Installation Carla-Ros-Bridge

Afin de réaliser cette installation, on se basera sur un conteneur docker sur la version Ubuntu 18.04 pour des raisons de compatibilité principalement, mais cela est également réalise sur Ubuntu 20.04.

Nous allons nous baser sur ce repository github réalisé par atinfinity.

Avant de commencer :

- Système Linux natif
→ Ubuntu / Debian recommandé
→ Problèmes de performance avec une VM

- Installer Docker
→ Référez vous aux liens ci-dessous

- Carte graphique Nvidia vivement recommandé

- Téléchargement de **Carla_0.9.11**
→ Pour Ubuntu choisir CARLA_0.9.11.tar.gz
→ Voir dernier lien ci-dessous

- Installer Docker sur Ubuntu
- Installer Docker sur Debian
- Installer Carla sur Linux
  → *Choisir une version*
  → *Extraire le tar gz*
  → *Lancer CarlaUE4.sh*

- Choisissez un dossier sur votre système pour l’emplacement du répertoire github que l’on va cloner

```
git clone https://github.com/atinfinity/carla_ros_bridge_docker
```

- Placez vous dans le répertoire (où il y a le fichier Dockerfile.melodic)

```
cd carla_ros_bridge_docker/
```

- Effectuez le build de l’image

```
docker build -t carla:0.9.11 --build-arg GID=$(id -g) --build-arg UID=$(id -u) -f Dockerfile.melodic .
```

- Lancer le conteneur

```
sudo ./launch_container.sh
```

Une fois cette 1ere étape réalisée, nous allons mettre en place les processus pour le bridge carla ros. Ce conteneur permet également d'utiliser RVIZ qui est une interface graphique pour ros.

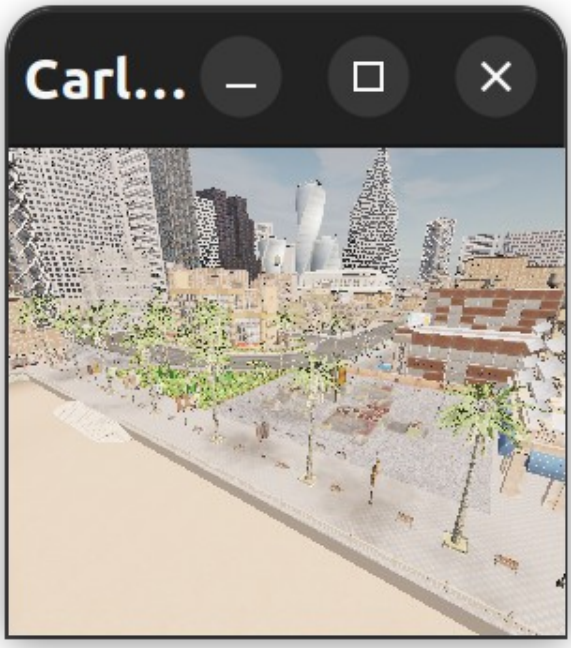
- Dans un terminal en dehors du conteneur, on lance le serveur Carla

```
sudo docker exec -it <id_conteneur> sh -c « /opt/carla-simulator/CarlaUE4.sh windowed -ResX=160 -ResY=120 -opengl »
```

La commande exec de docker permet d'effectuer des actions dans un terminal dans un conteneur déjà ouvert.

L'argument "-opengl" est nécessaire pour le bon fonctionnement avec GPU Nvidia

On choisit également une résolution basse pour que les performances ne soient pas trop impactées. Il existe également l'argument « -quality-level=Low » pour baisser la qualité.



Une fenêtre comme celle-ci va alors s'ouvrir, il est nécessaire qu'elle reste ouverte pour laisser le serveur de Carla tourner en fond.

Il s'agit de la carte par défaut, qui est modifiable, voir commande d'après.

- Ensuite on ouvre un nouveau terminal pour modifier la configuration de la fenêtre serveur de Carla (commande adaptable)

```
sudo docker exec -it <id_conteneur> sh -c "cd /opt/carla-simulator/PythonAPI \
&& python util/config.py -m Town03 --fps 10"
```

On charge la carte 3 et on modifie le taux de fps max à 10.

- Dans le terminal du conteneur on lance le bridge entre Carla et Ros avec une simulation exemple

```
roslaunch carla_ros_bridge carla_ros_bridge_with_example_ego_vehicle.launch \
vehicle_filter:='vehicle.toyota.prius*'
```

Voir les autres fichiers exemples dans le répertoire suivant

~/carla-ros-bridge/catkin_ws/src/ros-bridge/carla_ros_bridge/launch/

- On peut également aller dans un 3e terminal pour lancer des simulations en parallèle, ici on va utiliser un exemple fourni par Carla (non bridgée)

```
sudo docker exec -it <id_conteneur> sh -c \
"cd /opt/carla-simulator/PythonAPI/examples \
&& python automatic_control.py"
```

La simulation « automatic_control.py » présente un véhicule étant conduit automatiquement jusqu'à un point précis, où la simulations s'arrêtera.

Le choix du véhicule est aléatoire.

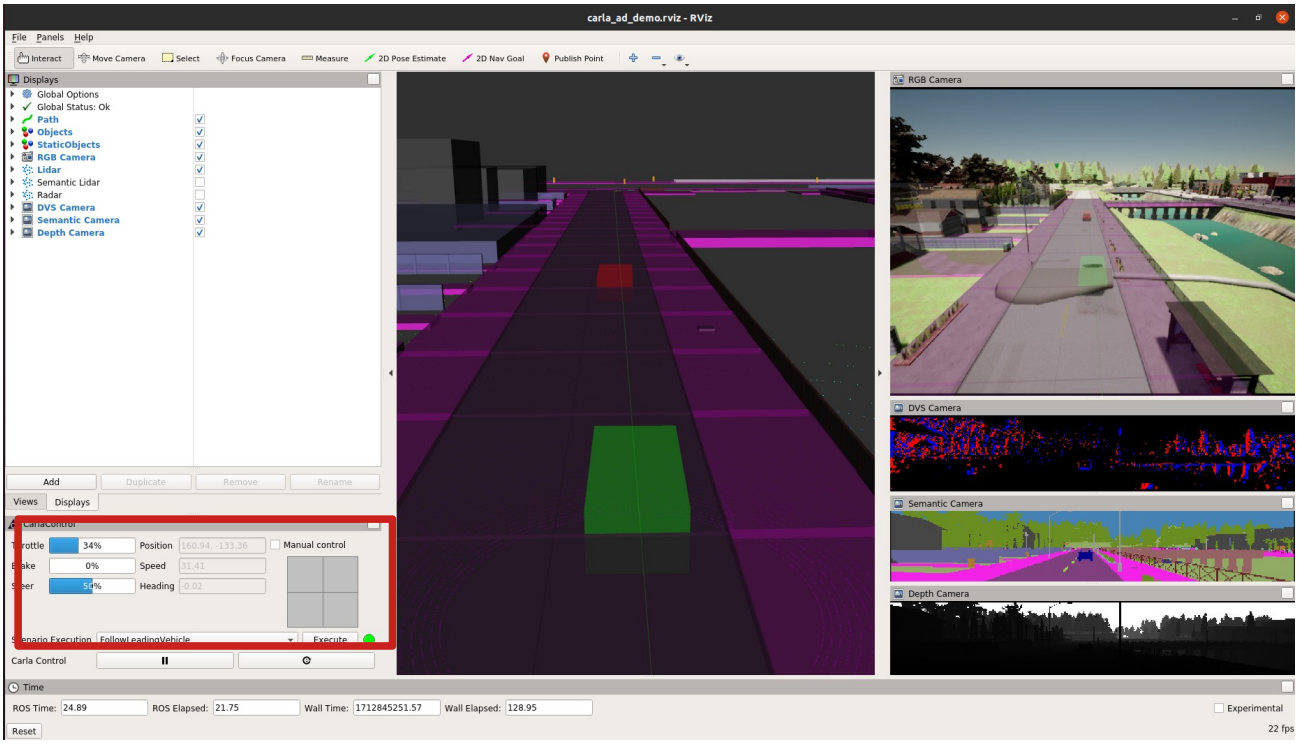
Si on souhaite utiliser l'interface graphique rviz, on peut directement charger des simulations dedans sans avoir ouvert le serveur Carla.

- Lancer le bridge carla-ros avec RVIZ qui offre une interface graphique

```
roslaunch carla_ad_demo carla_ad_demo_with_scenario.launch \
vehicle_filter:='vehicle.toyota.prius*'
```

Ici on choisit une simulation exemple pour la démo.
On a également préciser un véhicule spécifique avec l'argument « vehicle_filter ».

Quand on appuie sur « EXECUTE » le scénario commence à savoir la voiture verte qui suit la rouge.



À droite on a la simulation exemple comme dans Carla

On a également les données de ROS dans l'interface graphique RVIZ

Dans la zone qu'indique l'encadré rouge on peut voir entre autre

- *Throttle (la puissance sur l'accélérateur)*
- *La vitesse*
- *La position*