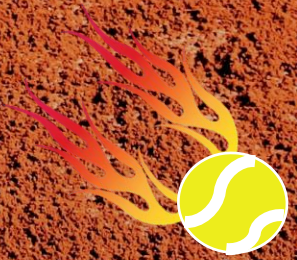
BALLS CATCHERS

INGÉNIERIE SYSTÈME ET MODÉLISATION ROBOTIQUE

Thomas TACHERON
Jonas SOUEIDAN
Rémi POREE
Simon GERVAISE
Augustin MORGE



SOMMAIRE

1) La description technique des solutions choisies

2) La définition initiale des tâches lors de la première séance

3) L'évolution et le suivi d’exécution des tâches pour les séances suivantes

4) L'adaptation au changement de spécifications client

5) Le retour d'expérience (RETEX) sur Gazebo (ce qui a fonctionné facilement et ce qui a posé des difficultés)

6) Le RETEX sur git/github

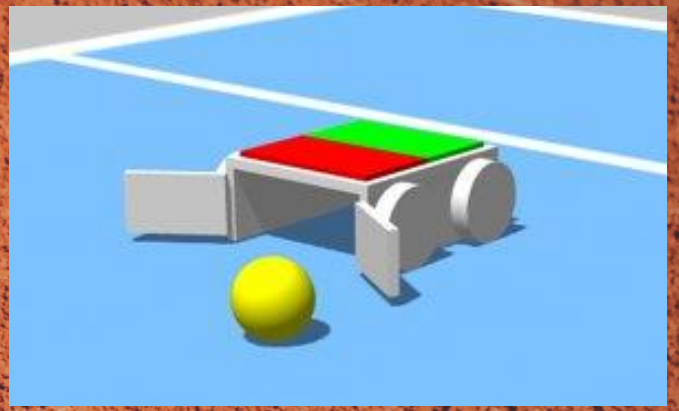
7) Le RETEX sur ROS 2

8) Le RETEX sur l'outil de gestion de projet (Taiga)

9) Recommandations/suggestions pour la prochaine promo

STRATÉGIE CHOISIE

- Robot avec un trou et 4 roues
- Détection des balles par image
- Récupération de plusieurs balles avec la stratégie de la plus proche
- Trou pour contenir plusieurs balles à la fois, et empêcher la détection de balles récupérées



PROCESSUS : DÉFINITION INITIALE DES TACHES

- Créer un modèle sur Gazebo
- Choisir les capteurs
- Choisir les actionneurs
- Définir l'architecture ROS2
- Configurer l'interface ROS2/Gazebo
- Tester le robot

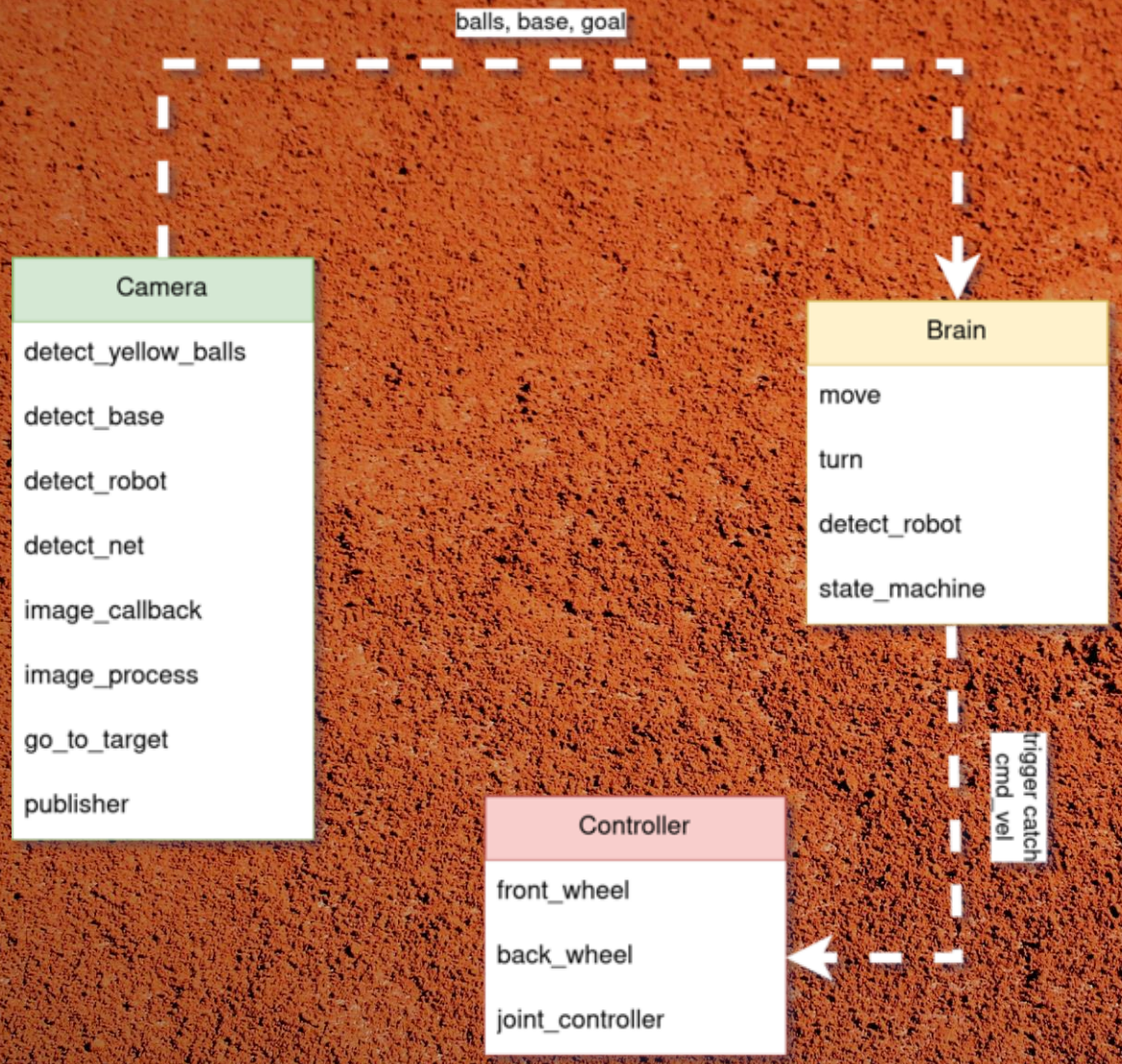
PROCESSUS : EVOLUTION DE LA DÉFINITION DES TACHES

- Avoir un modèle de robot fonctionnel
- Pouvoir détecter les balles sur le terrain et les ramasser
- Gérer les différents états du robot
- Contrôler le robot à la manette
- Améliorer le visuel du robot
- Créer une identité de marque

PROCESSUS : ADAPTATION AU CLIENT

- Visuel du robot : Des flammes
- Normaliser le code et GitHub Actions
- Dimensions du robot limitée
    - Doit pouvoir rentrer dans une boite 50x50x30 cm

RETEX LOGICIELS : ROS2

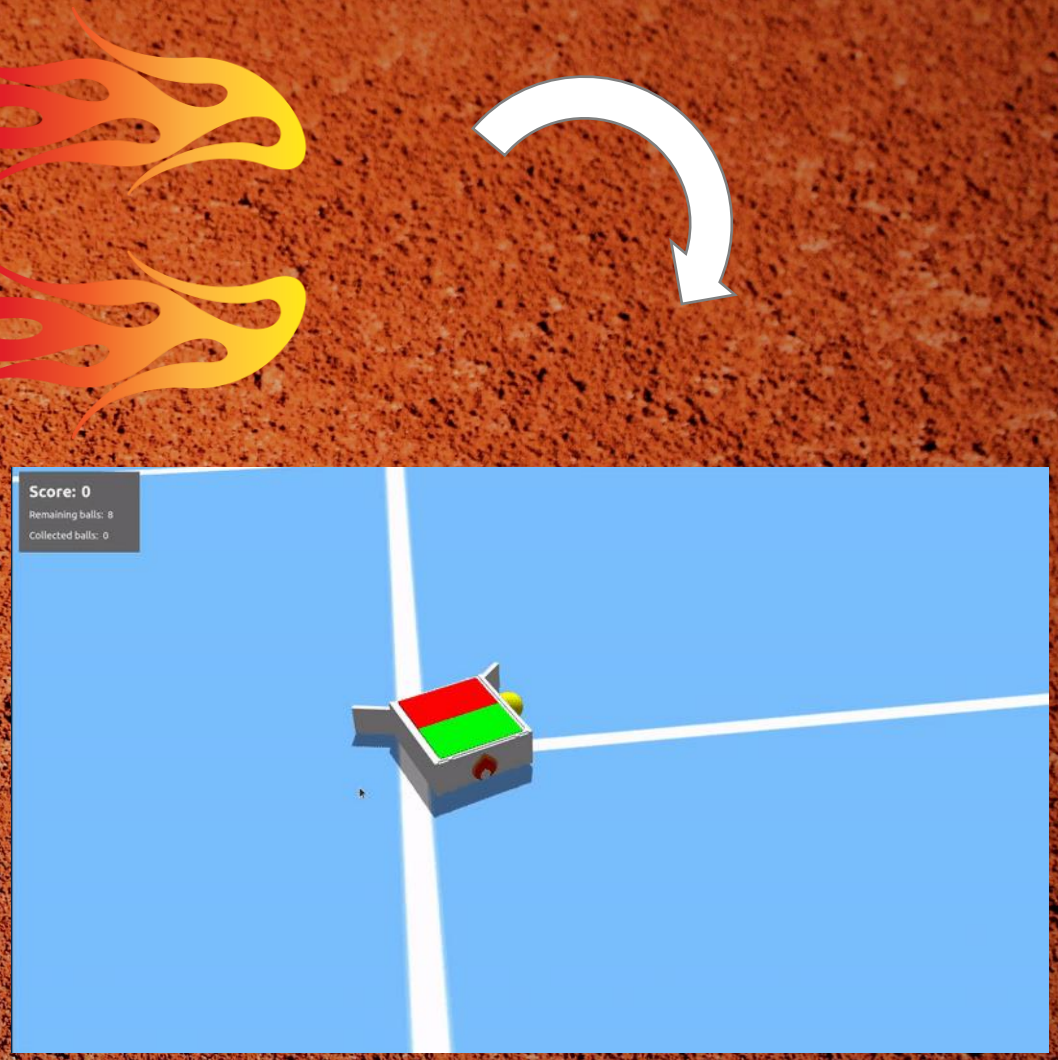


- Traitement de l'image du terrain.
- Trajectoire à suivre.
- Ramassage des balles.
- Les retourner à la base.

RETEX LOGICIELS : GAZEBO

Création du robot de base assez simple et rapide

Ajouts de fonctionnalités difficile (ex : papier peint de flamme sur le robot)



RETEX LOGICIELS : GIT

- Utilisation de fork pour que chacun puisse travailler de son côté
- Merge et pull request fait par le PO pour éviter des suppressions de fichier
- Utilisation de deux dossiers sur le local, un lié au ROS et un lié au git --> évite les accidents de suppression de travail
- GitHub actions pour valider

RETEX LOGICIELS : TAIGA

- Séparation du travail en taches élémentaires

- Répartition des taches dans plusieurs catégories et affectation de chacune des taches à une ou plusieurs personnes

CONCLUSION

- A chaque sprint, un des membres du groupe sera PO
  - Organiser le travail pour que le travail soit optimisé entre les membres, et pour prendre en compte les commentaires du dernier PO.
- Savoir dire non aux demandes excessives des clients
- Ne pas paraitre trop optimiste avec les clients, toujours garder une marge

DEMO