

LAB5 Deliverables

It contains everything for lab5

The specified deliverables are highlighted with large red squares

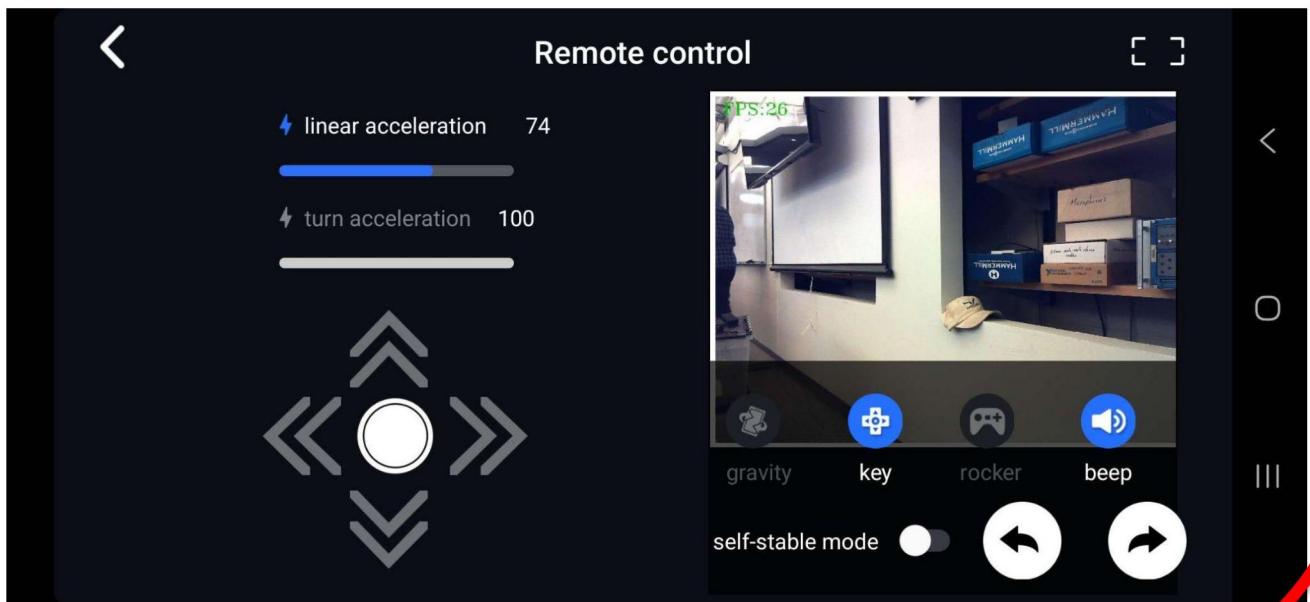
Task1

Testing with the App

Action4:



Action5:

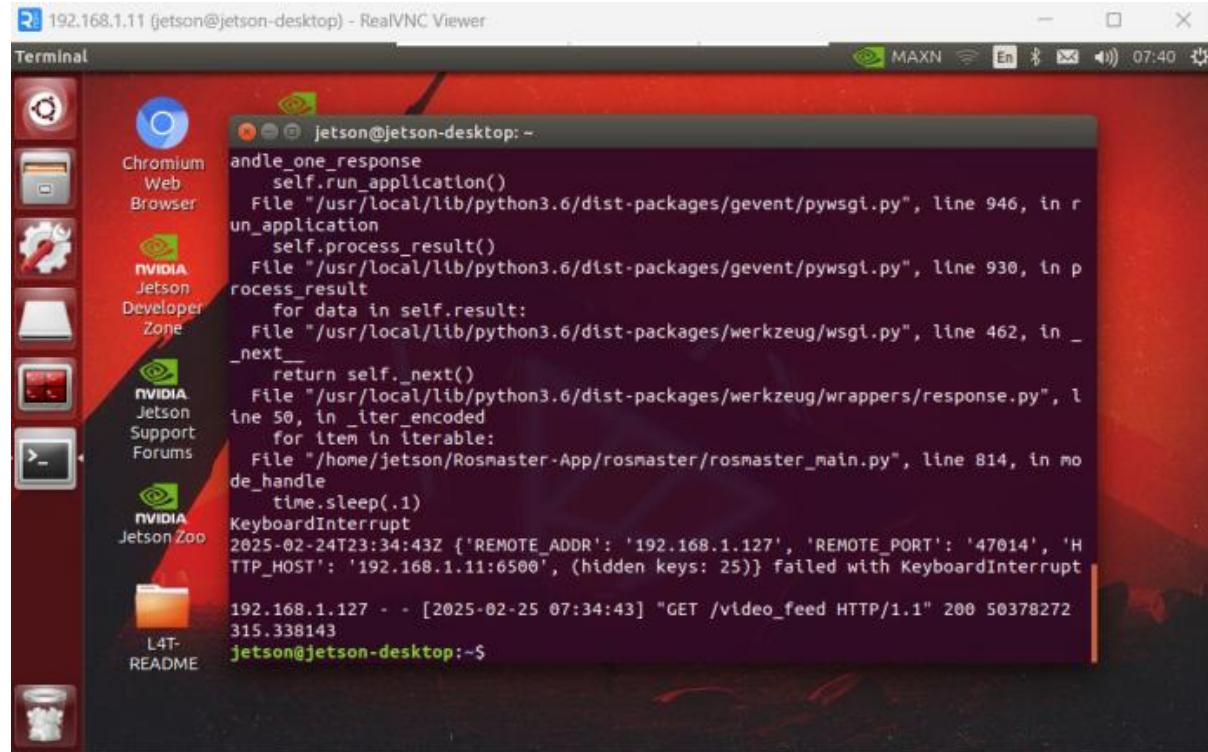


REQUIRED Question: Based on all of this testing, what are some pros and cons of using omni-directional wheels?

Omnidirectional wheels can move forward, backward, sideways, and rotate in place. They are used in places with limited space or frequent direction changes. But the contact area with the ground is small, so on uneven or smooth surfaces, they may slip or shake. When the load is large or the ground is not flat, energy use increases, and load capacity is lower than traditional wheels.

Using Docker in VNC Viewer

Action2



Action4 and Action7

```
root@jetson-desktop: /home/jetson/codes/lab5
root@jetson-desktop: /  x root@jetson-desktop: /h...  x root@jetson-desktop: /h...  x [+] ▾

MY_IP: 129.161.116.73
-----
jetson@jetson-desktop:~$ ./exec_docker.sh
-----
ROS_DOMAIN_ID: 1
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
-----
root@jetson-desktop:/# mkdir /home/jetson/codes/lab5
mkdir: cannot create directory '/home/jetson/codes/lab5': No such file or directory
root@jetson-desktop:/# mkdir -p /home/jetson/codes/lab5
root@jetson-desktop:/# cd /home/jetson/codes/lab5
root@jetson-desktop:/home/jetson/codes/lab5# ros2 run yahboomcar Bringup Mcnamu_driver_X3
Rosmaster Serial Opened! Baudrate=115200
X3
imu_link
1.0
1.0
5.0
-----create receive threading-----
```

Action8

```
root@jetson-desktop: /home/jetson/codes/lab5
root@jetson-desktop: /  x root@jetson-desktop: /h...  x root@jetson-desktop: /h...  x [+] ▾

my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
-----
root@jetson-desktop:/# cd home/jetson/codes/labs
root@jetson-desktop:/home/jetson/codes/labs# ros2 run yahboomcar_ctrl yahboom_keyboard

Control Your SLAM-Bot!
-----
Moving around:
  u   i   o
  j   k   l
  m   ,   .

q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
t/T : x and y speed switch
s/S : stop keyboard control
space key, k : force stop
anything else : stop smoothly

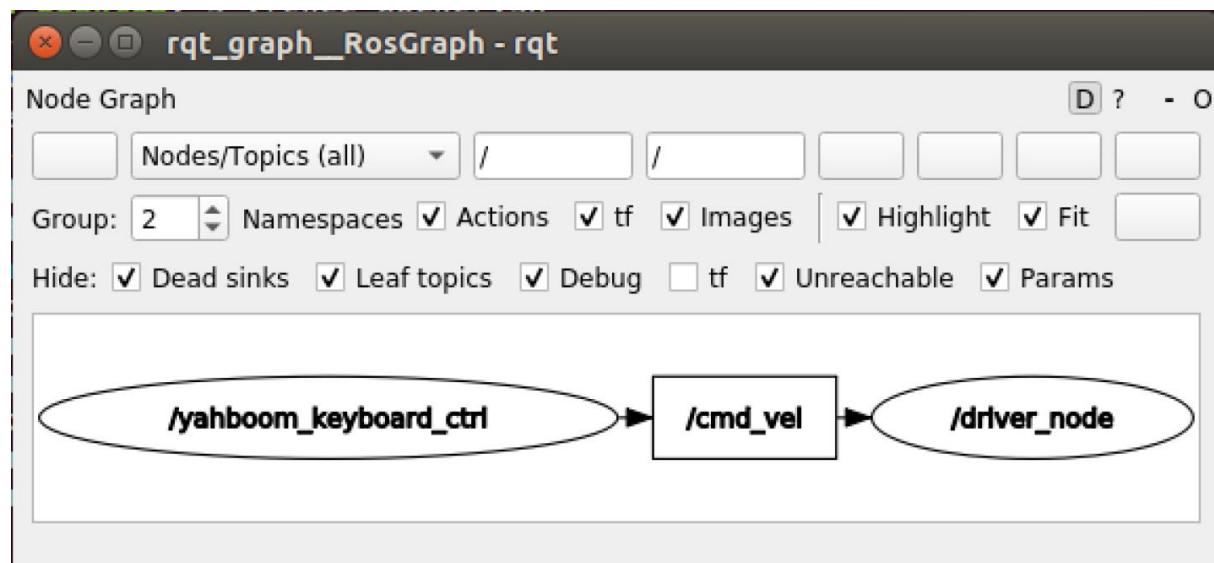
CTRL-C to quit

currently:      speed 0.2      turn 1.0
```

REQUIRED Question: Based on the previous step, what is the coordinate system for the robot?

The robot uses a right-hand coordinate system. In this system, the X-axis represents the forward direction, the Y-axis represents the lateral movement, and the Z-axis points upward. The Z-axis usually indicates the rotation axis.

Action11

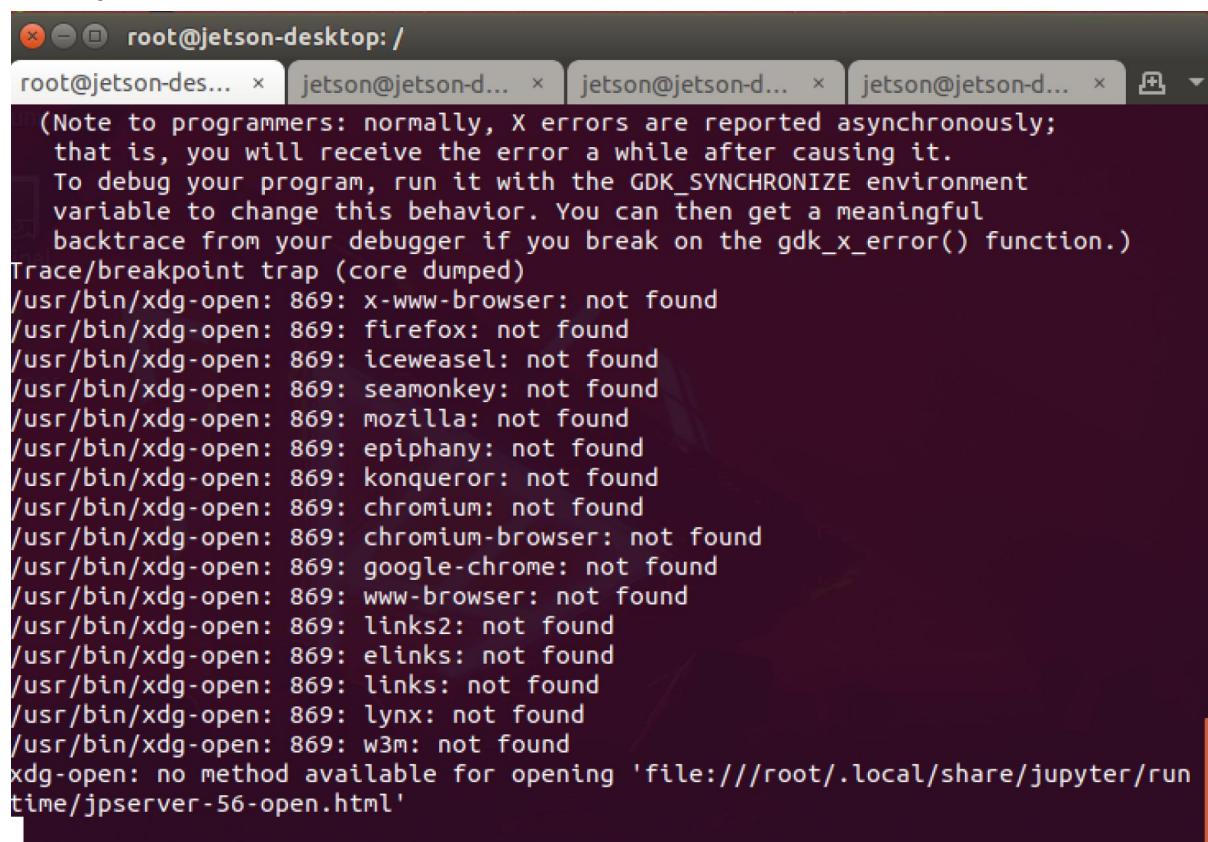


Action12

```
jetson@jetson-desktop: ~
jetson@jetson-d... x jetson@jetson-d... x jetson@jetson-d... x jetson@jetson-d... x
Rosmaster Serial Opened! Baudrate=115200
-----create receive threading-----
Battery voltage: 12.2
-----
MY_IP: 192.168.1.11
-----
jetson@jetson-desktop:~$ source run_docker.sh
access control disabled, clients can connect from any host
WARNING: Published ports are discarded when using host network mode
-----
ROS_DOMAIN_ID: 1
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
-----
root@jetson-desktop:/# ./exec_docker.sh
bash: ./exec_docker.sh: No such file or directory
root@jetson-desktop:/# exit
exit
jetson@jetson-desktop:~$
```

Using Jupyter Lab

Action3



The screenshot shows a terminal window with a dark background and light-colored text. At the top, there are four tabs labeled 'root@jetson-des...', 'jetson@jetson-d...', 'jetson@jetson-d...', and 'jetson@jetson-d...'. The main area of the terminal displays a core dump trace from a 'Trace/breakpoint trap (core dumped)' event. The trace lists various browser executables that were not found, such as 'x-www-browser', 'firefox', 'iceweasel', 'seamonkey', 'mozilla', 'epiphany', 'konqueror', 'chromium', 'chromium-browser', 'google-chrome', 'www-browser', 'links2', 'elinks', 'links', 'lynx', and 'w3m'. The final line of the trace is 'xdg-open: no method available for opening 'file:///root/.local/share/jupyter/run-time/jpserver-56-open.html''. The terminal window has a standard Linux-style title bar with icons for close, minimize, and maximize.

```
(Note to programmers: normally, X errors are reported asynchronously;
that is, you will receive the error a while after causing it.
To debug your program, run it with the GDK_SYNCHRONIZE environment
variable to change this behavior. You can then get a meaningful
backtrace from your debugger if you break on the gdk_x_error() function.)
Trace/breakpoint trap (core dumped)
/usr/bin/xdg-open: 869: x-www-browser: not found
/usr/bin/xdg-open: 869: firefox: not found
/usr/bin/xdg-open: 869: iceweasel: not found
/usr/bin/xdg-open: 869: seamonkey: not found
/usr/bin/xdg-open: 869: mozilla: not found
/usr/bin/xdg-open: 869: epiphany: not found
/usr/bin/xdg-open: 869: konqueror: not found
/usr/bin/xdg-open: 869: chromium: not found
/usr/bin/xdg-open: 869: chromium-browser: not found
/usr/bin/xdg-open: 869: google-chrome: not found
/usr/bin/xdg-open: 869: www-browser: not found
/usr/bin/xdg-open: 869: links2: not found
/usr/bin/xdg-open: 869: elinks: not found
/usr/bin/xdg-open: 869: links: not found
/usr/bin/xdg-open: 869: lynx: not found
/usr/bin/xdg-open: 869: w3m: not found
xdg-open: no method available for opening 'file:///root/.local/share/jupyter/run-time/jpserver-56-open.html'
```

Action4

The screenshot shows a JupyterLab interface running in a Chromium browser window. The left sidebar contains a file browser with a list of files in the 'Rosmaster / Sample /' directory. The main area is a code editor with tabs for '8.car_motion.ipynb' and '7.motor.ipynb'. The '8.car_motion.ipynb' tab is active, displaying Python code to initialize a Rosmaster object and start receiving data. The output pane shows the message 'Rosmaster Serial Opened! Baudrate=115200'. The '7.motor.ipynb' tab is also visible in the background.

```
[1]: #!/usr/bin/env python3
#coding=utf-8
import time
from Rosmaster_Lib import Rosmaster
from ipywidgets import interact
import ipywidgets as widgets

[2]: # 创建Rosmaster对象 bot Create the Rosmaster object
bot = Rosmaster()

[3]: # 启动接收数据, 只能启动一次, 所有读取数据的功能都是从这个线程启动的
# Start to receive data, can only start once
bot.create_receive_threading()
```

Action5

motor

The screenshot shows a JupyterLab interface running in a Chromium browser window. The code editor has several cells. Cells [1] and [2] are identical to those in the previous screenshot. Cell [4] contains code to control four motors (M1-M4) using sliders. Cell [5] contains code to stop the motors. Cell [6] is a note about deleting the Rosmaster object after the program is complete. The output pane shows the message 'Rosmaster Serial Opened! Baudrate=115200'.

```
[1]: #!/usr/bin/env python3
#coding=utf-8
import time
from Rosmaster_Lib import Rosmaster
from ipywidgets import interact
import ipywidgets as widgets

[2]: # 创建Rosmaster对象 bot Create the Rosmaster object
bot = Rosmaster()

[3]: # 控制电机运动 Control motor movement
def run_motor(M1, M2, M3, M4):
    bot.set_motor(M1, M2, M3, M4)
    return M1, M2, M3, M4

[4]: # 创建四个滑块来控制电机 Create four sliders to control the motor
interact(run_motor, \
         M1=widgets.IntSlider(min=-100,max=100,step=1,value=0), \
         M2=widgets.IntSlider(min=-100,max=100,step=1,value=0), \
         M3=widgets.IntSlider(min=-100,max=100,step=1,value=0), \
         M4=widgets.IntSlider(min=-100,max=100,step=1,value=0));

Error displaying widget: model not found

[5]: # 停止运动 stop motion
bot.set_motor(0, 0, 0, 0)

[6]: # 程序结束后请删除对象, 避免在其他程序中使用Rosmaster库造成冲突
# After the program is complete, delete the object to avoid conflicts caused by using the library in other programs
del bot

serial Close!
```

car motion

```
[1]: #!/usr/bin/env python3
#coding=utf-8
import time
from Rosmaster_Lib import Rosmaster
from ipywidgets import interact
import ipywidgets as widgets

[2]: # 创建Rosmaster对象 bot Create the Rosmaster object bot
bot = Rosmaster()

Rosmaster Serial Opened! Baudrate=115200

[3]: # 启动接收数据, 只能启动一次, 所有读取数据的功能都是基于此方法
# Start to receive data, can only start once, all read data function is based on this method
bot.create_receive_threading()
```

自动上报数据开关

Switch of automatic data reporting

```
[4]: # 开启自动发送数据
# enable=True, 底层扩展会每隔40毫秒发送一次数据, enable=False, 则不发送。
# forever=True永久保存, =False临时作用。
# Enable automatic data sending
# If enable=True, the underlying expansion module sends data every 40 milliseconds. If enable=False, the port is not sent.
# Forever =True for permanent, =False for temporary
enable = True
bot.set_auto_report_state(enable, forever=False)

[5]: # 关闭自动发送数据
# enable=True, 底层扩展会每隔40毫秒发送一次数据, enable=False, 则不发送。
# forever=True永久保存, =False临时作用。
# Disable automatic data sending
# If enable=True, the underlying expansion module sends data every 40 milliseconds. If enable=False, the port is not sent.
# Forever =True for permanent, =False for temporary
enable = False
bot.set_auto_report_state(enable, forever=False)

[6]: # 清除单片机自动发送过来的缓存数据 Clear the cache data automatically sent by the MCU
bot.clear_auto_report_data()
```

小车控制速度相关参数设置

```
[9]: # 控制电机运动 Control motor movement
def car_motion(V_x, V_y, V_z):
    speed_x = V_x / 10.0
    speed_y = V_y / 10.0
    speed_z = V_z / 10.0
    bot.set_car_motion(speed_x, speed_y, speed_z)
    return speed_x, speed_y, speed_z

# 创建三个滑块来控制小车的速度 Create three sliders to control the speed of the car
interact(car_motion, \
         V_x=widgets.IntSlider(min=-10,max=10,step=1,value=0), \
         V_y=widgets.IntSlider(min=-10,max=10,step=1,value=0), \
         V_z=widgets.IntSlider(min=-50,max=50,step=1,value=0));

Error displaying widget: model not found

[1]: # 停止运动 stop motion
bot.set_car_motion(0, 0, 0)

[1]: # 获取小车线速度和角速度数据
# Obtain the linear velocity and angular velocity data of the car
try:
    while True:
        V_x, V_y, V_z = bot.get_motion_data()
        print("speed:", V_x, V_y, V_z)
        bot.clear_auto_report_data()
        time.sleep(.1)
except KeyboardInterrupt:
    pass
```

PID相关参数设置 Set PID parameters

```
[ ]: # PID 参数控制，会影响set_car_motion函数控制小车的运动速度变化情况。默认情况下可不调整。  
# PID parameter control will affect the set_CAR_motion function to control the speed change of the car. This parameter is optional.  
kp = 0.8  
ki = 0.06  
kd = 0.5  
bot.set_pid_param(kp, ki, kd, forever=False)
```

```
[ ]: kp, ki, kd = bot.get_motion_pid()  
print("PID:", kp, ki, kd)
```

- 恢复出厂配置，重置Flash数据。

Restore factory Settings and reset Flash data

```
[ ]: # 恢复出厂配置 Restoring factory Settings  
bot.reset_flash_value()
```

```
[ ]: # 程序结束后请删除对象，避免在其他程序中使用Rosmaster库造成冲突  
# After the program is complete, delete the object to avoid conflicts caused by using the library in other programs  
del bot
```

```
[ ]:
```

REQUIRED Question: Based on these scripts, what is the minimal code needed to control the robot motors? What are the inputs for the key functions? What is the key difference between the scripts?

Among these scripts, the minimal code to control the robot motors includes three parts: importing the library from Rosmaster_Lib import Rosmaster, creating the object bot = Rosmaster(), and using bot.set_motor(M1, M2, M3, M4) to set the speeds of the four motors. The inputs for set_motor() are the speed values for each motor, typically in the range of -100 to 100. Another key function is set_car_motion(Vx, Vy, Vz), which takes linear velocities in the X and Y directions and angular velocity around the Z-axis to control the robot's movement. The main difference between the scripts lies in the control method: some use direct motor speed control, some use overall motion control, and others handle serial communication, automatic data reporting, and PID parameter configuration.

Task2

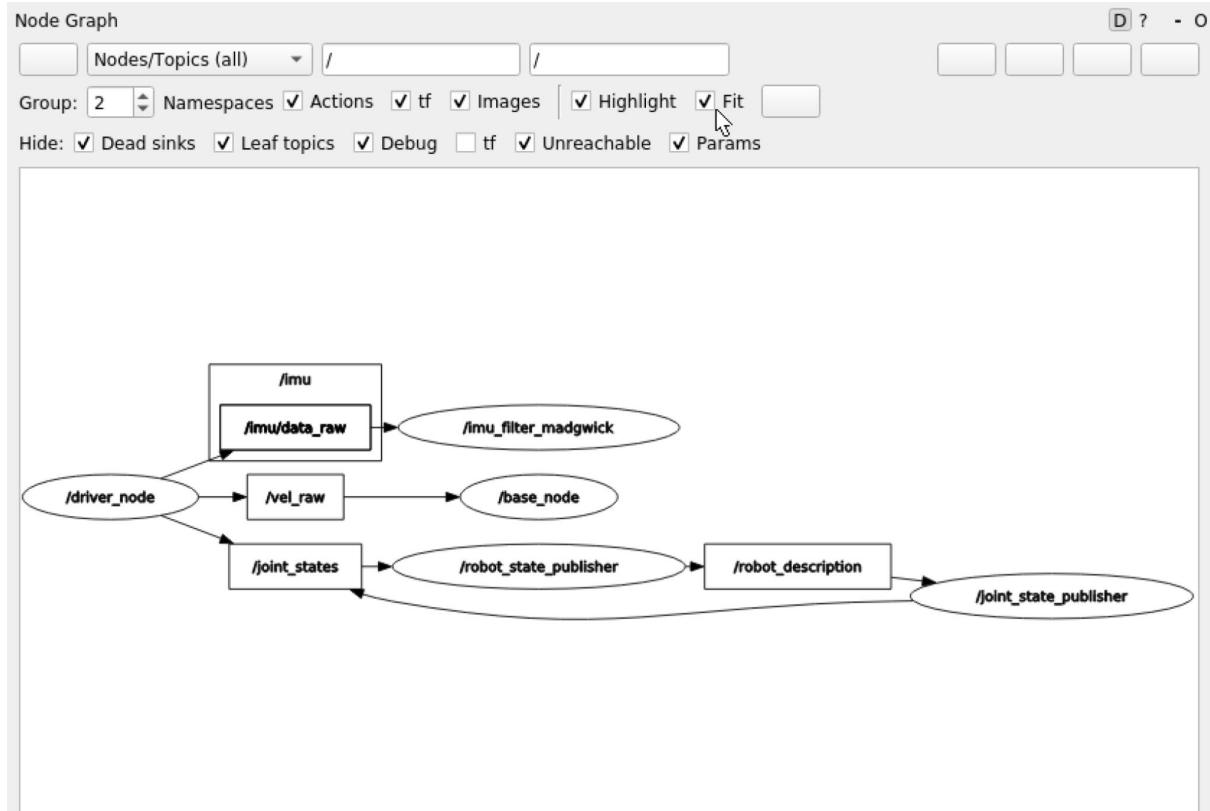
Action1

```
[INFO] [joint_state_publisher-1]: process started with pid [184]
[INFO] [robot_state_publisher-2]: process started with pid [186]
[INFO] [Mcnamu_driver_X3-3]: process started with pid [188]
[INFO] [base_node_X3-4]: process started with pid [190]
[INFO] [imu_filter_madgwick_node-5]: process started with pid [192]
[INFO] [ekf_node-6]: process started with pid [195]
[INFO] [yahboom_joy_X3-7]: process started with pid [197]
[robot_state_publisher-2] Parsing robot urdf xml string.
[robot_state_publisher-2] Link base_link had 7 children
[robot_state_publisher-2] Link back_left_wheel had 0 children
[robot_state_publisher-2] Link back_right_wheel had 0 children
[robot_state_publisher-2] Link imu_link had 0 children
[robot_state_publisher-2] Link camera_link had 0 children
[robot_state_publisher-2] Link front_left_wheel had 0 children
[robot_state_publisher-2] Link front_right_wheel had 0 children
[robot_state_publisher-2] Link laser_link had 0 children
[robot_state_publisher-2] [INFO] [1742330116.638329606] [robot_state_publisher]: got segment back_left_wheel
[robot_state_publisher-2] [INFO] [1742330116.638652733] [robot_state_publisher]: got segment back_right_wheel
[robot_state_publisher-2] [INFO] [1742330116.638709400] [robot_state_publisher]: got segment base_footprint
[robot_state_publisher-2] [INFO] [1742330116.638745494] [robot_state_publisher]: got segment base_link
[robot_state_publisher-2] [INFO] [1742330116.638776016] [robot_state_publisher]: got segment camera_link
[robot_state_publisher-2] [INFO] [1742330116.638808464] [robot_state_publisher]: got segment front_left_wheel
[robot_state_publisher-2] [INFO] [1742330116.638839349] [robot_state_publisher]: got segment front_right_wheel
[robot_state_publisher-2] [INFO] [1742330116.638868829] [robot_state_publisher]: got segment imu_link
[robot_state_publisher-2] [INFO] [1742330116.638897683] [robot_state_publisher]: got segment laser_link
[imu_filter_madgwick_node-5] [INFO] [1742330116.839984364] [imu_filter_madgwick]: Starting ImuFilter
[imu_filter_madgwick_node-5] [INFO] [1742330116.848947505] [imu_filter_madgwick]: Using dt computed from message headers
[imu_filter_madgwick_node-5] [INFO] [1742330116.849064485] [imu_filter_madgwick]: The gravity vector is kept in the IMU message.
[imu_filter_madgwick_node-5] [INFO] [1742330116.892042780] [imu_filter_madgwick]: Imu filter gain set to 0.100000
[imu_filter_madgwick_node-5] [INFO] [1742330116.932280012] [imu_filter_madgwick]: Gyro drift bias set to 0.000000
[imu_filter_madgwick_node-5] [INFO] [1742330116.941769667] [imu_filter_madgwick]: Magnetometer bias values: 0.000000 0.000000 0.000000
[joint_state_publisher-1] [INFO] [1742330118.826324684] [joint_state_publisher]: Waiting for robot_description to be published on the robot_description topic...
[imu_filter_madgwick_node-5] [INFO] [1742330119.111409501] [imu_filter_madgwick]: First IMU message received.
```

REQUIRED Question: Based on the printout to the terminal, what are some sensors the robot has?

From the terminal output, the robot has an IMU sensor and a laser sensor. The IMU is activated through the imu_filter_madgwick node and receives data from the gyroscope, accelerometer, and magnetometer. The laser sensor is identified by the laser_link segment, indicating the presence of a laser module.

Action3



Action4

```
root@jetson-desktop:/# cd /root/yahboomcar_ross2_ws/yahboomcar_ws/src/yahboomcar Bringup/launch/
root@jetson-desktop:~/yahboomcar_ross2_ws/yahboomcar_ws/src/yahboomcar Bringup/launch# ls -l
total 12
-rw-r--r-- 1 root root 4039 May 25 2023 yahboomcar_Bringup_R2_launch.py
-rw-r--r-- 1 root root 4070 Apr 21 2023 yahboomcar_Bringup_X1_launch.py
-rw-r--r-- 1 root root 4033 Jul 24 2023 yahboomcar_Bringup_X3_launch.py
```

Action5

```
root@jetson-desktop: / x root@jetson-desktop: ... x root@jetson-desktop: / x
root@jetson-desktop:/# ros2 run yahboomcar_ctrl yahboom_keyboard

Control Your SLAM-Bot!
-----
Moving around:
  u   i   o
  j   k   n   l   a
  m   ,   .   v   b
q/z : increase/decrease max speeds by 10%
w/x : increase/decrease only linear speed by 10%
e/c : increase/decrease only angular speed by 10%
t/T : x and y speed switch
s/S : stop keyboard control
space key, k : force stop
anything else : stop smoothly

CTRL-C to quit

currently:      speed 0.2          turn 1.0
```

Action6

```
root@jetson-desktop: /  
root@jet... x root@jets... x root@jets... x root@jets... x root@jets... x  
stamp:  
  sec: 1742334110  
  nanosec: 302087306  
frame_id: odom  
child_frame_id: base_footprint  
pose:  
  pose:  
    position:  
      x: 2.3244260743736276  
      y: 0.13887131195576904  
      z: 0.0  
    orientation:  
      x: 0.0  
      y: 0.0  
      z: 0.3219928591671434  
      w: 0.9467420972183334  
covariance:  
- 0.001  
- 0.0  
- 0.0  
- 0.0
```

after moving

```
root@jetson-desktop: /  
root@jetson... x root@jetson... x root@jetson... x root@jetson... x root@jetson... x  
frame_id: odom  
child_frame_id: base_footprint  
pose:  
  pose:  
    position:  
      x: 3.0133449375259103  
      y: 0.6806933521142551  
      z: 0.0  
    orientation:  
      x: 0.0  
      y: 0.0  
      z: 0.32789602184188105  
      w: 0.9447138184975747  
covariance:  
- 0.001  
- 0.0  
- 0.0  
- 0.0  
- 0.0  
- 0.0  
- 0.0
```

no raw

```
root@jetson-desktop: /  
root@jetson... x root@jetson... x root@jetson... x root@jetson... x root@jetson... x  
- 0.0  
- 0.0  
- 0.0  
- -1.850012366049885e-65  
- 4.794501474629942e-07  
- -5.73409940135275e-36  
- 0.0  
- 0.0  
- 0.0  
- 1.4415602063324894e-63  
- 3.043256349225315e-36  
- 4.794501474629942e-07  
- 0.0  
- 1.36473593035235e-60  
- -6.301376840540224e-59  
- 0.0  
- 0.0  
- 0.0  
- 2.3320457304230806e-05  
---
```

```
root@jetson-desktop: /  
root@jetson-desktop: ~ % ros2 topic echo /odom  
- 1.6272183056789317e-05  
---  
header:  
  stamp:  
    sec: 1742334193  
    nanosec: 304497786  
  frame_id: odom  
child_frame_id: base_footprint  
pose:  
  pose:  
    position:  
      x: 2.3244260743736276  
      y: 0.13887131195576904  
      z: 0.0  
    orientation:  
      x: 0.0  
      y: 0.0  
      z: 0.28440667465545877  
      w: 0.9587037307799652  
covariance:  
- 0.0008568297565181341
```

after moving

```
root@jetson-desktop: /  
root@jetson-desktop: ~ % ros2 topic echo /odom  
- 0.0  
- 0.0  
- 0.0  
- 7.560849124112436e-57  
- 3.4285642851222364e-07  
- -1.4538619791487152e-34  
- 0.0  
- 0.0  
- 0.0  
- 7.904259466867932e-55  
- 3.3616596798096625e-36  
- 4.783655192970861e-07  
- 0.0  
- 3.3743828546188564e-53  
- -1.814097254377e-50  
- 0.0  
- 0.0  
- 0.0  
- 1.743231679270317e-05  
---
```

```
root@jetson-desktop: /
```

```
root@jetson-desktop: ~ % ros2 topic echo /odom
```

```
---
```

```
header: { stamp: { sec: 1742334365, nanosec: 902474606 }, frame_id: "odom", child_frame_id: "base_footprint" }
```

```
pose: { position: { x: 3.0133449375259103, y: 0.6806933521142551, z: 0.0 }, orientation: { x: 0.0, y: 0.0, z: 0.2850819173614843, w: 0.9585031561729465 } }
```

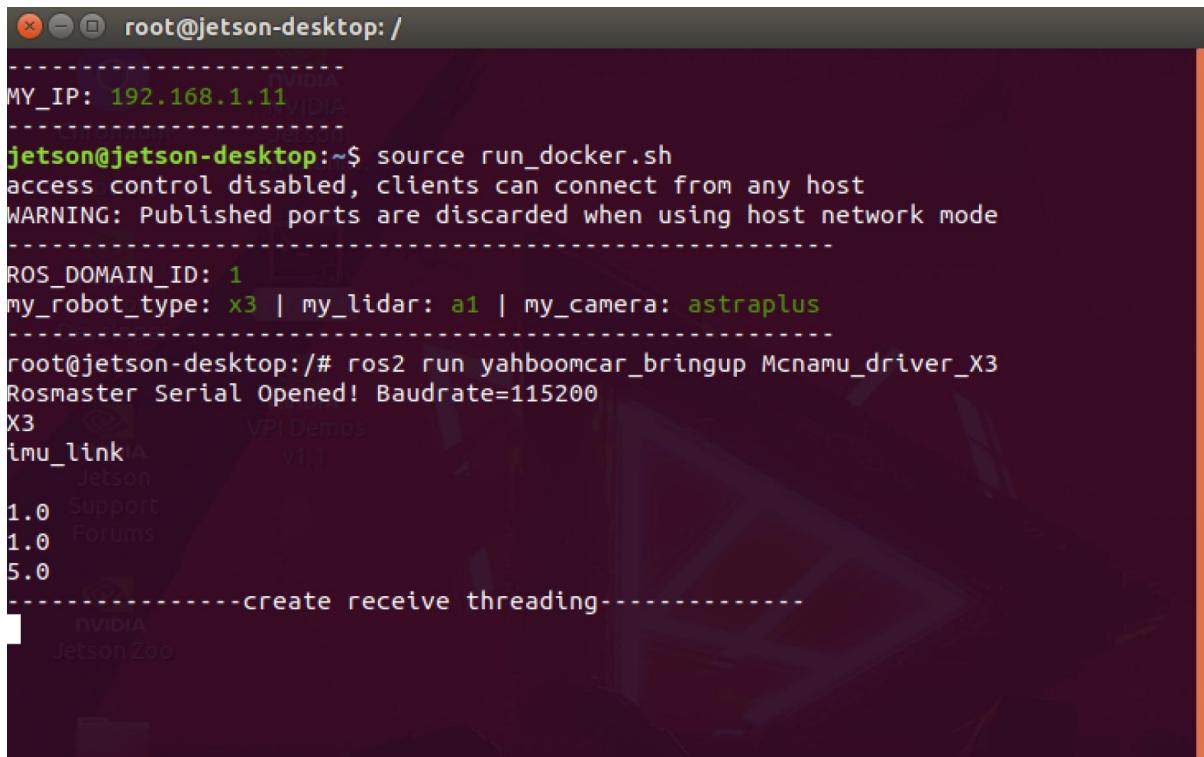
```
covariance: [ -0.0008550158110066053, -1.5222026098233978e-24 ]
```

REQUIRED Question: Based on the previous steps and any other ROS tools that you use, what is this example doing? What are some key topics and message types used in the example?

This example shows how multiple sensors and user input are used to produce various frames or states for the robot. The topics `odom_raw` and `odom` are used here, each providing a frame based on sensor readings and user commands. Other launch files may define different transform functions, which can result in different frames or states even when given the same user input.

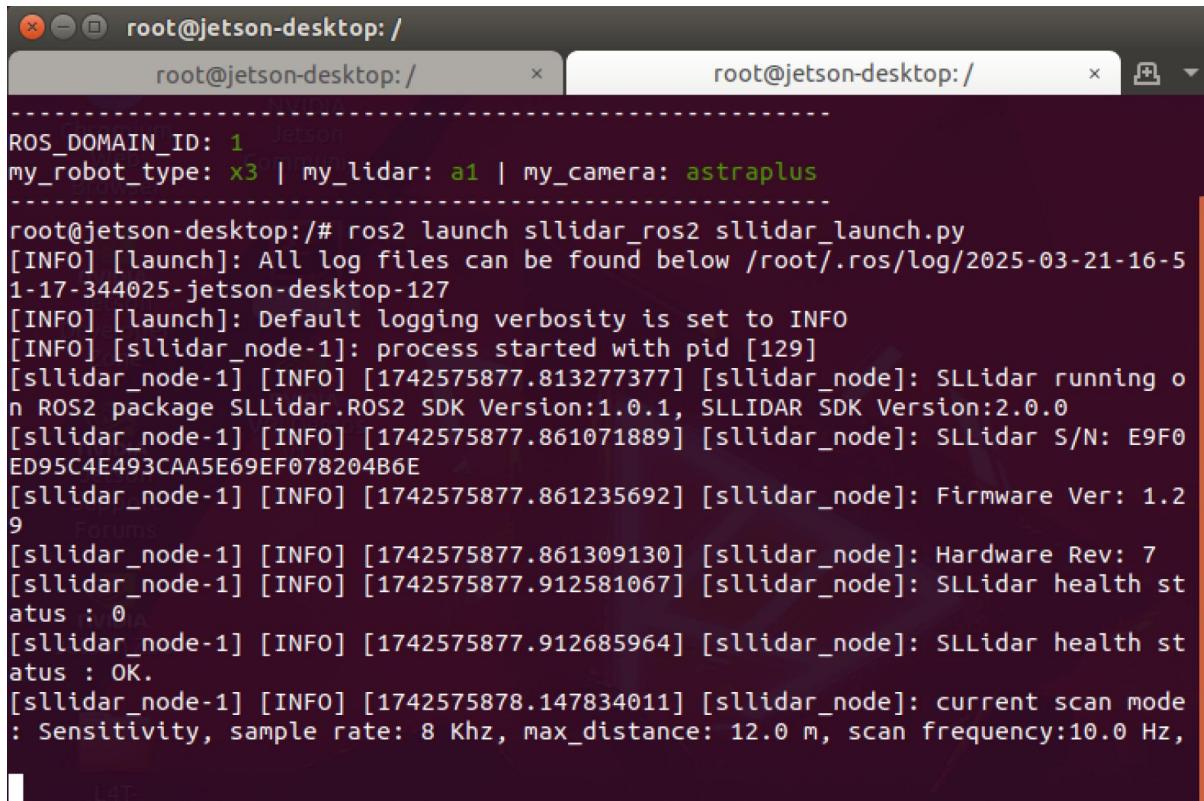
Task3

Action2



```
root@jetson-desktop: /  
-----  
MY_IP: 192.168.1.11  
-----  
jetson@jetson-desktop:~$ source run_docker.sh  
access control disabled, clients can connect from any host  
WARNING: Published ports are discarded when using host network mode  
-----  
ROS_DOMAIN_ID: 1  
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus  
-----  
root@jetson-desktop:/# ros2 run yahboomcar Bringup Mcnamu_driver_X3  
Rosmaster Serial Opened! Baudrate=115200  
X3  
imu_link  
Jetson  
1.0 Support  
1.0 Forums  
5.0  
-----create receive threading-----  
NVIDIA  
Jetson Zoo
```

Action3



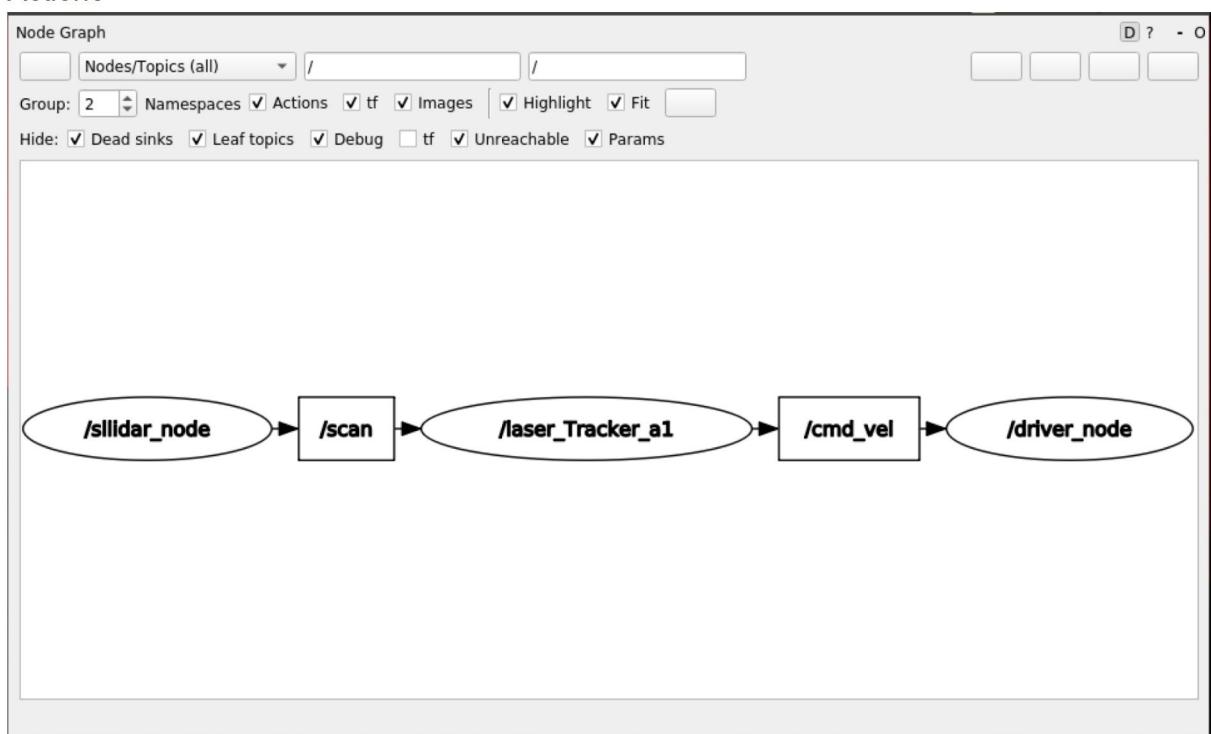
```
root@jetson-desktop: /  
root@jetson-desktop: /  
-----  
ROS_DOMAIN_ID: 1  
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus  
-----  
root@jetson-desktop:/# ros2 launch sllidar_ros2 sllidar_launch.py  
[INFO] [launch]: All log files can be found below /root/.ros/log/2025-03-21-16-5  
1-17-344025-jetson-desktop-127  
[INFO] [launch]: Default logging verbosity is set to INFO  
[INFO] [sllidar_node-1]: process started with pid [129]  
[sllidar_node-1] [INFO] [1742575877.813277377] [sllidar_node]: SLLidar running o  
n ROS2 package SLLidar.ROS2 SDK Version:1.0.1, SLLIDAR SDK Version:2.0.0  
[sllidar_node-1] [INFO] [1742575877.861071889] [sllidar_node]: SLLidar S/N: E9F0  
ED95C4E493CAA5E69EF078204B6E  
[sllidar_node-1] [INFO] [1742575877.861235692] [sllidar_node]: Firmware Ver: 1.2  
9  
[sllidar_node-1] [INFO] [1742575877.861309130] [sllidar_node]: Hardware Rev: 7  
[sllidar_node-1] [INFO] [1742575877.912581067] [sllidar_node]: SLLidar health st  
atus : 0  
[sllidar_node-1] [INFO] [1742575877.912685964] [sllidar_node]: SLLidar health st  
atus : OK.  
[sllidar_node-1] [INFO] [1742575878.147834011] [sllidar_node]: current scan mode  
: Sensitivity, sample rate: 8 KHz, max_distance: 12.0 m, scan frequency:10.0 Hz,
```

Action4

```
root@jetson-desktop: / 
root@jetson-desktop: / 
root@jetson-desktop: / 

MY_IP: 192.168.1.11
-----
jetson@jetson-desktop:~$ ./exec_docker.sh
-----
ROS_DOMAIN_ID: 1
my_robot_type: x3 | my_lidar: a1 | my_camera: astraplus
-----
root@jetson-desktop:/# ros2 run yahboomcar_laser laser_Tracker_a1_X3
improt done
init_pid: 0.1 0.0 0.1
init_pid: 2.0 0.0 2.0
init_pid: 3.0 0.0 5.0
start it
```

Action5



Action7

Parameter Reconfigure

Filter key:

[Collapse all](#) [Expand all](#)

driver_node

rqt_gui_py_node_292
sllidar_node

[Refresh](#)

(System message might be shown here when necessary)

/driver node

use_sim_time

car_type X3

imu_link imu_link

Prefix

xlinear_limit 1.0

ylinear_limit 1.0

angular_limit 5.0

Parameter Reconfigure

Filter key:

[Collapse all](#) [Expand all](#)

driver_node

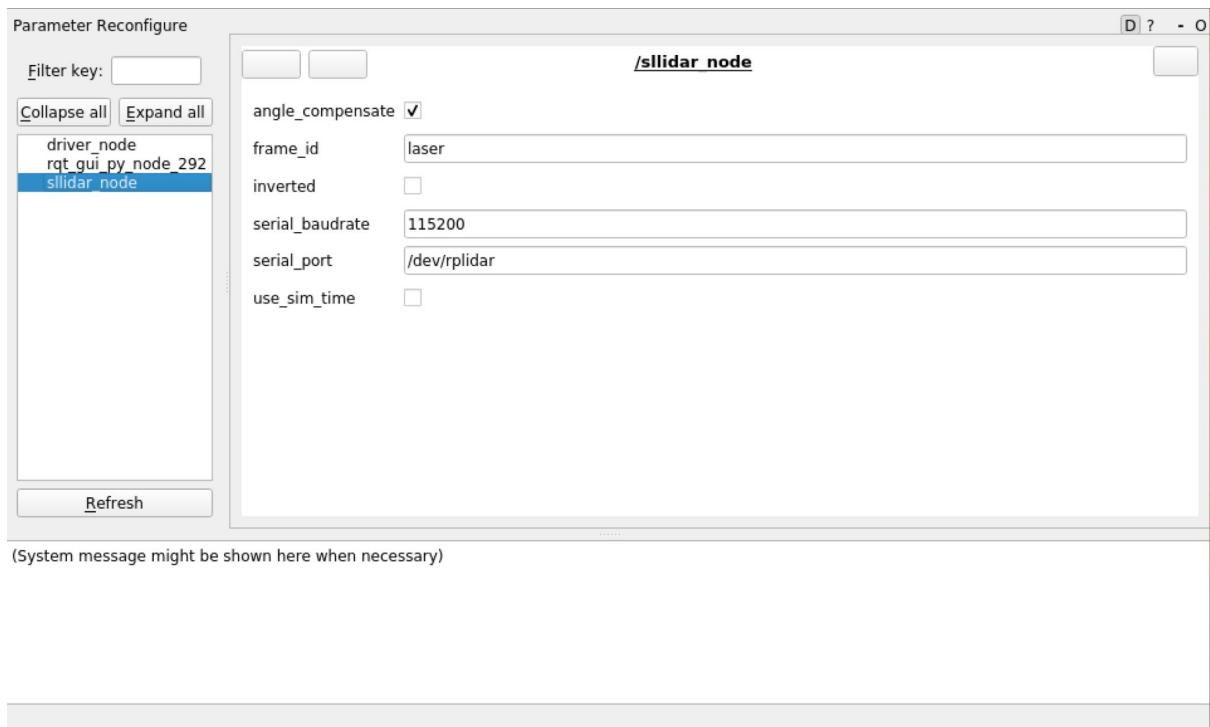
rqt_gui_py_node_292
sllidar_node

[Refresh](#)

(System message might be shown here when necessary)

/rqt gui py node 292

use_sim_time



after changing parameters

