4、Robot state estimation

1、Program function description

After the program runs, combined with the IMU data and speed vel data read to the ROS expansion board, an ODOM data that integrates IMU and ODOM data is output, which is applied when doing positioning functions.

2、Program code reference path

Raspberry Pi PI5 master needs to enter the docker container first, Orin motherboard does not need to enter,

the location of the source code of this function is located at,

```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_bringup/launch/yahboomcar_bringup_R2_launch.py
```

ekf fusion program code reference path,

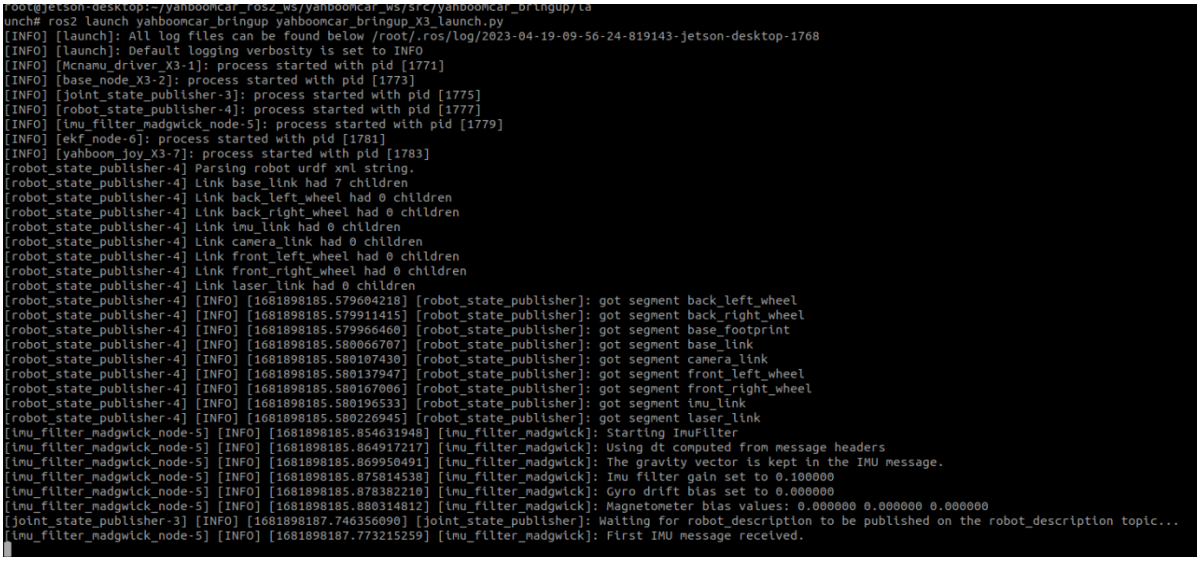
```
~/yahboomcar_ros2_ws/yahboomcar_ws/src/yahboomcar_bringup/launch/yahboomcar_bringup_R2_launch.py
```

3、The program starts

3.1、start the command

After entering the docker container, according to the actual model, the terminal input,

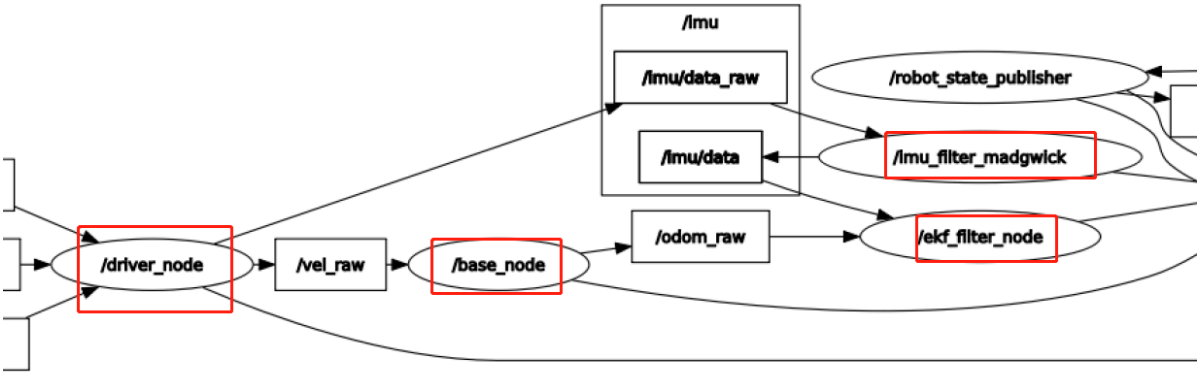
```
ros2 launch yahboomcar_bringup yahboomcar_bringup_R2_launch.py
```



3.2、View the node communication graph

Open the terminal and enter the container,

```
ros2 run rqt_graph rqt_graph
```



Mainly look at the node input and output of the red box in the figure above, you can see that / ekf_filter_node receive odom_raw data and imu_data data for fusion, and finally the output publishes an ODOM data, we can view through the ros2 node tool, terminal input,

```
ros2 node info /ekf_filter_node
```

```
root@jetson-desktop:~# ros2 node info /ekf_filter_node
/ekf_filter_node
  Subscribers:
    /example/odom2: nav_msgs/msg/Odometry
    /example/pose: geometry_msgs/msg/PoseWithCovarianceStamped
    /example/twist: geometry_msgs/msg/TwistWithCovarianceStamped
    /imu/data: sensor_msgs/msg/Imu
    /odom_raw: nav_msgs/msg/Odometry
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /set_pose: geometry_msgs/msg/PoseWithCovarianceStamped
  Publishers:
    /diagnostics: diagnostic_msgs/msg/DiagnosticArray
    /odom: nav_msgs/msg/Odometry
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
    /tf: tf2_msgs/msg/TFMessage
  Service Servers:
    /ekf_filter_node/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /ekf_filter_node/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /ekf_filter_node/get_parameters: rcl_interfaces/srv/GetParameters
    /ekf_filter_node/list_parameters: rcl_interfaces/srv/ListParameters
    /ekf_filter_node/set_parameters: rcl_interfaces/srv/SetParameters
    /ekf_filter_node/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
    /enable: std_srvs/srv/Empty
    /set_pose: robot_localization/srv/SetPose
    /toggle: robot_localization/srv/ToggleFilterProcessing
  Service Clients:

  Action Servers:

  Action Clients:
```

4、Launch file parsing

Let's take a look at the main relevant nodes of the launch file,

- /driver_node: Start the chassis of the trolley, obtain the speed vel data of the wheels, publish to the /base_node node, obtain IMU data, and publish to the / Imu_filter_madgwick node;
- /base_node: Receive vel data, convert it into odom_raw data through calculation, and publish it to /ekf_filter_node nodes;

- /Imu_filter_madgwick: Receive the IMU data released by the chassis, filter it through its own algorithm, and publish the filtered IMU/Data data to the /ekf_filter_node node;
- /ekf_filter_node: Receive the ODOM data published by the /base_node node and the IMU/Data data released by the /Imu_filter_madgwick, and release the ODOM data after fusing it through its own algorithm.