Chroma team simulator for BugWright2 project

Vincent LE DOZE
Chroma Team
INSA Lyon / Inria Grenoble Rhône-Alpes

vincent.le-doze@inria.fr
https://team.inria.fr/chroma/en/

Gazebo simulator
Introduction

Gazebo is a simulation tool box for robotics applications with many features such as

- Dynamics simulation
- 3D rendering of environments including lighting, shadows, and textures
- Generation of sensors data with optional and customizable noises and bias
- Complete API for custom plugins creation

Open Source Robotics Foundation

GAZEBO ROS.org CloudSim

The CHROMA team drone simulator
In brief

Gazebo+ROS based simulator that aims to make Unmaned Aerial Vehicle simulations easier to use and customize

- Generic plugins and definition files for mechanical parts, sensors, and interfaces
- Use of macros to simplify models creation
- A representation as close as possible to the real world physics
- Ability to perform “Software-in-the-loop”

Code available here :
https://gitlab.inria.fr/chroma/drone-simulator

The CHROMA team drone simulator
In details

The simulator includes the following functionality

- **Simulation of the mechanical behavior of an Unmanned Aerial Vehicle**
  - ✔ Modeling of the body's aerodynamics with lift, drag and moment
  - ✔ Modeling of rotors' aerodynamics using the forces and moments' expressions from Philppe Martin's and Erwan Salaün's 2010 IEEE Conference on Robotics and Automation paper "The True Role of Accelerometer Feedback in Quadrotor Control"

- **Gives groundtruth informations if needed**
  - ✔ Positions in East-North-Up reference frame
  - ✔ Linear velocity in East-North-Up and Front-Left-Up reference frames
  - ✔ Linear acceleration in East-North-Up and Front-Left-Up reference frames
  - ✔ Orientation from East-North-Up reference frame to Front-Left-Up reference frame (Quaternions)
  - ✔ Angular velocity of Front-Left-Up reference frame expressed in Front-Left-Up reference frame

- **Simulation of the following sensors**
  - ✔ Inertial Measurement Unit with 9DoF (Accelerometer + Gyroscope + Orientation)
  - ✔ Barometer using an ISA model for the troposphere (valid up to 11km above Mean Sea Level)
  - ✔ Magnetometer with the earth magnetic field declination
  - ✔ GPS Antenna with a geodesic map projection
  - ✔ Monocular, Stereo and Depth camera
  - ✔ UWB antenna for distance measurements

The BugWrigth2 simulation
Installations

- **Install ROS Kinetic or Melodic (full-desktop)**

http://wiki.ros.org/melodic/Installation/Ubuntu
http://wiki.ros.org/kinetic/Installation/Ubuntu

- **Install project repository**

```
git clone git@gitlab.georgiatech-metz.fr:bugwright2/bugwright2-ws.git
cd bugwright2-ws/
git checkout integration_chroma_simulator
git submodule init
git submodule update --recursive
./init.sh
./set_catkin_ignored_packages.sh

cd bugwright_ws/src/drone-simulator/
./init.sh
```

The BugWrigth2 simulation
Setting-up

- **Workspace ROS**

```
mkdir -p ~/catkin_ws/src
cd catkin_ws/src
catkin_init_workspace
ln -s ~/bugwright2-ws/bugwright_ws .
cd ..
catkin build
source devel/setup.bash
```

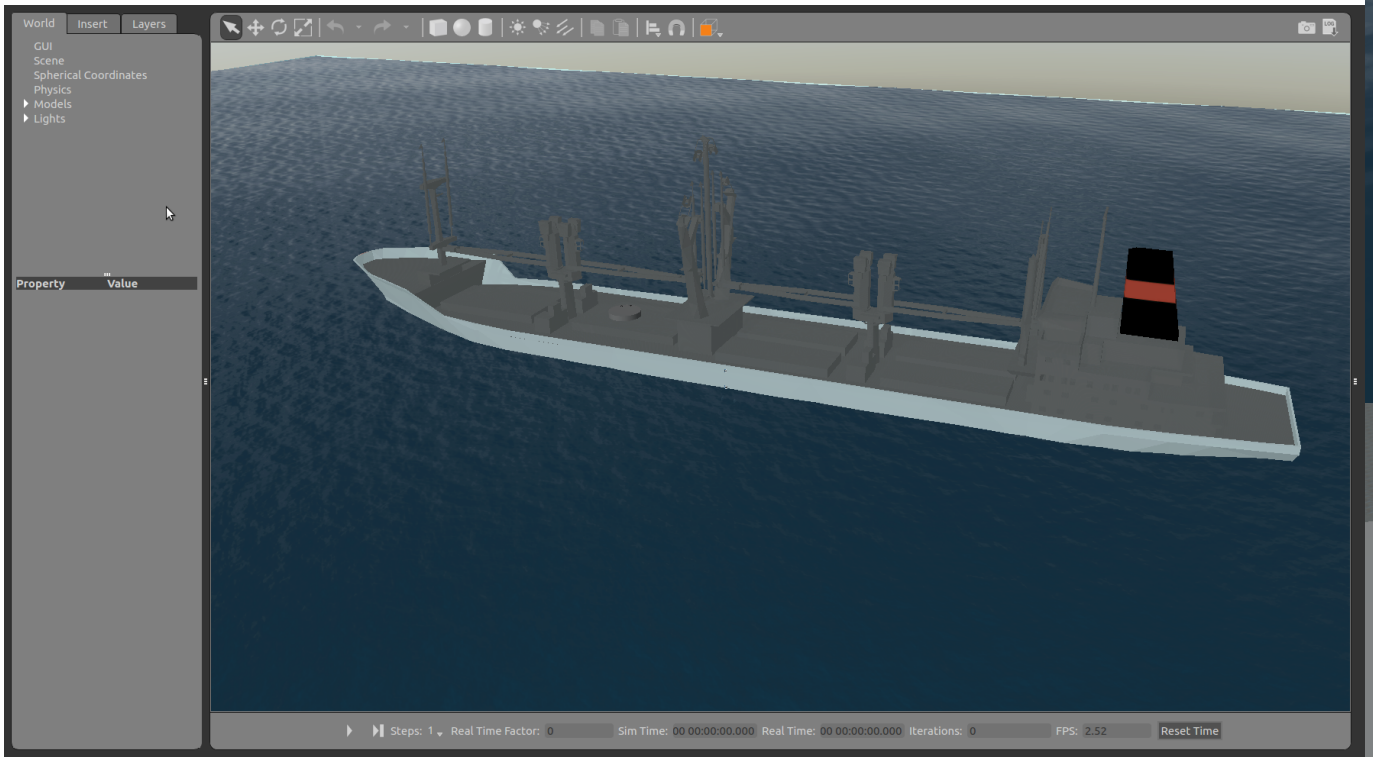
- **Link the scripts**

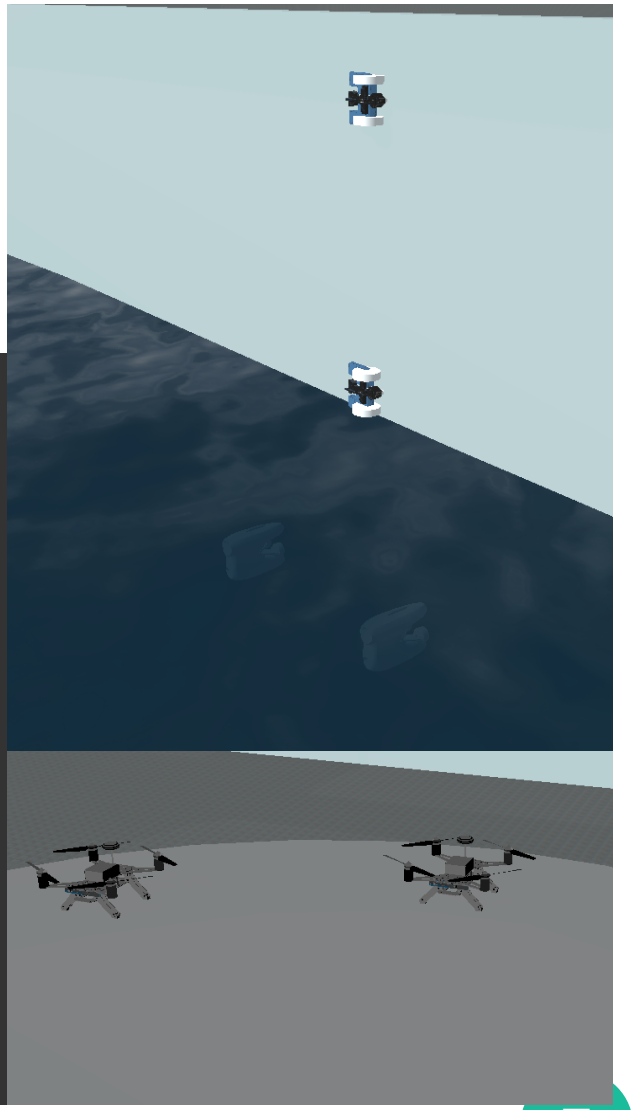
```
cd ~/catkin_ws
mkdir scripts
cd scripts
ln -s ~/bugwright2-ws/bugwright_ws/src/drone-simulator/scripts/run_simulation_bugwright.sh .
```

The BugWrigth2 simulation
First simulation

- **Launch the simulator**

```
cd ~/catkin_ws
source devel/setup.bash
./scripts/run_simulation_bugwright.sh --gui
```





The BugWrigth2 simulation
Simulation script

- **Existing options**
  - Set number of each sort of robot
    - ***-nc <arg>*** number of simulated crawlers
    - ***-np <arg>*** number of simulated pioneers
    - ***-nd <arg>*** number of simulated drones
  - Choose the environnement
    - ***-env <arg>*** choose between *bugwright_cargo (default)* and *bugwright_real*
  - With or without the gazebo rendering
    - ***--gui*** launch with redering (none by default)

How to ...
Launch a simple simulation

- **Create a simple simulation for debugging**

```
roslaunch gazebo_models drone_world.launch \
drone:=<name of drone> \
world:=<name of world>

roslaunch gazebo_models multidrone_world.launch \
number:=<number of drones> \
drone:=<name of drone> \
world:=<name of world>
```

- **Name of drones**
  - *crawler, pioneer, intelaero_bugwright, ...*
  - Check ***drone-simulator/packages/gazebo_models/models*** directory
- **Names of worlds**
  - empty, city, bugwright_real, ...
  - Check ***drone-simulator/packages/gazebo_world/worlds*** directory

How to …
Control the drones

- **Using Waypoints**
  - Global position in East-North-Up frame
  - Orientation Front-Left-Up frame relative to East-North-Up frame (only yaw taken in account)
- **Related ROS topic**
  - **name** : */drone_<id>/new_target*
  - **msg** *: geometry_msgs/Pose*

    https://docs.ros.org/kinetic/api/geometry_msgs/html/msg/Pose.html
  - convention *Front-Left-Up*

How to …
Control the crawlers

- **Using a differential drive controller**
  - Forward linear velocity
  - Vertical angular velocity
- **Related ROS topic**
  - **name** : */crawler_<id>/cmd_vel*
  - **msg** *: geometry_msgs/Twist*

    https://docs.ros.org/kinetic/api/geometry_msgs/html/msg/Twist.html
  - convention *Front-Left-Up*

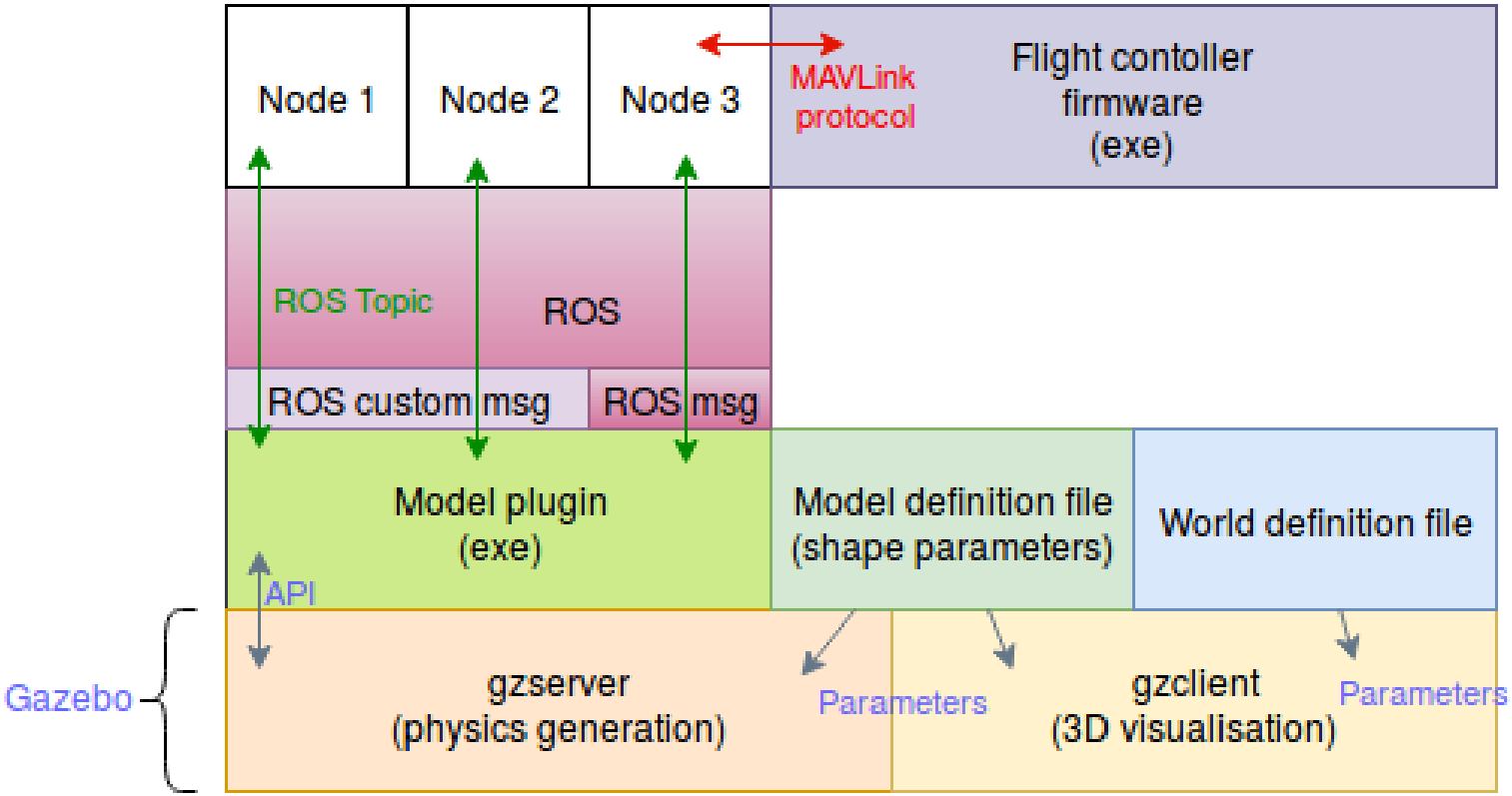
How to …
Control the pioneers (underwater vehicles)

- **No controller implemented at the moment**
  - Forward force & Upward force
  - Pitch moment & Yaw moment
- **Related ROS topic**
  - **name** : */pioneer_<id>/cmd_wrench*
  - **msg** *: geometry_msgs/Wrench*

    https://docs.ros.org/kinetic/api/geometry_msgs/html/msg/Wrench.html
  - convention *Front-Left-Up*

Inside the Simulator
How does it work ?

The Gazebo framework



Inside the Simulator
The packages structures

- gazebo_models
  - models
    - model1
      - model.xacro
      - parts.xacro
      - sensors.xacro
    - model2
    - ...
  - macros
    - parts
      - part1_macro.xacro
      - part2_macro.xacro
      - ...
    - sensors
      - sensor1_macro.xacro
      - sensor2_macro.xacro
      - ...
  - meshes
    - ...
  - launch
    - drone_world.launch
    - spawn_model1.launch
    - ...

The simulator contains 4 main packages

The **gazebo_models** package contains the definition files that describe :
- the existing drones models
- the available mechanical parts
- the embeddable sensors

- gazebo_world
  - worlds
    - world1.world
    - ...
- gazebo_plugins
  - include
    - common.h
    - plugin_part1.h
    - plugin_sensor1.h
    - ...
  - src
    - plugin_part1.cpp
    - plugin_sensor1.cpp
    - ...
- gazebo_plugins_msgs
  - msg
    - msg1.msg
    - msg2.msg

Inside the Simulator
The packages structures

- gazebo_models
  - models
    - model1
      - model.xacro
      - parts.xacro
      - sensors.xacro
    - model2
    - ...
  - macros
    - parts
      - part1_macro.xacro
      - part2_macro.xacro
      - ...
    - sensors
      - sensor1_macro.xacro
      - sensor2_macro.xacro
      - ...
  - meshes
    - ...
  - launch
    - drone_world.launch
    - spawn_model1.launch
    - ...

The project contains 4 main packages

The **gazebo_world** package contains the definition files that describe the 3D environment in which the drone(s) can evolve.

- gazebo_world
  - worlds
    - world1.world
    - ...
- gazebo_plugins
  - include
    - common.h
    - plugin_part1.h
    - plugin_sensor1.h
    - ...
  - src
    - plugin_part1.cpp
    - plugin_sensor1.cpp
    - ...
- gazebo_plugins_msgs
  - msg
    - msg1.msg
    - msg2.msg

Inside the Simulator
The packages structures

- gazebo_models
  - models
    - model1
      - model.xacro
      - parts.xacro
      - sensors.xacro
    - model2
    - ...
  - macros
    - parts
      - part1_macro.xacro
      - part2_macro.xacro
      - ...
    - sensors
      - sensor1_macro.xacro
      - sensor2_macro.xacro
      - ...
  - meshes
    - ...
  - launch
    - drone_world.launch
    - spawn_model1.launch
    - ...

The project contains 4 main packages

The **gazebo_plugins** package contains the C++ source files which communicate with the Gazebo physic engine in order to simulate the sensors or mechanical parts of the **gazebo_models** package.

These source files send informations or receive commands trough ROS topics.

- gazebo_world
  - worlds
    - world1.world
    - ...
- gazebo_plugins
  - include
    - common.h
    - plugin_part1.h
    - plugin_sensor1.h
    - ...
  - src
    - plugin_part1.cpp
    - plugin_sensor1.cpp
    - ...
- gazebo_plugins_msgs
  - msg
    - msg1.msg
    - msg2.msg

Inside the Simulator
The packages structures

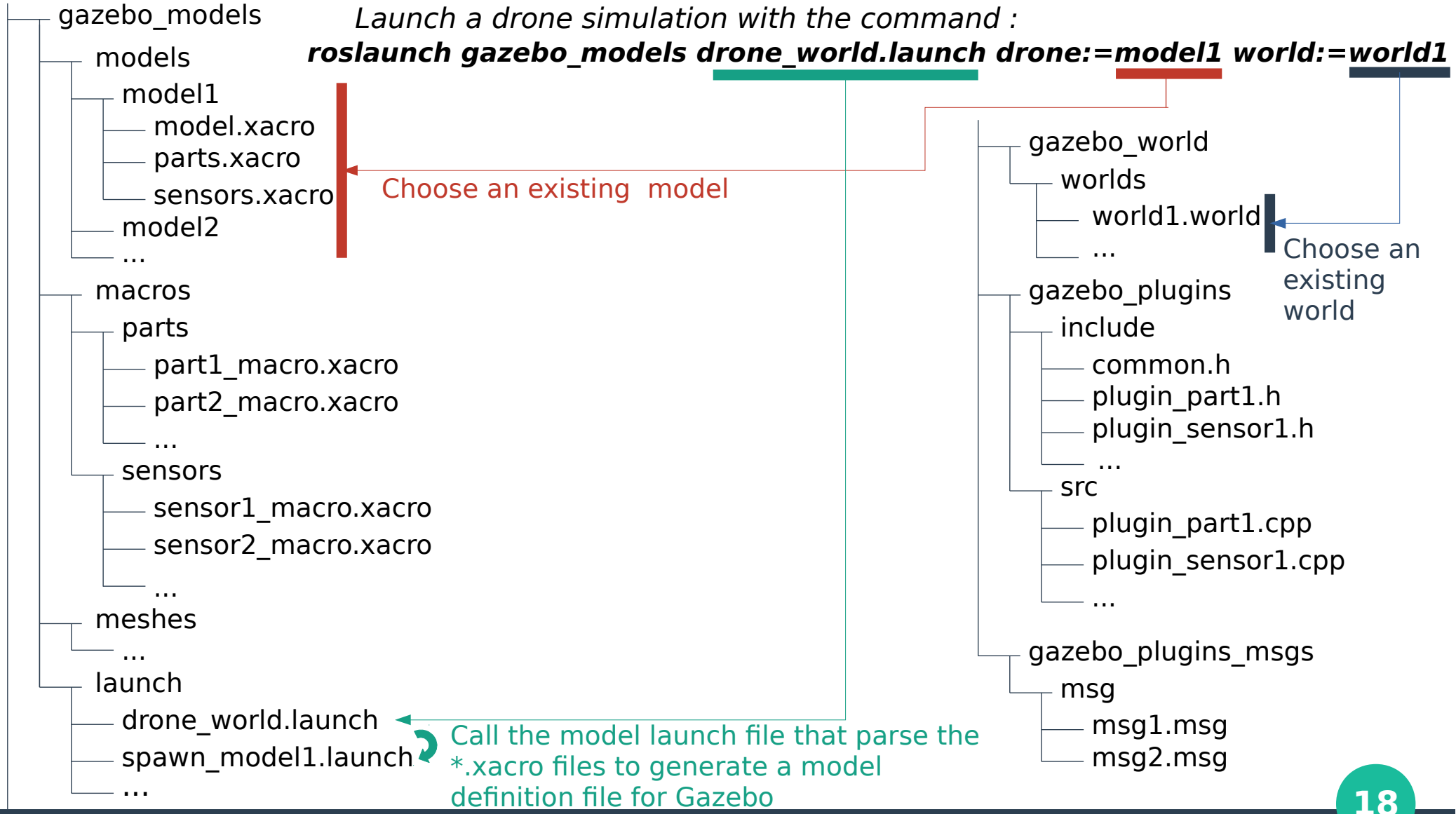
- gazebo_models
  - models
    - model1
      - model.xacro
      - parts.xacro
      - sensors.xacro
    - model2
    - ...
  - macros
    - parts
      - part1_macro.xacro
      - part2_macro.xacro
      - ...
    - sensors
      - sensor1_macro.xacro
      - sensor2_macro.xacro
      - ...
  - meshes
    - ...
  - launch
    - drone_world.launch
    - spawn_model1.launch
    - ...

The project contains 4 main packages

- gazebo_world
  - worlds
    - world1.world
    - ...
- gazebo_plugins
  - include
    - common.h
    - plugin_part1.h
    - plugin_sensor1.h
    - ...
  - src
    - plugin_part1.cpp
    - plugin_sensor1.cpp
    - ...
- gazebo_plugins_msgs
  - msg
    - msg1.msg
    - msg2.msg

The **gazebo_plugins_msgs** package contains ROS messages that have been built for the **gazebo_plugins** package.

Inside the Simulator
Launching process



Inside the Simulator
Creating a new drone model

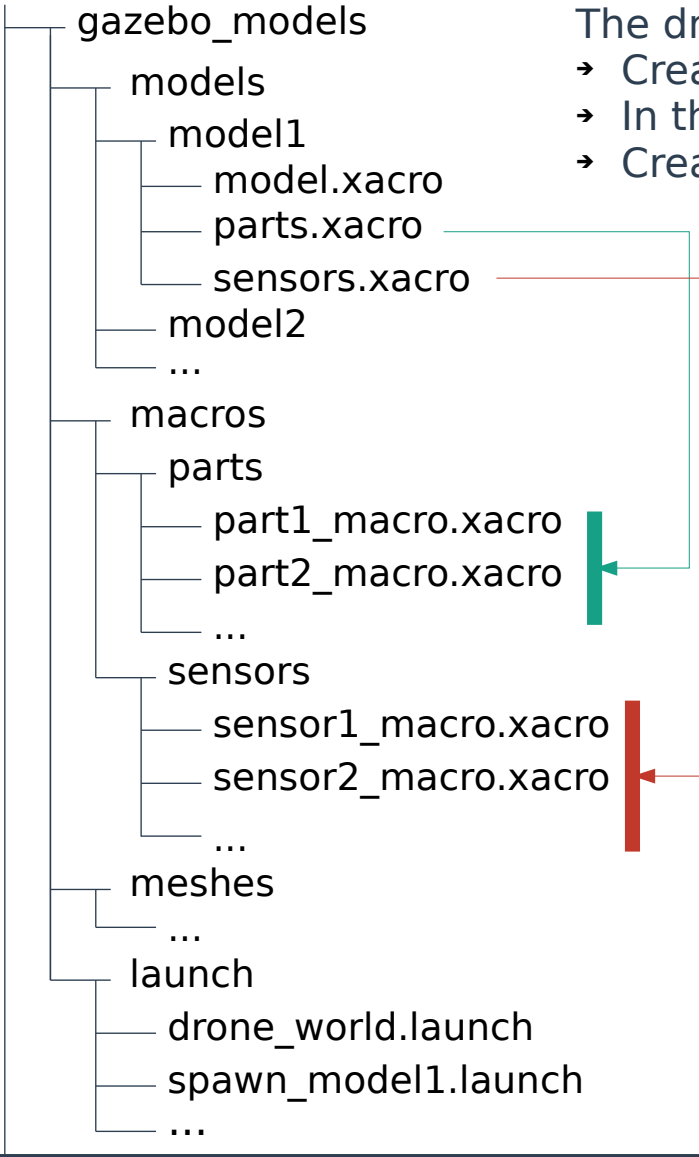
- gazebo_models
  - models
    - model1
      - model.xacro
      - parts.xacro
      - sensors.xacro
    - model2
    - ...
  - macros
    - parts
      - part1_macro.xacro
      - part2_macro.xacro
      - ...
    - sensors
      - sensor1_macro.xacro
      - sensor2_macro.xacro
      - ...
  - meshes
    - ...
  - launch
    - drone_world.launch
    - spawn_model1.launch
    - ...

The drone creation is similar to LEGO bricks :

- Create a model folder in gazebo_models/models
- In this folder create a model.xacro, parts.xacro and sensors.xacro files
- Create a spawn_model.launch file in gazebo_models/launch

The ***model.xacro*** includes the ***parts.xacro*** and ***sensors.xacro*** files

Inside the Simulator
Creating a new drone model



The drone creation is similar to LEGO bricks :

- Create a model folder in gazebo_models/models
- In this folder create a model.xacro, parts.xacro and sensors.xacro files
- Create a spawn_model.launch file in gazebo_models/launch

The ***parts.xacro*** instantiate the model's mechanical parts (body, wing, rotors, servos, ...)

The ***sensors.xacro*** instantiate the model's sensors (IMU, magnetometer, GPS, ...)