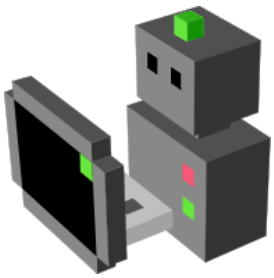
Tutoriel d'Installation

du simulateur robotique 3D

BLENDER/MORSE

Version 1

Rédacteur : Athmane AYOUNI



Descriptif :

Ce document est un tutoriel d'installation du simulateur robotique 3D BLENDER/MORSE.

Ce tutoriel est utilisable sur la distribution Linux Ubuntu 12.04 LTS ou supérieure et nécessite une connexion Internet (pour le téléchargement et l'installation des logiciels).

Il est composé de **deux** parties :

-L'installation des logiciels et bibliothèques nécessaires au fonctionnement du simulateur

-L'installation des extensions liées à la simulation du CAMELEON et à l'IHM

Liste des composants à installer :

- Blender 2.63
- MORSE 0.6 (**M**odular **O**pen **R**obots **S**imulator **E**ngine)
- ROS fuerte (**R**obot **O**perating **S**ystem)
- Python 3.2 (Permet de lier ROS à MORSE)
- WxWidets 2.8
- Terminal Konsole KDE (préconisé si vous utilisez l'IHM du simulateur)

Sommaire :

Noter que pendant l'installation des différents logiciels, l'opérateur sera amené à rentrer le mot de passe de la session à plusieurs reprises, ainsi qu'à répondre par des *YES/NO questions*, où il faudra bien sûr répondre par Yes (touche Y).

1. Installation du simulateur BLENDER/MORSE

1.1 Installation de **ROS Fuerte**

Préconfiguration d'installation

Ouvrir un terminal et copier-coller les commandes suivante :

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu precise main" > /etc/apt/sources.list.d/ros-latest.list'
```

```
wget http://packages.ros.org/ros.key -O - | sudo apt-key add -
```

```
sudo apt-get update
```

Installation de ROS Fuerte

```
sudo apt-get install ros-fuerte-desktop-full
```

L'installation va prendre quelques minutes.

Installation des variables d'environnement de ROS

```
echo "source /opt/ros/fuerte/setup.bash" >> ~/.bashrc
. ~/.bashrc
```

Vérification d'installation de ROS Fuerte

```
bash
```

```
roscore
```

Si l'installation s'est bien déroulée, vous devriez voir apparaître ceci :

```
hudsonservice@samarium:/opt/ros/fuerte$ roscore
... logging to /home/hudsonservice/.ros/log/e8f153a4-3564-11e2
.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://samarium:51024/
ros_comm version 1.8.10

SUMMARY
========

PARAMETERS
 * /rosdistro
 * /rosversion

NODES
```

Fermer le terminal.

1.2 Installation de Python 3.2

Ouvrir un terminal et copier-coller les commandes suivantes :

```
cd /opt/ros
```

```
sudo mkdir python
```

```
cd python
```

```
sudo wget http://sunsite.informatik.rwth-aachen.de/python/ftp/python/3.2.1/Python-3.2.1.tar.bz2
```

```
cd Python-3.2.1
```

```
sudo ./configure --prefix=/opt/ros/fuerte --without-pymalloc --with-wide-unicode --enable-shared LDFLAGS="-Wl,-rpath /usr/local/lib"
```

```
sudo make -j2
```

```
sudo make install
```

Fermer le terminal.

1.3 Installation de MORSE 0.6

Ouvrir un terminal et copier-coller les commandes suivantes :

```
mkdir Installation_morse
```

```
cd Installation_morse
```

```
git clone https://github.com/laas/morse.git -b 0.6_STABLE
```

```
cd morse
```

```
mkdir build && cd build
```

```
sudo cmake -DCMAKE_INSTALL_PREFIX=/usr/local/ -
DBUILD_CORE_SUPPORT=ON -DBUILD_ROS_SUPPORT=ON -
DPYMORSE_SUPPORT=ON -DCMAKE_BUILD_TYPE=Release -
DPYTHON3_EXECUTABLE=/opt/ros/fuerte/bin/python3.2 -
DPYTHON_EXECUTABLE=/usr/bin/python2.7 ..
```

```
sudo make –j2
```

```
sudo make install
```

Fermer le terminal.

1.4 Installation de Blender 2.63

Ouvrir un terminal et copier-coller les commandes suivantes :

```
sudo apt-get update; sudo apt-get install subversion build-essential gettext \
libxi-dev libsndfile1-dev \
libpng12-dev libjpeg-dev libfftw3-dev \
libopenexr-dev libopenjpeg-dev \
libopenal-dev libalut-dev libvorbis-dev \
libglu1-mesa-dev libsdl1.2-dev libfreetype6-dev \
libtiff4-dev libavdevice-dev cmake \
libavformat-dev libavutil-dev libavcodec-dev libjack-dev \
libswscale-dev libx264-dev libmp3lame-dev python3.2-dev git \
libspnav-dev libtheora-dev libjack-dev libglew1.6-dev curl
```

```
cd /opt
```

```
sudo mkdir blender && cd blender
```

```
sudo wget http://download.blender.org/source/blender-2.63a.tar.gz
```

```
sudo tar xvvf blender-2.63a.tar.gz
```

```
cd blender-2.63a
```

```
sudo mkdir build && cd build
```

```
sudo cmake -DCMAKE_INSTALL_PREFIX=/opt/ros/fuerte -
DWITH_BUILDINFO=ON -DWITH_IK_ITASC=ON -
DWITH_BULLET=ON -DWITH_GAMEENGINE=ON -
DWITH_PLAYER=ON -DWITH_PYTHON_INSTALL=ON -
WITH_OPENMP=ON ..
```

```
sudo make –j2 && make install
```

```
cd /usr/local/bin
```

```
ln –s  /opt/blender/blender-2.63a/build/bin/blender blender
```

```
ln –s /opt/blender/blender-2.63a/build/bin/blenderplayer blenderplayer
```

Fermer le terminal.

1.5 Installation de rospkg

Ouvrir un terminal et copier-coller les commandes suivantes :

```
cd Installation_morse
```

```
git clone git://github.com/ros/rospkg.git
```

```
cd rospkg
```

```
python3.2 setup.py install
```

Fermer le terminal

1.6 Installation de pyyaml

Ouvrir un terminal et copier-coller les commandes suivantes :

```
cd Installation_morse
```

```
wget http://pyyaml.org/download/pyyaml/PyYAML-3.10.tar.gz
```

```
tar xvvf PyYAML-3.10.tar.gz
```

```
cd PyYAML-3.10
```

```
python3.2 setup.py install
```

Fermer le terminal

1.7 Edition des variables d'environnements

L'édition des variables d'environnements permet de relier les bibliothèques des différents logiciels utilisés : ROS, Blender, Morse et Python.

Ouvrir un terminal et copier-coller la commande suivante :
(Ouverture de l'éditeur texte gedit pour édition du fichier contenant les variables d'environnements .bashrc)

```
gedit .bashrc
```

A la fin du fichier, copier-coller ces lignes et remplissant les champs en italique par les noms appropriés.

```
#MORSE
export MORSE_ROOT=/usr/local
export MORSE_BLENDER=/usr/local/bin/blender
export PATH=/usr/local/bin:${PATH}
export PYTHONPATH=/usr/local/lib/python3.2/site-packages:/opt/ros/fuerte/lib/python2.7/dist-packages
export ROS_PACKAGE_PATH=${ROS_PACKAGE_PATH}:/home/nom_du_pc/nom_du_dossier_ros
export ROS_WORKSPACE=/home/nom_du_pc/nom_du_dossier_contenant_noeud_ros
```

2 Installation des extensions spécifiques au CAMELEON et à l'IHM

2.1 Installation du terminal Konsole KDE

Si vous utilisez le simulateur avec son IHM, il est indispensable d'utiliser le terminal Konsole

Installation

Ouvrir un terminal et copier-coller la commande suivante :

```
sudo apt-get install konsole
```

Une fois l'installation terminée, le terminal est prêt à être utilisé.

2.2 Installation des fichiers spécifiques au caméléon

Une fois tous les logiciels et middlewares précédents installés, il est indispensable d'installer des fichiers nécessaires au fonctionnement du caméléon simulé et notamment pour sa commande.

Il y a 6 fichiers à copier et 1 à modifier.

Rappel de la méthode de copie de fichiers :

Ouvrir un terminal et se placer dans le dossier contenant les 6 fichiers.

Puis taper la commande suivante :

```
cp nom_du_fichier repertoire_de_destination
```

Liste des fichiers à copier et leur destination :

Fichiers	**Destination**
cameleon_v5.py	**/usr/local/lib/python3.2/site-packages/morse/robots**
read_cameleon.py	**/usr/local/lib/python3.2/site-packages/morse/middlewares/ros**
tracked_robot.py	**/usr/local/lib/python3.2/site-packages/morse/core**
wheeled_robot.py	**/usr/local/lib/python3.2/site-packages/morse/core**
v_omega_diff_cameleon.py	**/usr/local/lib/python3.2/site-packages/morse/actuators**
v_omega_diff_cameleon.blend	**/usr/local/share/morse/data/robots**

Fermer le terminal

Une fois ces fichiers copiés dans les répertoires indiqués, il reste à modifier le fichier **data.py** situé dans le répertoire **/usr/local/lib/python3.2/site-packages/morse/builder**

Méthode :

Ouvrir un terminal et taper la commande suivante :

```
sudo gedit /usr/local/lib/python3.2/site-packages/morse/builder/data.py
```

Le terminal vous demande le mot de passe session, entrez-le puis valider par la touche Entrée.

L'éditeur de texte **gedit** ouvre le fichier data.py.

- Se rendre à la ligne 93 : après la ligne 'v_omega' …
- Rajouter la ligne suivante :

```
'v_omega_diff_cameleon': [MORSE_MIDDLEWARE_MODULE['ros'], 'read_twist',
'morse/middleware/ros/read_cameleon'],
```

Sauvegarder et fermer l'éditeur de texte gedit.

Fermer le terminal.

2.3 Installation des packages ROS nécessaires à la simulation

De nombreux packages ROS sont indispensables au fonctionnement de la simulation : ces packages se présentent sous la forme d'un dossier à placer dans le répertoire commun des packages ros (dossier /home/ros-proteus sur le pc proteus).

Liste des packages ROS :

- chameleon_teleop
- chameleon_morse_controller
- gyro_data
- inclino_data
- inclinometre_data
- nd_local_nav
- chameleon_2dnav
- autonav_GUI
- scan_data
- chameleon_global_planner

Il faut ensuite compiler chacun d'entre eux, voici la méthode : (exemple à travers le PC proteus)

Ouvrir un terminal et taper les commandes suivantes pour une première compilation.

```
cd ros-proteus/
```

```
cd nom_du_package_ros
```

```
rosmake
```

Si le package a déjà été compilé au moins une fois, vous pouvez accéder directement au dossier directement en tapant la commande suivante :

```
roscd nom_du_package
```

```
rosmake
```

Attention : Si vous compilez sur un PC autre que le *proteus*, il faudra au préalable remplacer dans les fichiers CMakeCache.txt et Makefile situés dans le dossier *build* le nom du repertoire des packages ros. (Faire un Find and Replace : exemple /home/devel/ros-devel par /home/proteus/ros-proteus).

2.4 Installation de l'IHM

L'IHM de supervision a été développé avec la librairie C++ WxWidgets 2.8

Installation

Ouvrir un terminal copier-coller la commande suivante :

```
sudo apt-get install libwxgtk2.8-dev wx-common
```

Une fois terminé, la librairie est opérationnelle.

Les fichiers sources de simulation (IHM, blender, python) sont situés dans le dossier *Simulation_3D_cameleon*

Liste des fichiers de l'IHM *(dossier /Simulation_3D_cameleon/Fichiers_Blender_Morse)*:

- fenetre_bouton.cpp
- fenetre_bouton.h
- application.cpp
- application.h
- Makefile

Liste des fichiers python de la simulation *(dossier /Simulation_3D_cameleon/Fichiers_Blender_Morse)* :

- simulateur.py
- camera_arr.py
- camera_front.py
- controleur.py
- environnement.py
- gps.py
- gyrometre.py
- inclinometre.py
- odometry.py
- robot.py

Liste des fichiers blender *(dossier /Simulation_3D_cameleon/Fichiers_Blender/Environnement)* :

- eca_saclay_interieur.py
- eca_saclay_exterieur.py

Liste des fichiers blender *(dossier /Simulation_3D_cameleon/Fichiers_Blender/robot)* :

- cameleon_v5.blend
- Wifibot2.blend

Il est nécessaire de compiler le code de l'IHM **la première fois** pour créer l'exécutable.

Pour cela, **ouvrir** un terminal et copier-coller la commande suivante

(Entrée dans le répertoire de travail)

```
cd Simulation_3D_cameleon
```

(Compilation)

```
make
```

Si la compilation ne retourne pas d'erreur, un exécutable apparait dans le dossier */Simulation_3D_cameleon*

2.5 Intégration de modules de simulation

Parmi les options présentes dans l'IHM du simulateur, vous pouvez intégrer vous-même des modules externes de simulation, comme par exemple des modules SLAM (voir tutoriel de fonctionnement du simulateur).

Il suffira de se rendre dans le dossier /home/proteus/Simulation_3D_cameleon/Modules/

Ces modules externes sont des commandes lancées par un fichier script shell .sh .

Si vous voulez éditez un nouveau fichier script et y insérer des commandes, voici la procédure à suivre :

Ouvrir un terminal et copier-coller la commande suivante :

(Entrée dans le répertoire contenant les modules)

```
cd Simulation_3D_cameleon/Modules/
```

(Edition d'un nouveau fichier script shell)

```
gedit nom_fichier.sh &
```

L'éditeur de texte s'ouvre, éditez votre fichier de la manière suivante :

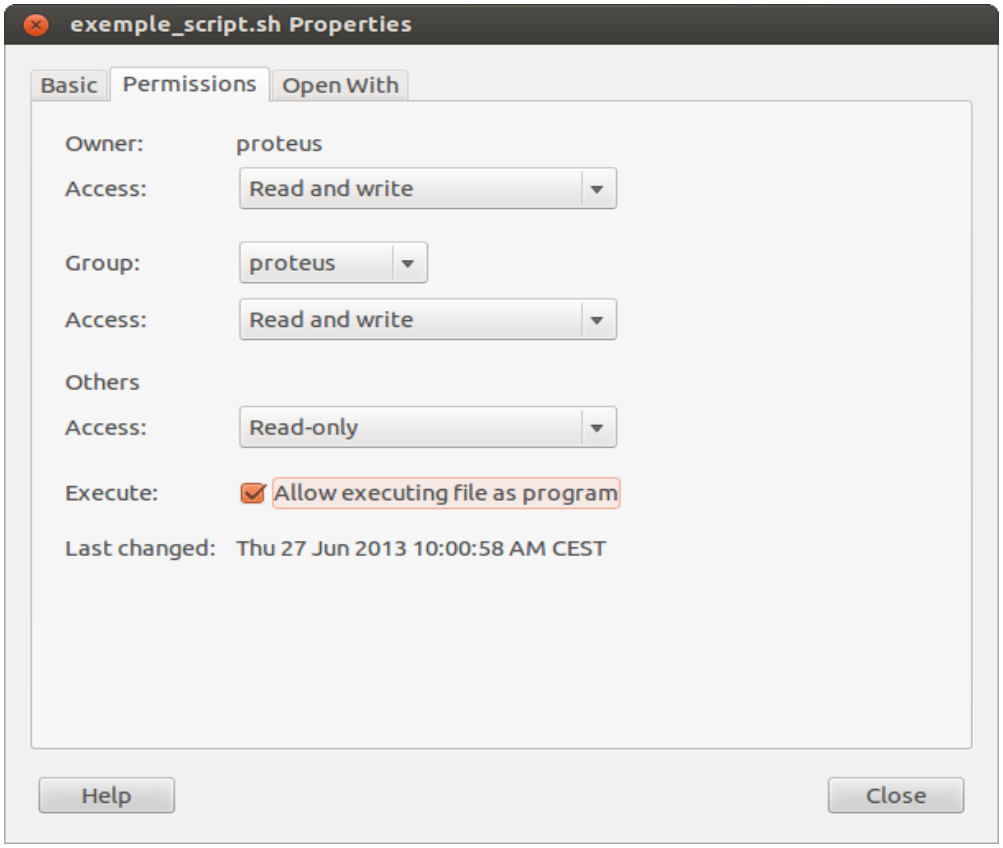
```
#!/bin/sh

commande1 &
commande2 &
```

Enregistrer votre fichier.

Retourner dans le dossier *Simulation_3D_cameleon/Modules/*

Faire un clic droit sur votre fichier script shell .sh et cliquer sur Propriétés.

La fenêtre suivante s'ouvre et **cocher** la case **Allow executing file as program** afin de rendre le script exécutable:



Votre script est opérationnel.

FIN de l'installation