

Lesson 4 Lab sheet- Intelligent mobile robotics

Install and Run Gazebo simulation for turtlebot3

Aims

- Install prerequisite packages for TurtleBot3 simulation
- Install Turtlebot3 Simulation Package
- Operate TurtleBot3 using Keyboard
- Write python code to show odometry data

Suppose that we have ROS1 installed on our machine, and we want to install TurtleBot3 simulation. The **TurtleBot3 Simulation Package** requires `turtlebot3` and `turtlebot3_msgs` packages as prerequisite. Open a project in ROS development studio and do the following steps:

1. Install turtlebot3 and turtlebot3_msgs packages :

Run the following commands to get and install the two prerequisite packages (Note that you should be in following folder: `~/catkin_ws/src`):

```
$cd ~/catkin_ws/src
$git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3\_msgs.git
$git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3.git
```

Go back to '`~/catkin_ws`' folder and compile the system using():

```
$catkin_make
```

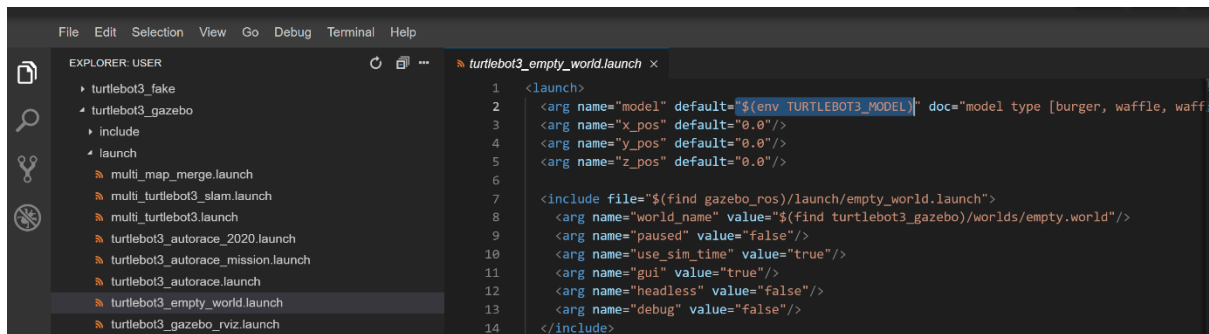
2. Install and run Turtlebot3 Simulation Package

By running the following command, install Turtlebot3 simulation package¹:

```
$ cd ~/catkin_ws/src/
$ git clone -b noetic-devel https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
$ cd ~/catkin_ws && catkin_make
```

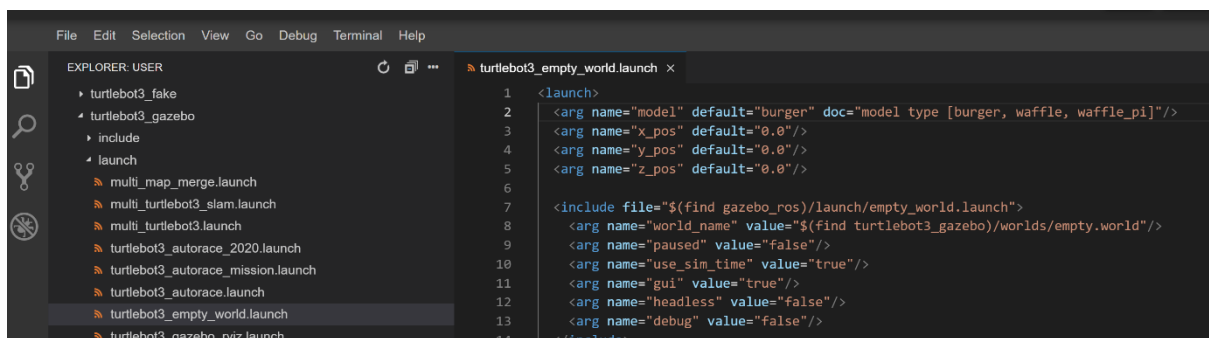
In your computer you need to run the following comment to change the robot name as an environment variable: '`$export TURTLEBOT3_MODEL=burger`'. Instead in ROS Development studio change '`default="$(env TURTLEBOT3_MODEL)"`' in the following launch file:

¹ Note that if you want to install on your personal pc (not in Ros development studio) you need run the following comment after `catkin_make`: '`echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc`', see [] for mor information.



```
1 <launch>
2   <arg name="model" default="$(env TURTLEBOT3_MODEL)" doc="model type [burger, waffle, waffle_pi]" />
3   <arg name="x_pos" default="0.0" />
4   <arg name="y_pos" default="0.0" />
5   <arg name="z_pos" default="0.0" />
6
7   <include file="$(find gazebo_ros)/launch/empty_world.launch">
8     <arg name="world_name" value="$(find turtlebot3_gazebo)/worlds/empty.world" />
9     <arg name="paused" value="false" />
10    <arg name="use_sim_time" value="true" />
11    <arg name="gui" value="true" />
12    <arg name="headless" value="false" />
13    <arg name="debug" value="false" />
14  </include>
```

Change 'default="\$(env TURTLEBOT3_MODEL)"' to default="burger" as shown in the following figure:

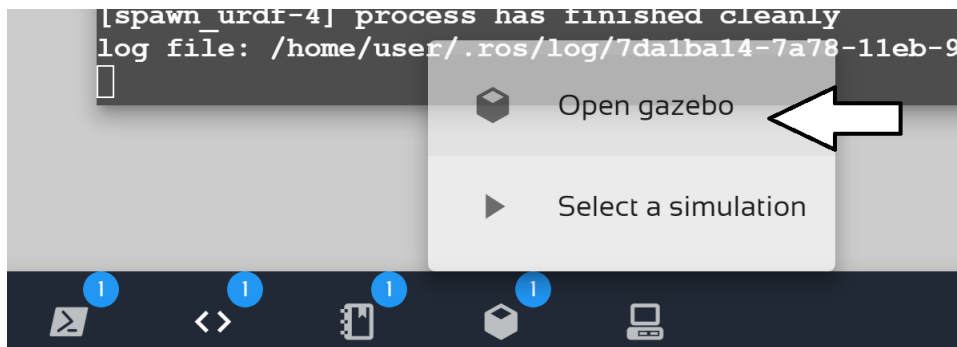


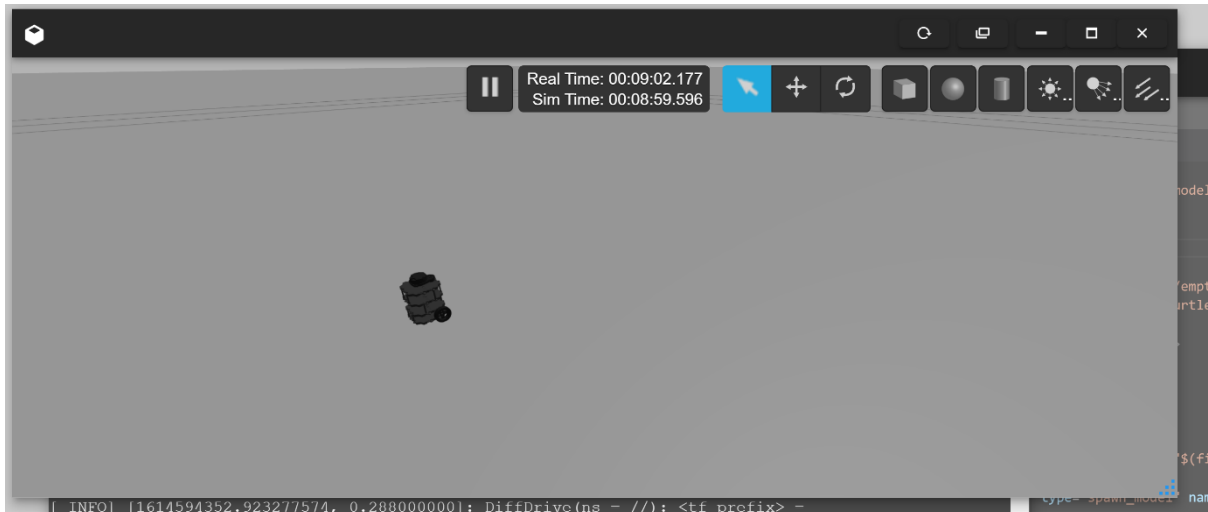
```
1 <launch>
2   <arg name="model" default="burger" doc="model type [burger, waffle, waffle_pi]" />
3   <arg name="x_pos" default="0.0" />
4   <arg name="y_pos" default="0.0" />
5   <arg name="z_pos" default="0.0" />
6
7   <include file="$(find gazebo_ros)/launch/empty_world.launch">
8     <arg name="world_name" value="$(find turtlebot3_gazebo)/worlds/empty.world" />
9     <arg name="paused" value="false" />
10    <arg name="use_sim_time" value="true" />
11    <arg name="gui" value="true" />
12    <arg name="headless" value="false" />
13    <arg name="debug" value="false" />
14  </include>
```

Then launch the launch file, turtlebot3_empty_world.launch, to see the robot in an empty file:

```
$ roslaunch turtlebot3_gazebo turtlebot3_empty_world.launch
```

Open Gazebo to see the robot in an empty world:





Note that if you want to run TurtleBot3 in another world for instance TurtleBot3 World, you need to kill the previous launch by, `ctrl+c` and run the following command:

```
$ export TURTLEBOT3_MODEL=waffle
$ roslaunch turtlebot3_gazebo turtlebot3_world.launch
```

To run TurtleBot3 House, kill the previous `roslaunch` and run the following command:

```
$ export TURTLEBOT3_MODEL=waffle_pi
$ roslaunch turtlebot3_gazebo turtlebot3_house.launch
```

3. Operate TurtleBot3 using Keyboard

In a new shell window run the following command to teleoperate the TurtleBot3 with the keyboard (note that it is supposed that you have launched the empty world so the environment variable should be set as: `TURTLEBOT3_MODEL=burger`):

```
$ export TURTLEBOT3_MODEL=burger
```

```
$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
```

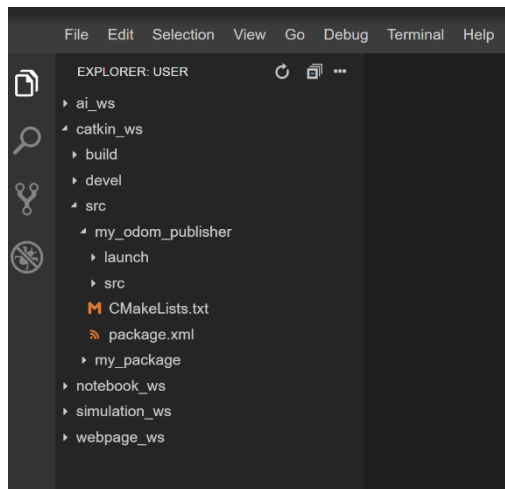
4. Write a launch file for a package refer to a python code

- Create a package called 'my_odom_publisher':

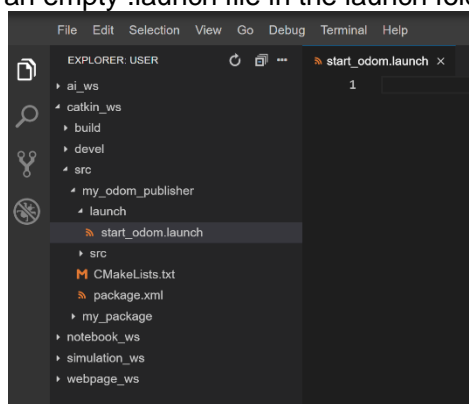
```
user:~$ cd catkin_ws/src
user:~/catkin_ws/src$ catkin_create_pkg my_odom_publisher rospy
```

- Create a 'launch' folder in the newly created folder, i.e. `my_odom_publisher`:

```
$ cd my_odom_publisher
$ mkdir launch
```



- Create an empty .launch file in the launch folder. Call the launch file as, 'start_odom.launch'.



- Define a node in the lunch file refer to a python code called 'odom.py'. The node should be in the previously created package and its type is .py file and it show output on the screen:

```
1 - <launch>
2   <node name="odom_pub" pkg="my_odom_publisher" type="odom.py" output="screen" />|
3 </launch>
```

5. Write python code called 'odom.py' to get data from Gazebo simulator and publish the odometry in a topic

-use 'rostopic list' to see the list of active topics and find the topic contain odometry data.

- Create a python empty file in the src folder in the newly created package, and call it odom.py.
- Use the code in the previous week to write a subscriber node to subscribe to the odom topic and print the message in the command window,
- Lonch your pakage using the lunch file:
\$roslaunch my_odom_publisher start_odom.launch

See the list of activated node/topic .

- write another python code to create a new node that subscribes to the odom topic, read the massage in the topic. Then the node take the odometry data and publishes

the data in a new topics called 'my_odom'.

- print apropiat message on terminals. And check the list of active nodes and topics.

- Run the code your 'paython <The-name-of- your- code.py>'

- Add the node in the previous lunch file

- Launch the created package,i.e. my_odom_publisher:

```
$roslaunch my_odom_publisher start_odom.launch
```

Then use 'rostopic list' to find your newly created topic, /my_odom. Use the following command to see the content of the message in this topic:

```
$rostopic echo /my_odom
```

Reference:

[1] How to have a Turtlebot 3 simulation in Gazebo with ROS running in 5 mins

<https://www.youtube.com/watch?v=y54BWXLnJDQ&t=4s>

[2] Installation of Turtlebot3 Gazebo simulation

<https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/>

[2] Gazebo installation,

<https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/#pc-setup>
