

Projet Robotique

Projet de développement

LU2IN013

Inès BENAMER BELKACEM 21204927

Claude CHIBOUT 21202978

Maëlle LIU 21204734

Thibaut MARCQ 21202966

Jérôme YU 21114103

Semestre 4 - L2



Groupe
Robocar Poli

Sommaire

Objectifs et description du projet	3
Démarrage et Interface 2D	3
Interface 3D et Robot réel	6
Architecture globale de notre projet	9
Description générale	9
Diagramme UML	9
Gestion du travail en groupe	11
Perspectives d'amélioration	11
Conclusion	12
Ce qui nous distingue des autres	12
Avis sur le projet	13
Remerciements	14

Liens

Lien Github

Lien Trello

Lien Canva (slides)

Objectifs et description du projet

Ce projet avait pour but le contrôle d'un robot **Gopigo** mis à notre disposition, équipé d'un Raspberry Pi et d'une carte Arduino. Nous devions implémenter certaines **stratégies** autour du mouvement, de la **gestion de distances**, ainsi que de la **reconnaissance d'image**. Les stratégies principales étaient : **tracer un carré** avec le robot, **avancer jusqu'à un mur** sans se crasher (s'arrêter avant) et **suivre une balise**.

Le projet s'est déroulé en deux grandes parties.

- Démarrage et Interface 2D

Dans un premier temps, nous avons dû simuler un robot dans un environnement à **deux dimensions**. Cette partie nous a permis de comprendre, d'un point de vue **conceptionnel**, le fonctionnement et les interactions du robot avec son environnement, ainsi que l'implémentation de ses fonctions. Nous devions donc simuler : le robot, l'environnement dans lequel il évoluait et les potentiels **obstacles** qu'il allait rencontrer, le tout dans une **interface 2D**.

Nous avons commencé par créer les modules du robot, de l'environnement, du contrôleur et des stratégies. Les trois nous ont permis de former une simulation, représentée par l'interface 2D.

Nous avons choisi d'utiliser **Tkinter** pour cette interface. C'est un module Python très facile à utiliser de part la simplicité de ses fonctions, et du fait qu'il soit directement intégré à Python. Nous l'avons préféré à d'autres alternatives comme **Pygame** qui étaient trop extensives par rapport à notre utilisation. Avec cette interface, nous avons donc pu **simuler** le robot, la gestion des obstacles, et **déployer** nos premières **stratégies**.

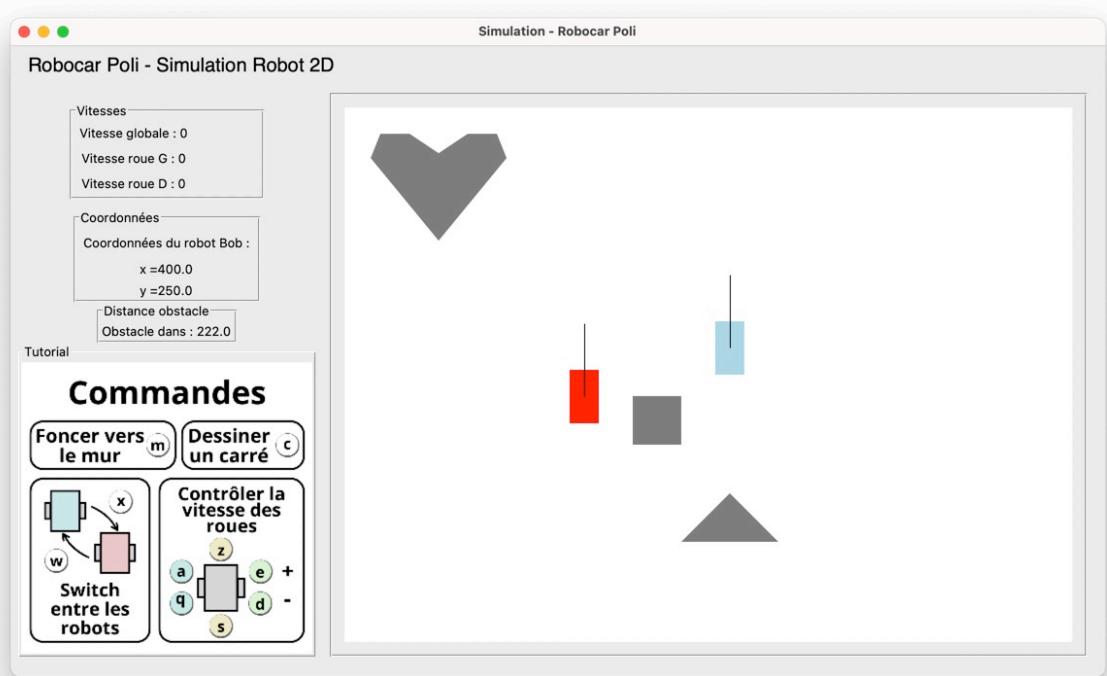


Figure 1: Interface 2D de démonstration

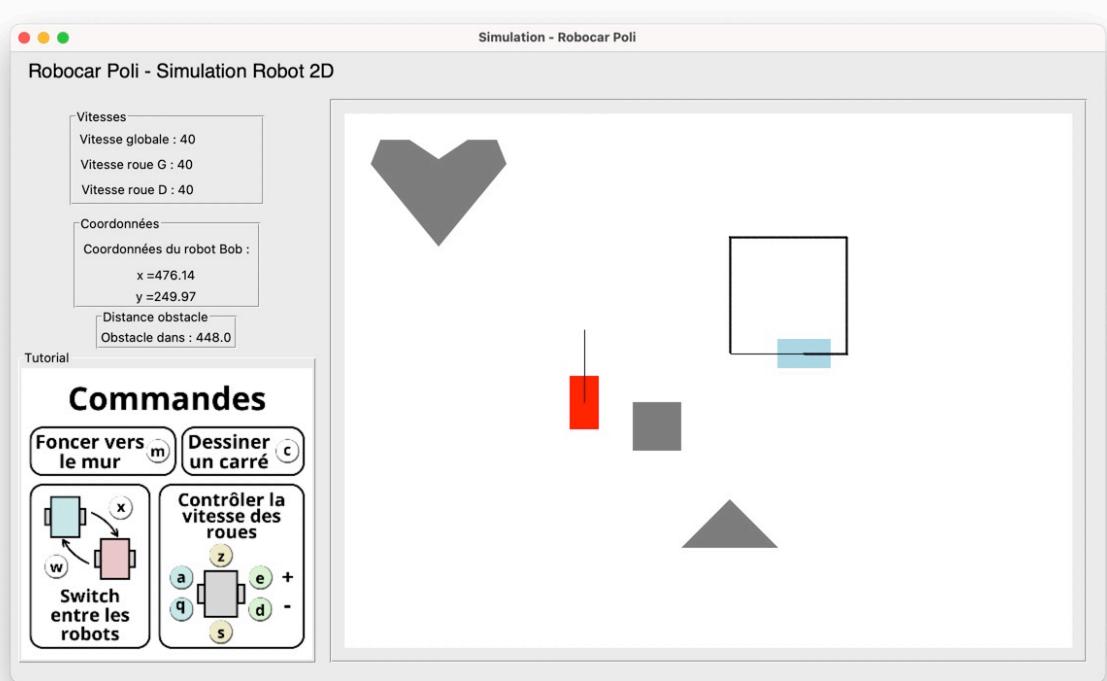


Figure 2: Robot effectuant la stratégie “Tracer carré”

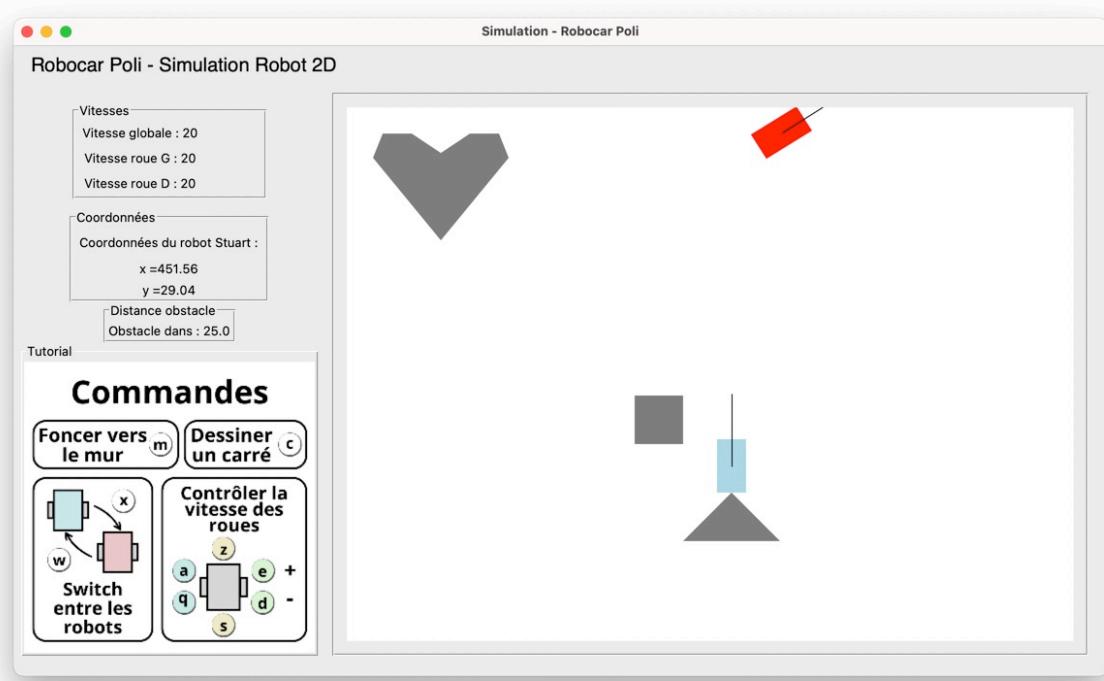


Figure 3: Les deux robots crashés,
Robot bleu dans un obstacle, robot rouge dans la bordure de l'interface

- Interface 3D et Robot réel

Une fois notre interface 2D fonctionnelle et les stratégies correctement implémentées, nous avons pu avoir accès au **robot réel** (Gopigo). Nous avons alors mis en place les modules nécessaires pour faire le lien entre les stratégies développées et le Gopigo. Le robot pouvait alors effectuer les **stratégies** et le **déplacement libre**, comme dans notre interface 2D.

L'étape suivante de notre projet était alors d'implémenter la **stratégie de suivi de la balise**.

Pour ce faire, nous avons dû créer une interface 3D. Notre choix s'est porté sur le module **Panda 3D** pour la réaliser. C'est un module **performant**, utilisé par plusieurs entreprises pour créer des jeux vidéos. Il permet de simuler un monde en 3D, d'y **inclure des objets** et de gérer des vues caméra. Il possède une documentation riche. Nous y avons facilement ajouté les éléments déjà présents dans l'interface 2D et ainsi que plusieurs vues : **vue du dessus** (2D), **vue arrière** et **vue avant** (simule la caméra du robot).

Par la suite, nous avons modélisé et ajouté une **balise** pour pouvoir simuler la stratégie de **suivi de balise**.

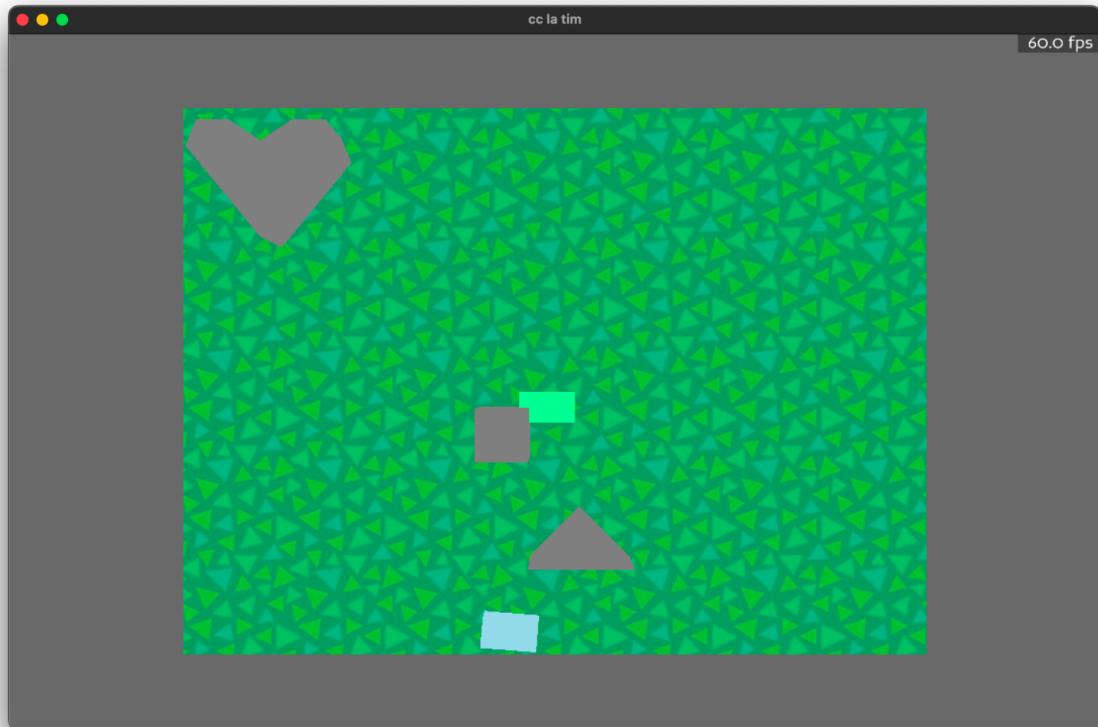


Figure 4: Vue du haut de l'interface 3D, avec les deux robots crash

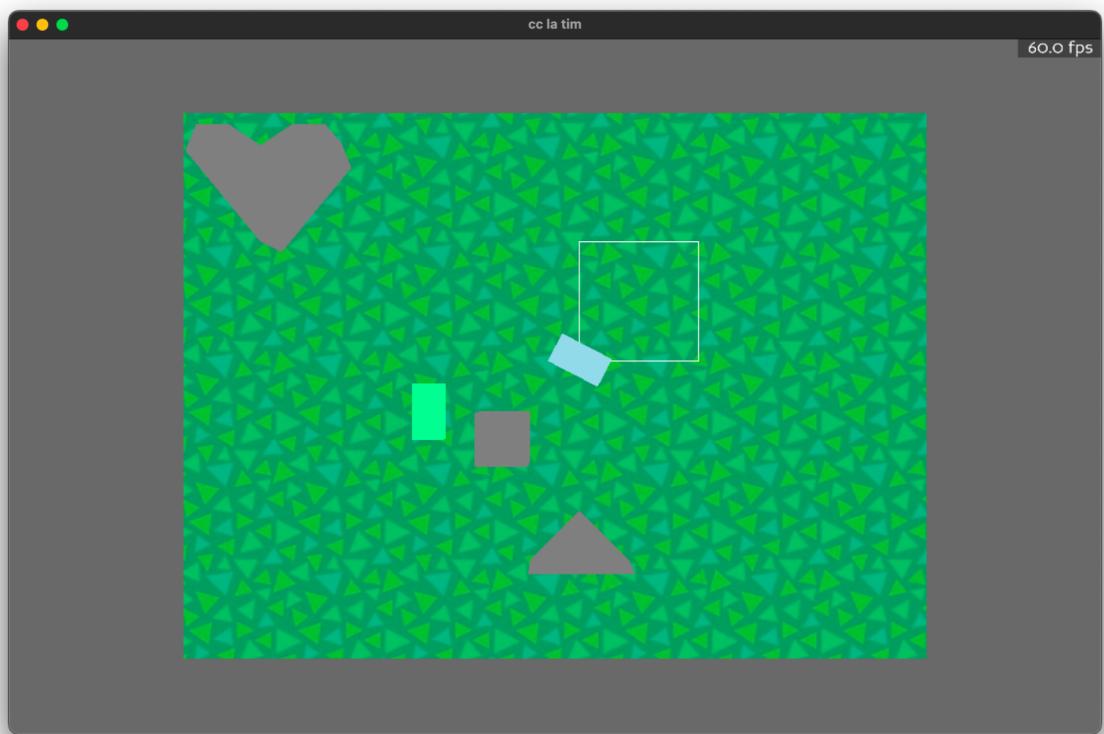


Figure 5: Vue de haut de l'interface 3D, avec "tracer carré" effectué

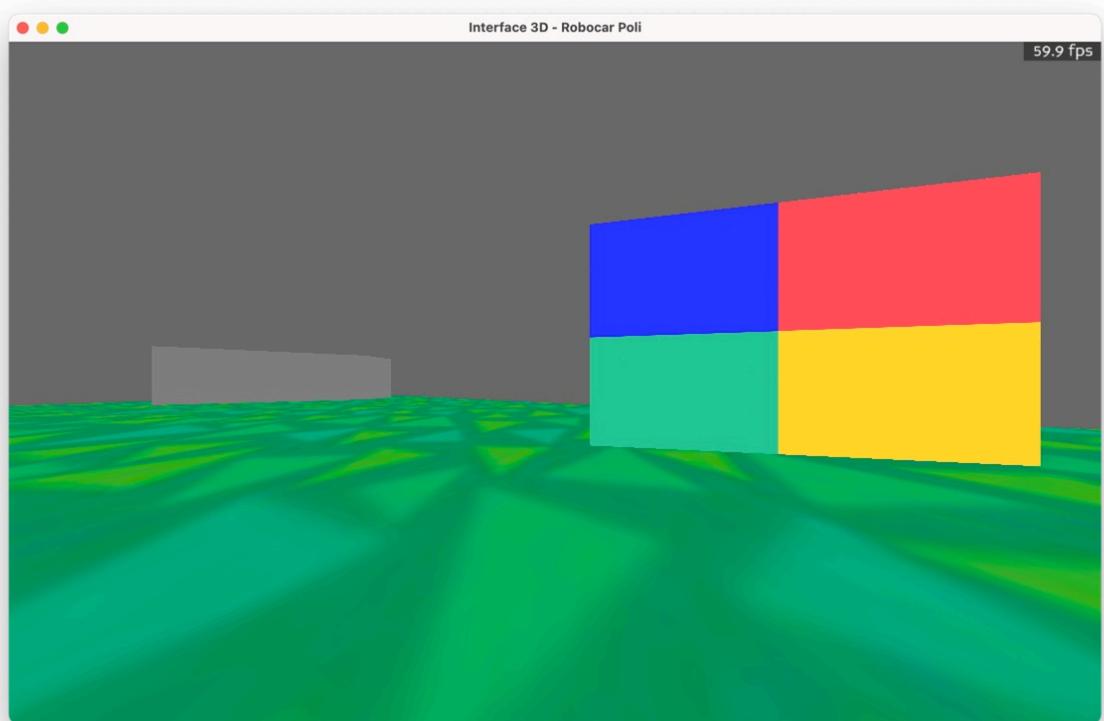


Figure 6: Vue de devant, avec la balise devant le robot

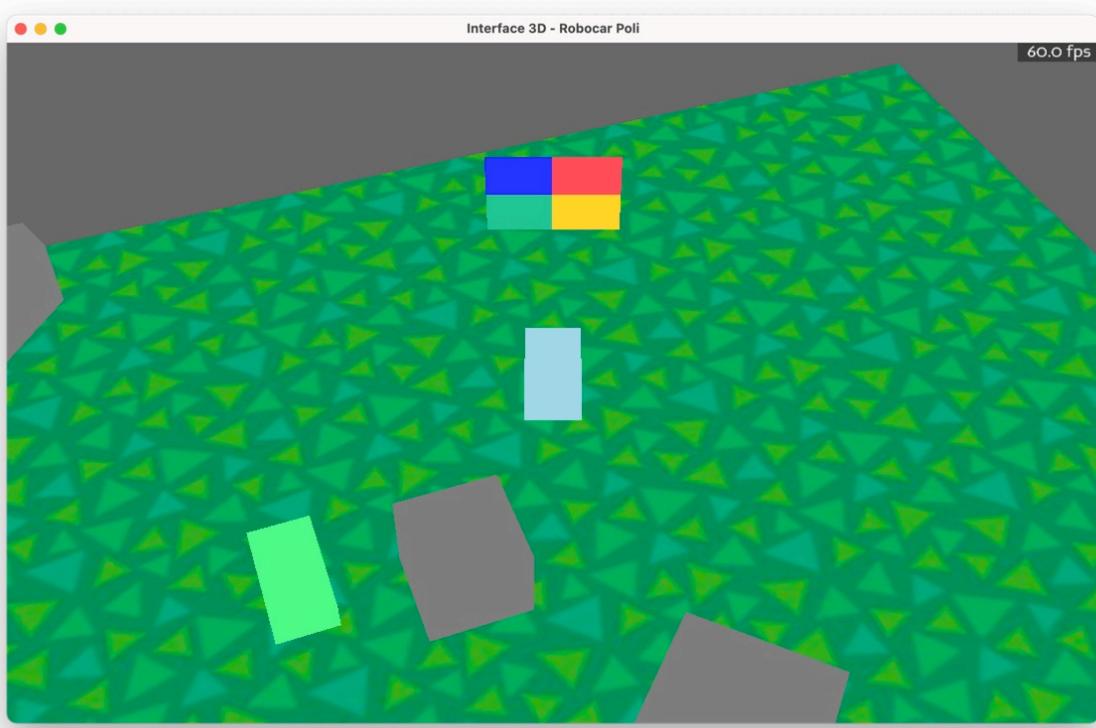


Figure 7: Vue arrière, avec la balise devant le robot

En parallèle, nous testions nos stratégies sur le robot réel. Plusieurs **changements** ont alors dû être faits, par rapport à notre vision du robot simulé. Nous avons par exemple dû modifier notre manière de calculer les **rotations** des roues ou encore gérer les fonctionnalités propres au robot Gopigo comme les **offset des roues**.

Nous nous sommes aussi intéressés à l'utilisation de la caméra du robot ainsi qu'à la **reconnaissance d'une balise**. La première étape était d'essayer de prendre plusieurs photos avec le robot, non sans difficulté. Il a ensuite fallu mettre au point une méthode de reconnaissance de la balise à l'aide du module **CV2**. Il s'agit d'une version adaptée de la bibliothèque **OpenCV**, utilisée pour du traitement d'images. Nous avons utilisé des **masks** pour détecter les quatre couleurs composant la balise. Ces derniers permettent d'obtenir des points (coordonnées) moyens pour chaque couleur trouvée, à partir desquels on peut déduire un point représentant le centre de la balise. Grâce à cette méthode, nous avons pu implémenter une nouvelle stratégie pour **reconnaître et suivre** la balise.

Architecture globale de notre projet

- Description générale

Le projet se trouve dans le répertoire Projet-2IN013. Il est structuré en trois dossiers et quatre fichiers exécutables, en plus des fichiers de test unitaires :

- **autre** comporte les fiches, les comptes-rendus de chaque séance, les différentes photos prises et un diagramme UML du projet.
- **src** est un module dans lequel se trouvent quatre fichiers `constantes.py`, `environnement.py`, `obstacle.py` et `outils.py`. Les fichiers `constantes.py` et `outils.py` nous permettent de définir des valeurs numériques et des fonctions utiles utilisées à travers le projet. `obstacle.py` contient la classe `Obstacle` nous permettant de définir les obstacles utilisés dans nos simulations, alors que `environnement.py` contient la définition de l'environnement simulé, ainsi que toutes les méthodes qui permettent de le manipuler.

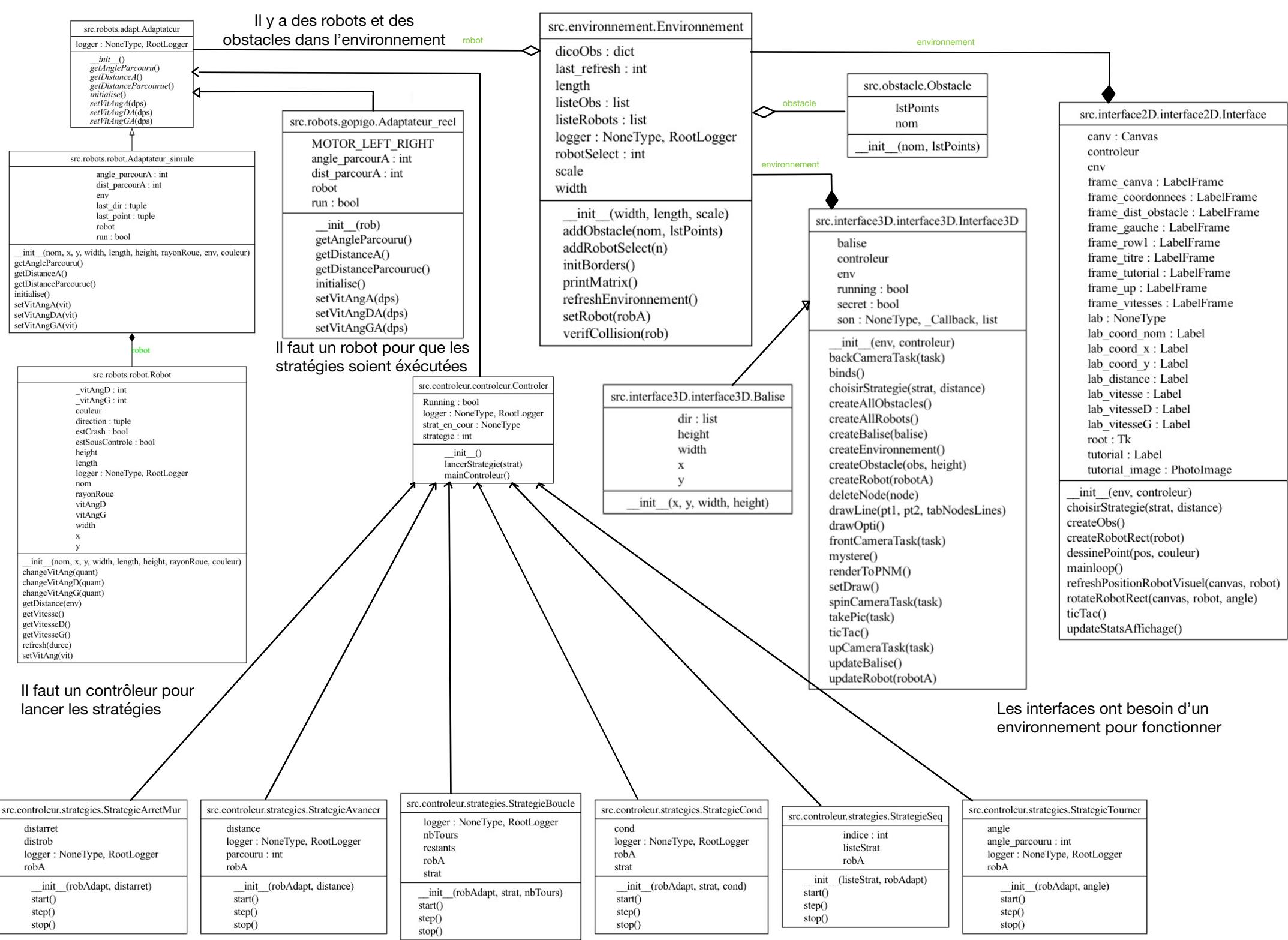
De plus, **src** contient quatre sous modules. Tout d'abord, `interface2D` et `interface3D` définissent les simulations d'interface en 2D et 3D respectivement, ainsi qu'une `démo` qui permet de créer un exemple d'interface générique qui peut être utilisé par l'utilisateur ou d'autres programmes du projet. On trouve dans le sous-module `contrôleur` tout ce qui est relatif à la gestion du contrôleur (qui agit sur le robot en lui donnant des instructions) ainsi qu'à la création des stratégies. Pour finir, le sous-module `robots` permet la définition de notre objet central, le robot (simulé avec `robot.py`, mockup du réel avec `mockupRobot.py` et réel avec `gopigo.py`), ainsi que des adaptateurs de chacun des robots. Les adaptateurs ont le rôle d'interprète entre le contrôleur et des robots de chaque type, en transmettant les ordres du contrôleur. `adapt.py` est la classe abstraite des Adaptateurs, dont héritent les adaptateurs des robots.

- `Test` contient les tests unitaires de chaque fonction du projet pouvant nécessiter un test.
- `main2D.py` permet de démarrer l'interface2D.
- `main3D.py` permet de lancer l'interface3D.
- `mainALL.py` permet d'exécuter les deux interfaces (une à la fois) sans fermer le programme. (la position des robots reste la même d'une interface à l'autre)
- `mainIRL.py` permet au robot réel de faire les différentes stratégies du contrôleur.

Il est nécessaire de noter que, dans chaque module ou sous-module, il y a un fichier `__init__.py` qui facilite les import en récupérant les méthodes, constantes ou instances qui pourraient être nécessaires en dehors du module.

- Diagramme UML

Notre projet dans son ensemble peut être représenté par le diagramme UML suivant. Le diagramme nous permet de voir les méthodes et attributs présents dans chaque module ainsi que les liens entre les modules.



Gestion du travail en groupe

Pour la gestion de notre projet, nous avons utilisé deux outils principaux : **Github** pour la partie code et **Trello** pour la partie organisationnelle. Ils nous ont permis, sur toute la durée du projet, d'avoir une prise en main accompagnée et facilitée.

La méthode **Agile**, avec les **Scrums** à compléter chaque semaine, a donné une certaine consistance dans notre projet, afin de répondre aux exigences de nos **clients** et de leur présenter une démonstration chaque semaine. La fragmentation des objectifs en tâches unitaires nous a offert une meilleure prise en main et répartition du travail, et nous arrivions en général assez bien à nous appropier les différentes tâches. La répartition des tâches se faisait ensuite de manière assez naturelle.

Github a été très utile d'un point de vue pratique pour nous permettre de travailler à plusieurs sur le projet, mais également sur plusieurs éléments différents en parallèle grâce aux **branches**. Nous pouvions ainsi développer différentes parties du projet **séparément** sans dépendre de l'avancée du travail d'une autre personne.

Nous organisions des **réunions hebdomadaires** en milieu de Sprint afin de faire un point sur l'avancée des tâches, mais surtout afin de partager d'éventuels problèmes rencontrés et de réfléchir à des solutions. En plus de ces réunions, nous avions également des séances de **brainstorming** avec tous ou bien une partie des membres du groupe selon la complexité de la fonctionnalité dont il était question.

Perspectives d'amélioration

Avec le recul, il y a quelques points sur lesquels nous aurions pu nous améliorer pour ce projet. En effet, nous n'avons pas utilisé toutes les fonctionnalités dont le robot dispose. C'est dommage car il y a de nombreuses options qui auraient permis de réaliser des stratégies **innovantes** et **intéressantes**, comme l'utilisation du **servo** (tête rotative).

Un autre aspect sur lequel nous aurions pu être plus performants est la **consistance** et la répartition du travail. Vers la fin du projet, le travail était plus déséquilibré entre les membres de l'équipe. Nous étions moins réguliers et nous aurions pu avoir une meilleure distribution des tâches entre co-équipiers (les examens des autres UE ont également eu un impact sur la quantité de travail fournie sur le projet).

Enfin, nous n'avons pas trouvé de moyen assez rapide et efficace pour **récupérer une image** de la vue avant du robot dans Panda3D. Sans ce problème, notre stratégie aurait été beaucoup plus performante dans la simulation. Il est à noter que cette stratégie fonctionne **parfaitement** sur le robot réel.

Conclusion

- Ce qui nous distingue des autres

De nombreux points de notre projet le rendent **original** et le distinguent des autres, tant au niveau de l'implémentation que de l'utilisation.

Tout d'abord, en ce qui concerne la partie simulée, nous utilisons les **threads**, ce qui nous permet d'avoir des robots qui fonctionnent **indépendamment** des simulations. Ils continuent donc à être en activité même si on **ferme une interface** (pour arrêter le robot il faut arrêter le programme). Notre interface 3D nous permet d'avoir **3** points de vue différents, un point de vue omniscient (équivalent au point de vue de la simulation 2D), un point de vue de l'arrière du robot, et un point de vue première personne, simulant la caméra à l'avant du robot réel.

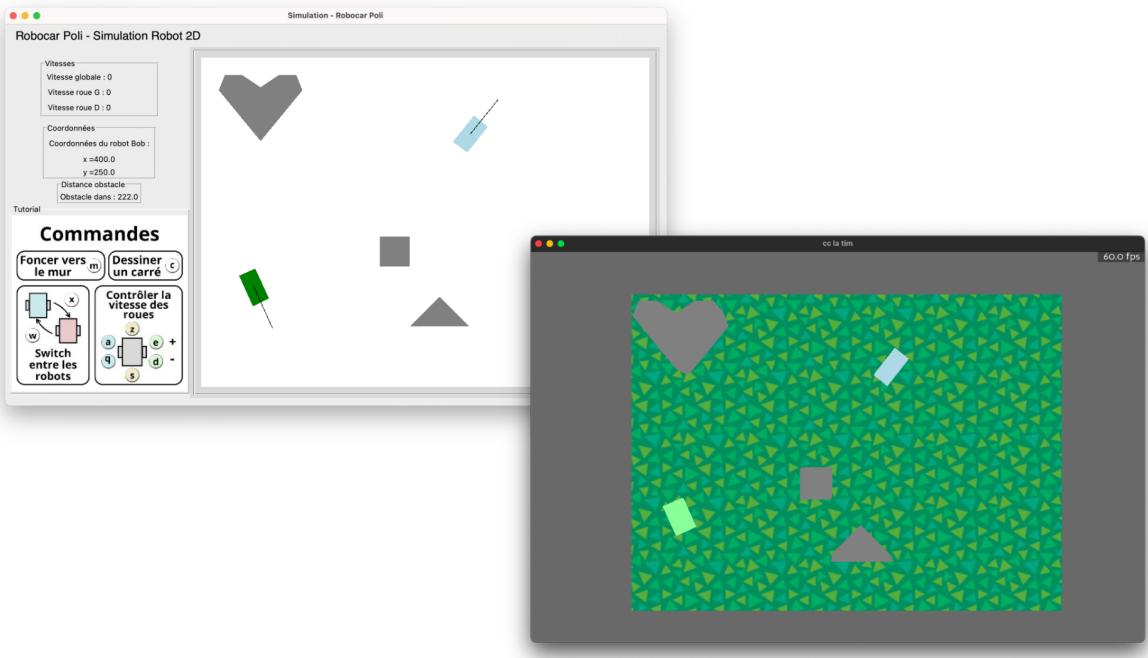


Figure 8: Interfaces 2D et 3D ouvertes pour le même environnement

Nous pouvons également ajouter des obstacles de forme **personnalisée** dans les simulations. Cela donne au client une représentation plus fidèle de la réalité et plus de flexibilité. De plus, il est possible de contrôler les mouvements du robot en dehors des stratégies définies, en modulant les vitesses des roues individuellement avec les touches du clavier.

D'un point de vue plus pratique, nous avons également résolu des problèmes avec le robot. Nous avons **remonté** le robot lorsqu'il était cassé, et nous avons retourné la caméra d'un des robots qui se tenait **à l'envers**. De plus, nous avons remarqué de petites inconsistances de l'API qui posaient des problèmes lors de l'acquisition d'images, et qui pouvaient **bloquer d'autres tentatives** d'utilisation de la caméra et d'instanciation du robot. Une fois qu'un robot était instancié pour la première fois, la caméra n'était plus utilisable avec une instance différente. Cela nous forçait à redémarrer le robot à chaque fois.

Pour finir, une autre caractéristique qui nous démarque est la stratégie supplémentaire que nous avons conçue. En effet, elle montre notre compréhension des axes du projet et nous permet d'aller au-delà de ses objectifs, en utilisant différentes fonctionnalités à la fois : mouvement, son et lumière, le tout en respectant le thème Robocar Poli de notre groupe : la boucle est bouclée !

- Avis sur le projet

En rétrospective, ce projet a pu être très frustrant par moments, mais il nous a apporté de l'**expérience** et des **compétences** précieuses.

Il nous a appris à travailler en équipe sur notre premier projet de grande ampleur, en terme de temps, d'objectifs, ou de travail fourni. Nous avons été introduits aux méthodes de travail Agile et Scrum. Ces méthodes étaient pour nous une première, qui présentent l'avantage d'une meilleure gestion de projet lorsque le nombre de tâches est important. Elles sont aussi très utiles quand le projet comporte plusieurs membres dans l'équipe, comme ici. Ces outils nous ont guidés dans notre manière de nous organiser et nous ont permis de maintenir un rythme de travail **stable**, semaine après semaine, afin de répondre aux attentes de nos **professeurs**.

Nous avons aussi été confrontés à la gestion de problèmes en tous genres. Nous avons dû chercher des **solutions** à ces problèmes en trouvant des personnes qui ont fait face à des situations similaires (sur des forums, par exemple), à nos **membres de groupe**, voire à d'autres **camarades** de classe.

Le projet était intéressant dans la mesure où nous avons pu voir une **application concrète** de notre travail avec un robot **physique** qui répondait à nos commandes. Cela le distingue des autres travaux que nous avons pu faire jusqu'alors dans d'autres UE. Leur influence était limitée à nos ordinateurs. Nous avons pu nous **amuser** à voir nos programmes avoir une incidence sur le monde physique. De plus, voir notre projet **se construire** et avancer chaque semaine a été satisfaisant pour nous. Savoir que nous partions d'un dossier vide nous rend d'autant plus **fiers**.

Remerciements

Enfin, nous souhaitons remercier nos deux professeurs **M. Baskiotis** et **M. Sigaud**.

Merci pour vos conseils précieux chaque semaine. Les rendez-vous avec vous ont au départ été exigeants et parfois "démoralisants", mais vous avez tous les deux su nous donner de la rigueur, de la **motivation** et une vision claire de notre projet. Grâce à votre encadrement et votre **soutien**, nous avons pu surmonter les obstacles et **progresser** de manière significative. Votre expertise et votre patience ont été **inestimables**, et nous vous en sommes **profondément reconnaissants**.

Nous espérons avoir su vous montrer nos compétences et vous convaincre, en tant que client, de choisir notre entreprise pour la réalisation de votre projet !