Université Pierre et Marie Curie

Manuel de l'utilisateur

PIAD de Master1 d'Informatique en Intelligence Artificielle et Décision

Semi-supervised Learning Agents

Auteurs :
Lan Zhou
Matthieu Zimmer

Superviseurs :
Paolo Viappiani
Paul Weng

9 mai 2013

Version 1.1

UPMC
SORBONNE UNIVERSITÉS

Table des matières

De part la nature du projet : bibliothèque + interfacement avec le simulateur, l'installation est un peu technique. De plus, le fonctionnement sans modification de code est assez limité, il faudra mieux se référer au Manuel du Programmeur pour tirer parti de la bibliothèque.

1 Procédure d'installation

1.1 Configuration Requise

La procédure d'installation doit se dérouler sur un environnement GNU/Linux. Le projet est pleinement fonctionnel sur les distributions Archlinux et Ubuntu.

Il vous faudra disposer des drivers graphiques "hardware" de votre carte et répondre aux spécifications matérielles suivantes :

- 1GHz CPU
- 512MB RAM
- OpenGL 1.3 compatible graphics card with 64 MB RAM
- 2 Go d'espace libre (si compilation)
- 1 Go d'espace libre (si installation)

Au vu de la taille du simulateur TORCS [TORCS Team, 2001] présenté dans le cahier des charges [Zhou and Zimmer, 2013a], il n'a été fourni en même temps que le reste du projet. Vous avez donc 2 méthodes pour le récupérer :

1.2 Installer le simulateur : Via la distribution (recommandé sur Ubuntu)

Si vous êtes sous Ubuntu, TORCS semble avoir quelques problèmes à se lancer après une compilation à la main, nous recommandons donc de l'installer directement :

```
sudo apt-get install torcs
```

Vous pouvez alors passer à l'étape 1.4 directement.

1.3 Installer le simulateur : Via compilation

Si toutefois, vous souhaitez vous lancer dans la compilation de torcs, voici la marche à suivre : (fonctionne sur Archlinux)

1.3.1 Liste des paquets pré-requis

Voici la liste des paquets que vous aurez besoin afin de compiler le simulateur :

- bash
- gcc (avec g++)
- mesa
- freeglut
- openAL (avec alut)
- vorbis
- libxi
- Xmu

- Xrender
- Xrandr
- libz
- libpng

Sur un live CD Ubuntu, la commande suivante est suffisante (en activant tout les dépôts) :

```
sudo apt-get install g++ mesa-common-dev freeglut3-dev libplib-dev
    libopenal-dev libalut-dev libvorbis-dev libxi-dev libxmu-dev
    libxrender-dev libxrandr-dev zlib1g-dev libpng12-dev
```

L'utilisation de sudo facilite l'installation et ne modifira que les chemins (local est optionnel en fonction de la distribution) :

- /usr/local/share/games/torcs/
- /usr/local/lib/torcs/
- /usr/local/bin/torcs

Si vous n'en avez pas l'envie, vous pouvez utiliser un live CD.

1.3.2 Récupérer le code du simulateur

Il faut maintenant récupérer le code du simulateur

Pour celà, ouvrez une console et placez vous dans le répertoire Sources/torcs/ et lancer le script

```
./getTorcs
```

Vous devriez alors avoir un dossier /Sources/torcs/torcs-1.3.4/ avec tout les fichiers nécessaires à la compilation.

1.3.3 Compiler le simulateur

Toujours dans Sources/torcs/, lancer :

```
./buildTorcs
```

Si vous avez des problèmes lors de cette phase, vous pouvez vous reportez sur l'aide en ligne de TORCS : FAQ

1.4 Compilation du projet

Voici la liste des paquets que vous aurez besoin afin de compiler le projet :

- make
- g++
- cmake
- sudo
- boost (system, serialization, filesystem)
- libplib

Sur ubuntu :

```
sudo apt-get install g++ cmake libboost-dev libboost-system-dev
    libboost-serialization-dev libboost-filesystem-dev libplib-dev
```

Vous êtes alors prêt à lancer la dernière compilation.
Pour celà, il faut se rendre dans le dossier Sources/ et lancer

```
./makeDriver
```

Voilà, tout est installé et les robots ont déjà pré-appris. Lire les sections suivantes pour l'utilisation.

2 Désinstallation

Pour désinstaller tous les fichiers, il suffit de se placer dans le dossier Sources/-torcs et lancer

```
./uninstall
```

3 Paramètres de fonctionnement

Il y a peu de paramètre à mettre en place sans modifier le code, donc voir la section associé dans le Manuel du Programmeur.

On parlera quand même des paramètres possible pour les scripts présents dans Sources/driver, plus précisement learnIA, clearLearn, playIA qui peuvent prendre quelques paramètres :

- playIA
  - le numéro du robot que l'on veut voir jouer (de 0 à 5)
- clearLearn
  - le numéro du robot dont on veut supprimer l'historique (de 0 à 5)
  - sans paramètre pour déclencher des questions pour supprimer successivement l'historique de chaque robot
- learnIA
  - le numéro du robot dont on lance l'apprentissage
  - on peut ajouter un second paramètre pour lancer plusieurs simulations sur différents coeurs (à utiliser de préférence avec un apprentissage restrictif, voir Manuel du Programmeur)

La liste des robots est décrite dans la section suivante.

D'autres paramètres propre à TORCS peuvent être modifié si on s'interesse par exemple au nombre de tours qu'un robot doit faire, voir les fichiers Sources/-torcs/xml.

4 Fonctionnalités et interactions utilisateur

Liste des robots prédéfinis :

Index Robot	Algorithme	Etat	Action	Feedback	Restriction
0	Q-Learning	distance au centre, tangente à la route	direction	Sans	vitesse = 30km/h
1	Q-Learning(λ)	distance au centre, tangente à la route	direction, 4 actions vitesse	Sans	vitesse < 70km/h
2	Q-Learning(λ) par descente de gradient	distance au centre, tangente à la route	direction, 4 actions vitesse	Sans	vitesse < 70km/h
3	Q-Learning(λ) par descente de gradient	distance au centre, tangente à la route vitesse prochain virage	direction, 4 actions vitesse	Contrôle	Sans
4	Q-Learning(λ) par descente de gradient	distance au centre, tangente à la route	direction, 4 actions vitesse	Superviseur	vitesse < 70km/h
5	Sarsa(λ)	distance au centre, tangente à la route, vitesse	direction, 4 actions vitesse	Contrôle	Sans

La pluspart des algorithmes sont tirés du livre [Sutton and Barto, 1998]. Notons les 2 types de feedback, dont nous allons détailler un peu plus l'utilisation :

Superviseur Le tuteur peut avec ce robot, lui envoyer un feedback sur ces récompenses à l'aide des touches S, D, F, G.

F et G pour des récompenses possitives.

S et D pour des récompenses négatives.

L'ordre étant le suivante S < D < F < G.

L'utilisateur peut vérifier que ces feedbacks sont bien intégrés en regardant la console après avoir lancé un playIA par exemple.

Contrôle Nous avons intégré un second type de feedback, plus intéressant, permettant à l'agent de voir quels actions le tuteur ferait à sa place et d'apprendre directement des actions du tuteur. Pour prendre le contrôle du véhicule, le tuteur doit presser la touche contrôle gauche durant un petit laps de temps jusqu'à qu'il voit le signal dans la console.

Il sera alors en mesure de diriger l'appareil à l'aide des flêches directionnelles du clavier.

Pour redonner la main à l'agent, il faut à nouveau exercer une pression sur la touche contrôle gauche.

La définition de nouveaux robots se voit dans le Manuel du Programmeur [Zhou and Zimmer, 2013b].

Références

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning : An Introduction*. MIT Press, Cambridge, MA.

[TORCS Team, 2001] TORCS Team (2001). The open racing car simulator. `http://torcs.sourceforge.net/`.

[Zhou and Zimmer, 2013a] Zhou, L. and Zimmer, M. (2013a). *Cahier des charges - PIAD Semi-supervised Learning Agents*.

[Zhou and Zimmer, 2013b] Zhou, L. and Zimmer, M. (2013b). *Manuel du Programmeur - PIAD Semi-supervised Learning Agents*.