Objectif : Réaliser un état de l’art des outils de simulation mono et multi-drones, avec démonstration des différents outils et un bilan comparatif.

1. Quel robot DJI ?

L’objectif est de travailler sur le drone DJI M600, au vu du peu de documentation à ce sujet nous étudierons d’abords des modèles moins récents.

Pour le DJI M100 :

* Beaucoup de documentation open source (github)
* Pas de documentation vraiment à jour sur le site officiel

Pour le DJI M210 :

* Beaucoup de documentation sur le site officiel pour le V2,
* Quasiment rien en open source (github)

Tout les deux des quadrirotors (donc attention à télécharger un pkg pour 6 rotors pour le M600).

→ On choisis le M100 car la documentation est assez claire, disponible en open source (assez de ressources disponible) et les codes sont à jour.

1. Quels pkg ?

Choix 1 : avec des packages présents sur GitHub :

Un package ROS pour simuler le drone DJI M100 dans Gazebo : <https://github.com/dji-m100-ros/dji_m100_gazebo>

* Sous ROS Kinetic mais compatible avec Melodic,
* Sous gazebo 7 ou 8,
* En C++,
* Capacités : possible d’être lancer en collision free (dans models), customisable : plusieurs mondes soit collision sphere soit search and rescue,

Fonctionne avec <https://github.com/dji-m100-ros/dji_m100_description>, qui ajoute des features et des parametres pour afficher des données sur Rviz (lesquelles ?)

Doit être connecté au drone ? Voir photo plus bas et fonctionnement. Pour la config doit allumer le drone et le connecter au simulateur du PC (possible sans vrai drone mais sous Gazebo? Utilise des param du vrai robot), il faut donc émuler les commandes avec

* + <https://github.com/dji-m100-ros/dji_m100_controllers_gazebo>, un package ROS pour controller le drone dans la simulation via un gamepad logitech ou Xbox
  + Et/ou avec <https://github.com/lharikrishnan1993/hector_keyboard_teleop> qui permet un control au clavier du drone

Dans dji\_m100\_controller\_gazebo : This class represents the DJI M100 hardware in Gazebo for the gazebo\_ros\_control plugin during SITL.

Attention, on doit installer des dépendances pour que les packages fonctionnent comme le hector\_quadrotor, …

Choix 2 : on peut utiliser le DJI assistant avec le Onboard computer OSDK :

**Est ce un assistant avc une simulation déjà implanté ou compatible avec une simulation ??**

Configuration software sous Linux:

* Install C, C++ Compiler and Development Tools by installing build-essential
* Install CMake 2.8.3 or newer
* Install ROS and its dependencies
* Operating System: Ubuntu 16.04 (x86/ARM)
* ROS version: ROS Kinetic
* Tutoriel : <https://developer.dji.com/onboard-sdk/documentation/quickstart/development-environment.html> OU <http://wiki.ros.org/dji_sdk/Tutorials/Getting%20Started>
* Le code à télécharger est disponible sur <https://github.com/dji-sdk/Onboard-SDK-ROS> , attention le drone M100 est disponible jusqu’à la branche 3.8.

Hardware : il n’est pas obligé d’avoir le OnBoard computer, on utilise un « third party » avec les fichiers open source disponibles et une carte STEM32 (obligatoire ? Ne pense pas si on a un ordinateur avec Linux et ROS)

Doit être connecté au drone ? Voir photo plus bas.

Capacités: flight control, camera control, telemetry, GPS missions…

Se trouve dans ce pkg dji\_sdk : This package provides a ROS interface for the DJI onboard SDK and enables the users to take full control of supported platforms using ROS messages and services.

En plus aide pour le M100 : The software in the Loop Rover Control

This tutorial explains the steps required to drive a simulated rover in the Gazebo simulator, <https://risc.readthedocs.io/1-z-gazebo-simulation.html>, nécessite la même configuration que les précédentes possibilities.

1. Tutoriel choix 1 :
2. Configuration :

Le PC sur lequel va être faite l’installation doit être au minimum sous :

* Ubuntu 16.04 (x86/ARM)
* Avoir ROS kinetic d’installé (possible compatibilité avec Melodic mais pour l’instant pas pour hector\_quadrotor\_teleop)
* Une version Gazebo 7 ou plus (à vérifier grâce à gazebo --version)
* Une version de cmake de 2.8.3 minimum (à vérifier grâce à cmake --version)
* (Un outil de développement en C, C++) → pour l’instant pas eu besoin avec The Construct

On se place donc dans un environnement dont les caractéristiques sont citées précédement.

1. Installation :

Pour installer les packages utiles, on se place dans le src de catkin et on suit cet ordre :

* On clone ce package depuis git « Dji\_m100\_gazebo » : *git clone https://github.com/dji-m100-ros/dji\_m100\_gazebo.git dji\_m100\_gazebo*
* De même pour « Dji\_m100\_description » : *git clone https://github.com/dji-m100-ros/dji\_m100\_description.git dji\_m100\_description*

→ attention aux noms des packages, ils doivent être nommé comme ceci (suivant les remaques des Packages.xml)

Dans catkin on fait un *catkin\_make* puis un *source devel/setup.bash* .

Ensuite pour diriger le robot on doit ajouter un controller, celui-ci dépend de hector\_quadrotor\_interface, hector\_quadrotor\_messager et hector\_quadrotor\_controller, on doit alors installer tout les packages suivants comme ceci :

* On vérifie à l’aide d’un *cd /opt/ros/kinetic/share (et d’un ls | grep nom\_du\_package,* au besoin) que l’on a bien ros\_control, gazebo\_ros\_control, unique\_identifier, geographic\_info, laser\_geometry, tf\_conversions, tf2\_geometry\_msgs, joy. Si il en manque on fait *sudo apt-get install ros-kinetic-nom\_du\_package.*

Ensuite on clone les repertoires dans le src de catkin :

* *git clone -b kinetic-devel https://github.com/tu-darmstadt-ros-pkg/hector\_quadrotor*
* *git clone -b catkin https://github.com/tu-darmstadt-ros-pkg/hector\_localization*
* *git clone -b kinetic-devel https://github.com/tu-darmstadt-ros-pkg/hector\_gazebo*
* *git clone -b kinetic-devel https://github.com/tu-darmstadt-ros-pkg/hector\_models*
* *git clone -b catkin https://github.com/tu-darmstadt-ros-pkg/hector\_slam*
* *cd ..*
* *catkin\_make*
* *source devel/setup.bash*

On a tout les hector d’installer on peut enfin ajouter le « dji\_m100\_controller » toujours avec *git clone https://github.com/dji-m100-ros/dji\_m100\_controllers\_gazebo.git dji\_m100\_controllers\_gazebo*

Le drone se pilotera avec une manette logitech\_gamepad.launch pour les gamepads Logitech et

xbox\_controller.launch pour les controller gamepads Xbox. Pour le piloter avec un keyboard on clonera le « hector\_quadrotor\_teleop » selon *git clone https://github.com/lharikrishnan1993/hector\_keyboard\_teleop.git hector\_keyboard\_teleop*

POUR LINSTANT EN EST LA

1. Choix final ?

Définitions :

OSDK is a development toolkit for developing applications, which could run on the onboard computer(such as Manifold 2). Developers can obtain the information from the drone by calling the interface specified in OSDK. Onboard : pour l’interfaçage avec un ordinateur embarqué avec le SDK.

Gazebo uses plugins to accept custom inputs, output custom data, and perform custom control in simulation.

ROS\_control\_pkg : sur hardware et simulation 