# Mobile Robot Navigation and Control, Lab3

Author: Ying Yiwen Weiyu Chen
Number: 12210159 12210460

**Abstract**

In this lab, we learn how to operate the laser distance sensor that comes with TurtleBot3.

## 1 Introduction

We use Turtlebot3 and HLS-LFCD2 LiDAR sensor to control the robot, get the lidar message. First set up the environment, to enable us to control the robot on our computer. Take one computer as master, another computer and turtlebot as host. Then we can control the robot by ros package. Also, we can write ros package ourselves to deal with the scan message.

## 2 Setting up TurtleBot 3

Using the command "roslaunch turtlebot3_bringup turtlebot3_robot.launch" on turtlebot, "roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch" on my computer, I can control the robot.

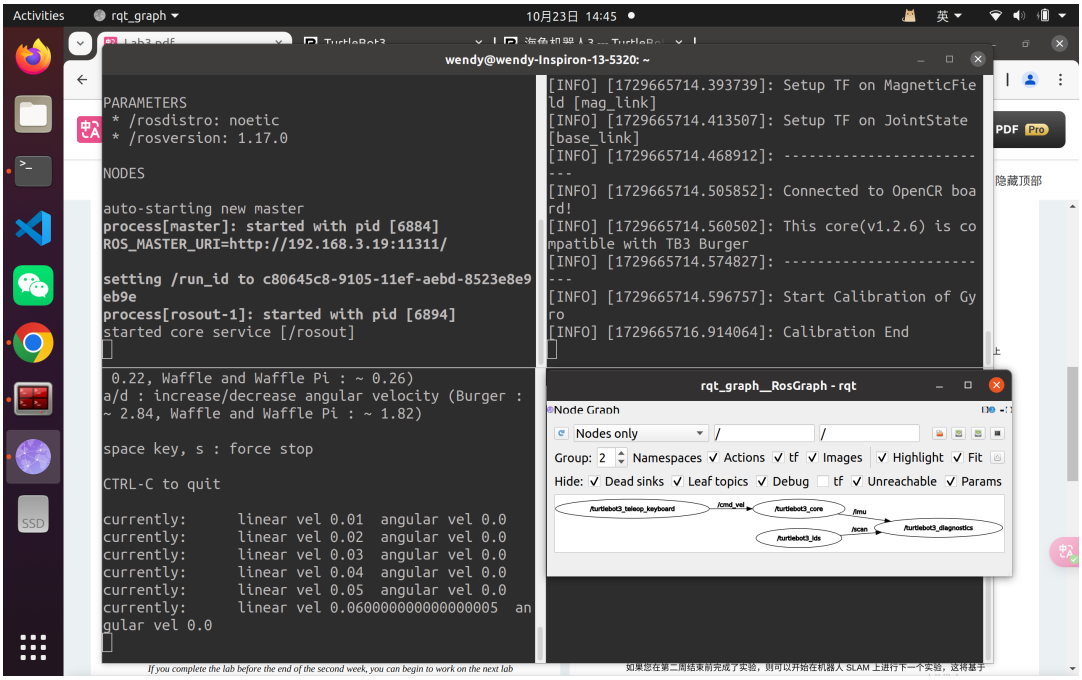



Fig. 1: rqt graph

# 3 HLS-LFCD2

The message /scan tells the information in the lidar. Using "rostopic echo /scan":

(a) The first page

(b) The second page

Fig. 2: The information in scan

Using the rviz, we can obeserve the pointcloud more clearly.

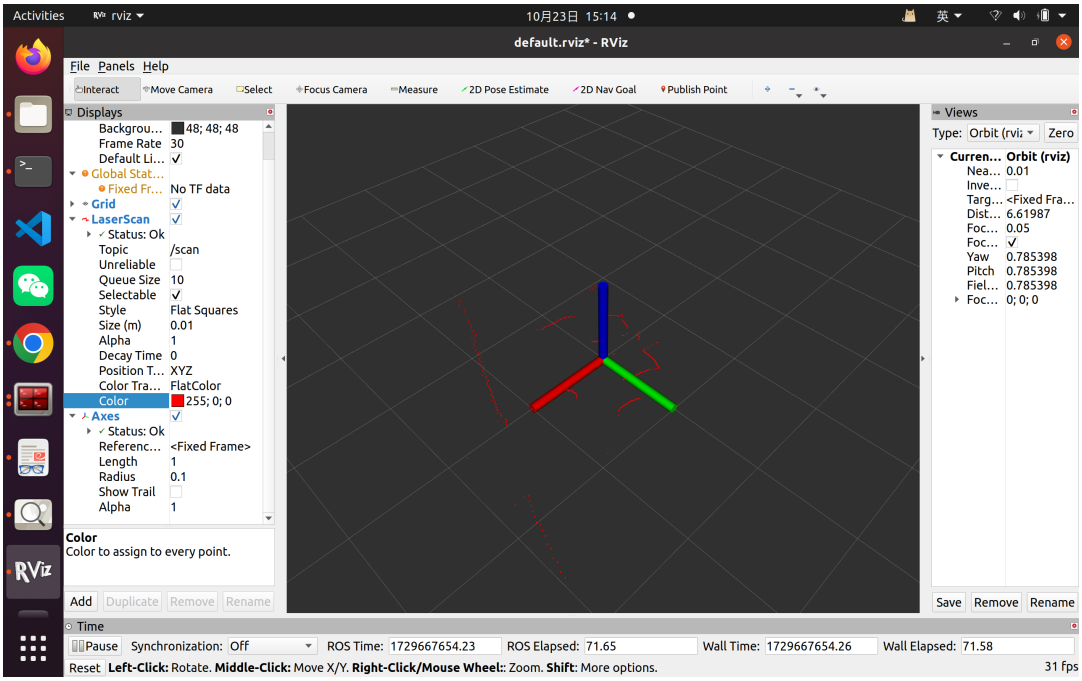



Fig. 3: rviz result

# 4 Verify the Performance of LiDAR

We put the robot beside a wall, when there is nothing else in other direction, the minimum distance (except 0) is the normal distance towards the wall. As the wall is a plane, we can easily get the true distance. According to the maunual, the distance is from the center of LiDAR to the obstacle. Like this:

Fig. 4: Experiment Scenario

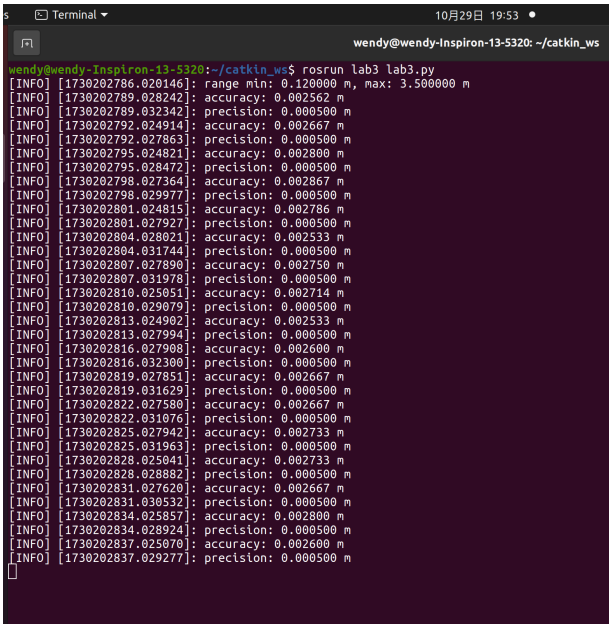
```
def callback(scan):
  global dis
  dis_min = scan.range_max
  for i in range(0, 360):
    distance = scan.ranges[i]
    if distance < dis_min and distance != 0:
      dis_min = distance

  # the minimum distance represent the measured result normal to the wall
  dis.append(dis_min)
```

```python
def process_data(_):
  global dis
  if len(dis) > 0:
    dis_array = np.array(dis)
    rospy.loginfo("accuracy: %f m", abs(dis_array.mean() - dis_true))
    rospy.loginfo("precision: %f m", (dis_array.max() - dis_array.min()) / 2)
    dis.clear()
```

We can also read min and max in the scan, as follow:

```python
# range_min and range_max
scan = rospy.wait_for_message('/scan', LaserScan)
rospy.loginfo("range min: %f m, max: %f m", scan.range_min, scan.range_max)
```

When the true distance is 0.464m.
The result is:
range minimum: 0.120000m, range maximum: 3.500000m
accuracy: 0.002667m, precision: 0.0005m




(a) My result

| | |
|---|---|
| Light source | Semiconductor Laser Diode(λ=785nm) |
| LASER safety | IEC60825-1 Class 1 |
| Current consumption | 400mA or less (Rush current 1A) |
| Detection distance | 120mm ~ 3,500mm |
| Interface | 3.3V USART (230,400 bps) 42bytes per 6 degrees, Full Duplex option |
| Ambient Light Resistance | 10,000 lux or less |
| Sampling Rate | 1.8kHz |
| Dimensions | 69.5(W) X 95.5(D) X 39.5(H)mm |
| Mass | Under 125g |

***Measurement Performance Specifications***

| Items | Specifications |
|---|---|
| Distance Range | 120 ~ 3,500mm |
| Distance Accuracy (120mm ~ 499mm) | ±15mm |
| Distance Accuracy(500mm ~ 3,500mm) | ±5.0% |
| Distance Precision(120mm ~ 499mm) | ±10mm |
| Distance Precision(500mm ~ 3,500mm) | ±3.5% |
| Scan Rate | 300±10 rpm |
| Angular Range | 360° |

(b) Manual

Fig. 5: The information in scan

Our experimental results are in accordance with the instructions.
The range is the same as the manual, 120mm-3500mm, and the tolerances are within the limits specified in the manual(120mm-499mm, ±15mm).

# 5 Conclusion

We successfully demonstrated that the HLS-LFCD2 LiDAR sensor operates effectively for distance measurement in a mobile robot navigation context. The findings are consistent with the manual specifications, confirming the sensor's suitability for navigation tasks.