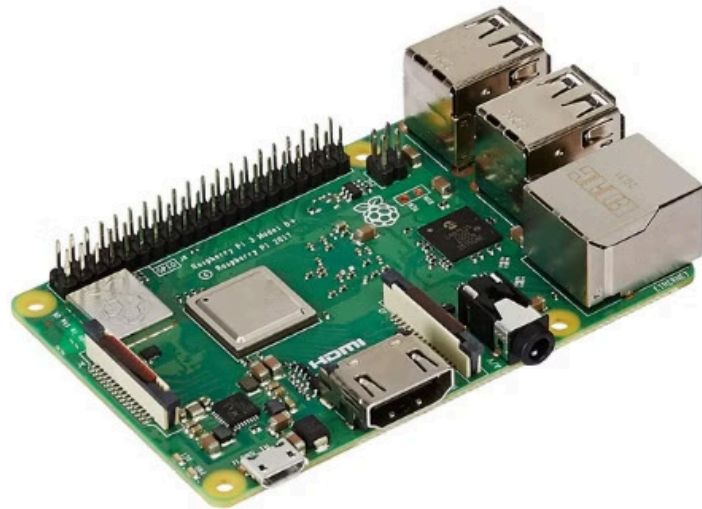
## Comparaison

Plus facilement rechargeable





# Choix Techniques - Plateformes et Langages



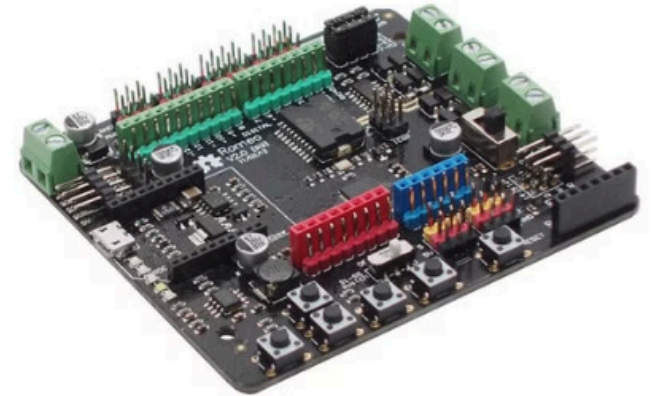
## Raspberry Pi : Carte Maitresse

- S'occupe du suivi de ligne
- Choisit la direction à prendre aux intersections

Ordre de déplacement



Localisation des obstacles

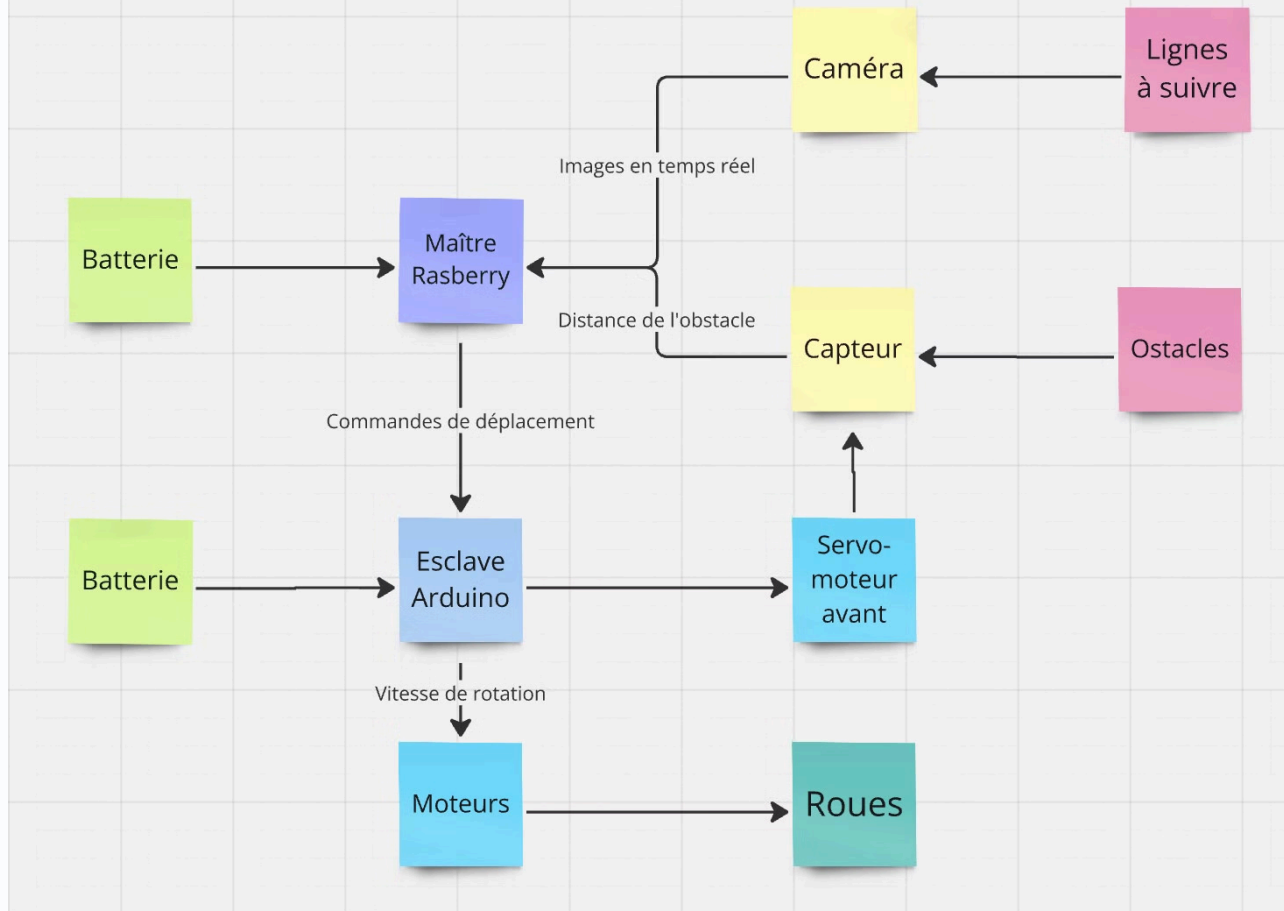


## Arduino : Carte Esclave

- Contrôle les moteurs
- S'occupe de la détection d'obstacle



# Schéma synoptique



## **II. Gestion des obstacles**



# Gestion des Obstacles - Détection

1

Capteur infrarouge

Angle de balayage de  $140^{\circ}$

2

Mode simple

Balayage continu sur  $30^{\circ}$  pour routes simples

3

Mode complexe

Balayage sur  $140^{\circ}$  aux intersections

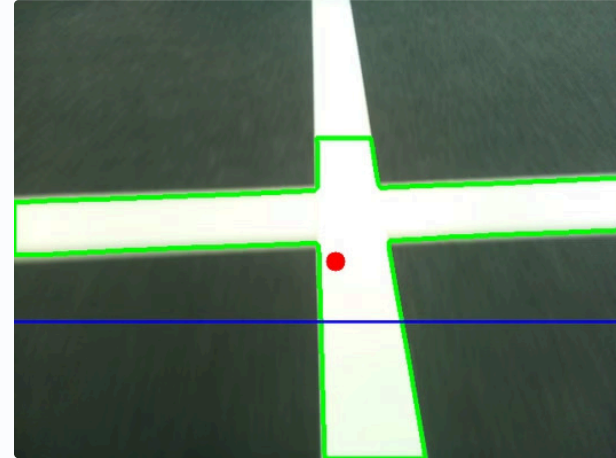
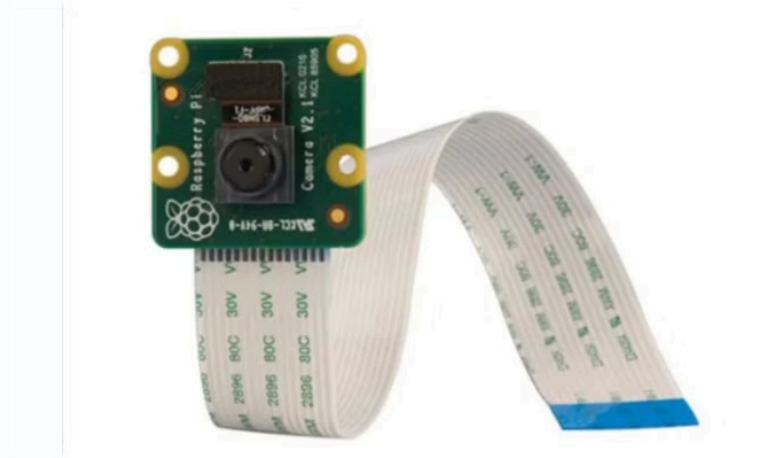
# Réaction aux Obstacles

- 1 **Sécurité**  
Vitesse limitée pour assurer fiabilité
- 2 **Obstacles mobiles**  
Attente jusqu'à ce que l'obstacle disparaisse
- 3 **Obstacles fixes**  
Fais un demi-tour après une durée fixée (ici 5 sec)
- 4 **Problème**  
Ralentissement de certaines opérations dû aux limitations



### **III. Suivi de Lignes**

# Suivi de Lignes - Caméra



## Problèmes

Champ de vision étroit,  
perturbations dans  
environnements variables

## Angle de vue

Champ de vision agrandi  
pour améliorer le suivi de  
ligne

## Luminosité

Luminosité automatique  
retirée pour éviter erreurs de  
détection

## Analyse d'image

Recherche du centroïde,  
détection d'intersection



# Itinéraires et Calcul de Trajectoires

## Algorithme

Simple et adapté à l'environnement avec un calcul en temps réel

## Forces

Rapidité de la prise de décision et de l'adaptation aux changements d'environnement: un huit ou une grille

## Faiblesse

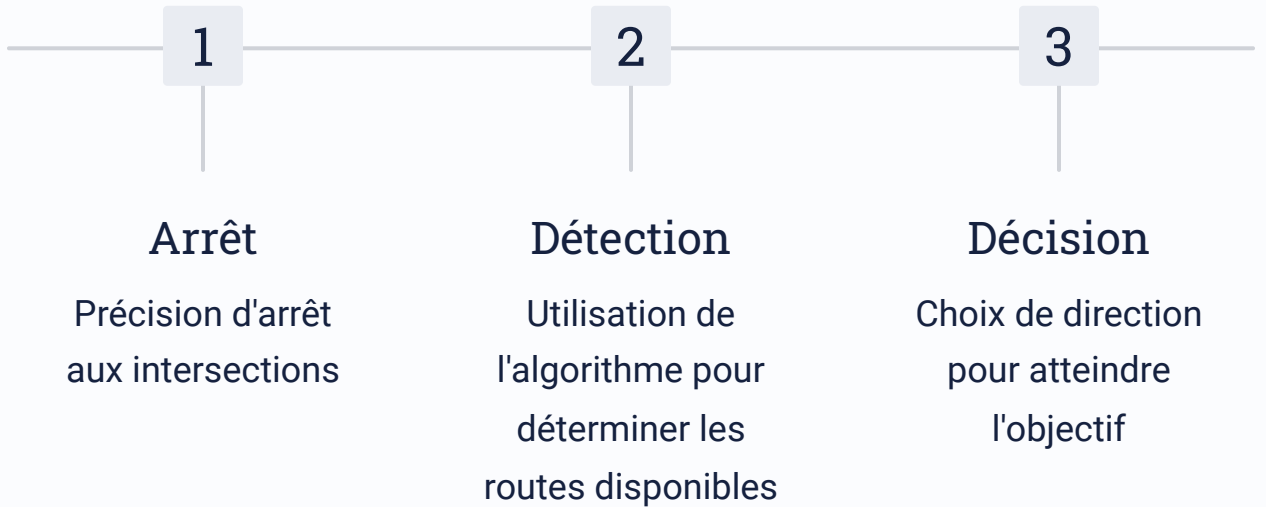
Nécessite des adaptations pour un environnement plus complexe







# Ajustements aux Intersections



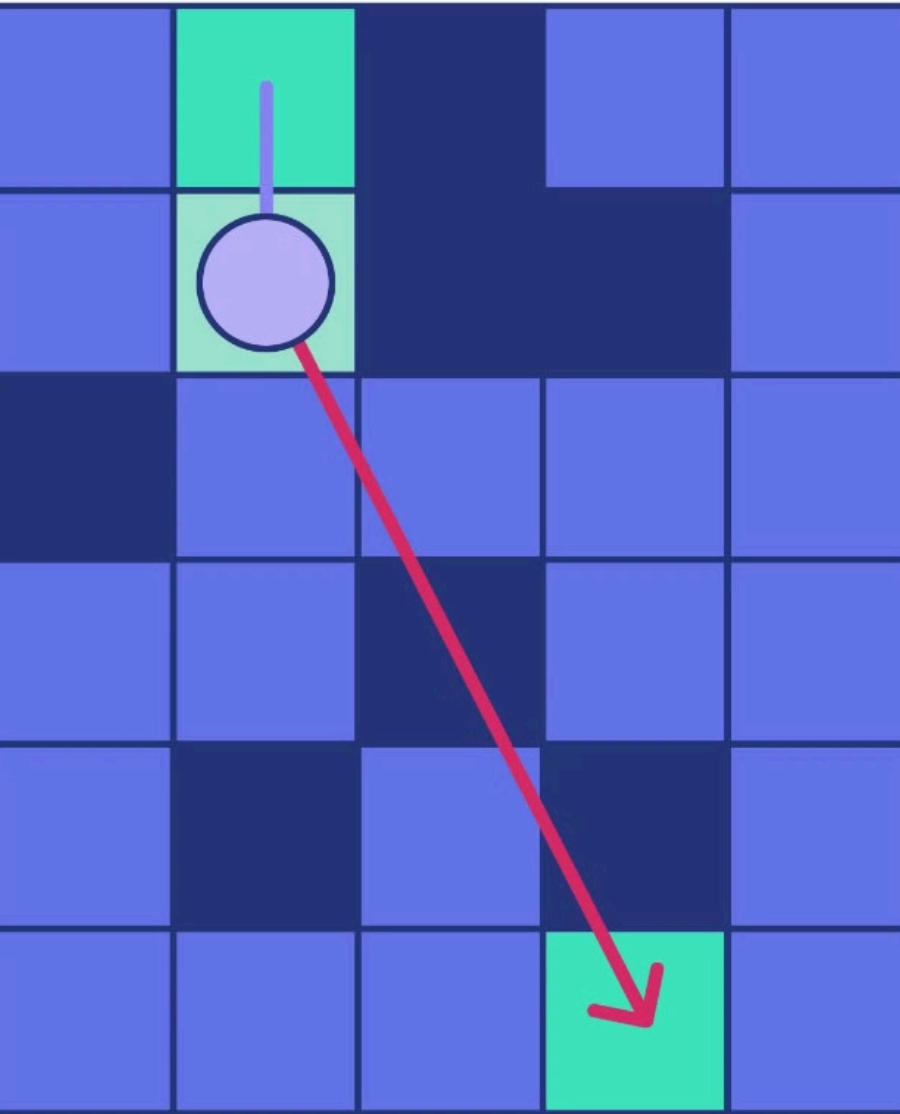
# Conclusion - Forces et Faiblesses

## Forces

- Fiabilité du suivi de ligne - Qualité de la vision
- Détection d'obstacles précise, fiable et préventive
- Ajustements rapides des trajectoires choisies
- Retour à la base à la fin de la mission pour réutilisation

## Faiblesses

- Vitesse réduite
- Limitations techniques des capteurs, caméra et cartes électroniques

**A****B**

## Perspectives d'amélioration

- Lenteur du robot → **sweep** des obstacles plus **rapide**, et **anticiper** avant l'arrêt à l'intersection
- Oscillation du robot entre 2 intersections →  
Implémentation d'un algorithme plus robuste: **A\***
- **Interface graphique** pour une meilleur contrôle et suivi du Robot en temps réel



# Questions et Discussions



Choix techniques



Défis rencontrés



Perspectives d'amélioration

# Répertoire des tâches accomplies

## Configuration de la caméra

Capturer des images en temps réel et pouvoir y accéder depuis la RPI.

## Création d'une classe Robot

Avec des attributs de position, direction, état, et mission pour gérer le comportement global du robot.

## Contrôle des mouvements du robot

Incluant les fonctions pour avancer, tourner, et faire demi-tour en fonction des obstacles détectés.

## Mise en place de la communication série

Avec l'Arduino pour envoyer des commandes de contrôle moteur et de navigation.

## Développement d'une fonction de suivi de ligne

Avec détection d'erreur et adaptation de la vitesse de chaque roue en fonction de l'erreur.

## Détection et gestion des intersections

Avec des choix de direction automatisés en fonction des priorités de mission.

## Intégration de la détection d'obstacles

A l'aide d'un capteur infrarouge, avec des seuils pour la distance d'arrêt.

## Implémentation d'une fonctionnalité de balayage

Avec le servo-moteur pour une détection étendue et robuste de l'environnement à 160°.

## Implémentation d'un retour automatique à la base

Lorsque le robot a terminé sa mission, avec ajustement de la direction en temps réel.

# Schéma synoptique

