Creating A Launch File And Setup

For the complete execution of task 1B, there are 2 python files that need to be run simultaneously. Launch files makes it easier by running them using a single command. The 2 files are `controller.py` and `service_node.py`.

First, lets create a launch file. Follow the steps given below:

- Navigate to the launch folder and create a file named `hb_task1b.launch.py`.

  *The **launch file** name needs to end with launch.py to be recognized and autocompleted by ros2 launch. Your launch file should define the `generate_launch_description()` function which returns a launch. `LaunchDescription()` to be used by the ros2 launch verb.*

- Change this python file into an executable using the following command:

```
chmod +x hb_task1b.launch.py
```

Launch File Structure

```
from launch import LaunchDescription
from launch_ros.actions import Node
def generate_launch_description():
    ld = LaunchDescription()
    controller_node = Node(
        package="<your_package_name>",          # Enter the name of your ROS2 package
        executable="<your_executable_name>",    # Enter the name of your executable
    )
    service_node = Node(
        package="<your_package_name>",          # Enter the name of your ROS2 package
        executable="<your_executable_name>",    # Enter the name of your executable
    )
```

Installing Dependencies

There are many dependencies that are needed to be installed for seamlessly running our **Gazebo simulator**.

Run the following commands in terminal:

```
sudo apt install ros-humble-tf-transformations
sudo pip3 install transforms3d
sudo apt install -f ros-humble-gazebo-ros-pkgs
```

Given below should be the structure of your package.

```
Package (hb_task_1b)
    - hb_task_1b
        - __init__.py
    - launch
```

```
        - gazebo.launch.py
        - hb_task1b.launch.py
    - urdf
        - hb_bot.urdf.xacro
        - materials.xacro
    - world
        - gazebo.world
    - scripts
        - controller.py
        - service_node.py
    - meshes
        - base.dae
        - wheel.stl
        - 17eyantra_logo_large e.png
```

Now, download the given packages and create the directory structure as mentioned above with the given downloadables.

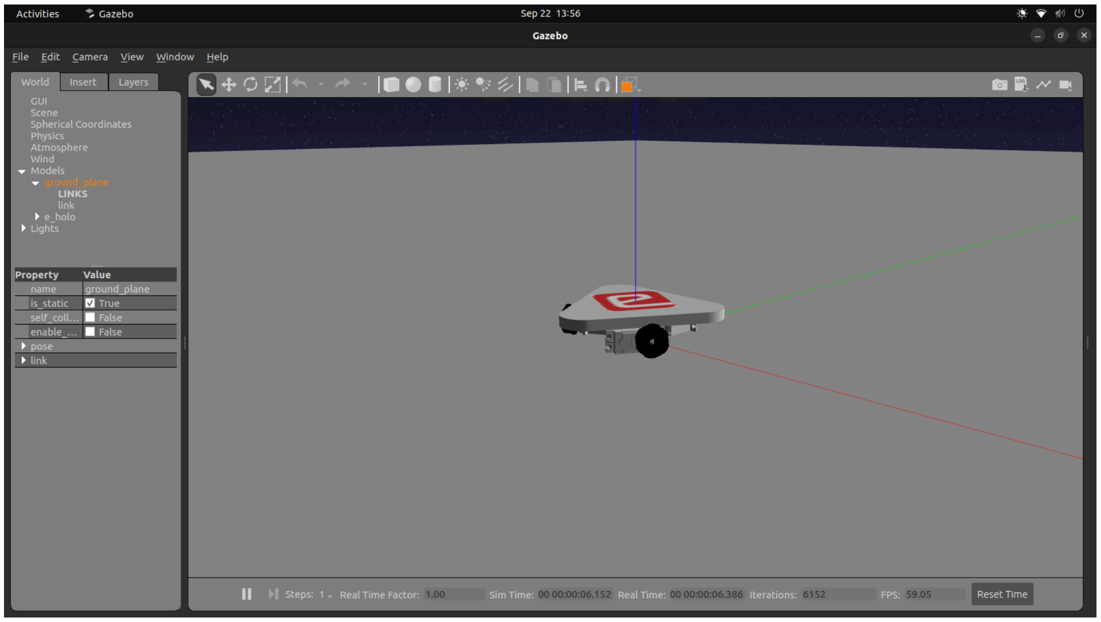
Once your directory structure is created, navigate to the workspace (`hb_task1b_ws`) folder, build it using `colcon build` command and source it using `source install/setup.bash` command.

> *Make sure your build and source your workspace everytime you make changes in it or open a new terminal.*

Now, launch gazebo:

```
ros2 launch hb_task_1b gazebo.launch.py
```

Gazebo will launch and you will see the output similar to the image below.

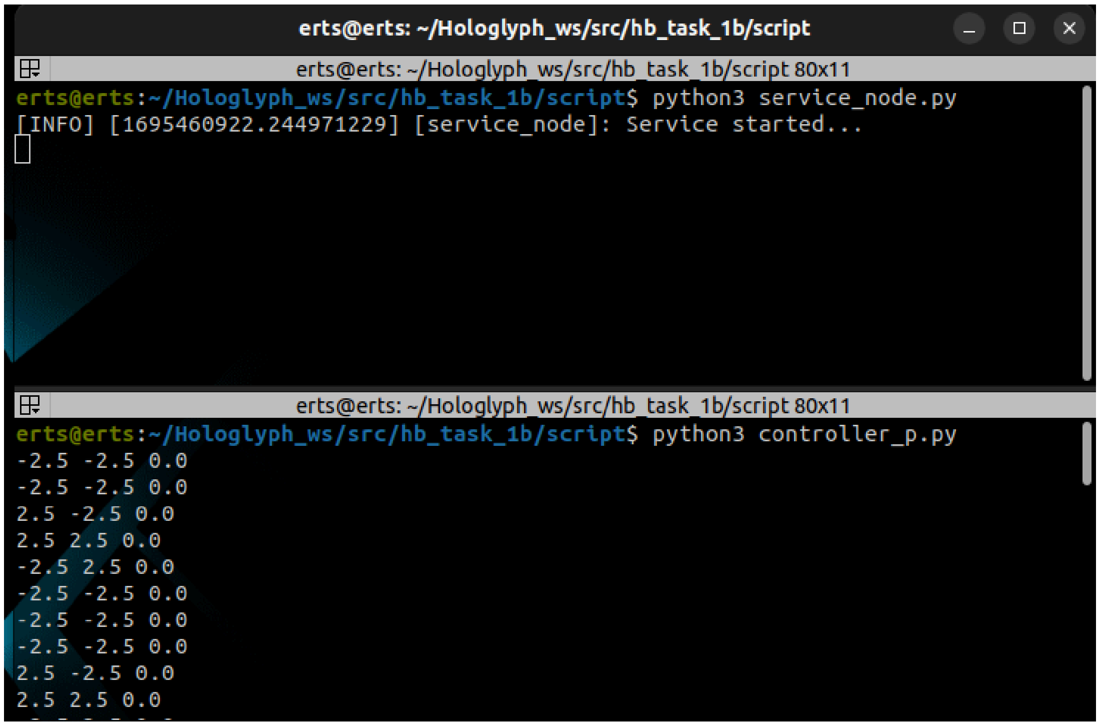


Expected Output

Now, open another terminal (don't forget to source it) and run the following command for launching the controller and service nodes.

```
ros2 launch hb_task_1b hb_task1b.launch.py
```

The expected output is shown below.


erts@erts: ~/Hologlyph_ws/src/hb_task_1b/script
erts@erts: ~/Hologlyph_ws/src/hb_task_1b/script 80x11
erts@erts: ~/Hologlyph_ws/src/hb_task_1b/script 80x11

```
erts@erts:~/Hologlyph_ws/src/hb_task_1b/script$ python3 service_node.py
[INFO] [1695460922.244971229] [service_node]: Service started...
```

```
erts@erts:~/Hologlyph_ws/src/hb_task_1b/script$ python3 controller_p.py
-2.5 -2.5 0.0
-2.5 -2.5 0.0
2.5 -2.5 0.0
2.5 2.5 0.0
-2.5 2.5 0.0
-2.5 -2.5 0.0
-2.5 -2.5 0.0
-2.5 -2.5 0.0
2.5 -2.5 0.0
2.5 2.5 0.0
```

Expected Output after launching `hb_task1b.launch.py`

If both the launches are successful, congratulations! the setup for Task 1B is complete!! You can now proceed with go2goal...