

# Robot state estimation

Note: The virtual machine needs to be in the same LAN as the car, and the ROS\_DOMAIN\_ID needs to be consistent. You can check [Must-read before use] to set the IP and ROS\_DOMAIN\_ID on the board.

## 1. Program function description

After the program is started, it will subscribe to imu and odom data, filter out part of the imu data, and then merge it with the odom data, and finally output a fused odom data to estimate the robot's state. This data is mostly used in mapping and navigation.

## 2. Start and connect the agent

Take the matching virtual machine as an example, enter the following command to start the agent,

```
sudo docker run -it --rm -v /dev:/dev -v /dev/shm:/dev/shm --privileged --net=host microros/micro-ros-agent:humble udp4 --port 8899 -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 8899 -v4
[1735179211.772044] info | UDPv4AgentLinux.cpp | init |
running... | port: 8899
[1735179211.772581] info | Root.cpp | set_verbose_level | 1
logger setup | verbose_level: 4
```

Then, turn on the car switch and wait for the car to connect to the agent. The connection is successful as shown in the figure below,

```
[1735179211.772044] info | UDPv4AgentLinux.cpp | init | running... | port: 8899
[1735179211.772581] info | Root.cpp | set_verbose_level | logger setup | verbose_level: 4
[1735179325.739277] info | Root.cpp | create_client | create | client_key: 0x0E5C3397, sess
ion_id: 0x81
[1735179325.739348] info | SessionManager.hpp | establish_session | session established | client_key: 0x0E5C3397, addr
ess: 192.168.2.102:49954
[1735179325.971694] info | ProxyClient.cpp | create_participant | participant created | client_key: 0x0E5C3397, part
icipant_id: 0x000(1)
[1735179326.046043] info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0E5C3397, topl
c_id: 0x000(2), participant_id: 0x000(1)
[1735179326.159287] info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0E5C3397, publ
isher_id: 0x000(3), participant_id: 0x000(1)
[1735179326.176344] info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0E5C3397, data
writer_id: 0x000(5), publisher_id: 0x000(3)
[1735179326.184566] info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0E5C3397, topl
c_id: 0x001(2), participant_id: 0x000(1)
[1735179326.263761] info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0E5C3397, publ
isher_id: 0x001(3), participant_id: 0x000(1)
[1735179326.276817] info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0E5C3397, data
writer_id: 0x001(5), publisher_id: 0x001(3)
[1735179326.285996] info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0E5C3397, topl
c_id: 0x002(2), participant_id: 0x000(1)
[1735179326.345401] info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0E5C3397, publ
isher_id: 0x002(3), participant_id: 0x000(1)
[1735179326.365619] info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0E5C3397, data
writer_id: 0x002(5), publisher_id: 0x002(3)
[1735179326.372863] info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0E5C3397, topl
c_id: 0x003(2), participant_id: 0x000(1)
[1735179326.379913] info | ProxyClient.cpp | create_publisher | publisher created | client_key: 0x0E5C3397, publ
isher_id: 0x003(3), participant_id: 0x000(1)
[1735179326.448851] info | ProxyClient.cpp | create_datawriter | datawriter created | client_key: 0x0E5C3397, data
writer_id: 0x003(5), publisher_id: 0x003(3)
[1735179326.548363] info | ProxyClient.cpp | create_topic | topic created | client_key: 0x0E5C3397, topl
c_id: 0x004(2), participant_id: 0x000(1)
[1735179326.565153] info | ProxyClient.cpp | create_subscriber | subscriber created | client_key: 0x0E5C3397, subs
criber_id: 0x000(4), participant_id: 0x000(1)
[1735179326.574254] info | ProxyClient.cpp | create_datareader | datareader created | client_key: 0x0E5C3397, data
reader_id: 0x000(6), subscriber_id: 0x000(4)
```

## 3. Start the program

### 3.1 Run command

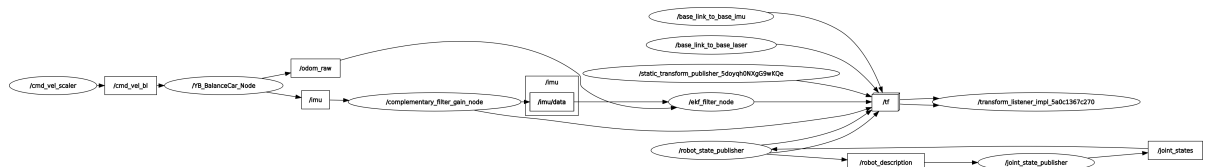
Take the matching virtual machine as an example, input in the terminal,

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

```
yahboom@yahboom-VN: $ ros2 launch yahboomcar_bringup yahboomcar_bringup_launch.py
[INFO] [launch]: all log files can be found below /home/yahboom/.ros/log/2024-12-27-16-20-09-700601-yahboom-VN-5819
[INFO] [launch]: Default logging verbosity is set to INFO
-----robot_type = stm32v2-----
[INFO] [ekf_node-8]: process started with pid [5848]
[INFO] [complementary_filter_node-1]: process started with pid [5821]
[INFO] [static_transform_publisher-2]: process started with pid [5823]
[INFO] [static_transform_publisher-3]: process started with pid [5825]
[INFO] [joint_state_publisher-4]: process started with pid [5827]
[INFO] [robot_state_publisher-5]: process started with pid [5829]
[INFO] [static_transform_publisher-6]: process started with pid [5833]
[INFO] [cndvel2b1-7]: process started with pid [5835]
[INFO] [ekf_node-8]: process started with pid [5848]
[static_transform_publisher-2] [WARN] [1735287610.032940353] []: Old-style arguments are deprecated; see --help for new-style arguments
[static_transform_publisher-3] [WARN] [1735287610.033194674] []: Old-style arguments are deprecated; see --help for new-style arguments
[static_transform_publisher-6] [WARN] [1735287610.107820208] []: Old-style arguments are deprecated; see --help for new-style arguments
[static_transform_publisher-6] [INFO] [1735287610.177399661] [static_transform_publisher_B4wjrlorhGwZqaw]: Spinning until stopped - publishing transform
[static_transform_publisher-6] translation: ('0.000000', '0.000000', '0.033500')
[static_transform_publisher-6] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-6] from 'base_footprint' to 'base_link'
[static_transform_publisher-3] [INFO] [1735287610.183102668] [base_link to base_laser]: Spinning until stopped - publishing transform
[static_transform_publisher-3] translation: ('0.000000', '0.000000', '0.130000')
[static_transform_publisher-3] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-3] from 'base_link' to 'laser_frame'
[static_transform_publisher-2] [INFO] [1735287610.209628170] [base_link to base_lmu]: Spinning until stopped - publishing transform
[static_transform_publisher-2] translation: ('0.000000', '0.016325', '0.080691')
[static_transform_publisher-2] rotation: ('0.000000', '0.000000', '0.000000', '1.000000')
[static_transform_publisher-2] from 'base_link' to 'lmu_frame'
[complementary_filter_node-1] [INFO] [1735287610.244833919] [complementary_filter_gain_node]: Starting ComplementaryFilterROS
[robot_state_publisher-5] [WARN] [1735287610.348195700] [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] [1735287610.348318344] [robot_state_publisher]: got segment Camera_Link
[robot_state_publisher-5] [INFO] [1735287610.348378990] [robot_state_publisher]: got segment LWheel_Link
[robot_state_publisher-5] [INFO] [1735287610.348383531] [robot_state_publisher]: got segment RWheel_Link
[robot_state_publisher-5] [INFO] [1735287610.348386795] [robot_state_publisher]: got segment base_link
[robot_state_publisher-4] [INFO] [1735287611.258169340] [joint_state_publisher]: Waiting for robot_description to be published on the robot_description topic...
[cndvel2b1-7] [INFO] [1735287611.264444558] [cnd_vel_scaler]: mode default...
```

Enter the following command to view the communication graph between nodes,

```
ros2 run rqt_graph rqt_graph
```



If it is not displayed at the beginning, select [Nodes/Topics(all)] and click the refresh button in the upper left corner.

The fused node is /ekf\_filter\_node. You can query the relevant information of this node by inputting in the terminal,

```
ros2 node info /ekf_filter_node
```

```

yahboom@yahboom-VM:~$ ros2 node info /ekf_filter_node
/ekf_filter_node
Subscribers:
  /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
  /reset: std_srvs/srv/Empty
  /set_pose: robot_localization/srv/SetPose
  /toggle: robot_localization/srv/ToggleFilterProcessing
Service Clients:

Action Servers:

Action Clients:

```

Combined with the node communication diagram above, it can be seen that the node subscribes to /imu/data and /odom\_raw data, and then publishes a /odom data.

## 4. Parse the launch file

Take the matching virtual machine as an example code path:

```
/home/yahboom/yahboomcar_ws/src/yahboomcar_bringup/launch
```

yahboomcar\_bringup\_launch.py

```

from ament_index_python.packages import get_package_share_path
from launch import LaunchDescription
from launch_ros.actions import Node
import os

from ament_index_python.packages import get_package_share_directory
from launch.actions import IncludeLaunchDescription, DeclareLaunchArgument
from launch.launch_description_sources import PythonLaunchDescriptionSource
from launch.substitutions import LaunchConfiguration

print("-----robot_type = stm32v2-----")

def generate_launch_description():
    imu_filter_node = IncludeLaunchDescription(
        PythonLaunchDescriptionSource([os.path.join(
            get_package_share_directory('imu_complementary_filter'), 'launch'),
            '/complementary_filter.launch.py'])
    )

    ekf_params_file = LaunchConfiguration('ekf_params_file')
    ekf_node = IncludeLaunchDescription(
        PythonLaunchDescriptionSource([os.path.join(

```

```

        get_package_share_directory('robot_localization'), 'launch'),
        '/ekf.launch.py']],
        launch_arguments={ 'ekf_params_file': ekf_params_file}.items()
    )

    description_launch = IncludeLaunchDescription(
        PythonLaunchDescriptionSource([os.path.join(
            get_package_share_directory('yahboomcar_description'), 'launch'),
            '/description_launch.py'])
    )

    base_link_to_imu_tf_node = Node(
        package='tf2_ros',
        executable='static_transform_publisher',
        name='base_link_to_base_imu',
        #arguments=['0.0', '0.016325', '0.080691', '0', '0', '0', 'base_link',
'mpu6050_frame']
        arguments=['0.0', '0.016325', '0.080691', '0', '0', '0', 'base_link',
'imu_frame']
    )

    base_link_to_laser_tf_node = Node(
        package='tf2_ros',
        executable='static_transform_publisher',
        name='base_link_to_base_laser',
        arguments=['0.0', '0.0', '0.138', '0', '0', '0', 'base_link',
'laser_frame']
    )

    #rf2o_laser_odometry
    rf2o_laser_odometry_launch = IncludeLaunchDescription(
        PythonLaunchDescriptionSource([os.path.join(
            get_package_share_directory('rf2o_laser_odometry'), 'launch'),
            '/rf2o_laser_odometry.launch.py'])
    )

    odom_to_base_footprint_node = Node(
        package='yahboomcar_bringup',
        executable='base_node',
        name='odom_to_base_footprint',
    )

    cmd_vel_scaler_node = Node(
        package='yahboomcar_bringup',
        executable='cmdvel2b1',
        name='cmd_vel_scaler',
        parameters=[{'mode': LaunchConfiguration('mode')}]
    )

    return LaunchDescription([
        imu_filter_node,
        base_link_to_imu_tf_node,
        base_link_to_laser_tf_node,
        description_launch,
        DeclareLaunchArgument(
            'mode',
            default_value='default',
            description='mode:=default or appslam,nav,slam'
        ),
    ],

```

```

    cmd_vel_scaler_node,
    #rf2o_laser_odometry_launch,
    DeclareLaunchArgument(
        'ekf_params_file',

    default_value=os.path.join(get_package_share_directory("robot_localization"),
    'params', 'ekf3.yaml'),
    ),
    ekf_node,
]

```

The launch file starts the following nodes:

- imu\_filter\_node: filter imu data node, mainly filter part of imu data;
- ekf\_node: fusion node, mainly fuse odom data and filtered imu data.
- base\_link\_to\_imu\_tf\_node: publish a static change, mainly publish the pose transformation of the imu module and the car.
- base\_link\_to\_laser\_tf\_node: publish a static change, mainly publish the pose transformation of the radar module and the car.
- description\_launch: load URDF model.
- cmd\_vel\_scaler\_node: cmd\_vel conversion node, mainly convert the car speed cmd\_vel into cmd\_vel\_bl topic.