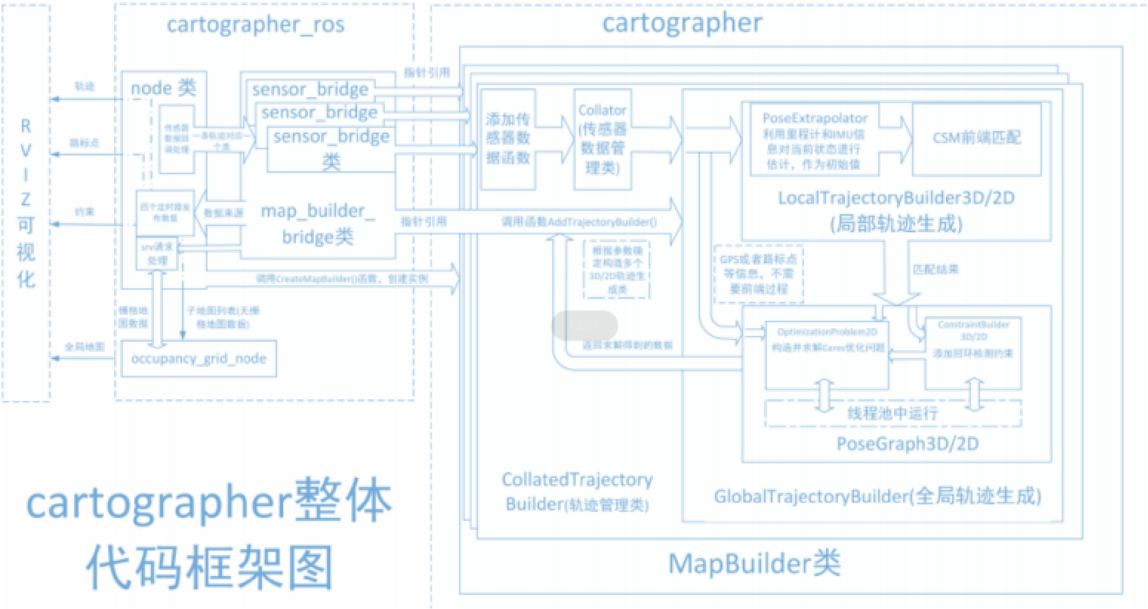
Cartographer建图

注：虚拟机需要与小车处在同一个局域网下，且ROS_DOMAIN_ID，需要一致，可以查看【使用前必看】来设置板子上的IP和ROS_DOMAIN_ID。

1、Cartographer简介

Cartographer是Google开源的一个ROS系统支持的2D和3D SLAM（simultaneous localization and mapping）库。基于图优化（多线程后端优化、cere构建的problem优化）的方法建图算法。可以结合来自多个传感器（比如，LIDAR、IMU 和 摄像头）的数据，同步计算传感器的位置并绘制传感器周围的环境。

cartographer的源码主要包括三个部分：cartographer、cartographer_ros和ceres-solver（后端优化）。



RVIZ可视化
cartographer_ros
轨迹
路标点
约束
全局地图
node 类
sensor_bridge
sensor_bridge
sensor_bridge类
数据来源
srv请求处理
map_builder_bridge类
指针引用
指针引用
调用CreateMapBuilder()函数，创建实例
栅格地图数据
子地图列表(无栅格地图数据)
occupancy_grid_node
cartographer
添加传感器数据函数
Collator(传感器数据管理类)
PoseExtrapolator 利用里程计和IMU信息对当前状态进行估计，作为初始值
CSM前端匹配
LocalTrajectoryBuilder3D/2D (局部轨迹生成)
调用函数AddTrajectoryBuilder()
根据参数确定构造多个3D/2D轨迹生成类
GPS或者路标点等信息，不需要前端过程
匹配结果
OptimizationProblem2D 构造并求解Ceres优化问题
ConstraintBuilder 3D/2D 添加回环检测约束
返回求解得到的数据
线程池中运行
PoseGraph3D/2D
CollatedTrajectory Builder(轨迹管理类)
GlobalTrajectoryBuilder(全局轨迹生成)
MapBuilder类
cartographer整体代码框架图

cartographer采用的是主流的SLAM框架，也就是特征提取、闭环检测、后端优化的三段式。由一定数量的LaserScan组成一个submap子图，一系列的submap子图构成了全局地图。用LaserScan构建submap的短时间过程累计误差不大，但是用submap构建全局地图的长时间过程就会存在很大的累计误差，所以需要利用闭环检测来修正这些submap的位置，闭环检测的基本单元是submap，闭环检测采用scan_match策略。

cartographer的重点内容就是融合多传感器数据（odometry、IMU、LaserScan等）的submap子图创建以及用于闭环检测的scan_match策略的实现。

cartographer_ros是在ROS下面运行的，可以以ROS消息的方式接受各种传感器数据，在处理过后又以消息的形式publish出去，便于调试和可视化。

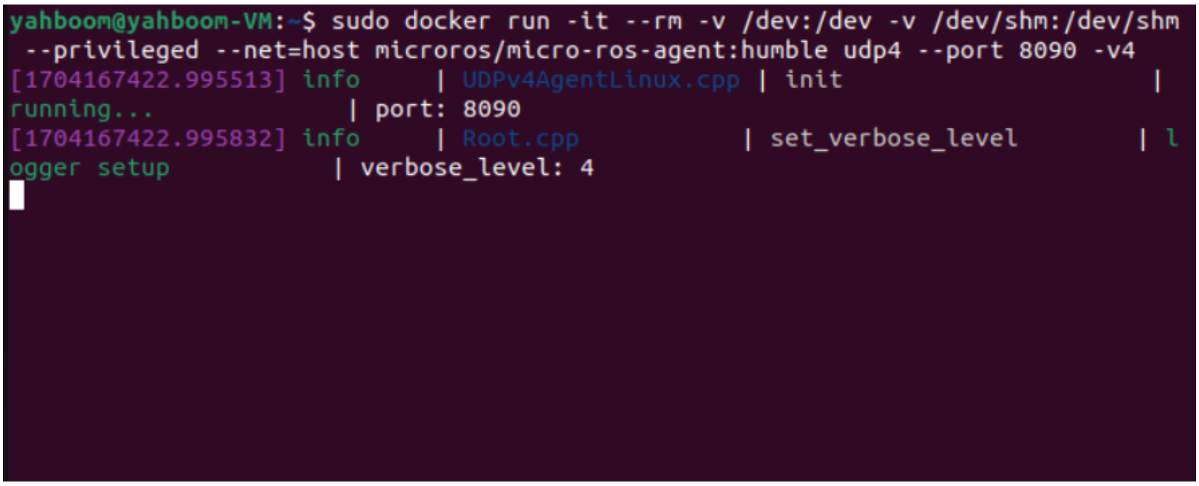
2、程序功能说明

小车连接上代理，运行程序，rviz中会显示建图的界面，用键盘或者手柄去控制小车运动，直到建完图。然后运行保存地图的指令保存地图。

3、启动并连接代理

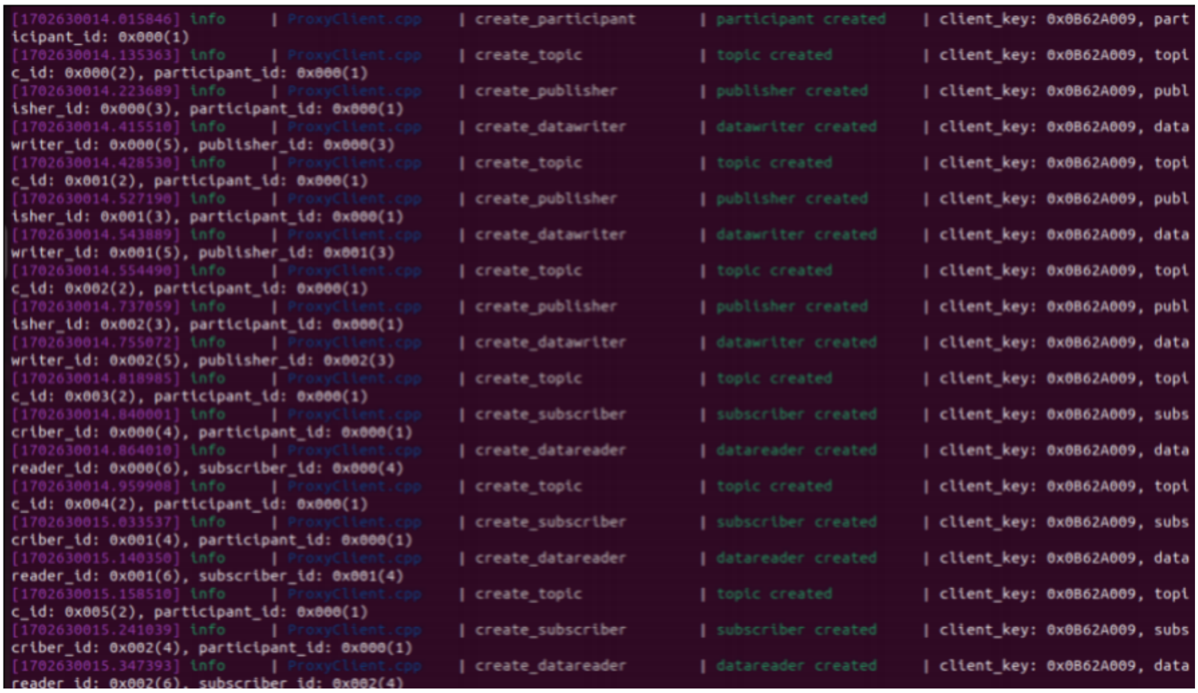
以配套虚拟机为例，输入以下指令启动代理，

```
sudo docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --privileged --net=host microros/micro-ros-agent:humble udp4 --port 8090 -v4
```



```
yahboom@yahboom-VM:~$ sudo docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm
 --privileged --net=host microros/micro-ros-agent:humble udp4 --port 8090 -v4
[1704167422.995513] info     | UDPv4AgentLinux.cpp | init                     |
running...              | port: 8090
[1704167422.995832] info     | Root.cpp            | set_verbose_level       | l
ogger setup             | verbose_level: 4
```

然后，打开小车开关，等待小车连接上代理，连接成功如下图所示，



```
[1702630014.015846] info     | ProxyClient.cpp     | create_participant     | participant created     | client_key: 0x0B62A009, part
icipant_id: 0x000(1)
[1702630014.135363] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x000(2), participant_id: 0x000(1)
[1702630014.223689] info     | ProxyClient.cpp     | create_publisher       | publisher created       | client_key: 0x0B62A009, publ
isher_id: 0x000(3), participant_id: 0x000(1)
[1702630014.415510] info     | ProxyClient.cpp     | create_datawriter      | datawriter created      | client_key: 0x0B62A009, data
writer_id: 0x000(5), publisher_id: 0x000(3)
[1702630014.428530] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x001(2), participant_id: 0x000(1)
[1702630014.527190] info     | ProxyClient.cpp     | create_publisher       | publisher created       | client_key: 0x0B62A009, publ
isher_id: 0x001(3), participant_id: 0x000(1)
[1702630014.543889] info     | ProxyClient.cpp     | create_datawriter      | datawriter created      | client_key: 0x0B62A009, data
writer_id: 0x001(5), publisher_id: 0x001(3)
[1702630014.554490] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x002(2), participant_id: 0x000(1)
[1702630014.737059] info     | ProxyClient.cpp     | create_publisher       | publisher created       | client_key: 0x0B62A009, publ
isher_id: 0x002(3), participant_id: 0x000(1)
[1702630014.755072] info     | ProxyClient.cpp     | create_datawriter      | datawriter created      | client_key: 0x0B62A009, data
writer_id: 0x002(5), publisher_id: 0x002(3)
[1702630014.818985] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x003(2), participant_id: 0x000(1)
[1702630014.840001] info     | ProxyClient.cpp     | create_subscriber      | subscriber created      | client_key: 0x0B62A009, subs
criber_id: 0x000(4), participant_id: 0x000(1)
[1702630014.864010] info     | ProxyClient.cpp     | create_datareader      | datareader created      | client_key: 0x0B62A009, data
reader_id: 0x000(6), subscriber_id: 0x000(4)
[1702630014.959908] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x004(2), participant_id: 0x000(1)
[1702630015.033537] info     | ProxyClient.cpp     | create_subscriber      | subscriber created      | client_key: 0x0B62A009, subs
criber_id: 0x001(4), participant_id: 0x000(1)
[1702630015.140350] info     | ProxyClient.cpp     | create_datareader      | datareader created      | client_key: 0x0B62A009, data
reader_id: 0x001(6), subscriber_id: 0x001(4)
[1702630015.158510] info     | ProxyClient.cpp     | create_topic           | topic created           | client_key: 0x0B62A009, topi
c_id: 0x005(2), participant_id: 0x000(1)
[1702630015.241039] info     | ProxyClient.cpp     | create_subscriber      | subscriber created      | client_key: 0x0B62A009, subs
criber_id: 0x002(4), participant_id: 0x000(1)
[1702630015.347393] info     | ProxyClient.cpp     | create_datareader      | datareader created      | client_key: 0x0B62A009, data
reader_id: 0x002(6), subscriber_id: 0x002(4)
```

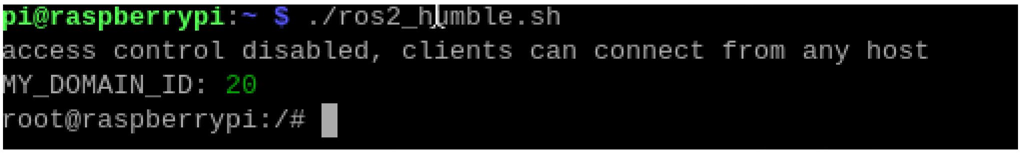
4、启动程序

4.1运行指令

如果是树莓派桌面版本和jetson nano桌面版本，需要提前进入docker，终端输入，

```
sh ros2_humble.sh
```

出现以下界面就是进入docker成功，



```
pi@raspberrypi:~ $ ./ros2_humble.sh
access control disabled, clients can connect from any host
MY_DOMAIN_ID: 20
root@raspberrypi:/#
```

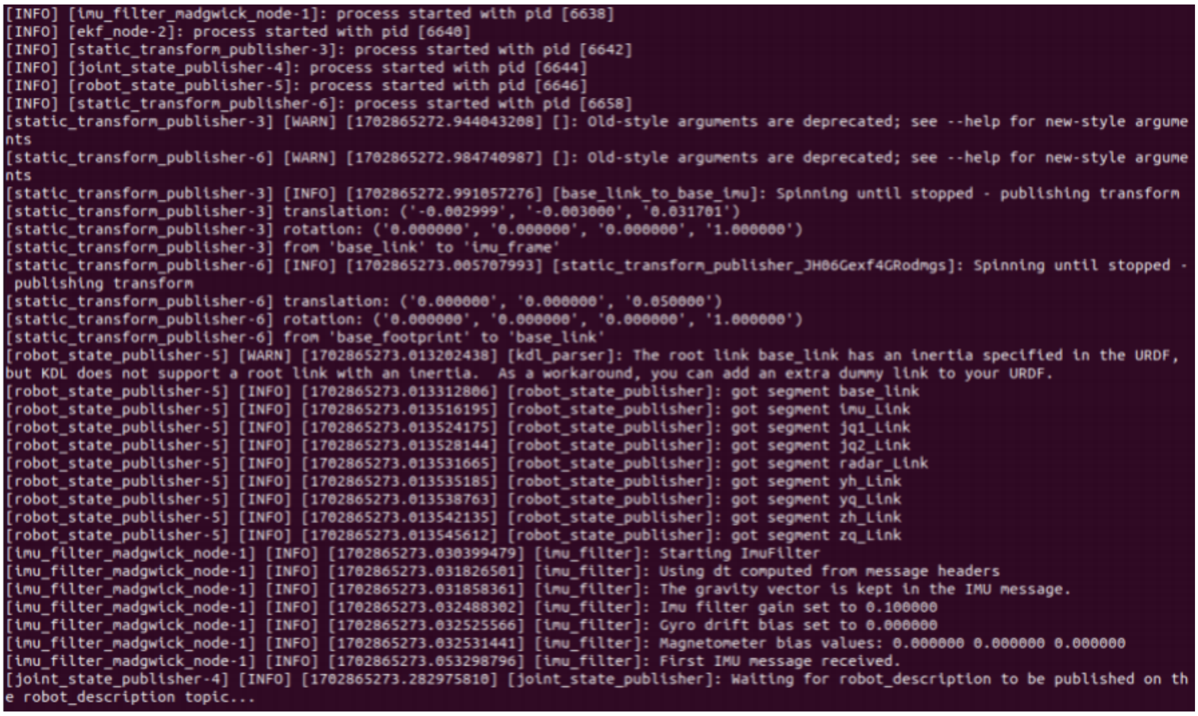
之后在docker里输入，**（查看【docker环境】章节，如何进入同一个docker终端）**

```
ros2 launch yahboomcar_bringup yahboomcar_bringup_launch.py      #底层数据程序
ros2 launch yahboomcar_nav display_launch.py        #建图可视化
ros2 launch yahboomcar_nav map_cartographer_launch.py          #建图节点
#键盘
ros2 run yahboomcar_ctrl yahboom_keyboard
#手柄
ros2 run yahboomcar_ctrl yahboom_joy
ros2 run joy joy_node
#保存地图
ros2 launch yahboomcar_nav save_map_launch.py
```

以配套的虚拟机为例，终端输入，

```
ros2 launch yahboomcar_bringup yahboomcar_bringup_launch.py
```

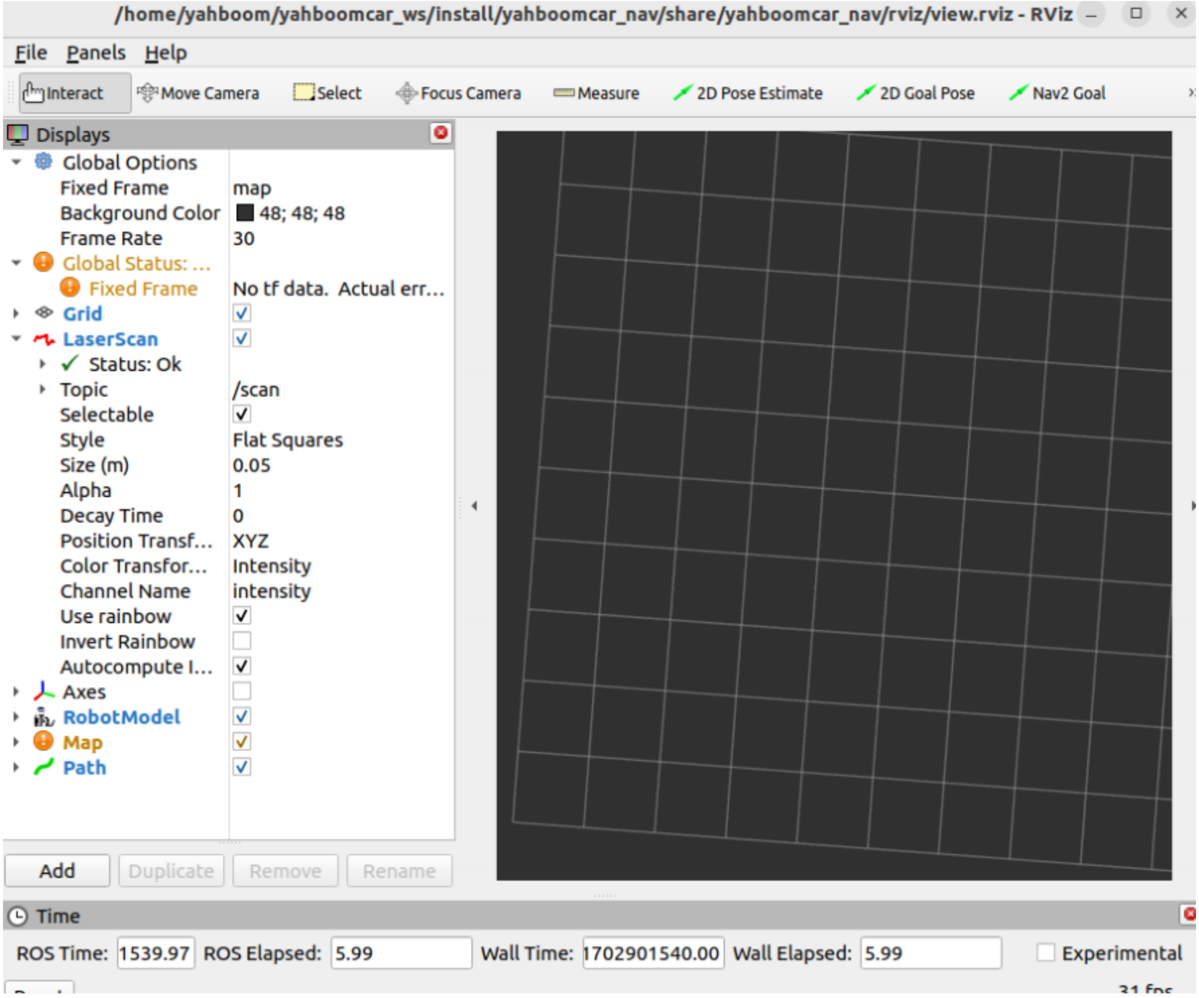
首先启动小车处理底层数据程序



```
[INFO] [imu_filter_madgwick_node-1]: process started with pid [6638]
[INFO] [ekf_node-2]: process started with pid [6640]
[INFO] [static_transform_publisher-3]: process started with pid [6642]
[INFO] [joint_state_publisher-4]: process started with pid [6644]
[INFO] [robot_state_publisher-5]: process started with pid [6646]
[INFO] [static_transform_publisher-6]: process started with pid [6658]
[static_transform_publisher-3] [WARN] [1702865272.944043208] []: Old-style arguments are deprecated; see --help for new-style argume
nts
[static_transform_publisher-6] [WARN] [1702865272.984740987] []: Old-style arguments are deprecated; see --help for new-style argume
nts
[static_transform_publisher-3] [INFO] [1702865272.991057276] [base_link_to_base_imu]: Spinning until stopped - publishing transform
[static_transform_publisher-3] translation: ('-0.002999', '-0.003000', '0.031701')
[static_transform_publisher-3] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-3] from 'base_link' to 'imu_frame'
[static_transform_publisher-6] [INFO] [1702865273.005707993] [static_transform_publisher_JH06Gexf4GRodmgs]: Spinning until stopped -
 publishing transform
[static_transform_publisher-6] translation: ('0.000000', '0.000000', '0.050000')
[static_transform_publisher-6] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-6] from 'base_footprint' to 'base_link'
[robot_state_publisher-5] [WARN] [1702865273.013202438] [kdl_parser]: The root link base_link has an inertia specified in the URDF,
but KDL does not support a root link with an inertia.  As a workaround, you can add an extra dummy link to your URDF.
[robot_state_publisher-5] [INFO] [1702865273.013312806] [robot_state_publisher]: got segment base_link
[robot_state_publisher-5] [INFO] [1702865273.013516195] [robot_state_publisher]: got segment imu_Link
[robot_state_publisher-5] [INFO] [1702865273.013524175] [robot_state_publisher]: got segment jq1_Link
[robot_state_publisher-5] [INFO] [1702865273.013528144] [robot_state_publisher]: got segment jq2_Link
[robot_state_publisher-5] [INFO] [1702865273.013531665] [robot_state_publisher]: got segment radar_Link
[robot_state_publisher-5] [INFO] [1702865273.013535185] [robot_state_publisher]: got segment yh_Link
[robot_state_publisher-5] [INFO] [1702865273.013538763] [robot_state_publisher]: got segment yq_Link
[robot_state_publisher-5] [INFO] [1702865273.013542135] [robot_state_publisher]: got segment zh_Link
[robot_state_publisher-5] [INFO] [1702865273.013545612] [robot_state_publisher]: got segment zq_Link
[imu_filter_madgwick_node-1] [INFO] [1702865273.030399479] [imu_filter]: Starting ImuFilter
[imu_filter_madgwick_node-1] [INFO] [1702865273.031826501] [imu_filter]: Using dt computed from message headers
[imu_filter_madgwick_node-1] [INFO] [1702865273.031858361] [imu_filter]: The gravity vector is kept in the IMU message.
[imu_filter_madgwick_node-1] [INFO] [1702865273.032488302] [imu_filter]: Imu filter gain set to 0.100000
[imu_filter_madgwick_node-1] [INFO] [1702865273.032525566] [imu_filter]: Gyro drift bias set to 0.000000
[imu_filter_madgwick_node-1] [INFO] [1702865273.032531441] [imu_filter]: Magnetometer bias values: 0.000000 0.000000 0.000000
[imu_filter_madgwick_node-1] [INFO] [1702865273.053298796] [imu_filter]: First IMU message received.
[joint_state_publisher-4] [INFO] [1702865273.282975810] [joint_state_publisher]: Waiting for robot_description to be published on th
e robot_description topic...
```

然后，启动rviz，可视化建图，终端输入，

```
ros2 launch yahboomcar_nav display_launch.py
```



/home/yahboom/yahboomcar_ws/install/yahboomcar_nav/share/yahboomcar_nav/rviz/view.rviz - RViz

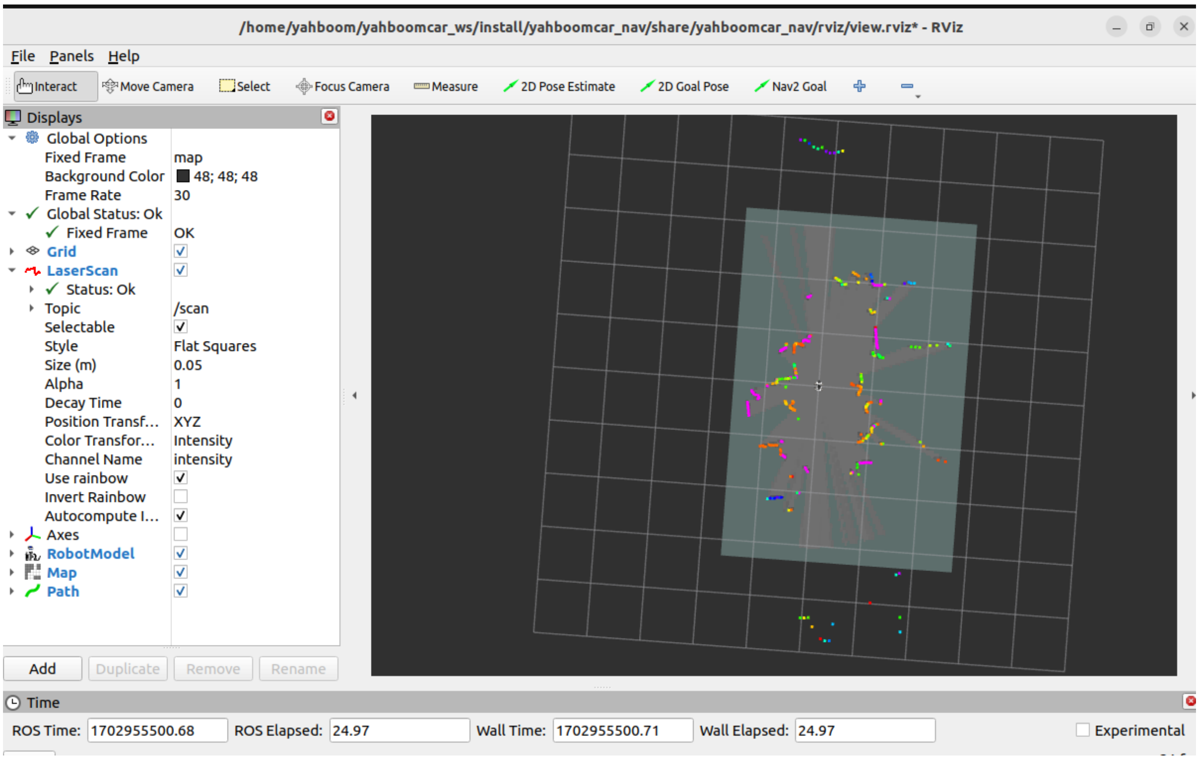
Global Options
Fixed Frame: map
Background Color: 48; 48; 48
Frame Rate: 30
Global Status: ...
Fixed Frame: No tf data. Actual err...
Grid ☑
LaserScan ☑
Status: Ok
Topic: /scan
Selectable ☑
Style: Flat Squares
Size (m): 0.05
Alpha: 1
Decay Time: 0
Position Transf...: XYZ
Color Transfor...: Intensity
Channel Name: intensity
Use rainbow ☑
Invert Rainbow □
Autocompute I... ☑
Axes □
RobotModel ☑
Map ☑
Path ☑

ROS Time: 1539.97 ROS Elapsed: 5.99 Wall Time: 1702901540.00 Wall Elapsed: 5.99 □ Experimental

31 fps

此时还没运行建图节点，所以没有数据。接下来运行建图节点，终端输入，

```
ros2 launch yahboomcar_nav map_cartographer_launch.py
```



/home/yahboom/yahboomcar_ws/install/yahboomcar_nav/share/yahboomcar_nav/rviz/view.rviz* - RViz

Global Options
Fixed Frame: map
Background Color: 48; 48; 48
Frame Rate: 30
Global Status: Ok
Fixed Frame: OK
Grid ☑
LaserScan ☑
Status: Ok
Topic: /scan
Selectable ☑
Style: Flat Squares
Size (m): 0.05
Alpha: 1
Decay Time: 0
Position Transf...: XYZ
Color Transfor...: Intensity
Channel Name: intensity
Use rainbow ☑
Invert Rainbow □
Autocompute I... ☑
Axes □
RobotModel ☑
Map ☑
Path ☑

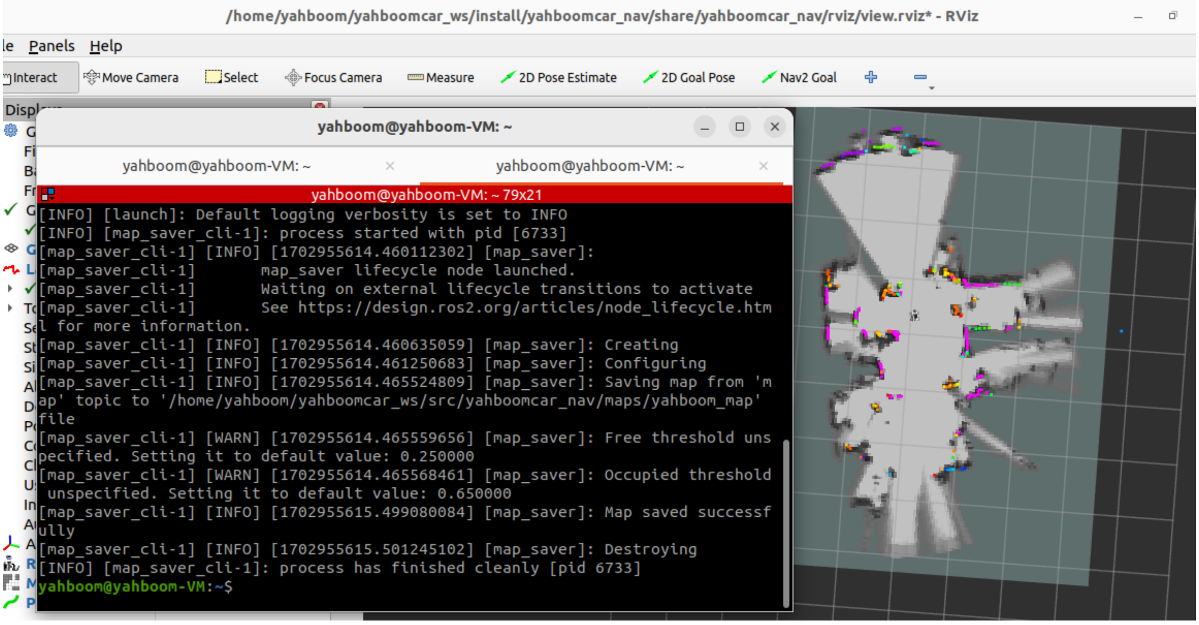
ROS Time: 1702955500.68 ROS Elapsed: 24.97 Wall Time: 1702955500.71 Wall Elapsed: 24.97 □ Experimental

然后运行手柄控制或者键盘控制，二选一，终端输入，

```
#键盘
ros2 run yahboomcar_ctrl yahboom_keyboard
#手柄
ros2 run yahboomcar_ctrl yahboom_joy
ros2 run joy joy_node
```

然后控制小车，缓慢的走完需要建图的区域，建图完毕后，输入以下指令保存地图，终端输入，

```
ros2 launch yahboomcar_nav save_map_launch.py
```



```
[INFO] [launch]: Default logging verbosity is set to INFO
[INFO] [map_saver_cli-1]: process started with pid [6733]
[map_saver_cli-1] [INFO] [1702955614.460112302] [map_saver]:
[map_saver_cli-1]       map_saver lifecycle node launched.
[map_saver_cli-1]       Waiting on external lifecycle transitions to activate
[map_saver_cli-1]       See https://design.ros2.org/articles/node_lifecycle.htm
l for more information.
[map_saver_cli-1] [INFO] [1702955614.460635059] [map_saver]: Creating
[map_saver_cli-1] [INFO] [1702955614.461250683] [map_saver]: Configuring
[map_saver_cli-1] [INFO] [1702955614.465524809] [map_saver]: Saving map from 'm
ap' topic to '/home/yahboom/yahboomcar_ws/src/yahboomcar_nav/maps/yahboom_map'
file
[map_saver_cli-1] [WARN] [1702955614.465559656] [map_saver]: Free threshold uns
pecified. Setting it to default value: 0.250000
[map_saver_cli-1] [WARN] [1702955614.465568461] [map_saver]: Occupied threshold
 unspecified. Setting it to default value: 0.650000
[map_saver_cli-1] [INFO] [1702955615.499080084] [map_saver]: Map saved successf
ully
[map_saver_cli-1] [INFO] [1702955615.501245102] [map_saver]: Destroying
[INFO] [map_saver_cli-1]: process has finished cleanly [pid 6733]
yahboom@yahboom-VM:~$
```

会保存一个命名为yahboom_map的地图，这个地图保存在，

以配套的虚拟机为例代码路径：

```
/home/yahboom/yahboomcar_ws/src/yahboomcar_nav/maps
```

jetson nano代码路径：

```
/root/yahboomcar_ws/src/yahboomcar_nav/maps
```

树莓派代码路径：

```
/root/yahboomcar_ws/src/yahboomcar_nav/maps
```

会有两个文件生成，一个是yahboom_map.pgm，一个是yahboom_map.yaml，看下yaml的内容，

```
image: yahboom_map.pgm
mode: trinary
resolution: 0.05
origin: [-10, -10, 0]
negate: 0
occupied_thresh: 0.65
free_thresh: 0.25
```

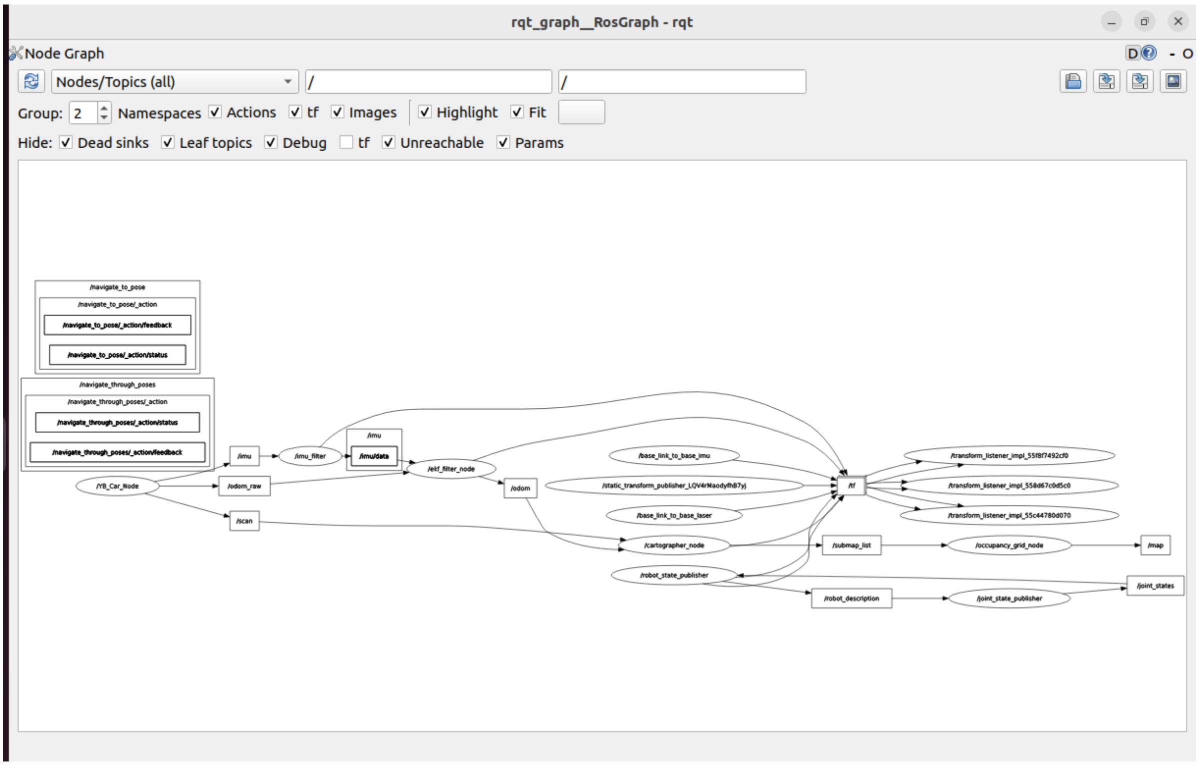
- image：表示地图的图片，也就是yahboom_map.pgm

- mode：该属性可以是trinary、scale或者raw之一，取决于所选择的mode，trinary模式是默认模式
- resolution：地图的分辨率， 米/像素
- 地图左下角的 2D 位姿(x,y,yaw), 这里的yaw是逆时针方向旋转的（yaw=0 表示没有旋转）。目前系统中的很多部分会忽略yaw值。
- negate：是否颠倒 白/黑 、自由/占用 的意义（阈值的解释不受影响）
- occupied_thresh：占用概率大于这个阈值的的像素，会被认为是完全占用。
- free_thresh：占用概率小于这个阈值的的像素，会被认为是完全自由。

5、查看节点通讯图

终端输入，

```
ros2 run rqt_graph rqt_graph
```



Nodes/Topics (all)　/　/

Group: 2　Namespaces ☑ Actions ☑ tf ☑ Images ☑ Highlight ☑ Fit

Hide: ☑ Dead sinks ☑ Leaf topics ☑ Debug □ tf ☑ Unreachable ☑ Params

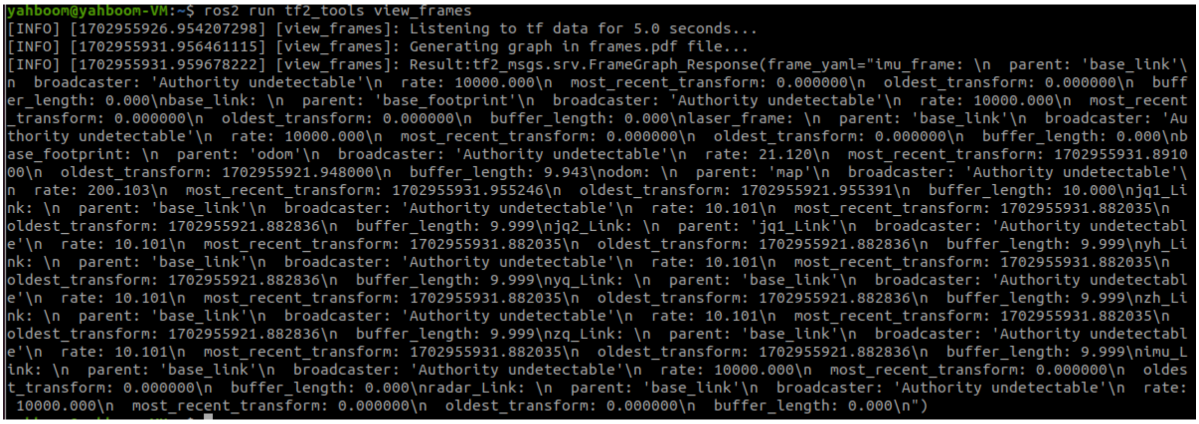
rqt_graph__RosGraph - rqt
Node Graph
D
/navigate_to_pose
/navigate_to_pose/_action
/navigate_to_pose/_action/feedback
/navigate_to_pose/_action/status
/navigate_through_poses
/navigate_through_poses/_action
/navigate_through_poses/_action/status
/navigate_through_poses/_action/feedback
/YB_Car_Node
/imu
/odom_raw
/scan
/imu_filter
/imu
/imu/data
/ekf_filter_node
/odom
/base_link_to_base_imu
/static_transform_publisher_LQV4rMaodyfhB7yj
/base_link_to_base_laser
/cartographer_node
/robot_state_publisher
/tf
/transform_listener_impl_55f8f7492cf0
/transform_listener_impl_558d67c0d5c0
/transform_listener_impl_55c44780d070
/submap_list
/occupancy_grid_node
/map
/robot_description
/joint_state_publisher
/joint_states

如果一开始没有显示，选择【Nodes/Topics(all)】，然后点击左上角的刷新按钮。

6、查看TF树

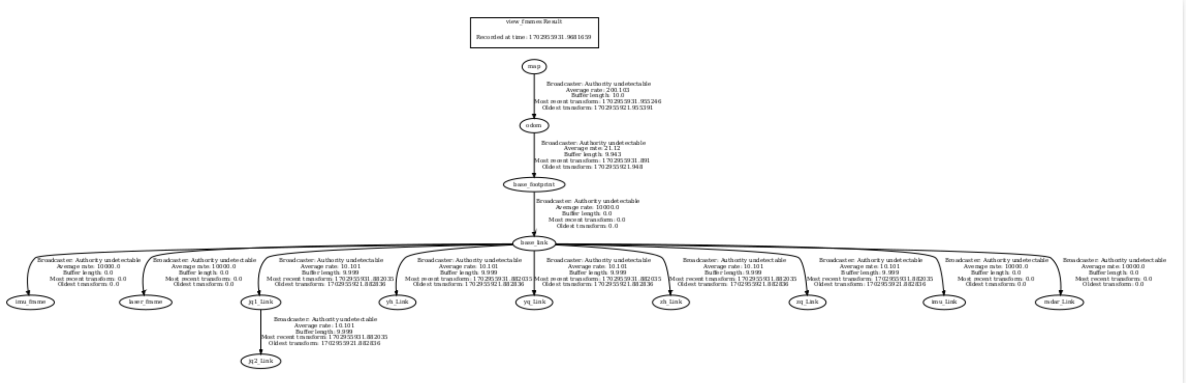
终端输入，

```
ros2 run tf2_tools view_frames
```



```
yahboom@yahboom-VM:~$ ros2 run tf2_tools view_frames
[INFO] [1702955926.954207298] [view_frames]: Listening to tf data for 5.0 seconds...
[INFO] [1702955931.956461115] [view_frames]: Generating graph in frames.pdf file...
[INFO] [1702955931.959678222] [view_frames]: Result:tf2_msgs.srv.FrameGraph_Response(frame_yaml="imu_frame: \n  parent: 'base_link'\
n  broadcaster: 'Authority undetectable'\n  rate: 10000.000\n  most_recent_transform: 0.000000\n  oldest_transform: 0.000000\n  buff
er_length: 0.000\nbase_link: \n  parent: 'base_footprint'\n  broadcaster: 'Authority undetectable'\n  rate: 10000.000\n  most_recent
_transform: 0.000000\n  oldest_transform: 0.000000\n  buffer_length: 0.000\nlaser_frame: \n  parent: 'base_link'\n  broadcaster: 'Au
thority undetectable'\n  rate: 10000.000\n  most_recent_transform: 0.000000\n  oldest_transform: 0.000000\n  buffer_length: 0.000\nb
ase_footprint: \n  parent: 'odom'\n  broadcaster: 'Authority undetectable'\n  rate: 21.120\n  most_recent_transform: 1702955931.8910
00\n  oldest_transform: 1702955921.948000\n  buffer_length: 9.943\nodom: \n  parent: 'map'\n  broadcaster: 'Authority undetectable'\
n  rate: 200.103\n  most_recent_transform: 1702955931.955246\n  oldest_transform: 1702955921.955391\n  buffer_length: 10.000\njq1_Li
nk: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectable'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n
oldest_transform: 1702955921.882836\n  buffer_length: 9.999\njq2_Link: \n  parent: 'jq1_Link'\n  broadcaster: 'Authority undetectabl
e'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n  oldest_transform: 1702955921.882836\n  buffer_length: 9.999\nyh_Li
nk: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectable'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n
oldest_transform: 1702955921.882836\n  buffer_length: 9.999\nyq_Link: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectabl
e'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n  oldest_transform: 1702955921.882836\n  buffer_length: 9.999\nzh_Li
nk: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectable'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n
oldest_transform: 1702955921.882836\n  buffer_length: 9.999\nzq_Link: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectabl
e'\n  rate: 10.101\n  most_recent_transform: 1702955931.882035\n  oldest_transform: 1702955921.882836\n  buffer_length: 9.999\nimu_L
ink: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectable'\n  rate: 10000.000\n  most_recent_transform: 0.000000\n  oldes
t_transform: 0.000000\n  buffer_length: 0.000\nradar_Link: \n  parent: 'base_link'\n  broadcaster: 'Authority undetectable'\n  rate:
 10000.000\n  most_recent_transform: 0.000000\n  oldest_transform: 0.000000\n  buffer_length: 0.000\n")
```

运行完毕后，会在终端的目录下生成两个文件分别是.gv和.pdf文件，其中的pdf文件就是TF树。


view_frames Result
Recorded at time: 1702955931.9681659
map
Broadcaster: Authority undetectable
Average rate: 200.103
Buffer length: 10.0
Most recent transform: 1702955931.955246
Oldest transform: 1702955921.955391
odom
Broadcaster: Authority undetectable
Average rate: 21.12
Buffer length: 9.943
Most recent transform: 1702955931.891
Oldest transform: 1702955921.948
base_footprint
Broadcaster: Authority undetectable
Average rate: 10000.0
Buffer length: 0.0
Most recent transform: 0.0
Oldest transform: 0.0
base_link
imu_frame
laser_frame
jq1_link
yh_link
yq_link
zh_link
zq_link
imu_link
radar_link
jq2_link

7、代码解析

以虚拟机为例，这里只说明建图的map_gmapping_launch.py，这个文件路径是，

```
/home/yahboom/yahboomcar_ws/src/yahboomcar_nav/launch
```

jetson nano代码路径：

```
/root/yahboomcar_ws/src/yahboomcar_nav/launch
```

树莓派代码路径：

```
/root/yahboomcar_ws/src/yahboomcar_nav/launch
```

map_gmapping_launch.py

```
import os
from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch_ros.actions import Node


def generate_launch_description():
    package_launch_path
=os.path.join(get_package_share_directory('yahboomcar_nav'), 'launch')

    cartographer_launch = IncludeLaunchDescription(PythonLaunchDescriptionSource(
        [package_launch_path, '/cartographer_launch.py'])
    )
    base_link_to_laser_tf_node = Node(
     package='tf2_ros',
     executable='static_transform_publisher',
     name='base_link_to_base_laser',
     arguments=['-0.0046412', '0' ,
'0.094079','0','0','0','base_link','laser_frame']
    )
    return LaunchDescription([cartographer_launch,base_link_to_laser_tf_node])
```

这里运行了一个launch文件-cartographer_launch和一个发布静态变换的节点-base_link_to_laser_tf_node，

以虚拟机为例，详细看下slam_gmapping_launch，该文件位于，

```
/home/yahboom/gmapping_ws/src/slam_gmapping/launch
```

jetson nano代码路径：

```
/root/gmapping_ws/src/slam_gmapping/launch
```

树莓派代码路径：

```
/root/gmapping_ws/src/slam_gmapping/launch
```

slam_gmapping.launch.py，

```
import os
from ament_index_python.packages import get_package_share_directory
from launch import LaunchDescription
from launch.actions import DeclareLaunchArgument
from launch_ros.actions import Node
from launch.substitutions import LaunchConfiguration
from launch.actions import IncludeLaunchDescription
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch.substitutions import ThisLaunchFileDir


def generate_launch_description():
    use_sim_time = LaunchConfiguration('use_sim_time', default='false')
    package_path = get_package_share_directory('yahboomcar_nav')
    configuration_directory = LaunchConfiguration('configuration_directory',
default=os.path.join(
                                                  package_path, 'params'))
    configuration_basename = LaunchConfiguration('configuration_basename',
default='lds_2d.lua')

    resolution = LaunchConfiguration('resolution', default='0.05')
    publish_period_sec = LaunchConfiguration(
        'publish_period_sec', default='1.0')

    return LaunchDescription([
        DeclareLaunchArgument(
            'configuration_directory',
            default_value=configuration_directory,
            description='Full path to config file to load'),
        DeclareLaunchArgument(
            'configuration_basename',
            default_value=configuration_basename,
            description='Name of lua file for cartographer'),
        DeclareLaunchArgument(
            'use_sim_time',
            default_value='false',
            description='Use simulation (Gazebo) clock if true'),

        Node(
            package='cartographer_ros',
            executable='cartographer_node',
            name='cartographer_node',
            output='screen',
            parameters=[{'use_sim_time': use_sim_time}],
            arguments=['-configuration_directory', configuration_directory,
                       '-configuration_basename', configuration_basename],
            remappings=[('/odom','/odom')]
                       ),

        DeclareLaunchArgument(
            'resolution',
            default_value=resolution,
            description='Resolution of a grid cell in the published occupancy
grid'),

        DeclareLaunchArgument(
            'publish_period_sec',
            default_value=publish_period_sec,
            description='OccupancyGrid publishing period'),

        IncludeLaunchDescription(
            PythonLaunchDescriptionSource(
                [ThisLaunchFileDir(), '/occupancy_grid_launch.py']),
            launch_arguments={'use_sim_time': use_sim_time, 'resolution':
resolution,
                              'publish_period_sec': publish_period_sec}.items(),
        ),
    ])
```

这里主要是运行了cartographer_node建图节点以及occupancy_grid_launch.py，另外加载了参数配置文件，

该文件位于（以配套虚拟机为例），

```
/home/yahboom/gmapping_ws/src/slam_gmapping/params
```

jetson nano代码路径：

```
/root/gmapping_ws/src/slam_gmapping/params
```

树莓派代码路径：

```
/root/gmapping_ws/src/slam_gmapping/params
```

lds_2d.lua，

```
include "map_builder.lua"
include "trajectory_builder.lua"

options = {
  map_builder = MAP_BUILDER,
  trajectory_builder = TRAJECTORY_BUILDER,
  map_frame = "map",
  tracking_frame = "base_footprint",
  published_frame = "odom",
  odom_frame = "odom",
  provide_odom_frame = false,
  publish_frame_projected_to_2d = false,
  use_odometry = true,
  use_nav_sat = false,
  use_landmarks = false,
  num_laser_scans = 1,
  num_multi_echo_laser_scans = 0,
  num_subdivisions_per_laser_scan = 1,
  num_point_clouds = 0,
  lookup_transform_timeout_sec = 0.2,
  submap_publish_period_sec = 0.3,
  pose_publish_period_sec = 5e-3,
  trajectory_publish_period_sec = 30e-3,
  rangefinder_sampling_ratio = 1.,
  odometry_sampling_ratio = 1.,
  fixed_frame_pose_sampling_ratio = 1.,
  imu_sampling_ratio = 1.,
  landmarks_sampling_ratio = 1.,
}


MAP_BUILDER.use_trajectory_builder_2d = true

TRAJECTORY_BUILDER_2D.use_imu_data = false
TRAJECTORY_BUILDER_2D.min_range = 0.10
TRAJECTORY_BUILDER_2D.max_range = 3.5
TRAJECTORY_BUILDER_2D.missing_data_ray_length = 3.
TRAJECTORY_BUILDER_2D.use_online_correlative_scan_matching = true
TRAJECTORY_BUILDER_2D.motion_filter.max_angle_radians = math.rad(0.1)

POSE_GRAPH.constraint_builder.min_score = 0.65
POSE_GRAPH.constraint_builder.global_localization_min_score = 0.7

return options
```