Manuel d'utilisation de l'environnement VacuumSG

I Aperçu rapide de la version de base

L'interface de la version par défaut se compose d'un ensemble de trois fenêtre :
- la fenêtre principale, affichant l'environnement,
- l'interface utilisateur, permettant à un utilisateur extérieur de contrôler le déroulement de la simulation,
- un affichage des informations concernant l'agent. Il faut noter que d'autres affichages peuvent être ajoutés (voir sections II-1 et II-5).

La fenêtre principale permet d'observer la simulation en cours et d'éditer l'environnement des agents. L'environnement est défini comme une matrice dont chaque cellule peut contenir un élément (3 éléments sont fournis dans la version de base). Les contrôles sont les suivants :
- clic gauche sur un agent pour que les afficheurs affichent les caractéristiques de cet agent. Le numéro de l'agent est affiché en haut à droite, devant le compteur de cycle de décision (non utilisé dans la version de base).
- clic gauche permet d'ajouter/retirer un bloc mur (vert)
- clic molette permet d'ajouter/retirer une proie (bleu)
- shift+clic droit permet d'ajouter/retirer une algue (rouge)

L'interface utilisateur permet de contrôler l'exécution de la simulation :
- *save* permet de sauvegarder des informations sur les agents (non utilisés dans la version de base, voir section II-1).
- *Play* agent permet de jouer la simulation de l'agent sélectionné,
- *Paus*e agent permet de mettre en pause la simulation de l'agent sélectionné
- *Reset* met à zéro la simulation
- *Clear* permet d'effacer les traces des agents
- *Play* permet de jouer la simulation de tous les agents
- *Pause* met en pause tous les agents.

L'afficheur proposé par la version de base affiche le point de vue de l'agent sélectionné. Son fonctionnement repose sur un moteur de rendu polaire rudimentaire (voir section II-5).

II Implémentation

L'implémentation de l'environnement est divisée en 5 packages :
- *platform* regroupe toutes les classes liées à la gestion, le contrôle et l'observation de la simulation.
- *environment* regroupe les classes associées à la physique de l'environnement et à l'affichage de la simulation.
- *decision* regroupe les systèmes décisionnels des agents.
- *sensorySystem* permet d'implémenter des modèles de capteurs pour les agents.
- *display* regroupe les systèmes d'affichage permettant d'observer les agents.

II-1 platform

Main : classe principale, elle joue la simulation. Il est possible de modifier la vitesse de la simulation en modifiant le délai d'attente (paramètre de la fonction *sleep*).

Agent : forme une plaque tournante entre les différents composants d'un agent. Cette classe contrôle le cycle d 'énaction de l'agent :
- prise de décision
- création de la commande motrice
- envoie de la commande motrice

Il est possible de modifier les temporisations du cycle d'énation ainsi que la longueur de la trace affichée à l'écran (paramètre *pathLength*)
L'agent peut être contraint à la matrice qui compose l'environnement. Si cela est nécessaire, il faut utiliser la procédure *center()* en fin de cycle.

Des liens vers des afficheurs peuvent être ajouté si cela est nécessaire (voir l'exemple donné : *ProbeDisplay*)
Si des informations doivent être sauvegardées, il est prévu une procédure *save()* utilisée lorsque l'utilisateur clique sur le bouton « save » de l'interface.

UserInterfaceFrame : affiche la fenêtre d'interface. Des boutons supplémentaires peuvent être ajoutés.

PrintableFrame et ***PrintablePanel*** : classes génériques d'afficheurs pouvant exporter des afficheurs au format JPEG et PDF. L'export au format JPG incrémente automatiquement le nom du fichier exporté.

II-2 environment

Block : définit les propriétés visuelles et tactiles d'un élément de l'environnement.

Environment : génère le contenu de l'environnement à partir d'un fichier texte (variable *DEFAULT_BOARD*). L'environnement par défaut propose quatre types d'éléments. D'autres peuvent être ajoutés sur le même modèle. Le fichier texte permet de définir la structure de l'environnement. Dans la version par défaut, les symboles "v", "^", "<", et ">" permettent de placer un agent, avec l'orientation voulue. "*" permet de placer une proie, "w" permet de placer un mur, "a" permet de placer une algue et "-" désigne un espace vide (voir le fichier *Board10x10.txt* en exemple).
L'interface utilisateur peut être modifié dans la procédure *drawGrid(int c)*.

EnvironmentFrame et ***EnvironmentPanel*** : gérent l'affichage de l'environnement. Il est possible de modifier les figures représentant les éléments de l'environnement.

Robot : sert de "corps" à l'agent. Il interprète les commandes motrices (*double[] act*) pour agir dans l'environnement, en gérant les collisions et l'interaction avec certains éléments. La version de base embarque un ensemble de capteurs et d'une "couronne" de 8 leds autour de l'agent (vecteur *display[]*) destinée à l'affichage.
La commande motrice proposée est constituée d'un vecteur de 8 éléments. Les deux premiers donnent un déplacement respectivement sur l'axe x (*act[0]*) et y (*act[1]*), le troisième une rotation d'axe z (*act[2]*). Les trois éléments suivants permettent d'utiliser des capteurs tactiles devant (*act[3]==1*), à gauche (*act[4]==1*), à droite (*act[5]==1*) et derrière l'agent (*act[6]==1*). Les autres éléments ne sont pas utilisés.

II-3 decision

Decision : classe générique de système de décision. La fonction `decision(float[] enacted)` retourne une commande motrice à destination du *robot*. *enacted* est le vecteur comportant les valeurs des capteurs du *robot*. C'est dans cette fonction que doit être implémenté le système décisionnel d'un agent.
Deux exemples de systèmes décisionnels sont fournis :

Braitenberg : implémente le comportement d'un véhicule de Valentino Braitenberg. Les proies servent ici de sources lumineuses. Deux capteurs de luminosité permettent de définir la commande motrice.

WallAvoider : uimplémente un comportement permettant de suivre les murs : l'agent avance en tournant à gauche, et en tournant à droite lorsqu'un mur est détecté.

II-4 sensorySystem

Probe : un moteur de rendu polaire rudimentaire permettant d'obtenir "l'image" vue du point de vue de l'agent, en obtenant la couleur et la distance des éléments détectables par l'agent avec une précision de 1°. La position par rapport au centre de l'agent peut être spécifié (*float posX*, *float posY*). Ce dispositif permet de simuler une grande variété de capteurs pouvant être placés en n'importe quel point par rapport à l'agent.

Sensor : classe générique d'un capteur retournant une valeur réelle en ce basant sur le résultat de *probe*. La version par défaut fournit un exemple d'un tel capteur :

LightSensor : capteur qui simule un capteur de luminosité, en considérant les proies comme des sources lumineuses. La valeur retournée dépend de la distance de la proie et de son orientation par rapport à l'agent.

II-5 display

EnvFrame et **EnvPanel** : classe générique d'afficheur permettant l'affichage de propriétés relatives à l'agent. La version de base fournit un exemple d'afficheur :

ProbeFrame et **ProbeDisplay** : afficheur qui montre le rendu du module probe de l'agent sélectionné. Cet afficheur montre un champs de vision de 360°.