Simulation User Manual

1. Set Up

For this project, a dual boot version of ubuntu 18.04 has been used. So, the first thing to do is to download the version of Ubuntu to use, create an account and download the antivirus offered for security reasons (Ubuntu Livepatch Service).

After this, it is important to connect to the Eduroam WIFI, so download the configurator at the following link and choose Universidad Politécnica de Madrid:

https://cat.eduroam.org/

To install and configure it, open the terminal and type the following commands in the same folder where the file "python3 eduroam-linux-UPdM.py" is located:

>> sudo apt-get update

>> sudo apt-get install python3.6

Once this is done, you can proceed to download both ros2 foxy and Ignition citadel.

- Installing Ros2 Foxy: https://docs.ros.org/en/foxy/Installation/Ubuntu-Install-Debians.html
- Configuring ROS2 environment: https://docs.ros.org/en/foxy/Tutorials/Configuring-ROS2-Environment.html
- Configuring Colcon with ROS2 tutorials: https://docs.ros.org/en/foxy/Tutorials/Colcon-Tutorial.html
- Installing Ignition Citadel: https://ignitionrobotics.org/docs/citadel/install_ubuntu
- Installing ros_ign: in terminal type >> sudo apt install ros-foxy-ros-ign.

The files of the models and the launch can be found on the team's GitHub. To make a clone of the GitHub, you must connect with SSH and not with https. https://github.com/Ingenia-SE/TrashE

In the GitHub, the folder that contains all the simulation files is the folder p3at_sim/src/ign_simulation which has a division into subfolders. The most important of them are the following:

- model/p3at: it contains the files of the different components and sensors that make up the pioneer3 AT. In this subfolder you can also find the model.sdf which is the main file where the different components are joined, and plugins are added.
- worlds: contains the sdf file for each of the worlds used.
- launch: contains the python executable to start the simulation in each of the worlds.

To create a directory and clone the TrashE GitHub repository, type the following commands in the terminal:

>> mkdir -p ~/TrashE/src

>> cd TrashE/src

>> git clone git@github.com:Ingenia-SE/TrashE.git

>> cd ..

>> sudo apt install python3-colcon-common-extensions (only once to install colcon)

>> colcon build –symlink-install

>> . install/setup.bash

2. Launch

It is important to know that every time a change is made to a file, a colcon build must be performed. To launch the simulation with different worlds, the following commands must be typed in the terminal:

>> cd TrashE

>> source install/setup.bash (this must be done every time a new terminal is opened, in order to source)

>> ros2 launch ign_simulation p3at_campus.launch.py (this is an example with campus world. If you want to launch another world, exchange “p3at_campus.launch.py” for “name_of_the_file.launch.py”. With this command, Ignition will launch with the chosen world, and it is important to click on play button located in in the bottom left-hand corner).

To understand the files developed or to make changes by adding new functionality or making new files, the following Ignition tutorials are recommended:

https://ignitionrobotics.org/docs/citadel/building_robot

3. Communication

Once the simulation has been launched, the model is ready to receive messages (subscribe to a topic) or send messages (publish a topic). This allows it to communicate with the robot, reading information from its sensors or controlling its actuators. This can be done through the Ubuntu terminal and interacting with the Ignition plugins or with a plugin called ROS bridge, which allows communication with the robot through ROS nodes.

To send messages to the robot, we are going to focus specifically on the cmd_vel topic by which a linear and angular velocity is sent to the robot through the terminal. This works in the same way for other topics. In a new terminal (ctrl+shift+t) type the following commands:

>> source install/setup.bash

>> ign topic -t "/cmd_vel" -m ignition.msgs.Twist -p "linear: {x: 0.5}, angular: {z: 0.05}"

To read information from the robot's sensors, we will type the following commands in a new terminal:

>> source install/setup.bash

>> ros2 topic list (provides a list of all the topics currently active in the system)

>> ros2 topic echo <topic name> (allows seeing data that is being published on a topic)

>> ros2 topic echo /imu (provides the lecture of the inertial accelerations of the robot provided by the IMU)

>> ros2 topic echo /lidar (provides the lecture of lidar sensor which consists of a laser that is sent and travels to an obstacle and through the bounce of the wave, the obstacle is detected)

>> ros2 topic echo /model/p3at_ugv/odometry (provides information on the linear and angular velocity of the robot as well as its position at all times).

To understand the files realised or to make changes by adding new functionality or making new files, the following Ignition tutorials are recommended:

https://ignitionrobotics.org/docs/citadel/moving_robot

https://ignitionrobotics.org/docs/citadel/sdf_worlds

https://ignitionrobotics.org/docs/citadel/sensors

https://ignitionrobotics.org/docs/citadel/ros2_integration

4. Simulation Description File SDF

SDF files are used to store databases in a structured format and are commonly used to transfer data between various programs. It is in this format in which the model has been implemented using XML code. The structure of the SDF model is made up of:

- Links of each different parts (visual aspect of the components has been taken from the images shown above and as collisions boxes of different sizes depending on the component have been considered):
    I. Chassis (including IMU as a sensor).
    II. Four wheels.
    III. Lidar.
    IV. Kinect
- Joints between components.
- Plugins.