

Ingénierie système

Robot ramasseur de balles

23 février 2023



F. JERRAM

L. DROUDUN

E. DECAUDAVEINE

D. POP

J. HODEK



Introduction

Objectif gestion : mettre en place la méthode agile et les outils associés

Objectif technique : Ramasser des balles de tennis qui arrivent une à une sur un terrain



Sommaire

- Stratégie initiale
- Demandes du client
- Architecture ROS2
- Réalisation technique
- Démonstration
- RETEX (ROS 2, Github, Taiga)



I. Premiers pas



- Choisir les capteurs
- Premier modèle du robot
- Détecter les balles grâce à la caméra zénithale
- Définir la stratégie de récupération des balles
- Définir le contrôle du robot
- Faire les ajustements nécessaires

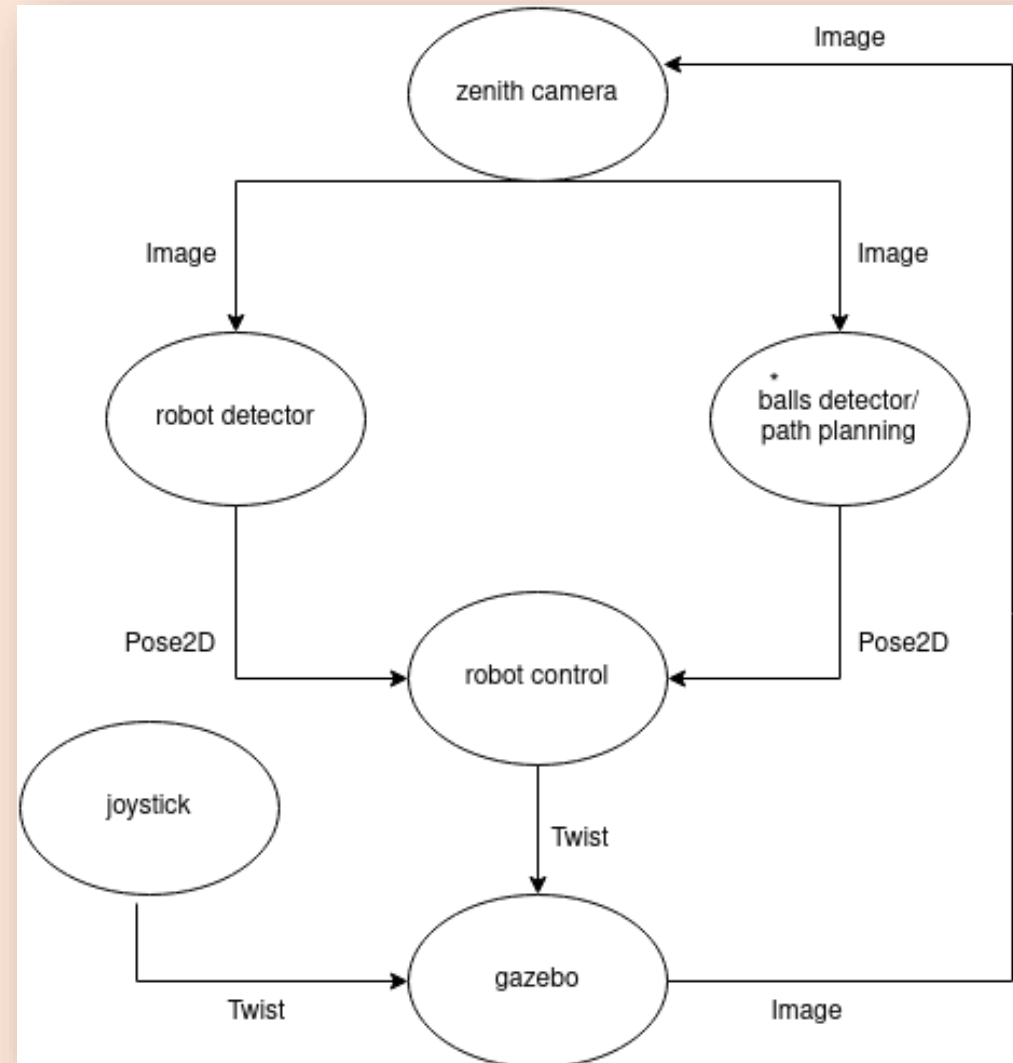


II. Demandes du client

- Plus la balle est ramassée rapidement, plus elle rapporte de points
- Robot rouge avec roues vertes
- Doit rentrer dans une boîte de 50x50x30 et peser moins de 15kg
- Le robot doit pouvoir être contrôlé avec une manette



III. Architecture ROS 2



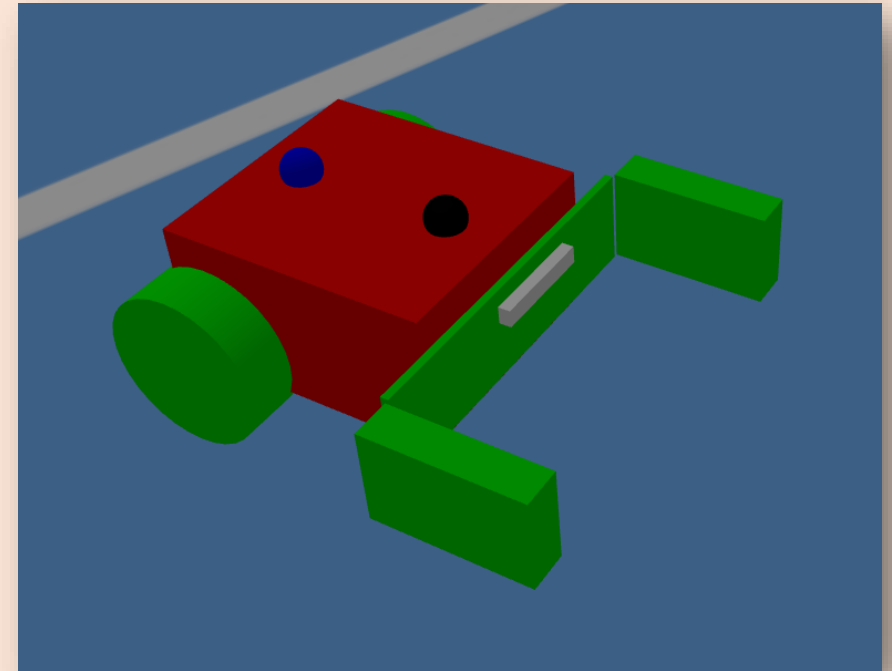
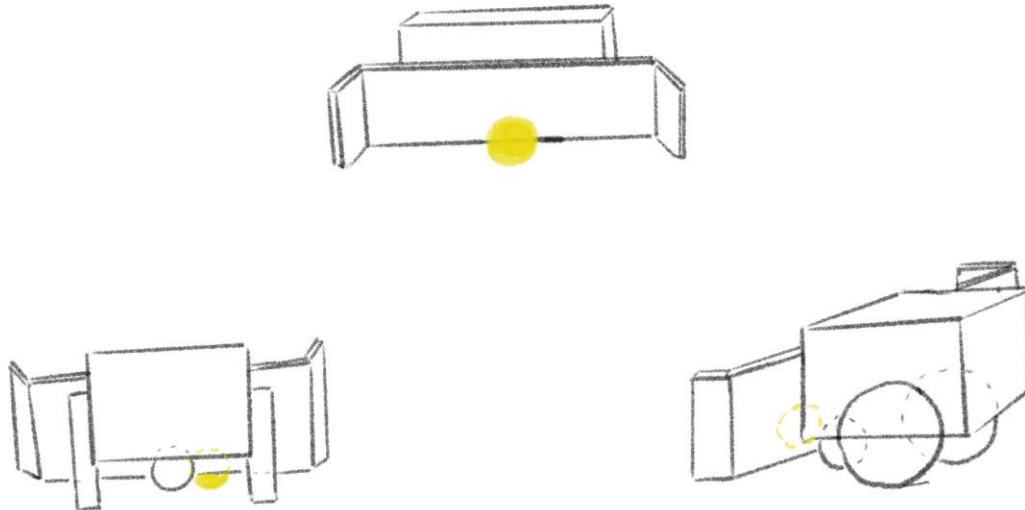
IV. Réalisation techniques

1. Le robot

Dimensions : 50x50x28 cm

Garde au sol : 2,75 cm

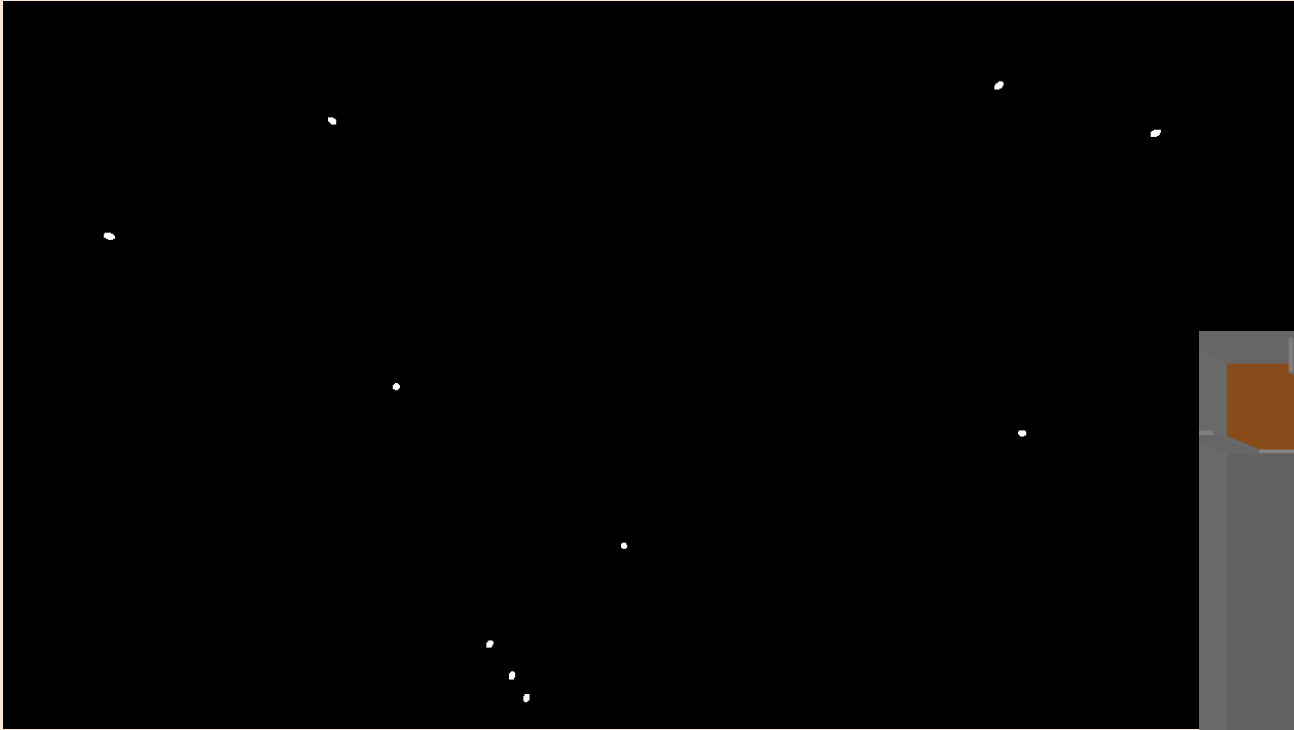
Poids : 2,850 kg





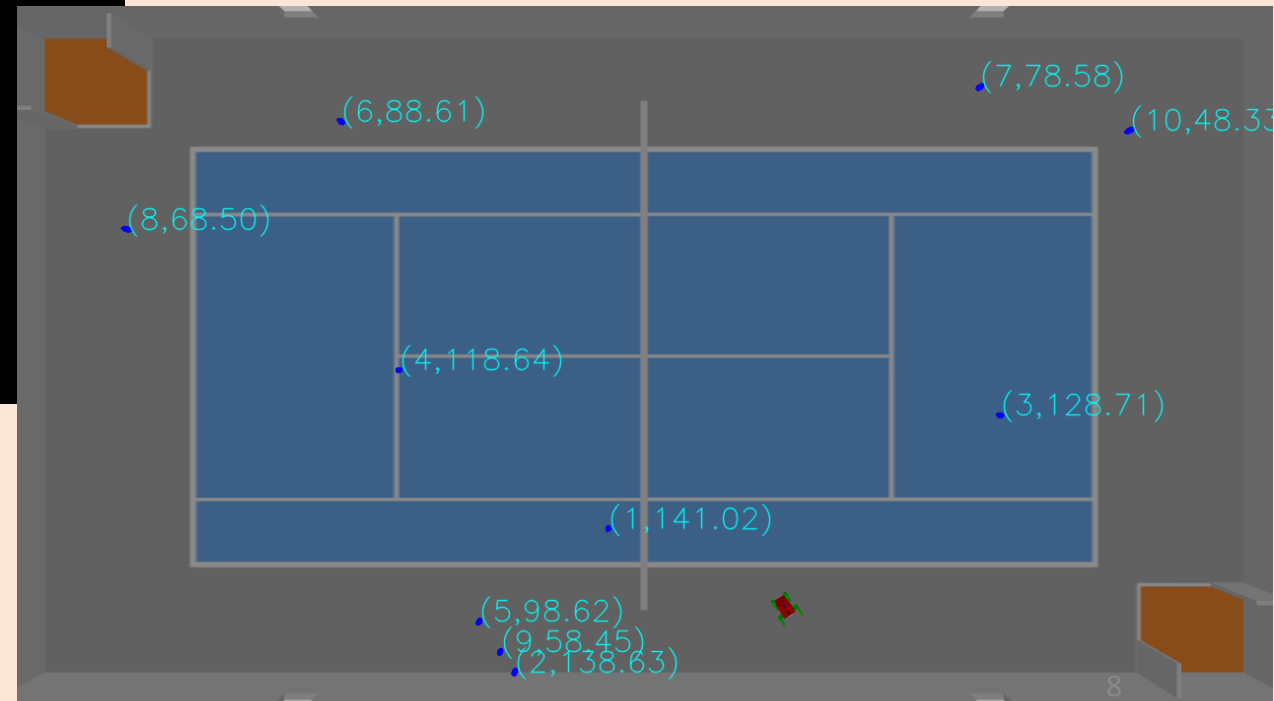
IV. Réalisation techniques

2. Détection des balles



Détection des balles (en blanc)

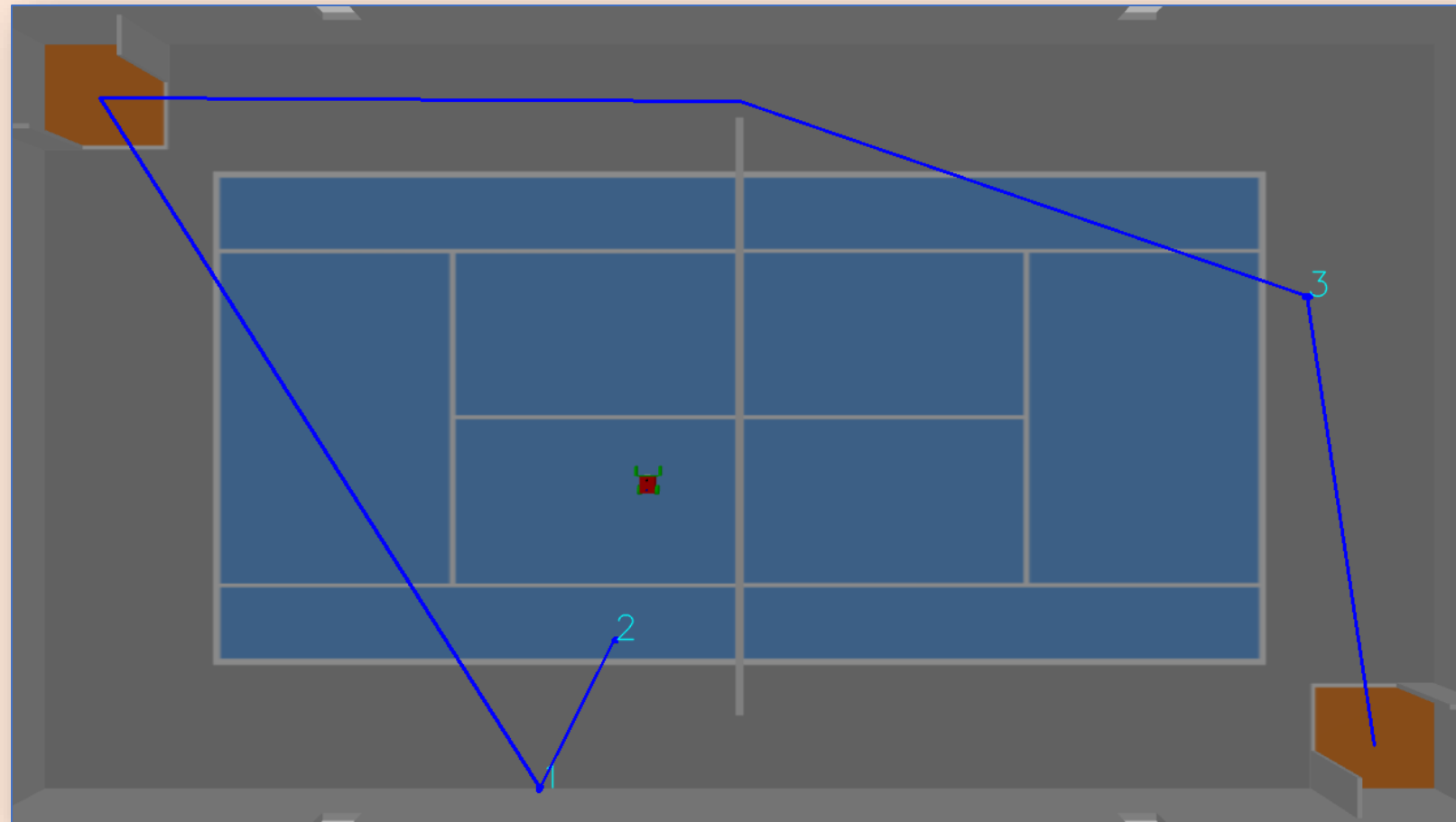
Identifiant et temps écoulé de chaque balle



IV. Réalisation techniques

3. Path planning

- Path planning simple dans l'ordre d'apparition
- Si une balle dans le même terrain apparaît : il la prend sur le chemin
- Dans le cas contraire : le robot évite le filet



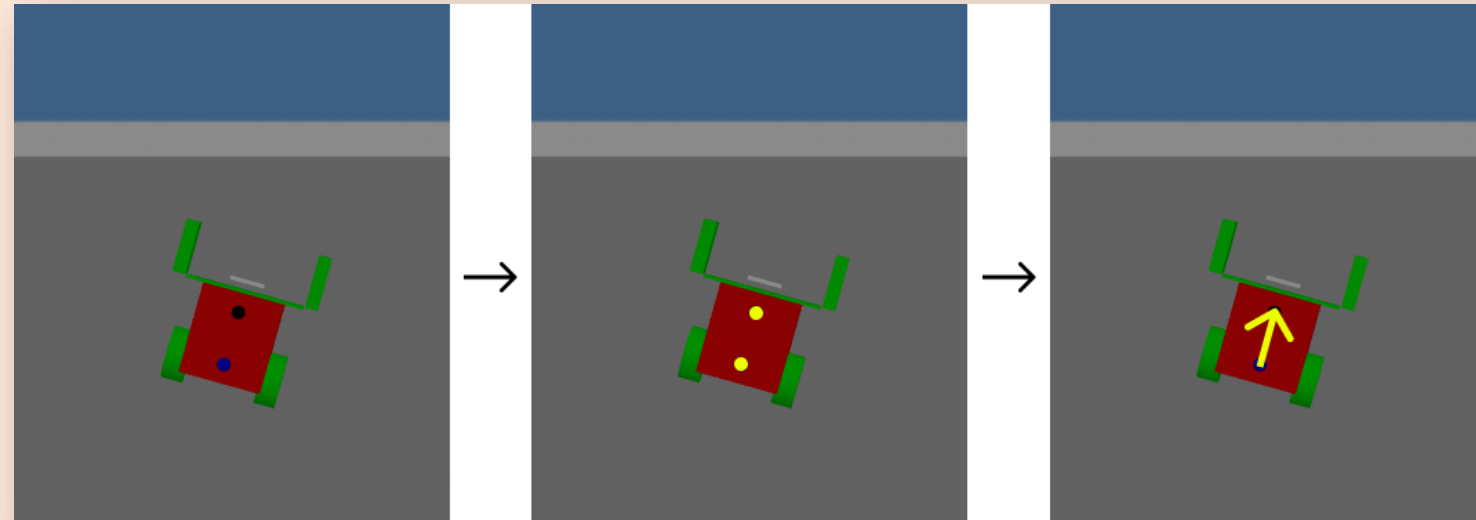
Path Planning

IV. Réalisation techniques

4. Localisation



- Localisation visuel (2 marqueurs */noirs et bleus/* sur le robot)
 - Position
 - Cap
- Contrôle par le contrôleur cinématique



IV. Réalisation techniques

5. Contrôle autonome et contrôle manuel par manette

```

$$u_{max} = 2.0$$

$$\theta_{max} = 0.5$$

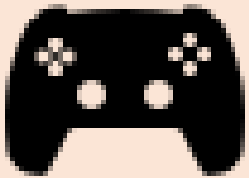
$$\theta_z = \text{sawtooth}(\hat{\theta} - \theta)$$

$$\text{msg.linear.x} = \min(K_p * d, K_p * u_{max} * \text{sign}(d))$$

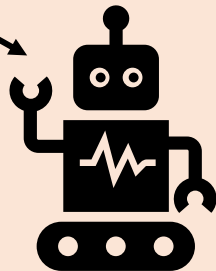
$$\text{msg.angular.z} = \min(\theta_z, \theta_{max} * \text{sign}(\theta_z))$$

```

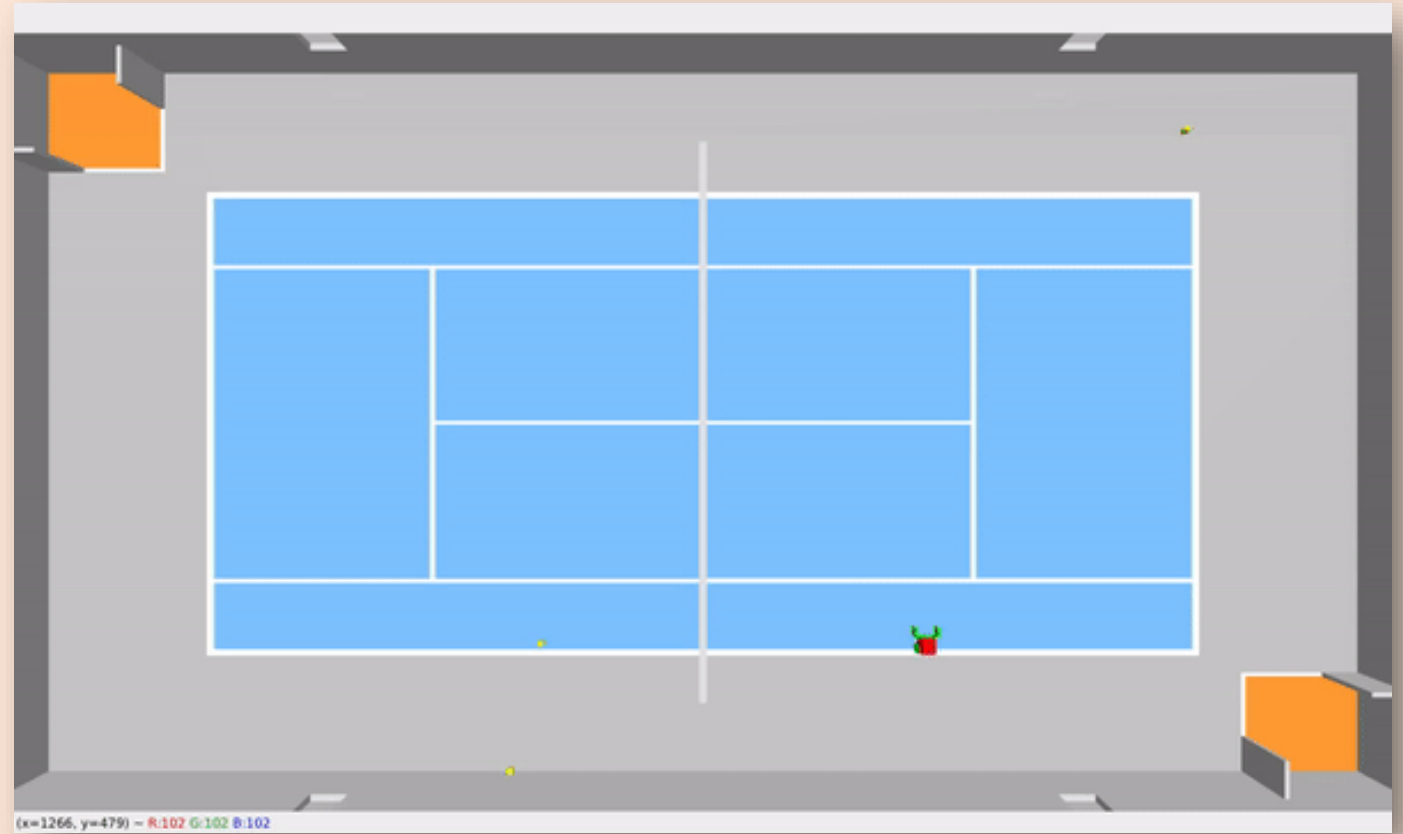
Type de contrôle?



Manette



Autonome

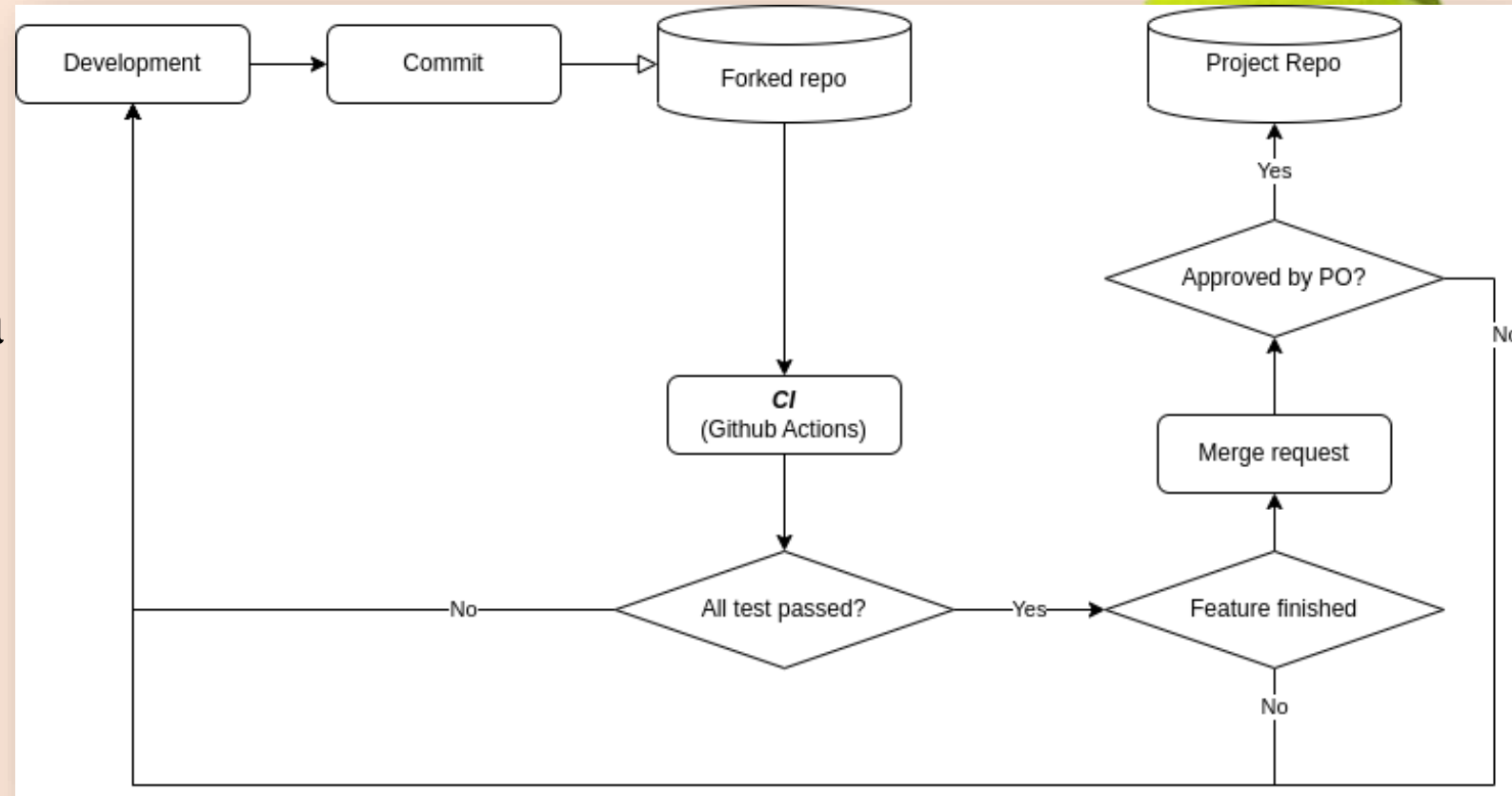


IV. Réalisation techniques

6. Git flow & Continuous Integration (CI)

- CI (Github Actions)
 - Code style PEP 8
 - Vérification de compilation
 - Automatique, badge

 collecte-balle-ros-foxy **passing**





V. RETEX



- Taiga : Pratique mais des mises à jour fréquentes sont nécessaires.
- ROS 2 : Très fluide et intuitif en connaissant ROS1.
- Gazebo : l'Urdf pas adaptable à grande échelle. Lancements multiples avec des résultats différents.
- Github : Outil très puissant. Issues redondantes si utilisation en parallèle de Taiga. Nécessite des pulls fréquents.





VI. Améliorations



- Modèle de robot (forme de la pelle)
- À-coups verticaux lorsque le robot avance
- Optimiser la trajectoire (score, actualisation,...)
- Détection des balles déjà récupérées





Merci de votre attention !

Des questions ?

