


Lio-Mapping Mapping

1、Radar configuration

Enter the command in the terminal and press the Enter key

```
sudo nano ~/aloam/src/vanjee_lidar_v2.4/src/config/config.yaml
```

Taking my car as an example, the Jetson nano's IP is 10.168.1.100 and the radar IP is 10.168.1.68, so modify it as shown in the picture below.



```
ros_send_imu_topic: wlr_720/imu
imu_frame_id: wlr_720
ros_send_point_cloud_topic: wlr_720/cloud_points
point_frame_id: wlr_720
proto:
  lidar_ip: 192.168.2.86
  dest_ip: 224.1.1.1
  dest_port: 3001
  local_ip: 192.168.2.88
```

2、Run mapping

There are two ways to run mapping, one is online mapping and the other is offline mapping. Online mapping uses lidar scanning and mapping in real time. Offline mapping uses a bag with radar data for mapping, which can be done without radar.

Before mapping, we need to calibrate the 16-line lidar and imu. Before calibration, you need to start the imu and radar, and perform imu and radar calibration through the bag that records imu data and radar data.

Enter the aloam workspace through the terminal, and enter the command in the terminal to start imu and radar data. The imu I use here is Vette's 10-axis imu. It should be noted that the transmission frequency of imu needs to be 200hz.

```
source devel/setup.bash
roslaunch vanjee_to_velodyne vanjee_to_velodyne.launch
```

This launch starts the radar package, as well as the data conversion package and imu package.

Then open a new terminal, enter the lio_Mapping workspace, and enter the command in the terminal:

```
source devel/setup.bash
rosbag record -O clib_imu.bag /imu/data /velodyne_points
```

At this time, rotate the car on the x-axis, y-axis, and z-axis. After recording the data for about 1 minute, press ctrl + c in the packet recording terminal to save. The picture below shows saving the calibration data package.

```
[ INFO] [1701765463.994169988]: Subscribing to /velodyne_points
[ INFO] [1701765464.007101415]: Recording to 'clib_imu.bag'.
^Cyahboom@yahboom-desktop:~$
yahboom@yahboom-desktop:~$
yahboom@yahboom-desktop:~$
yahboom@yahboom-desktop:~$
```

文件名 ^	大小	类型	修改时间	权限	用户/用户组
.bashrc	4.1 KB		2023/11/22 15:10	-rw-r--r--	yahboom/y...
.ICEauthority	52.6 KB		2023/12/04 19:26	-rw-----	yahboom/y...
.profile	807 B		2023/08/16 11:02	-rw-r--r--	yahboom/y...
.python_history	332 B		2023/11/08 18:11	-rw-----	yahboom/y...
.sudo_as_admin_successful	0		2023/08/16 13:58	-rw-r--r--	yahboom/y...
.viminfo	25.6 KB		2023/11/23 15:56	-rw-----	yahboom/y...
.Xauthority	244 B		2023/11/17 19:54	-rw-----	yahboom/y...
.xsessionrc	2 KB		2023/08/16 11:02	-rw-r--r--	yahboom/y...
calibrationdata.tar.gz	11.6 MB		2023/09/01 12:09	-rw-rw-r--	yahboom/y...
clib_imu.bag	713.1 MB		2023/12/05 16:40	-rw-rw-r--	yahboom/y...
cutecom.log	0		2023/10/12 17:51	-rw-rw-r--	yahboom/y...
espeak_tts.py	363 B		2023/09/27 10:13	-rw-rw-r--	yahboom/y...
espeak_zh.py	432 B		2023/09/27 10:48	-rw-rw-r--	yahboom/y...
examples.desktop	8.8 KB		2023/08/16 11:02	-rw-r--r--	yahboom/y...
file.mp3	0		2023/09/27 10:40	-rw-rw-r--	yahboom/y...

Open the file (note that src is preceded by your own workspace path)

```
src/imuCalibEx/src/imu_lidar_calibration/linkalibr/launch/ros_calib_init.launch
```

Modify the bag path you just saved as shown in the figure below:

```
1 <launch>
2
3 <!-- bag topics -->
4 <node name="ros_calib_init" pkg="linkalibr" type="ros_calib_init" output="screen" clear_params=
5 "true" required="true">
6   <!-- bag topics -->
7   <param name="topic_imu" type="string" value="/wit/imu" />
8   <param name="topic_lidar" type="string" value="/velodyne_points" />
9
10   <!-- bag parameters -->
11   <!--
12   <param name="path_bag" type="string" value=
13   "/media/usl/37150acd-8dc7-4336-b1fb-90ffdd3488c5/ouster_vectornav/vectornav@400Hz/linkalibr_ouster_vecto
14   rnav_data4_400Hz_far.bag" /> -->
15   <param name="path_bag" type="string" value="/home/yahboom/clib_imu.bag" />
16   <param name="bag_start" type="double" value="0" />
17   <param name="bag_end" type="double" value="0" />
18   <!-- NDT Resolution -->
19   <param name="ndt_resolution" type="double" value="0.25" />
20 </node>
21
22 <node name="ros_calib_init_optimizer" pkg="linkalibr" type="ros_calib_init_optimizer" output=
23 "screen" clear_params="true" required="true">
24   <param name="max_frames" type="int" value="650" />
25   <param name="gyroscope_noise_density" type="double" value="0.00000243" />
26   <param name="accelerometer_noise_density" type="double" value="0.0091179" />
27   <param name="calibration_result_filename" type="string" value=
28   "/home/wanji/catkin_lidar_imu_TEC/src/imu_lidar_calibration/linkalibr/data/I_T_L_init.txt"/>
29 </node>
```

Close the vanjee_to_velodyne node and enter the command in the terminal:

```
source devel/setup.bash
roslaunch linkalibr ros_calib_init.launch
```

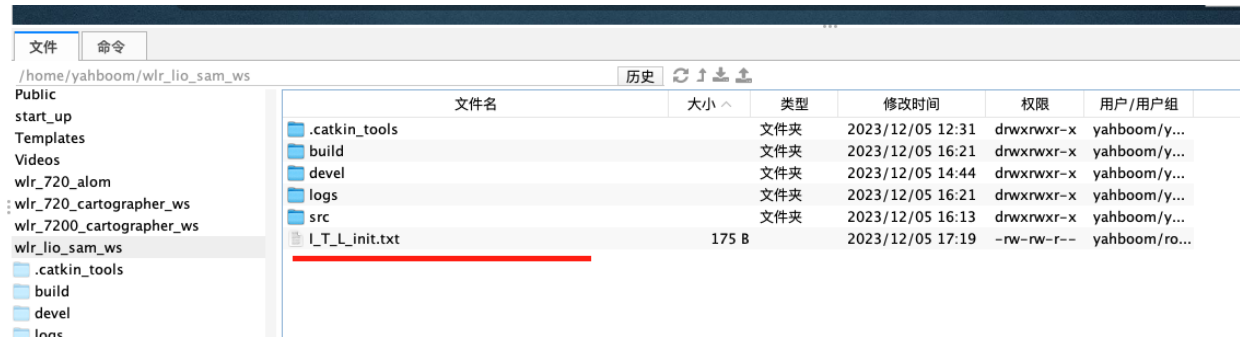
Wait for calibration to complete.

The picture below shows the result of successful calibration.

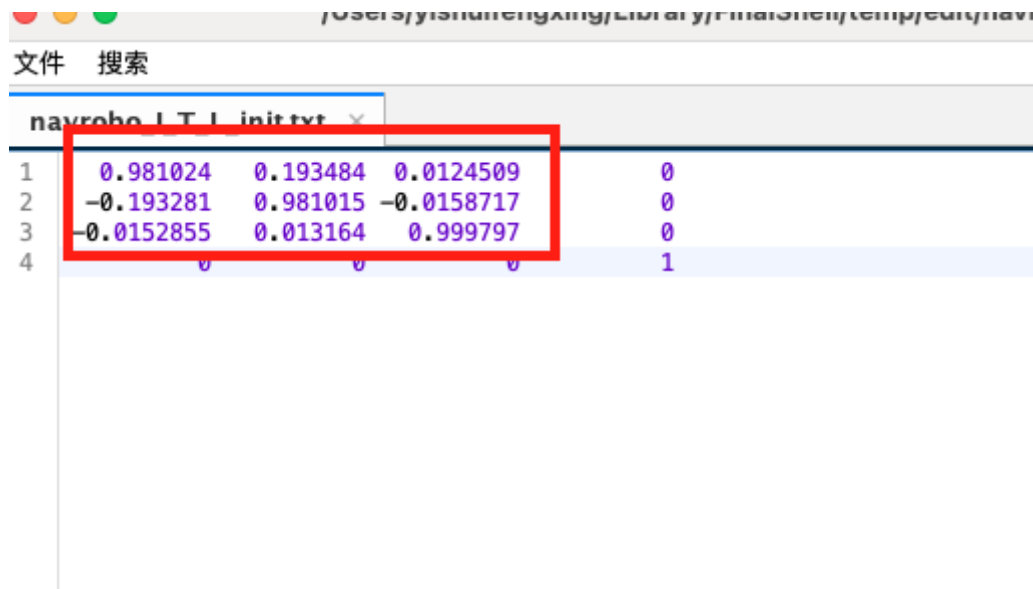
```
[ INFO] [1701767799.595114171]: Frame: 648 / 650
[ INFO] [1701767799.615602981]: Frame: 649 / 650
[ INFO] [1701767799.658650382]: Frame: 650 / 650
Rot3:

    0.999993 -0.00228046  0.00307144
    0.00226249    0.99998  0.00584106
    -0.0030847 -0.00583407  0.999978
Euler Angles: 179.665 179.824 -179.869
Marginal Covariance
    0.845547 0.00306384 -1.67901
    0.00306384  0.834835 -0.4306
    -1.67901  -0.4306  248.059
```

You can see the saved calibration files in the working directory.



We open the file and take the transpose of the three-dimensional matrix in the upper left corner as our calibration result.

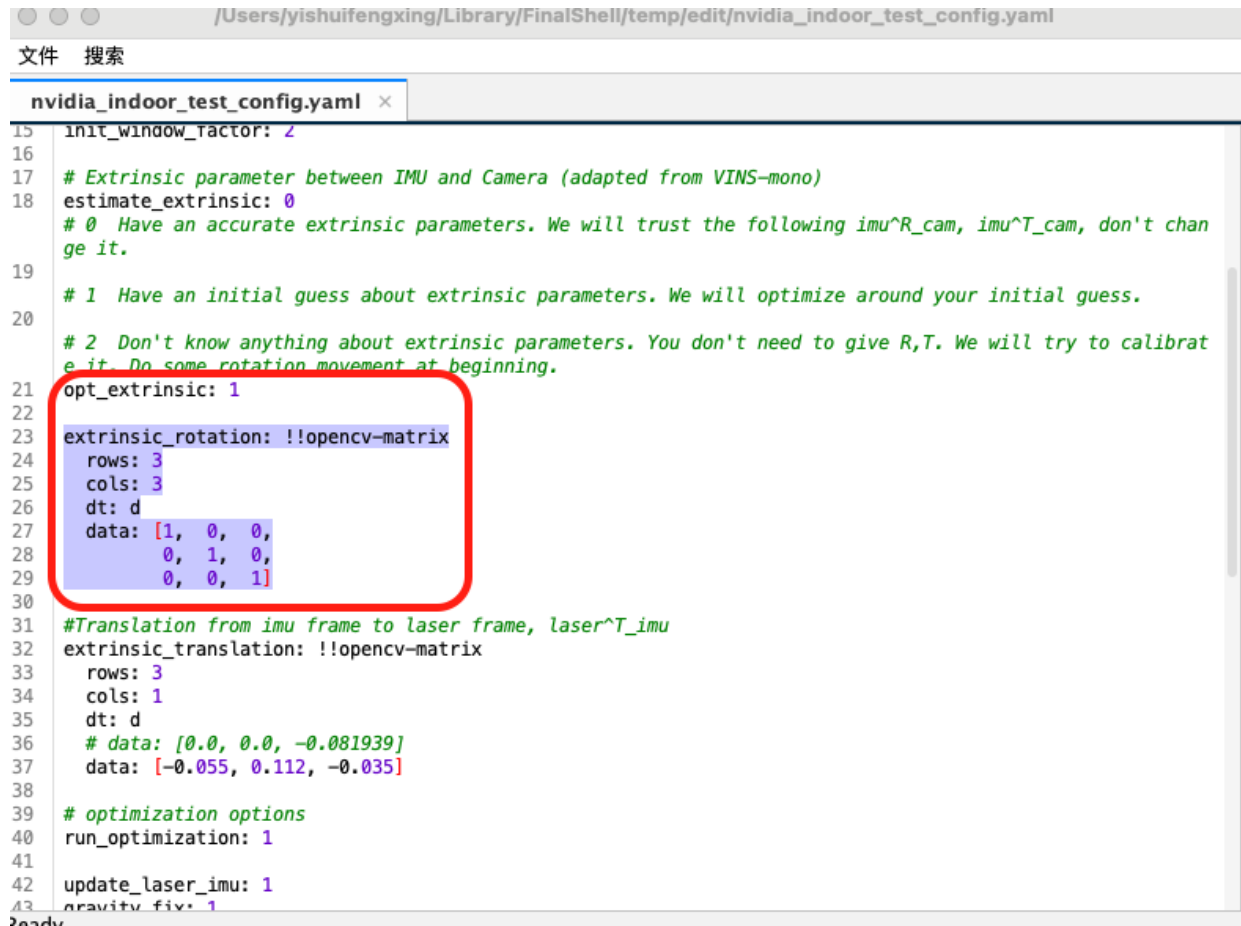


Modify the values of extrinsicRot and extrinsicRPY in the lio-sam configuration file by transposing the above result.

The configuration file path of Lio-sam is (note that src is preceded by the path of the workspace).

```
src/Lio_Sam/src/lio-mapping/config/indoor_test_config.yaml #Indoor parameters

src/Lio_Sam/src/lio-mapping/config/outdoor_test_config.yaml #Outdoor parameters
```



```
15 init_window_factor: 2
16
17 # Extrinsic parameter between IMU and Camera (adapted from VINS-mono)
18 estimate_extrinsic: 0
19 # 0 Have an accurate extrinsic parameters. We will trust the following imu^R_cam, imu^T_cam, don't change it.
20 # 1 Have an initial guess about extrinsic parameters. We will optimize around your initial guess.
21 # 2 Don't know anything about extrinsic parameters. You don't need to give R,T. We will try to calibrate it. Do some rotation movement at beginning.
22 opt_extrinsic: 1
23 extrinsic_rotation: !!opencv-matrix
24   rows: 3
25   cols: 3
26   dt: d
27   data: [1, 0, 0,
28          0, 1, 0,
29          0, 0, 1]
30
31 #Translation from imu frame to laser frame, laser^T_imu
32 extrinsic_translation: !!opencv-matrix
33   rows: 3
34   cols: 1
35   dt: d
36   # data: [0.0, 0.0, -0.081939]
37   data: [-0.055, 0.112, -0.035]
38
39 # optimization options
40 run_optimization: 1
41
42 update_laser_imu: 1
43 gravity_fix: 1
```

1. Offline mapping

The difference between offline mapping and online mapping is that offline mapping uses recorded bags for mapping. We have already mentioned the method of recording the bag when calibrating it above, so we won't go into it here. Now by default we already have the collected data bag.

Open a new terminal, enter the Lio-Mapping workspace, and enter the command in the terminal.

```
roslaunch test test.launch
```

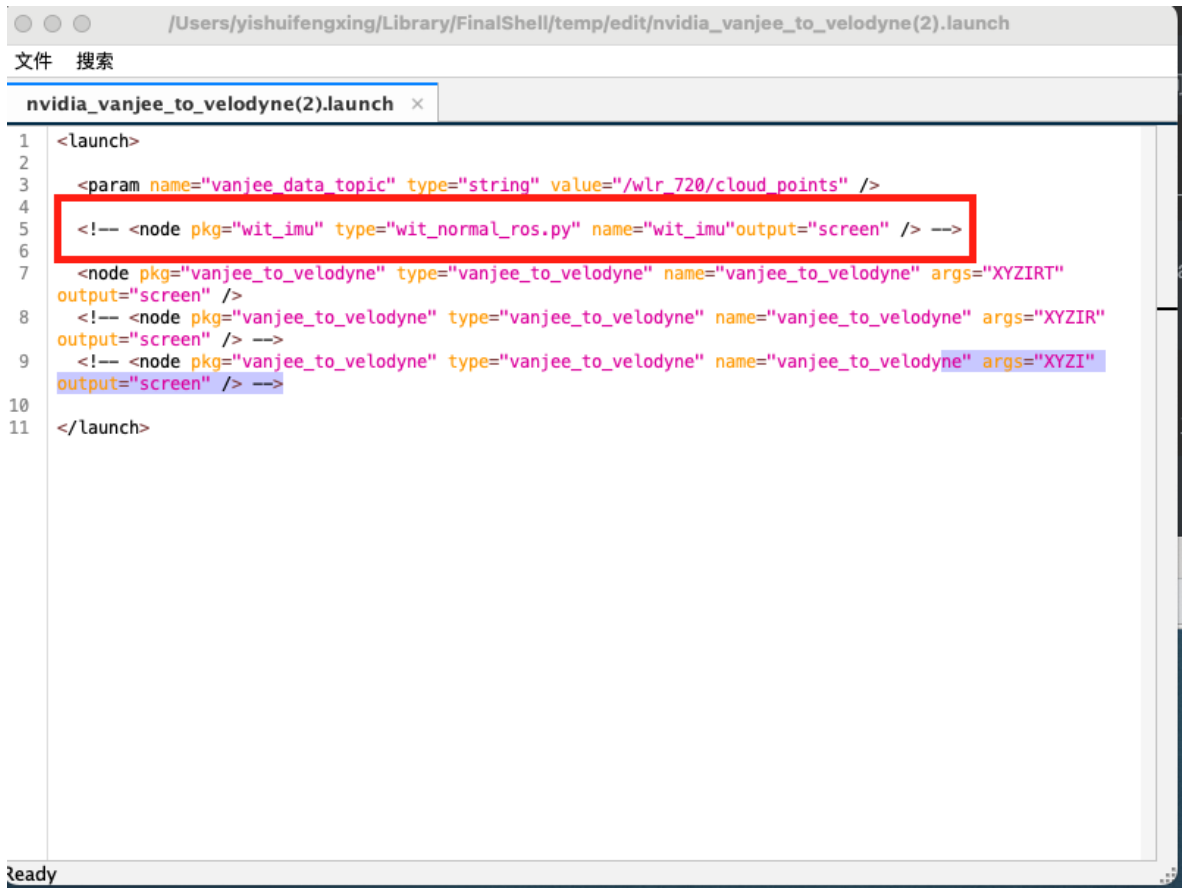
Open a new terminal, enter the Lio-Mapping workspace, and enter the command in the terminal.

```
source devel/setup.bash
roslaunch vanjee_to_velodyne vanjee_to_velodyne.launch
```

Note: If it is offline mapping, block the startup of imu in vanjee_to_velodyne.launch.

Code location:

liomapping的工作空间/src/vanjee_to_velodyne/launch/vanjee_to_velodyne.launch

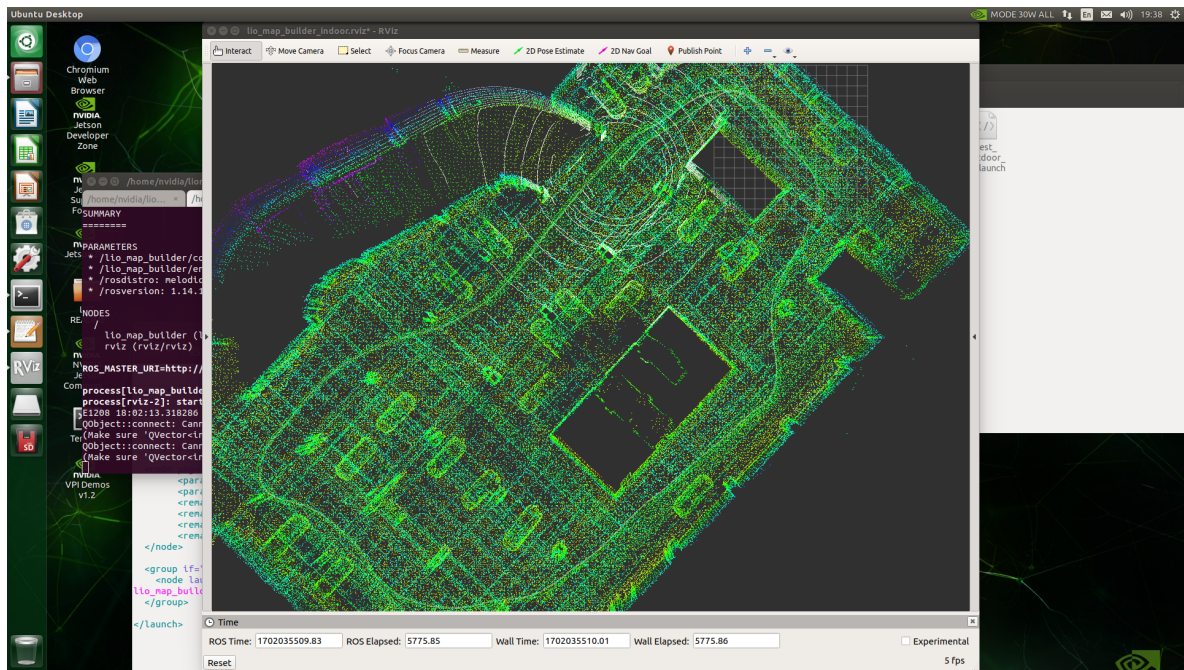


```
1 <launch>
2
3   <param name="vanjee_data_topic" type="string" value="/wlr_720/cloud_points" />
4
5   <!-- <node pkg="wit_imu" type="wit_normal_ros.py" name="wit_imu" output="screen" /> -->
6
7   <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZIRT"
8   output="screen" />
9   <!-- <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZIR"
10  output="screen" /> -->
11  <!-- <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZI"
12  output="screen" /> -->
13 </launch>
```

Enter the Lio-Mapping workspace through the terminal and enter the command in the terminal.

```
source devel/setup.bash
roslaunch lio map_4D_indoor.launch
```

As shown in the figure below, the operation status.

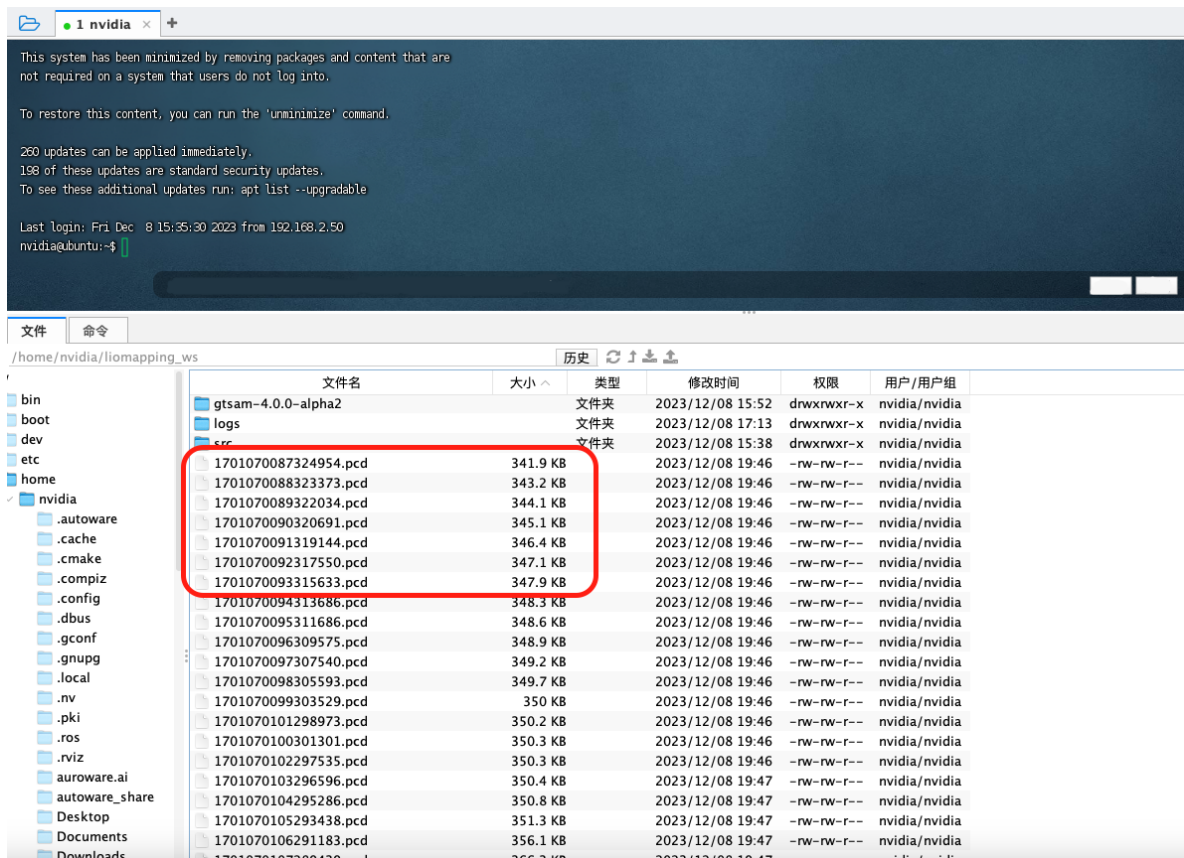


Then drive around to build the map. After the map is completed, reopen a terminal and enter the following command.

```
cd ~/liomapping_ws (Note that this is my workspace, everyone can define it themselves)
roslaunch lio_map_builder laser_cloud_surround
```

The picture below shows the saved map file

You can see the saved pcd file under the workspace.



2. Online mapping

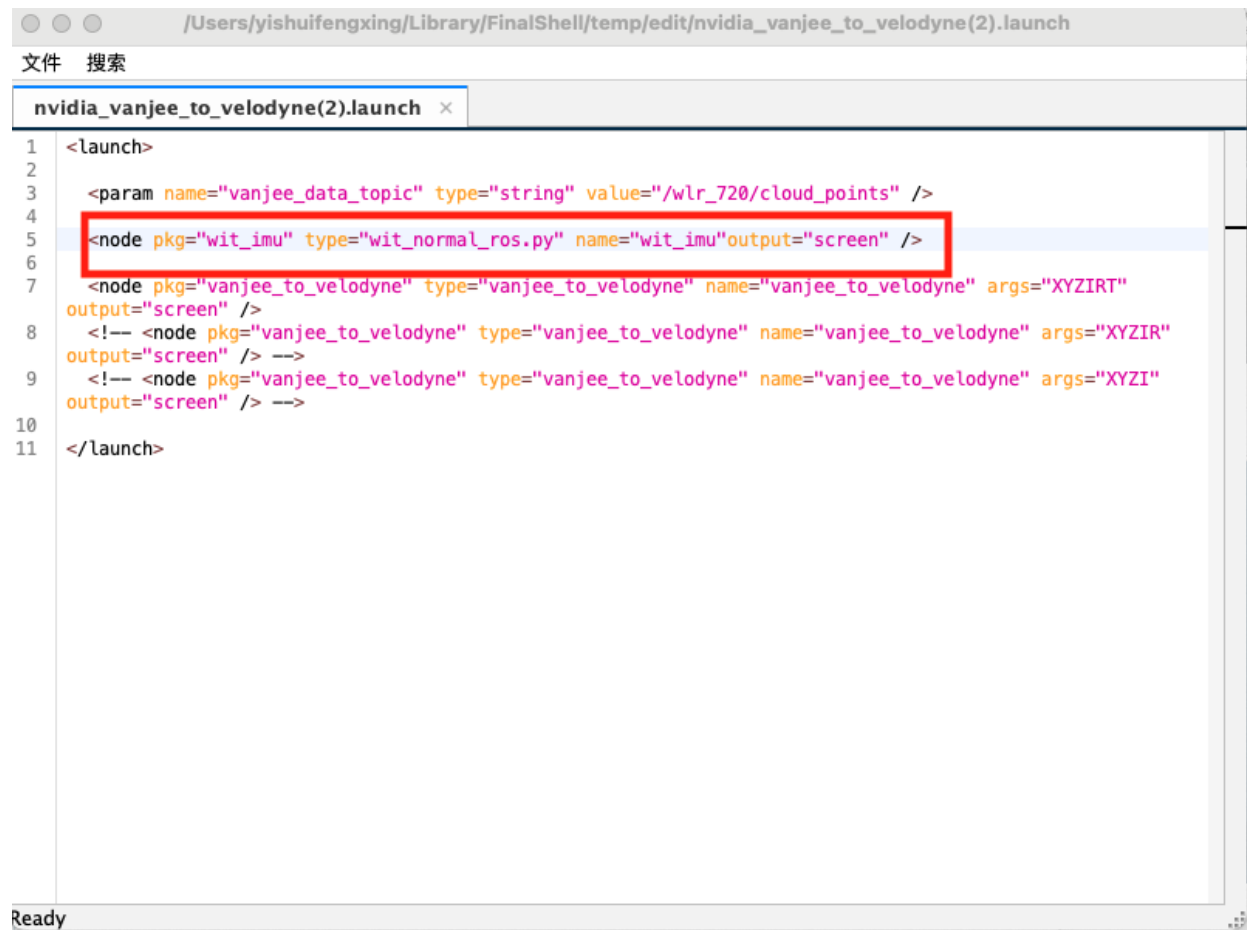
Open a new terminal, enter the Lio-Mapping workspace, and enter the command in the terminal.

```
source devel/setup.bash
roslaunch lio map_4D_indoor.launch
```

Open a new terminal, enter the Lio-Mapping workspace, and enter the command in the terminal:

```
source devel/setup.bash
roslaunch vanjee_to_velodyne vanjee_to_velodyne.launch
```

Note: If you have blocked imu before, you need to turn it on.



```
1 <launch>
2
3   <param name="vanjee_data_topic" type="string" value="/wlr_720/cloud_points" />
4
5   <node pkg="wit_imu" type="wit_normal_ros.py" name="wit_imu" output="screen" />
6
7   <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZIRT"
8   output="screen" />
9   <!-- <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZIR"
10  output="screen" /> -->
11  <!-- <node pkg="vanjee_to_velodyne" type="vanjee_to_velodyne" name="vanjee_to_velodyne" args="XYZI"
12  output="screen" /> -->
13 </launch>
```