Manuel d'utilisation

Configuration de l'environnement

Afin de configurer votre environnement et pouvoir tester nos implémentations, veuillez suivre les indications suivantes:
Avant tout, téléchargez le répertoire contenant nos implémentations de la manière suivante:

```
cd ddpg_simulation
git clone https://github.com/WissamAKRETCHE/DDPG-Keras-TORCS.git
cd DDPG-Keras-Torcs
```

Vous aurez besoin de Python 2.7 pour l'exécution de l'algorithme DDPG et de Python3 pour l'exécution des deux autres algorithmes, TD3 et PPO.
Tout sera fait dans le répertoire *ddpg_simulation*. Dans la suite du manuel on supposera que vous l'avez placé dans votre *home directory*.
Afin de nous assurer que les librairies que vous avez préalablement installées sur votre ordinateur ne soient pas en conflit avec les librairies requises pour ce projet, nous vous recommandons de créer un environnement virtuel.
Dans un premier temps, nous vérifions si la commande *virtualenv* est installée sur votre machine. Pour ce faire, exécutez la commande suivante:

```
virtualenv --version
```

Si cette commande n'existe pas, exécuter la commande suivante:

```
pip install virtualenv
```

Créez votre environnement Python 2.7:

cd $\sim$

créez l'environnement:

```
virtualenv -p python2 ddpg_simulation
```

placez vous dans le répertoire:

```
cd ddpg_simulation
```

activez l'environnement virtuel:

```
source bin/activate
```

Mettez le fichier *requirements.txt* disponible dans votre répertoire *DDPG-TORCS-simulation* dans le répertoire *ddpg_simulation* puis installez les en exécutant la commande suivante:

```
pip2 install -r requirements.txt
```

Le fichier de configuration contient les bibliothèques nécessaires à l'exécution de l'algorithme et à la simulation comme Keras pour créer et gérer les réseaux de neurones et *gym* qui est une collection d'environnements largement utilisés pour exécuter des tests sur les algorithmes d'apprentissage par renforcement.

Installation de gym_torcs

L'entraînement de nos réseaux de neurones se fait sur Gym-TORCS, un environnement TORCS ayant une interface compatible avec celle de Open-AI-gym. Afin de l'installer veuillez suivre les instructions suivantes:
Exécuter la commande suivante:

xautomation

Cette bibliothèque permet de simuler l'utilisation du clavier et de la souris, ainsi que de manipuler des fenêtres. Il est utilisé pour contrôler la simulation en cours via l'algorithme.
Pour plus d'informations, rendez-vous sur: http://linux.die.net/man/7/xautomation .

```
sudo apt-get install xautomation
```

Installation de TORCS

TORCS (*The Open Racing Car Simulator* est un simulateur de course automobile qui permet l'exécution de tests avec des pilotes *intelligents* préprogrammés.

Plib 1.8.5

Tout d'abord, vous devrez installer Plib, un ensemble de bibliothèques pour développer des jeux, tels que l'audio et le contrôle.

Pour l'installer, veuillez exécuter les commandes suivantes:

Installez les bibliothèques requises en exécutant la commande:

```
sudo apt-get install libglib2.0-dev libgl1-mesa-dev
libglu1-mesa-dev freeglut3-dev libplib-dev libopenal-dev
libalut-dev libxi-dev libxmu-dev libxrender-dev libxrandr-dev
libpng-dev
```

Téléchargez plib et extrayez dans le répertoire ddpg_simulation/gym_torcs:

```
wget http://plib.sourceforge.net/dist/plib-1.8.5.tar.gz
sudo updatedb
plib_folder=$(locate plib-1.8.5.tar.gz)
tar xfvz $plib_folder
```

Installez la:

```
cd plib-1.8.5
```

Si vous utilisez une version 64 bit version de Linux, exportez les variables suivantes (http://www.berniw.org/tutorials/robot/torcs/install/plib-install.html) :

```
export CFLAGS="-fPIC"
export CPPFLAGS=$CFLAGS
export CXXFLAGS=$CFLAGS
```

Exécutez les commandes suivantes (pour toutes les versions):

```
./configure
sudo make
sudo make install
```

Si vous utilisez une version 64 bit version de Linux, restaurez les variables précédentes:

```
export CFLAGS=
export CPPFLAGS=
export CXXFLAGS=
```

TORCS

Pour installer TORCS, veuillez exécuter les commandes suivantes:

```
cd ddpg_simulation/gym_torcs/vtorcs-RL-color directory
sudo ./configure
sudo make
sudo make install
sudo make datainstall
```

Terminé !
Pour vérifier si TORCS a été installé avec succès, exécutez:

```
sudo torcs
```

Torcs UI devrait s'ouvrir.

DDPG

Vous pouvez tester l'algorithme DDPG en exécutant les commandes suivantes:

```
cd ddpg_simulation directory
cd DDPG-Keras-Torcs
python2 ddpg.py
```

Changez la variable *train_indicator* $= 1$ dans le fichier ddpg.py si vous souhaitez entraîner le réseau.

Analyse du contrôleur obtenu avec DDPG

Si vous souhaitez analyser le comportement de votre contrôleur après l'avoir entraîné avec DDPG, cela se fera en plusieurs étapes. Générez d'abord les données d'activation du contrôleur sur le circuit souhaité (à modifier manuellement dans le fichier *ddpg_analysis.py*) en exécutant la commande suivante :

```
python2 ddpg_analysis.py
```

Les données d'activation apparaîtront alors dans des fichiers placés dans un dossier nommé Activations (à créer). Vous pourrez ensuite appliquer t-SNE sur ces données en exécutant la commande suivante :

```
python2 ActivationsAnalysis.py
```

Après cela, libre à vous d'utiliser les méthodes de votre choix pour interpréter les résultats fournis par t-SNE (la méthode des k-moyennes étant déjà implémentée dans *ActivationsAnalysis.py*).

PPO

Notez que pour tester cet algorithme vous aurez besoin d'installer Tensorflow (compatible avec Python3):

```
sudo apt update
sudo apt install python3-dev python3-pip
pip3 install tensorflow
```

Afin de tester l'algorithme PPO, veuillez exécuter les commandes suivantes:

```
cd to ddpg_simulation directory
cd DDPG-Keras-Torcs/PPO
python3 main.py
```

Changez la variable *train_test* $= 0$ dans le fichier main.py si vous souhaitez entraîner le réseau.
Changez la variable *irestart* $= 1$ dans le fichier ddpg.py si vous souhaitez exécuter l'algorithme avec les poids appris préalablement.

TD3

Notez que pour tester cet algorithme vous aurez besoin d'installer Pytorch (compatible avec Python3):

```
sudo apt update
sudo apt install python3-dev python3-pip
pip3 install torch
```

Afin de tester l'algorithme TD3, veuillez exécuter les commandes suivantes:

```
cd to ddpg_simulation directory
cd DDPG-Keras-Torcs/TD3
python3 main.py
```

Changez la variable $train\_test = 0$ dans le fichier main.py si vous souhaitez entraîner le réseau.
Changez la variable $irestart = 1$ dans le fichier ddpg.py si vous souhaitez exécuter l'algorithme avec les poids appris préalablement.

Notes

Il est possible que la vue de course soit définie sur "Tracks view" (Vue de piste) au lieu de celle du conducteur. Lorsque la simulation est en cours, appuyez sur F2 pour vérifier si vous êtes dans cette situation. Si vous n'allez pas de l'avant ou si la vue change complètement, cela signifie que vous l'êtes. Vous pouvez procéder comme suit pour modifier la configuration de TORCS:
Tout d'abord, vous devez modifier les paramètres pour pouvoir contrôler la voiture dans TORCS:

```
sudo torcs
```

Sélectionnez Race > Practice > Configure Race > Accept.
Sélectionnez scr_server1 dans le rectangle de gauche, et cliquez sur (De)Select.
Sélectionnez Player dans le rectangle de droite, puis cliquez sur (De)Select.
Cliquez sur Accept, puis Accept une seconde fois.
Vous pouvez maintenant démarrer une course, fixer la vue puis quitter la course:
Lancez une course: New Race
Appuyez sur F2 lorsque la course est lancée (afin de configurer la vue)
(Appuyez sur F1 si vous voulez explorez les autres options)
Appuyez sur Esc, puis Abandon Game
Changez la configuration du joueur afin de permettre à notre algorithme de contrôler la voiture: scr_server1:
Dans le menu *PracticeMenu*, sélectionnez $ConfigureRace$ > Accept
Sélectionnez Player dans le rectangle de gauche, puis cliquez sur (De)Select. Sélectionnez scr_server1 dans le rectangle de droite, et cliquez sur (De)Select.

Cliquez sur Accept, puis Accept une seconde fois.
Vous pouvez maintenant quittez TORCS: appuyer sur Esc jusqu'à ce que la fenêtre *Quit*? apparaîsse, cliquez sur Yes, Let's Quit.

Si vous souhaitez quitter l'environnement virtuel, exécutez la commande suivante :

```
deactivate
```