Nous utiliserons le simulateur Roborobo 4.0, dont le coeur est codé en C++ et qui est entièrement utilisable en Python (librairie pyRoborobo).

Pour l'installer:

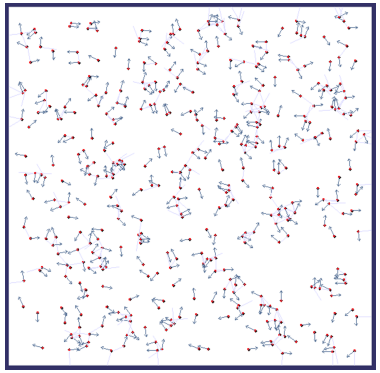
1. suivez les instructions données sur https://github.com/nekonaute/roborobo4
2. récupérez l'archive du TME sur Moodle et décompressez la où vous voulez
3. pour tester, exécutez la commande "python tutorial.py"

L'archive comprend un programme qui montre comment programmer les robots. Tous les éléments nécessaires pour vous aider à répondre aux questions se trouvent dans ce programme exemple, que vous dupliquerez et renommerez pour chaque question.

Ce programme utilise un fichier de configuration (config/simple.properties), qu'il est conseillé de dupliquer pour chaque question. Il n'est pas utile de le modifier, à part éventuellement pour changer le nombre de robots avec la valeur de la variable *gInitialNumberOfRobots*.

Remarques: le simulateur accepte des commandes pendant l'exécution:
- « h » pour afficher l'aide dans le terminal
- « p » pour faire une pause
- « j » pour changer l'affichage des senseurs (invisible, visible, visible si contact)
- « k » pour changer l'affichage de la direction (vecteur apparent ou non)
- « d » pour accélérer la simulation (3 modes).
- « f » pour activer le suivi visuel d'un agent en particulier.
- <tab> pour passer d'un agent à l'autre.
- <entrée> pour prendre le contrôle d'un agent.

Astuce: la prochaine fois, n'oubliez pas de réactiver l'environnement avec la commande "conda activate roborobo" avant d'exécuter votre programme.



———

M2 ANDROIDE, UE IA et Robotique nicolas.bredeche(at)sorbonne-universite.fr
TME ROBOTIQUE COLLECTIVE rev. 2022-10-14
Partie 1

EXERCICE 1: *évitement (prise en main et rappel)*

Fichier: avoider.py (à créer)

Etudiez le comportement d'évitement d'obstacles de l'exemple. Modifiez le pour que la réponse à un obstacle soit proportionnelle à la proximité, en utilisant plusieurs senseurs.

Evaluation: implémentation d'une réponse comportementale proportionnelle.

———

EXERCICE 2 : *Comportement d'agrégation simple*

Fichier: agregation.py (à créer)

Un essaim de robot s'agrège lorsque, chaque robot partant d'une position initiale au hasard, l'ensemble des robots se regroupent jusqu'à ne former qu'un seul agrégat. Version plus simple: plusieurs agrégats se forment.

Ecrivez un comportement d'agrégation simple. Chaque robot doit se diriger vers le robot le plus proche. Vous pouvez changer le nombre de robots (*gInitialNumberOfRobots*) en éditant votre fichier *properties* (qu'il est aussi conseillé de dupliquer pour cette question).

Evaluation: avec la touche « entrée » prenez le contrôle d'un robot et déplacez vous afin de perturber le comportement d'agrégation. Si votre robot est bloqué, vous pouvez changer de robot avec la touche tabulation (shift+tab pour revenir au précédent).

———

EXERCICE 3 : *Comportement de dispersion*

Fichier: dispersion.py (à créer)

Un essaim de robots se disperse lorsque chaque robot se positionne afin d'avoir un nombre minimum de voisins, chaque voisin étant le plus éloigné possible. A noter qu'il est pas toujours possible d'obtenir une dispersion parfaite (ex. environnement trop petit).

Ecrivez un comportement de dispersion. Chaque robot doit s'éloigner du robot le plus proche. Lorsqu'il ne voit rien, il tourne sur lui même. Ajustez le nombre de robots au départ (*gInitialNumberOfRobots*) afin qu'il existe un état stable ou les robots ne se déplacent plus, avec un maximum de robots.

Evaluation: avec la touche « entrée » vous pouvez prendre le contrôle d'un robot. Déplacer vous afin de perturber le comportement de dispersion.

———

EXERCICE 4 : *Comportement d'agrégation robuste*

Reprenez l'exercice 2 pour que votre essaim ne forme qu'un seul agrégat.
Astuce: la probabilité de rester dans un agrégat peut dépendre de la présence de voisins (nombre et distance).